

Doxygen Book

Generated by Doxygen 1.8.17

1 nntreamer	1
1.1 Introduction	1
1.2 Program Name	1
1.3 Input/output data	1
1.4 Code information	1
2 Todo List	3
3 Bug List	7
4 Namespace Index	13
4.1 Namespace List	13
5 Class Index	15
5.1 Class List	15
6 File Index	21
6.1 File List	21
7 Namespace Documentation	27
7.1 nntreamer Namespace Reference	27
8 Class Documentation	29
8.1 <code>_FilterTransformData</code> Struct Reference	29
8.1.1 Detailed Description	30
8.1.2 Member Data Documentation	30
8.1.2.1 <code>allocate_in_invoke</code>	30
8.1.2.2 <code>info</code>	30
8.1.2.3 <code>is_flexible</code>	30
8.1.2.4 <code>mem</code>	30
8.1.2.5 <code>meta</code>	30
8.1.2.6 <code>num_tensors</code>	31
8.1.2.7 <code>tensors</code>	31
8.2 <code>_GstDataRepoSink</code> Struct Reference	31
8.2.1 Detailed Description	31
8.2.2 Member Data Documentation	32
8.2.2.1 <code>cumulative_tensors</code>	32
8.2.2.2 <code>data_type</code>	32
8.2.2.3 <code>element</code>	32
8.2.2.4 <code>fd</code>	32
8.2.2.5 <code>fd_offset</code>	32
8.2.2.6 <code>filename</code>	33
8.2.2.7 <code>fixed_caps</code>	33
8.2.2.8 <code>is_static_tensors</code>	33
8.2.2.9 <code>json_filename</code>	33

8.2.2.10 json_object	33
8.2.2.11 sample_offset_array	34
8.2.2.12 sample_size	34
8.2.2.13 tensor_count_array	34
8.2.2.14 tensor_size_array	34
8.2.2.15 total_samples	34
8.3 _GstDataRepoSinkClass Struct Reference	35
8.3.1 Detailed Description	35
8.3.2 Member Data Documentation	35
8.3.2.1 parent_class	35
8.4 _GstDataRepoSrc Struct Reference	35
8.4.1 Detailed Description	37
8.4.2 Member Data Documentation	37
8.4.2.1 array_index	37
8.4.2.2 caps	37
8.4.2.3 config	38
8.4.2.4 current_sample_index	38
8.4.2.5 data_type	38
8.4.2.6 epochs	38
8.4.2.7 fd	38
8.4.2.8 fd_offset	39
8.4.2.9 file_size	39
8.4.2.10 filename	39
8.4.2.11 first_epoch_is_done	39
8.4.2.12 is_shuffle	39
8.4.2.13 is_start	40
8.4.2.14 json_filename	40
8.4.2.15 last_offset	40
8.4.2.16 n_frame	40
8.4.2.17 need_changed_caps	40
8.4.2.18 num_samples	41
8.4.2.19 parent	41
8.4.2.20 parser	41
8.4.2.21 rate_d	41
8.4.2.22 rate_n	41
8.4.2.23 read_position	42
8.4.2.24 running_time	42
8.4.2.25 sample_offset_array	42
8.4.2.26 sample_offset_array_len	42
8.4.2.27 sample_size	42
8.4.2.28 shuffled_index_array	43
8.4.2.29 src_pad	43

8.4.2.30	start_offset	43
8.4.2.31	start_sample_index	43
8.4.2.32	stop_sample_index	43
8.4.2.33	successful_read	44
8.4.2.34	tensor_count_array	44
8.4.2.35	tensor_count_array_len	44
8.4.2.36	tensor_size_array	44
8.4.2.37	tensor_size_array_len	44
8.4.2.38	tensors_offset	45
8.4.2.39	tensors_seq	45
8.4.2.40	tensors_seq_cnt	45
8.4.2.41	tensors_seq_str	45
8.4.2.42	tensors_size	45
8.4.2.43	total_samples	46
8.5	_GstDataRepoSrcClass Struct Reference	46
8.5.1	Detailed Description	46
8.5.2	Member Data Documentation	46
8.5.2.1	parent_class	46
8.6	_GstEdgeSink Struct Reference	47
8.6.1	Detailed Description	47
8.6.2	Member Data Documentation	47
8.6.2.1	cond	47
8.6.2.2	connect_type	47
8.6.2.3	connection_timeout	48
8.6.2.4	custom_lib	48
8.6.2.5	dest_host	48
8.6.2.6	dest_port	48
8.6.2.7	edge_h	48
8.6.2.8	element	48
8.6.2.9	host	49
8.6.2.10	is_connected	49
8.6.2.11	lock	49
8.6.2.12	port	49
8.6.2.13	topic	49
8.6.2.14	wait_connection	49
8.7	_GstEdgeSinkClass Struct Reference	50
8.7.1	Detailed Description	50
8.7.2	Member Data Documentation	50
8.7.2.1	parent_class	50
8.8	_GstEdgeSrc Struct Reference	50
8.8.1	Detailed Description	51
8.8.2	Member Data Documentation	51

8.8.2.1 connect_type	51
8.8.2.2 custom_lib	51
8.8.2.3 dest_host	51
8.8.2.4 dest_port	51
8.8.2.5 edge_h	51
8.8.2.6 element	52
8.8.2.7 msg_queue	52
8.8.2.8 playing	52
8.8.2.9 topic	52
8.9 _GstEdgeSrcClass Struct Reference	52
8.9.1 Detailed Description	53
8.9.2 Member Data Documentation	53
8.9.2.1 parent_class	53
8.10 _GstJoin Struct Reference	53
8.10.1 Detailed Description	53
8.10.2 Member Data Documentation	53
8.10.2.1 active_sinkpad	54
8.10.2.2 cond	54
8.10.2.3 element	54
8.10.2.4 have_group_id	54
8.10.2.5 lock	54
8.10.2.6 n_pads	54
8.10.2.7 padcount	55
8.10.2.8 srcpad	55
8.11 _GstJoinClass Struct Reference	55
8.11.1 Detailed Description	55
8.11.2 Member Data Documentation	55
8.11.2.1 parent_class	55
8.12 _GstJoinPad Struct Reference	56
8.12.1 Detailed Description	56
8.12.2 Member Data Documentation	56
8.12.2.1 group_id	56
8.12.2.2 parent	56
8.12.2.3 segment	56
8.12.2.4 segment_seqnum	57
8.13 _GstJoinPadClass Struct Reference	57
8.13.1 Detailed Description	57
8.13.2 Member Data Documentation	57
8.13.2.1 parent	57
8.14 _GstMQTTMessageHdr Struct Reference	57
8.14.1 Detailed Description	58
8.14.2 Member Data Documentation	58

8.14.2.1 "@6	58
8.14.2.2 _reserved_hdr	58
8.14.2.3 base_time_epoch	58
8.14.2.4 dts	59
8.14.2.5 duration	59
8.14.2.6 gst_caps_str	59
8.14.2.7 num_mems	59
8.14.2.8 pts	59
8.14.2.9 sent_time_epoch	59
8.14.2.10 size_mems	60
8.15 _GstMqttSink Struct Reference	60
8.15.1 Detailed Description	61
8.15.2 Member Data Documentation	61
8.15.2.1 base_time_epoch	61
8.15.2.2 debug	61
8.15.2.3 err	61
8.15.2.4 get_epoch_func	62
8.15.2.5 gquark_err_tag	62
8.15.2.6 in_caps	62
8.15.2.7 is_connected	62
8.15.2.8 max_msg_buf_size	62
8.15.2.9 mqtt_client_handle	62
8.15.2.10 mqtt_client_id	63
8.15.2.11 mqtt_conn_opts	63
8.15.2.12 mqtt_host_address	63
8.15.2.13 mqtt_host_port	63
8.15.2.14 mqtt_msg_buf	63
8.15.2.15 mqtt_msg_buf_size	63
8.15.2.16 mqtt_msg_hdr	64
8.15.2.17 mqtt_ntp_hnames	64
8.15.2.18 mqtt_ntp_num_srvs	64
8.15.2.19 mqtt_ntp_ports	64
8.15.2.20 mqtt_ntp_srvs	64
8.15.2.21 mqtt_ntp_sync	64
8.15.2.22 mqtt_pub_wait_timeout	65
8.15.2.23 mqtt_qos	65
8.15.2.24 mqtt_respn_opts	65
8.15.2.25 mqtt_sink_gcond	65
8.15.2.26 mqtt_sink_mutex	65
8.15.2.27 mqtt_sink_state	65
8.15.2.28 mqtt_topic	66
8.15.2.29 num_buffers	66

8.15.2.30 parent	66
8.16 _GstMqttSinkClass Struct Reference	66
8.16.1 Detailed Description	66
8.16.2 Member Data Documentation	67
8.16.2.1 parent_class	67
8.17 _GstMqttSrc Struct Reference	67
8.17.1 Detailed Description	68
8.17.2 Member Data Documentation	68
8.17.2.1 aqueue	68
8.17.2.2 base_time_epoch	68
8.17.2.3 caps	68
8.17.2.4 debug	68
8.17.2.5 err	69
8.17.2.6 gquark_err_tag	69
8.17.2.7 is_connected	69
8.17.2.8 is_live	69
8.17.2.9 is_subscribed	69
8.17.2.10 latency	69
8.17.2.11 mqtt_client_handle	70
8.17.2.12 mqtt_client_id	70
8.17.2.13 mqtt_conn_opts	70
8.17.2.14 mqtt_host_address	70
8.17.2.15 mqtt_host_port	70
8.17.2.16 mqtt_qos	70
8.17.2.17 mqtt_respn_opts	71
8.17.2.18 mqtt_src_gcond	71
8.17.2.19 mqtt_src_mutex	71
8.17.2.20 mqtt_sub_timeout	71
8.17.2.21 mqtt_topic	71
8.17.2.22 num_dumped	71
8.17.2.23 parent	72
8.18 _GstMqttSrcClass Struct Reference	72
8.18.1 Detailed Description	72
8.18.2 Member Data Documentation	72
8.18.2.1 parent_class	72
8.19 _GstTensorAggregator Struct Reference	73
8.19.1 Detailed Description	74
8.19.2 Member Data Documentation	74
8.19.2.1 adapter_table	74
8.19.2.2 concat	74
8.19.2.3 element	74
8.19.2.4 frames_dim	74

8.19.2.5 frames_flush	75
8.19.2.6 frames_in	75
8.19.2.7 frames_out	75
8.19.2.8 in_config	75
8.19.2.9 out_config	75
8.19.2.10 silent	76
8.19.2.11 sinkpad	76
8.19.2.12 srcpad	76
8.19.2.13 tensor_configured	76
8.20 _GstTensorAggregatorClass Struct Reference	76
8.20.1 Detailed Description	77
8.20.2 Member Data Documentation	77
8.20.2.1 parent_class	77
8.21 _GstTensorConverter Struct Reference	77
8.21.1 Detailed Description	78
8.21.2 Member Data Documentation	78
8.21.2.1 adapter_table	78
8.21.2.2 custom	79
8.21.2.3 do_not_append_header	79
8.21.2.4 element	79
8.21.2.5 ext_fw	79
8.21.2.6 externalConverter	79
8.21.2.7 frame_size	80
8.21.2.8 frames_per_tensor	80
8.21.2.9 have_segment	80
8.21.2.10 in_media_type	80
8.21.2.11 mode	80
8.21.2.12 mode_option	81
8.21.2.13 need_segment	81
8.21.2.14 old_timestamp	81
8.21.2.15 priv_data	81
8.21.2.16 remove_padding	81
8.21.2.17 segment	82
8.21.2.18 set_timestamp	82
8.21.2.19 silent	82
8.21.2.20 sinkpad	82
8.21.2.21 srcpad	82
8.21.2.22 tensors_config	83
8.21.2.23 tensors_configured	83
8.21.2.24 tensors_info	83
8.22 _GstTensorConverterClass Struct Reference	83
8.22.1 Detailed Description	83

8.22.2 Member Data Documentation	84
8.22.2.1 parent_class	84
8.23 _GstTensorCrop Struct Reference	84
8.23.1 Detailed Description	84
8.23.2 Member Data Documentation	84
8.23.2.1 collect	85
8.23.2.2 element	85
8.23.2.3 lateness	85
8.23.2.4 send_stream_start	85
8.23.2.5 silent	85
8.23.2.6 sinkpad_info	86
8.23.2.7 sinkpad_raw	86
8.23.2.8 srcpad	86
8.24 _GstTensorCropClass Struct Reference	86
8.24.1 Detailed Description	86
8.24.2 Member Data Documentation	87
8.24.2.1 parent_class	87
8.25 _GstTensorDebug Struct Reference	87
8.25.1 Detailed Description	87
8.25.2 Member Data Documentation	87
8.25.2.1 cap_mode	88
8.25.2.2 element	88
8.25.2.3 meta_mode	88
8.25.2.4 output_mode	88
8.25.2.5 silent	88
8.25.2.6 sinkpad	88
8.25.2.7 srcpad	89
8.26 _GstTensorDebugClass Struct Reference	89
8.26.1 Detailed Description	89
8.26.2 Member Data Documentation	89
8.26.2.1 parent_class	89
8.27 _GstTensorDecoder Struct Reference	90
8.27.1 Detailed Description	90
8.27.2 Member Data Documentation	90
8.27.2.1 config_path	91
8.27.2.2 configured	91
8.27.2.3 custom	91
8.27.2.4 decoder	91
8.27.2.5 element	91
8.27.2.6 is_custom	92
8.27.2.7 negotiated	92
8.27.2.8 option	92

8.27.2.9 plugin_data	92
8.27.2.10 silent	92
8.27.2.11 tensor_config	93
8.28 _GstTensorDecoderClass Struct Reference	93
8.28.1 Detailed Description	93
8.28.2 Member Data Documentation	93
8.28.2.1 parent_class	93
8.29 _GstTensorDecoderDef Struct Reference	94
8.29.1 Detailed Description	94
8.29.2 Member Data Documentation	94
8.29.2.1 decode	94
8.29.2.2 exit	95
8.29.2.3 getOutCaps	95
8.29.2.4 getTransformSize	95
8.29.2.5 init	96
8.29.2.6 modename	96
8.29.2.7 setOption	97
8.30 _GstTensorDemux Struct Reference	97
8.30.1 Detailed Description	98
8.30.2 Member Data Documentation	98
8.30.2.1 element	98
8.30.2.2 group_id	98
8.30.2.3 have_group_id	98
8.30.2.4 num_srcpads	99
8.30.2.5 silent	99
8.30.2.6 sinkpad	99
8.30.2.7 srcpads	99
8.30.2.8 tensorpick	99
8.30.2.9 tensors_config	99
8.31 _GstTensorDemuxClass Struct Reference	100
8.31.1 Detailed Description	100
8.31.2 Member Data Documentation	100
8.31.2.1 parent_class	100
8.32 _GstTensorFilter Struct Reference	100
8.32.1 Detailed Description	101
8.32.2 Member Data Documentation	101
8.32.2.1 element	101
8.32.2.2 prev_ts	101
8.32.2.3 priv	101
8.32.2.4 throttling_accum	102
8.32.2.5 throttling_delay	102
8.33 _GstTensorFilterClass Struct Reference	102

8.33.1 Detailed Description	102
8.33.2 Member Data Documentation	102
8.33.2.1 parent_class	103
8.34 _GstTensorFilterCombination Struct Reference	103
8.34.1 Detailed Description	103
8.34.2 Member Data Documentation	103
8.34.2.1 in_combi	103
8.34.2.2 in_combi_defined	104
8.34.2.3 out_combi_i	104
8.34.2.4 out_combi_i_defined	104
8.34.2.5 out_combi_o	104
8.34.2.6 out_combi_o_defined	104
8.35 _GstTensorFilterFramework Struct Reference	105
8.35.1 Detailed Description	106
8.35.2 Member Data Documentation	107
8.35.2.1 "@53	107
8.35.2.2 allocate_in_invoke	107
8.35.2.3 allocateInInvoke	107
8.35.2.4 allow_in_place	107
8.35.2.5 checkAvailability	108
8.35.2.6 close	108
8.35.2.7 destroyNotify	108
8.35.2.8 eventHandler	109
8.35.2.9 getFrameworkInfo	110
8.35.2.10 getInputDimension	110
8.35.2.11 getModelInfo	111
8.35.2.12 getOutputDimension	111
8.35.2.13 handleEvent	112
8.35.2.14 invoke	112
8.35.2.15 invoke_NN	113
8.35.2.16 name	113
8.35.2.17 open	113
8.35.2.18 reloadModel	114
8.35.2.19 run_without_model	114
8.35.2.20 setInputDimension	115
8.35.2.21 statistics	115
8.35.2.22 subplugin_data	115
8.35.2.23 verify_model_path	116
8.35.2.24 version	116
8.36 _GstTensorFilterFrameworkEventData Struct Reference	116
8.36.1 Detailed Description	117
8.36.2 Member Data Documentation	117

8.36.2.1 "@39	117
8.36.2.2 custom	117
8.36.2.3 custom_properties	118
8.36.2.4 data	118
8.36.2.5 hw	118
8.36.2.6 hw_list	118
8.36.2.7 info	118
8.36.2.8 layout	119
8.36.2.9 model_files	119
8.36.2.10 num_hw	119
8.36.2.11 num_models	119
8.37 _GstTensorFilterFrameworkInfo Struct Reference	120
8.37.1 Detailed Description	120
8.37.2 Member Data Documentation	120
8.37.2.1 accl_auto	121
8.37.2.2 accl_default	121
8.37.2.3 allocate_in_invoke	121
8.37.2.4 allow_in_place	121
8.37.2.5 hw_list	121
8.37.2.6 name	122
8.37.2.7 num_hw	122
8.37.2.8 run_without_model	122
8.37.2.9 statistics	122
8.37.2.10 verify_model_path	122
8.38 _GstTensorFilterFrameworkStatistics Struct Reference	123
8.38.1 Detailed Description	123
8.38.2 Member Data Documentation	123
8.38.2.1 total_invoke_latency	123
8.38.2.2 total_invoke_num	123
8.38.2.3 total_overhead_latency	124
8.39 _GstTensorFilterPrivate Struct Reference	124
8.39.1 Detailed Description	125
8.39.2 Member Data Documentation	125
8.39.2.1 combi	125
8.39.2.2 config_path	125
8.39.2.3 configured	125
8.39.2.4 fw	125
8.39.2.5 in_config	126
8.39.2.6 info	126
8.39.2.7 is_updatable	126
8.39.2.8 latency_mode	126
8.39.2.9 latency_reported	126

8.39.2.10 latency_reporting	127
8.39.2.11 out_config	127
8.39.2.12 privateData	127
8.39.2.13 prop	127
8.39.2.14 silent	127
8.39.2.15 stat	128
8.39.2.16 throughput_mode	128
8.39.2.17 watchdog_h	128
8.40 _GstTensorFilterProperties Struct Reference	128
8.40.1 Detailed Description	129
8.40.2 Member Data Documentation	129
8.40.2.1 accl_str	129
8.40.2.2 custom_properties	130
8.40.2.3 fw_opened	130
8.40.2.4 fwname	130
8.40.2.5 hw_list	130
8.40.2.6 input_configured	130
8.40.2.7 input_layout	131
8.40.2.8 input_meta	131
8.40.2.9 input_ranks	131
8.40.2.10 invoke_dynamic	131
8.40.2.11 latency	131
8.40.2.12 model_files	132
8.40.2.13 num_hw	132
8.40.2.14 num_models	132
8.40.2.15 output_configured	132
8.40.2.16 output_layout	132
8.40.2.17 output_meta	133
8.40.2.18 output_ranks	133
8.40.2.19 shared_tensor_filter_key	133
8.40.2.20 suspend	133
8.40.2.21 throughput	133
8.41 _GstTensorFilterStatistics Struct Reference	134
8.41.1 Detailed Description	134
8.41.2 Member Data Documentation	134
8.41.2.1 latency_ignore_count	134
8.41.2.2 latest_invoke_time	134
8.41.2.3 old_total_invoke_latency	135
8.41.2.4 old_total_invoke_num	135
8.41.2.5 recent_latencies	135
8.41.2.6 total_invoke_latency	135
8.41.2.7 total_invoke_num	135

8.42 <code>_GstTensorIf</code> Struct Reference	136
8.42.1 Detailed Description	137
8.42.2 Member Data Documentation	137
8.42.2.1 <code>act_else</code>	137
8.42.2.2 <code>act_then</code>	137
8.42.2.3 <code>custom</code>	137
8.42.2.4 <code>custom_configured</code>	137
8.42.2.5 <code>cv</code>	137
8.42.2.6 <code>cv_option</code>	138
8.42.2.7 <code>element</code>	138
8.42.2.8 <code>else_option</code>	138
8.42.2.9 <code>in_config</code>	138
8.42.2.10 <code>lock</code>	138
8.42.2.11 <code>num_srcpads</code>	139
8.42.2.12 <code>op</code>	139
8.42.2.13 <code>out_config</code>	139
8.42.2.14 <code>silent</code>	139
8.42.2.15 <code>sinkpad</code>	139
8.42.2.16 <code>srcpads</code>	139
8.42.2.17 <code>sv</code>	140
8.42.2.18 <code>then_option</code>	140
8.43 <code>_GstTensorIfClass</code> Struct Reference	140
8.43.1 Detailed Description	140
8.43.2 Member Data Documentation	140
8.43.2.1 <code>parent_class</code>	140
8.44 <code>_GstTensorMerge</code> Struct Reference	141
8.44.1 Detailed Description	141
8.44.2 Member Data Documentation	142
8.44.2.1 <code>"@22</code>	142
8.44.2.2 <code>collect</code>	142
8.44.2.3 <code>current_time</code>	142
8.44.2.4 <code>data_linear</code>	142
8.44.2.5 <code>element</code>	142
8.44.2.6 <code>loaded</code>	142
8.44.2.7 <code>mode</code>	143
8.44.2.8 <code>need_segment</code>	143
8.44.2.9 <code>need_set_time</code>	143
8.44.2.10 <code>need_stream_start</code>	143
8.44.2.11 <code>negotiated</code>	143
8.44.2.12 <code>option</code>	143
8.44.2.13 <code>send_stream_start</code>	144
8.44.2.14 <code>silent</code>	144

8.44.2.15 srcpad	144
8.44.2.16 sync	144
8.44.2.17 tensors_config	144
8.45 _GstTensorMergeClass Struct Reference	144
8.45.1 Detailed Description	145
8.45.2 Member Data Documentation	145
8.45.2.1 parent_class	145
8.46 _GstTensorMux Struct Reference	145
8.46.1 Detailed Description	146
8.46.2 Member Data Documentation	146
8.46.2.1 collect	146
8.46.2.2 current_time	146
8.46.2.3 element	146
8.46.2.4 need_segment	147
8.46.2.5 need_set_time	147
8.46.2.6 need_stream_start	147
8.46.2.7 negotiated	147
8.46.2.8 send_stream_start	147
8.46.2.9 silent	147
8.46.2.10 srcpad	148
8.46.2.11 sync	148
8.46.2.12 tensors_config	148
8.47 _GstTensorMuxClass Struct Reference	148
8.47.1 Detailed Description	148
8.47.2 Member Data Documentation	149
8.47.2.1 parent_class	149
8.48 _GstTensorQueryClient Struct Reference	149
8.48.1 Detailed Description	150
8.48.2 Member Data Documentation	150
8.48.2.1 config	150
8.48.2.2 connect_type	150
8.48.2.3 dest_host	151
8.48.2.4 dest_port	151
8.48.2.5 edge_h	151
8.48.2.6 element	151
8.48.2.7 host	151
8.48.2.8 in_caps_str	151
8.48.2.9 is_tensor	152
8.48.2.10 max_request	152
8.48.2.11 msg_queue	152
8.48.2.12 port	152
8.48.2.13 requested_num	152

8.48.2.14 silent	152
8.48.2.15 sinkpad	153
8.48.2.16 srcpad	153
8.48.2.17 timeout	153
8.48.2.18 topic	153
8.49 _GstTensorQueryClientClass Struct Reference	153
8.49.1 Detailed Description	154
8.49.2 Member Data Documentation	154
8.49.2.1 parent_class	154
8.50 _GstTensorQueryServerSink Struct Reference	154
8.50.1 Detailed Description	154
8.50.2 Member Data Documentation	155
8.50.2.1 connect_type	155
8.50.2.2 element	155
8.50.2.3 metaless_frame_count	155
8.50.2.4 metaless_frame_limit	155
8.50.2.5 sink_id	155
8.50.2.6 timeout	156
8.51 _GstTensorQueryServerSinkClass Struct Reference	156
8.51.1 Detailed Description	156
8.51.2 Member Data Documentation	156
8.51.2.1 parent_class	156
8.52 _GstTensorQueryServerSrc Struct Reference	157
8.52.1 Detailed Description	157
8.52.2 Member Data Documentation	157
8.52.2.1 configured	157
8.52.2.2 connect_type	157
8.52.2.3 dest_host	158
8.52.2.4 dest_port	158
8.52.2.5 element	158
8.52.2.6 host	158
8.52.2.7 msg_queue	158
8.52.2.8 playing	158
8.52.2.9 port	159
8.52.2.10 src_id	159
8.52.2.11 timeout	159
8.52.2.12 topic	159
8.53 _GstTensorQueryServerSrcClass Struct Reference	159
8.53.1 Detailed Description	160
8.53.2 Member Data Documentation	160
8.53.2.1 parent_class	160
8.54 _GstTensorRate Struct Reference	160

8.54.1 Detailed Description	161
8.54.2 Member Data Documentation	161
8.54.2.1 base_ts	161
8.54.2.2 drop	161
8.54.2.3 dup	161
8.54.2.4 element	161
8.54.2.5 from_rate_denominator	162
8.54.2.6 from_rate_numerator	162
8.54.2.7 in	162
8.54.2.8 last_ts	162
8.54.2.9 next_ts	162
8.54.2.10 out	163
8.54.2.11 out_frame_count	163
8.54.2.12 prev_ts	163
8.54.2.13 prevbuf	163
8.54.2.14 rate_d	163
8.54.2.15 rate_n	164
8.54.2.16 segment	164
8.54.2.17 sent_qos_on_passthrough	164
8.54.2.18 silent	164
8.54.2.19 throttle	164
8.54.2.20 to_rate_denominator	165
8.54.2.21 to_rate_numerator	165
8.55 _GstTensorRateClass Struct Reference	165
8.55.1 Detailed Description	165
8.55.2 Member Data Documentation	165
8.55.2.1 parent_class	166
8.56 _GstTensorRepoSink Struct Reference	166
8.56.1 Detailed Description	166
8.56.2 Member Data Documentation	166
8.56.2.1 element	166
8.56.2.2 in_caps	167
8.56.2.3 last_render_time	167
8.56.2.4 myid	167
8.56.2.5 o_myid	167
8.56.2.6 set_startid	167
8.56.2.7 signal_rate	167
8.56.2.8 silent	168
8.57 _GstTensorRepoSinkClass Struct Reference	168
8.57.1 Detailed Description	168
8.57.2 Member Data Documentation	168
8.57.2.1 parent_class	168

8.58 <code>_GstTensorRepoSrc</code> Struct Reference	169
8.58.1 Detailed Description	169
8.58.2 Member Data Documentation	170
8.58.2.1 <code>caps</code>	170
8.58.2.2 <code>config</code>	170
8.58.2.3 <code>fps_d</code>	170
8.58.2.4 <code>fps_n</code>	170
8.58.2.5 <code>ini</code>	170
8.58.2.6 <code>myid</code>	171
8.58.2.7 <code>negotiation</code>	171
8.58.2.8 <code>o_myid</code>	171
8.58.2.9 <code>parent</code>	171
8.58.2.10 <code>set_startid</code>	171
8.58.2.11 <code>silent</code>	171
8.59 <code>_GstTensorRepoSrcClass</code> Struct Reference	172
8.59.1 Detailed Description	172
8.59.2 Member Data Documentation	172
8.59.2.1 <code>parent_class</code>	172
8.60 <code>_GstTensorSink</code> Struct Reference	172
8.60.1 Detailed Description	173
8.60.2 Member Data Documentation	173
8.60.2.1 <code>element</code>	173
8.60.2.2 <code>emit_signal</code>	173
8.60.2.3 <code>last_render_time</code>	173
8.60.2.4 <code>mutex</code>	173
8.60.2.5 <code>signal_rate</code>	174
8.60.2.6 <code>silent</code>	174
8.61 <code>_GstTensorSinkClass</code> Struct Reference	174
8.61.1 Detailed Description	174
8.61.2 Member Data Documentation	174
8.61.2.1 <code>eos</code>	175
8.61.2.2 <code>new_data</code>	175
8.61.2.3 <code>parent_class</code>	175
8.61.2.4 <code>stream_start</code>	175
8.62 <code>_GstTensorSparseDec</code> Struct Reference	176
8.62.1 Detailed Description	176
8.62.2 Member Data Documentation	177
8.62.2.1 <code>element</code>	177
8.62.2.2 <code>in_config</code>	177
8.62.2.3 <code>out_config</code>	177
8.62.2.4 <code>silent</code>	177
8.62.2.5 <code>sinkpad</code>	177

8.62.2.6 srcpad	178
8.63 _GstTensorSparseDecClass Struct Reference	178
8.63.1 Detailed Description	178
8.63.2 Member Data Documentation	178
8.63.2.1 parent_class	178
8.64 _GstTensorSparseEnc Struct Reference	179
8.64.1 Detailed Description	179
8.64.2 Member Data Documentation	179
8.64.2.1 element	180
8.64.2.2 in_config	180
8.64.2.3 silent	180
8.64.2.4 sinkpad	180
8.64.2.5 srcpad	180
8.65 _GstTensorSparseEncClass Struct Reference	181
8.65.1 Detailed Description	181
8.65.2 Member Data Documentation	181
8.65.2.1 parent_class	181
8.66 _GstTensorSplit Struct Reference	182
8.66.1 Detailed Description	182
8.66.2 Member Data Documentation	183
8.66.2.1 element	183
8.66.2.2 group_id	183
8.66.2.3 have_group_id	183
8.66.2.4 num_srcpads	183
8.66.2.5 num_tensors	183
8.66.2.6 silent	184
8.66.2.7 sink_tensor_conf	184
8.66.2.8 sinkpad	184
8.66.2.9 srcpads	184
8.66.2.10 tensorpick	184
8.66.2.11 tensorseg	184
8.67 _GstTensorSplitClass Struct Reference	185
8.67.1 Detailed Description	185
8.67.2 Member Data Documentation	185
8.67.2.1 parent_class	185
8.68 _GstTensorSrcIIO Struct Reference	185
8.68.1 Detailed Description	187
8.68.2 Member Data Documentation	187
8.68.2.1 base_dir	187
8.68.2.2 buffer_capacity	187
8.68.2.3 buffer_data_fp	187
8.68.2.4 channels	187

8.68.2.5 channels_enabled	188
8.68.2.6 configured	188
8.68.2.7 custom_channel_table	188
8.68.2.8 default_buffer_capacity	188
8.68.2.9 default_sampling_frequency	188
8.68.2.10 default_trigger	189
8.68.2.11 dev_dir	189
8.68.2.12 device	189
8.68.2.13 element	189
8.68.2.14 is_tensor	189
8.68.2.15 merge_channels_data	190
8.68.2.16 mode	190
8.68.2.17 num_channels_enabled	190
8.68.2.18 poll_timeout	190
8.68.2.19 sampling_frequency	190
8.68.2.20 scan_size	191
8.68.2.21 silent	191
8.68.2.22 tensors_config	191
8.68.2.23 trigger	191
8.69 _GstTensorSrcIOChannelProperties Struct Reference	191
8.69.1 Detailed Description	192
8.69.2 Member Data Documentation	192
8.69.2.1 base_dir	192
8.69.2.2 base_file	192
8.69.2.3 big_endian	193
8.69.2.4 enabled	193
8.69.2.5 generic_name	193
8.69.2.6 index	193
8.69.2.7 is_signed	193
8.69.2.8 location	194
8.69.2.9 mask	194
8.69.2.10 name	194
8.69.2.11 offset	194
8.69.2.12 pre_enabled	194
8.69.2.13 scale	195
8.69.2.14 shift	195
8.69.2.15 storage_bits	195
8.69.2.16 storage_bytes	195
8.69.2.17 used_bits	195
8.70 _GstTensorSrcIOClass Struct Reference	196
8.70.1 Detailed Description	196
8.70.2 Member Data Documentation	196

8.70.2.1 parent_class	196
8.71 _GstTensorSrcIIODeviceProperties Struct Reference	196
8.71.1 Detailed Description	197
8.71.2 Member Data Documentation	197
8.71.2.1 base_dir	197
8.71.2.2 id	197
8.71.2.3 name	197
8.72 _GstTensorTrainer Struct Reference	198
8.72.1 Detailed Description	198
8.72.2 Member Data Documentation	199
8.72.2.1 cur_epoch_data_cnt	199
8.72.2.2 dummy_data_thread	199
8.72.2.3 element	199
8.72.2.4 epoch_completion_cond	199
8.72.2.5 epoch_completion_lock	199
8.72.2.6 fw	200
8.72.2.7 fw_created	200
8.72.2.8 fw_name	200
8.72.2.9 in_config	200
8.72.2.10 input_tensors	200
8.72.2.11 is_epoch_complete	200
8.72.2.12 is_training_complete	201
8.72.2.13 notifier	201
8.72.2.14 out_config	201
8.72.2.15 output_meta	201
8.72.2.16 privateData	201
8.72.2.17 prop	201
8.72.2.18 required_sample	202
8.72.2.19 sinkpad	202
8.72.2.20 srcpad	202
8.72.2.21 training_completion_cond	202
8.72.2.22 training_completion_lock	202
8.73 _GstTensorTrainerClass Struct Reference	202
8.73.1 Detailed Description	203
8.73.2 Member Data Documentation	203
8.73.2.1 parent_class	203
8.74 _GstTensorTrainerEventNotifier Struct Reference	203
8.74.1 Detailed Description	203
8.74.2 Member Data Documentation	204
8.74.2.1 notifier	204
8.74.2.2 version	204
8.75 _GstTensorTrainerFramework Struct Reference	204

8.75.1 Detailed Description	205
8.75.2 Member Data Documentation	205
8.75.2.1 create	205
8.75.2.2 destroy	205
8.75.2.3 getFrameworkInfo	206
8.75.2.4 getStatus	206
8.75.2.5 push_data	207
8.75.2.6 start	207
8.75.2.7 stop	208
8.75.2.8 version	209
8.76 _GstTensorTrainerFrameworkInfo Struct Reference	209
8.76.1 Detailed Description	209
8.76.2 Member Data Documentation	209
8.76.2.1 name	210
8.77 _GstTensorTrainerProperties Struct Reference	210
8.77.1 Detailed Description	211
8.77.2 Member Data Documentation	211
8.77.2.1 epoch_count	211
8.77.2.2 input_meta	211
8.77.2.3 model_config	211
8.77.2.4 model_load_path	211
8.77.2.5 model_save_path	212
8.77.2.6 num_epochs	212
8.77.2.7 num_inputs	212
8.77.2.8 num_labels	212
8.77.2.9 num_training_samples	212
8.77.2.10 num_validation_samples	213
8.77.2.11 training_accuracy	213
8.77.2.12 training_loss	213
8.77.2.13 validation_accuracy	213
8.77.2.14 validation_loss	213
8.78 _GstTensorTransform Struct Reference	214
8.78.1 Detailed Description	214
8.78.2 Member Data Documentation	215
8.78.2.1 "@37	215
8.78.2.2 acceleration	215
8.78.2.3 apply	215
8.78.2.4 data_arithmetic	215
8.78.2.5 data_clamp	215
8.78.2.6 data_dimchg	216
8.78.2.7 data_padding	216
8.78.2.8 data_stand	216

8.78.2.9 data_transpose	216
8.78.2.10 data_typecast	216
8.78.2.11 element	217
8.78.2.12 in_config	217
8.78.2.13 loaded	217
8.78.2.14 mode	217
8.78.2.15 operators	217
8.78.2.16 option	218
8.78.2.17 out_config	218
8.78.2.18 silent	218
8.79 _GstTensorTransformClass Struct Reference	218
8.79.1 Detailed Description	218
8.79.2 Member Data Documentation	219
8.79.2.1 parent_class	219
8.80 _GstTensorFilterSingle Struct Reference	219
8.80.1 Detailed Description	219
8.80.2 Member Data Documentation	219
8.80.2.1 element	219
8.80.2.2 priv	220
8.81 _GstTensorFilterSingleClass Struct Reference	220
8.81.1 Detailed Description	220
8.81.2 Member Data Documentation	220
8.81.2.1 allocate_in_invoke	220
8.81.2.2 destroy_notify	221
8.81.2.3 input_configured	221
8.81.2.4 invoke	221
8.81.2.5 output_configured	221
8.81.2.6 parent	221
8.81.2.7 set_input_info	222
8.81.2.8 start	222
8.81.2.9 stop	222
8.82 _GstTensorFilterSinglePrivate Struct Reference	222
8.82.1 Detailed Description	223
8.82.2 Member Data Documentation	223
8.82.2.1 allocate_in_invoke	223
8.82.2.2 filter_priv	223
8.83 _internal_data Struct Reference	224
8.83.1 Detailed Description	224
8.83.2 Member Data Documentation	225
8.83.2.1 customFW_private_data	225
8.83.2.2 methods	225
8.83.2.3 module	225

8.84 _NNStreamer_custom_class Struct Reference	225
8.84.1 Detailed Description	226
8.84.2 Member Data Documentation	227
8.84.2.1 allocate_invoke	227
8.84.2.2 destroy_notify	227
8.84.2.3 exitfunc	227
8.84.2.4 getInputDim	227
8.84.2.5 getOutputDim	228
8.84.2.6 initfunc	228
8.84.2.7 invoke	228
8.84.2.8 setInputDim	228
8.85 _NNStreamerExternalConverter Struct Reference	228
8.85.1 Detailed Description	229
8.85.2 Member Data Documentation	229
8.85.2.1 close	229
8.85.2.2 convert	229
8.85.2.3 get_out_config	230
8.85.2.4 name	230
8.85.2.5 open	230
8.85.2.6 query_caps	231
8.86 _NnstWatchdog Struct Reference	231
8.86.1 Detailed Description	231
8.86.2 Member Data Documentation	232
8.86.2.1 cond	232
8.86.2.2 context	232
8.86.2.3 lock	232
8.86.2.4 loop	232
8.86.2.5 source	232
8.86.2.6 thread	233
8.87 _ntp_packet_t Struct Reference	233
8.87.1 Detailed Description	234
8.87.2 Member Data Documentation	234
8.87.2.1 li_vn_mode	234
8.87.2.2 org_ts	234
8.87.2.3 poll	234
8.87.2.4 precision	235
8.87.2.5 recv_ts	235
8.87.2.6 ref_id	235
8.87.2.7 ref_ts	235
8.87.2.8 root_delay	235
8.87.2.9 root_dispersion	235
8.87.2.10 stratum	236

8.87.2.11 xmit_ts	236
8.88 _ntp_timestamp_t Struct Reference	236
8.88.1 Detailed Description	236
8.88.2 Member Data Documentation	236
8.88.2.1 frac	236
8.88.2.2 sec	237
8.89 _tensor_merge_linear Struct Reference	237
8.89.1 Detailed Description	237
8.89.2 Member Data Documentation	237
8.89.2.1 direction	237
8.90 _tensor_sync_basepad_data Struct Reference	237
8.90.1 Detailed Description	238
8.90.2 Member Data Documentation	238
8.90.2.1 duration	238
8.90.2.2 sink_id	238
8.91 _tensor_time_sync_data Struct Reference	238
8.91.1 Detailed Description	239
8.91.2 Member Data Documentation	239
8.91.2.1 "@61	239
8.91.2.2 data_basepad	239
8.91.2.3 mode	239
8.91.2.4 option	239
8.92 _tensor_transform_arithmetic Struct Reference	240
8.92.1 Detailed Description	240
8.92.2 Member Data Documentation	240
8.92.2.1 ch_dim	240
8.92.2.2 out_type	240
8.92.2.3 per_channel_arith	240
8.93 _tensor_transform_clamp Struct Reference	241
8.93.1 Detailed Description	241
8.93.2 Member Data Documentation	241
8.93.2.1 max	241
8.93.2.2 min	241
8.94 _tensor_transform_dimchg Struct Reference	241
8.94.1 Detailed Description	242
8.94.2 Member Data Documentation	242
8.94.2.1 from	242
8.94.2.2 to	242
8.95 _tensor_transform_padding Struct Reference	242
8.95.1 Detailed Description	243
8.95.2 Member Data Documentation	243
8.95.2.1 layout	243

8.95.2.2 pad	243
8.96 _tensor_transform_stand Struct Reference	243
8.96.1 Detailed Description	243
8.96.2 Member Data Documentation	244
8.96.2.1 mode	244
8.96.2.2 out_type	244
8.96.2.3 per_channel	244
8.97 _tensor_transform_transpose Struct Reference	244
8.97.1 Detailed Description	244
8.97.2 Member Data Documentation	245
8.97.2.1 trans_order	245
8.98 _tensor_transform_typecast Struct Reference	245
8.98.1 Detailed Description	245
8.98.2 Member Data Documentation	245
8.98.2.1 to	245
8.99 confdata Struct Reference	246
8.99.1 Detailed Description	246
8.99.2 Member Data Documentation	246
8.99.2.1 conf	246
8.99.2.2 conffile	247
8.99.2.3 enable_envvar	247
8.99.2.4 enable_symlink	247
8.99.2.5 extra_conf file	247
8.99.2.6 loaded	247
8.100 converter_custom_cb_s Struct Reference	248
8.100.1 Detailed Description	248
8.100.2 Member Data Documentation	248
8.100.2.1 data	248
8.100.2.2 func	248
8.101 custom_cb_s Struct Reference	248
8.101.1 Detailed Description	249
8.101.2 Member Data Documentation	249
8.101.2.1 data	249
8.101.2.2 func	249
8.101.2.3 name	249
8.102 decoder_custom_cb_s Struct Reference	249
8.102.1 Detailed Description	250
8.102.2 Member Data Documentation	250
8.102.2.1 data	250
8.102.2.2 func	250
8.103 dump_buf Struct Reference	250
8.103.1 Detailed Description	250

8.103.2 Member Data Documentation	250
8.103.2.1 base	251
8.103.2.2 pos	251
8.103.2.3 size	251
8.104 <code>gst_tensor_aggregation_data_s</code> Struct Reference	251
8.104.1 Detailed Description	251
8.104.2 Member Data Documentation	251
8.104.2.1 adapter	252
8.105 <code>GstMetaQuery</code> Struct Reference	252
8.105.1 Detailed Description	252
8.105.2 Member Data Documentation	252
8.105.2.1 <code>client_id</code>	252
8.105.2.2 <code>meta</code>	252
8.106 <code>GstSparseTensorInfo</code> Struct Reference	253
8.106.1 Detailed Description	253
8.106.2 Member Data Documentation	253
8.106.2.1 <code>nnz</code>	253
8.107 <code>GstTensorAllocator</code> Struct Reference	253
8.107.1 Detailed Description	253
8.107.2 Member Data Documentation	254
8.107.2.1 <code>parent</code>	254
8.108 <code>GstTensorAllocatorClass</code> Struct Reference	254
8.108.1 Detailed Description	254
8.108.2 Member Data Documentation	254
8.108.2.1 <code>parent_class</code>	254
8.109 <code>GstTensorCollectPadData</code> Struct Reference	255
8.109.1 Detailed Description	255
8.109.2 Member Data Documentation	255
8.109.2.1 <code>buffer</code>	255
8.109.2.2 <code>collect</code>	255
8.110 <code>GstTensorCropPadData</code> Struct Reference	256
8.110.1 Detailed Description	256
8.110.2 Member Data Documentation	256
8.110.2.1 <code>config</code>	257
8.110.2.2 <code>data</code>	257
8.111 <code>GstTensorExtraInfo</code> Struct Reference	257
8.111.1 Detailed Description	258
8.111.2 Member Data Documentation	258
8.111.2.1 <code>infos</code>	258
8.111.2.2 <code>magic</code>	258
8.111.2.3 <code>num_extra_tensors</code>	258
8.111.2.4 <code>reserved</code>	258

8.111.2.5 version	259
8.112 GstTensorFilterSharedModelRepresentation Struct Reference	259
8.112.1 Detailed Description	259
8.112.2 Member Data Documentation	259
8.112.2.1 referred_list	259
8.112.2.2 shared_interpreter	260
8.113 GstTensorInfo Struct Reference	260
8.113.1 Detailed Description	260
8.113.2 Member Data Documentation	260
8.113.2.1 dimension	260
8.113.2.2 name	261
8.113.2.3 type	261
8.114 GstTensorMemory Struct Reference	261
8.114.1 Detailed Description	261
8.114.2 Member Data Documentation	261
8.114.2.1 data	262
8.114.2.2 size	262
8.115 GstTensorMetaInfo Struct Reference	262
8.115.1 Detailed Description	263
8.115.2 Member Data Documentation	263
8.115.2.1 "@59	263
8.115.2.2 dimension	263
8.115.2.3 format	264
8.115.2.4 magic	264
8.115.2.5 media_type	264
8.115.2.6 sparse_info	264
8.115.2.7 type	264
8.115.2.8 version	264
8.116 GstTensorPad Struct Reference	265
8.116.1 Detailed Description	265
8.116.2 Member Data Documentation	265
8.116.2.1 last_ret	265
8.116.2.2 last_ts	265
8.116.2.3 nth	265
8.116.2.4 pad	266
8.117 GstTensorQueryEdgeInfo Struct Reference	266
8.117.1 Detailed Description	266
8.117.2 Member Data Documentation	266
8.117.2.1 cb	266
8.117.2.2 dest_host	267
8.117.2.3 dest_port	267
8.117.2.4 host	267

8.117.2.5 pdata	267
8.117.2.6 port	267
8.117.2.7 topic	267
8.118 GstTensorQueryServer Struct Reference	268
8.118.1 Detailed Description	268
8.118.2 Member Data Documentation	268
8.118.2.1 cond	268
8.118.2.2 configured	268
8.118.2.3 edge_h	268
8.118.2.4 id	269
8.118.2.5 lock	269
8.119 GstTensorRepo Struct Reference	269
8.119.1 Detailed Description	269
8.119.2 Member Data Documentation	269
8.119.2.1 hash	270
8.119.2.2 initialized	270
8.119.2.3 num_data	270
8.119.2.4 repo_cond	270
8.119.2.5 repo_lock	270
8.120 GstTensorRepoData Struct Reference	270
8.120.1 Detailed Description	271
8.120.2 Member Data Documentation	271
8.120.2.1 buffer	271
8.120.2.2 caps	271
8.120.2.3 cond_pull	271
8.120.2.4 cond_push	272
8.120.2.5 eos	272
8.120.2.6 lock	272
8.120.2.7 pushed	272
8.120.2.8 sink_changed	272
8.120.2.9 sink_id	272
8.120.2.10 src_changed	273
8.120.2.11 src_id	273
8.121 GstTensorsConfig Struct Reference	273
8.121.1 Detailed Description	274
8.121.2 Member Data Documentation	274
8.121.2.1 info	274
8.121.2.2 rate_d	274
8.121.2.3 rate_n	274
8.122 GstTensorsInfo Struct Reference	275
8.122.1 Detailed Description	275
8.122.2 Member Data Documentation	275

8.122.2.1 extra	275
8.122.2.2 format	276
8.122.2.3 info	276
8.122.2.4 num_tensors	276
8.123 parse_accl_args Struct Reference	276
8.123.1 Detailed Description	277
8.123.2 Member Data Documentation	277
8.123.2.1 auto_accl	277
8.123.2.2 def_accl	277
8.123.2.3 in_accl	277
8.123.2.4 sup_accl	278
8.124 runtime_data Struct Reference	278
8.124.1 Detailed Description	279
8.124.2 Member Data Documentation	280
8.124.2.1 model	280
8.125 subplugin_conf Struct Reference	280
8.125.1 Detailed Description	280
8.125.2 Member Data Documentation	280
8.125.2.1 files	280
8.125.2.2 names	280
8.125.2.3 path	281
8.126 subplugin_info_s Struct Reference	281
8.126.1 Detailed Description	281
8.126.2 Member Data Documentation	281
8.126.2.1 names	281
8.126.2.2 paths	281
8.127 tensor_crop_info_s Struct Reference	282
8.127.1 Detailed Description	282
8.127.2 Member Data Documentation	282
8.127.2.1 num	282
8.127.2.2 region	283
8.128 tensor_data_s Struct Reference	283
8.128.1 Detailed Description	283
8.128.2 Member Data Documentation	283
8.128.2.1 data	284
8.128.2.2 type	284
8.129 tensor_element Union Reference	284
8.129.1 Detailed Description	284
8.129.2 Member Data Documentation	284
8.129.2.1 _double	285
8.129.2.2 _float	285
8.129.2.3 _int16_t	285

8.129.2.4	<code>_int32_t</code>	285
8.129.2.5	<code>_int64_t</code>	285
8.129.2.6	<code>_int8_t</code>	285
8.129.2.7	<code>_uint16_t</code>	286
8.129.2.8	<code>_uint32_t</code>	286
8.129.2.9	<code>_uint64_t</code>	286
8.129.2.10	<code>_uint8_t</code>	286
8.130	<code>tensor_if_sv_s</code> Struct Reference	286
8.130.1	Detailed Description	287
8.130.2	Member Data Documentation	287
8.130.2.1	<code>data</code>	287
8.130.2.2	<code>num</code>	287
8.130.2.3	<code>type</code>	287
8.131	<code>tensor_region_s</code> Struct Reference	287
8.131.1	Detailed Description	288
8.131.2	Member Data Documentation	288
8.131.2.1	<code>h</code>	288
8.131.2.2	<code>w</code>	288
8.131.2.3	<code>x</code>	289
8.131.2.4	<code>y</code>	289
8.132	<code>tensor_transform_operator_s</code> Struct Reference	289
8.132.1	Detailed Description	290
8.132.2	Member Data Documentation	290
8.132.2.1	<code>applying_ch</code>	290
8.132.2.2	<code>op</code>	290
8.132.2.3	<code>value</code>	290
8.133	<code>vstr_helper</code> Struct Reference	290
8.133.1	Detailed Description	291
8.133.2	Member Data Documentation	291
8.133.2.1	<code>cursor</code>	291
8.133.2.2	<code>size</code>	291
8.133.2.3	<code>vstr</code>	291
9	File Documentation	293
9.1	<code>datarepo/gstdatarepo.c</code> File Reference	293
9.1.1	Detailed Description	294
9.1.2	Macro Definition Documentation	294
9.1.2.1	<code>PACKAGE</code>	294
9.1.3	Function Documentation	294
9.1.3.1	<code>gst_data_repo_get_data_type_from_caps()</code>	295
9.1.3.2	<code>plugin_init()</code>	295
9.2	<code>datarepo/gstdatarepo.h</code> File Reference	296

9.2.1 Detailed Description	297
9.2.2 Enumeration Type Documentation	297
9.2.2.1 GstDataRepoDataType	297
9.2.3 Function Documentation	297
9.2.3.1 gst_data_repo_get_data_type_from_caps()	298
9.3 datarepo/gstdatareposink.c File Reference	298
9.3.1 Detailed Description	300
9.3.2 Macro Definition Documentation	301
9.3.2.1 _do_init	301
9.3.2.2 AUDIO_CAPS	301
9.3.2.3 GST_CAT_DEFAULT	302
9.3.2.4 gst_data_repo_sink_parent_class	302
9.3.2.5 IMAGE_CAPS	302
9.3.2.6 OCTET_CAPS	302
9.3.2.7 SUPPORTED_AUDIO_FORMAT	302
9.3.2.8 SUPPORTED_VIDEO_FORMAT	303
9.3.2.9 TENSOR_CAPS	303
9.3.2.10 TEXT_CAPS	303
9.3.2.11 VIDEO_CAPS	303
9.3.3 Enumeration Type Documentation	303
9.3.3.1 anonymous enum	303
9.3.4 Function Documentation	304
9.3.4.1 __write_json()	304
9.3.4.2 G_DEFINE_TYPE_WITH_CODE()	304
9.3.4.3 gst_data_repo_sink_change_state()	305
9.3.4.4 gst_data_repo_sink_class_init()	305
9.3.4.5 gst_data_repo_sink_finalize()	306
9.3.4.6 gst_data_repo_sink_get_caps()	307
9.3.4.7 gst_data_repo_sink_get_image_filename()	307
9.3.4.8 gst_data_repo_sink_get_property()	308
9.3.4.9 gst_data_repo_sink_init()	308
9.3.4.10 gst_data_repo_sink_open_file()	309
9.3.4.11 gst_data_repo_sink_query()	309
9.3.4.12 gst_data_repo_sink_render()	310
9.3.4.13 gst_data_repo_sink_set_caps()	310
9.3.4.14 gst_data_repo_sink_set_is_static_tensors()	311
9.3.4.15 gst_data_repo_sink_set_property()	312
9.3.4.16 gst_data_repo_sink_stop()	312
9.3.4.17 gst_data_repo_sink_write_flexible_or_sparse_tensors()	313
9.3.4.18 gst_data_repo_sink_write_json_meta_file()	314
9.3.4.19 gst_data_repo_sink_write_multi_images()	314
9.3.4.20 gst_data_repo_sink_write_others()	315

9.3.4.21 GST_DEBUG_CATEGORY_STATIC()	315
9.3.5 Variable Documentation	315
9.3.5.1 sinktemplate	315
9.4 datarepo/gstdatareposink.h File Reference	316
9.4.1 Macro Definition Documentation	317
9.4.1.1 GST_DATA_REPO_SINK	317
9.4.1.2 GST_DATA_REPO_SINK_CAST	317
9.4.1.3 GST_DATA_REPO_SINK_CLASS	317
9.4.1.4 GST_IS_DATA_REPO_SINK	317
9.4.1.5 GST_IS_DATA_REPO_SINK_CLASS	318
9.4.1.6 GST_TYPE_DATA_REPO_SINK	318
9.4.2 Typedef Documentation	318
9.4.2.1 GstDataRepoSink	318
9.4.2.2 GstDataRepoSinkClass	318
9.4.3 Function Documentation	318
9.4.3.1 gst_data_repo_sink_get_type()	318
9.5 datarepo/gstdatareposrc.c File Reference	319
9.5.1 Detailed Description	321
9.5.2 Macro Definition Documentation	321
9.5.2.1 _do_init	322
9.5.2.2 DEFAULT_EPOCHS	322
9.5.2.3 DEFAULT_INDEX	322
9.5.2.4 DEFAULT_IS_SHUFFLE	322
9.5.2.5 GST_CAT_DEFAULT	322
9.5.2.6 gst_data_repo_src_parent_class	322
9.5.2.7 O_BINARY	323
9.5.2.8 S_ISDIR	323
9.5.2.9 S_ISREG	323
9.5.2.10 S_ISSOCK	323
9.5.2.11 struct_stat	323
9.5.3 Enumeration Type Documentation	323
9.5.3.1 anonymous enum	323
9.5.4 Function Documentation	324
9.5.4.1 G_DEFINE_TYPE_WITH_CODE()	324
9.5.4.2 gst_data_repo_get_caps_by_tensors_sequence()	324
9.5.4.3 gst_data_repo_src_change_state()	325
9.5.4.4 gst_data_repo_src_class_init()	326
9.5.4.5 gst_data_repo_src_create()	326
9.5.4.6 gst_data_repo_src_epoch_is_done()	327
9.5.4.7 gst_data_repo_src_finalize()	328
9.5.4.8 gst_data_repo_src_get_caps()	328
9.5.4.9 gst_data_repo_src_get_file_offset()	329

9.5.4.10	<code>gst_data_repo_src_get_image_filename()</code>	329
9.5.4.11	<code>gst_data_repo_src_get_num_tensors()</code>	330
9.5.4.12	<code>gst_data_repo_src_get_property()</code>	330
9.5.4.13	<code>gst_data_repo_src_init()</code>	331
9.5.4.14	<code>gst_data_repo_src_parse_caps()</code>	331
9.5.4.15	<code>gst_data_repo_src_read_flexible_or_sparse_tensors()</code>	332
9.5.4.16	<code>gst_data_repo_src_read_json_file()</code>	333
9.5.4.17	<code>gst_data_repo_src_read_multi_images()</code>	333
9.5.4.18	<code>gst_data_repo_src_read_others()</code>	334
9.5.4.19	<code>gst_data_repo_src_read_tensors()</code>	335
9.5.4.20	the offset of the second sample is as follows.	335
9.5.4.21	fd offset: [6352 9488 9528 12664]	336
9.5.4.22	<code>gst_data_repo_src_set_caps()</code>	336
9.5.4.23	<code>gst_data_repo_src_set_file_path()</code>	337
9.5.4.24	<code>gst_data_repo_src_set_property()</code>	337
9.5.4.25	<code>gst_data_repo_src_set_tensors_sequence()</code>	338
9.5.4.26	<code>gst_data_repo_src_set_timestamp()</code>	339
9.5.4.27	<code>gst_data_repo_src_shuffle_samples_index()</code>	339
9.5.4.28	<code>gst_data_repo_src_start()</code>	340
9.5.4.29	<code>gst_data_repo_src_stop()</code>	340
9.5.4.30	<code>GST_DEBUG_CATEGORY_STATIC()</code>	341
9.5.5	Variable Documentation	341
9.5.5.1	<code>srctemplate</code>	341
9.6	<code>datarepo/gstdatareposrc.h</code> File Reference	341
9.6.1	Detailed Description	342
9.6.2	Macro Definition Documentation	342
9.6.2.1	<code>GST_DATA_REPO_SRC</code>	343
9.6.2.2	<code>GST_DATA_REPO_SRC_CLASS</code>	343
9.6.2.3	<code>GST_IS_DATA_REPO_SRC</code>	343
9.6.2.4	<code>GST_IS_DATA_REPO_SRC_CLASS</code>	343
9.6.2.5	<code>GST_TYPE_DATA_REPO_SRC</code>	343
9.6.3	Typedef Documentation	343
9.6.3.1	<code>GstDataRepoSrc</code>	344
9.6.3.2	<code>GstDataRepoSrcClass</code>	344
9.6.4	Function Documentation	344
9.6.4.1	<code>gst_data_repo_src_get_type()</code>	344
9.7	<code>edge/edge_common.c</code> File Reference	344
9.7.1	Detailed Description	345
9.7.2	Function Documentation	345
9.7.2.1	<code>gst_edge_get_connect_type()</code>	345
9.8	<code>edge/edge_common.h</code> File Reference	346
9.8.1	Detailed Description	347

9.8.2 Macro Definition Documentation	347
9.8.2.1 DEFAULT_CONNECT_TYPE	347
9.8.2.2 DEFAULT_HOST	347
9.8.2.3 DEFAULT_PORT	347
9.8.2.4 GST_EDGE_ELEM_NAME_SINK	348
9.8.2.5 GST_EDGE_ELEM_NAME_SRC	348
9.8.2.6 GST_EDGE_PACKAGE	348
9.8.2.7 GST_TYPE_EDGE_CONNECT_TYPE	348
9.8.3 Function Documentation	348
9.8.3.1 gst_edge_get_connect_type()	348
9.9 edge/edge_elements.c File Reference	349
9.9.1 Detailed Description	349
9.9.2 Macro Definition Documentation	350
9.9.2.1 PACKAGE	350
9.9.3 Function Documentation	350
9.9.3.1 plugin_init()	350
9.10 edge/edge_sink.c File Reference	350
9.10.1 Detailed Description	352
9.10.2 Macro Definition Documentation	352
9.10.2.1 DEFAULT_MQTT_HOST	352
9.10.2.2 DEFAULT_MQTT_PORT	352
9.10.2.3 GST_CAT_DEFAULT	353
9.10.2.4 gst_edgesink_parent_class	353
9.10.3 Enumeration Type Documentation	353
9.10.3.1 anonymous enum	353
9.10.4 Function Documentation	353
9.10.4.1 _nns_edge_event_cb()	354
9.10.4.2 _wait_connection()	354
9.10.4.3 G_DEFINE_TYPE()	354
9.10.4.4 GST_DEBUG_CATEGORY_STATIC()	355
9.10.4.5 gst_edgesink_class_init()	355
9.10.4.6 gst_edgesink_finalize()	355
9.10.4.7 gst_edgesink_get_connect_type()	356
9.10.4.8 gst_edgesink_get_host()	357
9.10.4.9 gst_edgesink_get_port()	357
9.10.4.10 gst_edgesink_get_property()	357
9.10.4.11 gst_edgesink_init()	358
9.10.4.12 gst_edgesink_render()	358
9.10.4.13 gst_edgesink_set_caps()	359
9.10.4.14 gst_edgesink_set_connect_type()	360
9.10.4.15 gst_edgesink_set_host()	360
9.10.4.16 gst_edgesink_set_port()	361

9.10.4.17 <code>gst_edgesink_set_property()</code>	361
9.10.4.18 <code>gst_edgesink_start()</code>	362
9.10.4.19 <code>gst_edgesink_stop()</code>	363
9.10.5 Variable Documentation	363
9.10.5.1 <code>sinktemplate</code>	363
9.11 <code>edge/edge_sink.h</code> File Reference	363
9.11.1 Detailed Description	365
9.11.2 Macro Definition Documentation	365
9.11.2.1 <code>GST_EDGESINK</code>	365
9.11.2.2 <code>GST_EDGESINK_CAST</code>	365
9.11.2.3 <code>GST_EDGESINK_CLASS</code>	366
9.11.2.4 <code>GST_IS_EDGESINK</code>	366
9.11.2.5 <code>GST_IS_EDGESINK_CLASS</code>	366
9.11.2.6 <code>GST_TYPE_EDGESINK</code>	366
9.11.3 Typedef Documentation	366
9.11.3.1 <code>GstEdgeSink</code>	366
9.11.3.2 <code>GstEdgeSinkClass</code>	367
9.11.4 Function Documentation	367
9.11.4.1 <code>gst_edgesink_get_type()</code>	367
9.12 <code>edge/edge_src.c</code> File Reference	367
9.12.1 Detailed Description	369
9.12.2 Macro Definition Documentation	369
9.12.2.1 <code>GST_CAT_DEFAULT</code>	369
9.12.2.2 <code>gst_edgesrc_parent_class</code>	369
9.12.3 Enumeration Type Documentation	369
9.12.3.1 anonymous enum	369
9.12.4 Function Documentation	370
9.12.4.1 <code>_nns_edge_event_cb()</code>	370
9.12.4.2 <code>G_DEFINE_TYPE()</code>	370
9.12.4.3 <code>GST_DEBUG_CATEGORY_STATIC()</code>	371
9.12.4.4 <code>gst_edgesrc_change_state()</code>	371
9.12.4.5 <code>gst_edgesrc_class_finalize()</code>	371
9.12.4.6 <code>gst_edgesrc_class_init()</code>	372
9.12.4.7 <code>gst_edgesrc_create()</code>	373
9.12.4.8 <code>gst_edgesrc_get_connect_type()</code>	373
9.12.4.9 <code>gst_edgesrc_get_dest_host()</code>	374
9.12.4.10 <code>gst_edgesrc_get_dest_port()</code>	374
9.12.4.11 <code>gst_edgesrc_get_property()</code>	375
9.12.4.12 <code>gst_edgesrc_init()</code>	375
9.12.4.13 <code>gst_edgesrc_set_connect_type()</code>	376
9.12.4.14 <code>gst_edgesrc_set_dest_host()</code>	376
9.12.4.15 <code>gst_edgesrc_set_dest_port()</code>	377

9.12.4.16 <code>gst_edgesrc_set_property()</code>	377
9.12.4.17 <code>gst_edgesrc_start()</code>	378
9.12.4.18 <code>gst_edgesrc_stop()</code>	378
9.12.5 Variable Documentation	379
9.12.5.1 <code>srctemplate</code>	379
9.13 <code>edge/edge_src.h</code> File Reference	379
9.13.1 Detailed Description	380
9.13.2 Macro Definition Documentation	380
9.13.2.1 <code>GST_EDGESRC</code>	381
9.13.2.2 <code>GST_EDGESRC_CAST</code>	381
9.13.2.3 <code>GST_EDGESRC_CLASS</code>	381
9.13.2.4 <code>GST_IS_EDGESRC</code>	381
9.13.2.5 <code>GST_IS_EDGESRC_CLASS</code>	381
9.13.2.6 <code>GST_TYPE_EDGESRC</code>	381
9.13.3 Typedef Documentation	382
9.13.3.1 <code>GstEdgeSrc</code>	382
9.13.3.2 <code>GstEdgeSrcClass</code>	382
9.13.4 Function Documentation	382
9.13.4.1 <code>gst_edgesrc_get_type()</code>	382
9.14 <code>join/gstjoin.c</code> File Reference	382
9.14.1 Detailed Description	384
9.14.2 Macro Definition Documentation	385
9.14.2.1 <code>GST_CAT_DEFAULT</code>	385
9.14.2.2 <code>GST_IS_JOIN_PAD</code>	385
9.14.2.3 <code>GST_IS_JOIN_PAD_CLASS</code>	385
9.14.2.4 <code>GST_JOIN_GET_COND</code>	385
9.14.2.5 <code>GST_JOIN_GET_LOCK</code>	385
9.14.2.6 <code>GST_JOIN_LOCK</code>	386
9.14.2.7 <code>GST_JOIN_PAD</code>	386
9.14.2.8 <code>GST_JOIN_PAD_CAST</code>	386
9.14.2.9 <code>GST_JOIN_PAD_CLASS</code>	386
9.14.2.10 <code>gst_join_parent_class</code>	386
9.14.2.11 <code>GST_JOIN_UNLOCK</code>	386
9.14.2.12 <code>GST_JOIN_WAIT</code>	387
9.14.2.13 <code>GST_TYPE_JOIN_PAD</code>	387
9.14.2.14 <code>PACKAGE</code>	387
9.14.3 Typedef Documentation	387
9.14.3.1 <code>GstJoinPad</code>	387
9.14.3.2 <code>GstJoinPadClass</code>	387
9.14.4 Enumeration Type Documentation	387
9.14.4.1 anonymous enum	387
9.14.5 Function Documentation	388

9.14.5.1 forward_sticky_events()	388
9.14.5.2 G_DEFINE_TYPE()	388
9.14.5.3 G_DEFINE_TYPE_WITH_CODE()	388
9.14.5.4 GST_DEBUG_CATEGORY_STATIC()	389
9.14.5.5 gst_join_class_init()	389
9.14.5.6 gst_join_dispose()	390
9.14.5.7 gst_join_finalize()	390
9.14.5.8 gst_join_get_active_sinkpad()	391
9.14.5.9 gst_join_get_linked_pad()	391
9.14.5.10 gst_join_get_property()	392
9.14.5.11 gst_join_init()	392
9.14.5.12 gst_join_pad_chain()	393
9.14.5.13 gst_join_pad_class_init()	393
9.14.5.14 gst_join_pad_event()	394
9.14.5.15 gst_join_pad_finalize()	394
9.14.5.16 gst_join_pad_get_type()	395
9.14.5.17 gst_join_pad_init()	395
9.14.5.18 gst_join_pad_iterate_linked_pads()	395
9.14.5.19 gst_join_pad_query()	396
9.14.5.20 gst_join_pad_reset()	397
9.14.5.21 gst_join_request_new_pad()	397
9.14.5.22 gst_join_set_active_pad()	398
9.14.5.23 plugin_init()	398
9.14.6 Variable Documentation	399
9.14.6.1 gst_join_sink_factory	399
9.14.6.2 gst_join_src_factory	399
9.15 join/gstjoin.h File Reference	399
9.15.1 Detailed Description	401
9.15.2 Macro Definition Documentation	401
9.15.2.1 GST_IS_JOIN	401
9.15.2.2 GST_IS_JOIN_CLASS	401
9.15.2.3 GST_JOIN	401
9.15.2.4 GST_JOIN_CLASS	402
9.15.2.5 GST_TYPE_JOIN	402
9.15.3 Typedef Documentation	402
9.15.3.1 GstJoin	402
9.15.3.2 GstJoinClass	402
9.15.4 Function Documentation	402
9.15.4.1 gst_join_get_type()	402
9.16 mqtt/mqttcommon.h File Reference	403
9.16.1 Detailed Description	404
9.16.2 Macro Definition Documentation	404

9.16.2.1	DEFAULT_MQTT_CONN_TIMEOUT_SEC	404
9.16.2.2	GST_MQTT_ELEM_NAME_SINK	405
9.16.2.3	GST_MQTT_ELEM_NAME_SRC	405
9.16.2.4	GST_MQTT_LEN_MSG_HDR	405
9.16.2.5	GST_MQTT_MAX_LEN_GST_CAPS_STR	405
9.16.2.6	GST_MQTT_MAX_NUM_MEMS	405
9.16.2.7	GST_MQTT_PACKAGE	405
9.16.2.8	GST_US_TO_NS_MULTIPLIER	406
9.16.2.9	UNUSED	406
9.16.3	Typedef Documentation	406
9.16.3.1	GstMQTTMessageHdr	406
9.16.3.2	mqtt_get_unix_epoch	406
9.16.4	Function Documentation	406
9.16.4.1	default_mqtt_get_unix_epoch()	407
9.17	mqtt/mqttelements.c File Reference	407
9.17.1	Macro Definition Documentation	408
9.17.1.1	PACKAGE	408
9.17.2	Function Documentation	408
9.17.2.1	plugin_init()	408
9.18	mqtt/mqttsink.c File Reference	408
9.18.1	Detailed Description	412
9.18.2	Macro Definition Documentation	412
9.18.2.1	GST_CAT_DEFAULT	412
9.18.2.2	gst_mqtt_sink_parent_class	412
9.18.3	Enumeration Type Documentation	412
9.18.3.1	anonymous enum	412
9.18.3.2	anonymous enum	413
9.18.4	Function Documentation	413
9.18.4.1	_mqtt_set_msg_buf_hdr()	414
9.18.4.2	_put_timestamp_to_msg_buf_hdr()	414
9.18.4.3	cb_mqtt_on_connect()	415
9.18.4.4	cb_mqtt_on_connect_failure()	415
9.18.4.5	cb_mqtt_on_connection_lost()	416
9.18.4.6	cb_mqtt_on_delivery_complete()	416
9.18.4.7	cb_mqtt_on_disconnect()	417
9.18.4.8	cb_mqtt_on_disconnect_failure()	417
9.18.4.9	cb_mqtt_on_message_arrived()	418
9.18.4.10	cb_mqtt_on_send_failure()	418
9.18.4.11	cb_mqtt_on_send_success()	419
9.18.4.12	G_DEFINE_TYPE()	419
9.18.4.13	GST_DEBUG_CATEGORY_STATIC()	419
9.18.4.14	gst_mqtt_sink_change_state()	419

9.18.4.15	gst_mqtt_sink_class_finalize()	420
9.18.4.16	gst_mqtt_sink_class_init()	421
9.18.4.17	gst_mqtt_sink_event()	422
9.18.4.18	gst_mqtt_sink_get_client_id()	422
9.18.4.19	gst_mqtt_sink_get_debug()	423
9.18.4.20	gst_mqtt_sink_get_host_address()	423
9.18.4.21	gst_mqtt_sink_get_host_port()	423
9.18.4.22	gst_mqtt_sink_get_max_msg_buf_size()	424
9.18.4.23	gst_mqtt_sink_get_mqtt_ntp_srvs()	424
9.18.4.24	gst_mqtt_sink_get_mqtt_ntp_sync()	424
9.18.4.25	gst_mqtt_sink_get_mqtt_qos()	425
9.18.4.26	gst_mqtt_sink_get_num_buffers()	425
9.18.4.27	gst_mqtt_sink_get_opt_cleansession()	425
9.18.4.28	gst_mqtt_sink_get_opt_keep_alive_interval()	426
9.18.4.29	gst_mqtt_sink_get_property()	426
9.18.4.30	gst_mqtt_sink_get_pub_topic()	428
9.18.4.31	gst_mqtt_sink_get_pub_wait_timeout()	428
9.18.4.32	gst_mqtt_sink_init()	429
9.18.4.33	gst_mqtt_sink_query()	429
9.18.4.34	gst_mqtt_sink_render()	430
9.18.4.35	gst_mqtt_sink_render_list()	431
9.18.4.36	gst_mqtt_sink_set_caps()	431
9.18.4.37	gst_mqtt_sink_set_client_id()	432
9.18.4.38	gst_mqtt_sink_set_debug()	433
9.18.4.39	gst_mqtt_sink_set_host_address()	433
9.18.4.40	gst_mqtt_sink_set_host_port()	434
9.18.4.41	gst_mqtt_sink_set_max_msg_buf_size()	434
9.18.4.42	gst_mqtt_sink_set_mqtt_ntp_srvs()	435
9.18.4.43	gst_mqtt_sink_set_mqtt_ntp_sync()	435
9.18.4.44	gst_mqtt_sink_set_mqtt_qos()	436
9.18.4.45	gst_mqtt_sink_set_num_buffers()	436
9.18.4.46	gst_mqtt_sink_set_opt_cleansession()	436
9.18.4.47	gst_mqtt_sink_set_opt_keep_alive_interval()	437
9.18.4.48	gst_mqtt_sink_set_property()	437
9.18.4.49	gst_mqtt_sink_set_pub_topic()	439
9.18.4.50	gst_mqtt_sink_set_pub_wait_timeout()	439
9.18.4.51	gst_mqtt_sink_start()	440
9.18.4.52	gst_mqtt_sink_stop()	441
9.18.5	Variable Documentation	441
9.18.5.1	DEFAULT_MQTT_CLIENT_ID	441
9.18.5.2	DEFAULT_MQTT_CLIENT_ID_FORMAT	442
9.18.5.3	DEFAULT_MQTT_HOST_ADDRESS	442

9.18.5.4	DEFAULT_MQTT_HOST_PORT	442
9.18.5.5	DEFAULT_MQTT_NTP_SERVERS	442
9.18.5.6	DEFAULT_MQTT_PUB_TOPIC	442
9.18.5.7	DEFAULT_MQTT_PUB_TOPIC_FORMAT	442
9.18.5.8	sink_client_id	443
9.18.5.9	sink_pad_template	443
9.18.5.10	TAG_ERR_MQTTSINK	443
9.19	mqtt/mqttsink.h File Reference	443
9.19.1	Detailed Description	445
9.19.2	Macro Definition Documentation	445
9.19.2.1	GST_IS_MQTT_SINK	445
9.19.2.2	GST_IS_MQTT_SINK_CLASS	445
9.19.2.3	GST_MQTT_SINK	446
9.19.2.4	GST_MQTT_SINK_CAST	446
9.19.2.5	GST_MQTT_SINK_CLASS	446
9.19.2.6	GST_TYPE_MQTT_SINK	446
9.19.3	Typedef Documentation	446
9.19.3.1	GstMqttSink	446
9.19.3.2	GstMqttSinkClass	447
9.19.3.3	mqtt_sink_state_t	447
9.19.4	Enumeration Type Documentation	447
9.19.4.1	_mqtt_sink_state_t	447
9.19.5	Function Documentation	447
9.19.5.1	gst_mqtt_sink_get_type()	447
9.20	mqtt/mqttsrc.c File Reference	448
9.20.1	Detailed Description	451
9.20.2	Macro Definition Documentation	451
9.20.2.1	GST_CAT_DEFAULT	451
9.20.2.2	gst_mqtt_src_parent_class	451
9.20.3	Enumeration Type Documentation	451
9.20.3.1	anonymous enum	451
9.20.3.2	anonymous enum	452
9.20.4	Function Documentation	452
9.20.4.1	_extract_mqtt_msg_hdr_from()	452
9.20.4.2	_is_gst_buffer_timestamp_valid()	453
9.20.4.3	_put_timestamp_on_gst_buf()	453
9.20.4.4	_subscribe()	454
9.20.4.5	_unsubscribe()	454
9.20.4.6	cb_memory_wrapped_destroy()	455
9.20.4.7	cb_mqtt_on_connect()	456
9.20.4.8	cb_mqtt_on_connect_failure()	456
9.20.4.9	cb_mqtt_on_connection_lost()	457

9.20.4.10	<code>cb_mqtt_on_message_arrived()</code>	457
9.20.4.11	<code>cb_mqtt_on_subscribe()</code>	458
9.20.4.12	<code>cb_mqtt_on_subscribe_failure()</code>	459
9.20.4.13	<code>cb_mqtt_on_unsubscribe()</code>	459
9.20.4.14	<code>cb_mqtt_on_unsubscribe_failure()</code>	459
9.20.4.15	<code>G_DEFINE_TYPE()</code>	460
9.20.4.16	<code>GST_DEBUG_CATEGORY_STATIC()</code>	460
9.20.4.17	<code>gst_mqtt_src_change_state()</code>	460
9.20.4.18	<code>gst_mqtt_src_class_finalize()</code>	461
9.20.4.19	<code>gst_mqtt_src_class_init()</code>	461
9.20.4.20	<code>gst_mqtt_src_create()</code>	462
9.20.4.21	<code>gst_mqtt_src_get_caps()</code>	464
9.20.4.22	<code>gst_mqtt_src_get_client_id()</code>	464
9.20.4.23	<code>gst_mqtt_src_get_debug()</code>	464
9.20.4.24	<code>gst_mqtt_src_get_host_address()</code>	465
9.20.4.25	<code>gst_mqtt_src_get_host_port()</code>	465
9.20.4.26	<code>gst_mqtt_src_get_is_live()</code>	465
9.20.4.27	<code>gst_mqtt_src_get_mqtt_qos()</code>	466
9.20.4.28	<code>gst_mqtt_src_get_opt_cleansession()</code>	466
9.20.4.29	<code>gst_mqtt_src_get_opt_keep_alive_interval()</code>	466
9.20.4.30	<code>gst_mqtt_src_get_property()</code>	467
9.20.4.31	<code>gst_mqtt_src_get_sub_timeout()</code>	468
9.20.4.32	<code>gst_mqtt_src_get_sub_topic()</code>	468
9.20.4.33	<code>gst_mqtt_src_get_times()</code>	469
9.20.4.34	<code>gst_mqtt_src_init()</code>	469
9.20.4.35	<code>gst_mqtt_src_is_seekable()</code>	470
9.20.4.36	<code>gst_mqtt_src_query()</code>	470
9.20.4.37	<code>gst_mqtt_src_renegotiate()</code>	471
9.20.4.38	<code>gst_mqtt_src_set_client_id()</code>	471
9.20.4.39	<code>gst_mqtt_src_set_debug()</code>	472
9.20.4.40	<code>gst_mqtt_src_set_host_address()</code>	472
9.20.4.41	<code>gst_mqtt_src_set_host_port()</code>	473
9.20.4.42	<code>gst_mqtt_src_set_is_live()</code>	473
9.20.4.43	<code>gst_mqtt_src_set_mqtt_qos()</code>	474
9.20.4.44	<code>gst_mqtt_src_set_opt_cleansession()</code>	474
9.20.4.45	<code>gst_mqtt_src_set_opt_keep_alive_interval()</code>	475
9.20.4.46	<code>gst_mqtt_src_set_property()</code>	475
9.20.4.47	<code>gst_mqtt_src_set_sub_timeout()</code>	476
9.20.4.48	<code>gst_mqtt_src_set_sub_topic()</code>	477
9.20.4.49	<code>gst_mqtt_src_start()</code>	478
9.20.4.50	<code>gst_mqtt_src_stop()</code>	478
9.20.5	Variable Documentation	479

9.20.5.1	DEFAULT_MQTT_CLIENT_ID	479
9.20.5.2	DEFAULT_MQTT_CLIENT_ID_FORMAT	479
9.20.5.3	DEFAULT_MQTT_HOST_ADDRESS	479
9.20.5.4	DEFAULT_MQTT_HOST_PORT	479
9.20.5.5	src_client_id	480
9.20.5.6	src_pad_template	480
9.20.5.7	TAG_ERR_MQTTSRC	480
9.21	mqtt/mqttsrc.h File Reference	480
9.21.1	Detailed Description	482
9.21.2	Macro Definition Documentation	482
9.21.2.1	GST_IS_MQTT_SRC	482
9.21.2.2	GST_IS_MQTT_SRC_CLASS	482
9.21.2.3	GST_MQTT_SRC	482
9.21.2.4	GST_MQTT_SRC_CAST	483
9.21.2.5	GST_MQTT_SRC_CLASS	483
9.21.2.6	GST_TYPE_MQTT_SRC	483
9.21.3	Typedef Documentation	483
9.21.3.1	GstMqttSrc	483
9.21.3.2	GstMqttSrcClass	483
9.21.4	Function Documentation	483
9.21.4.1	gst_mqtt_src_get_type()	484
9.22	mqtt/ntputil.c File Reference	484
9.22.1	Detailed Description	485
9.22.2	Typedef Documentation	485
9.22.2.1	ntp_packet_t	485
9.22.2.2	ntp_timestamp_t	486
9.22.3	Function Documentation	486
9.22.3.1	_convert_to_host_byte_order()	486
9.22.3.2	ntputil_get_epoch()	486
9.22.4	Variable Documentation	487
9.22.4.1	NTPUTIL_DEFAULT_HNAME	487
9.22.4.2	NTPUTIL_DEFAULT_PORT	488
9.22.4.3	NTPUTIL_MAX_FRAC_DOUBLE	488
9.22.4.4	NTPUTIL_SEC_TO_USEC_MULTIPLIER	488
9.22.4.5	NTPUTIL_TIMESTAMP_DELTA	488
9.23	mqtt/ntputil.h File Reference	488
9.23.1	Detailed Description	489
9.23.2	Function Documentation	489
9.23.2.1	_convert_to_host_byte_order()	490
9.23.2.2	ntputil_get_epoch()	490
9.24	nnstreamer/elements/gsttensor_aggregator.c File Reference	491
9.24.1	Detailed Description	493

9.24.2 Macro Definition Documentation	494
9.24.2.1 CAPS_STRING	494
9.24.2.2 DBG	494
9.24.2.3 DEFAULT_CONCAT	494
9.24.2.4 DEFAULT_FRAMES_DIMENSION	494
9.24.2.5 DEFAULT_FRAMES_FLUSH	494
9.24.2.6 DEFAULT_FRAMES_IN	495
9.24.2.7 DEFAULT_FRAMES_OUT	495
9.24.2.8 DEFAULT_SILENT	495
9.24.2.9 GST_CAT_DEFAULT	495
9.24.2.10 gst_tensor_aggregator_parent_class	495
9.24.2.11 silent_debug_config	496
9.24.3 Enumeration Type Documentation	496
9.24.3.1 anonymous enum	496
9.24.4 Function Documentation	496
9.24.4.1 G_DEFINE_TYPE()	496
9.24.4.2 GST_DEBUG_CATEGORY_STATIC()	497
9.24.4.3 gst_tensor_aggregator_chain()	497
9.24.4.4 gst_tensor_aggregator_change_state()	498
9.24.4.5 gst_tensor_aggregator_check_concat_axis()	498
9.24.4.6 gst_tensor_aggregator_class_init()	499
9.24.4.7 gst_tensor_aggregator_concat()	500
9.24.4.8 gst_tensor_aggregator_finalize()	503
9.24.4.9 gst_tensor_aggregator_get_adapter()	503
9.24.4.10 gst_tensor_aggregator_get_property()	504
9.24.4.11 gst_tensor_aggregator_init()	504
9.24.4.12 gst_tensor_aggregator_parse_caps()	505
9.24.4.13 gst_tensor_aggregator_push()	506
9.24.4.14 gst_tensor_aggregator_query_caps()	507
9.24.4.15 gst_tensor_aggregator_reset()	507
9.24.4.16 gst_tensor_aggregator_set_property()	508
9.24.4.17 gst_tensor_aggregator_sink_event()	508
9.24.4.18 gst_tensor_aggregator_sink_query()	509
9.24.4.19 gst_tensor_aggregator_src_query()	510
9.24.5 Variable Documentation	511
9.24.5.1 sink_template	511
9.24.5.2 src_template	511
9.25 nnstreamer/elements/gsttensor_aggregator.h File Reference	512
9.25.1 Detailed Description	513
9.25.2 Macro Definition Documentation	513
9.25.2.1 GST_IS_TENSOR_AGGREGATOR	513
9.25.2.2 GST_IS_TENSOR_AGGREGATOR_CLASS	514

9.25.2.3	GST_TENSOR_AGGREGATOR	514
9.25.2.4	GST_TENSOR_AGGREGATOR_CLASS	514
9.25.2.5	GST_TYPE_TENSOR_AGGREGATOR	514
9.25.3	Typedef Documentation	514
9.25.3.1	GstTensorAggregator	514
9.25.3.2	GstTensorAggregatorClass	515
9.25.4	Function Documentation	515
9.25.4.1	gst_tensor_aggregator_get_type()	515
9.26	nnstreamer/elements/gsttensor_converter.c File Reference	515
9.26.1	Detailed Description	518
9.26.2	Macro Definition Documentation	519
9.26.2.1	append_flex_tensor_caps_template	519
9.26.2.2	append_octet_caps_template	519
9.26.2.3	append_text_caps_template	519
9.26.2.4	DBG	519
9.26.2.5	DEFAULT_FRAMES_PER_TENSOR	519
9.26.2.6	DEFAULT_SET_TIMESTAMP	520
9.26.2.7	DEFAULT_SILENT	520
9.26.2.8	GST_CAT_DEFAULT	520
9.26.2.9	gst_tensor_converter_parent_class	520
9.26.2.10	OCTET_CAPS_STR	520
9.26.2.11	silent_debug_timestamp	521
9.26.2.12	STRING_CUSTOM_MODE	521
9.26.2.13	TEXT_CAPS_STR	521
9.26.3	Enumeration Type Documentation	521
9.26.3.1	anonymous enum	521
9.26.4	Function Documentation	522
9.26.4.1	_gst_tensor_converter_chain_chunk()	522
9.26.4.2	_gst_tensor_converter_chain_flex_tensor()	523
9.26.4.3	_gst_tensor_converter_chain_octet()	524
9.26.4.4	_gst_tensor_converter_chain_push()	524
9.26.4.5	_gst_tensor_converter_chain_segment()	525
9.26.4.6	_gst_tensor_converter_chain_timestamp()	526
9.26.4.7	findExternalConverter()	526
9.26.4.8	G_DEFINE_TYPE()	527
9.26.4.9	GST_DEBUG_CATEGORY_STATIC()	527
9.26.4.10	gst_tensor_converter_chain()	527
9.26.4.11	gst_tensor_converter_change_state()	529
9.26.4.12	gst_tensor_converter_class_init()	529
9.26.4.13	gst_tensor_converter_finalize()	530
9.26.4.14	gst_tensor_converter_get_adapter()	531
9.26.4.15	gst_tensor_converter_get_format_list()	532

9.26.4.16 gst_tensor_converter_get_possible_media_caps()	533
9.26.4.17 gst_tensor_converter_get_property()	533
9.26.4.18 gst_tensor_converter_init()	534
9.26.4.19 gst_tensor_converter_parse_audio()	535
9.26.4.20 gst_tensor_converter_parse_caps()	536
9.26.4.21 gst_tensor_converter_parse_custom()	537
9.26.4.22 gst_tensor_converter_parse_octet()	538
9.26.4.23 gst_tensor_converter_parse_tensor()	539
9.26.4.24 gst_tensor_converter_parse_text()	540
9.26.4.25 gst_tensor_converter_parse_video()	541
9.26.4.26 gst_tensor_converter_query_caps()	543
9.26.4.27 gst_tensor_converter_reset()	543
9.26.4.28 gst_tensor_converter_set_property()	544
9.26.4.29 gst_tensor_converter_sink_event()	545
9.26.4.30 gst_tensor_converter_sink_query()	547
9.26.4.31 gst_tensor_converter_src_query()	547
9.26.4.32 gst_tensor_converter_update_caps()	548
9.26.4.33 gst_tensor_converter_video_stride()	549
9.26.4.34 nnstreamer_converter_custom_register()	549
9.26.4.35 nnstreamer_converter_custom_unregister()	550
9.26.4.36 nnstreamer_converter_find()	550
9.26.4.37 nnstreamer_converter_set_custom_property_desc()	551
9.26.4.38 nnstreamer_converter_validate()	552
9.26.4.39 registerExternalConverter()	552
9.26.4.40 unregisterExternalConverter()	553
9.27 nnstreamer/elements/gsttensor_converter.h File Reference	553
9.27.1 Detailed Description	555
9.27.2 Macro Definition Documentation	555
9.27.2.1 GST_IS_TENSOR_CONVERTER	555
9.27.2.2 GST_IS_TENSOR_CONVERTER_CLASS	556
9.27.2.3 GST_TENSOR_CONVERTER	556
9.27.2.4 GST_TENSOR_CONVERTER_CLASS	556
9.27.2.5 GST_TYPE_TENSOR_CONVERTER	556
9.27.3 Typedef Documentation	556
9.27.3.1 GstTensorConverter	556
9.27.3.2 GstTensorConverterClass	556
9.27.4 Enumeration Type Documentation	556
9.27.4.1 tensor_converter_mode	556
9.27.5 Function Documentation	557
9.27.5.1 gst_tensor_converter_get_type()	557
9.28 nnstreamer/elements/gsttensor_converter_media_info_audio.h File Reference	557
9.28.1 Detailed Description	558

9.28.2 Macro Definition Documentation	558
9.28.2.1 append_audio_caps_template	559
9.28.2.2 AUDIO_CAPS_STR	559
9.28.2.3 is_audio_supported	559
9.29 nntstreamer/elements/gsttensor_converter_media_info_video.h File Reference	559
9.29.1 Detailed Description	560
9.29.2 Macro Definition Documentation	561
9.29.2.1 append_video_caps_template	561
9.29.2.2 is_video_supported	561
9.29.2.3 NNS_VIDEO_FORMAT	561
9.29.2.4 VIDEO_CAPS_STR	561
9.30 nntstreamer/elements/gsttensor_converter_media_no_audio.h File Reference	561
9.30.1 Detailed Description	562
9.30.2 Macro Definition Documentation	562
9.30.2.1 append_audio_caps_template	563
9.30.2.2 gst_audio_format_to_string	563
9.30.2.3 GST_AUDIO_INFO_BPF	563
9.30.2.4 GST_AUDIO_INFO_CHANNELS	563
9.30.2.5 GST_AUDIO_INFO_FORMAT	563
9.30.2.6 gst_audio_info_from_caps	563
9.30.2.7 gst_audio_info_init	564
9.30.2.8 GST_AUDIO_INFO_RATE	564
9.30.2.9 GstAudioInfo	564
9.30.2.10 is_audio_supported	564
9.30.3 Enumeration Type Documentation	564
9.30.3.1 GstAudioFormat	564
9.31 nntstreamer/elements/gsttensor_converter_media_no_video.h File Reference	565
9.31.1 Detailed Description	565
9.31.2 Macro Definition Documentation	566
9.31.2.1 append_video_caps_template	566
9.31.2.2 gst_video_format_to_string	566
9.31.2.3 GST_VIDEO_INFO_FORMAT	566
9.31.2.4 GST_VIDEO_INFO_FPS_D	566
9.31.2.5 GST_VIDEO_INFO_FPS_N	566
9.31.2.6 gst_video_info_from_caps	567
9.31.2.7 GST_VIDEO_INFO_HEIGHT	567
9.31.2.8 gst_video_info_init	567
9.31.2.9 GST_VIDEO_INFO_SIZE	567
9.31.2.10 GST_VIDEO_INFO_WIDTH	567
9.31.2.11 GstVideoInfo	567
9.31.2.12 is_video_supported	568
9.31.3 Enumeration Type Documentation	568

9.31.3.1 GstVideoFormat	568
9.32 nnsreamer/elements/gsttensor_crop.c File Reference	568
9.32.1 Detailed Description	570
9.32.2 Macro Definition Documentation	571
9.32.2.1 DEFAULT_LATENESS	571
9.32.2.2 DEFAULT_SILENT	571
9.32.2.3 GST_CAT_DEFAULT	571
9.32.2.4 gst_tensor_crop_parent_class	571
9.32.3 Enumeration Type Documentation	571
9.32.3.1 anonymous enum	571
9.32.4 Function Documentation	572
9.32.4.1 G_DEFINE_TYPE()	572
9.32.4.2 GST_DEBUG_CATEGORY_STATIC()	572
9.32.4.3 gst_tensor_crop_chain()	572
9.32.4.4 gst_tensor_crop_change_state()	573
9.32.4.5 gst_tensor_crop_class_init()	574
9.32.4.6 gst_tensor_crop_collected()	575
9.32.4.7 gst_tensor_crop_do_cropping()	576
9.32.4.8 gst_tensor_crop_finalize()	577
9.32.4.9 gst_tensor_crop_get_crop_info()	577
9.32.4.10 gst_tensor_crop_get_property()	578
9.32.4.11 gst_tensor_crop_init()	579
9.32.4.12 gst_tensor_crop_negotiate()	579
9.32.4.13 gst_tensor_crop_pad_reset()	580
9.32.4.14 gst_tensor_crop_prepare_out_meta()	580
9.32.4.15 gst_tensor_crop_reset()	581
9.32.4.16 gst_tensor_crop_set_property()	582
9.32.4.17 gst_tensor_crop_sink_event()	583
9.32.4.18 gst_tensor_crop_src_event()	583
9.32.5 Variable Documentation	584
9.32.5.1 info_template	584
9.32.5.2 raw_template	584
9.32.5.3 src_template	585
9.33 nnsreamer/elements/gsttensor_crop.h File Reference	585
9.33.1 Detailed Description	586
9.33.2 Macro Definition Documentation	587
9.33.2.1 GST_IS_TENSOR_CROP	587
9.33.2.2 GST_IS_TENSOR_CROP_CLASS	587
9.33.2.3 GST_TENSOR_CROP	587
9.33.2.4 GST_TENSOR_CROP_CLASS	587
9.33.2.5 GST_TYPE_TENSOR_CROP	587
9.33.3 Typedef Documentation	588

9.33.3.1 GstTensorCrop	588
9.33.3.2 GstTensorCropClass	588
9.33.4 Function Documentation	588
9.33.4.1 gst_tensor_crop_get_type()	588
9.34 nntstreamer/elements/gsttensor_debug.c File Reference	588
9.34.1 Detailed Description	590
9.34.2 Macro Definition Documentation	590
9.34.2.1 C_FLAGS	590
9.34.2.2 CAPS_STRING	590
9.34.2.3 DBG	591
9.34.2.4 DEFAULT_SILENT	591
9.34.2.5 DEFAULT_TENSOR_DEBUG_CAP	591
9.34.2.6 DEFAULT_TENSOR_DEBUG_META_FLAGS	591
9.34.2.7 DEFAULT_TENSOR_DEBUG_OUTPUT_FLAGS	591
9.34.2.8 GST_CAT_DEFAULT	592
9.34.2.9 gst_tensor_debug_parent_class	592
9.34.2.10 TENSOR_DEBUG_TYPE_CAPS	592
9.34.2.11 TENSOR_DEBUG_TYPE_META_FLAGS	592
9.34.2.12 TENSOR_DEBUG_TYPE_OUTPUT_FLAGS	592
9.34.3 Enumeration Type Documentation	592
9.34.3.1 anonymous enum	592
9.34.4 Function Documentation	593
9.34.4.1 _gst_tensor_debug_output()	593
9.34.4.2 G_DEFINE_TYPE()	593
9.34.4.3 GST_DEBUG_CATEGORY_STATIC()	594
9.34.4.4 gst_tensor_debug_class_init()	594
9.34.4.5 gst_tensor_debug_finalize()	595
9.34.4.6 gst_tensor_debug_fixate_caps()	595
9.34.4.7 gst_tensor_debug_get_property()	596
9.34.4.8 gst_tensor_debug_init()	596
9.34.4.9 gst_tensor_debug_set_caps()	596
9.34.4.10 gst_tensor_debug_set_property()	597
9.34.4.11 gst_tensor_debug_transform_ip()	597
9.34.4.12 tensor_debug_cap_get_type()	598
9.34.4.13 tensor_debug_meta_flags_get_type()	598
9.34.4.14 tensor_debug_output_flags_get_type()	598
9.34.5 Variable Documentation	598
9.34.5.1 sink_factory	598
9.34.5.2 src_factory	599
9.35 nntstreamer/elements/gsttensor_debug.h File Reference	599
9.35.1 Detailed Description	601
9.35.2 Macro Definition Documentation	601

9.35.2.1	GST_IS_TENSOR_DEBUG	601
9.35.2.2	GST_IS_TENSOR_DEBUG_CLASS	601
9.35.2.3	GST_TENSOR_DEBUG	601
9.35.2.4	GST_TENSOR_DEBUG_CAST	602
9.35.2.5	GST_TENSOR_DEBUG_CLASS	602
9.35.2.6	GST_TYPE_TENSOR_DEBUG	602
9.35.3	Typedef Documentation	602
9.35.3.1	GstTensorDebug	602
9.35.3.2	GstTensorDebugClass	602
9.35.4	Enumeration Type Documentation	602
9.35.4.1	tdbg_cap_mode	602
9.35.4.2	tdbg_meta_mode	603
9.35.4.3	tdbg_output_mode	603
9.35.5	Function Documentation	604
9.35.5.1	gst_tensor_debug_get_type()	604
9.36	nstreamer/elements/gsttensor_decoder.c File Reference	604
9.36.1	Detailed Description	606
9.36.2	Macro Definition Documentation	607
9.36.2.1	CAPS_STRING	607
9.36.2.2	DBG	607
9.36.2.3	DEFAULT_SILENT	607
9.36.2.4	GST_CAT_DEFAULT	608
9.36.2.5	gst_tensor_decoder_clean_plugin	608
9.36.2.6	gst_tensordec_parent_class	608
9.36.2.7	PROP_MODE_OPTION	608
9.36.2.8	PROP_READ_OPTION	609
9.36.3	Enumeration Type Documentation	609
9.36.3.1	anonymous enum	609
9.36.4	Function Documentation	609
9.36.4.1	failed()	610
9.36.4.2	G_DEFINE_TYPE()	610
9.36.4.3	g_free()	610
9.36.4.4	g_value_set_string()	611
9.36.4.5	GST_DEBUG_CATEGORY_STATIC()	612
9.36.4.6	gst_tensordec_check_consistency()	612
9.36.4.7	gst_tensordec_class_finalize()	613
9.36.4.8	gst_tensordec_class_init()	613
9.36.4.9	gst_tensordec_configure()	614
9.36.4.10	gst_tensordec_fixate_caps()	615
9.36.4.11	gst_tensordec_get_property()	616
9.36.4.12	gst_tensordec_init()	616
9.36.4.13	gst_tensordec_media_caps_from_structure()	617

9.36.4.14 gst_tensordec_media_caps_from_tensor()	617
9.36.4.15 gst_tensordec_process_plugin_options()	618
9.36.4.16 gst_tensordec_set_caps()	619
9.36.4.17 gst_tensordec_set_property()	620
9.36.4.18 gst_tensordec_transform()	620
9.36.4.19 gst_tensordec_transform_caps()	621
9.36.4.20 gst_tensordec_transform_size()	622
9.36.4.21 if()	623
9.36.4.22 nnstreamer_decoder_custom_register()	624
9.36.4.23 nnstreamer_decoder_custom_unregister()	625
9.36.4.24 nnstreamer_decoder_exit()	626
9.36.4.25 nnstreamer_decoder_find()	627
9.36.4.26 nnstreamer_decoder_probe()	628
9.36.4.27 nnstreamer_decoder_set_custom_property_desc()	628
9.36.4.28 nnstreamer_decoder_validate()	629
9.36.5 Variable Documentation	629
9.36.5.1 opnum	629
9.36.5.2 option	629
9.36.5.3 sink_factory	630
9.36.5.4 src_factory	630
9.37 nnstreamer/elements/gsttensor_decoder.h File Reference	630
9.37.1 Detailed Description	632
9.37.2 Macro Definition Documentation	632
9.37.2.1 GST_IS_TENSOR_DECODER	632
9.37.2.2 GST_IS_TENSOR_DECODER_CLASS	632
9.37.2.3 GST_TENSOR_DECODER	633
9.37.2.4 GST_TENSOR_DECODER_CAST	633
9.37.2.5 GST_TENSOR_DECODER_CLASS	633
9.37.2.6 GST_TYPE_TENSOR_DECODER	633
9.37.2.7 TensorDecMaxOpNum	633
9.37.3 Typedef Documentation	633
9.37.3.1 GstTensorDecoder	634
9.37.3.2 GstTensorDecoderClass	634
9.37.4 Function Documentation	634
9.37.4.1 gst_tensordec_get_type()	634
9.38 nnstreamer/elements/gsttensor_demux.c File Reference	634
9.38.1 Detailed Description	636
9.38.2 Macro Definition Documentation	636
9.38.2.1 CAPS_STRING_SINK	636
9.38.2.2 CAPS_STRING_SRC	637
9.38.2.3 GST_CAT_DEFAULT	637
9.38.2.4 gst_tensor_demux_parent_class	637

9.38.3 Enumeration Type Documentation	637
9.38.3.1 anonymous enum	637
9.38.4 Function Documentation	637
9.38.4.1 G_DEFINE_TYPE()	638
9.38.4.2 GST_DEBUG_CATEGORY_STATIC()	638
9.38.4.3 gst_tensor_demux_chain()	638
9.38.4.4 gst_tensor_demux_change_state()	639
9.38.4.5 gst_tensor_demux_class_init()	640
9.38.4.6 gst_tensor_demux_combine_flows()	640
9.38.4.7 gst_tensor_demux_dispose()	641
9.38.4.8 gst_tensor_demux_event()	642
9.38.4.9 gst_tensor_demux_get_property()	643
9.38.4.10 gst_tensor_demux_get_tensor_config()	643
9.38.4.11 gst_tensor_demux_get_tensor_pad()	644
9.38.4.12 gst_tensor_demux_init()	645
9.38.4.13 gst_tensor_demux_parse_caps()	646
9.38.4.14 gst_tensor_demux_remove_src_pads()	647
9.38.4.15 gst_tensor_demux_set_property()	648
9.38.5 Variable Documentation	648
9.38.5.1 sink_tmpl	648
9.38.5.2 src_tmpl	649
9.39 nntstreamer/elements/gsttensor_demux.h File Reference	649
9.39.1 Detailed Description	650
9.39.2 Macro Definition Documentation	651
9.39.2.1 GST_IS_TENSOR_DEMUX	651
9.39.2.2 GST_IS_TENSOR_DEMUX_CLASS	651
9.39.2.3 GST_TENSOR_DEMUX	651
9.39.2.4 GST_TENSOR_DEMUX_CAST	651
9.39.2.5 GST_TENSOR_DEMUX_CLASS	651
9.39.2.6 GST_TENSOR_DEMUX_GET_CLASS	652
9.39.2.7 GST_TYPE_TENSOR_DEMUX	652
9.39.3 Typedef Documentation	652
9.39.3.1 GstTensorDemux	652
9.39.3.2 GstTensorDemuxClass	652
9.39.4 Function Documentation	652
9.39.4.1 gst_tensor_demux_get_type()	652
9.40 nntstreamer/elements/gsttensor_if.c File Reference	653
9.40.1 Detailed Description	655
9.40.2 Macro Definition Documentation	656
9.40.2.1 CAPS_STRING	656
9.40.2.2 DBG	656
9.40.2.3 GST_CAT_DEFAULT	656

9.40.2.4	gst_tensor_if_parent_class	657
9.40.2.5	GST_TYPE_TENSOR_IF_ACT	657
9.40.2.6	GST_TYPE_TENSOR_IF_CV	657
9.40.2.7	GST_TYPE_TENSOR_IF_OP	657
9.40.2.8	operator_func	657
9.40.3	Enumeration Type Documentation	657
9.40.3.1	anonymous enum	657
9.40.4	Function Documentation	658
9.40.4.1	G_DEFINE_TYPE()	658
9.40.4.2	GST_DEBUG_CATEGORY_STATIC()	658
9.40.4.3	gst_tensor_if_act_get_type()	658
9.40.4.4	gst_tensor_if_calculate_cv()	659
9.40.4.5	gst_tensor_if_chain()	659
9.40.4.6	gst_tensor_if_check_condition()	660
9.40.4.7	gst_tensor_if_class_init()	661
9.40.4.8	gst_tensor_if_combine_flows()	662
9.40.4.9	gst_tensor_if_configure_custom_prop()	663
9.40.4.10	gst_tensor_if_cv_get_type()	663
9.40.4.11	gst_tensor_if_dispose()	664
9.40.4.12	gst_tensor_if_event()	664
9.40.4.13	gst_tensor_if_get_property()	665
9.40.4.14	gst_tensor_if_get_property_supplied_value()	666
9.40.4.15	gst_tensor_if_get_tensor_average()	667
9.40.4.16	gst_tensor_if_get_tensor_pad()	667
9.40.4.17	gst_tensor_if_init()	669
9.40.4.18	gst_tensor_if_install_properties()	669
9.40.4.19	gst_tensor_if_op_get_type()	670
9.40.4.20	gst_tensor_if_parse_caps()	670
9.40.4.21	gst_tensor_if_property_to_string()	671
9.40.4.22	gst_tensor_if_remove_src_pads()	672
9.40.4.23	gst_tensor_if_set_property()	673
9.40.4.24	gst_tensor_if_set_property_cv_option()	673
9.40.4.25	gst_tensor_if_set_property_glist()	674
9.40.4.26	gst_tensor_if_set_property_supplied_value()	675
9.40.4.27	nstreamer_if_custom_register()	675
9.40.4.28	nstreamer_if_custom_unregister()	676
9.40.4.29	switch()	676
9.40.5	Variable Documentation	676
9.40.5.1	cv	677
9.40.5.2	data	677
9.40.5.3	result	677
9.40.5.4	sink_factory	677

9.40.5.5 src_factory	677
9.40.5.6 svtc_1	678
9.40.5.7 svtc_2	678
9.40.5.8 TIFOP_NE	678
9.40.5.9 TRUE	678
9.40.5.10 type	678
9.41 nnstreamer/elements/gsttensor_if.h File Reference	679
9.41.1 Detailed Description	681
9.41.2 Macro Definition Documentation	681
9.41.2.1 GST_IS_TENSOR_IF	681
9.41.2.2 GST_IS_TENSOR_IF_CLASS	681
9.41.2.3 GST_TENSOR_IF	682
9.41.2.4 GST_TENSOR_IF_CAST	682
9.41.2.5 GST_TENSOR_IF_CLASS	682
9.41.2.6 GST_TENSOR_IF_GET_CLASS	682
9.41.2.7 GST_TYPE_TENSOR_IF	682
9.41.3 Typedef Documentation	682
9.41.3.1 GstTensorIf	683
9.41.3.2 GstTensorIfClass	683
9.41.4 Enumeration Type Documentation	683
9.41.4.1 tensor_if_behavior	683
9.41.4.2 tensor_if_compared_value	683
9.41.4.3 tensor_if_operator	684
9.41.4.4 tensor_if_srcpads	684
9.41.5 Function Documentation	685
9.41.5.1 gst_tensor_if_get_type()	685
9.42 nnstreamer/elements/gsttensor_merge.c File Reference	685
9.42.1 Detailed Description	687
9.42.2 Macro Definition Documentation	687
9.42.2.1 CAPS_STRING	687
9.42.2.2 DBG	688
9.42.2.3 GST_CAT_DEFAULT	688
9.42.2.4 gst_tensor_merge_parent_class	688
9.42.3 Enumeration Type Documentation	688
9.42.3.1 anonymous enum	688
9.42.4 Function Documentation	688
9.42.4.1 G_DEFINE_TYPE()	689
9.42.4.2 GST_DEBUG_CATEGORY_STATIC()	689
9.42.4.3 gst_tensor_merge_change_state()	689
9.42.4.4 gst_tensor_merge_class_init()	690
9.42.4.5 gst_tensor_merge_collect_buffer()	690
9.42.4.6 gst_tensor_merge_collected()	691

9.42.4.7	gst_tensor_merge_finalize()	693
9.42.4.8	gst_tensor_merge_generate_mem()	693
9.42.4.9	gst_tensor_merge_get_merged_config()	694
9.42.4.10	gst_tensor_merge_get_mode()	695
9.42.4.11	gst_tensor_merge_get_property()	696
9.42.4.12	gst_tensor_merge_init()	696
9.42.4.13	gst_tensor_merge_ready_to_paused()	697
9.42.4.14	gst_tensor_merge_request_new_pad()	698
9.42.4.15	gst_tensor_merge_send_segment_event()	698
9.42.4.16	gst_tensor_merge_set_option_data()	699
9.42.4.17	gst_tensor_merge_set_property()	699
9.42.4.18	gst_tensor_merge_set_src_caps()	700
9.42.4.19	gst_tensor_merge_sink_event()	701
9.42.4.20	gst_tensor_merge_src_event()	702
9.42.5	Variable Documentation	702
9.42.5.1	gst_tensor_merge_linear_string	702
9.42.5.2	gst_tensor_merge_mode_string	703
9.42.5.3	sink_tmpl	703
9.42.5.4	src_tmpl	703
9.43	nnstreamer/elements/gsttensor_merge.h File Reference	704
9.43.1	Detailed Description	705
9.43.2	Macro Definition Documentation	706
9.43.2.1	GST_IS_TENSOR_MERGE	706
9.43.2.2	GST_IS_TENSOR_MERGE_CLASS	706
9.43.2.3	GST_TENSOR_MERGE	706
9.43.2.4	GST_TENSOR_MERGE_CAST	706
9.43.2.5	GST_TENSOR_MERGE_CLASS	706
9.43.2.6	GST_TENSOR_MERGE_GET_CLASS	707
9.43.2.7	GST_TYPE_TENSOR_MERGE	707
9.43.3	Typedef Documentation	707
9.43.3.1	GstTensorMerge	707
9.43.3.2	GstTensorMergeClass	707
9.43.3.3	tensor_merge_linear	707
9.43.4	Enumeration Type Documentation	707
9.43.4.1	tensor_merge_linear_mode	707
9.43.4.2	tensor_merge_mode	708
9.43.5	Function Documentation	708
9.43.5.1	gst_tensor_merge_get_type()	708
9.44	nnstreamer/elements/gsttensor_mux.c File Reference	708
9.44.1	Detailed Description	710
9.44.2	Macro Definition Documentation	711
9.44.2.1	CAPS_STRING_SINK	711

9.44.2.2 CAPS_STRING_SRC	711
9.44.2.3 DBG	711
9.44.2.4 GST_CAT_DEFAULT	712
9.44.2.5 gst_tensor_mux_parent_class	712
9.44.3 Enumeration Type Documentation	712
9.44.3.1 anonymous enum	712
9.44.4 Function Documentation	712
9.44.4.1 G_DEFINE_TYPE()	712
9.44.4.2 GST_DEBUG_CATEGORY_STATIC()	713
9.44.4.3 gst_tensor_mux_chain_flex_tensor()	713
9.44.4.4 gst_tensor_mux_change_state()	714
9.44.4.5 gst_tensor_mux_class_init()	715
9.44.4.6 gst_tensor_mux_collect_buffer()	715
9.44.4.7 gst_tensor_mux_collected()	716
9.44.4.8 gst_tensor_mux_do_clip()	718
9.44.4.9 gst_tensor_mux_finalize()	718
9.44.4.10 gst_tensor_mux_get_property()	719
9.44.4.11 gst_tensor_mux_init()	720
9.44.4.12 gst_tensor_mux_ready_to_paused()	720
9.44.4.13 gst_tensor_mux_request_new_pad()	721
9.44.4.14 gst_tensor_mux_send_segment_event()	722
9.44.4.15 gst_tensor_mux_set_property()	722
9.44.4.16 gst_tensor_mux_set_src_caps()	723
9.44.4.17 gst_tensor_mux_set_waiting()	724
9.44.4.18 gst_tensor_mux_sink_event()	724
9.44.4.19 gst_tensor_mux_src_event()	725
9.44.5 Variable Documentation	725
9.44.5.1 sink_tmpl	725
9.44.5.2 src_tmpl	726
9.45 nntstreamer/elements/gsttensor_mux.h File Reference	726
9.45.1 Detailed Description	727
9.45.2 Macro Definition Documentation	728
9.45.2.1 GST_IS_TENSOR_MUX	728
9.45.2.2 GST_IS_TENSOR_MUX_CLASS	728
9.45.2.3 GST_TENSOR_MUX	728
9.45.2.4 GST_TENSOR_MUX_CAST	728
9.45.2.5 GST_TENSOR_MUX_CLASS	728
9.45.2.6 GST_TENSOR_MUX_GET_CLASS	729
9.45.2.7 GST_TYPE_TENSOR_MUX	729
9.45.3 Typedef Documentation	729
9.45.3.1 GstTensorMux	729
9.45.3.2 GstTensorMuxClass	729

9.45.4 Function Documentation	729
9.45.4.1 <code>gst_tensor_mux_get_type()</code>	729
9.46 nstreamer/elements/gsttensor_rate.c File Reference	730
9.46.1 Detailed Description	732
9.46.2 Macro Definition Documentation	732
9.46.2.1 <code>ABSDIFF</code>	732
9.46.2.2 <code>CAPS_STRING</code>	732
9.46.2.3 <code>DBG</code>	733
9.46.2.4 <code>DEFAULT_SILENT</code>	733
9.46.2.5 <code>DEFAULT_THROTTLE</code>	733
9.46.2.6 <code>GST_CAT_DEFAULT</code>	733
9.46.2.7 <code>gst_tensor_rate_parent_class</code>	734
9.46.2.8 <code>GST_TENSOR_RATE_SCALED_TIME</code>	734
9.46.2.9 <code>MAGIC_LIMIT</code>	734
9.46.2.10 <code>THROTTLE_DELAY_RATIO</code>	734
9.46.3 Enumeration Type Documentation	734
9.46.3.1 anonymous enum	734
9.46.4 Function Documentation	735
9.46.4.1 <code>G_DEFINE_TYPE()</code>	735
9.46.4.2 <code>GST_DEBUG_CATEGORY_STATIC()</code>	735
9.46.4.3 <code>gst_tensor_rate_class_init()</code>	735
9.46.4.4 <code>gst_tensor_rate_finalize()</code>	736
9.46.4.5 <code>gst_tensor_rate_fixate_caps()</code>	737
9.46.4.6 <code>gst_tensor_rate_flush_prev()</code>	737
9.46.4.7 <code>gst_tensor_rate_get_property()</code>	738
9.46.4.8 <code>gst_tensor_rate_init()</code>	738
9.46.4.9 <code>gst_tensor_rate_install_properties()</code>	739
9.46.4.10 <code>gst_tensor_rate_notify_drop()</code>	739
9.46.4.11 <code>gst_tensor_rate_notify_duplicate()</code>	740
9.46.4.12 <code>gst_tensor_rate_push_buffer()</code>	740
9.46.4.13 <code>gst_tensor_rate_reset()</code>	741
9.46.4.14 <code>gst_tensor_rate_send_qos_throttle()</code>	742
9.46.4.15 <code>gst_tensor_rate_set_caps()</code>	742
9.46.4.16 <code>gst_tensor_rate_set_property()</code>	743
9.46.4.17 <code>gst_tensor_rate_sink_event()</code>	743
9.46.4.18 <code>gst_tensor_rate_start()</code>	744
9.46.4.19 <code>gst_tensor_rate_stop()</code>	745
9.46.4.20 <code>gst_tensor_rate_swap_prev()</code>	746
9.46.4.21 <code>gst_tensor_rate_transform_caps()</code>	746
9.46.4.22 <code>gst_tensor_rate_transform_ip()</code>	747
9.46.5 Variable Documentation	747
9.46.5.1 <code>pspec_drop</code>	747

9.46.5.2 pspec_duplicate	748
9.46.5.3 sink_factory	748
9.46.5.4 src_factory	748
9.47 nnstreamer/elements/gsttensor_rate.h File Reference	748
9.47.1 Detailed Description	750
9.47.2 Macro Definition Documentation	750
9.47.2.1 GST_IS_TENSOR_RATE	750
9.47.2.2 GST_IS_TENSOR_RATE_CLASS	751
9.47.2.3 GST_TENSOR_RATE	751
9.47.2.4 GST_TENSOR_RATE_CAST	751
9.47.2.5 GST_TENSOR_RATE_CLASS	751
9.47.2.6 GST_TENSOR_RATE_GET_CLASS	751
9.47.2.7 GST_TYPE_TENSOR_RATE	751
9.47.3 Typedef Documentation	752
9.47.3.1 GstTensorRate	752
9.47.3.2 GstTensorRateClass	752
9.47.4 Function Documentation	752
9.47.4.1 gst_tensor_rate_get_type()	752
9.48 nnstreamer/elements/gsttensor_repo.c File Reference	752
9.48.1 Detailed Description	754
9.48.2 Macro Definition Documentation	754
9.48.2.1 DBG	754
9.48.2.2 GST_REPO_BROADCAST	754
9.48.2.3 GST_REPO_LOCK	755
9.48.2.4 GST_REPO_UNLOCK	755
9.48.2.5 GST_REPO_WAIT	755
9.48.3 Function Documentation	755
9.48.3.1 gst_tensor_repo_add_repodata()	755
9.48.3.2 gst_tensor_repo_check_changed()	756
9.48.3.3 gst_tensor_repo_check_eos()	757
9.48.3.4 gst_tensor_repo_get_buffer()	757
9.48.3.5 gst_tensor_repo_get_repodata()	758
9.48.3.6 gst_tensor_repo_init()	758
9.48.3.7 gst_tensor_repo_release_repodata()	759
9.48.3.8 gst_tensor_repo_remove_repodata()	760
9.48.3.9 gst_tensor_repo_set_buffer()	760
9.48.3.10 gst_tensor_repo_set_changed()	761
9.48.3.11 gst_tensor_repo_set_eos()	762
9.48.3.12 gst_tensor_repo_wait()	763
9.48.4 Variable Documentation	763
9.48.4.1 _repo	763
9.49 nnstreamer/elements/gsttensor_repo.h File Reference	763

9.49.1 Detailed Description	765
9.49.2 Function Documentation	765
9.49.2.1 <code>gst_tensor_repo_add_repodata()</code>	765
9.49.2.2 <code>gst_tensor_repo_check_changed()</code>	766
9.49.2.3 <code>gst_tensor_repo_check_eos()</code>	767
9.49.2.4 <code>gst_tensor_repo_get_buffer()</code>	767
9.49.2.5 <code>gst_tensor_repo_get_repodata()</code>	768
9.49.2.6 <code>gst_tensor_repo_init()</code>	768
9.49.2.7 <code>gst_tensor_repo_remove_repodata()</code>	769
9.49.2.8 <code>gst_tensor_repo_set_buffer()</code>	770
9.49.2.9 <code>gst_tensor_repo_set_changed()</code>	770
9.49.2.10 <code>gst_tensor_repo_set_eos()</code>	771
9.49.2.11 <code>gst_tensor_repo_wait()</code>	772
9.50 <code>nnstreamer/elements/gsttensor_reposink.c</code> File Reference	772
9.50.1 Detailed Description	774
9.50.2 Macro Definition Documentation	774
9.50.2.1 <code>DEFAULT_INDEX</code>	774
9.50.2.2 <code>DEFAULT_QOS</code>	774
9.50.2.3 <code>DEFAULT_SIGNAL_RATE</code>	775
9.50.2.4 <code>DEFAULT_SILENT</code>	775
9.50.2.5 <code>GST_CAT_DEFAULT</code>	775
9.50.2.6 <code>gst_tensor_reposink_parent_class</code>	775
9.50.3 Enumeration Type Documentation	775
9.50.3.1 anonymous enum	775
9.50.4 Function Documentation	776
9.50.4.1 <code>G_DEFINE_TYPE()</code>	776
9.50.4.2 <code>GST_DEBUG_CATEGORY_STATIC()</code>	776
9.50.4.3 <code>gst_tensor_reposink_class_init()</code>	776
9.50.4.4 <code>gst_tensor_reposink_dispose()</code>	777
9.50.4.5 <code>gst_tensor_reposink_event()</code>	777
9.50.4.6 <code>gst_tensor_reposink_get_caps()</code>	778
9.50.4.7 <code>gst_tensor_reposink_get_property()</code>	778
9.50.4.8 <code>gst_tensor_reposink_init()</code>	779
9.50.4.9 <code>gst_tensor_reposink_query()</code>	779
9.50.4.10 <code>gst_tensor_reposink_render()</code>	780
9.50.4.11 <code>gst_tensor_reposink_render_buffer()</code>	780
9.50.4.12 <code>gst_tensor_reposink_render_list()</code>	781
9.50.4.13 <code>gst_tensor_reposink_set_caps()</code>	782
9.50.4.14 <code>gst_tensor_reposink_set_property()</code>	782
9.50.4.15 <code>gst_tensor_reposink_start()</code>	783
9.50.4.16 <code>gst_tensor_reposink_stop()</code>	784
9.51 <code>nnstreamer/elements/gsttensor_reposink.h</code> File Reference	784

9.51.1 Detailed Description	786
9.51.2 Macro Definition Documentation	786
9.51.2.1 GST_IS_TENSOR_REPOSINK	786
9.51.2.2 GST_IS_TENSOR_REPOSINK_CLASS	786
9.51.2.3 GST_TENSOR_REPOSINK	787
9.51.2.4 GST_TENSOR_REPOSINK_CLASS	787
9.51.2.5 GST_TYPE_TENSOR_REPOSINK	787
9.51.3 Typedef Documentation	787
9.51.3.1 GstTensorRepoSink	787
9.51.3.2 GstTensorRepoSinkClass	787
9.51.4 Function Documentation	787
9.51.4.1 gst_tensor_reposink_get_type()	788
9.52 nnstreamer/elements/gsttensor_reposrc.c File Reference	788
9.52.1 Detailed Description	789
9.52.2 Macro Definition Documentation	790
9.52.2.1 CAPS_STRING	790
9.52.2.2 DEFAULT_INDEX	790
9.52.2.3 DEFAULT_SILENT	790
9.52.2.4 GST_CAT_DEFAULT	790
9.52.2.5 gst_tensor_reposrc_parent_class	790
9.52.2.6 INVALID_INDEX	790
9.52.3 Enumeration Type Documentation	790
9.52.3.1 anonymous enum	790
9.52.4 Function Documentation	791
9.52.4.1 G_DEFINE_TYPE()	791
9.52.4.2 GST_DEBUG_CATEGORY_STATIC()	791
9.52.4.3 gst_tensor_reposrc_class_init()	791
9.52.4.4 gst_tensor_reposrc_create()	792
9.52.4.5 gst_tensor_reposrc_dispose()	793
9.52.4.6 gst_tensor_reposrc_gen_dummy_buffer()	794
9.52.4.7 gst_tensor_reposrc_get_property()	794
9.52.4.8 gst_tensor_reposrc_getcaps()	795
9.52.4.9 gst_tensor_reposrc_init()	796
9.52.4.10 gst_tensor_reposrc_set_property()	796
9.53 nnstreamer/elements/gsttensor_reposrc.h File Reference	797
9.53.1 Detailed Description	798
9.53.2 Macro Definition Documentation	798
9.53.2.1 GST_IS_TENSOR_REPOSRC	798
9.53.2.2 GST_IS_TENSOR_REPOSRC_CLASS	799
9.53.2.3 GST_TENSOR_REPOSRC	799
9.53.2.4 GST_TENSOR_REPOSRC_CLASS	799
9.53.2.5 GST_TYPE_TENSOR_REPOSRC	799

9.53.3 Typedef Documentation	799
9.53.3.1 GstTensorRepoSrc	799
9.53.3.2 GstTensorRepoSrcClass	800
9.53.4 Function Documentation	800
9.53.4.1 gst_tensor_reposrc_get_type()	800
9.54 nntstreamer/elements/gsttensor_sink.c File Reference	800
9.54.1 Detailed Description	802
9.54.2 Macro Definition Documentation	803
9.54.2.1 DBG	803
9.54.2.2 DEFAULT_EMIT_SIGNAL	803
9.54.2.3 DEFAULT_QOS	803
9.54.2.4 DEFAULT_SIGNAL_RATE	803
9.54.2.5 DEFAULT_SILENT	804
9.54.2.6 DEFAULT_SYNC	804
9.54.2.7 GST_CAT_DEFAULT	804
9.54.2.8 gst_tensor_sink_parent_class	804
9.54.2.9 silent_debug_timestamp	804
9.54.3 Enumeration Type Documentation	804
9.54.3.1 anonymous enum	804
9.54.3.2 anonymous enum	805
9.54.4 Function Documentation	805
9.54.4.1 G_DEFINE_TYPE()	805
9.54.4.2 GST_DEBUG_CATEGORY_STATIC()	805
9.54.4.3 gst_tensor_sink_class_init()	806
9.54.4.4 gst_tensor_sink_event()	807
9.54.4.5 gst_tensor_sink_finalize()	807
9.54.4.6 gst_tensor_sink_get_emit_signal()	808
9.54.4.7 gst_tensor_sink_get_last_render_time()	808
9.54.4.8 gst_tensor_sink_get_property()	809
9.54.4.9 gst_tensor_sink_get_signal_rate()	810
9.54.4.10 gst_tensor_sink_get_silent()	810
9.54.4.11 gst_tensor_sink_init()	810
9.54.4.12 gst_tensor_sink_query()	811
9.54.4.13 gst_tensor_sink_render()	811
9.54.4.14 gst_tensor_sink_render_buffer()	812
9.54.4.15 gst_tensor_sink_render_list()	813
9.54.4.16 gst_tensor_sink_set_emit_signal()	814
9.54.4.17 gst_tensor_sink_set_last_render_time()	815
9.54.4.18 gst_tensor_sink_set_property()	815
9.54.4.19 gst_tensor_sink_set_signal_rate()	816
9.54.4.20 gst_tensor_sink_set_silent()	816
9.54.5 Variable Documentation	817

9.54.5.1 <code>_tensor_sink_signals</code>	817
9.55 <code>nnstreamer/elements/gsttensor_sink.h</code> File Reference	817
9.55.1 Detailed Description	818
9.55.2 Macro Definition Documentation	819
9.55.2.1 <code>GST_IS_TENSOR_SINK</code>	819
9.55.2.2 <code>GST_IS_TENSOR_SINK_CLASS</code>	819
9.55.2.3 <code>GST_TENSOR_SINK</code>	819
9.55.2.4 <code>GST_TENSOR_SINK_CLASS</code>	819
9.55.2.5 <code>GST_TYPE_TENSOR_SINK</code>	819
9.55.3 Typedef Documentation	820
9.55.3.1 <code>GstTensorSink</code>	820
9.55.3.2 <code>GstTensorSinkClass</code>	820
9.55.4 Function Documentation	820
9.55.4.1 <code>gst_tensor_sink_get_type()</code>	820
9.56 <code>nnstreamer/elements/gsttensor_sparsedec.c</code> File Reference	820
9.56.1 Detailed Description	822
9.56.2 Macro Definition Documentation	822
9.56.2.1 <code>DBG</code>	822
9.56.2.2 <code>DEFAULT_SILENT</code>	823
9.56.2.3 <code>GST_CAT_DEFAULT</code>	823
9.56.2.4 <code>gst_tensor_sparse_dec_parent_class</code>	823
9.56.3 Enumeration Type Documentation	823
9.56.3.1 anonymous enum	823
9.56.4 Function Documentation	823
9.56.4.1 <code>G_DEFINE_TYPE()</code>	824
9.56.4.2 <code>GST_DEBUG_CATEGORY_STATIC()</code>	824
9.56.4.3 <code>gst_tensor_sparse_dec_chain()</code>	824
9.56.4.4 <code>gst_tensor_sparse_dec_class_init()</code>	825
9.56.4.5 <code>gst_tensor_sparse_dec_finalize()</code>	825
9.56.4.6 <code>gst_tensor_sparse_dec_get_property()</code>	826
9.56.4.7 <code>gst_tensor_sparse_dec_init()</code>	827
9.56.4.8 <code>gst_tensor_sparse_dec_query_caps()</code>	827
9.56.4.9 <code>gst_tensor_sparse_dec_set_property()</code>	828
9.56.4.10 <code>gst_tensor_sparse_dec_sink_event()</code>	828
9.56.4.11 <code>gst_tensor_sparse_dec_sink_query()</code>	829
9.56.5 Variable Documentation	830
9.56.5.1 <code>sink_template</code>	830
9.56.5.2 <code>src_template</code>	830
9.57 <code>nnstreamer/elements/gsttensor_sparsedec.h</code> File Reference	831
9.57.1 Detailed Description	832
9.57.2 Macro Definition Documentation	832
9.57.2.1 <code>GST_IS_TENSOR_SPARSE_DEC</code>	832

9.57.2.2	GST_IS_TENSOR_SPARSE_DEC_CLASS	832
9.57.2.3	GST_TENSOR_SPARSE_DEC	833
9.57.2.4	GST_TENSOR_SPARSE_DEC_CLASS	833
9.57.2.5	GST_TYPE_TENSOR_SPARSE_DEC	833
9.57.3	Typedef Documentation	833
9.57.3.1	GstTensorSparseDec	833
9.57.3.2	GstTensorSparseDecClass	833
9.57.4	Function Documentation	833
9.57.4.1	gst_tensor_sparse_dec_get_type()	834
9.58	nntstreamer/elements/gsttensor_sparseenc.c File Reference	834
9.58.1	Detailed Description	835
9.58.2	Macro Definition Documentation	836
9.58.2.1	DBG	836
9.58.2.2	DEFAULT_SILENT	836
9.58.2.3	GST_CAT_DEFAULT	836
9.58.2.4	gst_tensor_sparse_enc_parent_class	836
9.58.3	Enumeration Type Documentation	836
9.58.3.1	anonymous enum	836
9.58.4	Function Documentation	837
9.58.4.1	G_DEFINE_TYPE()	837
9.58.4.2	GST_DEBUG_CATEGORY_STATIC()	837
9.58.4.3	gst_tensor_sparse_enc_chain()	837
9.58.4.4	gst_tensor_sparse_enc_class_init()	838
9.58.4.5	gst_tensor_sparse_enc_finalize()	839
9.58.4.6	gst_tensor_sparse_enc_get_property()	840
9.58.4.7	gst_tensor_sparse_enc_init()	840
9.58.4.8	gst_tensor_sparse_enc_parse_caps()	841
9.58.4.9	gst_tensor_sparse_enc_query_caps()	841
9.58.4.10	gst_tensor_sparse_enc_set_property()	842
9.58.4.11	gst_tensor_sparse_enc_sink_event()	842
9.58.4.12	gst_tensor_sparse_enc_sink_query()	843
9.58.5	Variable Documentation	844
9.58.5.1	sink_template	844
9.58.5.2	src_template	844
9.59	nntstreamer/elements/gsttensor_sparseenc.h File Reference	845
9.59.1	Detailed Description	846
9.59.2	Macro Definition Documentation	846
9.59.2.1	GST_IS_TENSOR_SPARSE_ENC	846
9.59.2.2	GST_IS_TENSOR_SPARSE_ENC_CLASS	846
9.59.2.3	GST_TENSOR_SPARSE_ENC	847
9.59.2.4	GST_TENSOR_SPARSE_ENC_CLASS	847
9.59.2.5	GST_TYPE_TENSOR_SPARSE_ENC	847

9.59.3 Typedef Documentation	847
9.59.3.1 GstTensorSparseEnc	847
9.59.3.2 GstTensorSparseEncClass	847
9.59.4 Function Documentation	847
9.59.4.1 gst_tensor_sparse_enc_get_type()	848
9.60 nntstreamer/elements/gsttensor_sparseutil.c File Reference	848
9.60.1 Detailed Description	848
9.60.2 Function Documentation	849
9.60.2.1 gst_tensor_sparse_from_dense()	849
9.60.2.2 gst_tensor_sparse_to_dense()	850
9.61 nntstreamer/elements/gsttensor_sparseutil.h File Reference	851
9.61.1 Detailed Description	852
9.61.2 Function Documentation	852
9.61.2.1 gst_tensor_sparse_from_dense()	852
9.61.2.2 gst_tensor_sparse_to_dense()	853
9.62 nntstreamer/elements/gsttensor_split.c File Reference	854
9.62.1 Detailed Description	856
9.62.2 Macro Definition Documentation	856
9.62.2.1 CAPS_STRING	856
9.62.2.2 GST_CAT_DEFAULT	856
9.62.2.3 gst_tensor_split_parent_class	857
9.62.3 Enumeration Type Documentation	857
9.62.3.1 anonymous enum	857
9.62.4 Function Documentation	857
9.62.4.1 _clear_tensorseg()	857
9.62.4.2 G_DEFINE_TYPE()	858
9.62.4.3 GST_DEBUG_CATEGORY_STATIC()	858
9.62.4.4 gst_tensor_split_chain()	858
9.62.4.5 gst_tensor_split_change_state()	859
9.62.4.6 gst_tensor_split_class_init()	860
9.62.4.7 gst_tensor_split_combine_flows()	860
9.62.4.8 gst_tensor_split_event()	861
9.62.4.9 gst_tensor_split_finalize()	862
9.62.4.10 gst_tensor_split_get_capsparam()	862
9.62.4.11 gst_tensor_split_get_property()	863
9.62.4.12 gst_tensor_split_getSplitted()	864
9.62.4.13 gst_tensor_split_get_tensor_pad()	865
9.62.4.14 gst_tensor_split_init()	866
9.62.4.15 gst_tensor_split_remove_src_pads()	867
9.62.4.16 gst_tensor_split_set_property()	868
9.62.5 Variable Documentation	868
9.62.5.1 sink_tmpl	868

9.62.5.2 src_tmpl	869
9.63 nnstreamer/elements/gsttensor_split.h File Reference	869
9.63.1 Detailed Description	870
9.63.2 Macro Definition Documentation	871
9.63.2.1 GST_IS_TENSOR_SPLIT	871
9.63.2.2 GST_IS_TENSOR_SPLIT_CLASS	871
9.63.2.3 GST_TENSOR_SPLIT	871
9.63.2.4 GST_TENSOR_SPLIT_CAST	871
9.63.2.5 GST_TENSOR_SPLIT_CLASS	871
9.63.2.6 GST_TENSOR_SPLIT_GET_CLASS	872
9.63.2.7 GST_TYPE_TENSOR_SPLIT	872
9.63.3 Typedef Documentation	872
9.63.3.1 GstTensorSplit	872
9.63.3.2 GstTensorSplitClass	872
9.63.4 Function Documentation	872
9.63.4.1 gst_tensor_split_get_type()	872
9.64 nnstreamer/elements/gsttensor_srcio.c File Reference	873
9.64.1 Detailed Description	877
9.64.2 Macro Definition Documentation	877
9.64.2.1 AVAIL_FREQUENCY_FILE	878
9.64.2.2 BLOCKSIZE	878
9.64.2.3 BUFFER	878
9.64.2.4 CHANNELS	878
9.64.2.5 CHANNELS_ENABLED_ALL_CHAR	878
9.64.2.6 CHANNELS_ENABLED_AUTO_CHAR	878
9.64.2.7 CURRENT_TRIGGER	879
9.64.2.8 DBG	879
9.64.2.9 DEFAULT_BUFFER_CAPACITY	879
9.64.2.10 DEFAULT_FREQUENCY	880
9.64.2.11 DEFAULT_MERGE_CHANNELS	880
9.64.2.12 DEFAULT_OPERATING_CHANNELS_ENABLED	880
9.64.2.13 DEFAULT_OPERATING_MODE	880
9.64.2.14 DEFAULT_POLL_TIMEOUT	880
9.64.2.15 DEFAULT_PROP_BASE_DIRECTORY	880
9.64.2.16 DEFAULT_PROP_DEV_DIRECTORY	881
9.64.2.17 DEFAULT_PROP_DEVICE_NUM	881
9.64.2.18 DEFAULT_PROP_SILENT	881
9.64.2.19 DEFAULT_PROP_STRING	881
9.64.2.20 DEFAULT_PROP_TRIGGER_NUM	881
9.64.2.21 DEVICE	882
9.64.2.22 DEVICE_PREFIX	882
9.64.2.23 EN_SUFFIX	882

9.64.2.24	GST_CAT_DEFAULT	882
9.64.2.25	gst_tensor_src_iio_parent_class	882
9.64.2.26	IIO	883
9.64.2.27	INDEX_SUFFIX	883
9.64.2.28	MAX_BUFFER_CAPACITY	883
9.64.2.29	MAX_FREQUENCY	883
9.64.2.30	MAX_POLL_TIMEOUT	883
9.64.2.31	MIN_BUFFER_CAPACITY	883
9.64.2.32	MIN_FREQUENCY	884
9.64.2.33	MIN_POLL_TIMEOUT	884
9.64.2.34	MODE_CONTINUOUS	884
9.64.2.35	MODE_ONE_SHOT	884
9.64.2.36	NAME_FILE	884
9.64.2.37	OFFSET_SUFFIX	885
9.64.2.38	PROCESS_SCANNED_DATA	885
9.64.2.39	SAMPLING_FREQUENCY	885
9.64.2.40	SCALE_SUFFIX	885
9.64.2.41	TIMESTAMP	885
9.64.2.42	TRIGGER	886
9.64.2.43	TRIGGER_PREFIX	886
9.64.2.44	TYPE_SUFFIX	886
9.64.3	Enumeration Type Documentation	886
9.64.3.1	anonymous enum	886
9.64.4	Function Documentation	887
9.64.4.1	g_assert()	887
9.64.4.2	G_DEFINE_TYPE()	888
9.64.4.3	GST_DEBUG_CATEGORY_STATIC()	889
9.64.4.4	gst_tensor_channel_list_filter_enabled()	889
9.64.4.5	gst_tensor_channel_list_sort_cmp()	889
9.64.4.6	gst_tensor_get_size_from_channels()	890
9.64.4.7	gst_tensor_set_all_channels()	890
9.64.4.8	gst_tensor_src_iio_change_state()	890
9.64.4.9	gst_tensor_src_iio_channel_properties_free()	891
9.64.4.10	gst_tensor_src_iio_class_init()	892
9.64.4.11	gst_tensor_src_iio_create()	893
9.64.4.12	gst_tensor_src_iio_create_config()	893
9.64.4.13	gst_tensor_src_iio_device_properties_init()	894
9.64.4.14	gst_tensor_src_iio_event()	894
9.64.4.15	gst_tensor_src_iio_fill()	895
9.64.4.16	gst_tensor_src_iio_finalize()	896
9.64.4.17	gst_tensor_src_iio_fixate()	897
9.64.4.18	gst_tensor_src_iio_get_all_channel_info()	897

9.64.4.19	<code>gst_tensor_src_iio_get_available_frequency()</code>	898
9.64.4.20	<code>gst_tensor_src_iio_get_caps()</code>	899
9.64.4.21	<code>gst_tensor_src_iio_get_float_from_file()</code>	899
9.64.4.22	<code>gst_tensor_src_iio_get_generic_name()</code>	900
9.64.4.23	<code>gst_tensor_src_iio_get_id_by_name()</code>	900
9.64.4.24	<code>gst_tensor_src_iio_get_name_by_id()</code>	901
9.64.4.25	<code>gst_tensor_src_iio_get_property()</code>	901
9.64.4.26	<code>gst_tensor_src_iio_get_times()</code>	902
9.64.4.27	<code>gst_tensor_src_iio_init()</code>	902
9.64.4.28	<code>gst_tensor_src_iio_is_seekable()</code>	903
9.64.4.29	<code>gst_tensor_src_iio_process_scanned_data()</code>	903
9.64.4.30	<code>gst_tensor_src_iio_set_caps()</code>	904
9.64.4.31	<code>gst_tensor_src_iio_set_channel_type()</code>	905
9.64.4.32	<code>gst_tensor_src_iio_set_property()</code>	905
9.64.4.33	<code>gst_tensor_src_iio_setup_device_buffer()</code>	906
9.64.4.34	<code>gst_tensor_src_iio_setup_device_properties()</code>	907
9.64.4.35	<code>gst_tensor_src_iio_setup_sampling_frequency()</code>	907
9.64.4.36	<code>gst_tensor_src_iio_setup_scan_channels()</code>	908
9.64.4.37	<code>gst_tensor_src_iio_setup_trigger_properties()</code>	908
9.64.4.38	<code>gst_tensor_src_iio_start()</code>	909
9.64.4.39	<code>gst_tensor_src_iio_stop()</code>	910
9.64.4.40	<code>gst_tensor_src_merge_tensor_by_type()</code>	910
9.64.4.41	<code>gst_tensor_src_restore_iio_device()</code>	911
9.64.4.42	<code>gst_tensor_write_sysfs_int()</code>	912
9.64.4.43	<code>gst_tensor_write_sysfs_string()</code>	913
9.64.4.44	<code>if()</code>	913
9.64.4.45	<code>PROCESS_SCANNED_DATA()</code> [1/4]	914
9.64.4.46	<code>PROCESS_SCANNED_DATA()</code> [2/4]	914
9.64.4.47	<code>PROCESS_SCANNED_DATA()</code> [3/4]	914
9.64.4.48	<code>PROCESS_SCANNED_DATA()</code> [4/4]	914
9.64.5	Variable Documentation	914
9.64.5.1	<code>else</code>	914
9.64.5.2	<code>prop</code>	915
9.64.5.3	<code>value_float</code>	915
9.64.5.4	<code>value_unsigned</code>	915
9.65	<code>nnstreamer/elements/gsttensor_srcio.h</code> File Reference	915
9.65.1	Detailed Description	917
9.65.2	Macro Definition Documentation	917
9.65.2.1	<code>GST_IS_TENSOR_SRC_IIO</code>	917
9.65.2.2	<code>GST_IS_TENSOR_SRC_IIO_CLASS</code>	918
9.65.2.3	<code>GST_TENSOR_SRC_IIO</code>	918
9.65.2.4	<code>GST_TENSOR_SRC_IIO_CAST</code>	918

9.65.2.5	GST_TENSOR_SRC_IIO_CLASS	918
9.65.2.6	GST_TYPE_TENSOR_SRC_IIO	918
9.65.3	Typedef Documentation	918
9.65.3.1	GstTensorSrcIIO	919
9.65.3.2	GstTensorSrcIIOChannelProperties	919
9.65.3.3	GstTensorSrcIIOClass	919
9.65.3.4	GstTensorSrcIIODeviceProperties	919
9.65.4	Enumeration Type Documentation	919
9.65.4.1	channels_enabled_options	919
9.65.5	Function Documentation	920
9.65.5.1	gst_tensor_src_iio_get_type()	920
9.66	nstreamer/elements/gsttensor_trainer.c File Reference	920
9.66.1	Detailed Description	923
9.66.2	Macro Definition Documentation	923
9.66.2.1	DEFAULT_PROP_EPOCHS	923
9.66.2.2	DEFAULT_PROP_INPUT_LIST	923
9.66.2.3	DEFAULT_PROP_LABEL_LIST	924
9.66.2.4	DEFAULT_PROP_TRAIN_SAMPLES	924
9.66.2.5	DEFAULT_PROP_VALID_SAMPLES	924
9.66.2.6	DEFAULT_STR_PROP_VALUE	924
9.66.2.7	GST_CAT_DEFAULT	924
9.66.2.8	gst_tensor_trainer_parent_class	924
9.66.2.9	MODEL_STATS_SIZE	925
9.66.2.10	SINK_CAPS_STRING	925
9.66.2.11	SRC_CAPS_STRING	925
9.66.3	Enumeration Type Documentation	925
9.66.3.1	anonymous enum	925
9.66.3.2	anonymous enum	926
9.66.4	Function Documentation	926
9.66.4.1	G_DEFINE_TYPE()	926
9.66.4.2	GST_DEBUG_CATEGORY_STATIC()	926
9.66.4.3	gst_tensor_trainer_chain()	927
9.66.4.4	gst_tensor_trainer_change_state()	927
9.66.4.5	gst_tensor_trainer_check_buffer_drop_conditions()	928
9.66.4.6	gst_tensor_trainer_check_chain_conditions()	929
9.66.4.7	gst_tensor_trainer_check_invalid_param()	929
9.66.4.8	gst_tensor_trainer_class_init()	930
9.66.4.9	gst_tensor_trainer_convert_meta()	930
9.66.4.10	gst_tensor_trainer_create_event_notifier()	931
9.66.4.11	gst_tensor_trainer_create_framework()	931
9.66.4.12	gst_tensor_trainer_create_model()	932
9.66.4.13	gst_tensor_trainer_create_output()	932

9.66.4.14	<code>gst_tensor_trainer_dummy_data_generation_func()</code>	933
9.66.4.15	<code>gst_tensor_trainer_epochs_is_complete()</code>	934
9.66.4.16	<code>gst_tensor_trainer_finalize()</code>	934
9.66.4.17	<code>gst_tensor_trainer_find_framework()</code>	935
9.66.4.18	<code>gst_tensor_trainer_get_model_stats()</code>	936
9.66.4.19	<code>gst_tensor_trainer_get_property()</code>	936
9.66.4.20	<code>gst_tensor_trainer_get_tensor_size()</code>	937
9.66.4.21	<code>gst_tensor_trainer_init()</code>	937
9.66.4.22	<code>gst_tensor_trainer_push_input()</code>	938
9.66.4.23	<code>gst_tensor_trainer_query_caps()</code>	939
9.66.4.24	<code>gst_tensor_trainer_set_model_load_path()</code>	940
9.66.4.25	<code>gst_tensor_trainer_set_model_save_path()</code>	940
9.66.4.26	<code>gst_tensor_trainer_set_output_meta()</code>	941
9.66.4.27	<code>gst_tensor_trainer_set_prop_framework()</code>	942
9.66.4.28	<code>gst_tensor_trainer_set_prop_model_config_file_path()</code>	942
9.66.4.29	<code>gst_tensor_trainer_set_property()</code>	943
9.66.4.30	<code>gst_tensor_trainer_sink_event()</code>	944
9.66.4.31	<code>gst_tensor_trainer_sink_query()</code>	945
9.66.4.32	<code>gst_tensor_trainer_src_query()</code>	946
9.66.4.33	<code>gst_tensor_trainer_start_model_training()</code>	947
9.66.4.34	<code>gst_tensor_trainer_stop_model_training()</code>	947
9.66.4.35	<code>gst_tensor_trainer_wait_for_epoch_completion()</code>	948
9.66.4.36	<code>gst_tensor_trainer_wait_for_training_completion()</code>	948
9.66.4.37	<code>nnstreamer_trainer_exit()</code>	948
9.66.4.38	<code>nnstreamer_trainer_notify_event()</code>	949
9.66.4.39	<code>nnstreamer_trainer_probe()</code>	949
9.66.5	Variable Documentation	950
9.66.5.1	<code>sink_template</code>	950
9.66.5.2	<code>src_template</code>	950
9.67	<code>nnstreamer/elements/gsttensor_trainer.h</code> File Reference	951
9.67.1	Detailed Description	952
9.67.2	Macro Definition Documentation	952
9.67.2.1	<code>GST_IS_TENSOR_TRAINER</code>	952
9.67.2.2	<code>GST_IS_TENSOR_TRAINER_CLASS</code>	952
9.67.2.3	<code>GST_TENSOR_TRAINER</code>	953
9.67.2.4	<code>GST_TENSOR_TRAINER_CLASS</code>	953
9.67.2.5	<code>GST_TYPE_TENSOR_TRAINER</code>	953
9.67.3	Typedef Documentation	953
9.67.3.1	<code>GstTensorTrainer</code>	953
9.67.3.2	<code>GstTensorTrainerClass</code>	953
9.67.4	Function Documentation	953
9.67.4.1	<code>gst_tensor_trainer_get_type()</code>	954

9.68 nstreamer/elements/gsttensor_transform.c File Reference	954
9.68.1 Detailed Description	957
9.68.2 Macro Definition Documentation	957
9.68.2.1 _conv_from_f16	957
9.68.2.2 _conv_to_f16	957
9.68.2.3 _op_float16	958
9.68.2.4 CAPS_STRING	958
9.68.2.5 DBG	958
9.68.2.6 DEFAULT_ACCELERATION	958
9.68.2.7 GST_CAT_DEFAULT	959
9.68.2.8 gst_tensor_transform_parent_class	959
9.68.2.9 GST_TYPE_TENSOR_TRANSFORM_MODE	959
9.68.2.10 handle_operator	959
9.68.2.11 NNS_TENSOR_PADDING_RANK_LIMIT	959
9.68.2.12 NNS_TENSOR_TRANSPOSE_RANK_LIMIT	960
9.68.2.13 REGEX_ARITH_OPTION	960
9.68.2.14 REGEX_ARITH_OPTION_TYPECAST	960
9.68.2.15 REGEX_CLAMP_OPTION	960
9.68.2.16 REGEX_DIMCHG_OPTION	960
9.68.2.17 REGEX_PADDING_OPTION	961
9.68.2.18 REGEX_STAND_OPTION	961
9.68.2.19 REGEX_TRANSPOSE_OPTION	961
9.68.2.20 REGEX_TYPECAST_OPTION	961
9.68.2.21 transposeloop	961
9.68.3 Enumeration Type Documentation	962
9.68.3.1 anonymous enum	962
9.68.4 Function Documentation	962
9.68.4.1 float16_not_supported()	962
9.68.4.2 G_DEFINE_TYPE()	963
9.68.4.3 GST_DEBUG_CATEGORY_STATIC()	963
9.68.4.4 gst_tensor_transform_arithmetic()	963
9.68.4.5 gst_tensor_transform_clamp()	965
9.68.4.6 gst_tensor_transform_class_init()	966
9.68.4.7 gst_tensor_transform_convert_dimension()	967
9.68.4.8 gst_tensor_transform_dimchg()	968
9.68.4.9 gst_tensor_transform_finalize()	969
9.68.4.10 gst_tensor_transform_fixate_caps()	970
9.68.4.11 gst_tensor_transform_get_operator()	970
9.68.4.12 gst_tensor_transform_get_property()	971
9.68.4.13 gst_tensor_transform_get_stand_mode()	972
9.68.4.14 gst_tensor_transform_init()	973
9.68.4.15 gst_tensor_transform_mode_get_type()	973

9.68.4.16	gst_tensor_transform_padding()	973
9.68.4.17	gst_tensor_transform_read_caps()	974
9.68.4.18	gst_tensor_transform_set_caps()	975
9.68.4.19	gst_tensor_transform_set_option_data()	976
9.68.4.20	gst_tensor_transform_set_property()	977
9.68.4.21	gst_tensor_transform_stand()	978
9.68.4.22	gst_tensor_transform_transform()	979
9.68.4.23	gst_tensor_transform_transform_caps()	981
9.68.4.24	gst_tensor_transform_transform_size()	982
9.68.4.25	gst_tensor_transform_transpose()	983
9.68.4.26	gst_tensor_transform_typecast()	984
9.68.4.27	while()	985
9.68.5	Variable Documentation	985
9.68.5.1	__pad0__	985
9.68.5.2	break	985
9.68.5.3	FALSE	986
9.68.5.4	gst_tensor_transform_operator_string	986
9.68.5.5	gst_tensor_transform_stand_string	986
9.68.5.6	GTT_OP_MUL	986
9.68.5.7	operator%d", oper)	986
9.68.5.8	sink_factory	987
9.68.5.9	src_factory	987
9.69	nnstreamer/elements/gsttensor_transform.h File Reference	987
9.69.1	Detailed Description	989
9.69.2	Macro Definition Documentation	990
9.69.2.1	GST_IS_TENSOR_TRANSFORM	990
9.69.2.2	GST_IS_TENSOR_TRANSFORM_CLASS	990
9.69.2.3	GST_TENSOR_TRANSFORM	990
9.69.2.4	GST_TENSOR_TRANSFORM_CAST	991
9.69.2.5	GST_TENSOR_TRANSFORM_CLASS	991
9.69.2.6	GST_TYPE_TENSOR_TRANSFORM	991
9.69.3	Typedef Documentation	991
9.69.3.1	GstTensorTransform	991
9.69.3.2	GstTensorTransformClass	991
9.69.3.3	tensor_transform_arithmetic	992
9.69.3.4	tensor_transform_clamp	992
9.69.3.5	tensor_transform_dimchg	992
9.69.3.6	tensor_transform_mode	992
9.69.3.7	tensor_transform_padding	992
9.69.3.8	tensor_transform_stand	992
9.69.3.9	tensor_transform_transpose	993
9.69.3.10	tensor_transform_typecast	993

9.69.4 Enumeration Type Documentation	993
9.69.4.1 _tensor_transform_mode	993
9.69.4.2 tensor_transform_operator	993
9.69.4.3 tensor_transform_padding_axis	994
9.69.4.4 tensor_transform_stand_mode	994
9.69.5 Function Documentation	994
9.69.5.1 gst_tensor_transform_get_type()	994
9.70 nnstreamer/elements/ignore_warning.sh File Reference	995
9.71 nnstreamer/hw_accel.c File Reference	995
9.71.1 Detailed Description	995
9.71.2 Function Documentation	996
9.71.2.1 cpu_neon_accel_available()	996
9.72 nnstreamer/hw_accel.h File Reference	996
9.72.1 Detailed Description	997
9.72.2 Function Documentation	997
9.72.2.1 cpu_neon_accel_available()	997
9.73 nnstreamer/include/nnstreamer_plugin_api.h File Reference	998
9.73.1 Detailed Description	999
9.73.2 Function Documentation	1000
9.73.2.1 gst_structure_get_media_type()	1000
9.73.2.2 gst_structure_is_tensor_stream()	1000
9.73.2.3 gst_tensor_alloc_init()	1001
9.73.2.4 gst_tensor_buffer_append_memory()	1002
9.73.2.5 gst_tensor_buffer_get_count()	1003
9.73.2.6 gst_tensor_buffer_get_nth_memory()	1004
9.73.2.7 gst_tensor_caps_can_intersect()	1006
9.73.2.8 gst_tensor_caps_from_config()	1006
9.73.2.9 gst_tensor_caps_update_dimension()	1008
9.73.2.10 gst_tensor_meta_info_append_header()	1009
9.73.2.11 gst_tensor_meta_info_parse_memory()	1009
9.73.2.12 gst_tensors_caps_from_config()	1011
9.73.2.13 gst_tensors_config_from_peer()	1012
9.73.2.14 gst_tensors_config_from_structure()	1013
9.74 nnstreamer/include/nnstreamer_plugin_api_converter.h File Reference	1015
9.74.1 Detailed Description	1017
9.74.2 Typedef Documentation	1017
9.74.2.1 NNStreamerExternalConverter	1018
9.74.3 Function Documentation	1018
9.74.3.1 nnstreamer_converter_find()	1018
9.74.3.2 nnstreamer_converter_set_custom_property_desc()	1019
9.74.3.3 registerExternalConverter()	1019
9.74.3.4 unregisterExternalConverter()	1020

9.75 nnstreamer/include/nnstreamer_plugin_api_decoder.h File Reference	1020
9.75.1 Detailed Description	1022
9.75.2 Typedef Documentation	1022
9.75.2.1 GstTensorDecoderDef	1023
9.75.3 Function Documentation	1023
9.75.3.1 nnstreamer_decoder_exit()	1023
9.75.3.2 nnstreamer_decoder_find()	1023
9.75.3.3 nnstreamer_decoder_probe()	1024
9.75.3.4 nnstreamer_decoder_set_custom_property_desc()	1025
9.76 nnstreamer/include/nnstreamer_plugin_api_filter.h File Reference	1025
9.76.1 Detailed Description	1028
9.76.2 Macro Definition Documentation	1029
9.76.2.1 ACCL_AUTO_STR	1029
9.76.2.2 ACCL_CPU_NEON_STR	1029
9.76.2.3 ACCL_CPU_SIMD_STR	1029
9.76.2.4 ACCL_CPU_STR	1029
9.76.2.5 ACCL_DEFAULT_STR	1030
9.76.2.6 ACCL_GPU_STR	1030
9.76.2.7 ACCL_NONE_STR	1030
9.76.2.8 ACCL_NPU_EDGE_TPU_STR	1030
9.76.2.9 ACCL_NPU_MOVIDIUS_STR	1030
9.76.2.10 ACCL_NPU_SLSI_STR	1030
9.76.2.11 ACCL_NPU_SR_STR	1031
9.76.2.12 ACCL_NPU_SRCN_STR	1031
9.76.2.13 ACCL_NPU_STR	1031
9.76.2.14 ACCL_NPU_VIVANTE_STR	1031
9.76.2.15 checkGstTensorFilterFrameworkVersion	1031
9.76.2.16 GST_TENSOR_FILTER_API_VERSION_DEFINED	1032
9.76.2.17 GST_TENSOR_FILTER_API_VERSION_MAX	1032
9.76.2.18 GST_TENSOR_FILTER_API_VERSION_MIN	1032
9.76.2.19 GST_TENSOR_FILTER_FRAMEWORK_BASE	1032
9.76.2.20 GST_TENSOR_FILTER_FRAMEWORK_V0	1032
9.76.2.21 GST_TENSOR_FILTER_FRAMEWORK_V1	1032
9.76.2.22 parse_accl_hw	1033
9.76.3 Typedef Documentation	1033
9.76.3.1 GstTensorFilterFramework	1033
9.76.3.2 GstTensorFilterFrameworkEventData	1033
9.76.3.3 GstTensorFilterFrameworkInfo	1033
9.76.3.4 GstTensorFilterFrameworkStatistics	1033
9.76.3.5 GstTensorFilterProperties	1034
9.76.3.6 tensors_layout	1034
9.76.4 Enumeration Type Documentation	1034

9.76.4.1	accl_hw	1034
9.76.4.2	event_ops	1035
9.76.4.3	model_info_ops	1035
9.76.5	Function Documentation	1036
9.76.5.1	get_accl_hw_str()	1036
9.76.5.2	get_accl_hw_type()	1037
9.76.5.3	nnstreamer_filter_exit()	1037
9.76.5.4	nnstreamer_filter_find()	1038
9.76.5.5	nnstreamer_filter_probe()	1038
9.76.5.6	nnstreamer_filter_set_custom_property_desc()	1039
9.76.5.7	nnstreamer_filter_shared_model_get()	1040
9.76.5.8	nnstreamer_filter_shared_model_insert_and_get()	1040
9.76.5.9	nnstreamer_filter_shared_model_remove()	1040
9.76.5.10	nnstreamer_filter_shared_model_replace()	1041
9.76.5.11	parse_accl_hw_fill()	1041
9.77	nnstreamer/include/nnstreamer_plugin_api_trainer.h File Reference	1042
9.77.1	Detailed Description	1044
9.77.2	Macro Definition Documentation	1044
9.77.2.1	GST_TENSOR_TRAINER_FRAMEWORK_BASE	1044
9.77.2.2	GST_TENSOR_TRAINER_FRAMEWORK_V1	1045
9.77.3	Typedef Documentation	1045
9.77.3.1	GstTensorTrainerEventNotifier	1045
9.77.3.2	GstTensorTrainerFramework	1045
9.77.3.3	GstTensorTrainerFrameworkInfo	1045
9.77.3.4	GstTensorTrainerProperties	1045
9.77.4	Enumeration Type Documentation	1045
9.77.4.1	GstTensorTrainerEventType	1045
9.77.5	Function Documentation	1046
9.77.5.1	nnstreamer_trainer_exit()	1046
9.77.5.2	nnstreamer_trainer_notify_event()	1047
9.77.5.3	nnstreamer_trainer_probe()	1048
9.78	nnstreamer/include/nnstreamer_plugin_api_util.h File Reference	1049
9.78.1	Detailed Description	1052
9.78.2	Macro Definition Documentation	1052
9.78.2.1	_STR_NULL	1053
9.78.2.2	gst_tensors_config_is_flexible	1053
9.78.2.3	gst_tensors_config_is_sparse	1053
9.78.2.4	gst_tensors_config_is_static	1053
9.78.3	Function Documentation	1053
9.78.3.1	find_key_strv()	1053
9.78.3.2	gst_tensor_dimension_get_min_rank()	1055
9.78.3.3	gst_tensor_dimension_get_rank()	1056

9.78.3.4 <code>gst_tensor_dimension_is_equal()</code>	1057
9.78.3.5 <code>gst_tensor_dimension_is_valid()</code>	1058
9.78.3.6 <code>gst_tensor_dimension_string_is_equal()</code>	1060
9.78.3.7 <code>gst_tensor_get_dimension_string()</code>	1061
9.78.3.8 <code>gst_tensor_get_element_count()</code>	1063
9.78.3.9 <code>gst_tensor_get_element_size()</code>	1065
9.78.3.10 <code>gst_tensor_get_format()</code>	1065
9.78.3.11 <code>gst_tensor_get_format_string()</code>	1067
9.78.3.12 <code>gst_tensor_get_rank_dimension_string()</code>	1068
9.78.3.13 <code>gst_tensor_get_type()</code>	1069
9.78.3.14 <code>gst_tensor_get_type_string()</code>	1071
9.78.3.15 <code>gst_tensor_info_convert_to_meta()</code>	1071
9.78.3.16 <code>gst_tensor_info_copy()</code>	1072
9.78.3.17 <code>gst_tensor_info_copy_n()</code>	1074
9.78.3.18 <code>gst_tensor_info_free()</code>	1076
9.78.3.19 <code>gst_tensor_info_get_rank()</code>	1077
9.78.3.20 <code>gst_tensor_info_get_size()</code>	1078
9.78.3.21 <code>gst_tensor_info_init()</code>	1079
9.78.3.22 <code>gst_tensor_info_is_equal()</code>	1080
9.78.3.23 <code>gst_tensor_info_validate()</code>	1081
9.78.3.24 <code>gst_tensor_meta_info_convert()</code>	1083
9.78.3.25 <code>gst_tensor_meta_info_get_data_size()</code>	1084
9.78.3.26 <code>gst_tensor_meta_info_get_header_size()</code>	1085
9.78.3.27 <code>gst_tensor_meta_info_get_version()</code>	1086
9.78.3.28 <code>gst_tensor_meta_info_init()</code>	1086
9.78.3.29 <code>gst_tensor_meta_info_parse_header()</code>	1087
9.78.3.30 <code>gst_tensor_meta_info_update_header()</code>	1089
9.78.3.31 <code>gst_tensor_meta_info_validate()</code>	1090
9.78.3.32 <code>gst_tensor_parse_dimension()</code>	1091
9.78.3.33 <code>gst_tensors_config_copy()</code>	1092
9.78.3.34 <code>gst_tensors_config_free()</code>	1093
9.78.3.35 <code>gst_tensors_config_init()</code>	1094
9.78.3.36 <code>gst_tensors_config_is_equal()</code>	1096
9.78.3.37 <code>gst_tensors_config_to_string()</code>	1097
9.78.3.38 <code>gst_tensors_config_validate()</code>	1098
9.78.3.39 <code>gst_tensors_info_copy()</code>	1100
9.78.3.40 <code>gst_tensors_info_free()</code>	1101
9.78.3.41 <code>gst_tensors_info_get_dimensions_string()</code>	1103
9.78.3.42 <code>gst_tensors_info_get_names_string()</code>	1105
9.78.3.43 <code>gst_tensors_info_get_nth_info()</code>	1105
9.78.3.44 <code>gst_tensors_info_get_rank_dimensions_string()</code>	1106
9.78.3.45 <code>gst_tensors_info_get_size()</code>	1107

9.78.3.46	gst_tensors_info_get_types_string()	1108
9.78.3.47	gst_tensors_info_init()	1109
9.78.3.48	gst_tensors_info_is_equal()	1111
9.78.3.49	gst_tensors_info_parse_dimensions_string()	1112
9.78.3.50	gst_tensors_info_parse_names_string()	1114
9.78.3.51	gst_tensors_info_parse_types_string()	1115
9.78.3.52	gst_tensors_info_to_string()	1117
9.78.3.53	gst_tensors_info_validate()	1118
9.78.3.54	nnstreamer_version_fetch()	1119
9.78.3.55	nnstreamer_version_string()	1120
9.79	nnstreamer/include/nnstreamer_util.h File Reference	1120
9.79.1	Detailed Description	1121
9.79.2	Macro Definition Documentation	1121
9.79.2.1	_g_memdup	1121
9.79.2.2	UNUSED	1121
9.80	nnstreamer/include/tensor_converter_custom.h File Reference	1122
9.80.1	Detailed Description	1123
9.80.2	Function Documentation	1123
9.80.2.1	nnstreamer_converter_custom_register()	1123
9.80.2.2	nnstreamer_converter_custom_unregister()	1124
9.80.3	Variable Documentation	1125
9.80.3.1	tensor_converter_custom	1125
9.81	nnstreamer/include/tensor_decoder_custom.h File Reference	1125
9.81.1	Detailed Description	1127
9.81.2	Function Documentation	1127
9.81.2.1	nnstreamer_decoder_custom_register()	1127
9.81.2.2	nnstreamer_decoder_custom_unregister()	1128
9.81.3	Variable Documentation	1129
9.81.3.1	tensor_decoder_custom	1129
9.82	nnstreamer/include/tensor_filter_custom.h File Reference	1129
9.82.1	Detailed Description	1131
9.82.2	Typedef Documentation	1132
9.82.2.1	NNS_custom_allocate_invoke	1132
9.82.2.2	NNS_custom_destroy_notify	1133
9.82.2.3	NNS_custom_exit_func	1133
9.82.2.4	NNS_custom_get_input_dimension	1133
9.82.2.5	NNS_custom_get_output_dimension	1134
9.82.2.6	NNS_custom_init_func	1134
9.82.2.7	NNS_custom_invoke	1134
9.82.2.8	NNS_custom_set_input_dimension	1135
9.82.2.9	NNStreamer_custom_class	1135
9.82.3	Variable Documentation	1135

9.82.3.1 NNSreamer_custom	1135
9.83 nnstreamer/include/tensor_filter_custom_easy.h File Reference	1136
9.83.1 Detailed Description	1137
9.83.2 Typedef Documentation	1138
9.83.2.1 NNS_custom_invoke_dynamic	1138
9.83.3 Function Documentation	1139
9.83.3.1 NNS_custom_easy_dynamic_register()	1139
9.83.3.2 NNS_custom_easy_register()	1140
9.83.3.3 NNS_custom_easy_unregister()	1141
9.84 nnstreamer/include/tensor_if.h File Reference	1142
9.84.1 Detailed Description	1143
9.84.2 Function Documentation	1144
9.84.2.1 nnstreamer_if_custom_register()	1144
9.84.2.2 nnstreamer_if_custom_unregister()	1144
9.84.3 Variable Documentation	1145
9.84.3.1 tensor_if_custom	1145
9.85 nnstreamer/include/tensor_typedef.h File Reference	1146
9.85.1 Detailed Description	1148
9.85.2 Macro Definition Documentation	1149
9.85.2.1 GST_TENSOR_CAP_DEFAULT	1149
9.85.2.2 GST_TENSOR_FORMAT_ALL	1149
9.85.2.3 GST_TENSOR_NUM_TENSORS_RANGE	1149
9.85.2.4 GST_TENSOR_RATE_RANGE	1150
9.85.2.5 GST_TENSOR_TYPE_ALL	1150
9.85.2.6 GST_TENSORS_CAP_DEFAULT	1150
9.85.2.7 GST_TENSORS_CAP_MAKE	1150
9.85.2.8 GST_TENSORS_CAP_WITH_NUM	1151
9.85.2.9 GST_TENSORS_FLEX_CAP_DEFAULT	1151
9.85.2.10 GST_TENSORS_SPARSE_CAP_DEFAULT	1151
9.85.2.11 NNS_MIMETYPE_TENSOR	1151
9.85.2.12 NNS_MIMETYPE_TENSORS	1152
9.85.2.13 NNS_TENSOR_MEMORY_MAX	1152
9.85.2.14 NNS_TENSOR_RANK_LIMIT	1152
9.85.2.15 NNS_TENSOR_SIZE_EXTRA_LIMIT	1152
9.85.2.16 NNS_TENSOR_SIZE_LIMIT	1152
9.85.2.17 NNS_TENSOR_SIZE_LIMIT_STR	1153
9.85.3 Typedef Documentation	1153
9.85.3.1 media_type	1153
9.85.3.2 tensor_dim	1153
9.85.3.3 tensor_format	1153
9.85.3.4 tensor_layout	1153
9.85.3.5 tensor_type	1154

9.85.4 Enumeration Type Documentation	1154
9.85.4.1 <code>_nns_media_type</code>	1154
9.85.4.2 <code>_nns_tensor_layout</code>	1154
9.85.4.3 <code>_nns_tensor_type</code>	1155
9.85.4.4 <code>_tensor_format</code>	1155
9.86 <code>nnstreamer/ml_agent.c</code> File Reference	1156
9.86.1 Detailed Description	1157
9.86.2 Function Documentation	1157
9.86.2.1 <code>mlagent_get_model_path_from()</code>	1157
9.86.3 Variable Documentation	1158
9.86.3.1 <code>JSON_KEY_MODEL_PATH</code>	1158
9.86.3.2 <code>URI_KEYWORD_MODEL</code>	1158
9.86.3.3 <code>URI_SCHEME</code>	1158
9.87 <code>nnstreamer/ml_agent.h</code> File Reference	1159
9.87.1 Detailed Description	1159
9.87.2 Macro Definition Documentation	1160
9.87.2.1 <code>mlagent_get_model_path_from</code>	1160
9.88 <code>nnstreamer/nnstreamer_conf.c</code> File Reference	1160
9.88.1 Detailed Description	1162
9.88.2 Macro Definition Documentation	1163
9.88.2.1 <code>NNSTREAMER_PREFIX_CONVERTER</code>	1163
9.88.2.2 <code>NNSTREAMER_PREFIX_CUSTOMFILTERS</code>	1163
9.88.2.3 <code>NNSTREAMER_PREFIX_DECODER</code>	1163
9.88.2.4 <code>NNSTREAMER_PREFIX_FILTER</code>	1163
9.88.2.5 <code>NNSTREAMER_PREFIX_TRAINER</code>	1163
9.88.2.6 <code>STR_BOOL</code>	1163
9.88.3 Enumeration Type Documentation	1163
9.88.3.1 <code>conf_sources</code>	1163
9.88.4 Function Documentation	1164
9.88.4.1 <code>_fill_in_vstr()</code>	1164
9.88.4.2 <code>_fill_subplugin_path()</code>	1165
9.88.4.3 <code>_foreach_custom_property()</code>	1166
9.88.4.4 <code>_g_list_foreach_vstr_helper()</code>	1166
9.88.4.5 <code>_get_filenames()</code>	1167
9.88.4.6 <code>_get_subplugin_with_type()</code>	1168
9.88.4.7 <code>_parse_bool_string()</code>	1169
9.88.4.8 <code>_strdup_getenv()</code>	1169
9.88.4.9 <code>_validate_file()</code>	1170
9.88.4.10 <code>nnsconf_dump()</code>	1171
9.88.4.11 <code>nnsconf_get_custom_value_bool()</code>	1171
9.88.4.12 <code>nnsconf_get_custom_value_string()</code>	1172
9.88.4.13 <code>nnsconf_get_fullpath()</code>	1173

9.88.4.14	nnsconf_get_subplugin_info()	1174
9.88.4.15	nnsconf_get_subplugin_name_prefix()	1175
9.88.4.16	nnsconf_loadconf()	1175
9.88.4.17	nnsconf_subplugin_dump()	1176
9.88.4.18	nnsconf_validate_file()	1177
9.88.5	Variable Documentation	1178
9.88.5.1	conf	1178
9.88.5.2	custom_table	1178
9.88.5.3	NNSTREAMER_ENVVAR	1178
9.88.5.4	NNSTREAMER_PATH	1179
9.88.5.5	subplugin_prefixes	1179
9.89	nnstreamer/nnstreamer_conf.h File Reference	1179
9.89.1	Detailed Description	1181
9.89.2	Macro Definition Documentation	1182
9.89.2.1	NNSTREAMER_CONF_FILE	1182
9.89.2.2	NNSTREAMER_DEFAULT_CONF_FILE	1182
9.89.2.3	NNSTREAMER_ENVVAR_CONF_FILE	1182
9.89.2.4	NNSTREAMER_SO_FILE_EXTENSION	1182
9.89.2.5	NNSTREAMER_SYS_ROOT_PATH_PREFIX	1182
9.89.3	Enumeration Type Documentation	1182
9.89.3.1	nnsconf_type_path	1182
9.89.4	Function Documentation	1183
9.89.4.1	nnsconf_dump()	1183
9.89.4.2	nnsconf_get_custom_value_bool()	1184
9.89.4.3	nnsconf_get_custom_value_string()	1184
9.89.4.4	nnsconf_get_fullpath()	1185
9.89.4.5	nnsconf_get_subplugin_info()	1186
9.89.4.6	nnsconf_get_subplugin_name_prefix()	1187
9.89.4.7	nnsconf_loadconf()	1188
9.89.4.8	nnsconf_subplugin_dump()	1189
9.89.4.9	nnsconf_validate_file()	1189
9.90	nnstreamer/nnstreamer_internal.h File Reference	1190
9.90.1	Detailed Description	1192
9.90.2	Function Documentation	1192
9.90.2.1	gst_tensor_filter_check_hw_availability()	1192
9.90.2.2	gst_tensor_filter_detect_framework()	1193
9.90.2.3	nnsconf_get_custom_value_bool()	1194
9.90.2.4	nnsconf_get_custom_value_string()	1195
9.91	nnstreamer/nnstreamer_log.c File Reference	1196
9.91.1	Detailed Description	1197
9.91.2	Macro Definition Documentation	1197
9.91.2.1	_NNSTREAMER_ERROR_LENGTH	1197

9.91.3 Function Documentation	1197
9.91.3.1 <code>__attribute__()</code>	1197
9.91.3.2 <code>_backtrace_to_string()</code>	1198
9.91.3.3 <code>_nnstreamer_error()</code>	1198
9.91.3.4 <code>_nnstreamer_error_clean()</code>	1198
9.91.3.5 <code>G_LOCK_DEFINE_STATIC()</code>	1199
9.91.4 Variable Documentation	1199
9.91.4.1 <code>errmsg</code>	1199
9.91.4.2 <code>errmsg_reported</code>	1199
9.92 <code>nnstreamer/nnstreamer_log.h</code> File Reference	1199
9.92.1 Detailed Description	1200
9.92.2 Macro Definition Documentation	1201
9.92.2.1 <code>GST_ELEMENT_ERROR_BTRACE</code>	1201
9.92.2.2 <code>ml_log_stacktrace</code>	1201
9.92.2.3 <code>ml_logd</code>	1201
9.92.2.4 <code>ml_loge</code>	1202
9.92.2.5 <code>ml_loge_stacktrace</code>	1202
9.92.2.6 <code>ml_logf</code>	1202
9.92.2.7 <code>ml_logf_stacktrace</code>	1202
9.92.2.8 <code>ml_logi</code>	1202
9.92.2.9 <code>ml_logw</code>	1203
9.92.2.10 <code>nns_logd</code>	1203
9.92.2.11 <code>nns_loge</code>	1203
9.92.2.12 <code>nns_logf</code>	1203
9.92.2.13 <code>nns_logi</code>	1203
9.92.2.14 <code>nns_logw</code>	1203
9.92.2.15 <code>TAG_NAME</code>	1204
9.92.3 Function Documentation	1204
9.92.3.1 <code>_backtrace_to_string()</code>	1204
9.92.3.2 <code>_nnstreamer_error()</code>	1204
9.92.3.3 <code>_nnstreamer_error_clean()</code>	1205
9.92.3.4 <code>_nnstreamer_error_write()</code>	1205
9.93 <code>nnstreamer/nnstreamer_plugin_api_impl.c</code> File Reference	1206
9.93.1 Detailed Description	1208
9.93.2 Macro Definition Documentation	1209
9.93.2.1 <code>AGGREGATION_DEFAULT_KEY</code>	1209
9.93.2.2 <code>NNS_TENSOR_EXTRA_MAGIC</code>	1209
9.93.2.3 <code>NNS_TENSOR_RANK_LIMIT_PREV</code>	1209
9.93.3 Function Documentation	1209
9.93.3.1 <code>_append_prev_caps()</code>	1210
9.93.3.2 <code>_get_flexible_caps()</code>	1211
9.93.3.3 <code>_get_tensor_caps()</code>	1211

9.93.3.4	_get_tensors_caps()	1212
9.93.3.5	_gst_tensor_time_sync_buffer_update()	1213
9.93.3.6	_gst_tensor_time_sync_is_eos()	1214
9.93.3.7	_is_structure_dimension_same()	1215
9.93.3.8	gst_memory_map_is_extra_tensor()	1215
9.93.3.9	gst_structure_get_media_type()	1216
9.93.3.10	gst_structure_is_tensor_stream()	1217
9.93.3.11	gst_tensor_aggregation_add_data()	1218
9.93.3.12	gst_tensor_aggregation_clear()	1218
9.93.3.13	gst_tensor_aggregation_clear_all()	1219
9.93.3.14	gst_tensor_aggregation_clear_internal()	1220
9.93.3.15	gst_tensor_aggregation_free_data()	1221
9.93.3.16	gst_tensor_aggregation_get_adapter()	1221
9.93.3.17	gst_tensor_aggregation_get_data()	1222
9.93.3.18	gst_tensor_aggregation_init()	1223
9.93.3.19	gst_tensor_buffer_append_memory()	1223
9.93.3.20	gst_tensor_buffer_from_config()	1225
9.93.3.21	gst_tensor_buffer_get_count()	1226
9.93.3.22	gst_tensor_buffer_get_nth_memory()	1227
9.93.3.23	gst_tensor_caps_can_intersect()	1228
9.93.3.24	gst_tensor_caps_from_config()	1229
9.93.3.25	gst_tensor_caps_update_dimension()	1230
9.93.3.26	gst_tensor_extra_info_init()	1231
9.93.3.27	gst_tensor_meta_info_append_header()	1232
9.93.3.28	gst_tensor_meta_info_parse_memory()	1233
9.93.3.29	gst_tensor_pad_caps_from_config()	1234
9.93.3.30	gst_tensor_pad_get_format()	1236
9.93.3.31	gst_tensor_pad_possible_caps_from_config()	1237
9.93.3.32	gst_tensor_parse_config_file()	1238
9.93.3.33	gst_tensor_time_sync_buffer_from_collectpad()	1239
9.93.3.34	gst_tensor_time_sync_flush()	1240
9.93.3.35	gst_tensor_time_sync_get_current_time()	1241
9.93.3.36	gst_tensor_time_sync_get_mode()	1242
9.93.3.37	gst_tensor_time_sync_get_mode_string()	1243
9.93.3.38	gst_tensor_time_sync_set_option_data()	1244
9.93.3.39	gst_tensors_caps_from_config()	1244
9.93.3.40	gst_tensors_config_from_peer()	1245
9.93.3.41	gst_tensors_config_from_structure()	1246
9.93.3.42	set_property_value()	1248
9.93.4	Variable Documentation	1249
9.93.4.1	gst_tensor_time_sync_mode_string	1249
9.94	nnstreamer/nnstreamer_plugin_api_util_impl.c File Reference	1250

9.94.1 Detailed Description	1253
9.94.2 Macro Definition Documentation	1253
9.94.2.1 GST_TENSOR_META_IS_V1	1254
9.94.2.2 GST_TENSOR_META_IS_VALID	1254
9.94.2.3 GST_TENSOR_META_MAGIC	1254
9.94.2.4 GST_TENSOR_META_MAGIC_VALID	1254
9.94.2.5 GST_TENSOR_META_MAKE_VERSION	1254
9.94.2.6 GST_TENSOR_META_VERSION	1255
9.94.2.7 GST_TENSOR_META_VERSION_VALID	1255
9.94.3 Function Documentation	1255
9.94.3.1 _compare_rate()	1255
9.94.3.2 _gcd()	1256
9.94.3.3 find_key_strv()	1256
9.94.3.4 gst_tensor_dimension_get_min_rank()	1257
9.94.3.5 gst_tensor_dimension_get_rank()	1259
9.94.3.6 gst_tensor_dimension_is_equal()	1259
9.94.3.7 gst_tensor_dimension_is_valid()	1260
9.94.3.8 gst_tensor_dimension_string_is_equal()	1262
9.94.3.9 gst_tensor_get_dimension_string()	1263
9.94.3.10 gst_tensor_get_element_count()	1265
9.94.3.11 gst_tensor_get_element_size()	1267
9.94.3.12 gst_tensor_get_format()	1267
9.94.3.13 gst_tensor_get_format_string()	1269
9.94.3.14 gst_tensor_get_rank_dimension_string()	1270
9.94.3.15 gst_tensor_get_type()	1271
9.94.3.16 gst_tensor_get_type_string()	1273
9.94.3.17 gst_tensor_info_convert_to_meta()	1273
9.94.3.18 gst_tensor_info_copy()	1274
9.94.3.19 gst_tensor_info_copy_n()	1276
9.94.3.20 gst_tensor_info_free()	1278
9.94.3.21 gst_tensor_info_get_rank()	1279
9.94.3.22 gst_tensor_info_get_size()	1280
9.94.3.23 gst_tensor_info_init()	1281
9.94.3.24 gst_tensor_info_is_equal()	1282
9.94.3.25 gst_tensor_info_validate()	1283
9.94.3.26 gst_tensor_meta_info_convert()	1285
9.94.3.27 gst_tensor_meta_info_get_data_size()	1286
9.94.3.28 gst_tensor_meta_info_get_header_size()	1287
9.94.3.29 gst_tensor_meta_info_get_version()	1288
9.94.3.30 gst_tensor_meta_info_init()	1288
9.94.3.31 gst_tensor_meta_info_parse_header()	1289
9.94.3.32 gst_tensor_meta_info_update_header()	1291

9.94.3.33	gst_tensor_meta_info_validate()	1292
9.94.3.34	gst_tensor_parse_dimension()	1293
9.94.3.35	gst_tensors_config_copy()	1294
9.94.3.36	gst_tensors_config_free()	1295
9.94.3.37	gst_tensors_config_init()	1296
9.94.3.38	gst_tensors_config_is_equal()	1298
9.94.3.39	gst_tensors_config_to_string()	1299
9.94.3.40	gst_tensors_config_validate()	1300
9.94.3.41	gst_tensors_info_copy()	1302
9.94.3.42	gst_tensors_info_free()	1302
9.94.3.43	gst_tensors_info_get_dimensions_string()	1304
9.94.3.44	gst_tensors_info_get_names_string()	1306
9.94.3.45	gst_tensors_info_get_nth_info()	1306
9.94.3.46	gst_tensors_info_get_rank_dimensions_string()	1307
9.94.3.47	gst_tensors_info_get_size()	1308
9.94.3.48	gst_tensors_info_get_types_string()	1309
9.94.3.49	gst_tensors_info_init()	1310
9.94.3.50	gst_tensors_info_is_equal()	1312
9.94.3.51	gst_tensors_info_parse_dimensions_string()	1313
9.94.3.52	gst_tensors_info_parse_names_string()	1315
9.94.3.53	gst_tensors_info_parse_types_string()	1316
9.94.3.54	gst_tensors_info_to_string()	1318
9.94.3.55	gst_tensors_info_validate()	1319
9.94.3.56	nnstreamer_version_fetch()	1320
9.94.3.57	nnstreamer_version_string()	1321
9.94.4	Variable Documentation	1321
9.94.4.1	tensor_element_size	1321
9.94.4.2	tensor_element_typename	1322
9.94.4.3	tensor_format_name	1322
9.95	nnstreamer/nnstreamer_subplugin.c File Reference	1322
9.95.1	Detailed Description	1324
9.95.2	Enumeration Type Documentation	1324
9.95.2.1	subpluginSearchLogic	1324
9.95.3	Function Documentation	1324
9.95.3.1	_close_handle()	1325
9.95.3.2	_get_subplugin_data()	1325
9.95.3.3	_search_subplugin()	1326
9.95.3.4	_spdata_destroy()	1326
9.95.3.5	fini_subplugin()	1327
9.95.3.6	G_LOCK_DEFINE_STATIC()	1327
9.95.3.7	get_all_subplugins()	1328
9.95.3.8	get_subplugin()	1328

9.95.3.9	<code>init_subplugin()</code>	1329
9.95.3.10	<code>register_subplugin()</code>	1330
9.95.3.11	<code>subplugin_get_custom_property_desc()</code>	1331
9.95.3.12	<code>subplugin_set_custom_property_desc()</code>	1332
9.95.3.13	<code>unregister_subplugin()</code>	1333
9.95.4	Variable Documentation	1334
9.95.4.1	<code>handles</code>	1334
9.95.4.2	<code>searchAlgorithm</code>	1334
9.95.4.3	<code>subpluginData</code>	1334
9.95.4.4	<code>subplugins</code>	1334
9.96	<code>nnstreamer/nnstreamer_subplugin.h</code> File Reference	1335
9.96.1	Detailed Description	1336
9.96.2	Macro Definition Documentation	1337
9.96.2.1	<code>NNS_SUBPLUGIN_CHECKER</code>	1337
9.96.3	Enumeration Type Documentation	1337
9.96.3.1	<code>subpluginType</code>	1337
9.96.4	Function Documentation	1337
9.96.4.1	<code>get_all_subplugins()</code>	1337
9.96.4.2	<code>get_subplugin()</code>	1338
9.96.4.3	<code>register_subplugin()</code>	1339
9.96.4.4	<code>subplugin_get_custom_property_desc()</code>	1341
9.96.4.5	<code>subplugin_set_custom_property_desc()</code>	1342
9.96.4.6	<code>unregister_subplugin()</code>	1343
9.97	<code>nnstreamer/nnstreamer_watchdog.c</code> File Reference	1344
9.97.1	Detailed Description	1345
9.97.2	Typedef Documentation	1345
9.97.2.1	<code>NnstWatchdog</code>	1346
9.97.3	Function Documentation	1346
9.97.3.1	<code>_loop_running_cb()</code>	1346
9.97.3.2	<code>_nnstreamer_watchdog_thread()</code>	1346
9.97.3.3	<code>nnstreamer_watchdog_create()</code>	1347
9.97.3.4	<code>nnstreamer_watchdog_destroy()</code>	1347
9.97.3.5	<code>nnstreamer_watchdog_feed()</code>	1348
9.97.3.6	<code>nnstreamer_watchdog_release()</code>	1349
9.98	<code>nnstreamer/nnstreamer_watchdog.h</code> File Reference	1349
9.98.1	Detailed Description	1350
9.98.2	Typedef Documentation	1351
9.98.2.1	<code>nns_watchdog_h</code>	1351
9.98.3	Function Documentation	1351
9.98.3.1	<code>nnstreamer_watchdog_create()</code>	1351
9.98.3.2	<code>nnstreamer_watchdog_destroy()</code>	1352
9.98.3.3	<code>nnstreamer_watchdog_feed()</code>	1352

9.98.3.4 nstreamer_watchdog_release()	1353
9.99 nstreamer/registerer/nstreamer.c File Reference	1353
9.99.1 Detailed Description	1354
9.99.2 Macro Definition Documentation	1355
9.99.2.1 NNSTREAMER_INIT	1355
9.99.2.2 PACKAGE	1355
9.99.3 Function Documentation	1355
9.99.3.1 GST_ERROR()	1355
9.99.3.2 GST_PLUGIN_DEFINE()	1356
9.99.3.3 while()	1356
9.99.4 Variable Documentation	1356
9.99.4.1 FALSE	1356
9.100 nstreamer/tensor_allocator.c File Reference	1356
9.100.1 Detailed Description	1358
9.100.2 Macro Definition Documentation	1358
9.100.2.1 GST_TENSOR_ALLOCATOR	1358
9.100.3 Function Documentation	1358
9.100.3.1 _alloc()	1359
9.100.3.2 G_DEFINE_TYPE()	1359
9.100.3.3 gst_tensor_alloc_init()	1359
9.100.3.4 gst_tensor_allocator_class_init()	1360
9.100.3.5 gst_tensor_allocator_get_type()	1360
9.100.3.6 gst_tensor_allocator_init()	1361
9.100.4 Variable Documentation	1361
9.100.4.1 gst_tensor_allocator_alignment	1361
9.101 nstreamer/tensor_common.h File Reference	1361
9.101.1 Detailed Description	1363
9.101.2 Macro Definition Documentation	1364
9.101.2.1 gst_tensor_pad_caps_is_flexible	1364
9.101.2.2 gst_tensor_pad_caps_is_sparse	1364
9.101.2.3 gst_tensor_pad_caps_is_static	1364
9.101.2.4 nns_memcpy	1365
9.101.2.5 nns_memset	1365
9.101.2.6 silent_debug	1365
9.101.2.7 silent_debug_caps	1365
9.101.3 Typedef Documentation	1366
9.101.3.1 tensor_sync_basepad_data	1366
9.101.3.2 tensor_time_sync_data	1366
9.101.4 Enumeration Type Documentation	1366
9.101.4.1 tensor_time_sync_mode	1366
9.101.5 Function Documentation	1366
9.101.5.1 gst_tensor_aggregation_clear()	1367

9.101.5.2 <code>gst_tensor_aggregation_clear_all()</code>	1367
9.101.5.3 <code>gst_tensor_aggregation_get_adapter()</code>	1368
9.101.5.4 <code>gst_tensor_aggregation_init()</code>	1369
9.101.5.5 <code>gst_tensor_buffer_from_config()</code>	1370
9.101.5.6 <code>gst_tensor_pad_caps_from_config()</code>	1371
9.101.5.7 <code>gst_tensor_pad_get_format()</code>	1373
9.101.5.8 <code>gst_tensor_pad_possible_caps_from_config()</code>	1373
9.101.5.9 <code>gst_tensor_parse_config_file()</code>	1374
9.101.5.10 <code>gst_tensor_time_sync_buffer_from_collectpad()</code>	1375
9.101.5.11 <code>gst_tensor_time_sync_flush()</code>	1377
9.101.5.12 <code>gst_tensor_time_sync_get_current_time()</code>	1377
9.101.5.13 <code>gst_tensor_time_sync_get_mode()</code>	1378
9.101.5.14 <code>gst_tensor_time_sync_get_mode_string()</code>	1379
9.101.5.15 <code>gst_tensor_time_sync_set_option_data()</code>	1380
9.102 nnstreamer/tensor_data.c File Reference	1380
9.102.1 Detailed Description	1382
9.102.2 Macro Definition Documentation	1382
9.102.2.1 <code>td_get_data</code>	1382
9.102.2.2 <code>td_set_data</code>	1382
9.102.2.3 <code>td_typecast</code>	1383
9.102.2.4 <code>td_typecast_to</code>	1383
9.102.2.5 <code>td_typecast_to_fromf16</code>	1383
9.102.3 Function Documentation	1383
9.102.3.1 <code>gst_tensor_data_get()</code>	1383
9.102.3.2 <code>gst_tensor_data_raw_average()</code>	1384
9.102.3.3 <code>gst_tensor_data_raw_average_per_channel()</code>	1385
9.102.3.4 <code>gst_tensor_data_raw_std()</code>	1386
9.102.3.5 <code>gst_tensor_data_raw_std_per_channel()</code>	1387
9.102.3.6 <code>gst_tensor_data_raw_typecast()</code>	1388
9.102.3.7 <code>gst_tensor_data_typecast()</code>	1389
9.102.3.8 <code>while()</code>	1390
9.103 nnstreamer/tensor_data.h File Reference	1390
9.103.1 Detailed Description	1391
9.103.2 Function Documentation	1392
9.103.2.1 <code>gst_tensor_data_get()</code>	1392
9.103.2.2 <code>gst_tensor_data_raw_average()</code>	1393
9.103.2.3 <code>gst_tensor_data_raw_average_per_channel()</code>	1393
9.103.2.4 <code>gst_tensor_data_raw_std()</code>	1394
9.103.2.5 <code>gst_tensor_data_raw_std_per_channel()</code>	1395
9.103.2.6 <code>gst_tensor_data_raw_typecast()</code>	1396
9.103.2.7 <code>gst_tensor_data_set()</code>	1397
9.103.2.8 <code>gst_tensor_data_typecast()</code>	1398

9.104 nnstreamer/tensor_filter/tensor_filter.c File Reference	1399
9.104.1 Detailed Description	1402
9.104.2 Macro Definition Documentation	1402
9.104.2.1 CAPS_STRING	1402
9.104.2.2 EVENT_NAME_UPDATE_MODEL	1403
9.104.2.3 GST_CAT_DEFAULT	1403
9.104.2.4 gst_tensor_filter_parent_class	1403
9.104.2.5 LATENCY_REPORT_HEADROOM	1403
9.104.2.6 LATENCY_REPORT_THRESHOLD	1404
9.104.2.7 TF_MODELNAME	1404
9.104.2.8 THRESHOLD_CACHE_OLD	1404
9.104.2.9 THRESHOLD_DROP_OLD	1404
9.104.3 Typedef Documentation	1404
9.104.3.1 FilterTransformData	1404
9.104.4 Function Documentation	1404
9.104.4.1 _gst_tensor_filter_convert_meta()	1405
9.104.4.2 _gst_tensor_filter_release_mem_until_idx()	1405
9.104.4.3 _gst_tensor_filter_transform_check_invoke_result()	1406
9.104.4.4 _gst_tensor_filter_transform_get_all_input_data()	1406
9.104.4.5 _gst_tensor_filter_transform_get_invoke_tensors()	1407
9.104.4.6 _gst_tensor_filter_transform_get_output_data()	1407
9.104.4.7 _gst_tensor_filter_transform_prepare_output_tensors()	1408
9.104.4.8 _gst_tensor_filter_transform_update_outbuf()	1409
9.104.4.9 _gst_tensor_filter_transform_validate()	1409
9.104.4.10 accumulate_latency()	1410
9.104.4.11 G_DEFINE_TYPE()	1411
9.104.4.12 GST_DEBUG_CATEGORY_STATIC()	1411
9.104.4.13 gst_tensor_filter_check_throttling_delay()	1411
9.104.4.14 gst_tensor_filter_class_init()	1412
9.104.4.15 gst_tensor_filter_configure_tensor()	1413
9.104.4.16 gst_tensor_filter_destroy_notify()	1414
9.104.4.17 gst_tensor_filter_finalize()	1415
9.104.4.18 gst_tensor_filter_fixate_caps()	1416
9.104.4.19 gst_tensor_filter_get_property()	1417
9.104.4.20 gst_tensor_filter_get_tensor_size()	1417
9.104.4.21 gst_tensor_filter_get_wrapped_mem()	1418
9.104.4.22 gst_tensor_filter_init()	1419
9.104.4.23 gst_tensor_filter_query()	1419
9.104.4.24 gst_tensor_filter_set_caps()	1420
9.104.4.25 gst_tensor_filter_set_property()	1421
9.104.4.26 gst_tensor_filter_sink_event()	1422
9.104.4.27 gst_tensor_filter_src_event()	1423

9.104.4.28 <code>gst_tensor_filter_start()</code>	1424
9.104.4.29 <code>gst_tensor_filter_stop()</code>	1424
9.104.4.30 <code>gst_tensor_filter_transform()</code>	1425
9.104.4.31 <code>gst_tensor_filter_transform_caps()</code>	1427
9.104.4.32 <code>gst_tensor_filter_transform_size()</code>	1428
9.104.4.33 <code>gst_tensor_filter_watchdog_trigger()</code>	1429
9.104.4.34 <code>prepare_statistics()</code>	1429
9.104.4.35 <code>record_statistics()</code>	1430
9.104.4.36 <code>track_latency()</code>	1430
9.104.5 Variable Documentation	1431
9.104.5.1 <code>sink_factory</code>	1431
9.104.5.2 <code>src_factory</code>	1431
9.105 <code>nnstreamer/tensor_filter/tensor_filter.h</code> File Reference	1432
9.105.1 Detailed Description	1433
9.105.2 Macro Definition Documentation	1433
9.105.2.1 <code>GST_IS_TENSOR_FILTER</code>	1434
9.105.2.2 <code>GST_IS_TENSOR_FILTER_CLASS</code>	1434
9.105.2.3 <code>GST_TENSOR_FILTER</code>	1434
9.105.2.4 <code>GST_TENSOR_FILTER_CAST</code>	1434
9.105.2.5 <code>GST_TENSOR_FILTER_CLASS</code>	1434
9.105.2.6 <code>GST_TYPE_TENSOR_FILTER</code>	1434
9.105.3 Typedef Documentation	1435
9.105.3.1 <code>GstTensorFilter</code>	1435
9.105.3.2 <code>GstTensorFilterClass</code>	1435
9.105.4 Function Documentation	1435
9.105.4.1 <code>gst_tensor_filter_get_type()</code>	1435
9.106 <code>nnstreamer/tensor_filter/tensor_filter_common.c</code> File Reference	1435
9.106.1 Detailed Description	1440
9.106.2 Macro Definition Documentation	1440
9.106.2.1 <code>g_free_const</code>	1440
9.106.2.2 <code>g_strfreev_const</code>	1440
9.106.2.3 <code>REGEX_ACCL_DELIMITER</code>	1441
9.106.2.4 <code>REGEX_ACCL_ELEM_DELIMITER</code>	1441
9.106.2.5 <code>REGEX_ACCL_ELEM_END</code>	1441
9.106.2.6 <code>REGEX_ACCL_ELEM_PREFIX</code>	1441
9.106.2.7 <code>REGEX_ACCL_ELEM_START</code>	1441
9.106.2.8 <code>REGEX_ACCL_ELEM_SUFFIX</code>	1441
9.106.2.9 <code>REGEX_ACCL_END</code>	1442
9.106.2.10 <code>REGEX_ACCL_PREFIX</code>	1442
9.106.2.11 <code>REGEX_ACCL_START</code>	1442
9.106.2.12 <code>REGEX_ACCL_SUFFIX</code>	1442
9.106.2.13 <code>silent_debug_info</code>	1442

9.106.3 Function Documentation	1443
9.106.3.1 _detect_framework_from_config()	1443
9.106.3.2 _gtfc_setprop_ACCELERATOR()	1443
9.106.3.3 _gtfc_setprop_CUSTOM()	1444
9.106.3.4 _gtfc_setprop_DIMENSION()	1445
9.106.3.5 _gtfc_setprop_FRAMEWORK()	1445
9.106.3.6 _gtfc_setprop_INPUTCOMBINATION()	1446
9.106.3.7 _gtfc_setprop_IS_UPDATABLE()	1447
9.106.3.8 _gtfc_setprop_LATENCY()	1447
9.106.3.9 _gtfc_setprop_LAYOUT()	1448
9.106.3.10 _gtfc_setprop_MODEL()	1448
9.106.3.11 _gtfc_setprop_NAME()	1449
9.106.3.12 _gtfc_setprop_OUTPUTCOMBINATION()	1450
9.106.3.13 _gtfc_setprop_PROP_INVOKE_DYNAMIC()	1450
9.106.3.14 _gtfc_setprop_SHARED_TENSOR_FILTER_KEY()	1451
9.106.3.15 _gtfc_setprop_SUSPEND()	1451
9.106.3.16 _gtfc_setprop_THROUGHPUT()	1452
9.106.3.17 _gtfc_setprop_TYPE()	1452
9.106.3.18 accl_hw_get_type()	1453
9.106.3.19 add_basic_supported_accelerators()	1453
9.106.3.20 create_regex()	1453
9.106.3.21 filter_supported_accelerators()	1454
9.106.3.22 G_LOCK_DEFINE_STATIC()	1455
9.106.3.23 get_accl_hw_str()	1455
9.106.3.24 get_accl_hw_type()	1456
9.106.3.25 gst_tensor_filter_allocate_in_invoke()	1457
9.106.3.26 gst_tensor_filter_check_hw_availability()	1458
9.106.3.27 gst_tensor_filter_common_close_fw()	1459
9.106.3.28 gst_tensor_filter_common_free_property()	1459
9.106.3.29 gst_tensor_filter_common_get_combined_in_info()	1460
9.106.3.30 gst_tensor_filter_common_get_combined_out_info()	1461
9.106.3.31 gst_tensor_filter_common_get_out_info()	1461
9.106.3.32 gst_tensor_filter_common_get_property()	1462
9.106.3.33 gst_tensor_filter_common_init_property()	1463
9.106.3.34 gst_tensor_filter_common_open_fw()	1464
9.106.3.35 gst_tensor_filter_common_set_property()	1465
9.106.3.36 gst_tensor_filter_common_unload_fw()	1466
9.106.3.37 gst_tensor_filter_destroy_notify_util()	1466
9.106.3.38 gst_tensor_filter_detect_framework()	1467
9.106.3.39 gst_tensor_filter_framework_info_init()	1468
9.106.3.40 gst_tensor_filter_get_available_framework()	1468
9.106.3.41 gst_tensor_filter_get_dimension_string()	1469

9.106.3.42	<code>gst_tensor_filter_get_layout_string()</code>	1470
9.106.3.43	<code>gst_tensor_filter_get_name_string()</code>	1471
9.106.3.44	<code>gst_tensor_filter_get_rank_string()</code>	1472
9.106.3.45	<code>gst_tensor_filter_get_type_string()</code>	1473
9.106.3.46	<code>gst_tensor_filter_install_properties()</code>	1473
9.106.3.47	<code>gst_tensor_filter_load_tensor_info()</code>	1475
9.106.3.48	<code>gst_tensor_filter_parse_accelerator()</code>	1476
9.106.3.49	<code>gst_tensor_filter_parse_modelpaths_string()</code>	1476
9.106.3.50	<code>gst_tensor_filter_properties_init()</code>	1477
9.106.3.51	<code>gst_tensor_filter_property_to_string()</code>	1478
9.106.3.52	<code>gst_tensor_filter_statistics_init()</code>	1478
9.106.3.53	<code>gst_tensor_get_layout_string()</code>	1478
9.106.3.54	<code>gst_tensor_parse_layout_string()</code>	1479
9.106.3.55	<code>gst_tensors_layout_init()</code>	1480
9.106.3.56	<code>gst_tensors_parse_layouts_string()</code>	1480
9.106.3.57	<code>gst_tensors_rank_init()</code>	1481
9.106.3.58	<code>gst_tensorsinfo_compare_print()</code>	1481
9.106.3.59	<code>gst_tensorsinfo_compare_to_string()</code>	1482
9.106.3.60	<code>nnstreamer_filter_exit()</code>	1483
9.106.3.61	<code>nnstreamer_filter_find()</code>	1484
9.106.3.62	<code>nnstreamer_filter_find_best_fit()</code>	1484
9.106.3.63	<code>nnstreamer_filter_probe()</code>	1485
9.106.3.64	<code>nnstreamer_filter_set_custom_property_desc()</code>	1486
9.106.3.65	<code>nnstreamer_filter_shared_model_get()</code>	1486
9.106.3.66	<code>nnstreamer_filter_shared_model_insert_and_get()</code>	1487
9.106.3.67	<code>nnstreamer_filter_shared_model_remove()</code>	1487
9.106.3.68	<code>nnstreamer_filter_shared_model_replace()</code>	1488
9.106.3.69	<code>nnstreamer_filter_validate()</code>	1488
9.106.3.70	<code>parse_accl_hw_all()</code>	1489
9.106.3.71	<code>parse_accl_hw_fill()</code>	1490
9.106.3.72	<code>parse_accl_hw_util()</code>	1491
9.106.3.73	<code>runtime_check_supported_accelerator()</code>	1492
9.106.3.74	<code>strcpy2()</code>	1493
9.106.3.75	<code>verify_model_path()</code>	1494
9.106.4	Variable Documentation	1494
9.106.4.1	<code>regex_accl_elem_utils</code>	1495
9.106.4.2	<code>regex_accl_utils</code>	1495
9.106.4.3	<code>shared_model_table</code>	1495
9.107	<code>nnstreamer/tensor_filter/tensor_filter_common.h</code> File Reference	1495
9.107.1	Detailed Description	1498
9.107.2	Macro Definition Documentation	1498
9.107.2.1	DBG	1499

9.107.2.2	gst_tensor_filter_v0_call	1499
9.107.2.3	gst_tensor_filter_v1_call	1499
9.107.2.4	GST_TF_FW_INVOKE_COMPAT	1500
9.107.2.5	GST_TF_FW_V0	1500
9.107.2.6	GST_TF_FW_V1	1500
9.107.2.7	GST_TF_FW_VN	1500
9.107.2.8	GST_TF_STAT_MAX_RECENT	1501
9.107.3	Typedef Documentation	1501
9.107.3.1	GstTensorFilterCombination	1501
9.107.3.2	GstTensorFilterPrivate	1501
9.107.3.3	GstTensorFilterStatistics	1501
9.107.4	Enumeration Type Documentation	1501
9.107.4.1	anonymous enum	1502
9.107.5	Function Documentation	1502
9.107.5.1	gst_tensor_filter_allocate_in_invoke()	1502
9.107.5.2	gst_tensor_filter_check_hw_availability()	1503
9.107.5.3	gst_tensor_filter_common_close_fw()	1504
9.107.5.4	gst_tensor_filter_common_free_property()	1505
9.107.5.5	gst_tensor_filter_common_get_combined_in_info()	1505
9.107.5.6	gst_tensor_filter_common_get_combined_out_info()	1506
9.107.5.7	gst_tensor_filter_common_get_out_info()	1507
9.107.5.8	gst_tensor_filter_common_get_property()	1508
9.107.5.9	gst_tensor_filter_common_init_property()	1509
9.107.5.10	gst_tensor_filter_common_open_fw()	1509
9.107.5.11	gst_tensor_filter_common_set_property()	1510
9.107.5.12	gst_tensor_filter_common_unload_fw()	1511
9.107.5.13	gst_tensor_filter_destroy_notify_util()	1512
9.107.5.14	gst_tensor_filter_detect_framework()	1512
9.107.5.15	gst_tensor_filter_install_properties()	1513
9.107.5.16	gst_tensor_filter_load_tensor_info()	1515
9.107.5.17	gst_tensorsinfo_compare_print()	1516
9.107.5.18	gst_tensorsinfo_compare_to_string()	1516
9.108	nstreamer/tensor_filter/tensor_filter_custom.c File Reference	1517
9.108.1	Detailed Description	1519
9.108.2	Typedef Documentation	1519
9.108.2.1	internal_data	1520
9.108.3	Function Documentation	1520
9.108.3.1	custom_allocateInInvoke()	1520
9.108.3.2	custom_checkAvailability()	1520
9.108.3.3	custom_close()	1521
9.108.3.4	custom_destroyNotify()	1521
9.108.3.5	custom_getInputDim()	1522

9.108.3.6 custom_getOutputDim()	1522
9.108.3.7 custom_invoke()	1522
9.108.3.8 custom_loadlib()	1523
9.108.3.9 custom_open()	1524
9.108.3.10 custom_setInputDim()	1524
9.108.3.11 fini_filter_custom()	1524
9.108.3.12 init_filter_custom()	1525
9.108.4 Variable Documentation	1525
9.108.4.1 filter_subplugin_custom	1525
9.108.4.2 NNS_support_custom	1525
9.109 nnstreamer/tensor_filter/tensor_filter_custom_easy.c File Reference	1526
9.109.1 Detailed Description	1527
9.109.2 Function Documentation	1528
9.109.2.1 custom_close()	1528
9.109.2.2 custom_eventHandler()	1528
9.109.2.3 custom_free_internal_data()	1529
9.109.2.4 custom_getFrameworkInfo()	1529
9.109.2.5 custom_getModellInfo()	1530
9.109.2.6 custom_invoke()	1530
9.109.2.7 custom_open()	1531
9.109.2.8 fini_filter_custom_easy()	1531
9.109.2.9 init_filter_custom_easy()	1532
9.109.2.10 NNS_custom_easy_dynamic_register()	1532
9.109.2.11 NNS_custom_easy_register()	1533
9.109.2.12 NNS_custom_easy_unregister()	1533
9.109.3 Variable Documentation	1534
9.109.3.1 internal_data	1534
9.109.3.2 NNS_support_custom_easy	1534
9.110 nnstreamer/tensor_filter/tensor_filter_single.c File Reference	1535
9.110.1 Detailed Description	1536
9.110.2 Macro Definition Documentation	1537
9.110.2.1 g_tensor_filter_single_parent_class	1537
9.110.2.2 G_TENSOR_FILTER_SINGLE_PRIV	1537
9.110.3 Typedef Documentation	1537
9.110.3.1 GTensorFilterSinglePrivate	1537
9.110.4 Function Documentation	1537
9.110.4.1 G_DEFINE_TYPE_WITH_PRIVATE()	1537
9.110.4.2 g_tensor_filter_allocate_in_invoke()	1538
9.110.4.3 g_tensor_filter_destroy_notify()	1538
9.110.4.4 g_tensor_filter_input_configured()	1539
9.110.4.5 g_tensor_filter_output_configured()	1539
9.110.4.6 g_tensor_filter_set_input_info()	1540

9.110.4.7 g_tensor_filter_single_class_init()	1541
9.110.4.8 g_tensor_filter_single_finalize()	1541
9.110.4.9 g_tensor_filter_single_get_property()	1542
9.110.4.10 g_tensor_filter_single_init()	1543
9.110.4.11 g_tensor_filter_single_invoke()	1543
9.110.4.12 g_tensor_filter_single_set_property()	1545
9.110.4.13 g_tensor_filter_single_start()	1545
9.110.4.14 g_tensor_filter_single_stop()	1546
9.111 nntstreamer/tensor_filter/tensor_filter_single.h File Reference	1547
9.111.1 Detailed Description	1549
9.111.2 Macro Definition Documentation	1549
9.111.2.1 G_IS_TENSOR_FILTER_SINGLE	1549
9.111.2.2 G_IS_TENSOR_FILTER_SINGLE_CLASS	1549
9.111.2.3 G_TENSOR_FILTER_SINGLE	1550
9.111.2.4 G_TENSOR_FILTER_SINGLE_CAST	1550
9.111.2.5 G_TENSOR_FILTER_SINGLE_CLASS	1550
9.111.2.6 G_TYPE_TENSOR_FILTER_SINGLE	1550
9.111.3 Typedef Documentation	1550
9.111.3.1 GTensorFilterSingle	1550
9.111.3.2 GTensorFilterSingleClass	1551
9.111.4 Function Documentation	1551
9.111.4.1 g_tensor_filter_single_get_type()	1551
9.112 nntstreamer/tensor_filter/tensor_filter_support_cc.cc File Reference	1551
9.112.1 Detailed Description	1552
9.112.2 Macro Definition Documentation	1552
9.112.2.1 _RETURN_ERR_WITH_MSG	1552
9.112.2.2 _SANITY_CHECK	1553
9.112.2.3 GET_TFSP_WITH_CHECKS	1553
9.112.2.4 NO_ANONYMOUS_NESTED_STRUCT	1553
9.113 nntstreamer/tensor_meta.c File Reference	1553
9.113.1 Detailed Description	1554
9.113.2 Function Documentation	1555
9.113.2.1 gst_meta_query_api_get_type()	1555
9.113.2.2 gst_meta_query_free()	1555
9.113.2.3 gst_meta_query_get_info()	1555
9.113.2.4 gst_meta_query_init()	1556
9.113.2.5 gst_meta_query_transform()	1556
9.114 nntstreamer/tensor_meta.h File Reference	1557
9.114.1 Detailed Description	1558
9.114.2 Macro Definition Documentation	1558
9.114.2.1 gst_buffer_add_meta_query	1559
9.114.2.2 gst_buffer_get_meta_query	1559

9.114.2.3 GST_META_QUERY_API_TYPE	1559
9.114.2.4 GST_META_QUERY_INFO	1559
9.114.3 Function Documentation	1559
9.114.3.1 gst_meta_query_api_get_type()	1559
9.114.3.2 gst_meta_query_get_info()	1560
9.114.4 Variable Documentation	1560
9.114.4.1 query_client_id_t	1560
9.115 nnsreamer/tensor_query/tensor_query_client.c File Reference	1560
9.115.1 Detailed Description	1562
9.115.2 Macro Definition Documentation	1563
9.115.2.1 DBG	1563
9.115.2.2 DEFAULT_CLIENT_TIMEOUT	1563
9.115.2.3 DEFAULT_MAX_REQUEST	1563
9.115.2.4 DEFAULT_SILENT	1563
9.115.2.5 GST_CAT_DEFAULT	1563
9.115.2.6 gst_tensor_query_client_parent_class	1564
9.115.2.7 TCP_DEFAULT_CLIENT_SRC_PORT	1564
9.115.2.8 TCP_DEFAULT_HOST	1564
9.115.2.9 TCP_DEFAULT_SRV_SRC_PORT	1564
9.115.2.10 TCP_HIGHEST_PORT	1564
9.115.3 Enumeration Type Documentation	1564
9.115.3.1 anonymous enum	1564
9.115.4 Function Documentation	1565
9.115.4.1 _nns_edge_event_cb()	1565
9.115.4.2 _nns_edge_parse_caps()	1566
9.115.4.3 G_DEFINE_TYPE()	1566
9.115.4.4 GST_DEBUG_CATEGORY_STATIC()	1567
9.115.4.5 gst_tensor_query_client_chain()	1567
9.115.4.6 gst_tensor_query_client_class_init()	1568
9.115.4.7 gst_tensor_query_client_create_edge_handle()	1568
9.115.4.8 gst_tensor_query_client_finalize()	1569
9.115.4.9 gst_tensor_query_client_get_property()	1570
9.115.4.10 gst_tensor_query_client_init()	1570
9.115.4.11 gst_tensor_query_client_query_caps()	1571
9.115.4.12 gst_tensor_query_client_set_property()	1572
9.115.4.13 gst_tensor_query_client_sink_event()	1572
9.115.4.14 gst_tensor_query_client_sink_query()	1573
9.115.4.15 gst_tensor_query_client_update_caps()	1574
9.115.5 Variable Documentation	1575
9.115.5.1 sinktemplate	1575
9.115.5.2 srctemplate	1575
9.116 nnsreamer/tensor_query/tensor_query_client.h File Reference	1575

9.116.1 Detailed Description	1577
9.116.2 Macro Definition Documentation	1577
9.116.2.1 GST_IS_TENSOR_QUERY_CLIENT	1577
9.116.2.2 GST_IS_TENSOR_QUERY_CLIENT_CLASS	1577
9.116.2.3 GST_TENSOR_QUERY_CLIENT	1577
9.116.2.4 GST_TENSOR_QUERY_CLIENT_CAST	1578
9.116.2.5 GST_TENSOR_QUERY_CLIENT_CLASS	1578
9.116.2.6 GST_TYPE_TENSOR_QUERY_CLIENT	1578
9.116.3 Typedef Documentation	1578
9.116.3.1 GstTensorQueryClient	1578
9.116.3.2 GstTensorQueryClientClass	1578
9.116.4 Function Documentation	1578
9.116.4.1 gst_tensor_query_client_get_type()	1579
9.117 nnstreamer/tensor_query/tensor_query_common.c File Reference	1579
9.117.1 Detailed Description	1579
9.117.2 Macro Definition Documentation	1580
9.117.2.1 EREMOTEIO	1580
9.117.3 Function Documentation	1580
9.117.3.1 gst_tensor_query_get_connect_type()	1580
9.118 nnstreamer/tensor_query/tensor_query_common.h File Reference	1580
9.118.1 Detailed Description	1581
9.118.2 Macro Definition Documentation	1582
9.118.2.1 DEFAULT_CONNECT_TYPE	1582
9.118.2.2 DEFAULT_HOST	1582
9.118.2.3 GST_TYPE_QUERY_CONNECT_TYPE	1582
9.118.2.4 QUERY_DEFAULT_TIMEOUT_SEC	1582
9.118.3 Function Documentation	1582
9.118.3.1 gst_tensor_query_get_connect_type()	1582
9.119 nnstreamer/tensor_query/tensor_query_server.c File Reference	1583
9.119.1 Detailed Description	1584
9.119.2 Function Documentation	1584
9.119.2.1 fini_queryserver()	1584
9.119.2.2 G_LOCK_DEFINE_STATIC()	1585
9.119.2.3 gst_tensor_query_server_add_data()	1585
9.119.2.4 gst_tensor_query_server_get_handle()	1586
9.119.2.5 gst_tensor_query_server_prepare()	1586
9.119.2.6 gst_tensor_query_server_release_edge_handle()	1587
9.119.2.7 gst_tensor_query_server_remove_data()	1588
9.119.2.8 gst_tensor_query_server_send_buffer()	1588
9.119.2.9 gst_tensor_query_server_set_caps()	1589
9.119.2.10 gst_tensor_query_server_set_configured()	1589
9.119.2.11 gst_tensor_query_server_wait_sink()	1590

9.119.2.12 <code>init_queryserver()</code>	1591
9.119.3 Variable Documentation	1591
9.119.3.1 <code>_qs_table</code>	1591
9.120 <code>nnstreamer/tensor_query/tensor_query_server.h</code> File Reference	1591
9.120.1 Detailed Description	1593
9.120.2 Macro Definition Documentation	1593
9.120.2.1 <code>DEFAULT_QUERY_INFO_TIMEOUT</code>	1593
9.120.2.2 <code>DEFAULT_SERVER_ID</code>	1593
9.120.3 Function Documentation	1593
9.120.3.1 <code>gst_tensor_query_server_add_data()</code>	1594
9.120.3.2 <code>gst_tensor_query_server_prepare()</code>	1594
9.120.3.3 <code>gst_tensor_query_server_release_edge_handle()</code>	1595
9.120.3.4 <code>gst_tensor_query_server_remove_data()</code>	1596
9.120.3.5 <code>gst_tensor_query_server_send_buffer()</code>	1596
9.120.3.6 <code>gst_tensor_query_server_set_caps()</code>	1597
9.120.3.7 <code>gst_tensor_query_server_set_configured()</code>	1598
9.120.3.8 <code>gst_tensor_query_server_wait_sink()</code>	1599
9.121 <code>nnstreamer/tensor_query/tensor_query_serversink.c</code> File Reference	1599
9.121.1 Detailed Description	1601
9.121.2 Macro Definition Documentation	1601
9.121.2.1 <code>DEFAULT_METALESS_FRAME_LIMIT</code>	1601
9.121.2.2 <code>GST_CAT_DEFAULT</code>	1601
9.121.2.3 <code>gst_tensor_query_serversink_parent_class</code>	1601
9.121.3 Enumeration Type Documentation	1601
9.121.3.1 anonymous enum	1601
9.121.4 Function Documentation	1602
9.121.4.1 <code>_gst_tensor_query_serversink_playing()</code>	1602
9.121.4.2 <code>_gst_tensor_query_serversink_start()</code>	1603
9.121.4.3 <code>G_DEFINE_TYPE()</code>	1603
9.121.4.4 <code>GST_DEBUG_CATEGORY_STATIC()</code>	1603
9.121.4.5 <code>gst_tensor_query_serversink_change_state()</code>	1604
9.121.4.6 <code>gst_tensor_query_serversink_class_init()</code>	1604
9.121.4.7 <code>gst_tensor_query_serversink_finalize()</code>	1605
9.121.4.8 <code>gst_tensor_query_serversink_get_property()</code>	1606
9.121.4.9 <code>gst_tensor_query_serversink_init()</code>	1606
9.121.4.10 <code>gst_tensor_query_serversink_render()</code>	1606
9.121.4.11 <code>gst_tensor_query_serversink_set_caps()</code>	1607
9.121.4.12 <code>gst_tensor_query_serversink_set_property()</code>	1608
9.121.5 Variable Documentation	1608
9.121.5.1 <code>sinktemplate</code>	1608
9.122 <code>nnstreamer/tensor_query/tensor_query_serversink.h</code> File Reference	1608
9.122.1 Detailed Description	1610

9.122.2 Macro Definition Documentation	1610
9.122.2.1 GST_IS_TENSOR_QUERY_SERVERSINK	1610
9.122.2.2 GST_IS_TENSOR_QUERY_SERVERSINK_CLASS	1610
9.122.2.3 GST_TENSOR_QUERY_SERVERSINK	1611
9.122.2.4 GST_TENSOR_QUERY_SERVERSINK_CAST	1611
9.122.2.5 GST_TENSOR_QUERY_SERVERSINK_CLASS	1611
9.122.2.6 GST_TYPE_TENSOR_QUERY_SERVERSINK	1611
9.122.3 Typedef Documentation	1611
9.122.3.1 GstTensorQueryServerSink	1611
9.122.3.2 GstTensorQueryServerSinkClass	1612
9.122.4 Function Documentation	1612
9.122.4.1 gst_tensor_query_serversink_get_type()	1612
9.123 nntstreamer/tensor_query/tensor_query_serversrc.c File Reference	1612
9.123.1 Detailed Description	1614
9.123.2 Macro Definition Documentation	1614
9.123.2.1 DEFAULT_DATA_POP_TIMEOUT	1614
9.123.2.2 DEFAULT_IS_LIVE	1614
9.123.2.3 DEFAULT_MQTT_HOST	1614
9.123.2.4 DEFAULT_MQTT_PORT	1615
9.123.2.5 DEFAULT_PORT_SRC	1615
9.123.2.6 GST_CAT_DEFAULT	1615
9.123.2.7 gst_tensor_query_serversrc_parent_class	1615
9.123.3 Enumeration Type Documentation	1615
9.123.3.1 anonymous enum	1615
9.123.4 Function Documentation	1616
9.123.4.1 _gst_tensor_query_serversrc_get_buffer()	1616
9.123.4.2 _gst_tensor_query_serversrc_playing()	1616
9.123.4.3 _gst_tensor_query_serversrc_start()	1617
9.123.4.4 _nns_edge_event_cb()	1618
9.123.4.5 G_DEFINE_TYPE()	1618
9.123.4.6 GST_DEBUG_CATEGORY_STATIC()	1618
9.123.4.7 gst_tensor_query_serversrc_change_state()	1618
9.123.4.8 gst_tensor_query_serversrc_class_init()	1619
9.123.4.9 gst_tensor_query_serversrc_create()	1620
9.123.4.10 gst_tensor_query_serversrc_finalize()	1620
9.123.4.11 gst_tensor_query_serversrc_get_property()	1621
9.123.4.12 gst_tensor_query_serversrc_init()	1622
9.123.4.13 gst_tensor_query_serversrc_set_caps()	1622
9.123.4.14 gst_tensor_query_serversrc_set_property()	1623
9.123.5 Variable Documentation	1623
9.123.5.1 srctemplate	1623
9.124 nntstreamer/tensor_query/tensor_query_serversrc.h File Reference	1624

9.124.1 Detailed Description	1625
9.124.2 Macro Definition Documentation	1625
9.124.2.1 GST_IS_TENSOR_QUERY_SERVERSRC	1625
9.124.2.2 GST_IS_TENSOR_QUERY_SERVERSRC_CLASS	1626
9.124.2.3 GST_TENSOR_QUERY_SERVERSRC	1626
9.124.2.4 GST_TENSOR_QUERY_SERVERSRC_CAST	1626
9.124.2.5 GST_TENSOR_QUERY_SERVERSRC_CLASS	1626
9.124.2.6 GST_TYPE_TENSOR_QUERY_SERVERSRC	1626
9.124.3 Typedef Documentation	1626
9.124.3.1 GstTensorQueryServerSrc	1627
9.124.3.2 GstTensorQueryServerSrcClass	1627
9.124.4 Function Documentation	1627
9.124.4.1 gst_tensor_query_serversrc_get_type()	1627
Index	1629

Chapter 1

nnstreamer

nnstreamer registerer Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

1.1 Introduction

- Introduction : Neural Network Streamer for AI Projects

1.2 Program Name

- Program Name : nnstreamer
- Program Details : It provides a neural network framework connectivities (e.g., tensorflow, caffe) for gstreamer streams. Efficient Streaming for AI Projects: Neural network models wanted to use efficient and flexible streaming management as well. Intelligent Media Filters!: Use a neural network model as a media filter / converter. Composite Models!: Allow to use multiple neural network models in a single stream instance. Multi Model Intelligence!: Allow to use multiple sources for neural network models.

1.3 Input/output data

- INPUT : None
- OUTPUT : None

1.4 Code information

- Initial date : 2018/06/14
- Version : 0.1

Chapter 2

Todo List

Member [_backtrace_to_string](#) (void)

The .c file location of this function might be not appropriate.

Member [_get_filenames](#) (nnsconf_type_path type, const gchar *dir, GSList **listF, GSList **listN, guint *counter)

This assumes .so/.dylib for all sub plugins. Support Windows!

Member [_gst_tensor_debug_output](#) (GstTensorDebug *self, GstBuffer *buffer)

NYI: do the debug task

Member [_GstTensorDecoder::tensor_config](#)

support tensors in the future

Member [_strdup_getenv](#) (const gchar *name)

Evaluate if we need to use secure_getenv() here (and compatible with other OS

Member [_validate_file](#) (nnsconf_type_path type, const gchar *fullpath)

how to validate with nnsconf type.

Member [ACCL_NPU](#)

Define ACCL_DSP

Member [ACCL_NPU_STR](#)

Define ACCL_DSP_STR

Member [custom_loadlib](#) (const GstTensorFilterProperties *prop, void **private_data)

: Check the integrity of filter->data and filter->model_file, nnfw

Double check if this check is really required and safe

Member [EVENT_NAME_UPDATE_MODEL](#)

rename & move this to better location

Member [g_tensor_filter_single_invoke](#) (GTensorFilterSingle *self, const GstTensorMemory *input, GstTensorMemory *output, gboolean allocate)

refactor this local variable

how can we remove memcpy if output data is already allocated single-shot should fill the output data, but sub-plugin allocates new memory.

Member [gst_mqtt_sink_set_host_address](#) (GstMqttSink *self, const gchar *addr)

Handle the case where the addr is changed at runtime

Member [gst_mqtt_sink_start](#) (GstBaseSink *basesink)

Support other persistence mechanisms MQTTCLIENT_PERSISTENCE_NONE: A memory-based persistence mechanism MQTTCLIENT_PERSISTENCE_DEFAULT: The default file system-based persistence mechanism MQTTCLIENT_PERSISTENCE_USER: An application-specific persistence mechanism

Member `gst_mqtt_src_create` (`GstBaseSrc *basesrc`, `guint64 offset`, `guint size`, `GstBuffer **buf`)

DEFAULT_MQTT_SUB_TIMEOUT_MIN is too long

If the difference between new latency and old latency, `gst_element_post_message` (`GST_ELEMENT_CAST (self)`, `gst_message_new_latency` (`GST_OBJECT_CAST (self)`)); is needed.

: Send EoS here

Member `gst_mqtt_src_set_host_address` (`GstMqttSrc *self`, `const gchar *addr`)

Handle the case where the addr is changed at runtime

Member `gst_mqtt_src_start` (`GstBaseSrc *basesrc`)

Support other persistence mechanisms
 MQTTCLIENT_PERSISTENCE_NONE: A memory-based persistence mechanism
 MQTTCLIENT_PERSISTENCE_DEFAULT: The default file system-based persistence mechanism
 MQTTCLIENT_PERSISTENCE_USER: An application-specific persistence mechanism

Member `gst_tensor_aggregator_chain` (`GstPad *pad`, `GstObject *parent`, `GstBuffer *buf`)

flush data Invalid state, tried to flush large size. We have to determine how to handle this case. (flush the out-size or all available bytes) Now all available bytes in adapter will be flushed.

Member `gst_tensor_aggregator_parse_caps` (`GstTensorAggregator *self`, `const GstCaps *caps`)

flush data Check properties to detect invalid case. Assertion when in=5 out=10 flush=20 or in=10 out=5 flush=20

Member `gst_tensor_buffer_append_memory` (`GstBuffer *buffer`, `GstMemory *memory`, `const GstTensorInfo *info`)

Make custom `gst_allocator` later?

Member `gst_tensor_converter_chain` (`GstPad *pad`, `GstObject *parent`, `GstBuffer *buf`)

need rewrite. do not use assert

expand mem if given property is larger than mem size. Now compare same size, later we should modify mem block if developer sets different dimension.

identify and printout the given input stream caps.

Member `gst_tensor_converter_parse_video` (`GstTensorConverter *self`, `const GstCaps *caps`, `GstTensorsConfig *config`)

Add more conditions!

need rewrite.

Member `gst_tensor_converter_set_property` (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)

detects framework based on the script extension

Member `gst_tensor_converter_video_stride` (`GstVideoFormat format`, `gint width`)

The actual list is much longer, fill them. (read <https://gststreamer.freedesktop.org/documentation/design/html>)

Member `gst_tensor_crop_do_cropping` (`GstTensorCrop *self`, `GstBuffer *raw`, `tensor_crop_info_s *cinfo`)

Add various mode to crop tensor.

Member `gst_tensor_crop_get_crop_info` (`GstTensorCrop *self`, `GstBuffer *info`, `tensor_crop_info_s *cinfo`)

Add various mode to crop tensor. Now tensor-crop handles NHWC data format only.

Member `gst_tensor_debug_class_init` (`GstTensorDebugClass *klass`)

check the behavior of name and nick (output methods vs output)

Member `gst_tensor_filter_configure_tensor` (`GstTensorFilter *self`, `const GstCaps *incaps`)

We do not support this (flexible tensor for flexible input model). Cap-negotiation of the current tensor-filter requires either side of "model / set-property" or "incoming gscaps" to be static/explicit. Ideally, this should support flexible tensor for flexible input model, leaving the negotiation to other elements, but we didn't implement it yet.

framerate of output tensors How can we update the framerate? `GstTensorFilter` cannot assure the framerate. Simply set the framerate of out-tensor from incaps.

Member [gst_tensor_filter_transform_caps](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [GstCaps](#) *filter)

how to set the framerate of output tensors

We do not have a testcase hitting here. Thus, we do not ensure the validity here. However, according to gstreamer doxygen entry, if filter is given, that's not to be ignored. For now, we assume that if caps-size is 0, filter is "ANY".

Member [gst_tensor_info_convert_to_meta](#) ([GstTensorInfo](#) *info, [GstTensorMetaInfo](#) *meta)

handle rank from info.dimension

Member [gst_tensor_meta_info_parse_header](#) ([GstTensorMetaInfo](#) *meta, [gpointer](#) header)

update meta info for each version

Member [gst_tensor_query_client_set_property](#) ([GObject](#) *object, [guint](#) prop_id, [const GValue](#) *value, [GParamSpec](#) *pspec)

DO NOT update properties (host, port, ..) while pipeline is running.

Member [gst_tensor_sparse_dec_chain](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstBuffer](#) *buf)

consider more error handling

Member [gst_tensor_trainer_set_prop_framework](#) ([GstTensorTrainer](#) *trainer, [const GValue](#) *value)

Check valid framework

Member [gst_tensor_transform_arithmetic](#) ([GstTensorTransform](#) *filter, [GstTensorInfo](#) *in_info, [GstTensorInfo](#) *out_info, [const uint8_t](#) *inptr, [uint8_t](#) *outptr)

add more options

Member [gst_tensor_transform_dimchg](#) ([GstTensorTransform](#) *filter, [GstTensorInfo](#) *in_info, [GstTensorInfo](#) *out_info, [const uint8_t](#) *inptr, [uint8_t](#) *outptr)

do "IP" operation

CRITICAL-TODO: Optimize the performance!

NYI

Member [gst_tensor_transform_padding](#) ([GstTensorTransform](#) *filter, [GstTensorInfo](#) *in_info, [GstTensorInfo](#) *out_info, [const uint8_t](#) *inptr, [uint8_t](#) *outptr)

Add constant option instead of using zero padding value

Member [gst_tensor_transform_transform](#) ([GstBaseTransform](#) *trans, [GstBuffer](#) *inbuf, [GstBuffer](#) *outbuf)

max rank supported in tensor-transform is 4

Member [gst_tensor_transform_transform_caps](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [GstCaps](#) *filtercap)

Get to know what the heck is @filtercap and use it!

Member [gst_tensordec_transform_caps](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [GstCaps](#) *filter)

We may do more specific actions here

Member [gst_tensordec_transform_size](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [gsize](#) size, [GstCaps](#) *othercaps, [gsize](#) *othersize)

If direction = SRC, you may need different interpretation!

Member [gst_tensorsinfo_compare_print](#) ([const GstTensorsInfo](#) *info1, [const GstTensorsInfo](#) *info2)

If this is going to be used by other elements, move this to nnstreamer/tensor_common.

File [gsttensor_converter.c](#)

For flatbuffers, support other/tensors with properties

Subplugins are not tested, yet.

File [gsttensor_if.h](#)

Add "event/signal" to reload FILL_WITH_FILE* file??? (TBD)

File [gsttensor_srciiio.c](#)

support specific channels as input

handle timestamp received from device

Member [mlagent_get_model_path_from](#) (const GValue *val)

The specification of the data layout filled in the third argument (i.e., stringified_json) by the callee is not fully decided.

Parse stringified_json to get the model's path

Member [nnsconf_dump](#) (gchar *str, gulong size)

Add more configuration values to dump.

File [ntputil.c](#)

Need to support caching and polling timer mechanism

Member [register_subplugin](#) (subpluginType type, const char *name, const void *data)

data out of scope at add

Member [TDBG_OUTPUT_CIRCULARBUF](#)

NYI NOT_SUPPORTED)

Member [TDBG_OUTPUT_FILEWRITE](#)

NYI NOT_SUPPORTED)

Class [tensor_crop_info_s](#)

Add various mode to crop tensor. Now tensor-crop handles NHWC data format only.

File [tensor_filter.c](#)

set priority among properties

logic for dynamic properties(like model change)

File [tensor_filter.h](#)

TBD: Should we disable "in-place" mode? (what if output size > input size?)

Chapter 3

Bug List

File [edge_common.c](#)

No known bugs

File [edge_common.h](#)

No known bugs

File [edge_elements.c](#)

No known bugs

File [edge_sink.c](#)

No known bugs

File [edge_sink.h](#)

No known bugs

File [edge_src.c](#)

No known bugs

File [edge_src.h](#)

No known bugs

File [gstdatarepo.c](#)

No known bugs except for NYI items

File [gstdatarepo.h](#)

No known bugs except for NYI items

File [gstdatareposink.c](#)

No known bugs except for NYI items

No known bugs except for NYI items

File [gstdatareposrc.c](#)

No known bugs except for NYI items

File [gstdatareposrc.h](#)

No known bugs except for NYI items

File [gstjoin.c](#)

No known bugs except for NYI items

File [gstjoin.h](#)

No known bugs except for NYI items

File [gsttensor_aggregator.c](#)

No known bugs except for NYI items

File [gsttensor_aggregator.h](#)

No known bugs except for NYI items

File [gsttensor_converter.c](#)

No known bugs except for NYI items

File [gsttensor_converter.h](#)

No known bugs except for NYI items

File [gsttensor_converter_media_info_audio.h](#)

No known bugs except for NYI items

File [gsttensor_converter_media_info_video.h](#)

No known bugs except for NYI items

File [gsttensor_converter_media_no_audio.h](#)

No known bugs except for NYI items

File [gsttensor_converter_media_no_video.h](#)

No known bugs except for NYI items

File [gsttensor_crop.c](#)

No known bugs except for NYI items

File [gsttensor_crop.h](#)

No known bugs except for NYI items

File [gsttensor_debug.c](#)

No known bugs except for NYI items

File [gsttensor_debug.h](#)

No known bugs except for NYI items

File [gsttensor_decoder.c](#)

gst_tensordec_transform_size () may be incorrect if direction is SINK.

If configured = TRUE, it holds TRUE until exit. What if configuration changes in run-time?

File [gsttensor_decoder.h](#)

No known bugs except for NYI items

File [gsttensor_demux.c](#)

No known bugs except for NYI items

File [gsttensor_demux.h](#)

No known bugs except for NYI items

File [gsttensor_if.c](#)

No known bugs except for NYI items

File [gsttensor_if.h](#)

No known bugs except for NYI items

File [gsttensor_merge.c](#)

No known bugs except for NYI items

File [gsttensor_merge.h](#)

No known bugs except for NYI items

File [gsttensor_mux.c](#)

No known bugs except for NYI items

File [gsttensor_mux.h](#)

No known bugs except for NYI items

File [gsttensor_rate.c](#)

No known bugs except for NYI items

File [gsttensor_rate.h](#)

No known bugs except for NYI items

File [gsttensor_repo.c](#)

No known bugs except for NYI items

File [gsttensor_repo.h](#)

No known bugs except for NYI items

File [gsttensor_reposink.c](#)

No known bugs except for NYI items

File [gsttensor_reposink.h](#)

No known bugs except for NYI items

File [gsttensor_reposrc.c](#)

No known bugs except for NYI items

File [gsttensor_reposrc.h](#)

No known bugs except for NYI items

File [gsttensor_sink.c](#)

No known bugs except for NYI items

File [gsttensor_sink.h](#)

No known bugs except for NYI items

File [gsttensor_sparsedec.c](#)

No known bugs except for NYI items

File [gsttensor_sparsedec.h](#)

No known bugs except for NYI items

File [gsttensor_sparseenc.c](#)

No known bugs except for NYI items

File [gsttensor_sparseenc.h](#)

No known bugs except for NYI items

File [gsttensor_sparseutil.c](#)

No known bugs except for NYI items

File [gsttensor_sparseutil.h](#)

No known bugs except for NYI items

File [gsttensor_split.c](#)

No known bugs except for NYI items

File [gsttensor_split.h](#)

No known bugs except for NYI items

File [gsttensor_srciiio.c](#)

No known bugs except for NYI items

File [gsttensor_srciiio.h](#)

No known bugs except for NYI items

File [gsttensor_trainer.c](#)

No known bugs except for NYI items

File [gsttensor_trainer.h](#)

No known bugs except for NYI items

File [gsttensor_transform.c](#)

This is NYI.

File [gsttensor_transform.h](#)

No known bugs.

File [hw_accel.c](#)

No known bugs except for NYI items

File [hw_accel.h](#)

No known bugs except for NYI items

File [ml_agent.c](#)

No known bugs except for NYI items

File [ml_agent.h](#)

No known bugs except for NYI items

File [mqttcommon.h](#)

No known bugs except for NYI items

File [mqttpsink.c](#)

No known bugs except for NYI items

No known bugs except for NYI items

File [mqttpsink.h](#)

No known bugs except for NYI items

File [mqttpsrc.c](#)

No known bugs except for NYI items

File [mqttpsrc.h](#)

No known bugs except for NYI items

File [nnstreamer.c](#)

No known bugs except for NYI items

File [nnstreamer_conf.c](#)

No known bugs except for NYI items

File [nnstreamer_conf.h](#)

No known bugs except for NYI items

File [nnstreamer_internal.h](#)

No known bugs except for NYI items

File [nnstreamer_log.c](#)

No known bugs except for NYI items

File [nnstreamer_log.h](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api.h](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api_converter.h](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api_decoder.h](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api_filter.h](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api_impl.c](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api_trainer.h](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api_util.h](#)

No known bugs except for NYI items

File [nnstreamer_plugin_api_util_impl.c](#)

No known bugs except for NYI items

File [nnstreamer_subplugin.c](#)

No known bugs except for NYI items

File [nnstreamer_subplugin.h](#)

No known bugs except for NYI items

File [nnstreamer_util.h](#)

No known bugs except for NYI items

File [nnstreamer_watchdog.c](#)

No known bugs except for NYI items

File [nnstreamer_watchdog.h](#)

No known bugs except for NYI items

File [ntputil.c](#)

No known bugs except for NYI items

File [ntputil.h](#)

No known bugs except for NYI items

File [tensor_allocator.c](#)

No known bugs

File [tensor_common.h](#)

No known bugs except for NYI items

File [tensor_converter_custom.h](#)

No known bugs except for NYI items

File [tensor_data.c](#)

No known bugs except for NYI items

File [tensor_data.h](#)

No known bugs except for NYI items

File [tensor_decoder_custom.h](#)

No known bugs except for NYI items

File [tensor_filter.c](#)

No known bugs except for NYI items

File [tensor_filter.h](#)

No known bugs except for NYI items

File [tensor_filter_common.c](#)

No known bugs except for NYI items

File [tensor_filter_common.h](#)

No known bugs except for NYI items

File [tensor_filter_custom.c](#)

No known bugs except for NYI items

File [tensor_filter_custom.h](#)

No known bugs except for NYI items

File [tensor_filter_custom_easy.c](#)

No known bugs except for NYI items

File [tensor_filter_custom_easy.h](#)

No known bugs except for NYI items

File [tensor_filter_single.c](#)

No known bugs except for NYI items

File [tensor_filter_single.h](#)

No known bugs except for NYI items

File [tensor_filter_support_cc.cc](#)

No known bugs except for NYI items

File [tensor_if.h](#)

No known bugs except for NYI items

File [tensor_meta.c](#)

No known bugs

File [tensor_meta.h](#)

No known bugs

File [tensor_query_client.c](#)

No known bugs

File [tensor_query_client.h](#)

No known bugs

File [tensor_query_common.c](#)

No known bugs except for NYI items

File [tensor_query_common.h](#)

No known bugs except for NYI items

File [tensor_query_server.c](#)

No known bugs

File [tensor_query_server.h](#)

No known bugs

File [tensor_query_serversink.c](#)

No known bugs

File [tensor_query_serversink.h](#)

No known bugs

File [tensor_query_serversrc.c](#)

No known bugs

File [tensor_query_serversrc.h](#)

No known bugs

File [tensor_typedef.h](#)

No known bugs except for NYI items

Chapter 4

Namespace Index

4.1 Namespace List

Here is a list of all namespaces with brief descriptions:

[nnstreamer](#) 27

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

_FilterTransformData	Internal data structure for tensor_filter transform data	29
_GstDataRepoSink	GstDataRepoSink data structure	31
_GstDataRepoSinkClass	GstDataRepoSinkClass data structure	35
_GstDataRepoSrc	GstDataRepoSrc data structure	35
_GstDataRepoSrcClass	GstDataRepoSrcClass data structure	46
_GstEdgeSink	GstEdgeSink data structure	47
_GstEdgeSinkClass	GstEdgeSinkClass data structure	50
_GstEdgeSrc	GstEdgeSrc data structure	50
_GstEdgeSrcClass	GstEdgeSrcClass data structure	52
_GstJoin	Internal data structure for join instances	53
_GstJoinClass	GstJoinClass inherits GstElementClass	55
_GstJoinPad	GstJoinPad data structure	56
_GstJoinPadClass	_GstJoinPadClass data structure	57
_GstMQTTMessageHdr	Defined a custom data type, GstMQTTMessageHdr	57
_GstMqttSink	GstMqttSink data structure	60
_GstMqttSinkClass	GstMqttSinkClass data structure	66
_GstMqttSrc	GstMqttSrc data structure	67
_GstMqttSrcClass	GstMqttSrcClass data structure	72

_GstTensorAggregator	GstTensorAggregator data structure	73
_GstTensorAggregatorClass	GstTensorAggregatorClass data structure	76
_GstTensorConverter	Internal data structure for tensor_converter instances	77
_GstTensorConverterClass	GstTensorConverterClass data structure	83
_GstTensorCrop	GstTensorCrop data structure	84
_GstTensorCropClass	GstTensorCropClass data structure	86
_GstTensorDebug	Internal data structure for tensor_debug instances	87
_GstTensorDebugClass	GstTensorDebugClass data structure	89
_GstTensorDecoder	Internal data structure for tensordec instances	90
_GstTensorDecoderClass	GstTensorDecoderClass inherits GstBaseTransformClass	93
_GstTensorDecoderDef	Decoder definitions for different semantics of tensors This allows developers to create their own decoders	94
_GstTensorDemux	Tensor Muxer data structure	97
_GstTensorDemuxClass	GstTensorDeMuxClass inherits GstElementClass	100
_GstTensorFilter	Internal data structure for tensor_filter instances	100
_GstTensorFilterClass	GstTensorFilterClass inherits GstBaseTransformClass	102
_GstTensorFilterCombination	Structure definition for tensor-filter in/out combination	103
_GstTensorFilterFramework	Tensor_Filter Subplugin definition	105
_GstTensorFilterFrameworkEventData	User data for the tensor_tilter subplugin related events	116
_GstTensorFilterFrameworkInfo	Tensor_Filter Subplugin framework related information	120
_GstTensorFilterFrameworkStatistics	Structure definition for tensor-filter framework statistics	123
_GstTensorFilterPrivate	Structure definition for common tensor-filter properties	124
_GstTensorFilterProperties	GstTensorFilter's properties for NN framework (internal data structure)	128
_GstTensorFilterStatistics	Structure definition for tensor-filter statistics	134
_GstTensorIf	Tensor If data structure	136
_GstTensorIfClass	GstTensorIfClass inherits GstElementClass	140
_GstTensorMerge	Tensor Merge data structure	141
_GstTensorMergeClass	GstTensorMergeClass inherits GstElementClass	144
_GstTensorMux	Tensor Muxer data structure	145

_GstTensorMuxClass	
GstTensorMuxClass inherits GstElementClass	148
_GstTensorQueryClient	
GstTensorQueryClient data structure	149
_GstTensorQueryClientClass	
GstTensorQueryClientClass data structure	153
_GstTensorQueryServerSink	
GstTensorQueryServerSink data structure	154
_GstTensorQueryServerSinkClass	
GstTensorQueryServerSinkClass data structure	156
_GstTensorQueryServerSrc	
GstTensorQueryServerSrc data structure	157
_GstTensorQueryServerSrcClass	
GstTensorQueryServerSrcClass data structure	159
_GstTensorRate	
Tensor Rate data structure	160
_GstTensorRateClass	
GstTensorRateClass inherits GstElementClass	165
_GstTensorRepoSink	
GstTensorRepoSink data structure	166
_GstTensorRepoSinkClass	
GstTensorRepoSinkClass data structure	168
_GstTensorRepoSrc	
GstTensorRepoSrc data structure	169
_GstTensorRepoSrcClass	
GstTensorRepoSrcClass data structure	172
_GstTensorSink	
GstTensorSink data structure	172
_GstTensorSinkClass	
GstTensorSinkClass data structure	174
_GstTensorSparseDec	
GstTensorSparseDec data structure	176
_GstTensorSparseDecClass	
GstTensorSparseClass data structure	178
_GstTensorSparseEnc	
GstTensorSparseEnc data structure	179
_GstTensorSparseEncClass	
GstTensorSparseClass data structure	181
_GstTensorSplit	
Tensor Split data structure	182
_GstTensorSplitClass	
GstTensorSplitClass inherits GstElementClass	185
_GstTensorSrcIO	
GstTensorSrcIO data structure	185
_GstTensorSrcIOChannelProperties	
GstTensorSrcIO channel's properties (internal data structure)	191
_GstTensorSrcIOClass	
GstTensorSrcIOClass data structure	196
_GstTensorSrcIODeviceProperties	
GstTensorSrcIO devices's properties (internal data structure)	196
_GstTensorTrainer	
GstTensorTrainer data structure	198
_GstTensorTrainerClass	
GstTensorTrainerClass data structure	202
_GstTensorTrainerEventNotifier	
GstTensorTrainer's event notifier	203
_GstTensorTrainerFramework	
Tensor_trainer subplugin definition	204

_GstTensorTrainerFrameworkInfo	GstTensorTrainer's subplugin framework related information	209
_GstTensorTrainerProperties	GstTensorTrainer's properties for neural network framework (internal data structure)	210
_GstTensorTransform	Internal data structure for tensor_transform instances	214
_GstTensorTransformClass	GstTensorTransformClass inherits GstBaseTransformClass	218
_GTensorFilterSingle	Internal data structure for tensor_filter_single instances	219
_GTensorFilterSingleClass	GTensorFilterSingleClass inherits GObjectClass	220
_GTensorFilterSinglePrivate	Private data struct for tensor-filter single class	222
_internal_data	Internal_data	224
_NNStreamer_custom_class	Custom Filter Class	225
_NNStreamerExternalConverter	Converter's subplugin implementation	228
_NnstWatchdog	Structure for NNStreamer watchdog	231
_ntp_packet_t	A custom data type to represent NTP packet header format	233
_ntp_timestamp_t	A custom data type to represent NTP timestamp format	236
_tensor_merge_linear	Internal data structure for linear mode	237
_tensor_sync_basepad_data	Tensor Merge/Mux sync data for baspad mode	237
_tensor_time_sync_data	Tensor Merge/Mux time sync data	238
_tensor_transform_arithmetic	Internal data structure for arithmetic mode	240
_tensor_transform_clamp	Internal data structure for clamp mode	241
_tensor_transform_dimchg	Internal data structure for dimchg mode	241
_tensor_transform_padding	Internal data structure for padding mode	242
_tensor_transform_stand	Internal data structure for stand mode	243
_tensor_transform_transpose	Internal data structure for transpose mode	244
_tensor_transform_typecast	Internal data structure for typecast mode	245
confdata	246
converter_custom_cb_s	248
custom_cb_s	248
decoder_custom_cb_s	249
dump_buf	250
gst_tensor_aggregation_data_s	Internal struct to handle aggregation data in hash table	251
GstMetaQuery	GstMetaQuery meta structure	252
GstSparseTensorInfo	Internal data structure for sparse tensor info	253

GstTensorAllocator	
Struct for type GstTensorAllocator	253
GstTensorAllocatorClass	
Struct for class GstTensorAllocatorClass	254
GstTensorCollectPadData	
Internal data structure for Collect Pad in mux / merge	255
GstTensorCropPadData	
GstTensorCrop pad data	256
GstTensorExtralInfo	
Data structure to describe a "extra" tensor data. This represents the information of the NNS_ \leftrightarrow TENSOR_SIZE_LIMIT-th memory block for tensor stream	257
GstTensorFilterSharedModelRepresentation	
Data Structure to store shared table	259
GstTensorInfo	
Internal data structure for tensor info	260
GstTensorMemory	
The unit of each data tensors. It will be used as an input/output tensor of other/tensors	261
GstTensorMetalInfo	
Data structure to describe a tensor data. This represents the basic information of a memory block for tensor stream	262
GstTensorPad	
Internal data structure for pad in demux / split	265
GstTensorQueryEdgeInfo	
Internal data structure for nns-edge info to prepare edge connection	266
GstTensorQueryServer	
GstTensorQueryServer internal info data structure	268
GstTensorRepo	
GstTensorRepo data structure	269
GstTensorRepoData	
GstTensorRepo internal data structure	270
GstTensorsConfig	
Internal data structure for configured tensors info (for other/tensors)	273
GstTensorsInfo	
Internal meta data exchange format for a other/tensors instance	275
parse_accl_args	
Accelerator related arguments for parsing	276
runtime_data	
The easy-filter user's data	278
subplugin_conf	280
subplugin_info_s	281
tensor_crop_info_s	
Internal data structure to describe cropping tensor data	282
tensor_data_s	
Structure for tensor data	283
tensor_element	
To make the code simple with all the types. "C++ Template"-like	284
tensor_if_sv_s	
Internal data structure for supplied value	286
tensor_region_s	
Internal data structure to describe tensor region	287
tensor_transform_operator_s	
Internal data structure for operator of arithmetic mode	289
vstr_helper	
Data structure for <code>_g_list_foreach_vstr_helper</code>	290

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

datarepo/ gstdatarepo.c	
Register datarepo plugins	293
datarepo/ gstdatarepo.h	
GStreamer plugin to read file in MLOps Data repository into buffers	296
datarepo/ gstdatareposink.c	
GStreamer plugin that writes data from buffers to files in in MLOps Data repository	298
datarepo/ gstdatareposink.h	316
datarepo/ gstdatareposrc.c	
GStreamer plugin to read file in MLOps Data repository into buffers	319
datarepo/ gstdatareposrc.h	
GStreamer plugin to read file in MLOps Data repository into buffers	341
edge/ edge_common.c	
Common functions for edge sink and src	344
edge/ edge_common.h	
Common functions for edge sink and src	346
edge/ edge_elements.c	
Register edge plugins	349
edge/ edge_sink.c	
Publish incoming streams	350
edge/ edge_sink.h	
Publish incoming streams	363
edge/ edge_src.c	
Subscribe and push incoming data to the GStreamer pipeline	367
edge/ edge_src.h	
Subscribe and push incoming data to the GStreamer pipeline	379
join/ gstjoin.c	
Select the out that arrived first among the input streams	382
join/ gstjoin.h	
Select the out that arrived first among the input streams	399
mqtt/ mqttcommon.h	
Common macros and utility functions for GStreamer MQTT plugins	403
mqtt/ mqttelements.c	407
mqtt/ mqttsink.c	
Register sub-plugins included in libgstmqtt	408
mqtt/ mqttsink.h	
Publish incoming data streams as a MQTT topic	443

mqtt/mqttsrc.c	Subscribe a MQTT topic and push incoming data to the GStreamer pipeline	448
mqtt/mqttsrc.h	Subscribe a MQTT topic and push incoming data to the GStreamer pipeline	480
mqtt/nputil.c	NTP utility functions	484
mqtt/nputil.h	A header file of NTP utility functions	488
nnstreamer/hw_accel.c	Common hardware acceleration availability checks	995
nnstreamer/hw_accel.h	Common hardware acceleration availability header	996
nnstreamer/ml_agent.c	Internal helpers to make a bridge between NNS filters and the ML Agent service	1156
nnstreamer/ml_agent.h	Internal header to make a bridge between NNS filters and the ML Agent service	1159
nnstreamer/nnstreamer_conf.c	NNStreamer Configuration (conf file, env-var) Management	1160
nnstreamer/nnstreamer_conf.h	Internal header for conf/env-var management	1179
nnstreamer/nnstreamer_internal.h	Internal header for NNStreamer plugins and native single-shot APIs	1190
nnstreamer/nnstreamer_log.c	Internal log and error handling for NNStreamer plugins and core codes	1196
nnstreamer/nnstreamer_log.h	Internal log util for NNStreamer plugins and native APIs	1199
nnstreamer/nnstreamer_plugin_api_impl.c	Common data for NNStreamer, the GStreamer plugin for neural networks	1206
nnstreamer/nnstreamer_plugin_api_util_impl.c	Tensor common util functions for NNStreamer. (No gst dependency)	1250
nnstreamer/nnstreamer_subplugin.c	Subplugin Manager for NNStreamer	1322
nnstreamer/nnstreamer_subplugin.h	Subplugin Manager for NNStreamer	1335
nnstreamer/nnstreamer_watchdog.c	NNStreamer watchdog to manage the schedule in the element	1344
nnstreamer/nnstreamer_watchdog.h	NNStreamer watchdog header to manage the schedule in the element	1349
nnstreamer/tensor_allocator.c	Allocator for memory alignment	1356
nnstreamer/tensor_common.h	Common header file for NNStreamer, the GStreamer plugin for neural networks	1361
nnstreamer/tensor_data.c	Internal functions to handle various tensor type and value	1380
nnstreamer/tensor_data.h	Internal functions to handle various tensor type and value	1390
nnstreamer/tensor_meta.c	Internal tensor meta implementation for nnstreamer	1553
nnstreamer/tensor_meta.h	Internal tensor meta header for nnstreamer	1557
nnstreamer/elements/gsttensor_aggregator.c	GStreamer plugin to aggregate tensor stream	491
nnstreamer/elements/gsttensor_aggregator.h	GStreamer plugin to aggregate tensor stream	512
nnstreamer/elements/gsttensor_converter.c	GStreamer plugin to convert media types to tensors (as a filter for other general neural network filters)	515

nnstreamer/elements/ gsttensor_converter.h	
GStreamer plugin to convert media types to tensors (as a filter for other general neural network filters)	553
nnstreamer/elements/ gsttensor_converter_media_info_audio.h	
Define collection of media type and functions to parse media info for audio support	557
nnstreamer/elements/ gsttensor_converter_media_info_video.h	
Define collection of media type and functions to parse media info for video support	559
nnstreamer/elements/ gsttensor_converter_media_no_audio.h	
Define collection of media type and functions to parse media info for audio if there is no audio support	561
nnstreamer/elements/ gsttensor_converter_media_no_video.h	
Define collection of media type and functions to parse media info for video if there is no video support	565
nnstreamer/elements/ gsttensor_crop.c	
GStreamer element to crop the regions of incoming tensor	568
nnstreamer/elements/ gsttensor_crop.h	
GStreamer element to crop the regions of incoming tensor	585
nnstreamer/elements/ gsttensor_debug.c	
GStreamer plugin to help debug tensor streams	588
nnstreamer/elements/ gsttensor_debug.h	
GStreamer plugin to help debug tensor streams	599
nnstreamer/elements/ gsttensor_decoder.c	
GStreamer plugin to convert tensors (as a filter for other general neural network filters) to other media types	604
nnstreamer/elements/ gsttensor_decoder.h	
GStreamer plugin to convert tensors to media types	630
nnstreamer/elements/ gsttensor_demux.c	
GStreamer plugin to demux tensors (as a filter for other general neural network filters)	634
nnstreamer/elements/ gsttensor_demux.h	
GStreamer plugin to demux tensors (as a filter for other general neural network filters)	649
nnstreamer/elements/ gsttensor_if.c	
GStreamer plugin to control flow based on tensor values	653
nnstreamer/elements/ gsttensor_if.h	
GStreamer plugin to control flow based on tensor values	679
nnstreamer/elements/ gsttensor_merge.c	
GStreamer plugin to merge tensors (as a filter for other general neural network filters)	685
nnstreamer/elements/ gsttensor_merge.h	
GStreamer plugin to merge tensors (as a filter for other general neural network filters)	704
nnstreamer/elements/ gsttensor_mux.c	
GStreamer plugin to mux tensors (as a filter for other general neural network filters)	708
nnstreamer/elements/ gsttensor_mux.h	
GStreamer plugin to mux tensors (as a filter for other general neural network filters)	726
nnstreamer/elements/ gsttensor_rate.c	
GStreamer plugin to adjust tensor rate	730
nnstreamer/elements/ gsttensor_rate.h	
GStreamer plugin to adjust tensor rate	748
nnstreamer/elements/ gsttensor_repo.c	
Tensor repo file for NNStreamer, the GStreamer plugin for neural networks	752
nnstreamer/elements/ gsttensor_repo.h	
Tensor repo header file for NNStreamer, the GStreamer plugin for neural networks	763
nnstreamer/elements/ gsttensor_reposink.c	
GStreamer plugin to handle tensor repository	772
nnstreamer/elements/ gsttensor_reposink.h	
GStreamer plugin to handle tensor repository	784
nnstreamer/elements/ gsttensor_reposrc.c	
GStreamer plugin to handle tensor repository	788
nnstreamer/elements/ gsttensor_reposrc.h	
GStreamer plugin to handle tensor repository	797

nnstreamer/elements/ gsttensor_sink.c	
GStreamer plugin to handle tensor stream	800
nnstreamer/elements/ gsttensor_sink.h	
GStreamer plugin to handle tensor stream	817
nnstreamer/elements/ gsttensor_sparsedec.c	
GStreamer element to decode sparse tensors into dense tensors	820
nnstreamer/elements/ gsttensor_sparsedec.h	
GStreamer element to decode sparse tensors into dense tensors	831
nnstreamer/elements/ gsttensor_sparseenc.c	
GStreamer element to encode dense tensors into sparse tensors	834
nnstreamer/elements/ gsttensor_sparseenc.h	
GStreamer element to encode sparse tensors into dense tensors	845
nnstreamer/elements/ gsttensor_sparseutil.c	
Util functions for tensor_sparse encoder and decoder	848
nnstreamer/elements/ gsttensor_sparseutil.h	
Util functions for tensor_sparse encoder and decoder	851
nnstreamer/elements/ gsttensor_split.c	
GStreamer plugin to split tensor (as a filter for other general neural network filters)	854
nnstreamer/elements/ gsttensor_split.h	
GStreamer plugin to split tensor (as a filter for other general neural network filters)	869
nnstreamer/elements/ gsttensor_srcii.c	
GStreamer plugin to capture sensor data as tensor(s)	873
nnstreamer/elements/ gsttensor_srcii.h	
GStreamer plugin to support linux IIO as tensor(s)	915
nnstreamer/elements/ gsttensor_trainer.c	
GStreamer plugin to train tensor data using NN Frameworks	920
nnstreamer/elements/ gsttensor_trainer.h	
GStreamer plugin to train tensor data using NN Frameworks	951
nnstreamer/elements/ gsttensor_transform.c	
GStreamer plugin to transform tensor dimension or type	954
nnstreamer/elements/ gsttensor_transform.h	
GStreamer plugin to transform tensor dimension or type	987
nnstreamer/elements/ ignore_warning.sh	995
nnstreamer/include/ nnstreamer_plugin_api.h	
Optional/Additional NNStreamer APIs for sub-plugin writers. (Need Gst devel)	998
nnstreamer/include/ nnstreamer_plugin_api_converter.h	
Mandatory APIs for NNStreamer Converter sub-plugins (Need Gst Devel)	1015
nnstreamer/include/ nnstreamer_plugin_api_decoder.h	
Mandatory APIs for NNStreamer Decoder sub-plugins (Need Gst Devel)	1020
nnstreamer/include/ nnstreamer_plugin_api_filter.h	
Mandatory APIs for NNStreamer Filter sub-plugins (No External Dependencies)	1025
nnstreamer/include/ nnstreamer_plugin_api_trainer.h	
Mandatory APIs for NNStreamer Trainer sub-plugins (No External Dependencies)	1042
nnstreamer/include/ nnstreamer_plugin_api_util.h	
Optional/Additional NNStreamer APIs for sub-plugin writers. (No GStreamer dependency)	1049
nnstreamer/include/ nnstreamer_util.h	
Optional NNStreamer utility functions for sub-plugin writers and users	1120
nnstreamer/include/ tensor_converter_custom.h	
NNStreamer APIs for tensor_converter custom condition	1122
nnstreamer/include/ tensor_decoder_custom.h	
NNStreamer APIs for tensor_decoder custom condition	1125
nnstreamer/include/ tensor_filter_custom.h	
Custom tensor post-processing interface for NNStreamer suite for post-processing code developers	1129
nnstreamer/include/ tensor_filter_custom_easy.h	
Custom tensor processing interface for simple functions	1136
nnstreamer/include/ tensor_if.h	
NNStreamer APIs for tensor_if custom condition	1142

nnstreamer/include/ tensor_typedef.h	
Common header file for NNStreamer, the GStreamer plugin for neural networks	1146
nnstreamer/registerer/ nnstreamer.c	
Registers all nnstreamer plugins for gstreamer so that we can have a single big binary	1353
nnstreamer/tensor_filter/ tensor_filter.c	
GStreamer plugin to use general neural network frameworks as filters	1399
nnstreamer/tensor_filter/ tensor_filter.h	
GStreamer plugin to use general neural network frameworks as filters	1432
nnstreamer/tensor_filter/ tensor_filter_common.c	
Common functions for various tensor_filters	1435
nnstreamer/tensor_filter/ tensor_filter_common.h	
Common functions for various tensor_filters	1495
nnstreamer/tensor_filter/ tensor_filter_custom.c	
Custom tensor post-processing interface for NNStreamer suite between NN developer-plugins and NNstreamer	1517
nnstreamer/tensor_filter/ tensor_filter_custom_easy.c	
Custom tensor processing interface for simple functions	1526
nnstreamer/tensor_filter/ tensor_filter_single.c	
Element to use general neural network framework directly without gstreamer pipeline	1535
nnstreamer/tensor_filter/ tensor_filter_single.h	
Element to use general neural network framework individually without gstreamer pipeline	1547
nnstreamer/tensor_filter/ tensor_filter_support_cc.cc	
Base class for tensor_filter subplugins of C++ classes	1551
nnstreamer/tensor_query/ tensor_query_client.c	
GStreamer plugin to handle tensor query client	1560
nnstreamer/tensor_query/ tensor_query_client.h	
GStreamer plugin to handle tensor query client	1575
nnstreamer/tensor_query/ tensor_query_common.c	
Utility functions for tensor query	1579
nnstreamer/tensor_query/ tensor_query_common.h	
Utility functions for tensor query	1580
nnstreamer/tensor_query/ tensor_query_server.c	
GStreamer plugin to handle meta_query for server elements	1583
nnstreamer/tensor_query/ tensor_query_server.h	
GStreamer plugin to handle meta_query for server elements	1591
nnstreamer/tensor_query/ tensor_query_serversink.c	
GStreamer plugin to handle tensor query server sink	1599
nnstreamer/tensor_query/ tensor_query_serversink.h	
GStreamer plugin to handle tensor query_server sink	1608
nnstreamer/tensor_query/ tensor_query_serversrc.c	
GStreamer plugin to handle tensor query_server src	1612
nnstreamer/tensor_query/ tensor_query_serversrc.h	
GStreamer plugin to handle tensor query_server src	1624

Chapter 7

Namespace Documentation

7.1 nnstreamer Namespace Reference

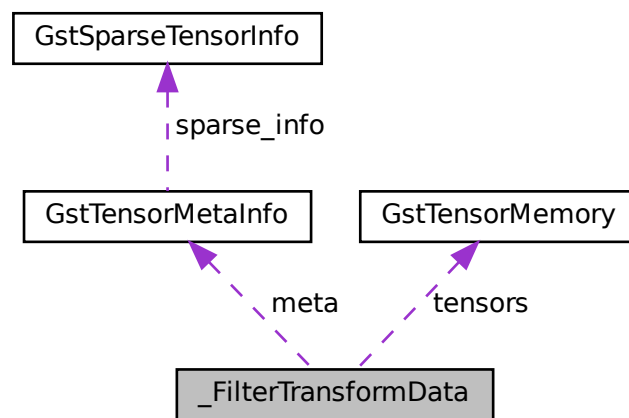
Chapter 8

Class Documentation

8.1 `_FilterTransformData` Struct Reference

Internal data structure for `tensor_filter` transform data.

Collaboration diagram for `_FilterTransformData`:



Public Attributes

- `GstMemory * mem` [`NNS_TENSOR_SIZE_LIMIT`]
- `GstMapInfo info` [`NNS_TENSOR_SIZE_LIMIT`]
- `GstTensorMetaInfo meta` [`NNS_TENSOR_SIZE_LIMIT`]
- `guint num_tensors`
- `GstTensorMemory tensors` [`NNS_TENSOR_SIZE_LIMIT`]
- `gboolean is_flexible`
- `gboolean allocate_in_invoke`

8.1.1 Detailed Description

Internal data structure for tensor_filter transform data.

Definition at line 153 of file tensor_filter.c.

8.1.2 Member Data Documentation

8.1.2.1 allocate_in_invoke

```
gboolean _FilterTransformData::allocate_in_invoke
```

Definition at line 162 of file tensor_filter.c.

8.1.2.2 info

```
GstMapInfo _FilterTransformData::info[NNS_TENSOR_SIZE_LIMIT]
```

Definition at line 156 of file tensor_filter.c.

8.1.2.3 is_flexible

```
gboolean _FilterTransformData::is_flexible
```

Definition at line 161 of file tensor_filter.c.

8.1.2.4 mem

```
GstMemory* _FilterTransformData::mem[NNS_TENSOR_SIZE_LIMIT]
```

Definition at line 155 of file tensor_filter.c.

8.1.2.5 meta

```
GstTensorMetaInfo _FilterTransformData::meta[NNS_TENSOR_SIZE_LIMIT]
```

Definition at line 157 of file tensor_filter.c.

8.1.2.6 num_tensors

```
guint _FilterTransformData::num_tensors
```

Definition at line 158 of file tensor_filter.c.

8.1.2.7 tensors

```
GstTensorMemory _FilterTransformData::tensors[NNS_TENSOR_SIZE_LIMIT]
```

Definition at line 159 of file tensor_filter.c.

The documentation for this struct was generated from the following file:

- nnstreamer/tensor_filter/tensor_filter.c

8.2 _GstDataRepoSink Struct Reference

GstDataRepoSink data structure.

```
#include <gstdatareposink.h>
```

Public Attributes

- GstBaseSink [element](#)
- GstCaps * [fixed_caps](#)
- JsonObject * [json_object](#)
- JsonArray * [sample_offset_array](#)
- JsonArray * [tensor_size_array](#)
- JsonArray * [tensor_count_array](#)
- guint [cumulative_tensors](#)
- gboolean [is_static_tensors](#)
- gint [fd](#)
- GstDataRepoDataType [data_type](#)
- gint [total_samples](#)
- guint64 [fd_offset](#)
- gsize [sample_size](#)
- gchar * [filename](#)
- gchar * [json_filename](#)

8.2.1 Detailed Description

GstDataRepoSink data structure.

Definition at line 40 of file gstdatareposink.h.

8.2.2 Member Data Documentation

8.2.2.1 cumulative_tensors

`guint _GstDataRepoSink::cumulative_tensors`

the number of cumulated tensors

Definition at line 49 of file `gstdatareposink.h`.

8.2.2.2 data_type

`GstDataRepoDataType _GstDataRepoSink::data_type`

data type

Definition at line 53 of file `gstdatareposink.h`.

8.2.2.3 element

`GstBaseSink _GstDataRepoSink::element`

Definition at line 42 of file `gstdatareposink.h`.

8.2.2.4 fd

`gint _GstDataRepoSink::fd`

open file descriptor

Definition at line 52 of file `gstdatareposink.h`.

8.2.2.5 fd_offset

`guint64 _GstDataRepoSink::fd_offset`

offset of fd

Definition at line 55 of file `gstdatareposink.h`.

8.2.2.6 filename

`gchar* _GstDataRepoSink::filename`

filename

Definition at line 59 of file `gstdatareposink.h`.

8.2.2.7 fixed_caps

`GstCaps* _GstDataRepoSink::fixed_caps`

to get meta info

Definition at line 44 of file `gstdatareposink.h`.

8.2.2.8 is_static_tensors

`gboolean _GstDataRepoSink::is_static_tensors`

Definition at line 51 of file `gstdatareposink.h`.

8.2.2.9 json_filename

`gchar* _GstDataRepoSink::json_filename`

"JSON file path to store the meta information

Definition at line 60 of file `gstdatareposink.h`.

8.2.2.10 json_object

`JsonObject* _GstDataRepoSink::json_object`

JSON object

Definition at line 45 of file `gstdatareposink.h`.

8.2.2.11 sample_offset_array

```
JsonArray* _GstDataRepoSink::sample_offset_array
```

offset array of sample

Definition at line 46 of file gstdataposink.h.

8.2.2.12 sample_size

```
gsize _GstDataRepoSink::sample_size
```

size of one sample

Definition at line 56 of file gstdataposink.h.

8.2.2.13 tensor_count_array

```
JsonArray* _GstDataRepoSink::tensor_count_array
```

array for the number of cumulative tensors

Definition at line 48 of file gstdataposink.h.

8.2.2.14 tensor_size_array

```
JsonArray* _GstDataRepoSink::tensor_size_array
```

size array of flexible tensor

Definition at line 47 of file gstdataposink.h.

8.2.2.15 total_samples

```
gint _GstDataRepoSink::total_samples
```

The number of total samples, in the case of multi-files, it is used as an index.

Definition at line 54 of file gstdataposink.h.

The documentation for this struct was generated from the following file:

- [datarepo/gstdataposink.h](#)

8.3 _GstDataRepoSinkClass Struct Reference

GstDataRepoSinkClass data structure.

```
#include <gstdatareposink.h>
```

Public Attributes

- GstBaseSinkClass [parent_class](#)

8.3.1 Detailed Description

GstDataRepoSinkClass data structure.

Definition at line 66 of file `gstdatareposink.h`.

8.3.2 Member Data Documentation

8.3.2.1 parent_class

```
GstBaseSinkClass _GstDataRepoSinkClass::parent_class
```

Definition at line 68 of file `gstdatareposink.h`.

The documentation for this struct was generated from the following file:

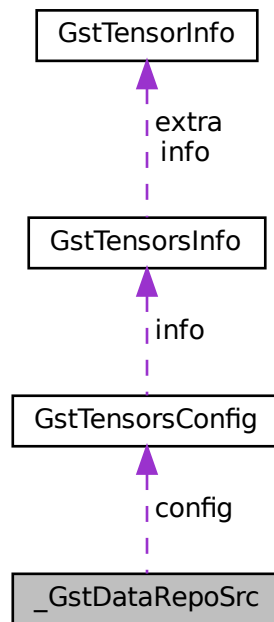
- `datarepo/gstdatareposink.h`

8.4 _GstDataRepoSrc Struct Reference

GstDataRepoSrc data structure.

```
#include <gstdatareposrc.h>
```

Collaboration diagram for `_GstDataRepoSrc`:



Public Attributes

- `GstPushSrc` [parent](#)
- `GstPad *` [src_pad](#)
- `gboolean` [is_start](#)
- `gboolean` [successful_read](#)
- `gint` [fd](#)
- `gint` [file_size](#)
- `guint64` [read_position](#)
- `guint64` [fd_offset](#)
- `guint64` [start_offset](#)
- `guint64` [last_offset](#)
- `gsize` [tensors_size](#) [`NNS_TENSOR_SIZE_LIMIT`]
- `gsize` [tensors_offset](#) [`NNS_TENSOR_SIZE_LIMIT`]
- `gint` [current_sample_index](#)
- `gboolean` [first_epoch_is_done](#)
- `guint` [total_samples](#)
- `guint` [num_samples](#)
- `gsize` [sample_size](#)
- `GstDataRepoDataType` [data_type](#)
- `gchar *` [filename](#)
- `gchar *` [json_filename](#)
- `gchar *` [tensors_seq_str](#)
- `guint` [start_sample_index](#)
- `guint` [stop_sample_index](#)

- guint [epochs](#)
- gboolean [is_shuffle](#)
- GArray * [shuffled_index_array](#)
- guint [array_index](#)
- guint [tensors_seq](#) [NNS_TENSOR_SIZE_LIMIT]
- guint [tensors_seq_cnt](#)
- gboolean [need_changed_caps](#)
- GstCaps * [caps](#)
- [GstTensorsConfig](#) [config](#)
- JsonArray * [sample_offset_array](#)
- JsonArray * [tensor_size_array](#)
- JsonArray * [tensor_count_array](#)
- JsonParser * [parser](#)
- guint [sample_offset_array_len](#)
- guint [tensor_size_array_len](#)
- guint [tensor_count_array_len](#)
- GstClockTime [running_time](#)
- gint [rate_n](#)
- gint [rate_d](#)
- guint64 [n_frame](#)

8.4.1 Detailed Description

GstDataRepoSrc data structure.

Definition at line 41 of file `gstdatareposrc.h`.

8.4.2 Member Data Documentation

8.4.2.1 `array_index`

```
guint _GstDataRepoSrc::array_index
```

element index of `shuffled_index_array`

Definition at line 73 of file `gstdatareposrc.h`.

8.4.2.2 `caps`

```
GstCaps* _GstDataRepoSrc::caps
```

optional property, `datareposrc` should get data format from JSON file `caps` field

Definition at line 78 of file `gstdatareposrc.h`.

8.4.2.3 config

`GstTensorsConfig` `_GstDataRepoSrc::config`

tensors information from current caps

Definition at line 81 of file `gstdatareposrc.h`.

8.4.2.4 current_sample_index

`gint` `_GstDataRepoSrc::current_sample_index`

current index of sample or file to read

Definition at line 56 of file `gstdatareposrc.h`.

8.4.2.5 data_type

`GstDataRepoDataType` `_GstDataRepoSrc::data_type`

media type

Definition at line 61 of file `gstdatareposrc.h`.

8.4.2.6 epochs

`guint` `_GstDataRepoSrc::epochs`

repetition of range of files or samples to read

Definition at line 69 of file `gstdatareposrc.h`.

8.4.2.7 fd

`gint` `_GstDataRepoSrc::fd`

open file descriptor

Definition at line 48 of file `gstdatareposrc.h`.

8.4.2.8 fd_offset

```
guint64 _GstDataRepoSrc::fd_offset
```

offset of fd

Definition at line 51 of file gstdataposrc.h.

8.4.2.9 file_size

```
gint _GstDataRepoSrc::file_size
```

file size, in bytes

Definition at line 49 of file gstdataposrc.h.

8.4.2.10 filename

```
gchar* _GstDataRepoSrc::filename
```

filename

Definition at line 64 of file gstdataposrc.h.

8.4.2.11 first_epoch_is_done

```
gboolean _GstDataRepoSrc::first_epoch_is_done
```

Definition at line 57 of file gstdataposrc.h.

8.4.2.12 is_shuffle

```
gboolean _GstDataRepoSrc::is_shuffle
```

shuffle the sample index

Definition at line 70 of file gstdataposrc.h.

8.4.2.13 is_start

```
gboolean _GstDataRepoSrc::is_start
```

check if datareposrc is started

Definition at line 46 of file gstdataposrc.h.

8.4.2.14 json_filename

```
gchar* _GstDataRepoSrc::json_filename
```

json filename containing meta information of the filename

Definition at line 65 of file gstdataposrc.h.

8.4.2.15 last_offset

```
guint64 _GstDataRepoSrc::last_offset
```

last offset to read

Definition at line 53 of file gstdataposrc.h.

8.4.2.16 n_frame

```
guint64 _GstDataRepoSrc::n_frame
```

Definition at line 92 of file gstdataposrc.h.

8.4.2.17 need_changed_caps

```
gboolean _GstDataRepoSrc::need_changed_caps
```

When tensors-sequence changes, caps need to be changed

Definition at line 77 of file gstdataposrc.h.

8.4.2.18 num_samples

```
guint _GstDataRepoSrc::num_samples
```

The number of samples to be used out of the total samples in the file

Definition at line 59 of file gstdataposrc.h.

8.4.2.19 parent

```
GstPushSrc _GstDataRepoSrc::parent
```

parent object

Definition at line 43 of file gstdataposrc.h.

8.4.2.20 parser

```
JsonParser* _GstDataRepoSrc::parser
```

Keep JSON data after parsing JSON file

Definition at line 85 of file gstdataposrc.h.

8.4.2.21 rate_d

```
gint _GstDataRepoSrc::rate_d
```

Definition at line 91 of file gstdataposrc.h.

8.4.2.22 rate_n

```
gint _GstDataRepoSrc::rate_n
```

Definition at line 91 of file gstdataposrc.h.

8.4.2.23 read_position

```
guint64 _GstDataRepoSrc::read_position
```

position of fd

Definition at line 50 of file gstdataposrc.h.

8.4.2.24 running_time

```
GstClockTime _GstDataRepoSrc::running_time
```

one frame running time

Definition at line 90 of file gstdataposrc.h.

8.4.2.25 sample_offset_array

```
JsonArray* _GstDataRepoSrc::sample_offset_array
```

offset array of sample

Definition at line 82 of file gstdataposrc.h.

8.4.2.26 sample_offset_array_len

```
guint _GstDataRepoSrc::sample_offset_array_len
```

Definition at line 86 of file gstdataposrc.h.

8.4.2.27 sample_size

```
gsize _GstDataRepoSrc::sample_size
```

size of one sample

Definition at line 60 of file gstdataposrc.h.

8.4.2.28 shuffled_index_array

GArray* _GstDataRepoSrc::shuffled_index_array

shuffled sample index array

Definition at line 72 of file gstdataposrc.h.

8.4.2.29 src_pad

GstPad* _GstDataRepoSrc::src_pad

Definition at line 44 of file gstdataposrc.h.

8.4.2.30 start_offset

guint64 _GstDataRepoSrc::start_offset

start offset to read

Definition at line 52 of file gstdataposrc.h.

8.4.2.31 start_sample_index

guint _GstDataRepoSrc::start_sample_index

start index of sample to read, in case of image, the starting index of the numbered files

Definition at line 67 of file gstdataposrc.h.

8.4.2.32 stop_sample_index

guint _GstDataRepoSrc::stop_sample_index

stop index of sample to read, in case of image, the stopping index of the numbered files

Definition at line 68 of file gstdataposrc.h.

8.4.2.33 `successful_read`

```
gboolean _GstDataRepoSrc::successful_read
```

used for checking EOS when reading more than one images(multi-files) from a path

Definition at line 47 of file `gstdatareposrc.h`.

8.4.2.34 `tensor_count_array`

```
JSONArray* _GstDataRepoSrc::tensor_count_array
```

array for the number of cumulative tensors

Definition at line 84 of file `gstdatareposrc.h`.

8.4.2.35 `tensor_count_array_len`

```
guint _GstDataRepoSrc::tensor_count_array_len
```

Definition at line 88 of file `gstdatareposrc.h`.

8.4.2.36 `tensor_size_array`

```
JSONArray* _GstDataRepoSrc::tensor_size_array
```

size array of flexible tensor to be stored in a Gstbuffer

Definition at line 83 of file `gstdatareposrc.h`.

8.4.2.37 `tensor_size_array_len`

```
guint _GstDataRepoSrc::tensor_size_array_len
```

Definition at line 87 of file `gstdatareposrc.h`.

8.4.2.38 tensors_offset

```
gsize _GstDataRepoSrc::tensors_offset[NNS_TENSOR_SIZE_LIMIT]
```

each tensors offset in a sample

Definition at line 55 of file gstdataposrc.h.

8.4.2.39 tensors_seq

```
guint _GstDataRepoSrc::tensors_seq[NNS_TENSOR_SIZE_LIMIT]
```

tensors sequence in a sample that will be read into gstbuffer

Definition at line 75 of file gstdataposrc.h.

8.4.2.40 tensors_seq_cnt

```
guint _GstDataRepoSrc::tensors_seq_cnt
```

Definition at line 76 of file gstdataposrc.h.

8.4.2.41 tensors_seq_str

```
gchar* _GstDataRepoSrc::tensors_seq_str
```

tensors in a sample are read into gstBuffer according to tensors_sequence

Definition at line 66 of file gstdataposrc.h.

8.4.2.42 tensors_size

```
gsize _GstDataRepoSrc::tensors_size[NNS_TENSOR_SIZE_LIMIT]
```

each tensors size in a sample

Definition at line 54 of file gstdataposrc.h.

8.4.2.43 total_samples

```
guint _GstDataRepoSrc::total_samples
```

The number of total samples

Definition at line 58 of file `gstdatareposrc.h`.

The documentation for this struct was generated from the following file:

- [datarepo/gstdatareposrc.h](#)

8.5 _GstDataRepoSrcClass Struct Reference

GstDataRepoSrcClass data structure.

```
#include <gstdatareposrc.h>
```

Public Attributes

- GstPushSrcClass [parent_class](#)

8.5.1 Detailed Description

GstDataRepoSrcClass data structure.

Definition at line 98 of file `gstdatareposrc.h`.

8.5.2 Member Data Documentation

8.5.2.1 parent_class

```
GstPushSrcClass _GstDataRepoSrcClass::parent_class
```

Definition at line 99 of file `gstdatareposrc.h`.

The documentation for this struct was generated from the following file:

- [datarepo/gstdatareposrc.h](#)

8.6 _GstEdgeSink Struct Reference

GstEdgeSink data structure.

```
#include <edge_sink.h>
```

Public Attributes

- GstBaseSink [element](#)
- gchar * [host](#)
- guint16 [port](#)
- gchar * [dest_host](#)
- guint16 [dest_port](#)
- gchar * [topic](#)
- nns_edge_connect_type_e [connect_type](#)
- nns_edge_h [edge_h](#)
- gboolean [wait_connection](#)
- guint64 [connection_timeout](#)
- gchar * [custom_lib](#)
- gboolean [is_connected](#)
- GMutex [lock](#)
- GCond [cond](#)

8.6.1 Detailed Description

GstEdgeSink data structure.

Definition at line 42 of file `edge_sink.h`.

8.6.2 Member Data Documentation

8.6.2.1 cond

GCond `_GstEdgeSink::cond`

Definition at line 60 of file `edge_sink.h`.

8.6.2.2 connect_type

nns_edge_connect_type_e `_GstEdgeSink::connect_type`

Definition at line 52 of file `edge_sink.h`.

8.6.2.3 connection_timeout

guint64 _GstEdgeSink::connection_timeout

Definition at line 55 of file edge_sink.h.

8.6.2.4 custom_lib

gchar* _GstEdgeSink::custom_lib

Definition at line 57 of file edge_sink.h.

8.6.2.5 dest_host

gchar* _GstEdgeSink::dest_host

Definition at line 48 of file edge_sink.h.

8.6.2.6 dest_port

guint16 _GstEdgeSink::dest_port

Definition at line 49 of file edge_sink.h.

8.6.2.7 edge_h

nns_edge_h _GstEdgeSink::edge_h

Definition at line 53 of file edge_sink.h.

8.6.2.8 element

GstBaseSink _GstEdgeSink::element

Definition at line 44 of file edge_sink.h.

8.6.2.9 `host`

`gchar* _GstEdgeSink::host`

Definition at line 46 of file `edge_sink.h`.

8.6.2.10 `is_connected`

`gboolean _GstEdgeSink::is_connected`

Definition at line 58 of file `edge_sink.h`.

8.6.2.11 `lock`

`GMutex _GstEdgeSink::lock`

Definition at line 59 of file `edge_sink.h`.

8.6.2.12 `port`

`guint16 _GstEdgeSink::port`

Definition at line 47 of file `edge_sink.h`.

8.6.2.13 `topic`

`gchar* _GstEdgeSink::topic`

Definition at line 50 of file `edge_sink.h`.

8.6.2.14 `wait_connection`

`gboolean _GstEdgeSink::wait_connection`

Definition at line 54 of file `edge_sink.h`.

The documentation for this struct was generated from the following file:

- [edge/edge_sink.h](#)

8.7 `_GstEdgeSinkClass` Struct Reference

`GstEdgeSinkClass` data structure.

```
#include <edge_sink.h>
```

Public Attributes

- `GstBaseSinkClass` [parent_class](#)

8.7.1 Detailed Description

`GstEdgeSinkClass` data structure.

Definition at line 66 of file `edge_sink.h`.

8.7.2 Member Data Documentation

8.7.2.1 `parent_class`

```
GstBaseSinkClass _GstEdgeSinkClass::parent_class
```

parent class

Definition at line 68 of file `edge_sink.h`.

The documentation for this struct was generated from the following file:

- [edge/edge_sink.h](#)

8.8 `_GstEdgeSrc` Struct Reference

`GstEdgeSrc` data structure.

```
#include <edge_src.h>
```

Public Attributes

- `GstBaseSrc` [element](#)
- `gchar *` [dest_host](#)
- `guint16` [dest_port](#)
- `gchar *` [topic](#)
- `nns_edge_connect_type_e` [connect_type](#)
- `nns_edge_h` [edge_h](#)
- `GAsyncQueue *` [msg_queue](#)
- `gboolean` [playing](#)
- `gchar *` [custom_lib](#)

8.8.1 Detailed Description

GstEdgeSrc data structure.

Definition at line 42 of file edge_src.h.

8.8.2 Member Data Documentation

8.8.2.1 connect_type

nns_edge_connect_type_e _GstEdgeSrc::connect_type

Definition at line 50 of file edge_src.h.

8.8.2.2 custom_lib

gchar* _GstEdgeSrc::custom_lib

Definition at line 56 of file edge_src.h.

8.8.2.3 dest_host

gchar* _GstEdgeSrc::dest_host

Definition at line 46 of file edge_src.h.

8.8.2.4 dest_port

guint16 _GstEdgeSrc::dest_port

Definition at line 47 of file edge_src.h.

8.8.2.5 edge_h

nns_edge_h _GstEdgeSrc::edge_h

Definition at line 51 of file edge_src.h.

8.8.2.6 element

```
GstBaseSrc _GstEdgeSrc::element
```

Definition at line 44 of file edge_src.h.

8.8.2.7 msg_queue

```
GAsyncQueue* _GstEdgeSrc::msg_queue
```

Definition at line 52 of file edge_src.h.

8.8.2.8 playing

```
gboolean _GstEdgeSrc::playing
```

Definition at line 54 of file edge_src.h.

8.8.2.9 topic

```
gchar* _GstEdgeSrc::topic
```

Definition at line 48 of file edge_src.h.

The documentation for this struct was generated from the following file:

- [edge/edge_src.h](#)

8.9 _GstEdgeSrcClass Struct Reference

GstEdgeSrcClass data structure.

```
#include <edge_src.h>
```

Public Attributes

- GstBaseSrcClass [parent_class](#)

8.9.1 Detailed Description

`GstEdgeSrcClass` data structure.

Definition at line 62 of file `edge_src.h`.

8.9.2 Member Data Documentation

8.9.2.1 `parent_class`

`GstBaseSrcClass` `_GstEdgeSrcClass::parent_class`

Definition at line 64 of file `edge_src.h`.

The documentation for this struct was generated from the following file:

- [edge/edge_src.h](#)

8.10 `_GstJoin` Struct Reference

Internal data structure for join instances.

```
#include <gstjoin.h>
```

Public Attributes

- `GstElement` [element](#)
- `GstPad` * [srcpad](#)
- `GstPad` * [active_sinkpad](#)
- `guint` [n_pads](#)
- `guint` [padcount](#)
- `gboolean` [have_group_id](#)
- `GMutex` [lock](#)
- `GCond` [cond](#)

8.10.1 Detailed Description

Internal data structure for join instances.

Definition at line 36 of file `gstjoin.h`.

8.10.2 Member Data Documentation

8.10.2.1 active_sinkpad

GstPad* _GstJoin::active_sinkpad

Definition at line 42 of file gstjoin.h.

8.10.2.2 cond

GCond _GstJoin::cond

Definition at line 49 of file gstjoin.h.

8.10.2.3 element

GstElement _GstJoin::element

Definition at line 38 of file gstjoin.h.

8.10.2.4 have_group_id

gboolean _GstJoin::have_group_id

Definition at line 46 of file gstjoin.h.

8.10.2.5 lock

GMutex _GstJoin::lock

Definition at line 48 of file gstjoin.h.

8.10.2.6 n_pads

guint _GstJoin::n_pads

Definition at line 43 of file gstjoin.h.

8.10.2.7 `padcount`

```
guint _GstJoin::padcount
```

Definition at line 44 of file `gstjoin.h`.

8.10.2.8 `srcpad`

```
GstPad* _GstJoin::srcpad
```

Definition at line 40 of file `gstjoin.h`.

The documentation for this struct was generated from the following file:

- [join/gstjoin.h](#)

8.11 `_GstJoinClass` Struct Reference

`GstJoinClass` inherits `GstElementClass`.

```
#include <gstjoin.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.11.1 Detailed Description

`GstJoinClass` inherits `GstElementClass`.

Definition at line 55 of file `gstjoin.h`.

8.11.2 Member Data Documentation

8.11.2.1 `parent_class`

```
GstElementClass _GstJoinClass::parent_class
```

Definition at line 57 of file `gstjoin.h`.

The documentation for this struct was generated from the following file:

- [join/gstjoin.h](#)

8.12 `_GstJoinPad` Struct Reference

`GstJoinPad` data structure.

Public Attributes

- `GstPad` [parent](#)
- `guint` [group_id](#)
- `GstSegment` [segment](#)
- `guint32` [segment_seqnum](#)

8.12.1 Detailed Description

`GstJoinPad` data structure.

Definition at line 93 of file `gstjoin.c`.

8.12.2 Member Data Documentation

8.12.2.1 `group_id`

`guint` `_GstJoinPad::group_id`

Definition at line 97 of file `gstjoin.c`.

8.12.2.2 `parent`

`GstPad` `_GstJoinPad::parent`

Definition at line 95 of file `gstjoin.c`.

8.12.2.3 `segment`

`GstSegment` `_GstJoinPad::segment`

Definition at line 99 of file `gstjoin.c`.

8.12.2.4 `segment_seqnum`

```
guint32 _GstJoinPad::segment_seqnum
```

Definition at line 100 of file `gstjoin.c`.

The documentation for this struct was generated from the following file:

- [join/gstjoin.c](#)

8.13 `_GstJoinPadClass` Struct Reference

[_GstJoinPadClass](#) data structure

Public Attributes

- `GstPadClass` [parent](#)

8.13.1 Detailed Description

[_GstJoinPadClass](#) data structure

Definition at line 106 of file `gstjoin.c`.

8.13.2 Member Data Documentation

8.13.2.1 `parent`

```
GstPadClass _GstJoinPadClass::parent
```

Definition at line 108 of file `gstjoin.c`.

The documentation for this struct was generated from the following file:

- [join/gstjoin.c](#)

8.14 `_GstMQTTMessageHdr` Struct Reference

Defined a custom data type, `GstMQTTMessageHdr`.

```
#include <mqttcommon.h>
```

Public Attributes

```

• union {
    struct {
        guint num_mems
        gsize size_mems [GST_MQTT_MAX_NUM_MEMS]
        gint64 base_time_epoch
        gint64 sent_time_epoch
        GstClockTime duration
        GstClockTime dts
        GstClockTime pts
        gchar gst_caps_str [GST_MQTT_MAX_LEN_GST_CAPS_STR]
    }
    guint8 _reserved_hdr [GST_MQTT_LEN_MSG_HDR]
};

```

8.14.1 Detailed Description

Defined a custom data type, GstMQTTMessageHdr.

GstMQTTMessageHdr contains the information needed to parse the message data at the subscriber side and is prepended to the original message data at the publisher side.

Definition at line 49 of file mqttcommon.h.

8.14.2 Member Data Documentation

8.14.2.1 "@6

```
union { ... }
```

8.14.2.2 _reserved_hdr

```
guint8 _GstMQTTMessageHdr::_reserved_hdr [GST_MQTT_LEN_MSG_HDR]
```

Definition at line 61 of file mqttcommon.h.

8.14.2.3 base_time_epoch

```
gint64 _GstMQTTMessageHdr::base_time_epoch
```

Definition at line 54 of file mqttcommon.h.

8.14.2.4 dts

GstClockTime _GstMQTTMessageHdr::dts

Definition at line 57 of file mqttcommon.h.

8.14.2.5 duration

GstClockTime _GstMQTTMessageHdr::duration

Definition at line 56 of file mqttcommon.h.

8.14.2.6 gst_caps_str

gchar _GstMQTTMessageHdr::gst_caps_str[GST_MQTT_MAX_LEN_GST_CAPS_STR]

Definition at line 59 of file mqttcommon.h.

8.14.2.7 num_mems

guint _GstMQTTMessageHdr::num_mems

Definition at line 52 of file mqttcommon.h.

8.14.2.8 pts

GstClockTime _GstMQTTMessageHdr::pts

Definition at line 58 of file mqttcommon.h.

8.14.2.9 sent_time_epoch

gint64 _GstMQTTMessageHdr::sent_time_epoch

Definition at line 55 of file mqttcommon.h.

8.14.2.10 size_mems

```
gsize _GstMQTTMessageHdr::size_mems [GST_MQTT_MAX_NUM_MEMS]
```

Definition at line 53 of file mqttcommon.h.

The documentation for this struct was generated from the following file:

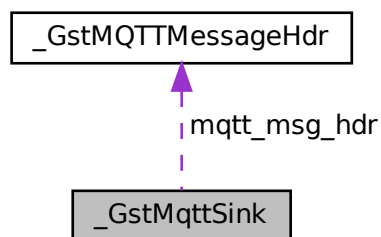
- [mqtt/mqttcommon.h](#)

8.15 _GstMqttSink Struct Reference

GstMqttSink data structure.

```
#include <mqttsink.h>
```

Collaboration diagram for _GstMqttSink:



Public Attributes

- `GstBaseSink` [parent](#)
- `GstCaps` * [in_caps](#)
- `gint` [num_buffers](#)
- `gsize` [max_msg_buf_size](#)
- `GQuark` [gquark_err_tag](#)
- `GError` * [err](#)
- `gint64` [base_time_epoch](#)
- `gchar` * [mqtt_client_id](#)
- `gchar` * [mqtt_host_address](#)
- `gchar` * [mqtt_host_port](#)
- `gchar` * [mqtt_topic](#)
- `gulong` [mqtt_pub_wait_timeout](#)
- `GMutex` [mqtt_sink_mutex](#)
- `GCond` [mqtt_sink_gcond](#)
- [mqtt_sink_state_t](#) [mqtt_sink_state](#)
- `gboolean` [debug](#)

- gint [mqtt_qos](#)
- gboolean [mqtt_ntp_sync](#)
- guint [mqtt_ntp_num_srvs](#)
- gchar * [mqtt_ntp_srvs](#)
- gchar ** [mqtt_ntp_hnames](#)
- guint16 * [mqtt_ntp_ports](#)
- gboolean [is_connected](#)
- [mqtt_get_unix_epoch](#) [get_epoch_func](#)
- [GstMQTTMessageHdr](#) [mqtt_msg_hdr](#)
- gpointer [mqtt_msg_buf](#)
- gsize [mqtt_msg_buf_size](#)
- MQTTAsync [mqtt_client_handle](#)
- MQTTAsync_connectOptions [mqtt_conn_opts](#)
- MQTTAsync_responseOptions [mqtt_respn_opts](#)

8.15.1 Detailed Description

GstMqttSink data structure.

GstMqttSink inherits GstBaseSink.

Definition at line 60 of file `mqttsink.h`.

8.15.2 Member Data Documentation

8.15.2.1 base_time_epoch

```
gint64 _GstMqttSink::base_time_epoch
```

Definition at line 67 of file `mqttsink.h`.

8.15.2.2 debug

```
gboolean _GstMqttSink::debug
```

Definition at line 76 of file `mqttsink.h`.

8.15.2.3 err

```
GError* _GstMqttSink::err
```

Definition at line 66 of file `mqttsink.h`.

8.15.2.4 `get_epoch_func`

`mqtt_get_unix_epoch` `_GstMqttSink::get_epoch_func`

Definition at line 85 of file `mqttpsink.h`.

8.15.2.5 `gquark_err_tag`

`GQuark` `_GstMqttSink::gquark_err_tag`

Definition at line 65 of file `mqttpsink.h`.

8.15.2.6 `in_caps`

`GstCaps*` `_GstMqttSink::in_caps`

Definition at line 62 of file `mqttpsink.h`.

8.15.2.7 `is_connected`

`gboolean` `_GstMqttSink::is_connected`

Definition at line 83 of file `mqttpsink.h`.

8.15.2.8 `max_msg_buf_size`

`gsize` `_GstMqttSink::max_msg_buf_size`

Definition at line 64 of file `mqttpsink.h`.

8.15.2.9 `mqtt_client_handle`

`MQTTAsync` `_GstMqttSink::mqtt_client_handle`

Definition at line 91 of file `mqttpsink.h`.

8.15.2.10 mqtt_client_id

`gchar* _GstMqttSink::mqtt_client_id`

Definition at line 68 of file `mqttpsink.h`.

8.15.2.11 mqtt_conn_opts

`MQTTAsync_connectOptions _GstMqttSink::mqtt_conn_opts`

Definition at line 92 of file `mqttpsink.h`.

8.15.2.12 mqtt_host_address

`gchar* _GstMqttSink::mqtt_host_address`

Definition at line 69 of file `mqttpsink.h`.

8.15.2.13 mqtt_host_port

`gchar* _GstMqttSink::mqtt_host_port`

Definition at line 70 of file `mqttpsink.h`.

8.15.2.14 mqtt_msg_buf

`gpointer _GstMqttSink::mqtt_msg_buf`

Definition at line 88 of file `mqttpsink.h`.

8.15.2.15 mqtt_msg_buf_size

`gsize _GstMqttSink::mqtt_msg_buf_size`

Definition at line 89 of file `mqttpsink.h`.

8.15.2.16 mqtt_msg_hdr

`GstMQTTMessageHdr` `_GstMqttSink::mqtt_msg_hdr`

Definition at line 87 of file `mqttpsink.h`.

8.15.2.17 mqtt_ntp_hnames

`gchar**` `_GstMqttSink::mqtt_ntp_hnames`

Definition at line 81 of file `mqttpsink.h`.

8.15.2.18 mqtt_ntp_num_srvs

`guint` `_GstMqttSink::mqtt_ntp_num_srvs`

Definition at line 79 of file `mqttpsink.h`.

8.15.2.19 mqtt_ntp_ports

`guint16*` `_GstMqttSink::mqtt_ntp_ports`

Definition at line 82 of file `mqttpsink.h`.

8.15.2.20 mqtt_ntp_srvs

`gchar*` `_GstMqttSink::mqtt_ntp_srvs`

Definition at line 80 of file `mqttpsink.h`.

8.15.2.21 mqtt_ntp_sync

`gboolean` `_GstMqttSink::mqtt_ntp_sync`

Definition at line 78 of file `mqttpsink.h`.

8.15.2.22 mqtt_pub_wait_timeout

gulong _GstMqttSink::mqtt_pub_wait_timeout

Definition at line 72 of file mqttpsink.h.

8.15.2.23 mqtt_qos

gint _GstMqttSink::mqtt_qos

Definition at line 77 of file mqttpsink.h.

8.15.2.24 mqtt_respn_opts

MQTTAsync_responseOptions _GstMqttSink::mqtt_respn_opts

Definition at line 93 of file mqttpsink.h.

8.15.2.25 mqtt_sink_gcond

GCCond _GstMqttSink::mqtt_sink_gcond

Definition at line 74 of file mqttpsink.h.

8.15.2.26 mqtt_sink_mutex

GMutex _GstMqttSink::mqtt_sink_mutex

Definition at line 73 of file mqttpsink.h.

8.15.2.27 mqtt_sink_state

mqtt_sink_state_t _GstMqttSink::mqtt_sink_state

Definition at line 75 of file mqttpsink.h.

8.15.2.28 mqtt_topic

```
gchar* _GstMqttSink::mqtt_topic
```

Definition at line 71 of file mqttsink.h.

8.15.2.29 num_buffers

```
gint _GstMqttSink::num_buffers
```

Definition at line 63 of file mqttsink.h.

8.15.2.30 parent

```
GstBaseSink _GstMqttSink::parent
```

Definition at line 61 of file mqttsink.h.

The documentation for this struct was generated from the following file:

- [mqtt/mqttsink.h](#)

8.16 _GstMqttSinkClass Struct Reference

GstMqttSinkClass data structure.

```
#include <mqttsink.h>
```

Public Attributes

- GstBaseSinkClass [parent_class](#)

8.16.1 Detailed Description

GstMqttSinkClass data structure.

GstMqttSinkClass inherits GstBaseSinkClass.

Definition at line 101 of file mqttsink.h.

8.16.2 Member Data Documentation

8.16.2.1 parent_class

```
GstBaseSinkClass _GstMqttSinkClass::parent_class
```

Definition at line 102 of file mqttsink.h.

The documentation for this struct was generated from the following file:

- [mqtt/mqttsink.h](#)

8.17 _GstMqttSrc Struct Reference

GstMqttSrc data structure.

```
#include <mqttsrc.h>
```

Public Attributes

- GstBaseSrc [parent](#)
- GstCaps * [caps](#)
- GQuark [gquark_err_tag](#)
- GError * [err](#)
- gint64 [base_time_epoch](#)
- GstClockTime [latency](#)
- gchar * [mqtt_client_id](#)
- gchar * [mqtt_host_address](#)
- gchar * [mqtt_host_port](#)
- gchar * [mqtt_topic](#)
- gint64 [mqtt_sub_timeout](#)
- gboolean [debug](#)
- gboolean [is_live](#)
- guint64 [num_dumped](#)
- gint [mqtt_qos](#)
- GAsyncQueue * [aqueue](#)
- GMutex [mqtt_src_mutex](#)
- GCond [mqtt_src_gcond](#)
- gboolean [is_connected](#)
- gboolean [is_subscribed](#)
- MQTTAsync [mqtt_client_handle](#)
- MQTTAsync_connectOptions [mqtt_conn_opts](#)
- MQTTAsync_responseOptions [mqtt_respn_opts](#)

8.17.1 Detailed Description

GstMqttSrc data structure.

GstMqttSrc inherits GstBaseSrc.

Definition at line 46 of file mqttsrc.h.

8.17.2 Member Data Documentation

8.17.2.1 aqueue

GAsyncQueue* _GstMqttSrc::aqueue

Definition at line 63 of file mqttsrc.h.

8.17.2.2 base_time_epoch

gint64 _GstMqttSrc::base_time_epoch

Definition at line 51 of file mqttsrc.h.

8.17.2.3 caps

GstCaps* _GstMqttSrc::caps

Definition at line 48 of file mqttsrc.h.

8.17.2.4 debug

gboolean _GstMqttSrc::debug

Definition at line 58 of file mqttsrc.h.

8.17.2.5 err

GError* _GstMqttSrc::err

Definition at line 50 of file mqttsrc.h.

8.17.2.6 gquark_err_tag

GQuark _GstMqttSrc::gquark_err_tag

Definition at line 49 of file mqttsrc.h.

8.17.2.7 is_connected

gboolean _GstMqttSrc::is_connected

Definition at line 66 of file mqttsrc.h.

8.17.2.8 is_live

gboolean _GstMqttSrc::is_live

Definition at line 59 of file mqttsrc.h.

8.17.2.9 is_subscribed

gboolean _GstMqttSrc::is_subscribed

Definition at line 67 of file mqttsrc.h.

8.17.2.10 latency

GstClockTime _GstMqttSrc::latency

Definition at line 52 of file mqttsrc.h.

8.17.2.11 mqtt_client_handle

MQTTAsync_GstMqttSrc::mqtt_client_handle

Definition at line 69 of file mqttsrc.h.

8.17.2.12 mqtt_client_id

gchar* _GstMqttSrc::mqtt_client_id

Definition at line 53 of file mqttsrc.h.

8.17.2.13 mqtt_conn_opts

MQTTAsync_connectOptions _GstMqttSrc::mqtt_conn_opts

Definition at line 70 of file mqttsrc.h.

8.17.2.14 mqtt_host_address

gchar* _GstMqttSrc::mqtt_host_address

Definition at line 54 of file mqttsrc.h.

8.17.2.15 mqtt_host_port

gchar* _GstMqttSrc::mqtt_host_port

Definition at line 55 of file mqttsrc.h.

8.17.2.16 mqtt_qos

gint _GstMqttSrc::mqtt_qos

Definition at line 61 of file mqttsrc.h.

8.17.2.17 mqtt_respn_opts

MQTTAsync_responseOptions _GstMqttSrc::mqtt_respn_opts

Definition at line 71 of file mqttsrc.h.

8.17.2.18 mqtt_src_gcond

GCond _GstMqttSrc::mqtt_src_gcond

Definition at line 65 of file mqttsrc.h.

8.17.2.19 mqtt_src_mutex

GMutex _GstMqttSrc::mqtt_src_mutex

Definition at line 64 of file mqttsrc.h.

8.17.2.20 mqtt_sub_timeout

gint64 _GstMqttSrc::mqtt_sub_timeout

Definition at line 57 of file mqttsrc.h.

8.17.2.21 mqtt_topic

gchar* _GstMqttSrc::mqtt_topic

Definition at line 56 of file mqttsrc.h.

8.17.2.22 num_dumped

guint64 _GstMqttSrc::num_dumped

Definition at line 60 of file mqttsrc.h.

8.17.2.23 parent

GstBaseSrc _GstMqttSrc::parent

Definition at line 47 of file mqttsrc.h.

The documentation for this struct was generated from the following file:

- [mqtt/mqttsrc.h](#)

8.18 _GstMqttSrcClass Struct Reference

GstMqttSrcClass data structure.

```
#include <mqttsrc.h>
```

Public Attributes

- GstBaseSrcClass [parent_class](#)

8.18.1 Detailed Description

GstMqttSrcClass data structure.

GstMqttSrcClass inherits GstBaseSrcClass.

Definition at line 79 of file mqttsrc.h.

8.18.2 Member Data Documentation

8.18.2.1 parent_class

GstBaseSrcClass _GstMqttSrcClass::parent_class

Definition at line 80 of file mqttsrc.h.

The documentation for this struct was generated from the following file:

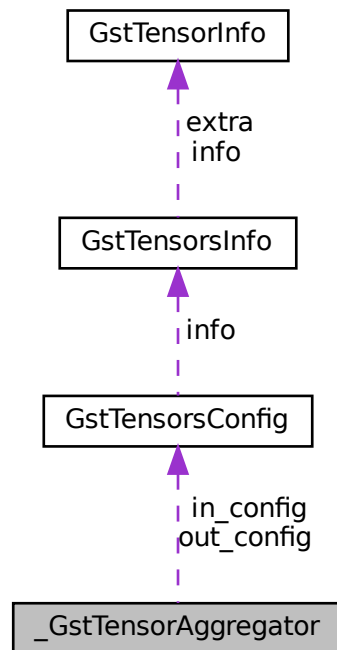
- [mqtt/mqttsrc.h](#)

8.19 _GstTensorAggregator Struct Reference

GstTensorAggregator data structure.

```
#include <gsttensor_aggregator.h>
```

Collaboration diagram for _GstTensorAggregator:



Public Attributes

- GstElement [element](#)
- GstPad * [sinkpad](#)
- GstPad * [srcpad](#)
- gboolean [silent](#)
- gboolean [concat](#)
- guint [frames_in](#)
- guint [frames_out](#)
- guint [frames_flush](#)
- guint [frames_dim](#)
- GHashTable * [adapter_table](#)
- gboolean [tensor_configured](#)
- [GstTensorsConfig](#) [in_config](#)
- [GstTensorsConfig](#) [out_config](#)

8.19.1 Detailed Description

GstTensorAggregator data structure.

Definition at line 52 of file gsttensor_aggregator.h.

8.19.2 Member Data Documentation

8.19.2.1 adapter_table

```
GHashTable* _GstTensorAggregator::adapter_table
```

adapt incoming tensor

Definition at line 66 of file gsttensor_aggregator.h.

8.19.2.2 concat

```
gboolean _GstTensorAggregator::concat
```

true to concatenate output buffer

Definition at line 60 of file gsttensor_aggregator.h.

8.19.2.3 element

```
GstElement _GstTensorAggregator::element
```

parent object

Definition at line 54 of file gsttensor_aggregator.h.

8.19.2.4 frames_dim

```
guint _GstTensorAggregator::frames_dim
```

index of frames in tensor dimension

Definition at line 64 of file gsttensor_aggregator.h.

8.19.2.5 frames_flush

`guint _GstTensorAggregator::frames_flush`

number of frames to flush

Definition at line 63 of file `gsttensor_aggregator.h`.

8.19.2.6 frames_in

`guint _GstTensorAggregator::frames_in`

number of frames in input buffer

Definition at line 61 of file `gsttensor_aggregator.h`.

8.19.2.7 frames_out

`guint _GstTensorAggregator::frames_out`

number of frames in output buffer

Definition at line 62 of file `gsttensor_aggregator.h`.

8.19.2.8 in_config

`GstTensorsConfig _GstTensorAggregator::in_config`

input tensor info

Definition at line 69 of file `gsttensor_aggregator.h`.

8.19.2.9 out_config

`GstTensorsConfig _GstTensorAggregator::out_config`

output tensor info

Definition at line 70 of file `gsttensor_aggregator.h`.

8.19.2.10 silent

```
gboolean _GstTensorAggregator::silent
```

true to print minimized log

Definition at line 59 of file `gsttensor_aggregator.h`.

8.19.2.11 sinkpad

```
GstPad* _GstTensorAggregator::sinkpad
```

sink pad

Definition at line 56 of file `gsttensor_aggregator.h`.

8.19.2.12 srcpad

```
GstPad* _GstTensorAggregator::srcpad
```

src pad

Definition at line 57 of file `gsttensor_aggregator.h`.

8.19.2.13 tensor_configured

```
gboolean _GstTensorAggregator::tensor_configured
```

True if already successfully configured tensor metadata

Definition at line 68 of file `gsttensor_aggregator.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_aggregator.h](#)

8.20 _GstTensorAggregatorClass Struct Reference

`GstTensorAggregatorClass` data structure.

```
#include <gsttensor_aggregator.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.20.1 Detailed Description

`GstTensorAggregatorClass` data structure.

Definition at line 76 of file `gsttensor_aggregator.h`.

8.20.2 Member Data Documentation

8.20.2.1 `parent_class`

```
GstElementClass _GstTensorAggregatorClass::parent_class
```

parent class

Definition at line 78 of file `gsttensor_aggregator.h`.

The documentation for this struct was generated from the following file:

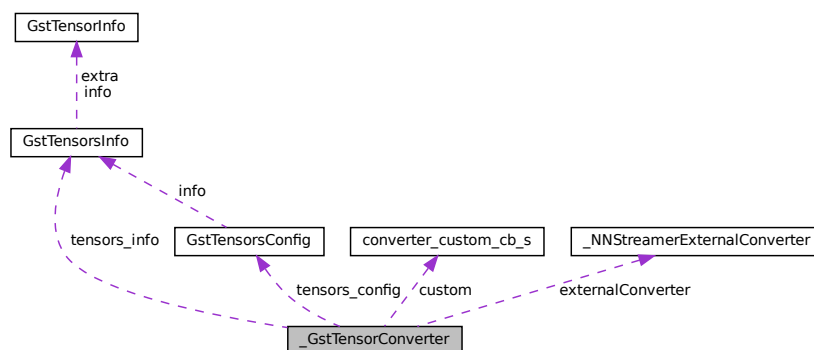
- `nntstreamer/elements/gsttensor_aggregator.h`

8.21 `_GstTensorConverter` Struct Reference

Internal data structure for `tensor_converter` instances.

```
#include <gsttensor_converter.h>
```

Collaboration diagram for `_GstTensorConverter`:



Public Attributes

- GstElement [element](#)
- GstPad * [sinkpad](#)
- GstPad * [srcpad](#)
- gboolean [silent](#)
- gboolean [set_timestamp](#)
- guint [frames_per_tensor](#)
- [GstTensorsInfo](#) [tensors_info](#)
- GHashTable * [adapter_table](#)
- [media_type](#) [in_media_type](#)
- const [NNStreamerExternalConverter](#) * [externalConverter](#)
- gsize [frame_size](#)
- gboolean [remove_padding](#)
- gboolean [tensors_configured](#)
- [GstTensorsConfig](#) [tensors_config](#)
- gboolean [have_segment](#)
- gboolean [need_segment](#)
- GstSegment [segment](#)
- GstClockTime [old_timestamp](#)
- [tensor_converter_mode](#) [mode](#)
- gchar * [mode_option](#)
- gchar * [ext_fw](#)
- [converter_custom_cb_s](#) [custom](#)
- gboolean [do_not_append_header](#)
- void * [priv_data](#)

8.21.1 Detailed Description

Internal data structure for `tensor_converter` instances.

Definition at line 75 of file `gsttensor_converter.h`.

8.21.2 Member Data Documentation

8.21.2.1 `adapter_table`

```
GHashTable* _GstTensorConverter::adapter_table
```

adapt incoming media stream

Definition at line 87 of file `gsttensor_converter.h`.

8.21.2.2 `custom`

`converter_custom_cb_s` `_GstTensorConverter::custom`

Definition at line 106 of file `gsttensor_converter.h`.

8.21.2.3 `do_not_append_header`

`gboolean` `_GstTensorConverter::do_not_append_header`

Definition at line 107 of file `gsttensor_converter.h`.

8.21.2.4 `element`

`GstElement` `_GstTensorConverter::element`

parent object

Definition at line 77 of file `gsttensor_converter.h`.

8.21.2.5 `ext_fw`

`gchar*` `_GstTensorConverter::ext_fw`

tensor converter custom mode framework

Definition at line 105 of file `gsttensor_converter.h`.

8.21.2.6 `externalConverter`

`const` `NNStreamerExternalConverter*` `_GstTensorConverter::externalConverter`

`ExternalConverter` is used if `in_media_type == _NNS_MEDIA_PLUGINS`

Definition at line 91 of file `gsttensor_converter.h`.

8.21.2.7 frame_size

`gsize _GstTensorConverter::frame_size`

size of one frame

Definition at line 93 of file `gsttensor_converter.h`.

8.21.2.8 frames_per_tensor

`guint _GstTensorConverter::frames_per_tensor`

number of frames in output tensor

Definition at line 84 of file `gsttensor_converter.h`.

8.21.2.9 have_segment

`gboolean _GstTensorConverter::have_segment`

True if received segment

Definition at line 98 of file `gsttensor_converter.h`.

8.21.2.10 in_media_type

`media_type _GstTensorConverter::in_media_type`

incoming media type

Definition at line 89 of file `gsttensor_converter.h`.

8.21.2.11 mode

`tensor_converter_mode _GstTensorConverter::mode`

tensor converter operating mode

Definition at line 103 of file `gsttensor_converter.h`.

8.21.2.12 mode_option

`gchar* _GstTensorConverter::mode_option`

tensor converter mode option

Definition at line 104 of file `gsttensor_converter.h`.

8.21.2.13 need_segment

`gboolean _GstTensorConverter::need_segment`

True to handle seg event

Definition at line 99 of file `gsttensor_converter.h`.

8.21.2.14 old_timestamp

`GstClockTime _GstTensorConverter::old_timestamp`

timestamp at prev buffer

Definition at line 101 of file `gsttensor_converter.h`.

8.21.2.15 priv_data

`void* _GstTensorConverter::priv_data`

plugin's private data

Definition at line 109 of file `gsttensor_converter.h`.

8.21.2.16 remove_padding

`gboolean _GstTensorConverter::remove_padding`

If true, zero-padding must be removed

Definition at line 94 of file `gsttensor_converter.h`.

8.21.2.17 segment

GstSegment _GstTensorConverter::segment

Segment, supposed time format

Definition at line 100 of file gsttensor_converter.h.

8.21.2.18 set_timestamp

gboolean _GstTensorConverter::set_timestamp

true to set timestamp when received a buffer with invalid timestamp

Definition at line 83 of file gsttensor_converter.h.

8.21.2.19 silent

gboolean _GstTensorConverter::silent

true to print minimized log

Definition at line 82 of file gsttensor_converter.h.

8.21.2.20 sinkpad

GstPad* _GstTensorConverter::sinkpad

sink pad

Definition at line 79 of file gsttensor_converter.h.

8.21.2.21 srcpad

GstPad* _GstTensorConverter::srcpad

src pad

Definition at line 80 of file gsttensor_converter.h.

8.21.2.22 tensors_config

`GstTensorsConfig` `_GstTensorConverter::tensors_config`

output tensors info

Definition at line 96 of file `gsttensor_converter.h`.

8.21.2.23 tensors_configured

`gboolean` `_GstTensorConverter::tensors_configured`

True if already successfully configured tensors metadata

Definition at line 95 of file `gsttensor_converter.h`.

8.21.2.24 tensors_info

`GstTensorsInfo` `_GstTensorConverter::tensors_info`

data structure to get/set tensor info

Definition at line 85 of file `gsttensor_converter.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_converter.h](#)

8.22 _GstTensorConverterClass Struct Reference

`GstTensorConverterClass` data structure.

```
#include <gsttensor_converter.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.22.1 Detailed Description

`GstTensorConverterClass` data structure.

Definition at line 115 of file `gsttensor_converter.h`.

8.22.2 Member Data Documentation

8.22.2.1 parent_class

```
GstElementClass _GstTensorConverterClass::parent_class
```

parent class

Definition at line 117 of file gsttensor_converter.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_converter.h](#)

8.23 _GstTensorCrop Struct Reference

GstTensorCrop data structure.

```
#include <gsttensor_crop.h>
```

Public Attributes

- GstElement [element](#)
- GstPad * [sinkpad_raw](#)
- GstPad * [sinkpad_info](#)
- GstPad * [srcpad](#)
- gint [lateness](#)
- gboolean [silent](#)
- gboolean [send_stream_start](#)
- GstCollectPads * [collect](#)

8.23.1 Detailed Description

GstTensorCrop data structure.

Definition at line 49 of file gsttensor_crop.h.

8.23.2 Member Data Documentation

8.23.2.1 collect

GstCollectPads* _GstTensorCrop::collect

sink pads

Definition at line 61 of file gsttensor_crop.h.

8.23.2.2 element

GstElement _GstTensorCrop::element

parent object

Definition at line 51 of file gsttensor_crop.h.

8.23.2.3 lateness

gint _GstTensorCrop::lateness

time-diff of raw and info buffer

Definition at line 58 of file gsttensor_crop.h.

8.23.2.4 send_stream_start

gboolean _GstTensorCrop::send_stream_start

flag to send STREAM_START event

Definition at line 60 of file gsttensor_crop.h.

8.23.2.5 silent

gboolean _GstTensorCrop::silent

true to print minimized log

Definition at line 59 of file gsttensor_crop.h.

8.23.2.6 sinkpad_info

```
GstPad* _GstTensorCrop::sinkpad_info
```

sink pad (crop info)

Definition at line 54 of file gsttensor_crop.h.

8.23.2.7 sinkpad_raw

```
GstPad* _GstTensorCrop::sinkpad_raw
```

sink pad (raw data)

Definition at line 53 of file gsttensor_crop.h.

8.23.2.8 srcpad

```
GstPad* _GstTensorCrop::srcpad
```

src pad

Definition at line 55 of file gsttensor_crop.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_crop.h](#)

8.24 _GstTensorCropClass Struct Reference

GstTensorCropClass data structure.

```
#include <gsttensor_crop.h>
```

Public Attributes

- GstElementClass [parent_class](#)

8.24.1 Detailed Description

GstTensorCropClass data structure.

Definition at line 67 of file gsttensor_crop.h.

8.24.2 Member Data Documentation

8.24.2.1 parent_class

```
GstElementClass _GstTensorCropClass::parent_class
```

parent class

Definition at line 69 of file `gsttensor_crop.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_crop.h](#)

8.25 _GstTensorDebug Struct Reference

Internal data structure for `tensor_debug` instances.

```
#include <gsttensor_debug.h>
```

Public Attributes

- GstBaseTransform [element](#)
- GstPad * [sinkpad](#)
- GstPad * [srcpad](#)
- gboolean [silent](#)
- [tdbg_output_mode](#) [output_mode](#)
- [tdbg_cap_mode](#) [cap_mode](#)
- [tdbg_meta_mode](#) [meta_mode](#)

8.25.1 Detailed Description

Internal data structure for `tensor_debug` instances.

Definition at line 84 of file `gsttensor_debug.h`.

8.25.2 Member Data Documentation

8.25.2.1 cap_mode

`tdbg_cap_mode` `_GstTensorDebug::cap_mode`

Definition at line 94 of file `gsttensor_debug.h`.

8.25.2.2 element

`GstBaseTransform` `_GstTensorDebug::element`

This is the parent object

Definition at line 86 of file `gsttensor_debug.h`.

8.25.2.3 meta_mode

`tdbg_meta_mode` `_GstTensorDebug::meta_mode`

Definition at line 95 of file `gsttensor_debug.h`.

8.25.2.4 output_mode

`tdbg_output_mode` `_GstTensorDebug::output_mode`

Definition at line 93 of file `gsttensor_debug.h`.

8.25.2.5 silent

`gboolean` `_GstTensorDebug::silent`

true to print minimized log

Definition at line 91 of file `gsttensor_debug.h`.

8.25.2.6 sinkpad

`GstPad*` `_GstTensorDebug::sinkpad`

sink pad

Definition at line 88 of file `gsttensor_debug.h`.

8.25.2.7 srcpad

```
GstPad* _GstTensorDebug::srcpad
```

src pad

Definition at line 89 of file gsttensor_debug.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_debug.h](#)

8.26 _GstTensorDebugClass Struct Reference

GstTensorDebugClass data structure.

```
#include <gsttensor_debug.h>
```

Public Attributes

- GstBaseTransformClass [parent_class](#)

8.26.1 Detailed Description

GstTensorDebugClass data structure.

Definition at line 101 of file gsttensor_debug.h.

8.26.2 Member Data Documentation

8.26.2.1 parent_class

```
GstBaseTransformClass _GstTensorDebugClass::parent_class
```

parent class = transform

Definition at line 103 of file gsttensor_debug.h.

The documentation for this struct was generated from the following file:

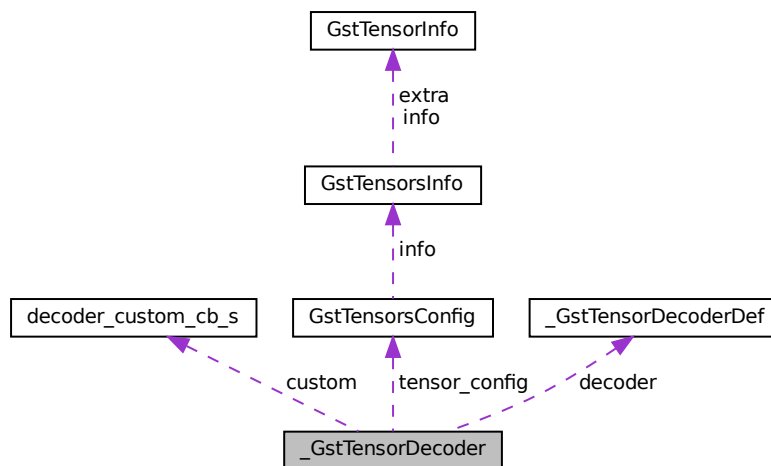
- [nnstreamer/elements/gsttensor_debug.h](#)

8.27 `_GstTensorDecoder` Struct Reference

Internal data structure for tensordec instances.

```
#include <gsttensor_decoder.h>
```

Collaboration diagram for `_GstTensorDecoder`:



Public Attributes

- GstBaseTransform [element](#)
- gboolean [negotiated](#)
- gboolean [silent](#)
- gchar * [config_path](#)
- gchar * [option](#) [TensorDecMaxOpNum]
- gboolean [configured](#)
- GstTensorsConfig [tensor_config](#)
- gboolean [is_custom](#)
- [decoder_custom_cb_s](#) [custom](#)
- const [GstTensorDecoderDef](#) * [decoder](#)
- void * [plugin_data](#)

8.27.1 Detailed Description

Internal data structure for tensordec instances.

Definition at line 67 of file `gsttensor_decoder.h`.

8.27.2 Member Data Documentation

8.27.2.1 `config_path`

```
gchar* _GstTensorDecoder::config_path
```

Path to configuration file

Definition at line 74 of file `gsttensor_decoder.h`.

8.27.2.2 `configured`

```
gboolean _GstTensorDecoder::configured
```

For Tensor TRUE if already successfully configured tensor metadata

Definition at line 78 of file `gsttensor_decoder.h`.

8.27.2.3 `custom`

```
decoder\_custom\_cb\_s _GstTensorDecoder::custom
```

Definition at line 83 of file `gsttensor_decoder.h`.

8.27.2.4 `decoder`

```
const GstTensorDecoderDef\* _GstTensorDecoder::decoder
```

Plugin object

Definition at line 85 of file `gsttensor_decoder.h`.

8.27.2.5 `element`

```
GstBaseTransform _GstTensorDecoder::element
```

This is the parent object

Definition at line 69 of file `gsttensor_decoder.h`.

8.27.2.6 is_custom

```
gboolean _GstTensorDecoder::is_custom
```

For tensor decoder custom

Definition at line 82 of file gsttensor_decoder.h.

8.27.2.7 negotiated

```
gboolean _GstTensorDecoder::negotiated
```

For transformer TRUE if tensor metadata is set

Definition at line 72 of file gsttensor_decoder.h.

8.27.2.8 option

```
gchar* _GstTensorDecoder::option[TensorDecMaxOpNum]
```

Assume we have two options

Definition at line 75 of file gsttensor_decoder.h.

8.27.2.9 plugin_data

```
void* _GstTensorDecoder::plugin_data
```

Definition at line 86 of file gsttensor_decoder.h.

8.27.2.10 silent

```
gboolean _GstTensorDecoder::silent
```

True if logging is minimized

Definition at line 73 of file gsttensor_decoder.h.

8.27.2.11 tensor_config

`GstTensorsConfig` `_GstTensorDecoder::tensor_config`

configured tensor info

Todo support tensors in the future

Definition at line 79 of file `gsttensor_decoder.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_decoder.h`

8.28 _GstTensorDecoderClass Struct Reference

`GstTensorDecoderClass` inherits `GstBaseTransformClass`.

```
#include <gsttensor_decoder.h>
```

Public Attributes

- `GstBaseTransformClass` `parent_class`

8.28.1 Detailed Description

`GstTensorDecoderClass` inherits `GstBaseTransformClass`.

Referring another child (sibling), `GstVideoFilter` (abstract class) and its child (concrete class) `GstVideoConverter`. Note that `GstTensorDecoderClass` is a concrete class; thus we need to look at both.

Definition at line 96 of file `gsttensor_decoder.h`.

8.28.2 Member Data Documentation

8.28.2.1 parent_class

`GstBaseTransformClass` `_GstTensorDecoderClass::parent_class`

Inherits `GstBaseTransformClass`

Definition at line 98 of file `gsttensor_decoder.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_decoder.h`

8.29 `_GstTensorDecoderDef` Struct Reference

Decoder definitions for different semantics of tensors This allows developers to create their own decoders.

```
#include <nnstreamer_plugin_api_decoder.h>
```

Public Attributes

- char * [modename](#)
- int(* [init](#))(void **private_data)
- void(* [exit](#))(void **private_data)
- int(* [setOption](#))(void **private_data, int opNum, const char *param)
- GstCaps *([getOutCaps](#))(void **private_data, const [GstTensorsConfig](#) *config)
- GstFlowReturn(* [decode](#))(void **private_data, const [GstTensorsConfig](#) *config, const [GstTensorMemory](#) *input, GstBuffer *outbuf)
- size_t(* [getTransformSize](#))(void **private_data, const [GstTensorsConfig](#) *config, GstCaps *caps, size_t size, GstCaps *othercaps, GstPadDirection direction)

8.29.1 Detailed Description

Decoder definitions for different semantics of tensors This allows developers to create their own decoders.

Definition at line 38 of file `nnstreamer_plugin_api_decoder.h`.

8.29.2 Member Data Documentation

8.29.2.1 `decode`

```
GstFlowReturn(* _GstTensorDecoderDef::decode) (void **private_data, const GstTensorsConfig *config, const GstTensorMemory *input, GstBuffer *outbuf)
```

The function to be called when the input tensor incomes into `tensor_decoder`. The sub-plugin should update the output buffer. `outbuf` must be allocated but empty (`gst_buffer_get_size (outbuf) == 0`).

Parameters

	<i>[in/out]</i>	<code>private_data</code> A sub-plugin may save its internal private data here. The sub-plugin is responsible for alloc/free of this pointer.
<code>in</code>	<i>config</i>	The structure of input tensor info.
<code>in</code>	<i>input</i>	The array of input tensor data. The maximum array size of input data is <code>NNS_TENSOR_SIZE_LIMIT</code> .
<code>out</code>	<i>outbuf</i>	A sub-plugin should update or append proper memory for the negotiated media type.

Returns

`GST_FLOW_OK` if OK.

Definition at line 71 of file `nnstreamer_plugin_api_decoder.h`.

8.29.2.2 exit

```
void(* _GstTensorDecoderDef::exit) (void **private_data)
```

Object destruction for the decoder.

Parameters

<i>[in/out]</i>	<code>private_data</code> A sub-plugin may save its internal private data here. The sub-plugin is responsible for alloc/free of this pointer. Normally, the sub-plugin may free the <code>private_data</code> with this function.
-----------------	---

Definition at line 48 of file `nnstreamer_plugin_api_decoder.h`.

8.29.2.3 getOutCaps

```
GstCaps*(* _GstTensorDecoderDef::getOutCaps) (void **private_data, const GstTensorsConfig *config)
```

The caller should unref the returned `GstCaps` using `gst_caps_unref()`. The sub-plugin should validate the information of input tensor and return proper media type. Note that the information of input tensor is not a fixed value and the pipeline may try different values during the cap negotiations. Do NOT allocate or fix internal data structure until decode is called.

Parameters

	<i>[in/out]</i>	<code>private_data</code> A sub-plugin may save its internal private data here. The sub-plugin is responsible for alloc/free of this pointer.
<code>in</code>	<i>config</i>	The structure of input tensor info.

Returns

`GstCaps` object describing media type.

Definition at line 61 of file `nnstreamer_plugin_api_decoder.h`.

8.29.2.4 getTransformSize

```
size_t(* _GstTensorDecoderDef::getTransformSize) (void **private_data, const GstTensorsConfig *config, GstCaps *caps, size_t size, GstCaps *othercaps, GstPadDirection direction)
```

Optional. The sub-plugin may calculate the size in bytes of a buffer. If this is NULL, `tensor_decoder` will pass the empty buffer and the sub-plugin should append the memory block when called `decode`. See `GstBaseTransform↔Class::transform_size` for the details.

Parameters

	<i>[in/out]</i>	<code>private_data</code> A sub-plugin may save its internal private data here. The sub-plugin is responsible for alloc/free of this pointer.
in	<i>config</i>	The structure of input tensor info.
in	<i>caps</i>	<code>GstCaps</code> object for the given direction.
in	<i>size</i>	The size of a buffer for the given direction.
in	<i>othercaps</i>	<code>GstCaps</code> object on the other pad for the given direction.
in	<i>direction</i>	The direction of a pad. Normally this is <code>GST_PAD_SINK</code> .

Returns

The size of a buffer.

Definition at line 82 of file `nnstreamer_plugin_api_decoder.h`.

8.29.2.5 init

```
int (* _GstTensorDecoderDef::init) (void **private_data)
```

Object initialization for the decoder.

Parameters

<i>[in/out]</i>	<code>private_data</code> A sub-plugin may save its internal private data here. The sub-plugin is responsible for alloc/free of this pointer. Normally, the sub-plugin may allocate the <code>private_data</code> with this function.
-----------------	---

Returns

TRUE if OK. FALSE if error.

Definition at line 42 of file `nnstreamer_plugin_api_decoder.h`.

8.29.2.6 modename

```
char* _GstTensorDecoderDef::modename
```

Unique decoder name. GST users choose decoders with `mode="modename"`.

Definition at line 40 of file `nnstreamer_plugin_api_decoder.h`.

8.29.2.7 setOption

```
int (* _GstTensorDecoderDef::setOption) (void **private_data, int opNum, const char *param)
```

Process with the given options. It can be called repeatedly.

Parameters

	<i>[in/out]</i>	<i>private_data</i> A sub-plugin may save its internal private data here. The sub-plugin is responsible for alloc/free of this pointer.
in	<i>opNum</i>	The index of the given options.
in	<i>param</i>	The option string. A sub-plugin should parse the string to get the proper value.

Returns

TRUE if OK. FALSE if error.

Definition at line 53 of file `nnstreamer_plugin_api_decoder.h`.

The documentation for this struct was generated from the following file:

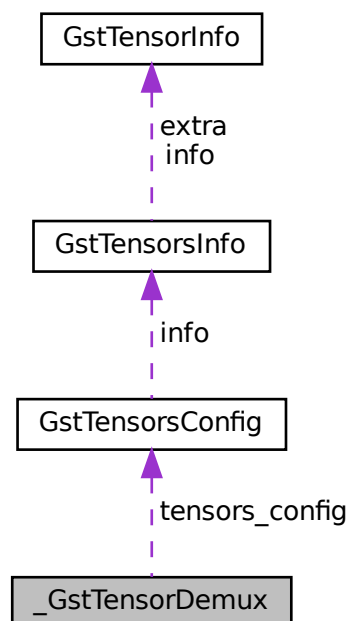
- [nnstreamer/include/nnstreamer_plugin_api_decoder.h](#)

8.30 _GstTensorDemux Struct Reference

Tensor Muxer data structure.

```
#include <gsttensor_demux.h>
```

Collaboration diagram for `_GstTensorDemux`:



Public Attributes

- GstElement [element](#)
- gboolean [silent](#)
- GstPad * [sinkpad](#)
- GSList * [srcpads](#)
- guint32 [num_srcpads](#)
- GList * [tensorpick](#)
- gboolean [have_group_id](#)
- guint [group_id](#)
- [GstTensorsConfig](#) [tensors_config](#)

8.30.1 Detailed Description

Tensor Muxer data structure.

Definition at line 50 of file `gsttensor_demux.h`.

8.30.2 Member Data Documentation

8.30.2.1 `element`

`GstElement _GstTensorDemux::element`

Definition at line 52 of file `gsttensor_demux.h`.

8.30.2.2 `group_id`

`guint _GstTensorDemux::group_id`

Definition at line 60 of file `gsttensor_demux.h`.

8.30.2.3 `have_group_id`

`gboolean _GstTensorDemux::have_group_id`

Definition at line 59 of file `gsttensor_demux.h`.

8.30.2.4 num_srcpads

```
guint32 _GstTensorDemux::num_srcpads
```

Definition at line 57 of file `gsttensor_demux.h`.

8.30.2.5 silent

```
gboolean _GstTensorDemux::silent
```

Definition at line 54 of file `gsttensor_demux.h`.

8.30.2.6 sinkpad

```
GstPad* _GstTensorDemux::sinkpad
```

Definition at line 55 of file `gsttensor_demux.h`.

8.30.2.7 srcpads

```
GSLIST* _GstTensorDemux::srcpads
```

Definition at line 56 of file `gsttensor_demux.h`.

8.30.2.8 tensorpick

```
GList* _GstTensorDemux::tensorpick
```

Definition at line 58 of file `gsttensor_demux.h`.

8.30.2.9 tensors_config

```
GstTensorsConfig _GstTensorDemux::tensors_config
```

input tensors info

Definition at line 62 of file `gsttensor_demux.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_demux.h](#)

8.31 `_GstTensorDemuxClass` Struct Reference

`GstTensorDeMuxClass` inherits `GstElementClass`.

```
#include <gsttensor_demux.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.31.1 Detailed Description

`GstTensorDeMuxClass` inherits `GstElementClass`.

Definition at line 68 of file `gsttensor_demux.h`.

8.31.2 Member Data Documentation

8.31.2.1 `parent_class`

```
GstElementClass _GstTensorDemuxClass::parent_class
```

Definition at line 70 of file `gsttensor_demux.h`.

The documentation for this struct was generated from the following file:

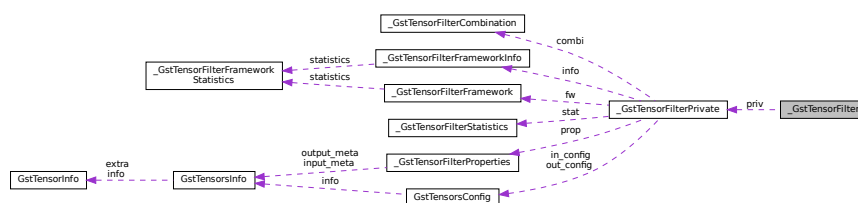
- `nntstreamer/elements/gsttensor_demux.h`

8.32 `_GstTensorFilter` Struct Reference

Internal data structure for `tensor_filter` instances.

```
#include <tensor_filter.h>
```

Collaboration diagram for `_GstTensorFilter`:



Public Attributes

- `GstBaseTransform` [element](#)
- `GstTensorFilterPrivate` [priv](#)
- `GstClockTime` [prev_ts](#)
- `GstClockTimeDiff` [throttling_delay](#)
- `GstClockTimeDiff` [throttling_accum](#)

8.32.1 Detailed Description

Internal data structure for `tensor_filter` instances.

Definition at line 61 of file `tensor_filter.h`.

8.32.2 Member Data Documentation

8.32.2.1 `element`

`GstBaseTransform` `_GstTensorFilter::element`

This is the parent object

Definition at line 63 of file `tensor_filter.h`.

8.32.2.2 `prev_ts`

`GstClockTime` `_GstTensorFilter::prev_ts`

previous timestamp

Definition at line 67 of file `tensor_filter.h`.

8.32.2.3 `priv`

`GstTensorFilterPrivate` `_GstTensorFilter::priv`

Internal properties for `tensor-filter`

Definition at line 65 of file `tensor_filter.h`.

8.32.2.4 throttling_accum

```
GstClockTimeDiff _GstTensorFilter::throttling_accum
```

accumulated frame durations for throttling

Definition at line 69 of file tensor_filter.h.

8.32.2.5 throttling_delay

```
GstClockTimeDiff _GstTensorFilter::throttling_delay
```

throttling delay from tensor rate

Definition at line 68 of file tensor_filter.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_filter/tensor_filter.h](#)

8.33 _GstTensorFilterClass Struct Reference

GstTensorFilterClass inherits GstBaseTransformClass.

```
#include <tensor_filter.h>
```

Public Attributes

- GstBaseTransformClass [parent_class](#)

8.33.1 Detailed Description

GstTensorFilterClass inherits GstBaseTransformClass.

Referring another child (sibling), GstVideoFilter (abstract class) and its child (concrete class) GstVideoConverter. Note that GstTensorFilterClass is a concrete class; thus we need to look at both.

Definition at line 79 of file tensor_filter.h.

8.33.2 Member Data Documentation

8.33.2.1 parent_class

```
GstBaseTransformClass _GstTensorFilterClass::parent_class
```

Inherits GstBaseTransformClass

Definition at line 81 of file tensor_filter.h.

The documentation for this struct was generated from the following file:

- nntstreamer/tensor_filter/[tensor_filter.h](#)

8.34 _GstTensorFilterCombination Struct Reference

Structure definition for tensor-filter in/out combination.

```
#include <tensor_filter_common.h>
```

Public Attributes

- GList * [in_combi](#)
- GList * [out_combi_i](#)
- GList * [out_combi_o](#)
- gboolean [in_combi_defined](#)
- gboolean [out_combi_i_defined](#)
- gboolean [out_combi_o_defined](#)

8.34.1 Detailed Description

Structure definition for tensor-filter in/out combination.

Definition at line 131 of file tensor_filter_common.h.

8.34.2 Member Data Documentation

8.34.2.1 in_combi

```
GList* _GstTensorFilterCombination::in_combi
```

Select the input tensor(s) to invoke the models

Definition at line 133 of file tensor_filter_common.h.

8.34.2.2 in_combi_defined

`gboolean _GstTensorFilterCombination::in_combi_defined`

True if input combination is defined

Definition at line 136 of file `tensor_filter_common.h`.

8.34.2.3 out_combi_i

`GList* _GstTensorFilterCombination::out_combi_i`

Select the output tensor(s) from the input tensor(s)

Definition at line 134 of file `tensor_filter_common.h`.

8.34.2.4 out_combi_i_defined

`gboolean _GstTensorFilterCombination::out_combi_i_defined`

True if output combination from input is defined

Definition at line 137 of file `tensor_filter_common.h`.

8.34.2.5 out_combi_o

`GList* _GstTensorFilterCombination::out_combi_o`

Select the output tensor(s) from the model output

Definition at line 135 of file `tensor_filter_common.h`.

8.34.2.6 out_combi_o_defined

`gboolean _GstTensorFilterCombination::out_combi_o_defined`

True if output combination from model output is defined

Definition at line 138 of file `tensor_filter_common.h`.

The documentation for this struct was generated from the following file:

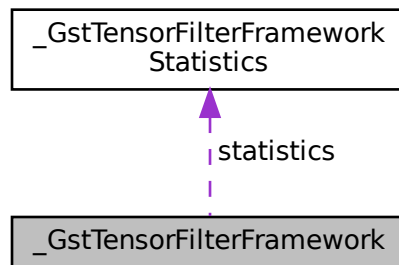
- `nntstreamer/tensor_filter/tensor_filter_common.h`

8.35 _GstTensorFilterFramework Struct Reference

Tensor_Filter Subplugin definition.

```
#include <nnstreamer_plugin_api_filter.h>
```

Collaboration diagram for _GstTensorFilterFramework:



Public Attributes

- `uint64_t` `version`
- `int`(* `open`)(const `GstTensorFilterProperties` *`prop`, void **`private_data`)
- `void`(* `close`)(const `GstTensorFilterProperties` *`prop`, void **`private_data`)
- union {
 - struct {
 - `char` * `name`
 - `int` `allow_in_place`
 - `int` `allocate_in_invoke`
 - `int` `run_without_model`
 - `int` `verify_model_path`
 - const `GstTensorFilterFrameworkStatistics` * `statistics`
 - `int`(* `invoke_NN`)(const `GstTensorFilterProperties` *`prop`, void **`private_data`, const `GstTensorMemory` *`input`, `GstTensorMemory` *`output`)
 - `int`(* `getInputDimension`)(const `GstTensorFilterProperties` *`prop`, void **`private_data`, `GstTensorsInfo` *`info`)
 - `int`(* `getOutputDimension`)(const `GstTensorFilterProperties` *`prop`, void **`private_data`, `GstTensorsInfo` *`info`)
 - `int`(* `setInputDimension`)(const `GstTensorFilterProperties` *`prop`, void **`private_data`, const `GstTensorsInfo` *`in_info`, `GstTensorsInfo` *`out_info`)

```

void(* destroyNotify )(void **private_data, void *data)
int(* reloadModel )(const
    GstTensorFilterProperties
    *prop, void **private_data)
int(* handleEvent )(event_ops ops,
    GstTensorFilterFrameworkEventData
    *data)
int(* checkAvailability )(accl_hw hw)
int(* allocateInInvoke )(void **private_data)
}
    Tensor_Filter Subplugin definition Version 0.
struct {
int(* invoke )(const
    GstTensorFilterFramework *self,
    GstTensorFilterProperties
    *prop, void *private_data,
    const GstTensorMemory *input,
    GstTensorMemory *output)
int(* getFrameworkInfo )(const
    GstTensorFilterFramework *self,
    const
    GstTensorFilterProperties
    *prop, void *private_data,
    GstTensorFilterFrameworkInfo
    *fw_info)
int(* getModelInfo )(const
    GstTensorFilterFramework *self,
    const
    GstTensorFilterProperties
    *prop, void *private_data,
    model_info_ops ops,
    GstTensorsInfo *in_info,
    GstTensorsInfo *out_info)
int(* eventHandler )(const
    GstTensorFilterFramework *self,
    const
    GstTensorFilterProperties
    *prop, void *private_data,
    event_ops ops,
    GstTensorFilterFrameworkEventData
    *data)
void * subplugin_data
}
    Tensor_Filter Subplugin definition Version 1.
};

```

Distinct elements between two versions of the subplugin interfaces.

8.35.1 Detailed Description

Tensor_Filter Subplugin definition.

Common callback parameters: prop Filter properties. Read Only. private_data Subplugin's private data. Set this (*private_data = XXX) if you want to change filter->private_data.

Definition at line 249 of file nnstreamer_plugin_api_filter.h.

8.35.2 Member Data Documentation

8.35.2.1 "@53

```
union { ... }
```

Distinct elements between two versions of the subplugin interfaces.

8.35.2.2 allocate_in_invoke

```
int _GstTensorFilterFramework::allocate_in_invoke
```

TRUE(nonzero) if invoke_NN is going to allocate output ptr by itself and return the address via output ptr. Do not change this value after cap negotiation is complete (or the stream has been started).

Definition at line 286 of file nnstreamer_plugin_api_filter.h.

8.35.2.3 allocateInInvoke

```
int (* _GstTensorFilterFramework::allocateInInvoke) (void **private_data)
```

Optional. Tensor-filter will call it when allocate_in_invoke is set to TRUE. This check if the provided model for the framework supports allocation at invoke or not. If this is not defined, then the value of allocate_in_invoke is assumed to be final for all models.

Parameters

in	<i>private_data</i>	A subplugin may save its internal private data here.
----	---------------------	--

Returns

0 if supported. -errno if not supported.

Definition at line 378 of file nnstreamer_plugin_api_filter.h.

8.35.2.4 allow_in_place

```
int _GstTensorFilterFramework::allow_in_place
```

TRUE(nonzero) if in-place transfer of input-to-output is allowed. Not supported in main, yet

Definition at line 285 of file nnstreamer_plugin_api_filter.h.

8.35.2.5 checkAvailability

```
int (* _GstTensorFilterFramework::checkAvailability) (accl_hw hw)
```

Optional. Check if the provided hardware accelerator is supported. This check is static or dynamic based on framework support. Positive response of this check does not guarantee successful running of model with this accelerator. The static check can be performed without opening the framework.

Parameters

in	<i>hw</i>	backend accelerator hardware
----	-----------	------------------------------

Returns

0 if supported. -errno if not supported.

Definition at line 371 of file `nstreamer_plugin_api_filter.h`.

8.35.2.6 close

```
void (* _GstTensorFilterFramework::close) (const GstTensorFilterProperties *prop, void **private_data)
```

Optional. Tensor-filter will not call other callbacks after calling close. Free-ing `private_data` is this function's responsibility. Set NULL after that.

Parameters

in	<i>prop</i>	read-only property values.
	<i>[in/out]</i>	<code>private_data</code> A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer. Normally, <code>close()</code> frees <code>private_data</code> and set NULL.

Definition at line 267 of file `nstreamer_plugin_api_filter.h`.

8.35.2.7 destroyNotify

```
void (* _GstTensorFilterFramework::destroyNotify) (void **private_data, void *data)
```

Optional. Tensor-filter will call it when 'allocate_in_invoke' flag of the framework is TRUE and `allocateInInvoke` also return enabled. Basically, it is called when the data element is destroyed. If it's set as NULL, `g_free()` will be used as a default. It will be helpful when the data pointer is included as an object of a `nnfw`. For instance, if the data pointer is removed when the object is gone, it occurs error. In this case, the objects should be maintained for a while first and destroyed when the data pointer is destroyed. Those kinds of logic could be defined at this method.

Parameters

	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
in	<i>data</i>	the data element.

Definition at line 346 of file nntstreamer_plugin_api_filter.h.

8.35.2.8 eventHandler

```
int(* _GstTensorFilterFramework::eventHandler) (const GstTensorFilterFramework *self, const
GstTensorFilterProperties *prop, void *private_data, event_ops ops, GstTensorFilterFrameworkEventData
*data)
```

Mandatory callback. Runs the event corresponding to the passed operation. If ops == DESTROY_NOTIFY: Tensor-filter will call it when 'allocate_in_invoke' property of the framework is TRUE. Basically, it is called when the data element is destroyed. If it's set as NULL, [g_free\(\)](#) will be used as a default. It will be helpful when the data pointer is included as an object of a nntfw. For instance, if the data pointer is removed when the object is gone, it occurs error. In this case, the objects should be maintained for a while first and destroyed when the data pointer is destroyed. Those kinds of logic could be defined at this method. If ops == RELOAD_MODEL: Tensor-filter will call it when a model property is newly configured. Also, 'is-updatable' property of the framework should be TRUE. This function reloads a new model passed in as argument via data. Note that it requires extra memory size enough to temporarily hold both old and new models during this function to hide the reload overhead. If ops == CUSTOM_PROP: Tensor-filter will call to update the custom properties of the subplugin. If ops == SET_INPUT_PROP: Tensor-filter will call to update the property of the subplugin. This function will take tensor info and layout as the argument. This operation can update input tensor shape, type, name and layout. If ops == SET_OUTPUT_PROP: Tensor-filter will call to update the property of the subplugin. This function will take tensor info and layout as the argument. This operation can update output tensor shape, type, name and layout. If ops == SET_ACCELERATOR: Tensor-filter will call to update the property of the subplugin. This function will take accelerator list as the argument. This operation will update the backend to be used by the corresponding subplugin. List of operations to be supported are optional. Note: In these operations, the argument 'prop' will not contain the updated information, but will be updated after the corresponding operation is succeeded.

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer. (can be NULL)
in	<i>ops</i>	operation to be performed
	<i>[in/out]</i>	data event data for the supported handlers (can be NULL)

Returns

0 if OK. non-zero if error. -ENOENT if operation is not supported. -EINVAL if operation is supported but provided arguments are invalid.

Definition at line 445 of file nntstreamer_plugin_api_filter.h.

8.35.2.9 getFrameworkInfo

```
int(* _GstTensorFilterFramework::getFrameworkInfo) (const GstTensorFilterFramework *self, const
GstTensorFilterProperties *prop, void *private_data, GstTensorFilterFrameworkInfo *fw_info)
```

Mandatory callback. Get the frameworks statically determined info. Argument 'private_data' can be NULL. If provided 'private_data' is not NULL, then some info, such as 'allocate_in_invoke', can be updated based on the model being used (inferred from the 'private_data' provided). This updated info is useful for custom filter, as some custom filter's ability to support 'allocate_in_invoke' depends on the opened model.

Parameters

in	<i>prop</i>	read-only property values
in	<i>private_data</i>	A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer. This parameter can be NULL.
out	<i>fw_info</i>	struct to hold frameworks info. Must be allocated by the caller (return value).

Returns

0 if OK. non-zero if error.

Note

CAUTION: private_data can be NULL if the framework is not yet opened by the caller.

Definition at line 407 of file nnstreamer_plugin_api_filter.h.

8.35.2.10 getInputDimension

```
int(* _GstTensorFilterFramework::getInputDimension) (const GstTensorFilterProperties *prop,
void **private_data, GstTensorsInfo *info)
```

Optional. Set NULL if not supported. Get dimension of input tensor If getInputDimension is NULL, setInputDimension must be defined. If getInputDimension is defined, it is recommended to define getOutputDimension.

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
out	<i>info</i>	structure of tensor info (return value)

Returns

0 if OK. non-zero if error.

Definition at line 303 of file nnstreamer_plugin_api_filter.h.

8.35.2.11 getModelInfo

```
int(* _GstTensorFilterFramework::getModelInfo) (const GstTensorFilterFramework *self, const
GstTensorFilterProperties *prop, void *private_data, model_info_ops ops, GstTensorsInfo *in←
_ininfo, GstTensorsInfo *out_info)
```

Mandatory callback. Gets the model related tensor info. If ops == GET_IN_OUT_INFO, in_info would contain the input tensor info, and out_info would contain the output tensor info. If ops == SET_INPUT_INFO, in_info would contain the provided input tensor info, and out_info would contain the updated output tensor info. At least one of SET_INPUT_INFO and GET_IN_OUT_INFO operations must be supported.

Note: Tensor_Filter::main will configure input dimension from pad-cap in run-time for the sub-plugin. Then, the sub-plugin is required to return corresponding output dimension, and will call SET_INPUT_INFO operation.

Note: With SET_INPUT_INFO operation, the caller must NOT allocate or fix internal data structure based on the return value until invoke is called. GStreamer may try different dimensions before settling down.

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
in	<i>ops</i>	operation to be performed
	<i>[in/out]</i>	in_info structure of input tensor info
out	<i>out_info</i>	structure of output tensor info (return value)

Returns

0 if OK. non-zero if error. -ENOENT is operation is not supported.

Definition at line 420 of file nnstreamer_plugin_api_filter.h.

8.35.2.12 getOutputDimension

```
int(* _GstTensorFilterFramework::getOutputDimension) (const GstTensorFilterProperties *prop,
void **private_data, GstTensorsInfo *info)
```

Optional. Set NULL if not supported. Get dimension of output tensor If getInputDimension is NULL, setInputDimension must be defined. If getInputDimension is defined, it is recommended to define getOutputDimension.

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
out	<i>info</i>	structure of tensor info (return value)

Returns

0 if OK. non-zero if error.

Definition at line 315 of file nnstreamer_plugin_api_filter.h.

8.35.2.13 handleEvent

```
int (* _GstTensorFilterFramework::handleEvent) (event_ops ops, GstTensorFilterFrameworkEventData
*data)
```

Optional. Runs the event corresponding to the passed operation. If ops == CHECK_HW_AVAILABILITY: tensor↔_filter will call to check the hw availability with custom option. List of operations to be supported are optional.

Parameters

in	ops	operation to be performed
	[in/out]	data event data for the supported handlers (can be NULL)

Returns

0 if OK. non-zero if error. -ENOENT if operation is not supported. -EINVAL if operation is supported but provided arguments are invalid.

Definition at line 361 of file nnstreamer_plugin_api_filter.h.

8.35.2.14 invoke

```
int (* _GstTensorFilterFramework::invoke) (const GstTensorFilterFramework *self, GstTensorFilterProperties
*prop, void *private_data, const GstTensorMemory *input, GstTensorMemory *output)
```

Mandatory callback. Invoke the given network model.

Parameters

	[in/out]	prop property values. In the case of dynamic invoke, the output tensors info must be filled.
	[in/out]	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
in	input	The array of input tensors. Allocated and filled by tensor_filter/main
out	output	The array of output tensors. Allocated by tensor_filter/main and to be filled by invoke. If allocate_in_invoke is TRUE, sub-plugin should allocate the memory block for output tensor. (data in GstTensorMemory)

Returns

0 if OK. non-zero if error.

Definition at line 395 of file nnstreamer_plugin_api_filter.h.

8.35.2.15 invoke_NN

```
int (* _GstTensorFilterFramework::invoke_NN) (const GstTensorFilterProperties *prop, void **private_data, const GstTensorMemory *input, GstTensorMemory *output)
```

Mandatory callback. Invoke the given network model.

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
in	<i>input</i>	The array of input tensors. Allocated and filled by tensor_filter/main
out	<i>output</i>	The array of output tensors. Allocated by tensor_filter/main and to be filled by invoke_NN. If allocate_in_invoke is TRUE, sub-plugin should allocate the memory block for output tensor. (data in GstTensorMemory)

Returns

0 if OK. non-zero if error.

Definition at line 292 of file nnstreamer_plugin_api_filter.h.

8.35.2.16 name

```
char* _GstTensorFilterFramework::name
```

Name of the neural network framework, searchable by FRAMEWORK property

Definition at line 284 of file nnstreamer_plugin_api_filter.h.

8.35.2.17 open

```
int (* _GstTensorFilterFramework::open) (const GstTensorFilterProperties *prop, void **private_data)
```

Optional. Tensor-filter will call this before any of other callbacks and will call once before calling close.

Note: If 'open' callback is not defined, then the private_data passed in other callbacks will be NULL.

Parameters

<i>in</i>	<i>prop</i>	read-only property values.
	<i>[in/out]</i>	<i>private_data</i> A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer. Normally, open() allocates memory for <i>private_data</i> .

Returns

0 if ok. < 0 if error.

Definition at line 257 of file `nntstreamer_plugin_api_filter.h`.

8.35.2.18 reloadModel

```
int (* _GstTensorFilterFramework::reloadModel) (const GstTensorFilterProperties *prop, void
**private_data)
```

Optional. Tensor-filter will call it when a model property is newly configured. Also, 'is-updatable' property of the framework should be TRUE. This function reloads a new model specified in the 'prop' argument. Note that it requires extra memory size enough to temporarily hold both old and new models during this function to hide the reload overhead.

Parameters

<i>in</i>	<i>prop</i>	read-only property values
	<i>[in/out]</i>	<i>private_data</i> A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer. Normally, close() frees <i>private_data</i> and set NULL.

Returns

0 if ok. < 0 if error.

Definition at line 353 of file `nntstreamer_plugin_api_filter.h`.

8.35.2.19 run_without_model

```
int _GstTensorFilterFramework::run_without_model
```

TRUE(nonzero) when the neural network framework does not need a model file. Tensor-filter will run `invoke_NN` without model.

Definition at line 287 of file `nntstreamer_plugin_api_filter.h`.

8.35.2.20 setInputDimension

```
int (* _GstTensorFilterFramework::setInputDimension) (const GstTensorFilterProperties *prop,
void **private_data, const GstTensorsInfo *in_info, GstTensorsInfo *out_info)
```

Optional. Set Null if not supported. Tensor_Filter::main will configure input dimension from pad-cap in run-time for the sub-plugin. Then, the sub-plugin is required to return corresponding output dimension. If this is NULL, both getInput/OutputDimension must be non-NULL.

When you use this, do NOT allocate or fix internal data structure based on it until invoke is called. GStreamer may try different dimensions before settling down.

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
in	<i>in_info</i>	structure of input tensor info
out	<i>out_info</i>	structure of output tensor info (return value)

Returns

0 if OK. non-zero if error.

Definition at line 327 of file nnstreamer_plugin_api_filter.h.

8.35.2.21 statistics

```
const GstTensorFilterFrameworkStatistics* _GstTensorFilterFramework::statistics
```

usage statistics by the framework. This is shared across all opened instances of this framework.

Definition at line 290 of file nnstreamer_plugin_api_filter.h.

8.35.2.22 subplugin_data

```
void* _GstTensorFilterFramework::subplugin_data
```

This is used by tensor_filter infrastructure. Subplugin authors should NEVER update this. Only the files in /gst/nnstreamer/tensor_filter/ are allowed to access this.

Definition at line 464 of file nnstreamer_plugin_api_filter.h.

8.35.2.23 verify_model_path

```
int _GstTensorFilterFramework::verify_model_path
```

TRUE(nonzero) when the NNS framework, not the sub-plugin, should verify the path of model files.

Definition at line 288 of file nnstreamer_plugin_api_filter.h.

8.35.2.24 version

```
uint64_t _GstTensorFilterFramework::version
```

Version of the struct | 32bit (validity check) | 16bit (API version) | 16bit (Subplugin's internal version. Tensor-filter does not care.) | API version will be 0x0 (earlier version (_GstTensorFilterFramework_v0)) or 0x1 (newer version (_GstTensorFilterFramework_v1))

Definition at line 251 of file nnstreamer_plugin_api_filter.h.

The documentation for this struct was generated from the following file:

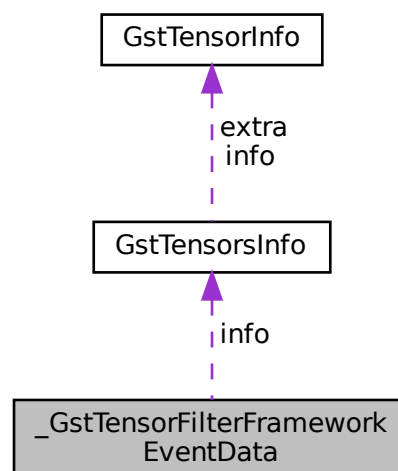
- [nnstreamer/include/nnstreamer_plugin_api_filter.h](#)

8.36 _GstTensorFilterFrameworkEventData Struct Reference

User data for the tensor_tilter subplugin related events.

```
#include <nnstreamer_plugin_api_filter.h>
```

Collaboration diagram for _GstTensorFilterFrameworkEventData:



Public Attributes

```

• union {
    struct {
        void * data
    }
    struct {
        const char ** model_files
        int num_models
    }
    struct {
        const char * custom_properties
    }
    struct {
        const GstTensorsInfo * info
        tensors_layout layout
    }
    struct {
        accl_hw * hw_list
        int num_hw
    }
    struct {
        accl_hw hw
        const char * custom
    }
};

```

8.36.1 Detailed Description

User data for the tensor_tilter subplugin related events.

Definition at line 200 of file nnstreamer_plugin_api_filter.h.

8.36.2 Member Data Documentation

8.36.2.1 "@39

```
union { ... }
```

Union of the user data for each event supported by eventHandler

8.36.2.2 custom

```
const char* _GstTensorFilterFrameworkEventData::custom
```

custom option for hardware detection

Definition at line 235 of file nnstreamer_plugin_api_filter.h.

8.36.2.3 custom_properties

```
const char* _GstTensorFilterFrameworkEventData::custom_properties
```

sub-plugin specific custom property values in string

Definition at line 217 of file nnstreamer_plugin_api_filter.h.

8.36.2.4 data

```
void* _GstTensorFilterFrameworkEventData::data
```

The data element to be destroyed

Definition at line 206 of file nnstreamer_plugin_api_filter.h.

8.36.2.5 hw

```
accl_hw _GstTensorFilterFrameworkEventData::hw
```

accelerator to check availability

Definition at line 234 of file nnstreamer_plugin_api_filter.h.

8.36.2.6 hw_list

```
accl_hw* _GstTensorFilterFrameworkEventData::hw_list
```

accelerators supported by framework intersected with the new user provided accelerator preference

Definition at line 228 of file nnstreamer_plugin_api_filter.h.

8.36.2.7 info

```
const GstTensorsInfo* _GstTensorFilterFrameworkEventData::info
```

The tensor info to be updated to

Definition at line 222 of file nnstreamer_plugin_api_filter.h.

8.36.2.8 `layout`

```
tensors_layout _GstTensorFilterFrameworkEventData::layout
```

The layout of the tensor to be updated to

Definition at line 223 of file `nnstreamer_plugin_api_filter.h`.

8.36.2.9 `model_files`

```
const char** _GstTensorFilterFrameworkEventData::model_files
```

File path to the new list of model files

Definition at line 211 of file `nnstreamer_plugin_api_filter.h`.

8.36.2.10 `num_hw`

```
int _GstTensorFilterFrameworkEventData::num_hw
```

number of hardware accelerators in the `hw_list` supported by the framework

Definition at line 229 of file `nnstreamer_plugin_api_filter.h`.

8.36.2.11 `num_models`

```
int _GstTensorFilterFrameworkEventData::num_models
```

Updated number of the model files

Definition at line 212 of file `nnstreamer_plugin_api_filter.h`.

The documentation for this struct was generated from the following file:

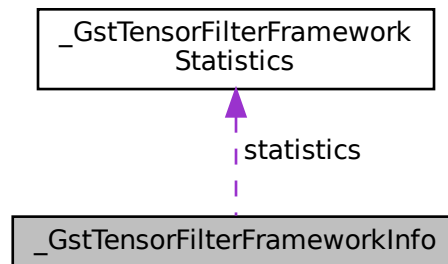
- `nnstreamer/include/nnstreamer_plugin_api_filter.h`

8.37 `_GstTensorFilterFrameworkInfo` Struct Reference

Tensor_Filter Subplugin framework related information.

```
#include <nnstreamer_plugin_api_filter.h>
```

Collaboration diagram for `_GstTensorFilterFrameworkInfo`:



Public Attributes

- `const char * name`
- `int allow_in_place`
- `int allocate_in_invoke`
- `int run_without_model`
- `int verify_model_path`
- `const accl_hw * hw_list`
- `int num_hw`
- `accl_hw accl_auto`
- `accl_hw accl_default`
- `const GstTensorFilterFrameworkStatistics * statistics`

8.37.1 Detailed Description

Tensor_Filter Subplugin framework related information.

All the information except the supported accelerator is provided statically. Accelerators can be provided based on static or dynamic check dependent on framework support.

Definition at line 158 of file `nnstreamer_plugin_api_filter.h`.

8.37.2 Member Data Documentation

8.37.2.1 `accl_auto`

```
accl_hw _GstTensorFilterFrameworkInfo::accl_auto
```

accelerator to be used in auto mode (acceleration to be used but accelerator is not specified for the filter) - default -1 implies use first entry from `hw_list`.

Definition at line 167 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.2 `accl_default`

```
accl_hw _GstTensorFilterFrameworkInfo::accl_default
```

accelerator to be used by default (valid user input is not provided) - default -1 implies use first entry from `hw_list`.

Definition at line 168 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.3 `allocate_in_invoke`

```
int _GstTensorFilterFrameworkInfo::allocate_in_invoke
```

TRUE(nonzero) if `invoke_NN` is going to allocate output ptr by itself and return the address via output ptr. Do not change this value after cap negotiation is complete (or the stream has been started).

Definition at line 162 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.4 `allow_in_place`

```
int _GstTensorFilterFrameworkInfo::allow_in_place
```

TRUE(nonzero) if in-place transfer of input-to-output is allowed. Not supported in main, yet.

Definition at line 161 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.5 `hw_list`

```
const accl_hw* _GstTensorFilterFrameworkInfo::hw_list
```

List of supported hardware accelerators by the framework. Positive response of this check does not guarantee successful running of model with this accelerator. Subplugin is supposed to allocate/deallocate.

Definition at line 165 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.6 name

```
const char* _GstTensorFilterFrameworkInfo::name
```

Name of the neural network framework, searchable by FRAMEWORK property. Subplugin is supposed to allocate/deallocate.

Definition at line 160 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.7 num_hw

```
int _GstTensorFilterFrameworkInfo::num_hw
```

number of hardware accelerators in the `hw_list` supported by the framework.

Definition at line 166 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.8 run_without_model

```
int _GstTensorFilterFrameworkInfo::run_without_model
```

TRUE(nonzero) when the neural network framework does not need a model file. Tensor-filter will run `invoke_NN` without model.

Definition at line 163 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.9 statistics

```
const GstTensorFilterFrameworkStatistics* _GstTensorFilterFrameworkInfo::statistics
```

usage statistics by the framework. This is shared across all opened instances of this framework.

Definition at line 169 of file `nnstreamer_plugin_api_filter.h`.

8.37.2.10 verify_model_path

```
int _GstTensorFilterFrameworkInfo::verify_model_path
```

TRUE(nonzero) when the NNS framework, not the sub-plugin, should verify the path of model files.

Definition at line 164 of file `nnstreamer_plugin_api_filter.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/nnstreamer_plugin_api_filter.h`

8.38 `_GstTensorFilterFrameworkStatistics` Struct Reference

Structure definition for tensor-filter framework statistics.

```
#include <nnstreamer_plugin_api_filter.h>
```

Public Attributes

- `int64_t total_invoke_num`
- `int64_t total_invoke_latency`
- `int64_t total_overhead_latency`

8.38.1 Detailed Description

Structure definition for tensor-filter framework statistics.

Definition at line 145 of file `nnstreamer_plugin_api_filter.h`.

8.38.2 Member Data Documentation

8.38.2.1 `total_invoke_latency`

```
int64_t _GstTensorFilterFrameworkStatistics::total_invoke_latency
```

The total accumulated invoke latency (usec)

Definition at line 148 of file `nnstreamer_plugin_api_filter.h`.

8.38.2.2 `total_invoke_num`

```
int64_t _GstTensorFilterFrameworkStatistics::total_invoke_num
```

The total number of calls to invoke

Definition at line 147 of file `nnstreamer_plugin_api_filter.h`.

8.38.2.3 total_overhead_latency

```
int64_t _GstTensorFilterFrameworkStatistics::total_overhead_latency
```

The total accumulated overhead latency from the extension (usec)

Definition at line 149 of file `nstreamer_plugin_api_filter.h`.

The documentation for this struct was generated from the following file:

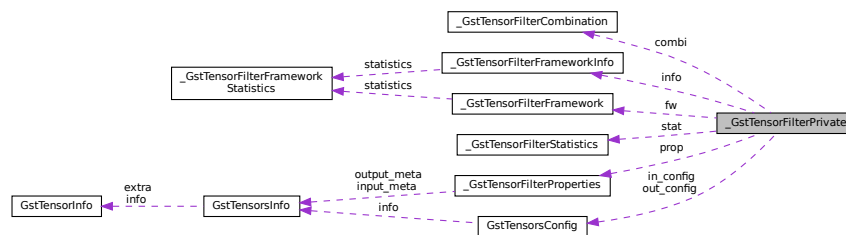
- `nstreamer/include/nstreamer_plugin_api_filter.h`

8.39 _GstTensorFilterPrivate Struct Reference

Structure definition for common tensor-filter properties.

```
#include <tensor_filter_common.h>
```

Collaboration diagram for `_GstTensorFilterPrivate`:



Public Attributes

- void * `privateData`
- `GstTensorFilterProperties` `prop`
- `GstTensorFilterFrameworkInfo` `info`
- `GstTensorFilterStatistics` `stat`
- const `GstTensorFilterFramework` * `fw`
- gboolean `silent`
- gboolean `configured`
- gboolean `is_updatable`
- gchar * `config_path`
- `GstTensorsConfig` `in_config`
- `GstTensorsConfig` `out_config`
- gint `latency_mode`
- gint `throughput_mode`
- gboolean `latency_reporting`
- gint64 `latency_reported`
- `GstTensorFilterCombination` `combi`
- `nns_watchdog_h` `watchdog_h`

8.39.1 Detailed Description

Structure definition for common tensor-filter properties.

Definition at line 152 of file `tensor_filter_common.h`.

8.39.2 Member Data Documentation

8.39.2.1 `combi`

`GstTensorFilterCombination` `_GstTensorFilterPrivate::combi`

Definition at line 173 of file `tensor_filter_common.h`.

8.39.2.2 `config_path`

`gchar*` `_GstTensorFilterPrivate::config_path`

Path to configuration file

Definition at line 164 of file `tensor_filter_common.h`.

8.39.2.3 `configured`

`gboolean` `_GstTensorFilterPrivate::configured`

True if already successfully configured tensor metadata

Definition at line 162 of file `tensor_filter_common.h`.

8.39.2.4 `fw`

`const` `GstTensorFilterFramework*` `_GstTensorFilterPrivate::fw`

The implementation core of the NNF. NULL if not configured

Definition at line 158 of file `tensor_filter_common.h`.

8.39.2.5 in_config

`GstTensorsConfig _GstTensorFilterPrivate::in_config`

input tensor info

Definition at line 165 of file `tensor_filter_common.h`.

8.39.2.6 info

`GstTensorFilterFrameworkInfo _GstTensorFilterPrivate::info`

NNFW framework info

Definition at line 156 of file `tensor_filter_common.h`.

8.39.2.7 is_updatable

`gboolean _GstTensorFilterPrivate::is_updatable`

a given model to the filter is updatable if TRUE

Definition at line 163 of file `tensor_filter_common.h`.

8.39.2.8 latency_mode

`gint _GstTensorFilterPrivate::latency_mode`

latency profiling mode (0: off, 1: on, ...)

Definition at line 168 of file `tensor_filter_common.h`.

8.39.2.9 latency_reported

`gint64 _GstTensorFilterPrivate::latency_reported`

latency value reported (ns) in last LATENCY query

Definition at line 171 of file `tensor_filter_common.h`.

8.39.2.10 `latency_reporting`

```
gboolean _GstTensorFilterPrivate::latency_reporting
```

reporting of estimated filter latency is enabled

Definition at line 170 of file `tensor_filter_common.h`.

8.39.2.11 `out_config`

```
GstTensorsConfig _GstTensorFilterPrivate::out_config
```

output tensor info

Definition at line 166 of file `tensor_filter_common.h`.

8.39.2.12 `privateData`

```
void* _GstTensorFilterPrivate::privateData
```

NNFW plugin's private data is stored here

Definition at line 154 of file `tensor_filter_common.h`.

8.39.2.13 `prop`

```
GstTensorFilterProperties _GstTensorFilterPrivate::prop
```

NNFW plugin's properties

Definition at line 155 of file `tensor_filter_common.h`.

8.39.2.14 `silent`

```
gboolean _GstTensorFilterPrivate::silent
```

Verbose mode if FALSE. int instead of gboolean for non-glib custom plugins

Definition at line 161 of file `tensor_filter_common.h`.

8.39.2.15 stat

`GstTensorFilterStatistics` `_GstTensorFilterPrivate::stat`

NNFW plugin's statistics

Definition at line 157 of file `tensor_filter_common.h`.

8.39.2.16 throughput_mode

`gint` `_GstTensorFilterPrivate::throughput_mode`

throughput profiling mode (0: off, 1: on, ...)

Definition at line 169 of file `tensor_filter_common.h`.

8.39.2.17 watchdog_h

`nns_watchdog_h` `_GstTensorFilterPrivate::watchdog_h`

Watchdog to monitor occasional inputs

Definition at line 174 of file `tensor_filter_common.h`.

The documentation for this struct was generated from the following file:

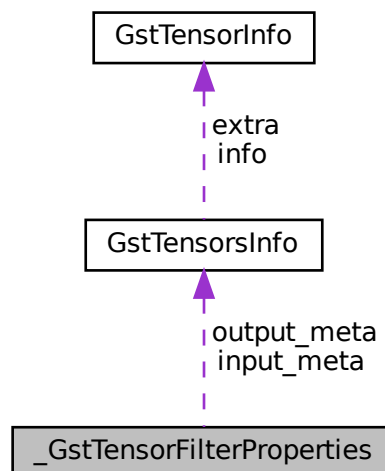
- `nnstreamer/tensor_filter/tensor_filter_common.h`

8.40 `_GstTensorFilterProperties` Struct Reference

`GstTensorFilter`'s properties for NN framework (internal data structure)

```
#include <nnstreamer_plugin_api_filter.h>
```

Collaboration diagram for `_GstTensorFilterProperties`:



Public Attributes

- const char * [fwname](#)
- int [fw_opened](#)
- const char ** [model_files](#)
- int [num_models](#)
- int [input_configured](#)
- [GstTensorsInfo](#) [input_meta](#)
- [tensors_layout](#) [input_layout](#)
- unsigned int [input_ranks](#) [[NNS_TENSOR_SIZE_LIMIT](#)]
- int [output_configured](#)
- [GstTensorsInfo](#) [output_meta](#)
- [tensors_layout](#) [output_layout](#)
- unsigned int [output_ranks](#) [[NNS_TENSOR_SIZE_LIMIT](#)]
- const char * [custom_properties](#)
- [accl_hw](#) * [hw_list](#)
- int [num_hw](#)
- const char * [accl_str](#)
- char * [shared_tensor_filter_key](#)
- int [latency](#)
- int [throughput](#)
- int [invoke_dynamic](#)
- uint32_t [suspend](#)

8.40.1 Detailed Description

GstTensorFilter's properties for NN framework (internal data structure)

Because custom filters of `tensor_filter` may need to access internal data of `GstTensorFilter`, we define this data structure here.

Definition at line 112 of file `nnstreamer_plugin_api_filter.h`.

8.40.2 Member Data Documentation

8.40.2.1 `accl_str`

```
const char* _GstTensorFilterProperties::accl_str
```

accelerator configuration passed in as parameter, use in `GstTensorFilterFramework V0` only

Definition at line 132 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.2 custom_properties

```
const char* _GstTensorFilterProperties::custom_properties
```

sub-plugin specific custom property values in string

Definition at line 129 of file nnstreamer_plugin_api_filter.h.

8.40.2.3 fw_opened

```
int _GstTensorFilterProperties::fw_opened
```

TRUE IF open() is called or tried. Use int instead of gboolean because this is referred by custom plugins.

Definition at line 115 of file nnstreamer_plugin_api_filter.h.

8.40.2.4 fwname

```
const char* _GstTensorFilterProperties::fwname
```

The name of NN Framework

Definition at line 114 of file nnstreamer_plugin_api_filter.h.

8.40.2.5 hw_list

```
accl_hw* _GstTensorFilterProperties::hw_list
```

accelerators supported by framework intersected with user provided accelerator preference, use in GstTensor↵
FilterFramework V1 only

Definition at line 130 of file nnstreamer_plugin_api_filter.h.

8.40.2.6 input_configured

```
int _GstTensorFilterProperties::input_configured
```

TRUE if input tensor is configured. Use int instead of gboolean because this is referred by custom plugins.

Definition at line 119 of file nnstreamer_plugin_api_filter.h.

8.40.2.7 `input_layout`

`tensors_layout` `_GstTensorFilterProperties::input_layout`

data layout info provided as a property to `tensor_filter` for the input, defaults to `_NNS_LAYOUT_ANY` for all the tensors

Definition at line 121 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.8 `input_meta`

`GstTensorsInfo` `_GstTensorFilterProperties::input_meta`

configured input tensor info

Definition at line 120 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.9 `input_ranks`

`unsigned int` `_GstTensorFilterProperties::input_ranks` [`NNS_TENSOR_SIZE_LIMIT`]

the rank list of input tensors, it is calculated based on the dimension string.

Definition at line 122 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.10 `invoke_dynamic`

`int` `_GstTensorFilterProperties::invoke_dynamic`

True for supporting invoke with flexible output.

Definition at line 137 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.11 `latency`

`int` `_GstTensorFilterProperties::latency`

The average latency over the recent 10 inferences in microseconds

Definition at line 135 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.12 model_files

```
const char** _GstTensorFilterProperties::model_files
```

File path to the model file (as an argument for NNFV). char instead of gchar for non-glib custom plugins

Definition at line 116 of file nnstreamer_plugin_api_filter.h.

8.40.2.13 num_hw

```
int _GstTensorFilterProperties::num_hw
```

number of hardware accelerators in the hw_list supported by the framework

Definition at line 131 of file nnstreamer_plugin_api_filter.h.

8.40.2.14 num_models

```
int _GstTensorFilterProperties::num_models
```

number of model files. Some frameworks need multiple model files to initialize the graph (caffe, caffe2)

Definition at line 117 of file nnstreamer_plugin_api_filter.h.

8.40.2.15 output_configured

```
int _GstTensorFilterProperties::output_configured
```

TRUE if output tensor is configured. Use int instead of gboolean because this is referred by custom plugins.

Definition at line 124 of file nnstreamer_plugin_api_filter.h.

8.40.2.16 output_layout

```
tensors_layout _GstTensorFilterProperties::output_layout
```

data layout info provided as a property to tensor_filter for the output, defaults to _NNS_LAYOUT_ANY for all the tensors

Definition at line 126 of file nnstreamer_plugin_api_filter.h.

8.40.2.17 `output_meta`

`GstTensorsInfo` `_GstTensorFilterProperties::output_meta`

configured output tensor info

Definition at line 125 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.18 `output_ranks`

`unsigned int` `_GstTensorFilterProperties::output_ranks` [`NNS_TENSOR_SIZE_LIMIT`]

the rank list of output tensors, it is calculated based on the dimension string.

Definition at line 127 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.19 `shared_tensor_filter_key`

`char*` `_GstTensorFilterProperties::shared_tensor_filter_key`

the shared instance key to use same model representation

Definition at line 133 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.20 `suspend`

`uint32_t` `_GstTensorFilterProperties::suspend`

Timeout (ms) for suspend. (Unload the framework)

Definition at line 139 of file `nnstreamer_plugin_api_filter.h`.

8.40.2.21 `throughput`

`int` `_GstTensorFilterProperties::throughput`

The average throughput in the number of outputs per second

Definition at line 136 of file `nnstreamer_plugin_api_filter.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/nnstreamer_plugin_api_filter.h`

8.41 `_GstTensorFilterStatistics` Struct Reference

Structure definition for tensor-filter statistics.

```
#include <tensor_filter_common.h>
```

Public Attributes

- gint64 [total_invoke_num](#)
- gint64 [total_invoke_latency](#)
- gint64 [old_total_invoke_num](#)
- gint64 [old_total_invoke_latency](#)
- gint64 [latest_invoke_time](#)
- void * [recent_latencies](#)
- guint [latency_ignore_count](#)

8.41.1 Detailed Description

Structure definition for tensor-filter statistics.

Definition at line 117 of file `tensor_filter_common.h`.

8.41.2 Member Data Documentation

8.41.2.1 `latency_ignore_count`

```
guint _GstTensorFilterStatistics::latency_ignore_count
```

Definition at line 125 of file `tensor_filter_common.h`.

8.41.2.2 `latest_invoke_time`

```
gint64 _GstTensorFilterStatistics::latest_invoke_time
```

the latest invoke time (usec)

Definition at line 123 of file `tensor_filter_common.h`.

8.41.2.3 **old_total_invoke_latency**

`gint64 _GstTensorFilterStatistics::old_total_invoke_latency`

cached value. accumulated invoke latency (usec)

Definition at line 122 of file `tensor_filter_common.h`.

8.41.2.4 **old_total_invoke_num**

`gint64 _GstTensorFilterStatistics::old_total_invoke_num`

cached value. number of total invokes

Definition at line 121 of file `tensor_filter_common.h`.

8.41.2.5 **recent_latencies**

`void* _GstTensorFilterStatistics::recent_latencies`

data structure (e.g., queue) to hold recent latencies

Definition at line 124 of file `tensor_filter_common.h`.

8.41.2.6 **total_invoke_latency**

`gint64 _GstTensorFilterStatistics::total_invoke_latency`

accumulated invoke latency (usec)

Definition at line 120 of file `tensor_filter_common.h`.

8.41.2.7 **total_invoke_num**

`gint64 _GstTensorFilterStatistics::total_invoke_num`

number of total invokes

Definition at line 119 of file `tensor_filter_common.h`.

The documentation for this struct was generated from the following file:

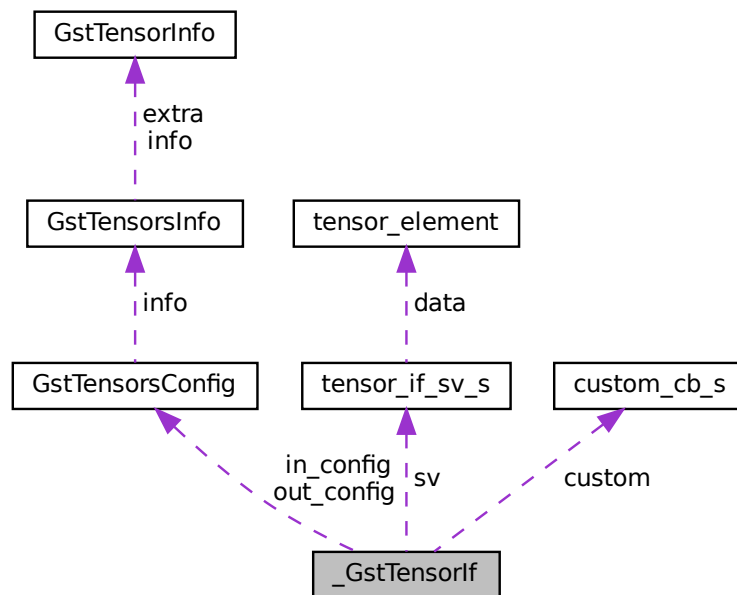
- `nntstreamer/tensor_filter/tensor_filter_common.h`

8.42 `_GstTensorIf` Struct Reference

Tensor If data structure.

```
#include <gsttensor_if.h>
```

Collaboration diagram for `_GstTensorIf`:



Public Attributes

- `GstElement` `element`
- `GstPad` * `sinkpad`
- `GSList` * `srcpads`
- `gboolean` `silent`
- `GstTensorsConfig` `in_config`
- `GstTensorsConfig` `out_config` [2]
- `guint32` `num_srcpads`
- `tensor_if_compared_value` `cv`
- `tensor_if_operator` `op`
- `tensor_if_behavior` `act_then`
- `tensor_if_behavior` `act_else`
- `tensor_if_sv_s` `sv` [2]
- `GList` * `cv_option`
- `GList` * `then_option`
- `GList` * `else_option`
- `gboolean` `custom_configured`
- `custom_cb_s` `custom`
- `GMutex` `lock`

8.42.1 Detailed Description

Tensor If data structure.

Definition at line 122 of file `gsttensor_if.h`.

8.42.2 Member Data Documentation

8.42.2.1 `act_else`

`tensor_if_behavior` `_GstTensorIf::act_else`

Definition at line 136 of file `gsttensor_if.h`.

8.42.2.2 `act_then`

`tensor_if_behavior` `_GstTensorIf::act_then`

Definition at line 135 of file `gsttensor_if.h`.

8.42.2.3 `custom`

`custom_cb_s` `_GstTensorIf::custom`

Definition at line 143 of file `gsttensor_if.h`.

8.42.2.4 `custom_configured`

`gboolean` `_GstTensorIf::custom_configured`

Definition at line 142 of file `gsttensor_if.h`.

8.42.2.5 `cv`

`tensor_if_compared_value` `_GstTensorIf::cv`

compared value

Definition at line 133 of file `gsttensor_if.h`.

8.42.2.6 cv_option

GList* _GstTensorIf::cv_option

Definition at line 138 of file gsttensor_if.h.

8.42.2.7 element

GstElement _GstTensorIf::element

This is the parent object

Definition at line 124 of file gsttensor_if.h.

8.42.2.8 else_option

GList* _GstTensorIf::else_option

Definition at line 140 of file gsttensor_if.h.

8.42.2.9 in_config

GstTensorsConfig _GstTensorIf::in_config

input tensor info

Definition at line 129 of file gsttensor_if.h.

8.42.2.10 lock

GMutex _GstTensorIf::lock

Lock for custom callback

Definition at line 145 of file gsttensor_if.h.

8.42.2.11 `num_srcpads`

`guint32 _GstTensorIf::num_srcpads`

Definition at line 131 of file `gsttensor_if.h`.

8.42.2.12 `op`

`tensor_if_operator _GstTensorIf::op`

Definition at line 134 of file `gsttensor_if.h`.

8.42.2.13 `out_config`

`GstTensorsConfig _GstTensorIf::out_config[2]`

output tensor info

Definition at line 130 of file `gsttensor_if.h`.

8.42.2.14 `silent`

`gboolean _GstTensorIf::silent`

Definition at line 127 of file `gsttensor_if.h`.

8.42.2.15 `sinkpad`

`GstPad* _GstTensorIf::sinkpad`

Definition at line 125 of file `gsttensor_if.h`.

8.42.2.16 `srcpads`

`GSLIST* _GstTensorIf::srcpads`

Definition at line 126 of file `gsttensor_if.h`.

8.42.2.17 sv

```
tensor_if_sv_s _GstTensorIf::sv[2]
```

Definition at line 137 of file gsttensor_if.h.

8.42.2.18 then_option

```
GList* _GstTensorIf::then_option
```

Definition at line 139 of file gsttensor_if.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_if.h](#)

8.43 _GstTensorIfClass Struct Reference

GstTensorIfClass inherits GstElementClass.

```
#include <gsttensor_if.h>
```

Public Attributes

- GstElementClass [parent_class](#)

8.43.1 Detailed Description

GstTensorIfClass inherits GstElementClass.

Definition at line 151 of file gsttensor_if.h.

8.43.2 Member Data Documentation

8.43.2.1 parent_class

```
GstElementClass _GstTensorIfClass::parent_class
```

Inherits GstElementClass

Definition at line 153 of file gsttensor_if.h.

The documentation for this struct was generated from the following file:

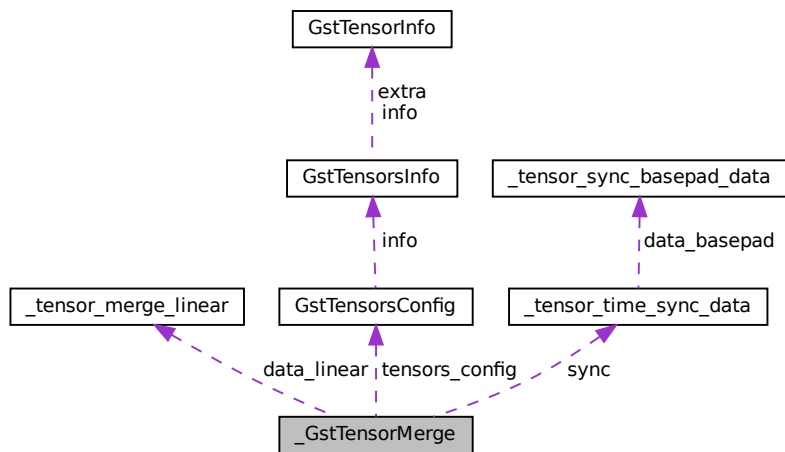
- [nnstreamer/elements/gsttensor_if.h](#)

8.44 _GstTensorMerge Struct Reference

Tensor Merge data structure.

```
#include <gsttensor_merge.h>
```

Collaboration diagram for _GstTensorMerge:



Public Attributes

- GstElement [element](#)
- gboolean [silent](#)
- [tensor_time_sync_data](#) [sync](#)
- GstPad * [srcpad](#)
- gchar * [option](#)
- [tensor_merge_mode](#) [mode](#)
- union {
 - [tensor_merge_linear](#) [data_linear](#)
- };
- gboolean [loaded](#)
- GstCollectPads * [collect](#)
- gboolean [negotiated](#)
- gboolean [need_segment](#)
- gboolean [need_stream_start](#)
- gboolean [send_stream_start](#)
- GstClockTime [current_time](#)
- gboolean [need_set_time](#)
- [GstTensorsConfig](#) [tensors_config](#)

8.44.1 Detailed Description

Tensor Merge data structure.

Definition at line 71 of file `gsttensor_merge.h`.

8.44.2 Member Data Documentation

8.44.2.1 "@22

```
union { ... }
```

8.44.2.2 collect

```
GstCollectPads* _GstTensorMerge::collect
```

Definition at line 85 of file gsttensor_merge.h.

8.44.2.3 current_time

```
GstClockTime _GstTensorMerge::current_time
```

Definition at line 91 of file gsttensor_merge.h.

8.44.2.4 data_linear

```
tensor_merge_linear _GstTensorMerge::data_linear
```

Definition at line 81 of file gsttensor_merge.h.

8.44.2.5 element

```
GstElement _GstTensorMerge::element
```

Definition at line 73 of file gsttensor_merge.h.

8.44.2.6 loaded

```
gboolean _GstTensorMerge::loaded
```

Definition at line 84 of file gsttensor_merge.h.

8.44.2.7 `mode`

`tensor_merge_mode` `_GstTensorMerge::mode`

Definition at line 79 of file `gsttensor_merge.h`.

8.44.2.8 `need_segment`

`gboolean` `_GstTensorMerge::need_segment`

Definition at line 87 of file `gsttensor_merge.h`.

8.44.2.9 `need_set_time`

`gboolean` `_GstTensorMerge::need_set_time`

Definition at line 92 of file `gsttensor_merge.h`.

8.44.2.10 `need_stream_start`

`gboolean` `_GstTensorMerge::need_stream_start`

Definition at line 88 of file `gsttensor_merge.h`.

8.44.2.11 `negotiated`

`gboolean` `_GstTensorMerge::negotiated`

Definition at line 86 of file `gsttensor_merge.h`.

8.44.2.12 `option`

`gchar*` `_GstTensorMerge::option`

Definition at line 78 of file `gsttensor_merge.h`.

8.44.2.13 send_stream_start

```
gboolean _GstTensorMerge::send_stream_start
```

Definition at line 89 of file `gsttensor_merge.h`.

8.44.2.14 silent

```
gboolean _GstTensorMerge::silent
```

Definition at line 75 of file `gsttensor_merge.h`.

8.44.2.15 srcpad

```
GstPad* _GstTensorMerge::srcpad
```

Definition at line 77 of file `gsttensor_merge.h`.

8.44.2.16 sync

```
tensor_time_sync_data _GstTensorMerge::sync
```

Definition at line 76 of file `gsttensor_merge.h`.

8.44.2.17 tensors_config

```
GstTensorsConfig _GstTensorMerge::tensors_config
```

output tensors info

Definition at line 93 of file `gsttensor_merge.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_merge.h](#)

8.45 _GstTensorMergeClass Struct Reference

GstTensorMergeClass inherits GstElementClass.

```
#include <gsttensor_merge.h>
```


Public Attributes

- GstElementClass [parent_class](#)

8.45.1 Detailed Description

GstTensorMergeClass inherits GstElementClass.

Definition at line 99 of file gsttensor_merge.h.

8.45.2 Member Data Documentation

8.45.2.1 parent_class

GstElementClass `_GstTensorMergeClass::parent_class`

Definition at line 101 of file gsttensor_merge.h.

The documentation for this struct was generated from the following file:

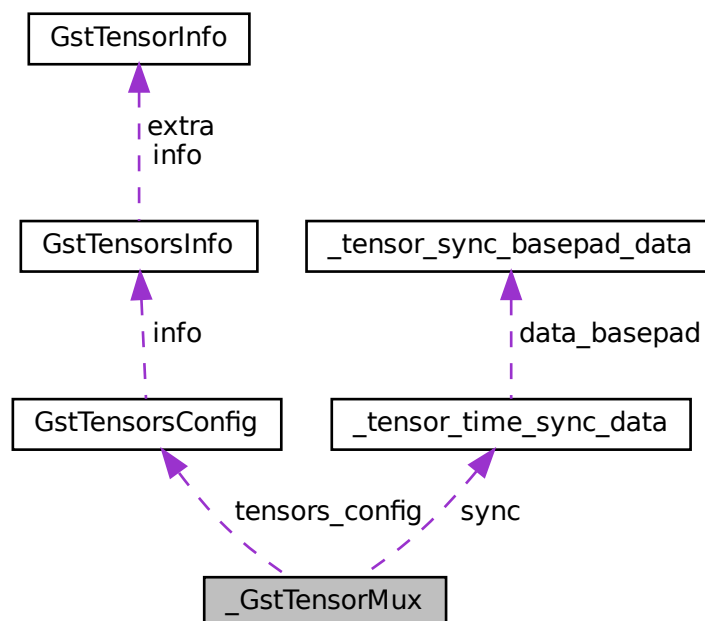
- `nnstreamer/elements/gsttensor_merge.h`

8.46 _GstTensorMux Struct Reference

Tensor Muxer data structure.

```
#include <gsttensor_mux.h>
```

Collaboration diagram for `_GstTensorMux`:



Public Attributes

- GstElement [element](#)
- gboolean [silent](#)
- [tensor_time_sync_data](#) sync
- GstPad * [srcpad](#)
- GstCollectPads * [collect](#)
- gboolean [negotiated](#)
- gboolean [need_segment](#)
- gboolean [need_stream_start](#)
- gboolean [send_stream_start](#)
- gboolean [need_set_time](#)
- GstClockTime [current_time](#)
- [GstTensorsConfig](#) [tensors_config](#)

8.46.1 Detailed Description

Tensor Muxer data structure.

Definition at line 48 of file `gsttensor_mux.h`.

8.46.2 Member Data Documentation

8.46.2.1 collect

```
GstCollectPads* _GstTensorMux::collect
```

Definition at line 56 of file `gsttensor_mux.h`.

8.46.2.2 current_time

```
GstClockTime _GstTensorMux::current_time
```

Definition at line 62 of file `gsttensor_mux.h`.

8.46.2.3 element

```
GstElement _GstTensorMux::element
```

Definition at line 50 of file `gsttensor_mux.h`.

8.46.2.4 `need_segment`

`gboolean _GstTensorMux::need_segment`

Definition at line 58 of file `gsttensor_mux.h`.

8.46.2.5 `need_set_time`

`gboolean _GstTensorMux::need_set_time`

Definition at line 61 of file `gsttensor_mux.h`.

8.46.2.6 `need_stream_start`

`gboolean _GstTensorMux::need_stream_start`

Definition at line 59 of file `gsttensor_mux.h`.

8.46.2.7 `negotiated`

`gboolean _GstTensorMux::negotiated`

Definition at line 57 of file `gsttensor_mux.h`.

8.46.2.8 `send_stream_start`

`gboolean _GstTensorMux::send_stream_start`

Definition at line 60 of file `gsttensor_mux.h`.

8.46.2.9 `silent`

`gboolean _GstTensorMux::silent`

Definition at line 52 of file `gsttensor_mux.h`.

8.46.2.10 srcpad

`GstPad* _GstTensorMux::srcpad`

Definition at line 54 of file `gsttensor_mux.h`.

8.46.2.11 sync

`tensor_time_sync_data _GstTensorMux::sync`

Definition at line 53 of file `gsttensor_mux.h`.

8.46.2.12 tensors_config

`GstTensorsConfig _GstTensorMux::tensors_config`

output tensors info

Definition at line 64 of file `gsttensor_mux.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_mux.h](#)

8.47 _GstTensorMuxClass Struct Reference

`GstTensorMuxClass` inherits `GstElementClass`.

```
#include <gsttensor_mux.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.47.1 Detailed Description

`GstTensorMuxClass` inherits `GstElementClass`.

Definition at line 70 of file `gsttensor_mux.h`.

8.47.2 Member Data Documentation

8.47.2.1 `parent_class`

```
GstElementClass _GstTensorMuxClass::parent_class
```

Definition at line 72 of file `gsttensor_mux.h`.

The documentation for this struct was generated from the following file:

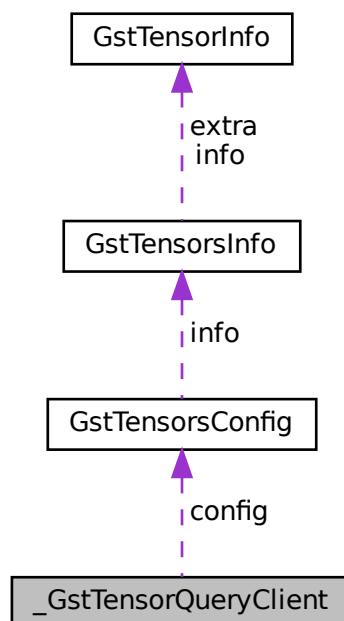
- [nnstreamer/elements/gsttensor_mux.h](#)

8.48 `_GstTensorQueryClient` Struct Reference

`GstTensorQueryClient` data structure.

```
#include <tensor_query_client.h>
```

Collaboration diagram for `_GstTensorQueryClient`:



Public Attributes

- GstElement [element](#)
- GstPad * [sinkpad](#)
- GstPad * [srcpad](#)
- gboolean [silent](#)
- gchar * [in_caps_str](#)
- gboolean [is_tensor](#)
- [GstTensorsConfig](#) [config](#)
- guint [timeout](#)
- gchar * [topic](#)
- gchar * [host](#)
- guint16 [port](#)
- gchar * [dest_host](#)
- guint16 [dest_port](#)
- nns_edge_connect_type_e [connect_type](#)
- nns_edge_h [edge_h](#)
- GAsyncQueue * [msg_queue](#)
- guint [max_request](#)
- guint [requested_num](#)

8.48.1 Detailed Description

GstTensorQueryClient data structure.

Definition at line 41 of file `tensor_query_client.h`.

8.48.2 Member Data Documentation

8.48.2.1 config

`GstTensorsConfig` `_GstTensorQueryClient::config`

Definition at line 50 of file `tensor_query_client.h`.

8.48.2.2 connect_type

`nns_edge_connect_type_e` `_GstTensorQueryClient::connect_type`

Definition at line 60 of file `tensor_query_client.h`.

8.48.2.3 `dest_host`

```
gchar* _GstTensorQueryClient::dest_host
```

Definition at line 57 of file `tensor_query_client.h`.

8.48.2.4 `dest_port`

```
guint16 _GstTensorQueryClient::dest_port
```

Definition at line 58 of file `tensor_query_client.h`.

8.48.2.5 `edge_h`

```
nns_edge_h _GstTensorQueryClient::edge_h
```

Definition at line 61 of file `tensor_query_client.h`.

8.48.2.6 `element`

```
GstElement _GstTensorQueryClient::element
```

parent object

Definition at line 43 of file `tensor_query_client.h`.

8.48.2.7 `host`

```
gchar* _GstTensorQueryClient::host
```

Definition at line 55 of file `tensor_query_client.h`.

8.48.2.8 `in_caps_str`

```
gchar* _GstTensorQueryClient::in_caps_str
```

Definition at line 48 of file `tensor_query_client.h`.

8.48.2.9 is_tensor

`gboolean _GstTensorQueryClient::is_tensor`

Definition at line 49 of file `tensor_query_client.h`.

8.48.2.10 max_request

`guint _GstTensorQueryClient::max_request`

Definition at line 64 of file `tensor_query_client.h`.

8.48.2.11 msg_queue

`GAsyncQueue* _GstTensorQueryClient::msg_queue`

Definition at line 62 of file `tensor_query_client.h`.

8.48.2.12 port

`guint16 _GstTensorQueryClient::port`

Definition at line 56 of file `tensor_query_client.h`.

8.48.2.13 requested_num

`guint _GstTensorQueryClient::requested_num`

Definition at line 65 of file `tensor_query_client.h`.

8.48.2.14 silent

`gboolean _GstTensorQueryClient::silent`

True if logging is minimized

Definition at line 47 of file `tensor_query_client.h`.

8.48.2.15 `sinkpad`

```
GstPad* _GstTensorQueryClient::sinkpad
```

sink pad

Definition at line 44 of file `tensor_query_client.h`.

8.48.2.16 `srcpad`

```
GstPad* _GstTensorQueryClient::srcpad
```

src pad

Definition at line 45 of file `tensor_query_client.h`.

8.48.2.17 `timeout`

```
guint _GstTensorQueryClient::timeout
```

timeout value (in ms) to wait message from server

Definition at line 51 of file `tensor_query_client.h`.

8.48.2.18 `topic`

```
gchar* _GstTensorQueryClient::topic
```

Main operation such as 'object_detection' or 'image_segmentation'

Definition at line 54 of file `tensor_query_client.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_query/tensor_query_client.h](#)

8.49 `_GstTensorQueryClientClass` Struct Reference

`GstTensorQueryClientClass` data structure.

```
#include <tensor_query_client.h>
```

Public Attributes

- GstElementClass [parent_class](#)

8.49.1 Detailed Description

GstTensorQueryClientClass data structure.

Definition at line 71 of file `tensor_query_client.h`.

8.49.2 Member Data Documentation

8.49.2.1 `parent_class`

```
GstElementClass _GstTensorQueryClientClass::parent_class
```

parent class

Definition at line 73 of file `tensor_query_client.h`.

The documentation for this struct was generated from the following file:

- `nnsreamer/tensor_query/tensor_query_client.h`

8.50 `_GstTensorQueryServerSink` Struct Reference

GstTensorQueryServerSink data structure.

```
#include <tensor_query_serversink.h>
```

Public Attributes

- GstBaseSink [element](#)
- guint [sink_id](#)
- guint [timeout](#)
- gint [metaless_frame_limit](#)
- gint [metaless_frame_count](#)
- nns_edge_connect_type_e [connect_type](#)

8.50.1 Detailed Description

GstTensorQueryServerSink data structure.

Definition at line 40 of file `tensor_query_serversink.h`.

8.50.2 Member Data Documentation

8.50.2.1 connect_type

nns_edge_connect_type_e _GstTensorQueryServerSink::connect_type

Definition at line 49 of file tensor_query_serversink.h.

8.50.2.2 element

GstBaseSink _GstTensorQueryServerSink::element

parent object

Definition at line 42 of file tensor_query_serversink.h.

8.50.2.3 metaless_frame_count

gint _GstTensorQueryServerSink::metaless_frame_count

Definition at line 47 of file tensor_query_serversink.h.

8.50.2.4 metaless_frame_limit

gint _GstTensorQueryServerSink::metaless_frame_limit

Definition at line 46 of file tensor_query_serversink.h.

8.50.2.5 sink_id

guint _GstTensorQueryServerSink::sink_id

Definition at line 43 of file tensor_query_serversink.h.

8.50.2.6 timeout

```
guint _GstTensorQueryServerSink::timeout
```

Definition at line 45 of file `tensor_query_serversink.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_query/tensor_query_serversink.h](#)

8.51 _GstTensorQueryServerSinkClass Struct Reference

GstTensorQueryServerSinkClass data structure.

```
#include <tensor_query_serversink.h>
```

Public Attributes

- GstBaseSinkClass [parent_class](#)

8.51.1 Detailed Description

GstTensorQueryServerSinkClass data structure.

Definition at line 55 of file `tensor_query_serversink.h`.

8.51.2 Member Data Documentation

8.51.2.1 parent_class

```
GstBaseSinkClass _GstTensorQueryServerSinkClass::parent_class
```

parent class

Definition at line 57 of file `tensor_query_serversink.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_query/tensor_query_serversink.h](#)

8.52 _GstTensorQueryServerSrc Struct Reference

GstTensorQueryServerSrc data structure.

```
#include <tensor_query_serversrc.h>
```

Public Attributes

- GstPushSrc [element](#)
- guint [src_id](#)
- gboolean [configured](#)
- gchar * [host](#)
- guint16 [port](#)
- gchar * [dest_host](#)
- guint16 [dest_port](#)
- guint [timeout](#)
- gchar * [topic](#)
- nns_edge_connect_type_e [connect_type](#)
- GAsyncQueue * [msg_queue](#)
- gboolean [playing](#)

8.52.1 Detailed Description

GstTensorQueryServerSrc data structure.

Definition at line 39 of file `tensor_query_serversrc.h`.

8.52.2 Member Data Documentation

8.52.2.1 configured

```
gboolean _GstTensorQueryServerSrc::configured
```

Definition at line 43 of file `tensor_query_serversrc.h`.

8.52.2.2 connect_type

```
nns_edge_connect_type_e _GstTensorQueryServerSrc::connect_type
```

Definition at line 54 of file `tensor_query_serversrc.h`.

8.52.2.3 dest_host

`gchar* _GstTensorQueryServerSrc::dest_host`

Definition at line 47 of file `tensor_query_serversrc.h`.

8.52.2.4 dest_port

`guint16 _GstTensorQueryServerSrc::dest_port`

Definition at line 48 of file `tensor_query_serversrc.h`.

8.52.2.5 element

`GstPushSrc _GstTensorQueryServerSrc::element`

Definition at line 41 of file `tensor_query_serversrc.h`.

8.52.2.6 host

`gchar* _GstTensorQueryServerSrc::host`

Definition at line 45 of file `tensor_query_serversrc.h`.

8.52.2.7 msg_queue

`GAsyncQueue* _GstTensorQueryServerSrc::msg_queue`

Definition at line 55 of file `tensor_query_serversrc.h`.

8.52.2.8 playing

`gboolean _GstTensorQueryServerSrc::playing`

Definition at line 56 of file `tensor_query_serversrc.h`.

8.52.2.9 `port`

```
guint16 _GstTensorQueryServerSrc::port
```

Definition at line 46 of file `tensor_query_serversrc.h`.

8.52.2.10 `src_id`

```
guint _GstTensorQueryServerSrc::src_id
```

Definition at line 42 of file `tensor_query_serversrc.h`.

8.52.2.11 `timeout`

```
guint _GstTensorQueryServerSrc::timeout
```

Definition at line 49 of file `tensor_query_serversrc.h`.

8.52.2.12 `topic`

```
gchar* _GstTensorQueryServerSrc::topic
```

Main operation such as 'object_detection' or 'image_segmentation'

Definition at line 52 of file `tensor_query_serversrc.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/tensor_query/tensor_query_serversrc.h`

8.53 `_GstTensorQueryServerSrcClass` Struct Reference

`GstTensorQueryServerSrcClass` data structure.

```
#include <tensor_query_serversrc.h>
```

Public Attributes

- `GstPushSrcClass` [parent_class](#)

8.53.1 Detailed Description

GstTensorQueryServerSrcClass data structure.

Definition at line 62 of file tensor_query_serversrc.h.

8.53.2 Member Data Documentation

8.53.2.1 parent_class

```
GstPushSrcClass _GstTensorQueryServerSrcClass::parent_class
```

parent class

Definition at line 64 of file tensor_query_serversrc.h.

The documentation for this struct was generated from the following file:

- [nntstreamer/tensor_query/tensor_query_serversrc.h](#)

8.54 _GstTensorRate Struct Reference

Tensor Rate data structure.

```
#include <gsttensor_rate.h>
```

Public Attributes

- GstBaseTransform [element](#)
- GstBuffer * [prevbuf](#)
- GstSegment [segment](#)
- guint64 [out_frame_count](#)
- gboolean [sent_qos_on_passthrough](#)
- gint [from_rate_numerator](#)
- gint [from_rate_denominator](#)
- gint [to_rate_numerator](#)
- gint [to_rate_denominator](#)
- guint64 [base_ts](#)
- guint64 [prev_ts](#)
- guint64 [next_ts](#)
- guint64 [last_ts](#)
- guint64 [in](#)
- guint64 [out](#)
- guint64 [dup](#)
- guint64 [drop](#)
- gint [rate_n](#)
- gint [rate_d](#)
- gboolean [silent](#)
- gboolean [throttle](#)

8.54.1 Detailed Description

Tensor Rate data structure.

Definition at line 37 of file `gsttensor_rate.h`.

8.54.2 Member Data Documentation

8.54.2.1 `base_ts`

```
guint64 _GstTensorRate::base_ts
```

Timestamp used in `next_ts` calculation

Definition at line 55 of file `gsttensor_rate.h`.

8.54.2.2 `drop`

```
guint64 _GstTensorRate::drop
```

stat property

Definition at line 61 of file `gsttensor_rate.h`.

8.54.2.3 `dup`

```
guint64 _GstTensorRate::dup
```

Definition at line 61 of file `gsttensor_rate.h`.

8.54.2.4 `element`

```
GstBaseTransform _GstTensorRate::element
```

This is the parent object

Definition at line 39 of file `gsttensor_rate.h`.

8.54.2.5 from_rate_denominator

```
gint _GstTensorRate::from_rate_denominator
```

framerate denominator (From)

Definition at line 49 of file gsttensor_rate.h.

8.54.2.6 from_rate_numerator

```
gint _GstTensorRate::from_rate_numerator
```

Caps negotiation framerate numerator (From)

Definition at line 48 of file gsttensor_rate.h.

8.54.2.7 in

```
guint64 _GstTensorRate::in
```

Properties

Definition at line 61 of file gsttensor_rate.h.

8.54.2.8 last_ts

```
guint64 _GstTensorRate::last_ts
```

Timestamp of last input buffer

Definition at line 58 of file gsttensor_rate.h.

8.54.2.9 next_ts

```
guint64 _GstTensorRate::next_ts
```

Timestamp of next buffer to output

Definition at line 57 of file gsttensor_rate.h.

8.54.2.10 `out`

```
guint64 _GstTensorRate::out
```

Definition at line 61 of file `gsttensor_rate.h`.

8.54.2.11 `out_frame_count`

```
guint64 _GstTensorRate::out_frame_count
```

number of frames output

Definition at line 43 of file `gsttensor_rate.h`.

8.54.2.12 `prev_ts`

```
guint64 _GstTensorRate::prev_ts
```

Previous buffer timestamp

Definition at line 56 of file `gsttensor_rate.h`.

8.54.2.13 `prevbuf`

```
GstBuffer* _GstTensorRate::prevbuf
```

previous buffer

Definition at line 41 of file `gsttensor_rate.h`.

8.54.2.14 `rate_d`

```
gint _GstTensorRate::rate_d
```

framerate property

Definition at line 62 of file `gsttensor_rate.h`.

8.54.2.15 rate_n

`gint _GstTensorRate::rate_n`

Definition at line 62 of file `gsttensor_rate.h`.

8.54.2.16 segment

`GstSegment _GstTensorRate::segment`

current segment

Definition at line 42 of file `gsttensor_rate.h`.

8.54.2.17 sent_qos_on_passthrough

`gboolean _GstTensorRate::sent_qos_on_passthrough`

qos event on passthrough

Definition at line 45 of file `gsttensor_rate.h`.

8.54.2.18 silent

`gboolean _GstTensorRate::silent`

debug property

Definition at line 63 of file `gsttensor_rate.h`.

8.54.2.19 throttle

`gboolean _GstTensorRate::throttle`

throttle property

Definition at line 64 of file `gsttensor_rate.h`.

8.54.2.20 `to_rate_denominator`

```
gint _GstTensorRate::to_rate_denominator
```

framerate denominator (To)

Definition at line 52 of file `gsttensor_rate.h`.

8.54.2.21 `to_rate_numerator`

```
gint _GstTensorRate::to_rate_numerator
```

framerate numerator (To)

Definition at line 51 of file `gsttensor_rate.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_rate.h`

8.55 `_GstTensorRateClass` Struct Reference

`GstTensorRateClass` inherits `GstElementClass`.

```
#include <gsttensor_rate.h>
```

Public Attributes

- `GstBaseTransformClass` `parent_class`

8.55.1 Detailed Description

`GstTensorRateClass` inherits `GstElementClass`.

Definition at line 70 of file `gsttensor_rate.h`.

8.55.2 Member Data Documentation

8.55.2.1 parent_class

```
GstBaseTransformClass _GstTensorRateClass::parent_class
```

Inherits GstBaseTransformClass

Definition at line 72 of file gsttensor_rate.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_rate.h](#)

8.56 _GstTensorRepoSink Struct Reference

GstTensorRepoSink data structure.

```
#include <gsttensor_reposink.h>
```

Public Attributes

- GstBaseSink [element](#)
- gboolean [silent](#)
- guint [signal_rate](#)
- GstClockTime [last_render_time](#)
- GstCaps * [in_caps](#)
- gboolean [set_startid](#)
- guint [myid](#)
- guint [o_myid](#)

8.56.1 Detailed Description

GstTensorRepoSink data structure.

GstTensorRepoSink inherits GstBaseSink.

Definition at line 54 of file gsttensor_reposink.h.

8.56.2 Member Data Documentation

8.56.2.1 element

```
GstBaseSink _GstTensorRepoSink::element
```

Definition at line 56 of file gsttensor_reposink.h.

8.56.2.2 in_caps

GstCaps* _GstTensorRepoSink::in_caps

Definition at line 61 of file gsttensor_reposink.h.

8.56.2.3 last_render_time

GstClockTime _GstTensorRepoSink::last_render_time

Definition at line 60 of file gsttensor_reposink.h.

8.56.2.4 myid

guint _GstTensorRepoSink::myid

Definition at line 63 of file gsttensor_reposink.h.

8.56.2.5 o_myid

guint _GstTensorRepoSink::o_myid

Definition at line 64 of file gsttensor_reposink.h.

8.56.2.6 set_startid

gboolean _GstTensorRepoSink::set_startid

Definition at line 62 of file gsttensor_reposink.h.

8.56.2.7 signal_rate

guint _GstTensorRepoSink::signal_rate

Definition at line 59 of file gsttensor_reposink.h.

8.56.2.8 silent

```
gboolean _GstTensorRepoSink::silent
```

Definition at line 58 of file `gsttensor_reposink.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_reposink.h](#)

8.57 _GstTensorRepoSinkClass Struct Reference

`GstTensorRepoSinkClass` data structure.

```
#include <gsttensor_reposink.h>
```

Public Attributes

- `GstBaseSinkClass` [parent_class](#)

8.57.1 Detailed Description

`GstTensorRepoSinkClass` data structure.

`GstTensorRepoSink` inherits `GstBaseSink`.

Definition at line 72 of file `gsttensor_reposink.h`.

8.57.2 Member Data Documentation

8.57.2.1 parent_class

```
GstBaseSinkClass _GstTensorRepoSinkClass::parent_class
```

Definition at line 74 of file `gsttensor_reposink.h`.

The documentation for this struct was generated from the following file:

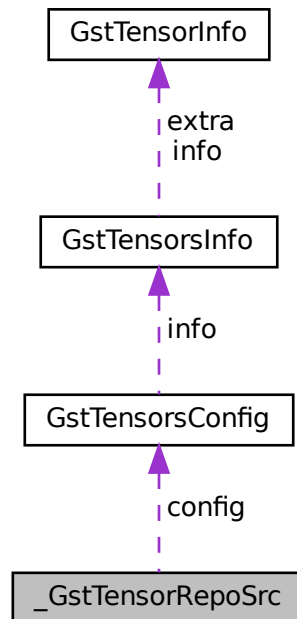
- [nnstreamer/elements/gsttensor_reposink.h](#)

8.58 _GstTensorRepoSrc Struct Reference

GstTensorRepoSrc data structure.

```
#include <gsttensor_reposrc.h>
```

Collaboration diagram for _GstTensorRepoSrc:



Public Attributes

- GstPushSrc [parent](#)
- [GstTensorsConfig config](#)
- gboolean [silent](#)
- guint [myid](#)
- guint [o_myid](#)
- GstCaps * [caps](#)
- gboolean [ini](#)
- gint [fps_n](#)
- gint [fps_d](#)
- gboolean [negotiation](#)
- gboolean [set_startid](#)

8.58.1 Detailed Description

GstTensorRepoSrc data structure.

GstTensorRepoSrc inherits GstPushSrc

Definition at line 54 of file `gsttensor_reposrc.h`.

8.58.2 Member Data Documentation

8.58.2.1 caps

`GstCaps* _GstTensorRepoSrc::caps`

Definition at line 61 of file `gsttensor_reposrc.h`.

8.58.2.2 config

`GstTensorsConfig _GstTensorRepoSrc::config`

Definition at line 57 of file `gsttensor_reposrc.h`.

8.58.2.3 fps_d

`gint _GstTensorRepoSrc::fps_d`

Definition at line 64 of file `gsttensor_reposrc.h`.

8.58.2.4 fps_n

`gint _GstTensorRepoSrc::fps_n`

Definition at line 63 of file `gsttensor_reposrc.h`.

8.58.2.5 ini

`gboolean _GstTensorRepoSrc::ini`

Definition at line 62 of file `gsttensor_reposrc.h`.

8.58.2.6 `myid`

```
guint _GstTensorRepoSrc::myid
```

Definition at line 59 of file `gsttensor_reposrc.h`.

8.58.2.7 `negotiation`

```
gboolean _GstTensorRepoSrc::negotiation
```

Definition at line 65 of file `gsttensor_reposrc.h`.

8.58.2.8 `o_myid`

```
guint _GstTensorRepoSrc::o_myid
```

Definition at line 60 of file `gsttensor_reposrc.h`.

8.58.2.9 `parent`

```
GstPushSrc _GstTensorRepoSrc::parent
```

Definition at line 56 of file `gsttensor_reposrc.h`.

8.58.2.10 `set_startid`

```
gboolean _GstTensorRepoSrc::set_startid
```

Definition at line 66 of file `gsttensor_reposrc.h`.

8.58.2.11 `silent`

```
gboolean _GstTensorRepoSrc::silent
```

Definition at line 58 of file `gsttensor_reposrc.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_reposrc.h](#)

8.59 `_GstTensorRepoSrcClass` Struct Reference

`GstTensorRepoSrcClass` data structure.

```
#include <gsttensor_reposrc.h>
```

Public Attributes

- `GstPushSrcClass` [parent_class](#)

8.59.1 Detailed Description

`GstTensorRepoSrcClass` data structure.

`GstTensorRepoSrc` inherits `GstPushSrc`

Definition at line 74 of file `gsttensor_reposrc.h`.

8.59.2 Member Data Documentation

8.59.2.1 `parent_class`

```
GstPushSrcClass _GstTensorRepoSrcClass::parent_class
```

Definition at line 76 of file `gsttensor_reposrc.h`.

The documentation for this struct was generated from the following file:

- `nntstreamer/elements/gsttensor_reposrc.h`

8.60 `_GstTensorSink` Struct Reference

`GstTensorSink` data structure.

```
#include <gsttensor_sink.h>
```

Public Attributes

- `GstBaseSink` [element](#)
- `GMutex` [mutex](#)
- `gboolean` [silent](#)
- `gboolean` [emit_signal](#)
- `guint` [signal_rate](#)
- `GstClockTime` [last_render_time](#)

8.60.1 Detailed Description

`GstTensorSink` data structure.

`GstTensorSink` inherits `GstBaseSink`.

Definition at line 55 of file `gsttensor_sink.h`.

8.60.2 Member Data Documentation

8.60.2.1 `element`

`GstBaseSink` `_GstTensorSink::element`

parent object

Definition at line 57 of file `gsttensor_sink.h`.

8.60.2.2 `emit_signal`

`gboolean` `_GstTensorSink::emit_signal`

true to emit signal for new data, eos

Definition at line 61 of file `gsttensor_sink.h`.

8.60.2.3 `last_render_time`

`GstClockTime` `_GstTensorSink::last_render_time`

buffer rendered time

Definition at line 63 of file `gsttensor_sink.h`.

8.60.2.4 `mutex`

`GMutex` `_GstTensorSink::mutex`

mutex for processing

Definition at line 59 of file `gsttensor_sink.h`.

8.60.2.5 `signal_rate`

```
guint _GstTensorSink::signal_rate
```

new data signals per second

Definition at line 62 of file `gsttensor_sink.h`.

8.60.2.6 `silent`

```
gboolean _GstTensorSink::silent
```

true to print minimized log

Definition at line 60 of file `gsttensor_sink.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_sink.h](#)

8.61 `_GstTensorSinkClass` Struct Reference

`GstTensorSinkClass` data structure.

```
#include <gsttensor_sink.h>
```

Public Attributes

- `GstBaseSinkClass` [parent_class](#)
- `void(* new_data)(GstElement *element, GstBuffer *buffer)`
- `void(* stream_start)(GstElement *element)`
- `void(* eos)(GstElement *element)`

8.61.1 Detailed Description

`GstTensorSinkClass` data structure.

`GstTensorSink` inherits `GstBaseSink`.

Definition at line 71 of file `gsttensor_sink.h`.

8.61.2 Member Data Documentation

8.61.2.1 `eos`

```
void(* _GstTensorSinkClass::eos) (GstElement *element)
```

signal when end of stream reached

Definition at line 78 of file `gsttensor_sink.h`.

8.61.2.2 `new_data`

```
void(* _GstTensorSinkClass::new_data) (GstElement *element, GstBuffer *buffer)
```

signals signal when new data received

Definition at line 76 of file `gsttensor_sink.h`.

8.61.2.3 `parent_class`

```
GstBaseSinkClass _GstTensorSinkClass::parent_class
```

parent class

Definition at line 73 of file `gsttensor_sink.h`.

8.61.2.4 `stream_start`

```
void(* _GstTensorSinkClass::stream_start) (GstElement *element)
```

signal when stream started

Definition at line 77 of file `gsttensor_sink.h`.

The documentation for this struct was generated from the following file:

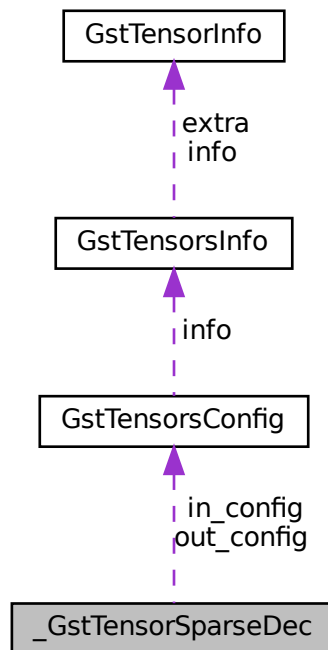
- [nnstreamer/elements/gsttensor_sink.h](#)

8.62 `_GstTensorSparseDec` Struct Reference

`GstTensorSparseDec` data structure.

```
#include <gsttensor_sparsedec.h>
```

Collaboration diagram for `_GstTensorSparseDec`:



Public Attributes

- `GstElement` `element`
- `GstPad` * `sinkpad`
- `GstPad` * `srcpad`
- `GstTensorsConfig` `in_config`
- `GstTensorsConfig` `out_config`
- `gboolean` `silent`

8.62.1 Detailed Description

`GstTensorSparseDec` data structure.

Definition at line 39 of file `gsttensor_sparsedec.h`.

8.62.2 Member Data Documentation

8.62.2.1 element

`GstElement` `_GstTensorSparseDec::element`

parent object

Definition at line 41 of file `gsttensor_sparsedec.h`.

8.62.2.2 in_config

`GstTensorsConfig` `_GstTensorSparseDec::in_config`

input tensors config

Definition at line 46 of file `gsttensor_sparsedec.h`.

8.62.2.3 out_config

`GstTensorsConfig` `_GstTensorSparseDec::out_config`

output tensors config

Definition at line 47 of file `gsttensor_sparsedec.h`.

8.62.2.4 silent

`gboolean` `_GstTensorSparseDec::silent`

true to print minimized log

Definition at line 48 of file `gsttensor_sparsedec.h`.

8.62.2.5 sinkpad

`GstPad*` `_GstTensorSparseDec::sinkpad`

sink pad

Definition at line 42 of file `gsttensor_sparsedec.h`.

8.62.2.6 srcpad

```
GstPad* _GstTensorSparseDec::srcpad
```

src pad

Definition at line 43 of file gsttensor_sparsedec.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_sparsedec.h](#)

8.63 _GstTensorSparseDecClass Struct Reference

GstTensorSparseClass data structure.

```
#include <gsttensor_sparsedec.h>
```

Public Attributes

- GstElementClass [parent_class](#)

8.63.1 Detailed Description

GstTensorSparseClass data structure.

Definition at line 54 of file gsttensor_sparsedec.h.

8.63.2 Member Data Documentation

8.63.2.1 parent_class

```
GstElementClass _GstTensorSparseDecClass::parent_class
```

parent class

Definition at line 56 of file gsttensor_sparsedec.h.

The documentation for this struct was generated from the following file:

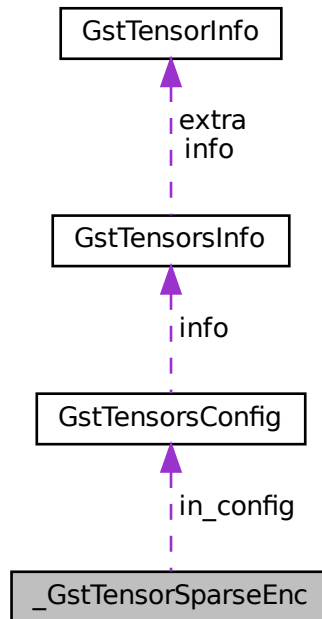
- [nnstreamer/elements/gsttensor_sparsedec.h](#)

8.64 _GstTensorSparseEnc Struct Reference

GstTensorSparseEnc data structure.

```
#include <gsttensor_sparseenc.h>
```

Collaboration diagram for _GstTensorSparseEnc:



Public Attributes

- GstElement [element](#)
- GstPad * [sinkpad](#)
- GstPad * [srcpad](#)
- [GstTensorsConfig](#) `in_config`
- gboolean [silent](#)

8.64.1 Detailed Description

GstTensorSparseEnc data structure.

Definition at line 39 of file `gsttensor_sparseenc.h`.

8.64.2 Member Data Documentation

8.64.2.1 element

`GstElement _GstTensorSparseEnc::element`

parent object

Definition at line 41 of file `gsttensor_sparseenc.h`.

8.64.2.2 in_config

`GstTensorsConfig _GstTensorSparseEnc::in_config`

input tensors config

Definition at line 46 of file `gsttensor_sparseenc.h`.

8.64.2.3 silent

`gboolean _GstTensorSparseEnc::silent`

true to print minimized log

Definition at line 47 of file `gsttensor_sparseenc.h`.

8.64.2.4 sinkpad

`GstPad* _GstTensorSparseEnc::sinkpad`

sink pad

Definition at line 42 of file `gsttensor_sparseenc.h`.

8.64.2.5 srcpad

`GstPad* _GstTensorSparseEnc::srcpad`

src pad

Definition at line 43 of file `gsttensor_sparseenc.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_sparseenc.h](#)

8.65 `_GstTensorSparseEncClass` Struct Reference

`GstTensorSparseClass` data structure.

```
#include <gsttensor_sparseenc.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.65.1 Detailed Description

`GstTensorSparseClass` data structure.

Definition at line 53 of file `gsttensor_sparseenc.h`.

8.65.2 Member Data Documentation

8.65.2.1 `parent_class`

```
GstElementClass _GstTensorSparseEncClass::parent_class
```

parent class

Definition at line 55 of file `gsttensor_sparseenc.h`.

The documentation for this struct was generated from the following file:

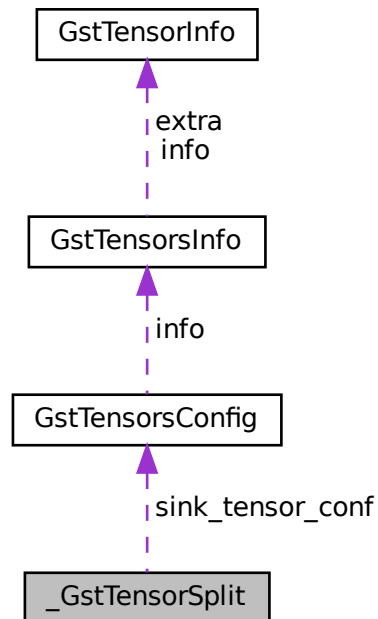
- `nntstreamer/elements/gsttensor_sparseenc.h`

8.66 `_GstTensorSplit` Struct Reference

Tensor Split data structure.

```
#include <gsttensor_split.h>
```

Collaboration diagram for `_GstTensorSplit`:



Public Attributes

- `GstElement` `element`
- `gboolean` `silent`
- `GstPad *` `sinkpad`
- `GSList *` `srcpads`
- `guint32` `num_tensors`
- `guint32` `num_srcpads`
- `GList *` `tensorpick`
- `GArray *` `tensorseg`
- `gboolean` `have_group_id`
- `guint` `group_id`
- `GstTensorsConfig` `sink_tensor_conf`

8.66.1 Detailed Description

Tensor Split data structure.

Definition at line 50 of file `gsttensor_split.h`.

8.66.2 Member Data Documentation

8.66.2.1 element

GstElement _GstTensorSplit::element

Definition at line 52 of file gsttensor_split.h.

8.66.2.2 group_id

guint _GstTensorSplit::group_id

Definition at line 62 of file gsttensor_split.h.

8.66.2.3 have_group_id

gboolean _GstTensorSplit::have_group_id

Definition at line 61 of file gsttensor_split.h.

8.66.2.4 num_srcpads

guint32 _GstTensorSplit::num_srcpads

Definition at line 58 of file gsttensor_split.h.

8.66.2.5 num_tensors

guint32 _GstTensorSplit::num_tensors

Definition at line 57 of file gsttensor_split.h.

8.66.2.6 silent

`gboolean _GstTensorSplit::silent`

Definition at line 54 of file `gsttensor_split.h`.

8.66.2.7 sink_tensor_conf

`GstTensorsConfig _GstTensorSplit::sink_tensor_conf`

Definition at line 63 of file `gsttensor_split.h`.

8.66.2.8 sinkpad

`GstPad* _GstTensorSplit::sinkpad`

Definition at line 55 of file `gsttensor_split.h`.

8.66.2.9 srcpads

`GSList* _GstTensorSplit::srcpads`

Definition at line 56 of file `gsttensor_split.h`.

8.66.2.10 tensorpick

`GList* _GstTensorSplit::tensorpick`

Definition at line 59 of file `gsttensor_split.h`.

8.66.2.11 tensorseg

`GArray* _GstTensorSplit::tensorseg`

Definition at line 60 of file `gsttensor_split.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_split.h](#)

8.67 `_GstTensorSplitClass` Struct Reference

`GstTensorSplitClass` inherits `GstElementClass`.

```
#include <gsttensor_split.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.67.1 Detailed Description

`GstTensorSplitClass` inherits `GstElementClass`.

Definition at line 69 of file `gsttensor_split.h`.

8.67.2 Member Data Documentation

8.67.2.1 `parent_class`

```
GstElementClass _GstTensorSplitClass::parent_class
```

Definition at line 71 of file `gsttensor_split.h`.

The documentation for this struct was generated from the following file:

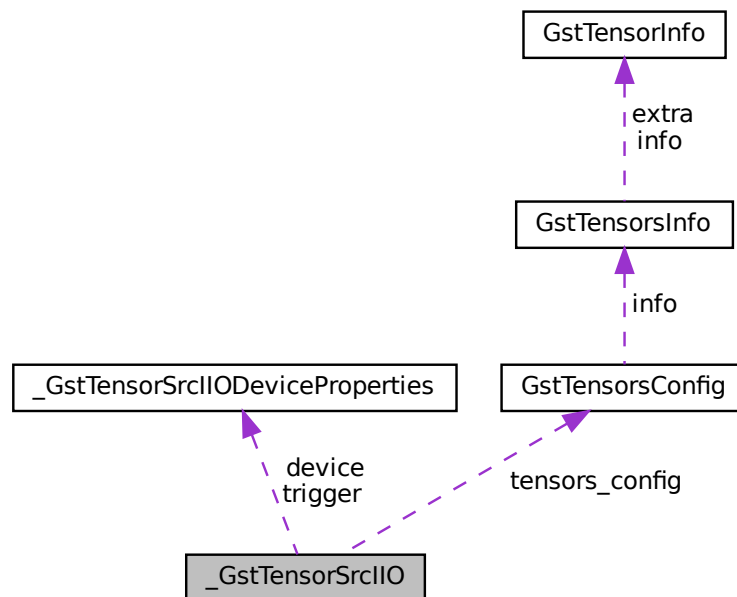
- `nnstreamer/elements/gsttensor_split.h`

8.68 `_GstTensorSrcIIO` Struct Reference

`GstTensorSrcIIO` data structure.

```
#include <gsttensor_srciiio.h>
```

Collaboration diagram for `_GstTensorSrcIO`:



Public Attributes

- `GstBaseSrc` `element`
- `gboolean` `silent`
- `gboolean` `configured`
- `gchar *` `mode`
- `gchar *` `base_dir`
- `gchar *` `dev_dir`
- `GstTensorSrcIODeviceProperties` `device`
- `GstTensorSrcIODeviceProperties` `trigger`
- `GList *` `channels`
- `GHashTable *` `custom_channel_table`
- `channels_enabled_options` `channels_enabled`
- `guint` `scan_size`
- `struct pollfd *` `buffer_data_fp`
- `guint` `num_channels_enabled`
- `gboolean` `merge_channels_data`
- `gboolean` `is_tensor`
- `guint` `buffer_capacity`
- `guint64` `sampling_frequency`
- `guint64` `default_sampling_frequency`
- `guint` `default_buffer_capacity`
- `gchar *` `default_trigger`
- `gint` `poll_timeout`
- `GstTensorsConfig *` `tensors_config`

8.68.1 Detailed Description

GstTensorSrcIIO data structure.

GstTensorSrcIIO inherits GstBaseSrcIIO.

Definition at line 104 of file gsttensor_srcio.h.

8.68.2 Member Data Documentation

8.68.2.1 base_dir

```
gchar* _GstTensorSrcIIO::base_dir
```

Base directory for IIO devices

Definition at line 114 of file gsttensor_srcio.h.

8.68.2.2 buffer_capacity

```
guint _GstTensorSrcIIO::buffer_capacity
```

size of the buffer

Definition at line 126 of file gsttensor_srcio.h.

8.68.2.3 buffer_data_fp

```
struct pollfd* _GstTensorSrcIIO::buffer_data_fp
```

pollfd for reading data buffer

Definition at line 122 of file gsttensor_srcio.h.

8.68.2.4 channels

```
GList* _GstTensorSrcIIO::channels
```

list of enabled channels

Definition at line 118 of file gsttensor_srcio.h.

8.68.2.5 channels_enabled

`channels_enabled_options` `_GstTensorSrcIIO::channels_enabled`

enabling which channels

Definition at line 120 of file `gsttensor_srcio.h`.

8.68.2.6 configured

`gboolean` `_GstTensorSrcIIO::configured`

true if device is configured and ready

Definition at line 110 of file `gsttensor_srcio.h`.

8.68.2.7 custom_channel_table

`GHashTable*` `_GstTensorSrcIIO::custom_channel_table`

table of idx of channels to be enabled

Definition at line 119 of file `gsttensor_srcio.h`.

8.68.2.8 default_buffer_capacity

`guint` `_GstTensorSrcIIO::default_buffer_capacity`

size of the buffer

Definition at line 130 of file `gsttensor_srcio.h`.

8.68.2.9 default_sampling_frequency

`guint64` `_GstTensorSrcIIO::default_sampling_frequency`

default set value of sampling frequency

Definition at line 129 of file `gsttensor_srcio.h`.

8.68.2.10 default_trigger

`gchar* _GstTensorSrcIIO::default_trigger`

default set value of sampling frequency

Definition at line 131 of file `gsttensor_srcio.h`.

8.68.2.11 dev_dir

`gchar* _GstTensorSrcIIO::dev_dir`

Directory for device files

Definition at line 115 of file `gsttensor_srcio.h`.

8.68.2.12 device

[GstTensorSrcIIODeviceProperties](#) `_GstTensorSrcIIO::device`

IIO device

Definition at line 116 of file `gsttensor_srcio.h`.

8.68.2.13 element

`GstBaseSrc _GstTensorSrcIIO::element`

parent class object

Definition at line 106 of file `gsttensor_srcio.h`.

8.68.2.14 is_tensor

`gboolean _GstTensorSrcIIO::is_tensor`

False if tensors is used for data

Definition at line 125 of file `gsttensor_srcio.h`.

8.68.2.15 merge_channels_data

`gboolean _GstTensorSrcIIO::merge_channels_data`

merge channel data with same type/size

Definition at line 124 of file `gsttensor_srcio.h`.

8.68.2.16 mode

`gchar* _GstTensorSrcIIO::mode`

linux IIO related properties IIO device operating mode

Definition at line 113 of file `gsttensor_srcio.h`.

8.68.2.17 num_channels_enabled

`guint _GstTensorSrcIIO::num_channels_enabled`

channels to be enabled

Definition at line 123 of file `gsttensor_srcio.h`.

8.68.2.18 poll_timeout

`gint _GstTensorSrcIIO::poll_timeout`

timeout for polling the fifo file

Definition at line 132 of file `gsttensor_srcio.h`.

8.68.2.19 sampling_frequency

`guint64 _GstTensorSrcIIO::sampling_frequency`

sampling frequency for the device

Definition at line 127 of file `gsttensor_srcio.h`.

8.68.2.20 scan_size

```
guint _GstTensorSrcIIO::scan_size
```

size for a single scan of buffer length 1

Definition at line 121 of file gsttensor_srcio.h.

8.68.2.21 silent

```
gboolean _GstTensorSrcIIO::silent
```

gststreamer related properties true to print minimized log

Definition at line 109 of file gsttensor_srcio.h.

8.68.2.22 tensors_config

```
GstTensorsConfig* _GstTensorSrcIIO::tensors_config
```

Only first element is filled when is_tensor is true tensors for storing data config

Definition at line 135 of file gsttensor_srcio.h.

8.68.2.23 trigger

```
GstTensorSrcIIODeviceProperties _GstTensorSrcIIO::trigger
```

IIO trigger

Definition at line 117 of file gsttensor_srcio.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_srcio.h](#)

8.69 _GstTensorSrcIIOChannelProperties Struct Reference

GstTensorSrcIIO channel's properties (internal data structure)

```
#include <gsttensor_srcio.h>
```

Public Attributes

- gboolean [enabled](#)
- gboolean [pre_enabled](#)
- gchar * [name](#)
- gchar * [generic_name](#)
- gchar * [base_dir](#)
- gchar * [base_file](#)
- gint [index](#)
- gboolean [big_endian](#)
- gboolean [is_signed](#)
- guint [used_bits](#)
- guint64 [mask](#)
- guint [storage_bytes](#)
- guint [storage_bits](#)
- guint [shift](#)
- guint [location](#)
- gfloat [offset](#)
- gfloat [scale](#)

8.69.1 Detailed Description

GstTensorSrcIIO channel's properties (internal data structure)

Definition at line 77 of file `gsttensor_srciiio.h`.

8.69.2 Member Data Documentation

8.69.2.1 `base_dir`

```
gchar* _GstTensorSrcIIOChannelProperties::base_dir
```

The base directory for the channel

Definition at line 83 of file `gsttensor_srciiio.h`.

8.69.2.2 `base_file`

```
gchar* _GstTensorSrcIIOChannelProperties::base_file
```

The base filename for the channel

Definition at line 84 of file `gsttensor_srciiio.h`.

8.69.2.3 big_endian

gboolean _GstTensorSrcIIOChannelProperties::big_endian

endian-ness of the data in buffer

Definition at line 87 of file gsttensor_srciiio.h.

8.69.2.4 enabled

gboolean _GstTensorSrcIIOChannelProperties::enabled

currently state enabled/disabled

Definition at line 79 of file gsttensor_srciiio.h.

8.69.2.5 generic_name

gchar* _GstTensorSrcIIOChannelProperties::generic_name

The generic name of the channel

Definition at line 82 of file gsttensor_srciiio.h.

8.69.2.6 index

gint _GstTensorSrcIIOChannelProperties::index

index of the channel in the buffer

Definition at line 85 of file gsttensor_srciiio.h.

8.69.2.7 is_signed

gboolean _GstTensorSrcIIOChannelProperties::is_signed

sign property of the data

Definition at line 88 of file gsttensor_srciiio.h.

8.69.2.8 location

```
guint _GstTensorSrcIIOChannelProperties::location
```

location of channel data in buffer

Definition at line 94 of file `gsttensor_srciiio.h`.

8.69.2.9 mask

```
guint64 _GstTensorSrcIIOChannelProperties::mask
```

size of the bits used for the data

Definition at line 90 of file `gsttensor_srciiio.h`.

8.69.2.10 name

```
gchar* _GstTensorSrcIIOChannelProperties::name
```

The name of the channel

Definition at line 81 of file `gsttensor_srciiio.h`.

8.69.2.11 offset

```
gfloat _GstTensorSrcIIOChannelProperties::offset
```

offset applied on raw data read from device

Definition at line 95 of file `gsttensor_srciiio.h`.

8.69.2.12 pre_enabled

```
gboolean _GstTensorSrcIIOChannelProperties::pre_enabled
```

already in enabled/disabled state

Definition at line 80 of file `gsttensor_srciiio.h`.

8.69.2.13 scale

```
gfloat _GstTensorSrcIIOChannelProperties::scale
```

scale applied on offset-ed data read from device

Definition at line 96 of file `gsttensor_srciiio.h`.

8.69.2.14 shift

```
guint _GstTensorSrcIIOChannelProperties::shift
```

shift to be applied on the read data

Definition at line 93 of file `gsttensor_srciiio.h`.

8.69.2.15 storage_bits

```
guint _GstTensorSrcIIOChannelProperties::storage_bits
```

exact bit size for the data

Definition at line 92 of file `gsttensor_srciiio.h`.

8.69.2.16 storage_bytes

```
guint _GstTensorSrcIIOChannelProperties::storage_bytes
```

total storage size for the data

Definition at line 91 of file `gsttensor_srciiio.h`.

8.69.2.17 used_bits

```
guint _GstTensorSrcIIOChannelProperties::used_bits
```

size of the bits used for the data

Definition at line 89 of file `gsttensor_srciiio.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_srciiio.h](#)

8.70 `_GstTensorSrcIIOClass` Struct Reference

`GstTensorSrcIIOClass` data structure.

```
#include <gsttensor_srciiio.h>
```

Public Attributes

- `GstBaseSrcClass` [parent_class](#)

8.70.1 Detailed Description

`GstTensorSrcIIOClass` data structure.

`GstTensorSrcIIO` inherits `GstBaseSrc`.

Definition at line 143 of file `gsttensor_srciiio.h`.

8.70.2 Member Data Documentation

8.70.2.1 `parent_class`

```
GstBaseSrcClass _GstTensorSrcIIOClass::parent_class
```

inherits class object

Definition at line 145 of file `gsttensor_srciiio.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_srciiio.h`

8.71 `_GstTensorSrcIIODeviceProperties` Struct Reference

`GstTensorSrcIIO` devices's properties (internal data structure)

```
#include <gsttensor_srciiio.h>
```

Public Attributes

- `gchar *` [name](#)
- `gchar *` [base_dir](#)
- `gint` [id](#)

8.71.1 Detailed Description

`GstTensorSrcIIO` devices's properties (internal data structure)

This data structure is used for both device/triggers, as triggers are also iio devices

Definition at line 67 of file `gsttensor_srciiio.h`.

8.71.2 Member Data Documentation

8.71.2.1 `base_dir`

```
gchar* _GstTensorSrcIIODeviceProperties::base_dir
```

The base directory for the device

Definition at line 70 of file `gsttensor_srciiio.h`.

8.71.2.2 `id`

```
gint _GstTensorSrcIIODeviceProperties::id
```

The id of the device

Definition at line 71 of file `gsttensor_srciiio.h`.

8.71.2.3 `name`

```
gchar* _GstTensorSrcIIODeviceProperties::name
```

The name of the device

Definition at line 69 of file `gsttensor_srciiio.h`.

The documentation for this struct was generated from the following file:

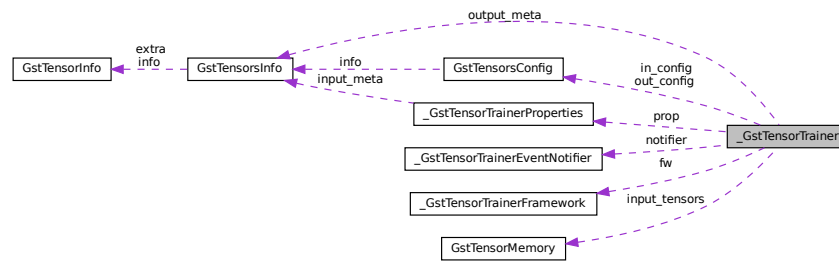
- [nnstreamer/elements/gsttensor_srciiio.h](#)

8.72 `_GstTensorTrainer` Struct Reference

`GstTensorTrainer` data structure.

```
#include <gsttensor_trainer.h>
```

Collaboration diagram for `_GstTensorTrainer`:



Public Attributes

- `GstElement` [element](#)
- `GstPad` * [sinkpad](#)
- `GstPad` * [srcpad](#)
- `gchar` * [fw_name](#)
- `gboolean` [fw_created](#)
- `gboolean` [is_training_complete](#)
- `gboolean` [is_epoch_complete](#)
- `GstTensorMemory` [input_tensors](#) [[NNS_TENSOR_SIZE_LIMIT](#)]
- `GstTensorsInfo` [output_meta](#)
- `GstTensorsConfig` [out_config](#)
- `GstTensorsConfig` [in_config](#)
- `guint` [required_sample](#)
- `guint` [cur_epoch_data_cnt](#)
- `void` * [privateData](#)
- `const` `GstTensorTrainerFramework` * [fw](#)
- `GstTensorTrainerProperties` [prop](#)
- `GstTensorTrainerEventNotifier` [notifier](#)
- `GMutex` [training_completion_lock](#)
- `GCond` [training_completion_cond](#)
- `GMutex` [epoch_completion_lock](#)
- `GCond` [epoch_completion_cond](#)
- `GThread` * [dummy_data_thread](#)

8.72.1 Detailed Description

`GstTensorTrainer` data structure.

Definition at line 43 of file `gsttensor_trainer.h`.

8.72.2 Member Data Documentation

8.72.2.1 cur_epoch_data_cnt

guint _GstTensorTrainer::cur_epoch_data_cnt

number of total push data in one eposh

Definition at line 62 of file gsttensor_trainer.h.

8.72.2.2 dummy_data_thread

GThread* _GstTensorTrainer::dummy_data_thread

Dummy data generation thread

Definition at line 74 of file gsttensor_trainer.h.

8.72.2.3 element

GstElement _GstTensorTrainer::element

parent object

Definition at line 45 of file gsttensor_trainer.h.

8.72.2.4 epoch_completion_cond

GCond _GstTensorTrainer::epoch_completion_cond

Definition at line 72 of file gsttensor_trainer.h.

8.72.2.5 epoch_completion_lock

GMutex _GstTensorTrainer::epoch_completion_lock

Definition at line 71 of file gsttensor_trainer.h.

8.72.2.6 fw

```
const GstTensorTrainerFramework\* _GstTensorTrainer::fw
```

Subplugin definition

Definition at line 65 of file gsttensor_trainer.h.

8.72.2.7 fw_created

```
gboolean _GstTensorTrainer::fw_created
```

Definition at line 52 of file gsttensor_trainer.h.

8.72.2.8 fw_name

```
gchar* _GstTensorTrainer::fw_name
```

Definition at line 50 of file gsttensor_trainer.h.

8.72.2.9 in_config

```
GstTensorsConfig _GstTensorTrainer::in_config
```

Definition at line 59 of file gsttensor_trainer.h.

8.72.2.10 input_tensors

```
GstTensorMemory _GstTensorTrainer::input_tensors[NNS_TENSOR_SIZE_LIMIT]
```

Definition at line 56 of file gsttensor_trainer.h.

8.72.2.11 is_epoch_complete

```
gboolean _GstTensorTrainer::is_epoch_complete
```

Definition at line 54 of file gsttensor_trainer.h.

8.72.2.12 is_training_complete

`gboolean _GstTensorTrainer::is_training_complete`

Definition at line 53 of file `gsttensor_trainer.h`.

8.72.2.13 notifier

`GstTensorTrainerEventNotifier _GstTensorTrainer::notifier`

Event notifier

Definition at line 67 of file `gsttensor_trainer.h`.

8.72.2.14 out_config

`GstTensorsConfig _GstTensorTrainer::out_config`

Definition at line 58 of file `gsttensor_trainer.h`.

8.72.2.15 output_meta

`GstTensorsInfo _GstTensorTrainer::output_meta`

Definition at line 57 of file `gsttensor_trainer.h`.

8.72.2.16 privateData

`void* _GstTensorTrainer::privateData`

NNFW plugin's private data is stored here

Definition at line 64 of file `gsttensor_trainer.h`.

8.72.2.17 prop

`GstTensorTrainerProperties _GstTensorTrainer::prop`

NNFW plugin's properties

Definition at line 66 of file `gsttensor_trainer.h`.

8.72.2.18 required_sample

```
guint _GstTensorTrainer::required_sample
```

Definition at line 61 of file gsttensor_trainer.h.

8.72.2.19 sinkpad

```
GstPad* _GstTensorTrainer::sinkpad
```

Definition at line 47 of file gsttensor_trainer.h.

8.72.2.20 srcpad

```
GstPad* _GstTensorTrainer::srcpad
```

Definition at line 48 of file gsttensor_trainer.h.

8.72.2.21 training_completion_cond

```
GCond _GstTensorTrainer::training_completion_cond
```

Definition at line 70 of file gsttensor_trainer.h.

8.72.2.22 training_completion_lock

```
GMutex _GstTensorTrainer::training_completion_lock
```

Definition at line 69 of file gsttensor_trainer.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_trainer.h](#)

8.73 _GstTensorTrainerClass Struct Reference

GstTensorTrainerClass data structure.

```
#include <gsttensor_trainer.h>
```

Public Attributes

- `GstElementClass` [parent_class](#)

8.73.1 Detailed Description

`GstTensorTrainerClass` data structure.

Definition at line 80 of file `gsttensor_trainer.h`.

8.73.2 Member Data Documentation

8.73.2.1 `parent_class`

```
GstElementClass _GstTensorTrainerClass::parent_class
```

parent class

Definition at line 82 of file `gsttensor_trainer.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_trainer.h`

8.74 `_GstTensorTrainerEventNotifier` Struct Reference

`GstTensorTrainer`'s event notifier.

```
#include <nnstreamer_plugin_api_trainer.h>
```

Public Attributes

- `uint64_t` [version](#)
- `void *` [notifier](#)

8.74.1 Detailed Description

`GstTensorTrainer`'s event notifier.

Subplugins must send events to `tensor_trainer` with a notifier.

Definition at line 79 of file `nnstreamer_plugin_api_trainer.h`.

8.74.2 Member Data Documentation

8.74.2.1 notifier

```
void* _GstTensorTrainerEventNotifier::notifier
```

Version of the struct | 32bit (validity check) | 16bit (API version) | 16bit (Subplugin's internal version) |

Definition at line 82 of file nnstreamer_plugin_api_trainer.h.

8.74.2.2 version

```
uint64_t _GstTensorTrainerEventNotifier::version
```

Definition at line 81 of file nnstreamer_plugin_api_trainer.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/include/nnstreamer_plugin_api_trainer.h](#)

8.75 _GstTensorTrainerFramework Struct Reference

tensor_trainer subplugin definition

```
#include <nnstreamer_plugin_api_trainer.h>
```

Public Attributes

- [uint64_t version](#)
- [int\(* create \)\(const GstTensorTrainerFramework *self, const GstTensorTrainerProperties *prop, void **private_data\)](#)
- [int\(* destroy \)\(const GstTensorTrainerFramework *self, const GstTensorTrainerProperties *prop, void **private_data\)](#)
- [int\(* start \)\(const GstTensorTrainerFramework *self, const GstTensorTrainerProperties *prop, GstTensorTrainerEventNotifier *notifier, void *private_data\)](#)
- [int\(* stop \)\(const GstTensorTrainerFramework *self, const GstTensorTrainerProperties *prop, void **private_data\)](#)
- [int\(* push_data \)\(const GstTensorTrainerFramework *self, const GstTensorTrainerProperties *prop, void *private_data, const GstTensorMemory *input\)](#)
- [int\(* getStatus \)\(const GstTensorTrainerFramework *self, GstTensorTrainerProperties *prop, void *private_data\)](#)
- [int\(* getFrameworkInfo \)\(const GstTensorTrainerFramework *self, const GstTensorTrainerProperties *prop, void *private_data, GstTensorTrainerFrameworkInfo *fw_info\)](#)

8.75.1 Detailed Description

tensor_trainer subplugin definition

Common callback parameters: prop Trainer properties. Read Only. private_data Subplugin's private data. Set this (*private_data = XXX) if you want to change trainer->private_data.

Definition at line 95 of file nnstreamer_plugin_api_trainer.h.

8.75.2 Member Data Documentation

8.75.2.1 create

```
int(* _GstTensorTrainerFramework::create) (const GstTensorTrainerFramework *self, const GstTensorTrainerProper
*prop, void **private_data)
```

tensor_trainer call this to create the model

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data, a subplugin may save its internal private data here.

Returns

0 if ok. < 0 if error.

Definition at line 102 of file nnstreamer_plugin_api_trainer.h.

8.75.2.2 destroy

```
int(* _GstTensorTrainerFramework::destroy) (const GstTensorTrainerFramework *self, const GstTensorTrainerProper
*prop, void **private_data)
```

tensor_trainer call this to destroy the model, Set NULL after that.

Parameters

in	<i>prop</i>	read-only property values
	<i>[in/out]</i>	private_data, a subplugin may save its internal private data here.

Returns

0 if ok. < 0 if error.

Definition at line 110 of file nnstreamer_plugin_api_trainer.h.

8.75.2.3 getFrameworkInfo

```
int(* _GstTensorTrainerFramework::getFrameworkInfo) (const GstTensorTrainerFramework *self,
const GstTensorTrainerProperties *prop, void *private_data, GstTensorTrainerFrameworkInfo *fw↔
_info)
```

Mandatory callback. Get the frameworks statically determined info.

Parameters

in	<i>prop</i>	read-only property values
in	<i>private_data</i>	A subplugin may save its internal private data here.
out	<i>fw_info</i>	struct to hold frameworks info. Must be allocated by the caller (return value).

Returns

0 if OK. < 0 if error.

Note

CAUTION: *private_data* can be NULL if the framework is not yet opened by the caller.

Definition at line 154 of file nnstreamer_plugin_api_trainer.h.

8.75.2.4 getStatus

```
int(* _GstTensorTrainerFramework::getStatus) (const GstTensorTrainerFramework *self, GstTensorTrainerPropertie
*prop, void *private_data)
```

tensor_trainer call this to get status of subplugin

Parameters

in	<i>prop</i>	property values
in	<i>private_data,a</i>	subplugin may save its internal private data here.

Returns

0 if ok. < 0 if error.

Definition at line 146 of file nnstreamer_plugin_api_trainer.h.

8.75.2.5 push_data

```
int(* _GstTensorTrainerFramework::push_data) (const GstTensorTrainerFramework *self, const
GstTensorTrainerProperties *prop, void *private_data, const GstTensorMemory *input)
```

tensor_trainer call this to push tensor data to subplugin, subplugin constructs a data set using input.

Parameters

in	<i>prop</i>	read-only property values
in	<i>private_data,a</i>	subplugin may save its internal private data here.
in	<i>input</i>	The array of input tensors. Allocated and filled by tensor_trainer

Returns

0 if ok. < 0 if error.

Definition at line 136 of file nnstreamer_plugin_api_trainer.h.

8.75.2.6 start

```
int(* _GstTensorTrainerFramework::start) (const GstTensorTrainerFramework *self, const GstTensorTrainerPropert
*prop, GstTensorTrainerEventNotifier *notifier, void *private_data)
```

tensor_trainer call this to start training the model

Parameters

in	<i>prop</i>	read-only property values
in	<i>notify,a</i>	handle of event notify
in	<i>private_data,a</i>	subplugin may save its internal private data here.

Returns

0 if ok. < 0 if error.

Definition at line 118 of file nnstreamer_plugin_api_trainer.h.

8.75.2.7 stop

```
int (* _GstTensorTrainerFramework::stop) (const GstTensorTrainerFramework *self, const GstTensorTrainerProperti  
*prop, void **private_data)
```

tensor_trainer call this to stop training the model

Parameters

<code>in</code>	<code>prop</code>	read-only property values
	<code>[in/out]</code>	<code>private_data</code> , a subplugin may save its internal private data here.

Returns

0 if ok. < 0 if error.

Definition at line 128 of file `nnstreamer_plugin_api_trainer.h`.

8.75.2.8 `version`

```
uint64_t _GstTensorTrainerFramework::version
```

Version of the struct | 32bit (validity check) | 16bit (API version) | 16bit (Subplugin's internal version) |

Definition at line 97 of file `nnstreamer_plugin_api_trainer.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/nnstreamer_plugin_api_trainer.h`

8.76 `_GstTensorTrainerFrameworkInfo` Struct Reference

`GstTensorTrainer`'s subplugin framework related information.

```
#include <nnstreamer_plugin_api_trainer.h>
```

Public Attributes

- `const char * name`

8.76.1 Detailed Description

`GstTensorTrainer`'s subplugin framework related information.

All the information is provided statically.

Definition at line 55 of file `nnstreamer_plugin_api_trainer.h`.

8.76.2 Member Data Documentation

8.76.2.1 name

```
const char* _GstTensorTrainerFrameworkInfo::name
```

Name of the neural network framework, searchable by FRAMEWORK property.

Definition at line 57 of file nnstreamer_plugin_api_trainer.h.

The documentation for this struct was generated from the following file:

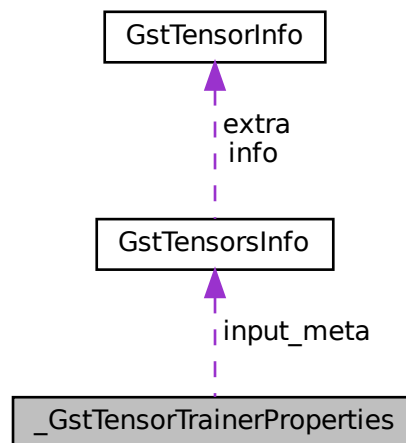
- [nnstreamer/include/nnstreamer_plugin_api_trainer.h](#)

8.77 `_GstTensorTrainerProperties` Struct Reference

`GstTensorTrainer`'s properties for neural network framework (internal data structure)

```
#include <nnstreamer_plugin_api_trainer.h>
```

Collaboration diagram for `_GstTensorTrainerProperties`:



Public Attributes

- [GstTensorsInfo](#) `input_meta`
- `const char *` [model_config](#)
- `const char *` [model_save_path](#)
- `const char *` [model_load_path](#)
- `unsigned int` [num_inputs](#)
- `unsigned int` [num_labels](#)
- `unsigned int` [num_training_samples](#)
- `unsigned int` [num_validation_samples](#)
- `unsigned int` [num_epochs](#)
- `unsigned int` [epoch_count](#)
- `double` [training_loss](#)
- `double` [training_accuracy](#)
- `double` [validation_loss](#)
- `double` [validation_accuracy](#)

8.77.1 Detailed Description

GstTensorTrainer's properties for neural network framework (internal data structure)

Internal data of GstTensorTrainer required by tensor_trainer's custom subplugin.

Definition at line 31 of file nnstreamer_plugin_api_trainer.h.

8.77.2 Member Data Documentation

8.77.2.1 epoch_count

```
unsigned int _GstTensorTrainerProperties::epoch_count
```

Number of currently completed epochs

Definition at line 43 of file nnstreamer_plugin_api_trainer.h.

8.77.2.2 input_meta

```
GstTensorsInfo _GstTensorTrainerProperties::input_meta
```

configured input tensor info

Definition at line 33 of file nnstreamer_plugin_api_trainer.h.

8.77.2.3 model_config

```
const char* _GstTensorTrainerProperties::model_config
```

The configuration file path for creating model

Definition at line 34 of file nnstreamer_plugin_api_trainer.h.

8.77.2.4 model_load_path

```
const char* _GstTensorTrainerProperties::model_load_path
```

The file path to load an existing model to use for training a new model

Definition at line 36 of file nnstreamer_plugin_api_trainer.h.

8.77.2.5 model_save_path

```
const char* _GstTensorTrainerProperties::model_save_path
```

The file path to save the new model

Definition at line 35 of file nnstreamer_plugin_api_trainer.h.

8.77.2.6 num_epochs

```
unsigned int _GstTensorTrainerProperties::num_epochs
```

The number of repetition of total training and validation sample. subplugin must receive total samples((num_training_samples + num_validation_samples) * num_epochs)

Definition at line 41 of file nnstreamer_plugin_api_trainer.h.

8.77.2.7 num_inputs

```
unsigned int _GstTensorTrainerProperties::num_inputs
```

The number of input lists, the input is where framework receive the features to train the model, num_inputs indicates how many inputs there are.

Definition at line 37 of file nnstreamer_plugin_api_trainer.h.

8.77.2.8 num_labels

```
unsigned int _GstTensorTrainerProperties::num_labels
```

The number of label lists, the label is where framework receive the class to train the model, num_labels indicates how many labels there are.

Definition at line 38 of file nnstreamer_plugin_api_trainer.h.

8.77.2.9 num_training_samples

```
unsigned int _GstTensorTrainerProperties::num_training_samples
```

The number of training sample used to train the model.

Definition at line 39 of file nnstreamer_plugin_api_trainer.h.

8.77.2.10 num_validation_samples

```
unsigned int _GstTensorTrainerProperties::num_validation_samples
```

The number of validation sample used to valid the model.

Definition at line 40 of file nnstreamer_plugin_api_trainer.h.

8.77.2.11 training_accuracy

```
double _GstTensorTrainerProperties::training_accuracy
```

Training accuracy of the model being trained in the subplugin

Definition at line 45 of file nnstreamer_plugin_api_trainer.h.

8.77.2.12 training_loss

```
double _GstTensorTrainerProperties::training_loss
```

Training loss of the model being trained in the subplugin

Definition at line 44 of file nnstreamer_plugin_api_trainer.h.

8.77.2.13 validation_accuracy

```
double _GstTensorTrainerProperties::validation_accuracy
```

Validation accuracy of the model being trained in the subplugin

Definition at line 47 of file nnstreamer_plugin_api_trainer.h.

8.77.2.14 validation_loss

```
double _GstTensorTrainerProperties::validation_loss
```

Validation loss of the model being trained in the subplugin

Definition at line 46 of file nnstreamer_plugin_api_trainer.h.

The documentation for this struct was generated from the following file:

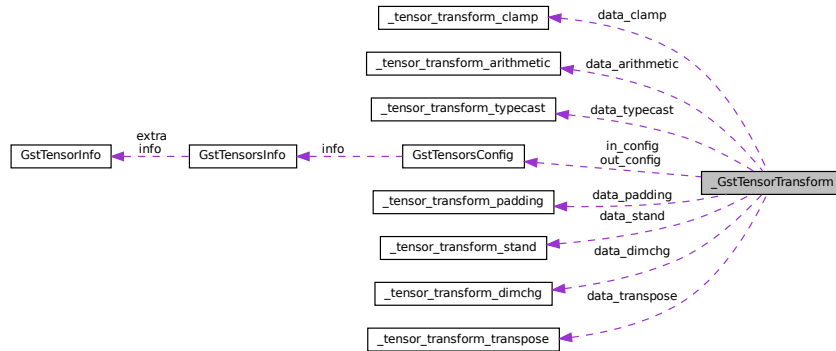
- [nnstreamer/include/nnstreamer_plugin_api_trainer.h](#)

8.78 `_GstTensorTransform` Struct Reference

Internal data structure for `tensor_transform` instances.

```
#include <gsttensor_transform.h>
```

Collaboration diagram for `_GstTensorTransform`:



Public Attributes

- `GstBaseTransform` [element](#)
- `gboolean` [silent](#)
- `tensor_transform_mode` [mode](#)
- `gchar *` [option](#)
- `union` {
 - `tensor_transform_dimchg` [data_dimchg](#)
 - `tensor_transform_typecast` [data_typecast](#)
 - `tensor_transform_arithmetic` [data_arithmetic](#)
 - `tensor_transform_transpose` [data_transpose](#)
 - `tensor_transform_stand` [data_stand](#)
 - `tensor_transform_clamp` [data_clamp](#)
 - `tensor_transform_padding` [data_padding](#)
- };
- `gboolean` [loaded](#)
- `gboolean` [acceleration](#)
- `GSList *` [operators](#)
- `GstTensorsConfig` [in_config](#)
- `GstTensorsConfig` [out_config](#)
- `GList *` [apply](#)

8.78.1 Detailed Description

Internal data structure for `tensor_transform` instances.

Definition at line 166 of file `gsttensor_transform.h`.

8.78.2 Member Data Documentation

8.78.2.1 "@37

```
union { ... }
```

8.78.2.2 acceleration

```
gboolean _GstTensorTransform::acceleration
```

TRUE to set orc acceleration

Definition at line 183 of file gsttensor_transform.h.

8.78.2.3 apply

```
GList* _GstTensorTransform::apply
```

Select the tensors to apply transformation

Definition at line 188 of file gsttensor_transform.h.

8.78.2.4 data_arithmetic

```
tensor_transform_arithmetic _GstTensorTransform::data_arithmetic
```

Parsed option value for "arithmetic" mode.

Definition at line 176 of file gsttensor_transform.h.

8.78.2.5 data_clamp

```
tensor_transform_clamp _GstTensorTransform::data_clamp
```

Parsed option value for "clamp" mode.

Definition at line 179 of file gsttensor_transform.h.

8.78.2.6 data_dimchg

`tensor_transform_dimchg` `_GstTensorTransform::data_dimchg`

Parsed option value for "dimchg" mode

Definition at line 174 of file `gsttensor_transform.h`.

8.78.2.7 data_padding

`tensor_transform_padding` `_GstTensorTransform::data_padding`

Parsed option value for "padding" mode.

Definition at line 180 of file `gsttensor_transform.h`.

8.78.2.8 data_stand

`tensor_transform_stand` `_GstTensorTransform::data_stand`

Parsed option value for "stand" mode.

Definition at line 178 of file `gsttensor_transform.h`.

8.78.2.9 data_transpose

`tensor_transform_transpose` `_GstTensorTransform::data_transpose`

Parsed option value for "transpose" mode.

Definition at line 177 of file `gsttensor_transform.h`.

8.78.2.10 data_typecast

`tensor_transform_typecast` `_GstTensorTransform::data_typecast`

Parsed option value for "typecast" mode.

Definition at line 175 of file `gsttensor_transform.h`.

8.78.2.11 `element`

`GstBaseTransform` `_GstTensorTransform::element`

This is the parent object

Definition at line 168 of file `gsttensor_transform.h`.

8.78.2.12 `in_config`

`GstTensorsConfig` `_GstTensorTransform::in_config`

input tensors config

Definition at line 186 of file `gsttensor_transform.h`.

8.78.2.13 `loaded`

`gboolean` `_GstTensorTransform::loaded`

TRUE if mode & option are loaded

Definition at line 182 of file `gsttensor_transform.h`.

8.78.2.14 `mode`

`tensor_transform_mode` `_GstTensorTransform::mode`

Transform mode. `GTT_UNKNOWN` if invalid.

Definition at line 171 of file `gsttensor_transform.h`.

8.78.2.15 `operators`

`GSList*` `_GstTensorTransform::operators`

operators list

Definition at line 184 of file `gsttensor_transform.h`.

8.78.2.16 option

```
gchar* _GstTensorTransform::option
```

Stored option value

Definition at line 172 of file gsttensor_transform.h.

8.78.2.17 out_config

```
GstTensorsConfig _GstTensorTransform::out_config
```

output tensors config

Definition at line 187 of file gsttensor_transform.h.

8.78.2.18 silent

```
gboolean _GstTensorTransform::silent
```

True if logging is minimized

Definition at line 170 of file gsttensor_transform.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.79 _GstTensorTransformClass Struct Reference

GstTensorTransformClass inherits GstBaseTransformClass.

```
#include <gsttensor_transform.h>
```

Public Attributes

- GstBaseTransformClass [parent_class](#)

8.79.1 Detailed Description

GstTensorTransformClass inherits GstBaseTransformClass.

Referring another child (sibling), GstVideoFilter (abstract class) and its child (concrete class) GstVideoTransform. Note that GstTensorTransformClass is a concrete class; thus we need to look at both.

Definition at line 198 of file gsttensor_transform.h.

8.79.2 Member Data Documentation

8.79.2.1 parent_class

```
GstBaseTransformClass _GstTensorTransformClass::parent_class
```

Inherits GstBaseTransformClass

Definition at line 200 of file gsttensor_transform.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.80 _GTensorFilterSingle Struct Reference

Internal data structure for tensor_filter_single instances.

```
#include <tensor_filter_single.h>
```

Public Attributes

- GObject [element](#)
- gpointer [priv](#)

8.80.1 Detailed Description

Internal data structure for tensor_filter_single instances.

Definition at line 51 of file tensor_filter_single.h.

8.80.2 Member Data Documentation

8.80.2.1 element

```
GObject _GTensorFilterSingle::element
```

This is the parent object

Definition at line 53 of file tensor_filter_single.h.

8.80.2.2 priv

```
gpointer _GTensorFilterSingle::priv
```

Internal properties for tensor-filter single class

Definition at line 56 of file tensor_filter_single.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_filter/tensor_filter_single.h](#)

8.81 _GTensorFilterSingleClass Struct Reference

GTensorFilterSingleClass inherits GObjectClass.

```
#include <tensor_filter_single.h>
```

Public Attributes

- GObjectClass [parent](#)
- gboolean(* [invoke](#))(GTensorFilterSingle *self, const [GstTensorMemory](#) *input, [GstTensorMemory](#) *output, gboolean allocate)
- gboolean(* [start](#))(GTensorFilterSingle *self)
- gboolean(* [stop](#))(GTensorFilterSingle *self)
- gboolean(* [input_configured](#))(GTensorFilterSingle *self)
- gboolean(* [output_configured](#))(GTensorFilterSingle *self)
- gint(* [set_input_info](#))(GTensorFilterSingle *self, const [GstTensorsInfo](#) *in_info, [GstTensorsInfo](#) *out_info)
- gboolean(* [allocate_in_invoke](#))(GTensorFilterSingle *self)
- void(* [destroy_notify](#))(GTensorFilterSingle *self, [GstTensorMemory](#) *mem)

8.81.1 Detailed Description

GTensorFilterSingleClass inherits GObjectClass.

Definition at line 62 of file tensor_filter_single.h.

8.81.2 Member Data Documentation

8.81.2.1 allocate_in_invoke

```
gboolean(* _GTensorFilterSingleClass::allocate_in_invoke) (GTensorFilterSingle *self)
```

Check if the filter performs allocate_in_invoke

Definition at line 81 of file tensor_filter_single.h.

8.81.2.2 destroy_notify

```
void(* _GTensorFilterSingleClass::destroy_notify) (GTensorFilterSingle *self, GstTensorMemory *mem)
```

Free the data allocated by the tensor filter in invoke

Definition at line 83 of file tensor_filter_single.h.

8.81.2.3 input_configured

```
gboolean(* _GTensorFilterSingleClass::input_configured) (GTensorFilterSingle *self)
```

Check if the input is already configured

Definition at line 74 of file tensor_filter_single.h.

8.81.2.4 invoke

```
gboolean(* _GTensorFilterSingleClass::invoke) (GTensorFilterSingle *self, const GstTensorMemory *input, GstTensorMemory *output, gboolean allocate)
```

Invoke the filter for execution.

Definition at line 67 of file tensor_filter_single.h.

8.81.2.5 output_configured

```
gboolean(* _GTensorFilterSingleClass::output_configured) (GTensorFilterSingle *self)
```

Check if the output is already configured

Definition at line 76 of file tensor_filter_single.h.

8.81.2.6 parent

```
GObjectClass _GTensorFilterSingleClass::parent
```

inherits GObjectClass

Definition at line 64 of file tensor_filter_single.h.

8.81.2.7 set_input_info

```
gint (*_GTensorFilterSingleClass::set_input_info) (GTensorFilterSingle *self, const GstTensorsInfo
*in_info, GstTensorsInfo *out_info)
```

Set the info about the input tensor

Definition at line 78 of file tensor_filter_single.h.

8.81.2.8 start

```
gboolean (*_GTensorFilterSingleClass::start) (GTensorFilterSingle *self)
```

Start the filter, must be called before invoke.

Definition at line 70 of file tensor_filter_single.h.

8.81.2.9 stop

```
gboolean (*_GTensorFilterSingleClass::stop) (GTensorFilterSingle *self)
```

Stop the filter.

Definition at line 72 of file tensor_filter_single.h.

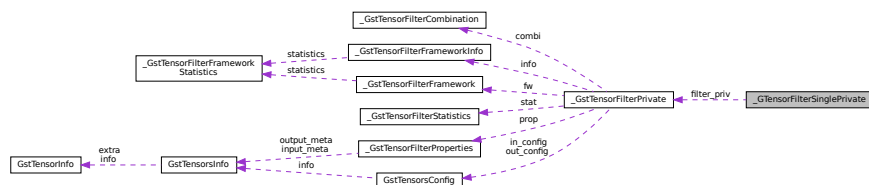
The documentation for this struct was generated from the following file:

- nnstreamer/tensor_filter/tensor_filter_single.h

8.82 _GTensorFilterSinglePrivate Struct Reference

Private data struct for tensor-filter single class.

Collaboration diagram for _GTensorFilterSinglePrivate:



Public Attributes

- [GstTensorFilterPrivate filter_priv](#)
- gboolean [allocate_in_invoke](#)

8.82.1 Detailed Description

Private data struct for tensor-filter single class.

SECTION:element-tensor_filter_single

An element that invokes neural network models and their framework or an independent shared object implementing [tensor_filter_custom.h](#). The input and output are always in the format of other/tensor or other/tensors. This element is going to be the basis of single shot api.

Definition at line 53 of file [tensor_filter_single.c](#).

8.82.2 Member Data Documentation

8.82.2.1 allocate_in_invoke

```
gboolean _GTensorFilterSinglePrivate::allocate_in_invoke
```

cached value after first invoke

Definition at line 56 of file [tensor_filter_single.c](#).

8.82.2.2 filter_priv

```
GstTensorFilterPrivate _GTensorFilterSinglePrivate::filter_priv
```

Internal properties for tensor-filter

Definition at line 55 of file [tensor_filter_single.c](#).

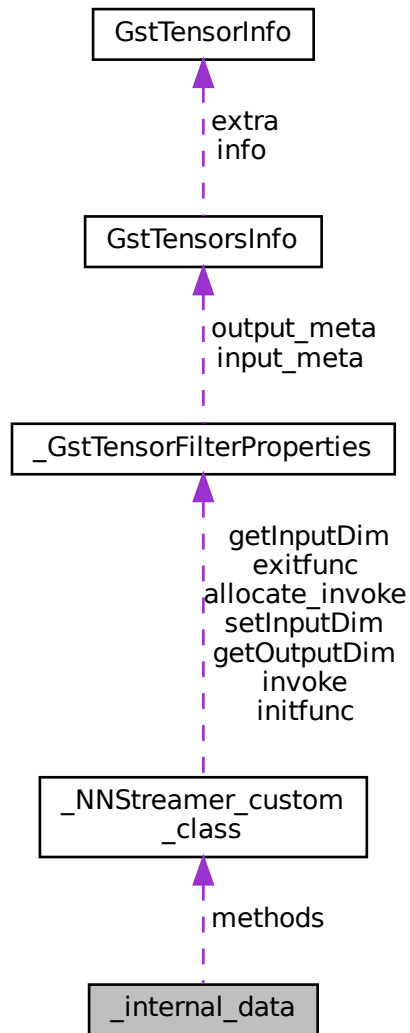
The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_filter/tensor_filter_single.c](#)

8.83 `_internal_data` Struct Reference

`internal_data`

Collaboration diagram for `_internal_data`:



Public Attributes

- `GModule *` [module](#)
- `NNStreamer_custom_class *` [methods](#)
- `void *` [customFW_private_data](#)

8.83.1 Detailed Description

`internal_data`

Definition at line 50 of file `tensor_filter_custom.c`.

8.83.2 Member Data Documentation

8.83.2.1 `customFW_private_data`

```
void* _internal_data::customFW_private_data
```

Definition at line 55 of file `tensor_filter_custom.c`.

8.83.2.2 methods

```
NNStreamer\_custom\_class* _internal_data::methods
```

Definition at line 53 of file `tensor_filter_custom.c`.

8.83.2.3 module

```
GModule* _internal_data::module
```

Definition at line 52 of file `tensor_filter_custom.c`.

The documentation for this struct was generated from the following file:

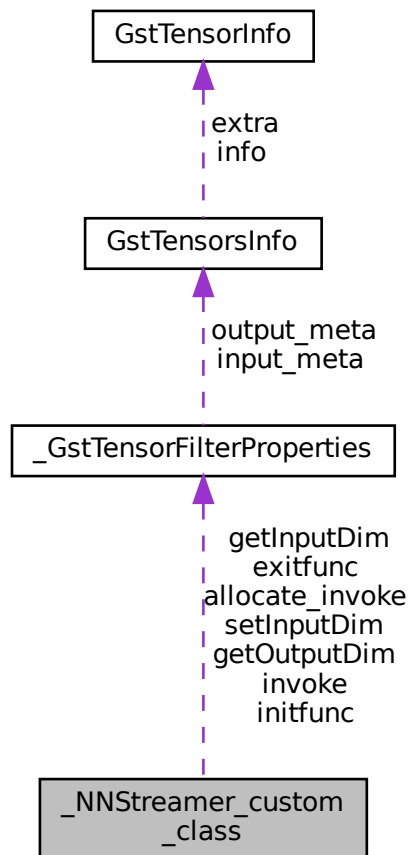
- `nnstreamer/tensor_filter/tensor_filter_custom.c`

8.84 `_NNStreamer_custom_class` Struct Reference

Custom Filter Class.

```
#include <tensor_filter_custom.h>
```

Collaboration diagram for `_NNStreamer_custom_class`:



Public Attributes

- [NNS_custom_init_func](#) `initfunc`
- [NNS_custom_exit_func](#) `exitfunc`
- [NNS_custom_get_input_dimension](#) `getInputDim`
- [NNS_custom_get_output_dimension](#) `getOutputDim`
- [NNS_custom_set_input_dimension](#) `setInputDim`
- [NNS_custom_invoke](#) `invoke`
- [NNS_custom_allocate_invoke](#) `allocate_invoke`
- [NNS_custom_destroy_notify](#) `destroy_notify`

8.84.1 Detailed Description

Custom Filter Class.

Note that every function pointer is MANDATORY!

Definition at line 125 of file `tensor_filter_custom.h`.

8.84.2 Member Data Documentation

8.84.2.1 `allocate_invoke`

`NNS_custom_allocate_invoke` `_NNStreamer_custom_class::allocate_invoke`

the main function, "allocate & invoke", that transforms input to output. `allocate_invoke` is supposed to allocate output buffer by itself. `(invoke) XOR (allocate_invoke) MUST` hold.

Definition at line 133 of file `tensor_filter_custom.h`.

8.84.2.2 `destroy_notify`

`NNS_custom_destroy_notify` `_NNStreamer_custom_class::destroy_notify`

it handles the data pointer allocated in the custom framework. when the data pointer has been destroyed at the pipeline, this method will be called. the data pointer or an object including data pointer could be deleted safely with this function. this method is only used when `allocate_invoke` is `TRUE`

Definition at line 134 of file `tensor_filter_custom.h`.

8.84.2.3 `exitfunc`

`NNS_custom_exit_func` `_NNStreamer_custom_class::exitfunc`

will not call other callbacks after this call

Definition at line 128 of file `tensor_filter_custom.h`.

8.84.2.4 `getInputDim`

`NNS_custom_get_input_dimension` `_NNStreamer_custom_class::getInputDim`

a custom filter is required to provide input tensor dimension unless `setInputdim` is defined.

Definition at line 129 of file `tensor_filter_custom.h`.

8.84.2.5 `getOutputDim`

`NNS_custom_get_output_dimension` `_NNStreamer_custom_class::getOutputDim`

a custom filter is required to provide output tensor dimension unless `setInputDim` is defined.

Definition at line 130 of file `tensor_filter_custom.h`.

8.84.2.6 `initfunc`

`NNS_custom_init_func` `_NNStreamer_custom_class::initfunc`

called before any other callbacks from `tensor_filter_custom.c`

Definition at line 127 of file `tensor_filter_custom.h`.

8.84.2.7 `invoke`

`NNS_custom_invoke` `_NNStreamer_custom_class::invoke`

the main function, "invoke", that transforms input to output. `invoke` is supposed to fill in the given output buffer. (`invoke`) XOR (`allocate_invoke`) MUST hold.

Definition at line 132 of file `tensor_filter_custom.h`.

8.84.2.8 `setInputDim`

`NNS_custom_set_input_dimension` `_NNStreamer_custom_class::setInputDim`

without `getI/O-Dim`, this allows framework to set input dimension and get output dimension from the custom filter according to the input dimension

Definition at line 131 of file `tensor_filter_custom.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/tensor_filter_custom.h`

8.85 `_NNStreamerExternalConverter` Struct Reference

Converter's subplugin implementation.

```
#include <nnstreamer_plugin_api_converter.h>
```

Public Attributes

- const char * [name](#)
- GstBuffer *(* [convert](#))(GstBuffer *in_buf, [GstTensorsConfig](#) *config, void *priv_data)
- gboolean(* [get_out_config](#))(const GstCaps *in_caps, [GstTensorsConfig](#) *config)
- GstCaps *(* [query_caps](#))(const [GstTensorsConfig](#) *config)
- int(* [open](#))(const gchar *script_path, void **priv_data)
- void(* [close](#))(void **priv_data)

8.85.1 Detailed Description

Converter's subplugin implementation.

Definition at line 41 of file nnstreamer_plugin_api_converter.h.

8.85.2 Member Data Documentation

8.85.2.1 close

```
void(* _NNStreamerExternalConverter::close) (void **priv_data)
```

tensor_converter will call this to close subplugin.

Parameters

in	<i>private_data</i>	freets private_data and set NULL.
----	---------------------	-----------------------------------

Definition at line 82 of file nnstreamer_plugin_api_converter.h.

8.85.2.2 convert

```
GstBuffer*(* _NNStreamerExternalConverter::convert) (GstBuffer *in_buf, GstTensorsConfig *config,  
void *priv_data)
```

Convert the given input stream to tensor/tensors stream.

Parameters

in	<i>buf</i>	The input stream buffer
out	<i>config</i>	tensors config structure to be filled

Return values

<i>Return</i>	input buffer(<i>in_buf</i>) if the data is to be kept untouched.
<i>Return</i>	a new <code>GstBuf</code> if the data is to be modified.

Definition at line 48 of file `nnstreamer_plugin_api_converter.h`.

8.85.2.3 `get_out_config`

```
gboolean(* _NNStreamerExternalConverter::get_out_config) (const GstCaps *in_caps, GstTensorsConfig *config)
```

Set the tensor config structure from the given stream frame.

Parameters

<i>in</i>	<i>in_caps</i>	The input (original/media data) stream's metadata
<i>out</i>	<i>config</i>	The output (tensor/tensors) metadata

Return values

<i>Return</i>	True if get caps successfully, FALSE if not.
---------------	--

Definition at line 58 of file `nnstreamer_plugin_api_converter.h`.

8.85.2.4 `name`

```
const char* _NNStreamerExternalConverter::name
```

Definition at line 45 of file `nnstreamer_plugin_api_converter.h`.

8.85.2.5 `open`

```
int(* _NNStreamerExternalConverter::open) (const gchar *script_path, void **priv_data)
```

`tensor_converter` will call this to open subplugin.

Parameters

<i>in</i>	<i>script_path</i>	script path of the subplugin.
	<i>[in/out]</i>	<i>private_data</i> A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer. Normally, <code>open()</code> allocates memory for <i>private_data</i> .

Returns

0 if ok. < 0 if error.

Definition at line 75 of file `nnstreamer_plugin_api_converter.h`.

8.85.2.6 `query_caps`

```
GstCaps* (* _NNStreamerExternalConverter::query_caps) (const GstTensorConfig *config)
```

Filters (narrows down) the `GstCap` (st) with the given config.

Parameters

<code>in</code>	<code>config</code>	The config of output tensor/tensors
-----------------	---------------------	-------------------------------------

Return values

<i>Return</i>	subplugin caps (if config is NULL, return default caps)
---------------	---

Definition at line 68 of file `nnstreamer_plugin_api_converter.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/nnstreamer_plugin_api_converter.h`

8.86 `_NnstWatchdog` Struct Reference

Structure for `NNStreamer` watchdog.

Public Attributes

- `GMainContext` * `context`
- `GMainLoop` * `loop`
- `GThread` * `thread`
- `GSource` * `source`
- `GMutex` `lock`
- `GCond` `cond`

8.86.1 Detailed Description

Structure for `NNStreamer` watchdog.

Definition at line 21 of file `nnstreamer_watchdog.c`.

8.86.2 Member Data Documentation

8.86.2.1 cond

GCond _NnstWatchdog::cond

Definition at line 28 of file nnstreamer_watchdog.c.

8.86.2.2 context

GMainContext* _NnstWatchdog::context

Definition at line 23 of file nnstreamer_watchdog.c.

8.86.2.3 lock

GMutex _NnstWatchdog::lock

Definition at line 27 of file nnstreamer_watchdog.c.

8.86.2.4 loop

GMainLoop* _NnstWatchdog::loop

Definition at line 24 of file nnstreamer_watchdog.c.

8.86.2.5 source

GSource* _NnstWatchdog::source

Definition at line 26 of file nnstreamer_watchdog.c.

8.86.2.6 thread

```
GThread* _NnstWatchdog::thread
```

Definition at line 25 of file nnstreamer_watchdog.c.

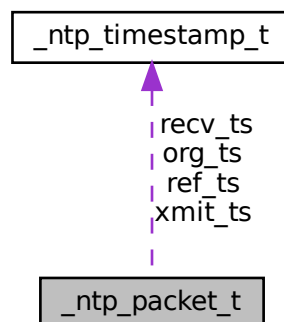
The documentation for this struct was generated from the following file:

- [nnstreamer/nnstreamer_watchdog.c](#)

8.87 _ntp_packet_t Struct Reference

A custom data type to represent NTP packet header format.

Collaboration diagram for _ntp_packet_t:



Public Attributes

- [uint8_t li_vn_mode](#)
- [uint8_t stratum](#)
- [uint8_t poll](#)
- [uint8_t precision](#)
- [uint32_t root_delay](#)
- [uint32_t root_dispersion](#)
- [uint32_t ref_id](#)
- [ntp_timestamp_t ref_ts](#)
- [ntp_timestamp_t org_ts](#)
- [ntp_timestamp_t recv_ts](#)
- [ntp_timestamp_t xmit_ts](#)

8.87.2.4 precision

`uint8_t _ntp_packet_t::precision`

Definition at line 105 of file ntputil.c.

8.87.2.5 recv_ts

`ntp_timestamp_t _ntp_packet_t::recv_ts`

Definition at line 111 of file ntputil.c.

8.87.2.6 ref_id

`uint32_t _ntp_packet_t::ref_id`

Definition at line 108 of file ntputil.c.

8.87.2.7 ref_ts

`ntp_timestamp_t _ntp_packet_t::ref_ts`

Definition at line 109 of file ntputil.c.

8.87.2.8 root_delay

`uint32_t _ntp_packet_t::root_delay`

Definition at line 106 of file ntputil.c.

8.87.2.9 root_dispersion

`uint32_t _ntp_packet_t::root_dispersion`

Definition at line 107 of file ntputil.c.

8.87.2.10 stratum

`uint8_t _ntp_packet_t::stratum`

Definition at line 103 of file ntputil.c.

8.87.2.11 xmit_ts

`ntp_timestamp_t _ntp_packet_t::xmit_ts`

Definition at line 112 of file ntputil.c.

The documentation for this struct was generated from the following file:

- [mqtt/ntputil.c](#)

8.88 _ntp_timestamp_t Struct Reference

A custom data type to represent NTP timestamp format.

Public Attributes

- `uint32_t sec`
- `uint32_t frac`

8.88.1 Detailed Description

A custom data type to represent NTP timestamp format.

NTP Timestamp Format (<https://www.ietf.org/rfc/rfc5905.txt> p.12) 0 1 2 3 0 1 2 3 4 5 6 7 8
 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 ++++++ | Seconds | ++++++ | Fraction | ++++++
 ++++++ |

Definition at line 40 of file ntputil.c.

8.88.2 Member Data Documentation

8.88.2.1 frac

`uint32_t _ntp_timestamp_t::frac`

Definition at line 43 of file ntputil.c.

8.88.2.2 `sec`

```
uint32_t _ntp_timestamp_t::sec
```

Definition at line 42 of file `ntputil.c`.

The documentation for this struct was generated from the following file:

- [mqtt/ntputil.c](#)

8.89 `_tensor_merge_linear` Struct Reference

Internal data structure for linear mode.

```
#include <gsttensor_merge.h>
```

Public Attributes

- [`tensor_merge_linear_mode` direction](#)

8.89.1 Detailed Description

Internal data structure for linear mode.

Definition at line 64 of file `gsttensor_merge.h`.

8.89.2 Member Data Documentation

8.89.2.1 `direction`

```
tensor_merge_linear_mode _tensor_merge_linear::direction
```

Definition at line 65 of file `gsttensor_merge.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_merge.h](#)

8.90 `_tensor_sync_basepad_data` Struct Reference

Tensor Merge/Mux sync data for baspad mode.

```
#include <tensor_common.h>
```

Public Attributes

- quint [sink_id](#)
- GstClockTime [duration](#)

8.90.1 Detailed Description

Tensor Merge/Mux sync data for baspad mode.

Definition at line 74 of file `tensor_common.h`.

8.90.2 Member Data Documentation

8.90.2.1 duration

```
GstClockTime _tensor_sync_basepad_data::duration
```

Definition at line 76 of file `tensor_common.h`.

8.90.2.2 sink_id

```
quint _tensor_sync_basepad_data::sink_id
```

Definition at line 75 of file `tensor_common.h`.

The documentation for this struct was generated from the following file:

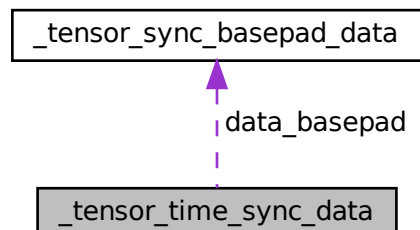
- `nnstreamer/tensor_common.h`

8.91 `_tensor_time_sync_data` Struct Reference

Tensor Merge/Mux time sync data.

```
#include <tensor_common.h>
```

Collaboration diagram for `_tensor_time_sync_data`:



Public Attributes

- [tensor_time_sync_mode](#) mode
 - `gchar * option`
 - union {
 - [tensor_sync_basepad_data](#) data_basepad
- };

8.91.1 Detailed Description

Tensor Merge/Mux time sync data.

Definition at line 82 of file `tensor_common.h`.

8.91.2 Member Data Documentation

8.91.2.1 "@61

```
union { ... }
```

8.91.2.2 `data_basepad`

[tensor_sync_basepad_data](#) `_tensor_time_sync_data::data_basepad`

Definition at line 86 of file `tensor_common.h`.

8.91.2.3 `mode`

[tensor_time_sync_mode](#) `_tensor_time_sync_data::mode`

Definition at line 83 of file `tensor_common.h`.

8.91.2.4 `option`

`gchar* _tensor_time_sync_data::option`

Definition at line 84 of file `tensor_common.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/tensor_common.h`

8.92 `_tensor_transform_arithmetic` Struct Reference

Internal data structure for arithmetic mode.

```
#include <gsttensor_transform.h>
```

Public Attributes

- [tensor_type out_type](#)
- gboolean [per_channel_arith](#)
- guint [ch_dim](#)

8.92.1 Detailed Description

Internal data structure for arithmetic mode.

Definition at line 126 of file `gsttensor_transform.h`.

8.92.2 Member Data Documentation

8.92.2.1 `ch_dim`

```
guint _tensor_transform_arithmetic::ch_dim
```

Definition at line 129 of file `gsttensor_transform.h`.

8.92.2.2 `out_type`

```
tensor_type _tensor_transform_arithmetic::out_type
```

Definition at line 127 of file `gsttensor_transform.h`.

8.92.2.3 `per_channel_arith`

```
gboolean _tensor_transform_arithmetic::per_channel_arith
```

Definition at line 128 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.93 `_tensor_transform_clamp` Struct Reference

Internal data structure for clamp mode.

```
#include <gsttensor_transform.h>
```

Public Attributes

- double `min`
- double `max`

8.93.1 Detailed Description

Internal data structure for clamp mode.

Definition at line 151 of file `gsttensor_transform.h`.

8.93.2 Member Data Documentation

8.93.2.1 `max`

```
double _tensor_transform_clamp::max
```

Definition at line 152 of file `gsttensor_transform.h`.

8.93.2.2 `min`

```
double _tensor_transform_clamp::min
```

Definition at line 152 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.94 `_tensor_transform_dimchg` Struct Reference

Internal data structure for dimchg mode.

```
#include <gsttensor_transform.h>
```

Public Attributes

- int [from](#)
- int [to](#)

8.94.1 Detailed Description

Internal data structure for dimchg mode.

Definition at line 101 of file `gsttensor_transform.h`.

8.94.2 Member Data Documentation

8.94.2.1 from

```
int _tensor_transform_dimchg::from
```

Definition at line 102 of file `gsttensor_transform.h`.

8.94.2.2 to

```
int _tensor_transform_dimchg::to
```

Definition at line 103 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.95 _tensor_transform_padding Struct Reference

Internal data structure for padding mode.

```
#include <gsttensor_transform.h>
```

Public Attributes

- guint [pad](#) [NNS_TENSOR_RANK_LIMIT]
- [tensor_layout](#) layout

8.95.1 Detailed Description

Internal data structure for padding mode.

Definition at line 158 of file `gsttensor_transform.h`.

8.95.2 Member Data Documentation

8.95.2.1 `layout`

`tensor_layout` `_tensor_transform_padding::layout`

Definition at line 160 of file `gsttensor_transform.h`.

8.95.2.2 `pad`

`guint` `_tensor_transform_padding::pad`[`NNS_TENSOR_RANK_LIMIT`]

Definition at line 159 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_transform.h`

8.96 `_tensor_transform_stand` Struct Reference

Internal data structure for stand mode.

```
#include <gsttensor_transform.h>
```

Public Attributes

- `tensor_transform_stand_mode` `mode`
- `tensor_type` `out_type`
- `gboolean` `per_channel`

8.96.1 Detailed Description

Internal data structure for stand mode.

Definition at line 142 of file `gsttensor_transform.h`.

8.96.2 Member Data Documentation

8.96.2.1 mode

`tensor_transform_stand_mode` `_tensor_transform_stand::mode`

Definition at line 143 of file `gsttensor_transform.h`.

8.96.2.2 out_type

`tensor_type` `_tensor_transform_stand::out_type`

Definition at line 144 of file `gsttensor_transform.h`.

8.96.2.3 per_channel

`gboolean` `_tensor_transform_stand::per_channel`

Definition at line 145 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.97 `_tensor_transform_transpose` Struct Reference

Internal data structure for transpose mode.

```
#include <gsttensor_transform.h>
```

Public Attributes

- `uint8_t` `trans_order` [`NNS_TENSOR_RANK_LIMIT`]

8.97.1 Detailed Description

Internal data structure for transpose mode.

Definition at line 135 of file `gsttensor_transform.h`.

8.97.2 Member Data Documentation

8.97.2.1 `trans_order`

```
uint8_t _tensor_transform_transpose::trans_order[NNS_TENSOR_RANK_LIMIT]
```

Definition at line 136 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.98 `_tensor_transform_typecast` Struct Reference

Internal data structure for typecast mode.

```
#include <gsttensor_transform.h>
```

Public Attributes

- [`tensor_type`](#) to

8.98.1 Detailed Description

Internal data structure for typecast mode.

Definition at line 109 of file `gsttensor_transform.h`.

8.98.2 Member Data Documentation

8.98.2.1 `to`

```
tensor\_type _tensor_transform_typecast::to
```

`tensor_type` after cast. `_NNS_END` if unknown

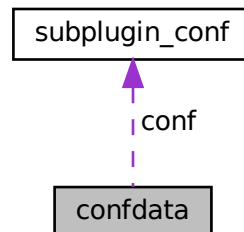
Definition at line 110 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.99 confdata Struct Reference

Collaboration diagram for confdata:



Public Attributes

- gboolean [loaded](#)
- gboolean [enable_envvar](#)
- gboolean [enable_symlink](#)
- gchar * [conffile](#)
- gchar * [extra_conffile](#)
- [subplugin_conf](#) `conf` [`NNSCONF_PATH_END`]

8.99.1 Detailed Description

Definition at line 97 of file `nnstreamer_conf.c`.

8.99.2 Member Data Documentation

8.99.2.1 `conf`

`subplugin_conf` `confdata::conf` [`NNSCONF_PATH_END`]

Definition at line 106 of file `nnstreamer_conf.c`.

8.99.2.2 conffile

```
gchar* confdata::conffile
```

Location of conf file.

Definition at line 103 of file nnstreamer_conf.c.

8.99.2.3 enable_envvar

```
gboolean confdata::enable_envvar
```

TRUE to parse env variables

Definition at line 100 of file nnstreamer_conf.c.

8.99.2.4 enable_symlink

```
gboolean confdata::enable_symlink
```

TRUE to allow symbolic link file

Definition at line 101 of file nnstreamer_conf.c.

8.99.2.5 extra_conffile

```
gchar* confdata::extra_conffile
```

Location of extra configuration file.

Definition at line 104 of file nnstreamer_conf.c.

8.99.2.6 loaded

```
gboolean confdata::loaded
```

TRUE if loaded at least once

Definition at line 99 of file nnstreamer_conf.c.

The documentation for this struct was generated from the following file:

- [nnstreamer/nnstreamer_conf.c](#)

8.100 converter_custom_cb_s Struct Reference

```
#include <gsttensor_converter.h>
```

Public Attributes

- [tensor_converter_custom func](#)
- void * [data](#)

8.100.1 Detailed Description

Definition at line 57 of file `gsttensor_converter.h`.

8.100.2 Member Data Documentation

8.100.2.1 data

```
void* converter_custom_cb_s::data
```

Definition at line 60 of file `gsttensor_converter.h`.

8.100.2.2 func

```
tensor_converter_custom converter_custom_cb_s::func
```

Definition at line 59 of file `gsttensor_converter.h`.

The documentation for this struct was generated from the following file:

- [nntstreamer/elements/gsttensor_converter.h](#)

8.101 custom_cb_s Struct Reference

```
#include <gsttensor_if.h>
```

Public Attributes

- gchar * [name](#)
- [tensor_if_custom func](#)
- void * [data](#)

8.101.1 Detailed Description

Definition at line 112 of file gsttensor_if.h.

8.101.2 Member Data Documentation

8.101.2.1 data

```
void* custom_cb_s::data
```

Definition at line 116 of file gsttensor_if.h.

8.101.2.2 func

```
tensor_if_custom custom_cb_s::func
```

Definition at line 115 of file gsttensor_if.h.

8.101.2.3 name

```
gchar* custom_cb_s::name
```

Definition at line 114 of file gsttensor_if.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_if.h](#)

8.102 decoder_custom_cb_s Struct Reference

```
#include <gsttensor_decoder.h>
```

Public Attributes

- [tensor_decoder_custom func](#)
- void * [data](#)

8.102.1 Detailed Description

Definition at line 56 of file gsttensor_decoder.h.

8.102.2 Member Data Documentation

8.102.2.1 data

```
void* decoder_custom_cb_s::data
```

Definition at line 59 of file gsttensor_decoder.h.

8.102.2.2 func

```
tensor\_decoder\_custom decoder_custom_cb_s::func
```

Definition at line 58 of file gsttensor_decoder.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_decoder.h](#)

8.103 dump_buf Struct Reference

Public Attributes

- gchar * [base](#)
- gulong [size](#)
- gulong [pos](#)

8.103.1 Detailed Description

Definition at line 670 of file nnstreamer_conf.c.

8.103.2 Member Data Documentation

8.103.2.1 `base`

```
gchar* dump_buf::base
```

Definition at line 672 of file `nnstreamer_conf.c`.

8.103.2.2 `pos`

```
gulong dump_buf::pos
```

Definition at line 674 of file `nnstreamer_conf.c`.

8.103.2.3 `size`

```
gulong dump_buf::size
```

Definition at line 673 of file `nnstreamer_conf.c`.

The documentation for this struct was generated from the following file:

- [nnstreamer/nnstreamer_conf.c](#)

8.104 `gst_tensor_aggregation_data_s` Struct Reference

Internal struct to handle aggregation data in hash table.

Public Attributes

- `GstAdapter` * [adapter](#)

8.104.1 Detailed Description

Internal struct to handle aggregation data in hash table.

Definition at line 657 of file `nnstreamer_plugin_api_impl.c`.

8.104.2 Member Data Documentation

8.104.2.1 adapter

```
GstAdapter* gst_tensor_aggregation_data_s::adapter
```

Definition at line 659 of file `nnstreamer_plugin_api_impl.c`.

The documentation for this struct was generated from the following file:

- [nnstreamer/nnstreamer_plugin_api_impl.c](#)

8.105 GstMetaQuery Struct Reference

[GstMetaQuery](#) meta structure.

```
#include <tensor_meta.h>
```

Public Attributes

- GstMeta [meta](#)
- [query_client_id_t](#) [client_id](#)

8.105.1 Detailed Description

[GstMetaQuery](#) meta structure.

Definition at line 26 of file `tensor_meta.h`.

8.105.2 Member Data Documentation

8.105.2.1 client_id

```
query\_client\_id\_t GstMetaQuery::client_id
```

Definition at line 30 of file `tensor_meta.h`.

8.105.2.2 meta

```
GstMeta GstMetaQuery::meta
```

Definition at line 28 of file `tensor_meta.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_meta.h](#)

8.106 GstSparseTensorInfo Struct Reference

Internal data structure for sparse tensor info.

```
#include <tensor_typedef.h>
```

Public Attributes

- uint32_t [nnz](#)

8.106.1 Detailed Description

Internal data structure for sparse tensor info.

Definition at line 294 of file `tensor_typedef.h`.

8.106.2 Member Data Documentation

8.106.2.1 nnz

```
uint32_t GstSparseTensorInfo::nnz
```

the number of "non-zero" elements

Definition at line 296 of file `tensor_typedef.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/tensor_typedef.h`

8.107 GstTensorAllocator Struct Reference

struct for type [GstTensorAllocator](#)

Public Attributes

- GstAllocator [parent](#)

8.107.1 Detailed Description

struct for type [GstTensorAllocator](#)

Definition at line 24 of file `tensor_allocator.c`.

8.107.2 Member Data Documentation

8.107.2.1 parent

`GstAllocator GstTensorAllocator::parent`

Definition at line 26 of file `tensor_allocator.c`.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_allocator.c](#)

8.108 GstTensorAllocatorClass Struct Reference

struct for class [GstTensorAllocatorClass](#)

Public Attributes

- `GstAllocatorClass` [parent_class](#)

8.108.1 Detailed Description

struct for class [GstTensorAllocatorClass](#)

Definition at line 32 of file `tensor_allocator.c`.

8.108.2 Member Data Documentation

8.108.2.1 parent_class

`GstAllocatorClass GstTensorAllocatorClass::parent_class`

Definition at line 34 of file `tensor_allocator.c`.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_allocator.c](#)

8.109 GstTensorCollectPadData Struct Reference

Internal data structure for Collect Pad in mux / merge.

```
#include <tensor_common.h>
```

Public Attributes

- GstCollectData [collect](#)
- GstBuffer * [buffer](#)

8.109.1 Detailed Description

Internal data structure for Collect Pad in mux / merge.

Definition at line 93 of file `tensor_common.h`.

8.109.2 Member Data Documentation

8.109.2.1 `buffer`

```
GstBuffer* GstTensorCollectPadData::buffer
```

Definition at line 96 of file `tensor_common.h`.

8.109.2.2 `collect`

```
GstCollectData GstTensorCollectPadData::collect
```

Definition at line 95 of file `tensor_common.h`.

The documentation for this struct was generated from the following file:

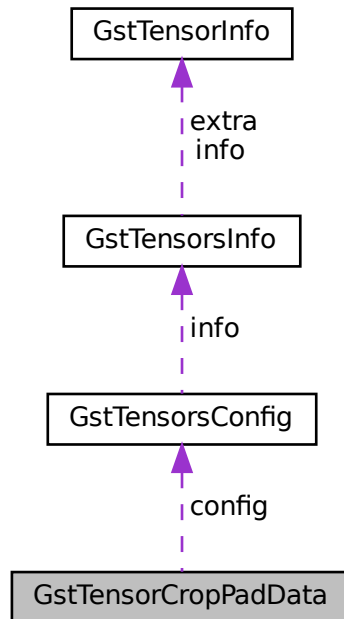
- `nnstreamer/tensor_common.h`

8.110 GstTensorCropPadData Struct Reference

GstTensorCrop pad data.

```
#include <gsttensor_crop.h>
```

Collaboration diagram for GstTensorCropPadData:



Public Attributes

- `GstCollectData` [data](#)
- `GstTensorsConfig` [config](#)

8.110.1 Detailed Description

GstTensorCrop pad data.

Definition at line 39 of file `gsttensor_crop.h`.

8.110.2 Member Data Documentation

8.110.2.1 config

`GstTensorsConfig` `GstTensorCropPadData::config`

Definition at line 43 of file `gsttensor_crop.h`.

8.110.2.2 data

`GstCollectData` `GstTensorCropPadData::data`

Definition at line 41 of file `gsttensor_crop.h`.

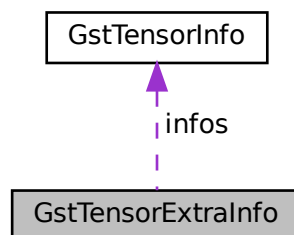
The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_crop.h`

8.111 GstTensorExtraInfo Struct Reference

Data structure to describe a "extra" tensor data. This represents the information of the `NNS_TENSOR_SIZE_LIMIT`-th memory block for tensor stream.

Collaboration diagram for `GstTensorExtraInfo`:



Public Attributes

- `uint32_t` `magic`
- `uint32_t` `version`
- `uint32_t` `num_extra_tensors`
- `uint64_t` `reserved`
- `GstTensorInfo` `infos` [`NNS_TENSOR_SIZE_EXTRA_LIMIT`]

8.111.1 Detailed Description

Data structure to describe a "extra" tensor data. This represents the information of the NNS_TENSOR_SIZE_LI←MIT-th memory block for tensor stream.

Definition at line 39 of file nnstreamer_plugin_api_impl.c.

8.111.2 Member Data Documentation

8.111.2.1 infos

`GstTensorInfo` `GstTensorExtraInfo::infos` [`NNS_TENSOR_SIZE_EXTRA_LIMIT`]

Definition at line 45 of file nnstreamer_plugin_api_impl.c.

8.111.2.2 magic

`uint32_t` `GstTensorExtraInfo::magic`

Definition at line 41 of file nnstreamer_plugin_api_impl.c.

8.111.2.3 num_extra_tensors

`uint32_t` `GstTensorExtraInfo::num_extra_tensors`

Definition at line 43 of file nnstreamer_plugin_api_impl.c.

8.111.2.4 reserved

`uint64_t` `GstTensorExtraInfo::reserved`

Definition at line 44 of file nnstreamer_plugin_api_impl.c.

8.111.2.5 version

```
uint32_t GstTensorExtraInfo::version
```

Definition at line 42 of file `nnstreamer_plugin_api_impl.c`.

The documentation for this struct was generated from the following file:

- [nnstreamer/nnstreamer_plugin_api_impl.c](#)

8.112 GstTensorFilterSharedModelRepresentation Struct Reference

Data Structure to store shared table.

```
#include <tensor_filter_common.h>
```

Public Attributes

- void * [shared_interpreter](#)
- GList * [referred_list](#)

8.112.1 Detailed Description

Data Structure to store shared table.

Definition at line 144 of file `tensor_filter_common.h`.

8.112.2 Member Data Documentation

8.112.2.1 referred_list

```
GList* GstTensorFilterSharedModelRepresentation::referred_list
```

the referred list about the instances sharing the same key

Definition at line 146 of file `tensor_filter_common.h`.

8.112.2.2 `shared_interpreter`

```
void* GstTensorFilterSharedModelRepresentation::shared_interpreter
```

the model representation for each sub-plugins

Definition at line 145 of file `tensor_filter_common.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/tensor_filter/tensor_filter_common.h`

8.113 `GstTensorInfo` Struct Reference

Internal data structure for tensor info.

```
#include <tensor_typedef.h>
```

Public Attributes

- `char * name`
- `tensor_type` type
- `tensor_dim` dimension

8.113.1 Detailed Description

Internal data structure for tensor info.

Definition at line 261 of file `tensor_typedef.h`.

8.113.2 Member Data Documentation

8.113.2.1 `dimension`

```
tensor_dim GstTensorInfo::dimension
```

Dimension. We support up to 16th ranks.

Definition at line 267 of file `tensor_typedef.h`.

8.113.2.2 name

```
char* GstTensorInfo::name
```

Name of each element in the tensor. User must designate this in a few NFW frameworks (tensorflow) and some (tensorflow-lite) do not need this.

Definition at line 263 of file tensor_typedef.h.

8.113.2.3 type

```
tensor_type GstTensorInfo::type
```

Type of each element in the tensor. User must designate this.

Definition at line 266 of file tensor_typedef.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/include/tensor_typedef.h](#)

8.114 GstTensorMemory Struct Reference

The unit of each data tensors. It will be used as an input/output tensor of other/tensors.

```
#include <tensor_typedef.h>
```

Public Attributes

- void * [data](#)
- size_t [size](#)

8.114.1 Detailed Description

The unit of each data tensors. It will be used as an input/output tensor of other/tensors.

Definition at line 252 of file tensor_typedef.h.

8.114.2 Member Data Documentation

8.114.2.1 data

```
void* GstTensorMemory::data
```

The instance of tensor data.

Definition at line 254 of file `tensor_typedef.h`.

8.114.2.2 size

```
size_t GstTensorMemory::size
```

The size of tensor.

Definition at line 255 of file `tensor_typedef.h`.

The documentation for this struct was generated from the following file:

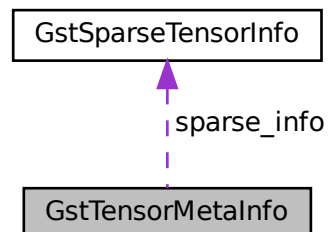
- `nntstreamer/include/tensor_typedef.h`

8.115 GstTensorMetaInfo Struct Reference

Data structure to describe a tensor data. This represents the basic information of a memory block for tensor stream.

```
#include <tensor_typedef.h>
```

Collaboration diagram for `GstTensorMetaInfo`:



Public Attributes

- uint32_t [magic](#)
- uint32_t [version](#)
- uint32_t [type](#)
- [tensor_dim](#) dimension
- uint32_t [format](#)
- uint32_t [media_type](#)
- union {
 - [GstSparseTensorInfo](#) [sparse_info](#)
- };

Union of the required information for processing each tensor "format".

8.115.1 Detailed Description

Data structure to describe a tensor data. This represents the basic information of a memory block for tensor stream.

Internally NNStreamer handles a buffer with capability other/tensors-flexible using this information.

- version: The version of tensor meta.
- type: The type of each element in the tensor. This should be a value of enumeration [tensor_type](#).
- dimension: The dimension of tensor. This also denotes the rank of tensor. (e.g., [3:224:224:0] means rank 3.)
- format: The data format in the tensor. This should be a value of enumeration [tensor_format](#).
- media_type: The media type of tensor. This should be a value of enumeration [media_type](#).

Definition at line 310 of file [tensor_typedef.h](#).

8.115.2 Member Data Documentation

8.115.2.1 "@59

```
union { ... }
```

Union of the required information for processing each tensor "format".

8.115.2.2 dimension

```
tensor\_dim GstTensorMetaInfo::dimension
```

Definition at line 315 of file [tensor_typedef.h](#).

8.115.2.3 format

`uint32_t GstTensorMetaInfo::format`

Definition at line 316 of file `tensor_typedef.h`.

8.115.2.4 magic

`uint32_t GstTensorMetaInfo::magic`

Definition at line 312 of file `tensor_typedef.h`.

8.115.2.5 media_type

`uint32_t GstTensorMetaInfo::media_type`

Definition at line 317 of file `tensor_typedef.h`.

8.115.2.6 sparse_info

`GstSparseTensorInfo GstTensorMetaInfo::sparse_info`

Definition at line 323 of file `tensor_typedef.h`.

8.115.2.7 type

`uint32_t GstTensorMetaInfo::type`

Definition at line 314 of file `tensor_typedef.h`.

8.115.2.8 version

`uint32_t GstTensorMetaInfo::version`

Definition at line 313 of file `tensor_typedef.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/tensor_typedef.h`

8.116 GstTensorPad Struct Reference

Internal data structure for pad in demux / split.

```
#include <tensor_common.h>
```

Public Attributes

- GstPad * [pad](#)
- GstClockTime [last_ts](#)
- GstFlowReturn [last_ret](#)
- guint [nth](#)

8.116.1 Detailed Description

Internal data structure for pad in demux / split.

Definition at line 102 of file tensor_common.h.

8.116.2 Member Data Documentation

8.116.2.1 last_ret

```
GstFlowReturn GstTensorPad::last_ret
```

Definition at line 106 of file tensor_common.h.

8.116.2.2 last_ts

```
GstClockTime GstTensorPad::last_ts
```

Definition at line 105 of file tensor_common.h.

8.116.2.3 nth

```
guint GstTensorPad::nth
```

Definition at line 107 of file tensor_common.h.

8.116.2.4 pad

GstPad* GstTensorPad::pad

Definition at line 104 of file tensor_common.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_common.h](#)

8.117 GstTensorQueryEdgeInfo Struct Reference

Internal data structure for nns-edge info to prepare edge connection.

```
#include <tensor_query_server.h>
```

Public Attributes

- gchar * [host](#)
- guint16 [port](#)
- gchar * [dest_host](#)
- guint16 [dest_port](#)
- gchar * [topic](#)
- nns_edge_event_cb [cb](#)
- void * [pdata](#)

8.117.1 Detailed Description

Internal data structure for nns-edge info to prepare edge connection.

Definition at line 28 of file tensor_query_server.h.

8.117.2 Member Data Documentation

8.117.2.1 cb

nns_edge_event_cb GstTensorQueryEdgeInfo::cb

Definition at line 37 of file tensor_query_server.h.

8.117.2.2 dest_host

```
gchar* GstTensorQueryEdgeInfo::dest_host
```

Definition at line 32 of file tensor_query_server.h.

8.117.2.3 dest_port

```
guint16 GstTensorQueryEdgeInfo::dest_port
```

Definition at line 33 of file tensor_query_server.h.

8.117.2.4 host

```
gchar* GstTensorQueryEdgeInfo::host
```

Definition at line 30 of file tensor_query_server.h.

8.117.2.5 pdata

```
void* GstTensorQueryEdgeInfo::pdata
```

Definition at line 38 of file tensor_query_server.h.

8.117.2.6 port

```
guint16 GstTensorQueryEdgeInfo::port
```

Definition at line 31 of file tensor_query_server.h.

8.117.2.7 topic

```
gchar* GstTensorQueryEdgeInfo::topic
```

Definition at line 34 of file tensor_query_server.h.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_query/tensor_query_server.h](#)

8.118 GstTensorQueryServer Struct Reference

[GstTensorQueryServer](#) internal info data structure.

```
#include <tensor_query_server.h>
```

Public Attributes

- guint [id](#)
- gboolean [configured](#)
- GMutex [lock](#)
- GCond [cond](#)
- nns_edge_h [edge_h](#)

8.118.1 Detailed Description

[GstTensorQueryServer](#) internal info data structure.

Definition at line 44 of file `tensor_query_server.h`.

8.118.2 Member Data Documentation

8.118.2.1 cond

GCond `GstTensorQueryServer::cond`

Definition at line 49 of file `tensor_query_server.h`.

8.118.2.2 configured

gboolean `GstTensorQueryServer::configured`

Definition at line 47 of file `tensor_query_server.h`.

8.118.2.3 edge_h

nns_edge_h `GstTensorQueryServer::edge_h`

Definition at line 51 of file `tensor_query_server.h`.

8.118.2.4 id

```
guint GstTensorQueryServer::id
```

Definition at line 46 of file `tensor_query_server.h`.

8.118.2.5 lock

```
GMutex GstTensorQueryServer::lock
```

Definition at line 48 of file `tensor_query_server.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/tensor_query/tensor_query_server.h](#)

8.119 GstTensorRepo Struct Reference

[GstTensorRepo](#) data structure.

```
#include <gsttensor_repo.h>
```

Public Attributes

- guint [num_data](#)
- GMutex [repo_lock](#)
- GCond [repo_cond](#)
- GHashTable * [hash](#)
- gboolean [initialized](#)

8.119.1 Detailed Description

[GstTensorRepo](#) data structure.

Definition at line 58 of file `gsttensor_repo.h`.

8.119.2 Member Data Documentation

8.119.2.1 hash

```
GHashTable* GstTensorRepo::hash
```

Definition at line 63 of file `gsttensor_repo.h`.

8.119.2.2 initialized

```
gboolean GstTensorRepo::initialized
```

Definition at line 64 of file `gsttensor_repo.h`.

8.119.2.3 num_data

```
guint GstTensorRepo::num_data
```

Definition at line 60 of file `gsttensor_repo.h`.

8.119.2.4 repo_cond

```
GCond GstTensorRepo::repo_cond
```

Definition at line 62 of file `gsttensor_repo.h`.

8.119.2.5 repo_lock

```
GMutex GstTensorRepo::repo_lock
```

Definition at line 61 of file `gsttensor_repo.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_repo.h](#)

8.120 GstTensorRepoData Struct Reference

[GstTensorRepo](#) internal data structure.

```
#include <gsttensor_repo.h>
```

Public Attributes

- GstBuffer * [buffer](#)
- GstCaps * [caps](#)
- GCond [cond_push](#)
- GCond [cond_pull](#)
- GMutex [lock](#)
- gboolean [eos](#)
- gboolean [src_changed](#)
- guint [src_id](#)
- gboolean [sink_changed](#)
- guint [sink_id](#)
- gboolean [pushed](#)

8.120.1 Detailed Description

[GstTensorRepo](#) internal data structure.

[GstTensorRepo](#) has GList of [GstTensorRepoData](#).

Definition at line 40 of file `gsttensor_repo.h`.

8.120.2 Member Data Documentation

8.120.2.1 `buffer`

```
GstBuffer* GstTensorRepoData::buffer
```

Definition at line 42 of file `gsttensor_repo.h`.

8.120.2.2 `caps`

```
GstCaps* GstTensorRepoData::caps
```

Definition at line 43 of file `gsttensor_repo.h`.

8.120.2.3 `cond_pull`

```
GCond GstTensorRepoData::cond_pull
```

Definition at line 45 of file `gsttensor_repo.h`.

8.120.2.4 cond_push

GCond GstTensorRepoData::cond_push

Definition at line 44 of file gsttensor_repo.h.

8.120.2.5 eos

gboolean GstTensorRepoData::eos

Definition at line 47 of file gsttensor_repo.h.

8.120.2.6 lock

GMutex GstTensorRepoData::lock

Definition at line 46 of file gsttensor_repo.h.

8.120.2.7 pushed

gboolean GstTensorRepoData::pushed

Definition at line 52 of file gsttensor_repo.h.

8.120.2.8 sink_changed

gboolean GstTensorRepoData::sink_changed

Definition at line 50 of file gsttensor_repo.h.

8.120.2.9 sink_id

guint GstTensorRepoData::sink_id

Definition at line 51 of file gsttensor_repo.h.

8.120.2.10 src_changed

```
gboolean GstTensorRepoData::src_changed
```

Definition at line 48 of file `gsttensor_repo.h`.

8.120.2.11 src_id

```
guint GstTensorRepoData::src_id
```

Definition at line 49 of file `gsttensor_repo.h`.

The documentation for this struct was generated from the following file:

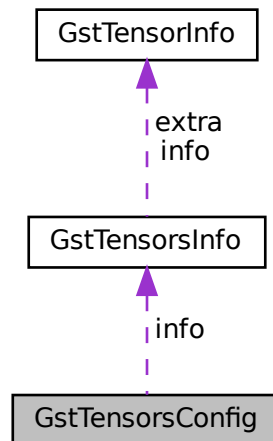
- [nnstreamer/elements/gsttensor_repo.h](#)

8.121 GstTensorsConfig Struct Reference

Internal data structure for configured tensors info (for other/tensors).

```
#include <tensor_typedef.h>
```

Collaboration diagram for GstTensorsConfig:



Public Attributes

- [GstTensorsInfo info](#)
- int [rate_n](#)
- int [rate_d](#)

8.121.1 Detailed Description

Internal data structure for configured tensors info (for other/tensors).

Definition at line 284 of file `tensor_typedef.h`.

8.121.2 Member Data Documentation

8.121.2.1 info

```
GstTensorsInfo GstTensorsConfig::info
```

tensor info

Definition at line 286 of file `tensor_typedef.h`.

8.121.2.2 rate_d

```
int GstTensorsConfig::rate_d
```

framerate is in fraction, which is numerator/denominator

Definition at line 288 of file `tensor_typedef.h`.

8.121.2.3 rate_n

```
int GstTensorsConfig::rate_n
```

framerate is in fraction, which is numerator/denominator

Definition at line 287 of file `tensor_typedef.h`.

The documentation for this struct was generated from the following file:

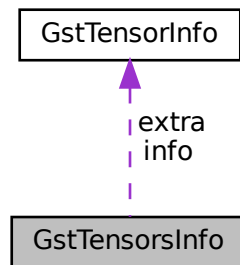
- `nnstreamer/include/tensor_typedef.h`

8.122 GstTensorInfo Struct Reference

Internal meta data exchange format for a other/tensors instance.

```
#include <tensor_typedef.h>
```

Collaboration diagram for GstTensorInfo:



Public Attributes

- unsigned int `num_tensors`
- `GstTensorInfo info` [`NNS_TENSOR_MEMORY_MAX`]
- `GstTensorInfo * extra`
- `tensor_format format`

8.122.1 Detailed Description

Internal meta data exchange format for a other/tensors instance.

Definition at line 273 of file `tensor_typedef.h`.

8.122.2 Member Data Documentation

8.122.2.1 `extra`

```
GstTensorInfo* GstTensorInfo::extra
```

The list of tensor info for tensors whose index is larger than `NNS_TENSOR_MEMORY_MAX`

Definition at line 277 of file `tensor_typedef.h`.

8.122.2.2 format

`tensor_format` `GstTensorsInfo::format`

tensor stream type

Definition at line 278 of file `tensor_typedef.h`.

8.122.2.3 info

`GstTensorInfo` `GstTensorsInfo::info[NNS_TENSOR_MEMORY_MAX]`

The list of tensor info (max `NNS_TENSOR_MEMORY_MAX` as static)

Definition at line 276 of file `tensor_typedef.h`.

8.122.2.4 num_tensors

`unsigned int` `GstTensorsInfo::num_tensors`

The number of tensors

Definition at line 275 of file `tensor_typedef.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/include/tensor_typedef.h`

8.123 parse_accl_args Struct Reference

Accelerator related arguments for parsing.

```
#include <nnstreamer_plugin_api_filter.h>
```

Public Attributes

- `const char *` `in_accl`
- `const char **` `sup_accl`
- `const char *` `auto_accl`
- `const char *` `def_accl`

8.123.1 Detailed Description

Accelerator related arguments for parsing.

Note

if optional accelerators are not set, first entry from supported accelerator is used.

Definition at line 523 of file nnstreamer_plugin_api_filter.h.

8.123.2 Member Data Documentation

8.123.2.1 auto_accl

```
const char* parse_accl_args::auto_accl
```

auto accelerator (optional)

Definition at line 526 of file nnstreamer_plugin_api_filter.h.

8.123.2.2 def_accl

```
const char* parse_accl_args::def_accl
```

default accelerator (optional)

Definition at line 527 of file nnstreamer_plugin_api_filter.h.

8.123.2.3 in_accl

```
const char* parse_accl_args::in_accl
```

user given input

Definition at line 524 of file nnstreamer_plugin_api_filter.h.

8.123.2.4 sup_accl

```
const char** parse_accl_args::sup_accl
```

list of supported accelerator

Definition at line 525 of file nnstreamer_plugin_api_filter.h.

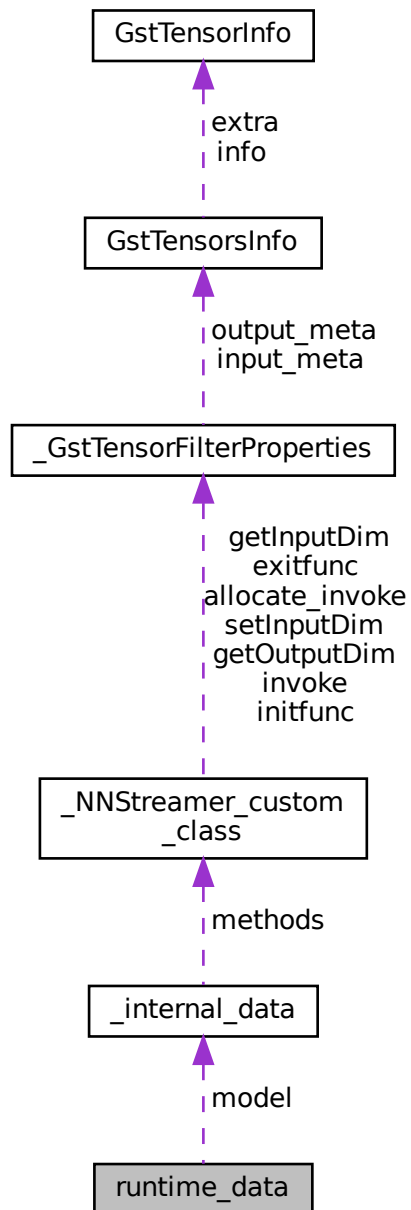
The documentation for this struct was generated from the following file:

- [nnstreamer/include/nnstreamer_plugin_api_filter.h](#)

8.124 runtime_data Struct Reference

The easy-filter user's data.

Collaboration diagram for runtime_data:



Public Attributes

- const [internal_data](#) * [model](#)

8.124.1 Detailed Description

The easy-filter user's data.

Definition at line 53 of file `tensor_filter_custom_easy.c`.

8.124.2 Member Data Documentation

8.124.2.1 model

```
const internal\_data* runtime_data::model
```

Definition at line 55 of file `tensor_filter_custom_easy.c`.

The documentation for this struct was generated from the following file:

- `nnstreamer/tensor_filter/tensor_filter_custom_easy.c`

8.125 subplugin_conf Struct Reference

Public Attributes

- `gchar * path` [[CONF_SOURCE_END](#)]
- `gchar ** files`
- `gchar ** names`

8.125.1 Detailed Description

Definition at line 82 of file `nnstreamer_conf.c`.

8.125.2 Member Data Documentation

8.125.2.1 files

```
gchar** subplugin_conf::files
```

Null terminated list of full filepaths

Definition at line 93 of file `nnstreamer_conf.c`.

8.125.2.2 names

```
gchar** subplugin_conf::names
```

Null terminated list of subplugin names

Definition at line 94 of file `nnstreamer_conf.c`.

8.125.2.3 path

```
gchar* subplugin_conf::path[CONF_SOURCE_END]
```

directory paths

Definition at line 88 of file nnstreamer_conf.c.

The documentation for this struct was generated from the following file:

- nnstreamer/[nnstreamer_conf.c](#)

8.126 subplugin_info_s Struct Reference

```
#include <nnstreamer_conf.h>
```

Public Attributes

- gchar** [names](#)
- gchar** [paths](#)

8.126.1 Detailed Description

Definition at line 75 of file nnstreamer_conf.h.

8.126.2 Member Data Documentation

8.126.2.1 names

```
gchar** subplugin_info_s::names
```

Definition at line 77 of file nnstreamer_conf.h.

8.126.2.2 paths

```
gchar** subplugin_info_s::paths
```

Definition at line 78 of file nnstreamer_conf.h.

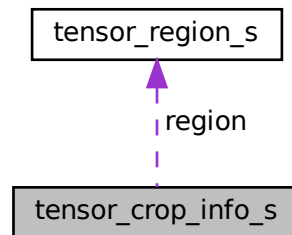
The documentation for this struct was generated from the following file:

- nnstreamer/[nnstreamer_conf.h](#)

8.127 tensor_crop_info_s Struct Reference

Internal data structure to describe cropping tensor data.

Collaboration diagram for tensor_crop_info_s:



Public Attributes

- quint [num](#)
- [tensor_region_s region](#) [NNS_TENSOR_SIZE_LIMIT]

8.127.1 Detailed Description

Internal data structure to describe cropping tensor data.

Todo Add various mode to crop tensor. Now tensor-crop handles NHWC data format only.

Definition at line 62 of file gsttensor_crop.c.

8.127.2 Member Data Documentation

8.127.2.1 num

```
quint tensor_crop_info_s::num
```

Definition at line 64 of file gsttensor_crop.c.

8.127.2.2 region

```
tensor_region_s tensor_crop_info_s::region[NNS_TENSOR_SIZE_LIMIT]
```

Definition at line 65 of file gsttensor_crop.c.

The documentation for this struct was generated from the following file:

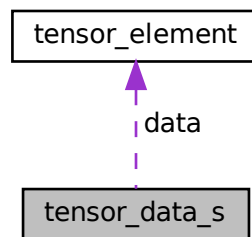
- [nnstreamer/elements/gsttensor_crop.c](#)

8.128 tensor_data_s Struct Reference

Structure for tensor data.

```
#include <tensor_data.h>
```

Collaboration diagram for tensor_data_s:



Public Attributes

- [tensor_type](#) type
- [tensor_element](#) data

8.128.1 Detailed Description

Structure for tensor data.

Definition at line 23 of file tensor_data.h.

8.128.2 Member Data Documentation

8.128.2.1 data

`tensor_element` `tensor_data_s::data`

Definition at line 26 of file `tensor_data.h`.

8.128.2.2 type

`tensor_type` `tensor_data_s::type`

Definition at line 25 of file `tensor_data.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/tensor_data.h`

8.129 tensor_element Union Reference

To make the code simple with all the types. "C++ Template"-like.

```
#include <tensor_typedef.h>
```

Public Attributes

- `int32_t_int32_t`
- `uint32_t_uint32_t`
- `int16_t_int16_t`
- `uint16_t_uint16_t`
- `int8_t_int8_t`
- `uint8_t_uint8_t`
- `double_double`
- `float_float`
- `int64_t_int64_t`
- `uint64_t_uint64_t`

8.129.1 Detailed Description

To make the code simple with all the types. "C++ Template"-like.

Definition at line 231 of file `tensor_typedef.h`.

8.129.2 Member Data Documentation

8.129.2.1 `_double`

```
double tensor_element::_double
```

Definition at line 238 of file `tensor_typedef.h`.

8.129.2.2 `_float`

```
float tensor_element::_float
```

Definition at line 239 of file `tensor_typedef.h`.

8.129.2.3 `_int16_t`

```
int16_t tensor_element::_int16_t
```

Definition at line 234 of file `tensor_typedef.h`.

8.129.2.4 `_int32_t`

```
int32_t tensor_element::_int32_t
```

Definition at line 232 of file `tensor_typedef.h`.

8.129.2.5 `_int64_t`

```
int64_t tensor_element::_int64_t
```

Definition at line 240 of file `tensor_typedef.h`.

8.129.2.6 `_int8_t`

```
int8_t tensor_element::_int8_t
```

Definition at line 236 of file `tensor_typedef.h`.

8.129.2.7 `_uint16_t`

```
uint16_t tensor_element::_uint16_t
```

Definition at line 235 of file `tensor_typedef.h`.

8.129.2.8 `_uint32_t`

```
uint32_t tensor_element::_uint32_t
```

Definition at line 233 of file `tensor_typedef.h`.

8.129.2.9 `_uint64_t`

```
uint64_t tensor_element::_uint64_t
```

Definition at line 241 of file `tensor_typedef.h`.

8.129.2.10 `_uint8_t`

```
uint8_t tensor_element::_uint8_t
```

Definition at line 237 of file `tensor_typedef.h`.

The documentation for this union was generated from the following file:

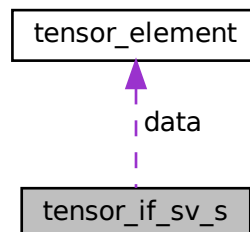
- `nnstreamer/include/tensor_typedef.h`

8.130 `tensor_if_sv_s` Struct Reference

Internal data structure for supplied value.

```
#include <gsttensor_if.h>
```

Collaboration diagram for `tensor_if_sv_s`:



Public Attributes

- `guint32 num`
- `tensor_type type`
- `tensor_element data [2]`

8.130.1 Detailed Description

Internal data structure for supplied value.

Definition at line 105 of file `gsttensor_if.h`.

8.130.2 Member Data Documentation

8.130.2.1 `data`

```
tensor_element tensor_if_sv_s::data[2]
```

Definition at line 109 of file `gsttensor_if.h`.

8.130.2.2 `num`

```
guint32 tensor_if_sv_s::num
```

Definition at line 107 of file `gsttensor_if.h`.

8.130.2.3 `type`

```
tensor_type tensor_if_sv_s::type
```

Definition at line 108 of file `gsttensor_if.h`.

The documentation for this struct was generated from the following file:

- `nnstreamer/elements/gsttensor_if.h`

8.131 `tensor_region_s` Struct Reference

Internal data structure to describe tensor region.

Public Attributes

- [guint x](#)
- [guint y](#)
- [guint w](#)
- [guint h](#)

8.131.1 Detailed Description

Internal data structure to describe tensor region.

SECTION:element-tensor_crop

tensor_crop is a GStreamer element to crop the regions of incoming tensor.

tensor_crop has two always sink pads - raw and info. The raw pad accepts tensor (other/tensor) which will be cropped with crop info. The info pad has capability for flexible tensor stream (other/tensors-flexible), that can have a various buffer size for crop info. Incoming buffer on info pad should be an array of crop info. Note that NNStreamer supports maximum NNS_TENSOR_SIZE_LIMIT memory blocks in a buffer. So, when incoming buffer on info pad has more than NNS_TENSOR_SIZE_LIMIT crop-info array, tensor_crop will ignore the data.

The output is always in the format of other/tensors-flexible.

```
<refsect2> <title>Example launch line</title> [[ gst-launch-1.0 tensor_crop name=crop ! (cropped tensors) ... \
videotestsrc ! videoconvert ! video/x-raw,format=RGB ! tensor_converter ! tee name=t \ t. ! queue ! crop.raw \ t. !
queue ! (process raw video tensor and push buffer which includes crop info) ! crop.info ]] </refsect2>
```

Definition at line 50 of file gsttensor_crop.c.

8.131.2 Member Data Documentation

8.131.2.1 h

```
guint tensor_region_s::h
```

Definition at line 55 of file gsttensor_crop.c.

8.131.2.2 w

```
guint tensor_region_s::w
```

Definition at line 54 of file gsttensor_crop.c.

8.131.2.3 x

```
guint tensor_region_s::x
```

Definition at line 52 of file gsttensor_crop.c.

8.131.2.4 y

```
guint tensor_region_s::y
```

Definition at line 53 of file gsttensor_crop.c.

The documentation for this struct was generated from the following file:

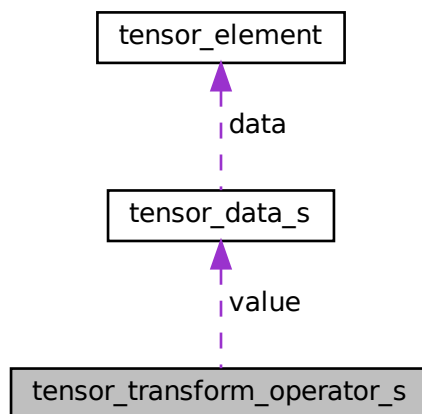
- [nnstreamer/elements/gsttensor_crop.c](#)

8.132 tensor_transform_operator_s Struct Reference

Internal data structure for operator of arithmetic mode.

```
#include <gsttensor_transform.h>
```

Collaboration diagram for tensor_transform_operator_s:



Public Attributes

- [tensor_transform_operator](#) op
- int [applying_ch](#)
- [tensor_data_s](#) value

8.132.1 Detailed Description

Internal data structure for operator of arithmetic mode.

Definition at line 116 of file `gsttensor_transform.h`.

8.132.2 Member Data Documentation

8.132.2.1 `applying_ch`

```
int tensor_transform_operator_s::applying_ch
```

Definition at line 119 of file `gsttensor_transform.h`.

8.132.2.2 `op`

```
tensor_transform_operator tensor_transform_operator_s::op
```

Definition at line 118 of file `gsttensor_transform.h`.

8.132.2.3 `value`

```
tensor_data_s tensor_transform_operator_s::value
```

Definition at line 120 of file `gsttensor_transform.h`.

The documentation for this struct was generated from the following file:

- [nnstreamer/elements/gsttensor_transform.h](#)

8.133 `vstr_helper` Struct Reference

Data structure for `_g_list_foreach_vstr_helper`.

Public Attributes

- `gchar **` [vstr](#)
- `guint` [cursor](#)
- `guint` [size](#)

8.133.1 Detailed Description

Data structure for `_g_list_foreach_vstr_helper`.

Definition at line 255 of file `nnstreamer_conf.c`.

8.133.2 Member Data Documentation

8.133.2.1 cursor

```
guint vstr_helper::cursor
```

The first "empty" element in `vstr`

Definition at line 258 of file `nnstreamer_conf.c`.

8.133.2.2 size

```
guint vstr_helper::size
```

The number of "`g_char *`" in `vstr`, excluding the terminator

Definition at line 259 of file `nnstreamer_conf.c`.

8.133.2.3 vstr

```
gchar** vstr_helper::vstr
```

The `vstr` data (string array)

Definition at line 257 of file `nnstreamer_conf.c`.

The documentation for this struct was generated from the following file:

- [nnstreamer/nnstreamer_conf.c](#)

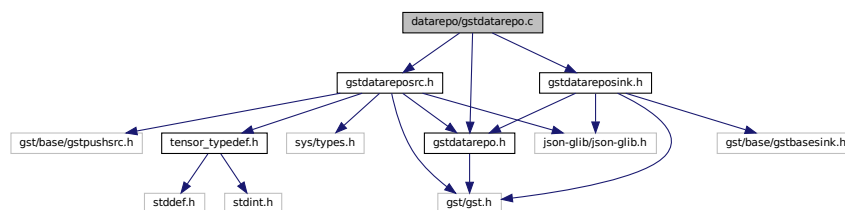
Chapter 9

File Documentation

9.1 datarepo/gstdatarepo.c File Reference

Register datarepo plugins.

```
#include "gstdatarepo.h"  
#include "gstdatareposrc.h"  
#include "gstdatareposink.h"  
Include dependency graph for gstdatarepo.c:
```



Macros

- `#define PACKAGE "NNStreamer MLOps Data Repository Plugins"`

Functions

- `GstDataRepoDataType gst_data_repo_get_data_type_from_caps` (const `GstCaps *caps`)
Get data type from caps.
- static gboolean `plugin_init` (`GstPlugin *plugin`)
The entry point of the Gstreamer datarepo plugin.

9.1.1 Detailed Description

Register datarepo plugins.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

31 January 2023

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

9.1.2 Macro Definition Documentation

9.1.2.1 PACKAGE

```
#define PACKAGE "NNStreamer MLOps Data Repository Plugins"
```

Definition at line 69 of file gstdatarepo.c.

9.1.3 Function Documentation

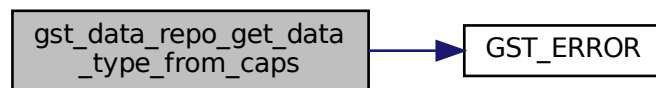
9.1.3.1 `gst_data_repo_get_data_type_from_caps()`

```
GstDataRepoDataType gst_data_repo_get_data_type_from_caps (
    const GstCaps * caps )
```

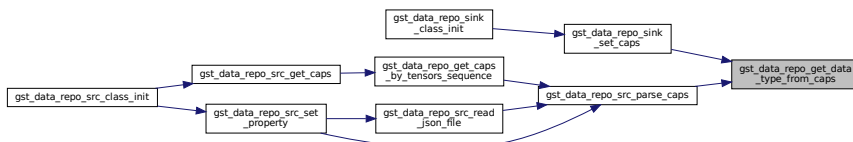
Get data type from caps.

Definition at line 21 of file `gstdatarepo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.1.3.2 `plugin_init()`

```
static gboolean plugin_init (
    GstPlugin * plugin ) [static]
```

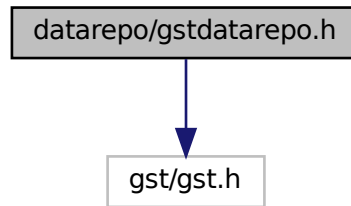
The entry point of the Gstreamer `datarepo` plugin.

Definition at line 55 of file `gstdatarepo.c`.

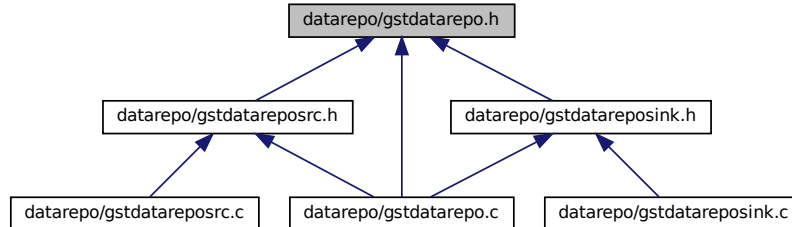
9.2 datarepo/gstdatarepo.h File Reference

GStreamer plugin to read file in MLOps Data repository into buffers.

```
#include <gst/gst.h>
Include dependency graph for gstdatarepo.h:
```



This graph shows which files directly or indirectly include this file:



Enumerations

- enum [GstDataRepoDataType](#) {
[GST_DATA_REPO_DATA_UNKNOWN](#) = 0, [GST_DATA_REPO_DATA_VIDEO](#), [GST_DATA_REPO_DATA_AUDIO](#),
[GST_DATA_REPO_DATA_TEXT](#),
[GST_DATA_REPO_DATA_OCTET](#), [GST_DATA_REPO_DATA_TENSOR](#), [GST_DATA_REPO_DATA_IMAGE](#),
[GST_DATA_REPO_DATA_MAX](#) }

Data type of incoming buffer.

Functions

- [GstDataRepoDataType](#) [gst_data_repo_get_data_type_from_caps](#) (const [GstCaps](#) *caps)
Get data type from caps.

9.2.1 Detailed Description

GStreamer plugin to read file in MLOps Data repository into buffers.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

27 June 2023

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

9.2.2 Enumeration Type Documentation

9.2.2.1 GstDataRepoDataType

enum `GstDataRepoDataType`

Data type of incoming buffer.

Enumerator

GST_DATA_REPO_DATA_UNKNOWN	
GST_DATA_REPO_DATA_VIDEO	
GST_DATA_REPO_DATA_AUDIO	
GST_DATA_REPO_DATA_TEXT	
GST_DATA_REPO_DATA_OCTET	
GST_DATA_REPO_DATA_TENSOR	
GST_DATA_REPO_DATA_IMAGE	
GST_DATA_REPO_DATA_MAX	

Definition at line 23 of file `gstdatarepo.h`.

9.2.3 Function Documentation

Macros

- #define [TENSOR_CAPS](#) [GST_TENSORS_CAP_MAKE](#) ("{ static, flexible, sparse }")
Tensors caps.
- #define [SUPPORTED_VIDEO_FORMAT](#) "{RGB, BGR, RGBx, BGRx, xRGB, xBGR, RGBA, BGRA, ARGB, ABGR, GRAY8}"
Video caps.
- #define [VIDEO_CAPS](#)
- #define [SUPPORTED_AUDIO_FORMAT](#) "{S8, U8, S16LE, S16BE, U16LE, U16BE, S32LE, S32BE, U32LE, U32BE, F32LE, F32BE, F64LE, F64BE}"
Audio caps.
- #define [AUDIO_CAPS](#)
- #define [TEXT_CAPS](#) "text/x-raw, format = (string) utf8"
Text caps.
- #define [OCTET_CAPS](#) "application/octet-stream"
Octet caps.
- #define [IMAGE_CAPS](#)
Image caps.
- #define [GST_CAT_DEFAULT](#) [gst_data_repo_sink_debug](#)
- #define [_do_init](#) [GST_DEBUG_CATEGORY_INIT](#) ([gst_data_repo_sink_debug](#), "datareposink", 0, "datareposink element");
- #define [gst_data_repo_sink_parent_class](#) [parent_class](#)

Enumerations

- enum { [PROP_0](#), [PROP_LOCATION](#), [PROP_JSON](#) }
datareposink properties.

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_data_repo_sink_debug](#))
- [G_DEFINE_TYPE_WITH_CODE](#) ([GstDataRepoSink](#), [gst_data_repo_sink](#), [GST_TYPE_BASE_SINK](#), [_do_init](#))
- static void [gst_data_repo_sink_set_property](#) (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
Setter for datareposink properties.
- static void [gst_data_repo_sink_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
Getter datareposink properties.
- static void [gst_data_repo_sink_finalize](#) (GObject *object)
finalize datareposink.
- static gboolean [gst_data_repo_sink_stop](#) (GstBaseSink *basesink)
Stop datareposink.
- static GstStateChangeReturn [gst_data_repo_sink_change_state](#) (GstElement *element, GstStateChange transition)
Change state of datareposink.
- static GstFlowReturn [gst_data_repo_sink_render](#) (GstBaseSink *bsink, GstBuffer *buffer)
Called when a buffer should be presented or output.
- static GstCaps * [gst_data_repo_sink_get_caps](#) (GstBaseSink *bsink, GstCaps *filter)
Get caps of datareposink.
- static gboolean [gst_data_repo_sink_set_caps](#) (GstBaseSink *bsink, GstCaps *caps)

Set caps of datareposink.

- static gboolean [gst_data_repo_sink_query](#) (GstBaseSink *bsink, GstQuery *query)

Perform a GstQuery on datareposink.

- static void [gst_data_repo_sink_class_init](#) (GstDataRepoSinkClass *klass)

Initialize datareposink class.

- static void [gst_data_repo_sink_init](#) (GstDataRepoSink *sink)

Initialize datareposink.

- static GstFlowReturn [gst_data_repo_sink_write_others](#) (GstDataRepoSink *sink, GstBuffer *buffer)

Function to write others media type (tensors(fixed), video, audio, octet and text)

- static GstFlowReturn [gst_data_repo_sink_write_flexible_or_sparse_tensors](#) (GstDataRepoSink *sink, GstBuffer *buffer)

Function to write flexible tensors or sparse tensors.

- static gchar * [gst_data_repo_sink_get_image_filename](#) (GstDataRepoSink *sink)

Get image filename.

- static GstFlowReturn [gst_data_repo_sink_write_multi_images](#) (GstDataRepoSink *sink, GstBuffer *buffer)

Function to read multi image files.

- static void [gst_data_repo_sink_set_is_static_tensors](#) (GstDataRepoSink *sink)

Set whether the given pad caps are static or not.

- static gboolean [gst_data_repo_sink_open_file](#) (GstDataRepoSink *sink)

Function to open file.

- static gboolean [__write_json](#) (JsonObject *object, const gchar *filename)

Write json to file.

- static gboolean [gst_data_repo_sink_write_json_meta_file](#) (GstDataRepoSink *sink)

write the meta information to a JSON file

Variables

- static GstStaticPadTemplate [sinktemplate](#)

9.3.1 Detailed Description

GStreamer plugin that writes data from buffers to files in in MLOps Data repository.

Copyright (C) 2023 Samsung Electronics Co., Ltd.

Date

30 March 2023

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

```
#!/ Example launch line |[ gst-launch-1.0 videotestsrc ! datareposink location=filename json=video.json gst-launch-1.0 videotestsrc ! pngenc ! datareposink location=image_%02d.png json=video.json gst-launch-1.0 audiotestsrc samplesperbuffer=44100 ! audio/x-raw, format=S16LE, layout=interleaved, rate=44100, channels=1 ! \ datareposink location=filename json=audio.json gst-launch-1.0 datarepositor location=file.dat json=file.json tensors-sequence=2,3 start-sample-index=0 stop-sample-index=199 epochs=1 ! \ other/tensors, format=static, num_↵ tensors=2, framerate=0/1, dimensions=1:1:784:1.1:1:10:1, types=float32.float32 ! \ datareposink location=hyunil.↵ dat json=file.json ]|
```

Copyright (C) 2023 Samsung Electronics Co., Ltd.

Date

30 March 2023

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

9.3.2 Macro Definition Documentation

9.3.2.1 `_do_init`

```
#define _do_init GST_DEBUG_CATEGORY_INIT (gst_data_repo_sink_debug, "datareposink", 0, "datareposink element");
```

Definition at line 92 of file `gstdatareposink.c`.

9.3.2.2 `AUDIO_CAPS`

```
#define AUDIO_CAPS
```

Value:

```
GST_AUDIO_CAPS_MAKE (SUPPORTED_AUDIO_FORMAT) ", " \ "layout = (string) interleaved"
```

Definition at line 55 of file `gstdatareposink.c`.

9.3.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_data_repo_sink_debug
```

Definition at line 91 of file gstdatereposink.c.

9.3.2.4 gst_data_repo_sink_parent_class

```
#define gst_data_repo_sink_parent_class parent_class
```

Definition at line 94 of file gstdatereposink.c.

9.3.2.5 IMAGE_CAPS

```
#define IMAGE_CAPS
```

Value:

```
"image/png, width = (int) [ 16, 1000000 ], height = (int) [ 16, 1000000 ], framerate = (fraction) [ 0/1,
MAX];" \
"image/jpeg, width = (int) [ 16, 65535 ], height = (int) [ 16, 65535 ], framerate = (fraction) [ 0/1,
MAX], sof-marker = (int) { 0, 1, 2, 4, 9 };" \
"image/tiff, endianness = (int) { BIG_ENDIAN, LITTLE_ENDIAN };" \
"image/gif;" \
"image/bmp"
```

Image caps.

Definition at line 68 of file gstdatereposink.c.

9.3.2.6 OCTET_CAPS

```
#define OCTET_CAPS "application/octet-stream"
```

Octet caps.

Definition at line 64 of file gstdatereposink.c.

9.3.2.7 SUPPORTED_AUDIO_FORMAT

```
#define SUPPORTED_AUDIO_FORMAT "{S8, U8, S16LE, S16BE, U16LE, U16BE, S32LE, S32BE, U32LE,
U32BE, F32LE, F32BE, F64LE, F64BE}"
```

Audio caps.

Definition at line 53 of file gstdatereposink.c.

9.3.2.8 SUPPORTED_VIDEO_FORMAT

```
#define SUPPORTED_VIDEO_FORMAT "{RGB, BGR, RGBx, BGRx, xRGB, xBGR, RGBA, BGRA, ARGB, ABGR, GRAY8}"
```

Video caps.

Definition at line 46 of file gstdatareposink.c.

9.3.2.9 TENSOR_CAPS

```
#define TENSOR_CAPS GST_TENSORS_CAP_MAKE ("{ static, flexible, sparse }")
```

Tensors caps.

Definition at line 42 of file gstdatareposink.c.

9.3.2.10 TEXT_CAPS

```
#define TEXT_CAPS "text/x-raw, format = (string) utf8"
```

Text caps.

Definition at line 60 of file gstdatareposink.c.

9.3.2.11 VIDEO_CAPS

```
#define VIDEO_CAPS
```

Value:

```
    GST_VIDEO_CAPS_MAKE (SUPPORTED_VIDEO_FORMAT) ", " \
    "interlace-mode = (string) progressive"
```

Definition at line 48 of file gstdatareposink.c.

9.3.3 Enumeration Type Documentation

9.3.3.1 anonymous enum

anonymous enum

datareposink properties.

Enumerator

PROP_0	
PROP_LOCATION	
PROP_JSON	

Definition at line 83 of file `gstdatareposink.c`.

9.3.4 Function Documentation

9.3.4.1 `__write_json()`

```
static gboolean __write_json (
    JsonObject * object,
    const gchar * filename ) [static]
```

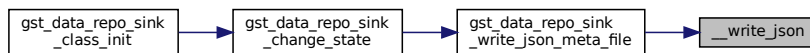
Write json to file.

Definition at line 668 of file `gstdatareposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.3.4.2 `G_DEFINE_TYPE_WITH_CODE()`

```
G_DEFINE_TYPE_WITH_CODE (
    GstDataRepoSink ,
    gst_data_repo_sink ,
    GST_TYPE_BASE_SINK ,
    _do_init )
```

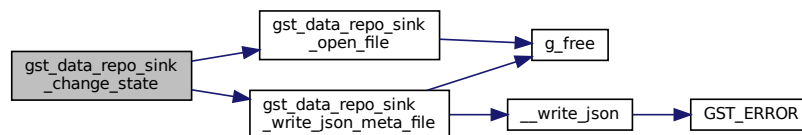

9.3.4.3 `gst_data_repo_sink_change_state()`

```
static GstStateChangeReturn gst_data_repo_sink_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

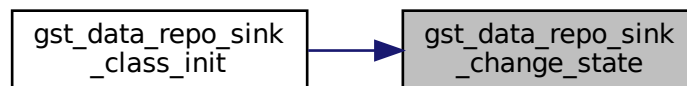
Change state of datareposink.

Definition at line 750 of file `gstdatareposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



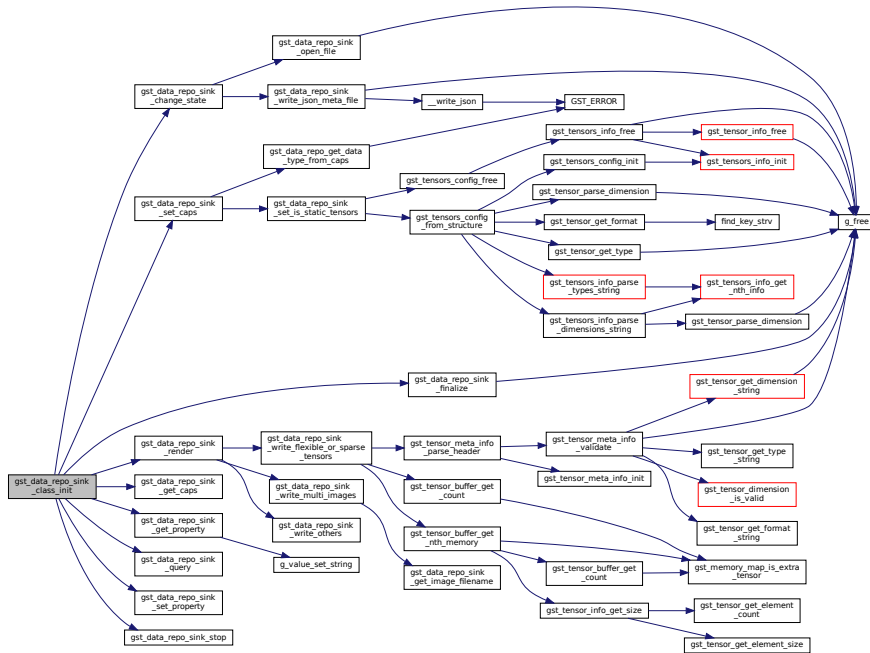
9.3.4.4 `gst_data_repo_sink_class_init()`

```
static void gst_data_repo_sink_class_init (
    GstDataRepoSinkClass * klass ) [static]
```

Initialize datareposink class.

Definition at line 118 of file `gstdatareposink.c`.

Here is the call graph for this function:



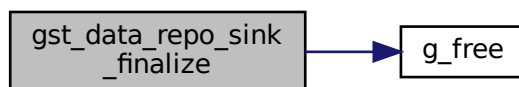
9.3.4.5 gst_data_repo_sink_finalize()

```
static void gst_data_repo_sink_finalize (
    GObject * object ) [static]
```

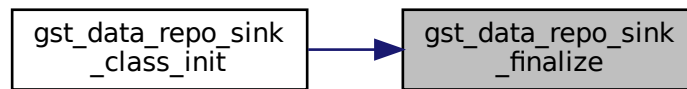
finalize datareposink.

Definition at line 195 of file gstdataposink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.3.4.6 `gst_data_repo_sink_get_caps()`

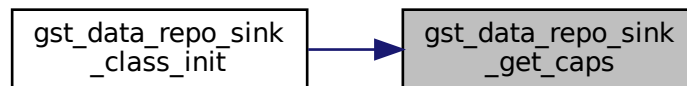
```

static GstCaps * gst_data_repo_sink_get_caps (
    GstBaseSink * bsink,
    GstCaps * filter ) [static]
  
```

Get caps of datareposink.

Definition at line 494 of file `gstdatareposink.c`.

Here is the caller graph for this function:



9.3.4.7 `gst_data_repo_sink_get_image_filename()`

```

static gchar* gst_data_repo_sink_get_image_filename (
    GstDataRepoSink * sink ) [static]
  
```

Get image filename.

Definition at line 400 of file `gstdatareposink.c`.

Here is the caller graph for this function:



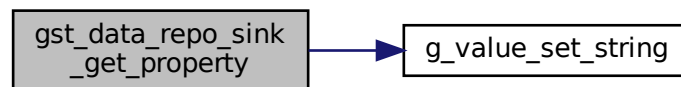
9.3.4.8 `gst_data_repo_sink_get_property()`

```
static void gst_data_repo_sink_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

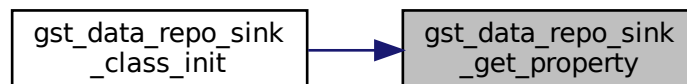
Getter datareposink properties.

Definition at line 251 of file `gstdatareposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.3.4.9 `gst_data_repo_sink_init()`

```
static void gst_data_repo_sink_init (  
    GstDataRepoSink * sink ) [static]
```

Initialize datareposink.

Definition at line 174 of file `gstdatareposink.c`.

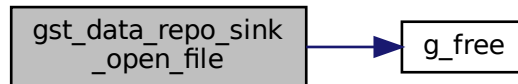
9.3.4.10 `gst_data_repo_sink_open_file()`

```
static gboolean gst_data_repo_sink_open_file (  
    GstDataRepoSink * sink ) [static]
```

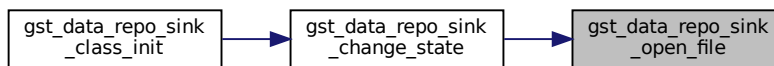
Function to open file.

Definition at line 590 of file gstdatareposink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



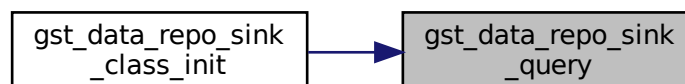
9.3.4.11 `gst_data_repo_sink_query()`

```
static gboolean gst_data_repo_sink_query (  
    GstBaseSink * sink,  
    GstQuery * query ) [static]
```

Perform a GstQuery on datareposink.

Definition at line 564 of file gstdatareposink.c.

Here is the caller graph for this function:



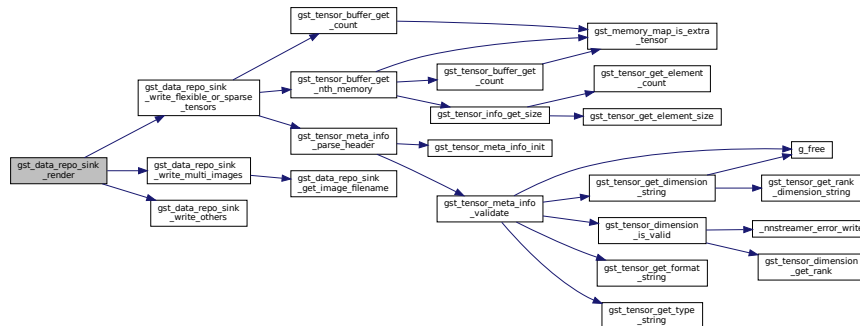
9.3.4.12 `gst_data_repo_sink_render()`

```
static GstFlowReturn gst_data_repo_sink_render (
    GstBaseSink * bsink,
    GstBuffer * buffer ) [static]
```

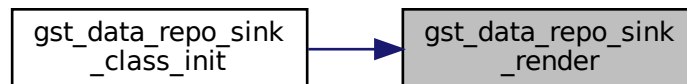
Called when a buffer should be presented or output.

Definition at line 467 of file `gstdatareposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



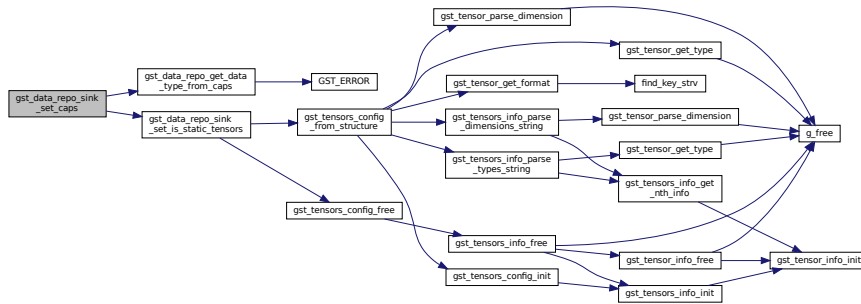
9.3.4.13 `gst_data_repo_sink_set_caps()`

```
static gboolean gst_data_repo_sink_set_caps (
    GstBaseSink * bsink,
    GstCaps * caps ) [static]
```

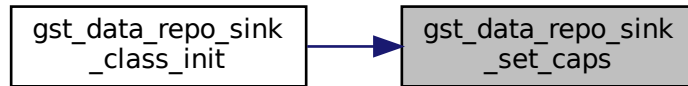
Set caps of `datareposink`.

Definition at line 539 of file `gstdatareposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



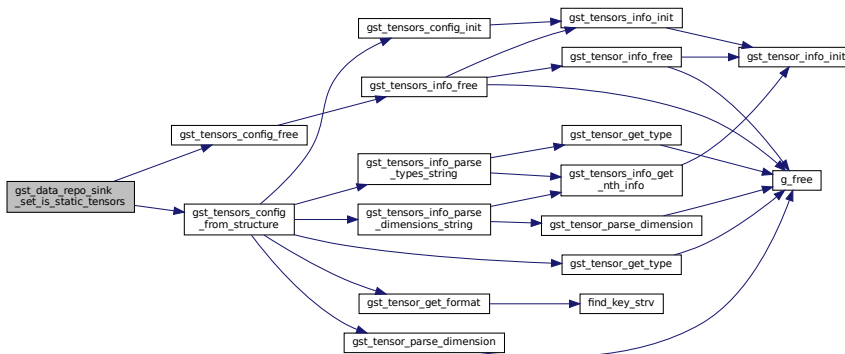
9.3.4.14 gst_data_repo_sink_set_is_static_tensors()

```
static void gst_data_repo_sink_set_is_static_tensors (
    GstDataRepoSink * sink ) [static]
```

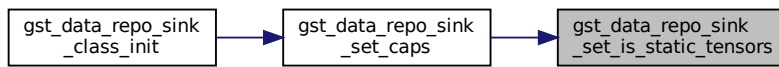
Set whether the given pad caps are static or not.

Definition at line 520 of file gstdatareposink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.3.4.15 `gst_data_repo_sink_set_property()`

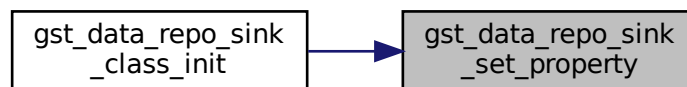
```

static void gst_data_repo_sink_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

Setter for datareposink properties.

Definition at line 227 of file `gstdatareposink.c`.

Here is the caller graph for this function:



9.3.4.16 `gst_data_repo_sink_stop()`

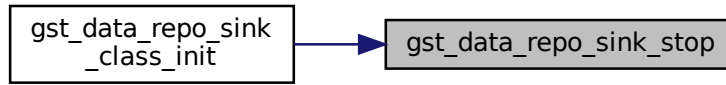
```

static gboolean gst_data_repo_sink_stop (
    GstBaseSink * basesink ) [static]
  
```

Stop datareposink.

Definition at line 652 of file `gstdatareposink.c`.

Here is the caller graph for this function:



9.3.4.17 gst_data_repo_sink_write_flexible_or_sparse_tensors()

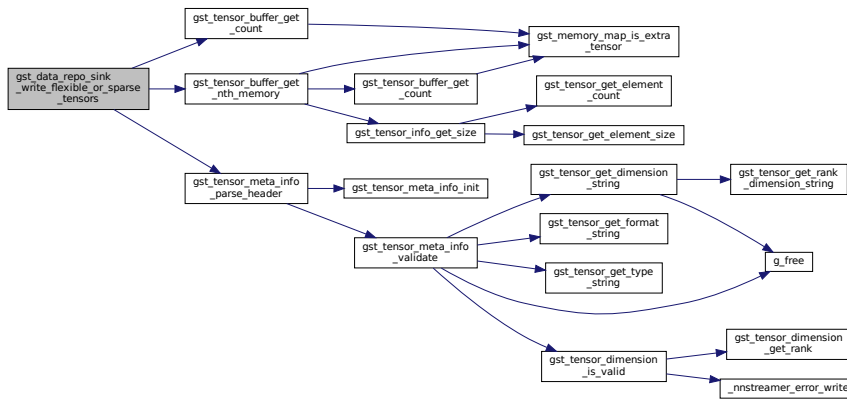
```

static GstFlowReturn gst_data_repo_sink_write_flexible_or_sparse_tensors (
    GstDataRepoSink * sink,
    GstBuffer * buffer ) [static]
  
```

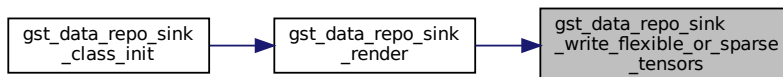
Function to write flexible tensors or sparse tensors.

Definition at line 317 of file gstdatareposink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



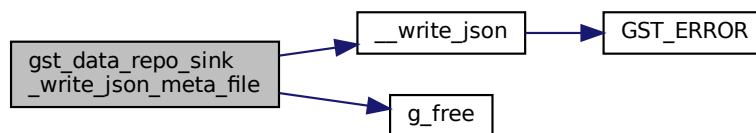
9.3.4.18 `gst_data_repo_sink_write_json_meta_file()`

```
static gboolean gst_data_repo_sink_write_json_meta_file (
    GstDataRepoSink * sink ) [static]
```

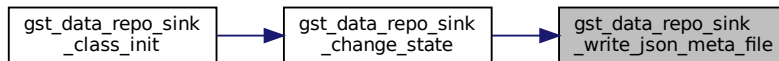
write the meta information to a JSON file

Definition at line 696 of file `gstdatareposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



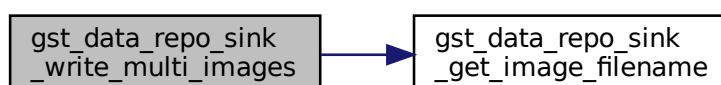
9.3.4.19 `gst_data_repo_sink_write_multi_images()`

```
static GstFlowReturn gst_data_repo_sink_write_multi_images (
    GstDataRepoSink * sink,
    GstBuffer * buffer ) [static]
```

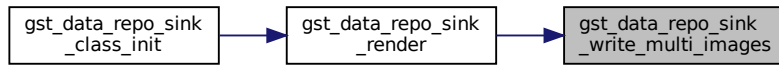
Function to read multi image files.

Definition at line 424 of file `gstdatareposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.3.4.20 `gst_data_repo_sink_write_others()`

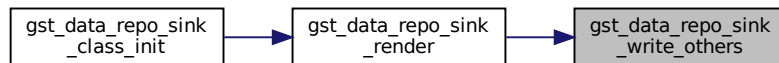
```

static GstFlowReturn gst_data_repo_sink_write_others (
    GstDataRepoSink * sink,
    GstBuffer * buffer ) [static]
  
```

Function to write others media type (tensors(fixed), video, audio, octet and text)

Definition at line 275 of file `gstdatareposink.c`.

Here is the caller graph for this function:



9.3.4.21 `GST_DEBUG_CATEGORY_STATIC()`

```

GST_DEBUG_CATEGORY_STATIC (
    gst_data_repo_sink_debug )
  
```

9.3.5 Variable Documentation

9.3.5.1 `sinktemplate`

```

GstStaticPadTemplate sinktemplate [static]
  
```

Initial value:

=

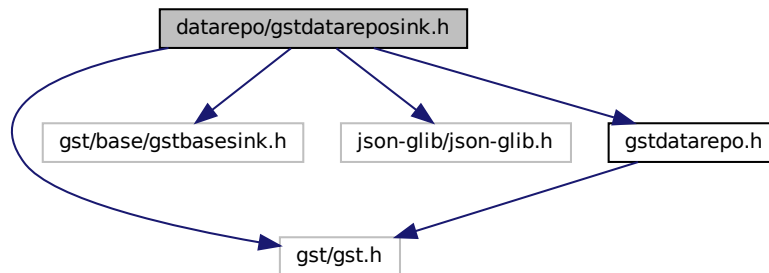
```

GST_STATIC_PAD_TEMPLATE ("sink", GST_PAD_SINK, GST_PAD_ALWAYS,
    GST_STATIC_CAPS ( TENSOR_CAPS ";" VIDEO_CAPS ";" AUDIO_CAPS ";" IMAGE_CAPS
        ";" TEXT_CAPS ";" OCTET_CAPS ))
  
```

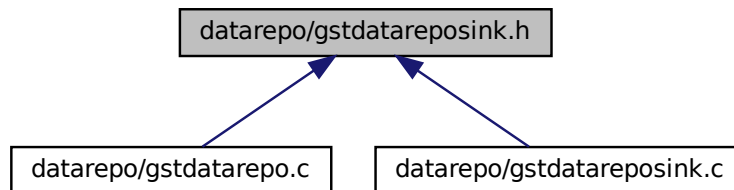
Definition at line 75 of file `gstdatareposink.c`.

9.4 datarepo/gstdatareposink.h File Reference

```
#include <gst/gst.h>
#include <gst/base/gstbasesink.h>
#include <json-glib/json-glib.h>
#include "gstdatarepo.h"
Include dependency graph for gstdatareposink.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_GstDataRepoSink`](#)
GstDataRepoSink data structure.
- [struct `_GstDataRepoSinkClass`](#)
GstDataRepoSinkClass data structure.

Macros

- [#define `GST_TYPE_DATA_REPO_SINK`](#) (`gst_data_repo_sink_get_type()`)
- [#define `GST_DATA_REPO_SINK\(obj\)`](#) (`G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_DATA_REPO_SINK, GstDataRepoSink)`)
- [#define `GST_DATA_REPO_SINK_CLASS\(klass\)`](#) (`G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_DATA_REPO_SINK, GstDataRepoSinkClass)`)
- [#define `GST_IS_DATA_REPO_SINK\(obj\)`](#) (`G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_DATA_REPO_SINK)`)
- [#define `GST_IS_DATA_REPO_SINK_CLASS\(klass\)`](#) (`G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_DATA_REPO_SINK)`)
- [#define `GST_DATA_REPO_SINK_CAST\(obj\)`](#) (`((GstDataRepoSink *)obj)`)

Typedefs

- typedef struct [_GstDataRepoSink](#) [GstDataRepoSink](#)
- typedef struct [_GstDataRepoSinkClass](#) [GstDataRepoSinkClass](#)

Functions

- GType [gst_data_repo_sink_get_type](#) (void)

9.4.1 Macro Definition Documentation

9.4.1.1 GST_DATA_REPO_SINK

```
#define GST_DATA_REPO_SINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_DATA_REPO_SINK, GstDataRepoSink))
```

Definition at line 24 of file `gstdatareposink.h`.

9.4.1.2 GST_DATA_REPO_SINK_CAST

```
#define GST_DATA_REPO_SINK_CAST(  
    obj ) ((GstDataRepoSink *)obj)
```

Definition at line 32 of file `gstdatareposink.h`.

9.4.1.3 GST_DATA_REPO_SINK_CLASS

```
#define GST_DATA_REPO_SINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_DATA_REPO_SINK, GstDataRepoSinkClass))
```

Definition at line 26 of file `gstdatareposink.h`.

9.4.1.4 GST_IS_DATA_REPO_SINK

```
#define GST_IS_DATA_REPO_SINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_DATA_REPO_SINK))
```

Definition at line 28 of file `gstdatareposink.h`.

9.4.1.5 GST_IS_DATA_REPO_SINK_CLASS

```
#define GST_IS_DATA_REPO_SINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_DATA_REPO_SINK))
```

Definition at line 30 of file gstdatereposink.h.

9.4.1.6 GST_TYPE_DATA_REPO_SINK

```
#define GST_TYPE_DATA_REPO_SINK (gst_data_repo_sink_get_type())
```

Definition at line 22 of file gstdatereposink.h.

9.4.2 Typedef Documentation

9.4.2.1 GstDataRepoSink

```
typedef struct _GstDataRepoSink GstDataRepoSink
```

Definition at line 34 of file gstdatereposink.h.

9.4.2.2 GstDataRepoSinkClass

```
typedef struct _GstDataRepoSinkClass GstDataRepoSinkClass
```

Definition at line 35 of file gstdatereposink.h.

9.4.3 Function Documentation

9.4.3.1 gst_data_repo_sink_get_type()

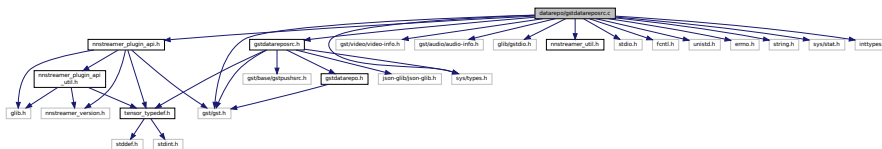
```
GType gst_data_repo_sink_get_type (  
    void )
```

9.5 datarepo/gstdatareposrc.c File Reference

GStreamer plugin to read file in MLOps Data repository into buffers.

```
#include <gst/gst.h>
#include <gst/video/video-info.h>
#include <gst/audio/audio-info.h>
#include <glib/gstdio.h>
#include <nnstreamer_plugin_api.h>
#include <nnstreamer_util.h>
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <inttypes.h>
#include "gstdatareposrc.h"
```

Include dependency graph for `gstdatareposrc.c`:



Macros

- #define `struct_stat` struct stat
- #define `S_ISREG(mode)` ((mode)&_S_IFREG)
- #define `S_ISDIR(mode)` ((mode)&_S_IFDIR)
- #define `S_ISSOCK(x)` (0)
- #define `O_BINARY` (0)
- #define `GST_CAT_DEFAULT` `gst_data_repo_src_debug`
- #define `DEFAULT_INDEX` 0
- #define `DEFAULT_EPOCHS` 1
- #define `DEFAULT_IS_SHUFFLE` TRUE
- #define `_do_init` GST_DEBUG_CATEGORY_INIT (gst_data_repo_src_debug, "datareposrc", 0, "datareposrc element");
- #define `gst_data_repo_src_parent_class` parent_class

Enumerations

- enum {
`PROP_0`, `PROP_LOCATION`, `PROP_JSON`, `PROP_START_SAMPLE_INDEX`,
`PROP_STOP_SAMPLE_INDEX`, `PROP_EPOCHS`, `PROP_IS_SHUFFLE`, `PROP_TENSORS_SEQUENCE`,
`PROP_CAPS` }

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_data_repo_src_debug](#))
- static void [gst_data_repo_src_finalize](#) (GObject *object)

Function to finalize instance.
- static GstStateChangeReturn [gst_data_repo_src_change_state](#) (GstElement *element, GstStateChange transition)

Change state of datarepositor.
- static void [gst_data_repo_src_set_property](#) (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)

Setter for datarepositor properties.
- static void [gst_data_repo_src_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)

Getter datarepositor properties.
- static gboolean [gst_data_repo_src_stop](#) (GstBaseSrc *basesrc)

Stop datarepositor, unmap and close the file.
- static GstCaps * [gst_data_repo_src_get_caps](#) (GstBaseSrc *basesrc, GstCaps *filter)

Get caps for caps negotiation.
- static gboolean [gst_data_repo_src_set_caps](#) (GstBaseSrc *basesrc, GstCaps *caps)

caps after caps negotiation
- static GstFlowReturn [gst_data_repo_src_create](#) (GstPushSrc *pushsrc, GstBuffer **buffer)

Function to create a buffer.
- [G_DEFINE_TYPE_WITH_CODE](#) (GstDataRepoSrc, [gst_data_repo_src](#), GST_TYPE_PUSH_SRC, [_do_init](#))
- static void [gst_data_repo_src_class_init](#) (GstDataRepoSrcClass *klass)

initialize the datarepositor's class
- static void [gst_data_repo_src_init](#) (GstDataRepoSrc *src)

Initialize datarepositor.
- static gboolean [gst_data_repo_src_parse_caps](#) (GstDataRepoSrc *src, GstCaps *caps)

Parse gst-caps to get media type and data size.
- static gboolean [gst_data_repo_src_set_file_path](#) (GstDataRepoSrc *src, const int prop, const gchar *file_path, GError **err)

Function to set file path.
- static gboolean [gst_data_repo_src_set_tensors_sequence](#) (GstDataRepoSrc *src)

Function to set tensors sequence.
- static guint64 [gst_data_repo_src_get_file_offset](#) (GstDataRepoSrc *src, guint sample_index)

Function to get file offset with sample index.
- static void [gst_data_repo_src_shuffle_samples_index](#) (GstDataRepoSrc *src)

Function to shuffle samples index.
- static gboolean [gst_data_repo_src_epoch_is_done](#) (GstDataRepoSrc *src)

Function to check epoch and EOS.
- static GstFlowReturn [gst_data_repo_src_read_tensors](#) (GstDataRepoSrc *src, GstBuffer **buffer)

Function to read tensors.
- static guint [gst_data_repo_src_get_num_tensors](#) (GstDataRepoSrc *src, guint shuffled_index)

Function to get num_tensors from tensor_count_array.
- static GstFlowReturn [gst_data_repo_src_read_flexible_or_sparse_tensors](#) (GstDataRepoSrc *src, GstBuffer **buffer)

Function to read flexible or sparse tensors.
- static gchar * [gst_data_repo_src_get_image_filename](#) (GstDataRepoSrc *src)

Get image filename.
- static GstFlowReturn [gst_data_repo_src_read_multi_images](#) (GstDataRepoSrc *src, GstBuffer **buffer)

Function to read multi image files.
- static GstFlowReturn [gst_data_repo_src_read_others](#) (GstDataRepoSrc *src, GstBuffer **buffer)

- Function to read others media type (video, audio, octet and text)*

 - static gboolean [gst_data_repo_src_start](#) ([GstDataRepoSrc](#) *src)

Start datareposrc, open the file.
- static void [gst_data_repo_src_set_timestamp](#) ([GstDataRepoSrc](#) *src, [GstBuffer](#) *buffer)
- Set timestamp.*

 - static gboolean [gst_data_repo_get_caps_by_tensors_sequence](#) ([GstDataRepoSrc](#) *src)

Get caps with tensors_sequence applied.
- static gboolean [gst_data_repo_src_read_json_file](#) ([GstDataRepoSrc](#) *src)
- Read JSON file.*

Variables

- static [GstStaticPadTemplate](#) [srctemplate](#)

9.5.1 Detailed Description

GStreamer plugin to read file in MLOps Data repository into buffers.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

31 January 2023

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

```
#!/ Example launch line [[ gst-launch-1.0 datareposrc location=mnist.data json=mnist.json start-sample-index=3
stop-sample-index=202 epochs=5 ! \ ! tensor_sink gst-launch-1.0 datareposrc location=image_%02ld.png
json=image.json start-sample-index=3 stop-sample-index=9 epochs=2 ! fakesink gst-launch-1.0 datareposrc loca-
tion=audiofile json=audio.json ! fakesink gst-launch-1.0 datareposrc location=videofile json=video.json ! fakesink
]] [[ Unknown sample file(has not JSON) need to set caps and blocksize or set caps to tensors type without
blocksize gst-launch-1.0 datareposrc blocksize=3176 location=unknown.data start-sample-index=3 stop-sam-
ple-index=202 epochs=5 \ caps ="application/octet-stream" ! tensor_converter input-dim=1:1:784:1,1:1:10:1 input-
type=float32,float32 ! fakesink ]] or [[ gst-launch-1.0 datareposrc location=unknown.data start-sample-index=3
stop-sample-index=202 epochs=5 \ caps ="other/tensors, format=(string)static, framerate=(fraction)0/1, num_←
tensors=(int)2, dimensions=(string)1:1:784:1.1:1:10:1, types=(string)float32.float32" \ ! fakesink ]]
```

9.5.2 Macro Definition Documentation

9.5.2.1 `_do_init`

```
#define _do_init GST_DEBUG_CATEGORY_INIT (gst_data_repo_src_debug, "datareposrc", 0, "datareposrc  
element");
```

Definition at line 108 of file `gstdatareposrc.c`.

9.5.2.2 `DEFAULT_EPOCHS`

```
#define DEFAULT_EPOCHS 1
```

Definition at line 91 of file `gstdatareposrc.c`.

9.5.2.3 `DEFAULT_INDEX`

```
#define DEFAULT_INDEX 0
```

Definition at line 90 of file `gstdatareposrc.c`.

9.5.2.4 `DEFAULT_IS_SHUFFLE`

```
#define DEFAULT_IS_SHUFFLE TRUE
```

Definition at line 92 of file `gstdatareposrc.c`.

9.5.2.5 `GST_CAT_DEFAULT`

```
#define GST_CAT_DEFAULT gst_data_repo_src_debug
```

Definition at line 74 of file `gstdatareposrc.c`.

9.5.2.6 `gst_data_repo_src_parent_class`

```
#define gst_data_repo_src_parent_class parent_class
```

Definition at line 111 of file `gstdatareposrc.c`.

9.5.2.7 O_BINARY

```
#define O_BINARY (0)
```

Definition at line 65 of file gstdatareposrc.c.

9.5.2.8 S_ISDIR

```
#define S_ISDIR(  
    mode ) ((mode) &_S_IFDIR)
```

Definition at line 58 of file gstdatareposrc.c.

9.5.2.9 S_ISREG

```
#define S_ISREG(  
    mode ) ((mode) &_S_IFREG)
```

Definition at line 55 of file gstdatareposrc.c.

9.5.2.10 S_ISSOCK

```
#define S_ISSOCK(  
    x ) (0)
```

Definition at line 62 of file gstdatareposrc.c.

9.5.2.11 struct_stat

```
#define struct_stat struct stat
```

Definition at line 52 of file gstdatareposrc.c.

9.5.3 Enumeration Type Documentation

9.5.3.1 anonymous enum

```
anonymous enum
```

Enumerator

PROP_0	
PROP_LOCATION	
PROP_JSON	
PROP_START_SAMPLE_INDEX	
PROP_STOP_SAMPLE_INDEX	
PROP_EPOCHS	
PROP_IS_SHUFFLE	
PROP_TENSORS_SEQUENCE	
PROP_CAPS	

Definition at line 77 of file gstdatareposrc.c.

9.5.4 Function Documentation

9.5.4.1 G_DEFINE_TYPE_WITH_CODE()

```
G_DEFINE_TYPE_WITH_CODE (  
    GstDataRepoSrc ,  
    gst_data_repo_src ,  
    GST_TYPE_PUSH_SRC ,  
    _do_init )
```

9.5.4.2 gst_data_repo_get_caps_by_tensors_sequence()

```
static gboolean gst_data_repo_get_caps_by_tensors_sequence (  
    GstDataRepoSrc * src ) [static]
```

Get caps with tensors_sequence applied.

Definition at line 1326 of file gstdatareposrc.c.

Here is the caller graph for this function:



9.5.4.9 gst_data_repo_src_get_file_offset()

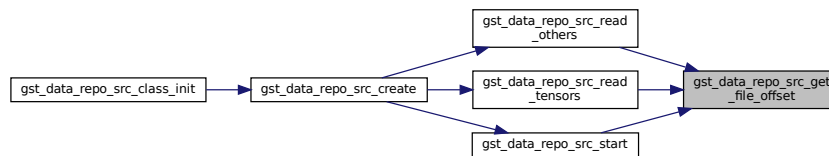
```

static guint64 gst_data_repo_src_get_file_offset (
    GstDataRepoSrc * src,
    guint sample_index ) [static]
  
```

Function to get file offset with sample index.

Definition at line 504 of file gstdatareposrc.c.

Here is the caller graph for this function:



9.5.4.10 gst_data_repo_src_get_image_filename()

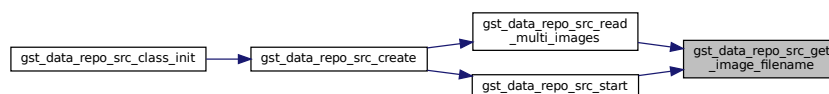
```

static gchar* gst_data_repo_src_get_image_filename (
    GstDataRepoSrc * src ) [static]
  
```

Get image filename.

Definition at line 894 of file gstdatareposrc.c.

Here is the caller graph for this function:



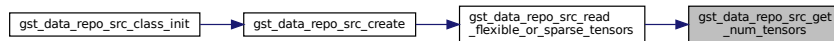
9.5.4.11 `gst_data_repo_src_get_num_tensors()`

```
static guint gst_data_repo_src_get_num_tensors (
    GstDataRepoSrc * src,
    guint shuffled_index ) [static]
```

Function to get `num_tensors` from `tensor_count_array`.

Definition at line 708 of file `gstdatareposrc.c`.

Here is the caller graph for this function:



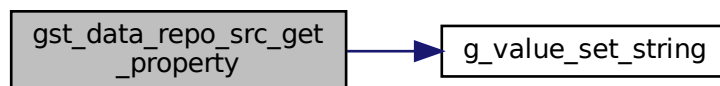
9.5.4.12 `gst_data_repo_src_get_property()`

```
static void gst_data_repo_src_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

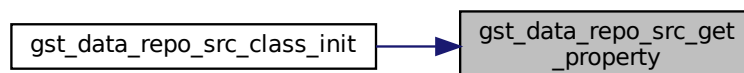
Getter `datareposrc` properties.

Definition at line 1630 of file `gstdatareposrc.c`.

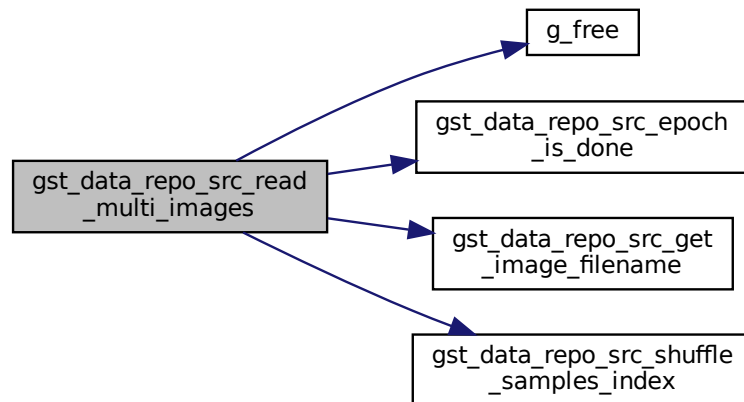
Here is the call graph for this function:



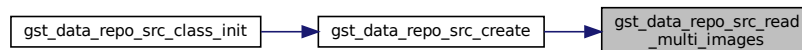
Here is the caller graph for this function:



Here is the call graph for this function:



Here is the caller graph for this function:



9.5.4.18 `gst_data_repo_src_read_others()`

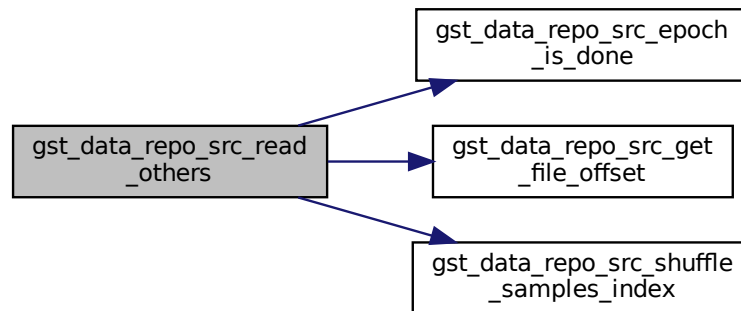
```

static GstFlowReturn gst_data_repo_src_read_others (
    GstDataRepoSrc * src,
    GstBuffer ** buffer ) [static]
  
```

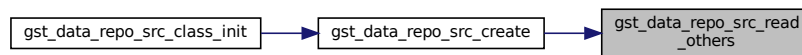
Function to read others media type (video, audio, octet and text)

Definition at line 1002 of file `gstdatareposrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.5.4.19 `gst_data_repo_src_read_tensors()`

```

static GstFlowReturn gst_data_repo_src_read_tensors (
    GstDataRepoSrc * src,
    GstBuffer ** buffer ) [static]
  
```

Function to read tensors.

offset and sample_offset(byte size) are from 0. if the size of one sample is 6352 and num_tensors is 4, dimensions are '1:1:784:1', '1:1:10:1', '1:1:784:1' and '1:1:10:1' with float32.

9.5.4.20 the offset of the second sample is as follows.

```

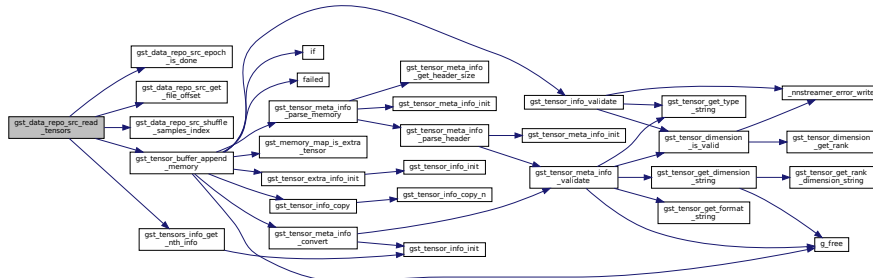
sample_offset: 6352 tensors index: [ 0 | 1 | 2 | 3 ] tensors_size: [ 3136 | 40 | 3136 | 40 ] tensors_offset: [ 0 | 3136 | 3176 | 6312 ]
  
```

9.5.4.21 fd offset: [6352 | 9488 | 9528 | 12664]

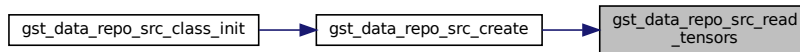
if user sets "tensor-sequence=2,1", datareposrc read offset 9528 then 9488.

Definition at line 570 of file gstdatareposrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.5.4.22 gst_data_reposrc_set_caps()

```
static gboolean gst_data_reposrc_set_caps (
    GstBaseSrc * basesrc,
    GstCaps * caps ) [static]
```

caps after caps negotiation

Definition at line 1401 of file gstdatareposrc.c.

Here is the caller graph for this function:



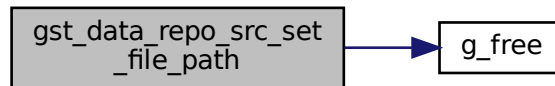
9.5.4.23 `gst_data_repo_src_set_file_path()`

```
static gboolean gst_data_repo_src_set_file_path (
    GstDataRepoSrc * src,
    const int prop,
    const gchar * file_path,
    GError ** err ) [static]
```

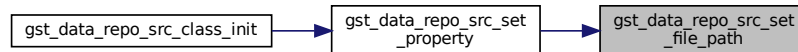
Function to set file path.

Definition at line 396 of file `gstdatareposrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.5.4.24 `gst_data_repo_src_set_property()`

```
static void gst_data_repo_src_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

Setter for `datareposrc` properties.

To get caps, read JSON before Caps negotiation, to get information on sample data

let's retry set tensors-sequence. if caps property is set later than tensors-sequence property, setting tensors-sequence fails because caps information is unknown.

Definition at line 1562 of file `gstdatareposrc.c`.

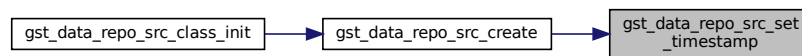
9.5.4.26 `gst_data_repo_src_set_timestamp()`

```
static void gst_data_repo_src_set_timestamp (  
    GstDataRepoSrc * src,  
    GstBuffer * buffer ) [static]
```

Set timestamp.

Definition at line 1225 of file `gstdatareposrc.c`.

Here is the caller graph for this function:



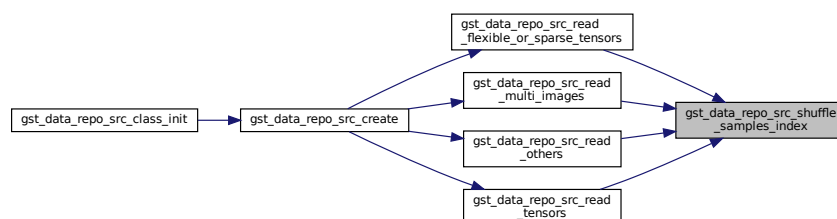
9.5.4.27 `gst_data_repo_src_shuffle_samples_index()`

```
static void gst_data_repo_src_shuffle_samples_index (  
    GstDataRepoSrc * src ) [static]
```

Function to shuffle samples index.

Definition at line 521 of file `gstdatareposrc.c`.

Here is the caller graph for this function:



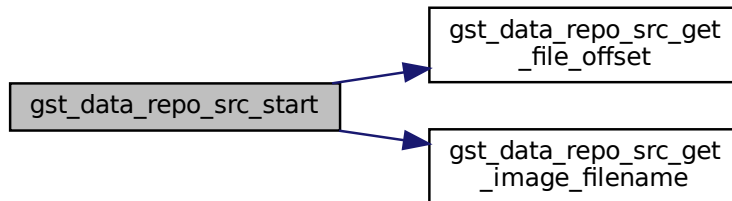
9.5.4.28 `gst_data_repo_src_start()`

```
static gboolean gst_data_repo_src_start (
    GstDataRepoSrc * src ) [static]
```

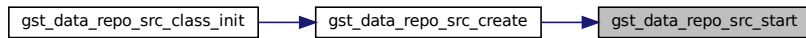
Start datareposrc, open the file.

Definition at line 1102 of file `gstdatareposrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



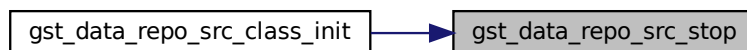
9.5.4.29 `gst_data_repo_src_stop()`

```
static gboolean gst_data_repo_src_stop (
    GstBaseSrc * basesrc ) [static]
```

Stop datareposrc, unmap and close the file.

Definition at line 1311 of file `gstdatareposrc.c`.

Here is the caller graph for this function:



9.5.4.30 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_data_repo_src_debug )
```

9.5.5 Variable Documentation

9.5.5.1 srctemplate

```
GstStaticPadTemplate srctemplate [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS_ANY)
```

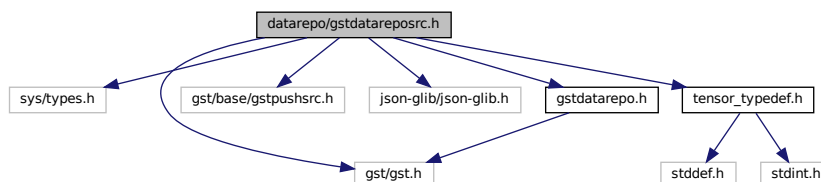
Definition at line 68 of file gstdatareposrc.c.

9.6 datarepo/gstdatareposrc.h File Reference

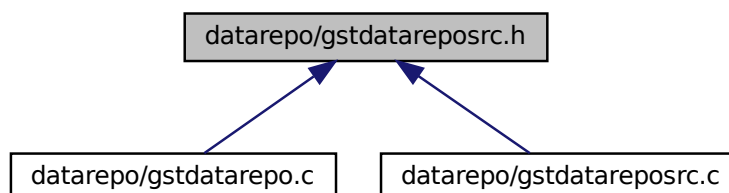
GStreamer plugin to read file in MLOps Data repository into buffers.

```
#include <sys/types.h>
#include <gst/gst.h>
#include <gst/base/gstpushsrc.h>
#include <json-glib/json-glib.h>
#include <tensor_typedef.h>
#include "gstdatarepo.h"
```

Include dependency graph for gstdatareposrc.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstDataRepoSrc](#)
GstDataRepoSrc data structure.
- struct [_GstDataRepoSrcClass](#)
GstDataRepoSrcClass data structure.

Macros

- #define [GST_TYPE_DATA_REPO_SRC](#) ([gst_data_repo_src_get_type\(\)](#))
- #define [GST_DATA_REPO_SRC\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_DATA_REPO_SRC, GstDataRepoSrc\)](#))
- #define [GST_DATA_REPO_SRC_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_DATA_REPO_SRC, GstDataRepoSrcClass\)](#))
- #define [GST_IS_DATA_REPO_SRC\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_DATA_REPO_SRC\)](#))
- #define [GST_IS_DATA_REPO_SRC_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_DATA_REPO_SRC\)](#))

Typedefs

- typedef struct [_GstDataRepoSrc](#) [GstDataRepoSrc](#)
- typedef struct [_GstDataRepoSrcClass](#) [GstDataRepoSrcClass](#)

Functions

- GType [gst_data_repo_src_get_type](#) (void)

9.6.1 Detailed Description

GStreamer plugin to read file in MLOps Data repository into buffers.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

31 January 2023

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

9.6.2 Macro Definition Documentation

9.6.2.1 GST_DATA_REPO_SRC

```
#define GST_DATA_REPO_SRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_DATA_REPO_SRC, GstDataRepoSrc))
```

Definition at line 26 of file `gstdatareposrc.h`.

9.6.2.2 GST_DATA_REPO_SRC_CLASS

```
#define GST_DATA_REPO_SRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_DATA_REPO_SRC, GstDataRepoSrcClass))
```

Definition at line 28 of file `gstdatareposrc.h`.

9.6.2.3 GST_IS_DATA_REPO_SRC

```
#define GST_IS_DATA_REPO_SRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_DATA_REPO_SRC))
```

Definition at line 30 of file `gstdatareposrc.h`.

9.6.2.4 GST_IS_DATA_REPO_SRC_CLASS

```
#define GST_IS_DATA_REPO_SRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_DATA_REPO_SRC))
```

Definition at line 32 of file `gstdatareposrc.h`.

9.6.2.5 GST_TYPE_DATA_REPO_SRC

```
#define GST_TYPE_DATA_REPO_SRC (gst_data_repo_src_get_type())
```

Definition at line 24 of file `gstdatareposrc.h`.

9.6.3 Typedef Documentation

9.6.3.1 GstDataRepoSrc

```
typedef struct _GstDataRepoSrc GstDataRepoSrc
```

Definition at line 35 of file `gstdatareposrc.h`.

9.6.3.2 GstDataRepoSrcClass

```
typedef struct _GstDataRepoSrcClass GstDataRepoSrcClass
```

Definition at line 36 of file `gstdatareposrc.h`.

9.6.4 Function Documentation

9.6.4.1 `gst_data_repo_src_get_type()`

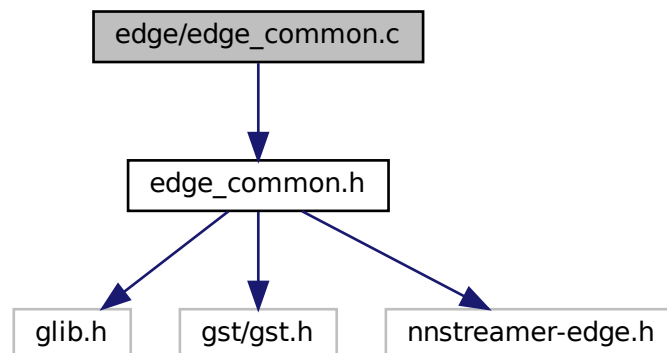
```
GType gst_data_repo_src_get_type (  
    void )
```

9.7 `edge/edge_common.c` File Reference

Common functions for edge sink and src.

```
#include "edge_common.h"
```

Include dependency graph for `edge_common.c`:



Functions

- GType [gst_edge_get_connect_type](#) (void)
register GEnumValue array for edge protocol property handling

9.7.1 Detailed Description

Common functions for edge sink and src.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

01 Aug 2022

Author

Yechan Choi yechan9.choi@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.7.2 Function Documentation

9.7.2.1 [gst_edge_get_connect_type\(\)](#)

```
GType gst_edge_get_connect_type (  
    void )
```

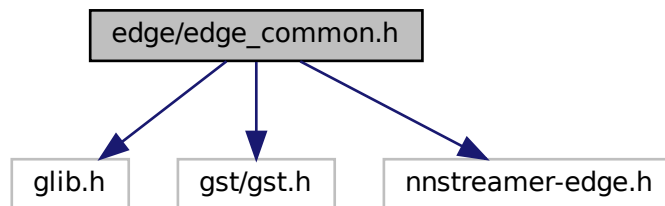
register GEnumValue array for edge protocol property handling

Definition at line 23 of file edge_common.c.

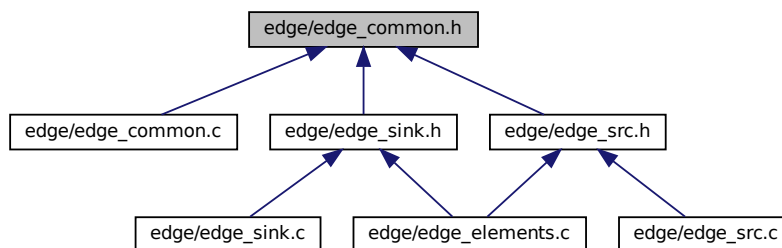
9.8 edge/edge_common.h File Reference

Common functions for edge sink and src.

```
#include <glib.h>
#include <gst/gst.h>
#include <nnstreamer-edge.h>
Include dependency graph for edge_common.h:
```



This graph shows which files directly or indirectly include this file:



Macros

- #define [GST_EDGE_PACKAGE](#) "GStreamer Edge Plugins"
- #define [GST_EDGE_ELEM_NAME_SINK](#) "edgesink"
- #define [GST_EDGE_ELEM_NAME_SRC](#) "edgesrc"
- #define [DEFAULT_HOST](#) "localhost"
- #define [DEFAULT_PORT](#) 3000
- #define [DEFAULT_CONNECT_TYPE](#) (NNS_EDGE_CONNECT_TYPE_TCP)
- #define [GST_TYPE_EDGE_CONNECT_TYPE](#) ([gst_edge_get_connect_type](#) ())

Functions

- G_BEGIN_DECLS GType [gst_edge_get_connect_type](#) (void)
register GEnumValue array for edge protocol property handling

9.8.1 Detailed Description

Common functions for edge sink and src.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

01 Aug 2022

Author

Yechan Choi yechan9.choi@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.8.2 Macro Definition Documentation

9.8.2.1 DEFAULT_CONNECT_TYPE

```
#define DEFAULT_CONNECT_TYPE (NNS_EDGE_CONNECT_TYPE_TCP)
```

Definition at line 27 of file edge_common.h.

9.8.2.2 DEFAULT_HOST

```
#define DEFAULT_HOST "localhost"
```

Definition at line 25 of file edge_common.h.

9.8.2.3 DEFAULT_PORT

```
#define DEFAULT_PORT 3000
```

Definition at line 26 of file edge_common.h.

9.8.2.4 GST_EDGE_ELEM_NAME_SINK

```
#define GST_EDGE_ELEM_NAME_SINK "edgesink"
```

Definition at line 23 of file edge_common.h.

9.8.2.5 GST_EDGE_ELEM_NAME_SRC

```
#define GST_EDGE_ELEM_NAME_SRC "edgesrc"
```

Definition at line 24 of file edge_common.h.

9.8.2.6 GST_EDGE_PACKAGE

```
#define GST_EDGE_PACKAGE "GStreamer Edge Plugins"
```

Definition at line 21 of file edge_common.h.

9.8.2.7 GST_TYPE_EDGE_CONNECT_TYPE

```
#define GST_TYPE_EDGE_CONNECT_TYPE (gst_edge_get_connect_type ())
```

Definition at line 28 of file edge_common.h.

9.8.3 Function Documentation

9.8.3.1 gst_edge_get_connect_type()

```
G_BEGIN_DECLS GType gst_edge_get_connect_type (  
    void )
```

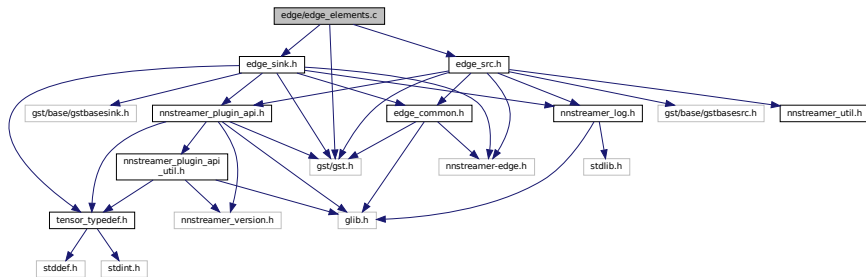
register GEnumValue array for edge protocol property handling

Definition at line 23 of file edge_common.c.

9.9 edge/edge_elements.c File Reference

Register edge plugins.

```
#include <gst/gst.h>
#include "edge_sink.h"
#include "edge_src.h"
Include dependency graph for edge_elements.c:
```



Macros

- #define `PACKAGE_GST_EDGE_PACKAGE`

Functions

- static gboolean `plugin_init` (GstPlugin *plugin)
The entry point of the Gstreamer Edge plugin.

9.9.1 Detailed Description

Register edge plugins.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

02 Aug 2022

Author

Yechan Choi yechan9.choi@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.9.2 Macro Definition Documentation

9.9.2.1 PACKAGE

```
#define PACKAGE GST_EDGE_PACKAGE
```

Definition at line 38 of file edge_elements.c.

9.9.3 Function Documentation

9.9.3.1 plugin_init()

```
static gboolean plugin_init (
    GstPlugin * plugin ) [static]
```

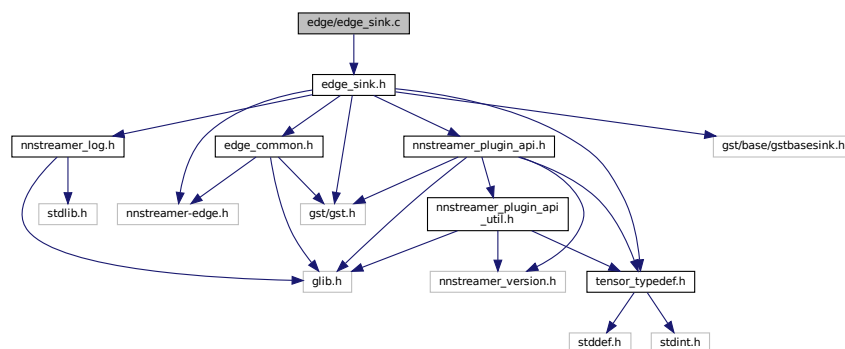
The entry point of the Gstreamer Edge plugin.

Definition at line 22 of file edge_elements.c.

9.10 edge/edge_sink.c File Reference

Publish incoming streams.

```
#include "edge_sink.h"
Include dependency graph for edge_sink.c:
```



Macros

- #define [GST_CAT_DEFAULT](#) `gst_edgesink_debug`
- #define [DEFAULT_MQTT_HOST](#) `"127.0.0.1"`
- #define [DEFAULT_MQTT_PORT](#) `1883`
- #define [gst_edgesink_parent_class](#) `parent_class`

Enumerations

- enum {
[PROP_0](#), [PROP_HOST](#), [PROP_PORT](#), [PROP_DEST_HOST](#),
[PROP_DEST_PORT](#), [PROP_CONNECT_TYPE](#), [PROP_TOPIC](#), [PROP_WAIT_CONNECTION](#),
[PROP_CONNECTION_TIMEOUT](#), [PROP_CUSTOM_LIB](#), [PROP_LAST](#) }
edgesink properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) (`gst_edgesink_debug`)
- [G_DEFINE_TYPE](#) (`GstEdgeSink`, `gst_edgesink`, `GST_TYPE_BASE_SINK`)
- static void [gst_edgesink_set_property](#) (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
set property
- static void [gst_edgesink_get_property](#) (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)
get property
- static void [gst_edgesink_finalize](#) (`GObject *object`)
finalize the object
- static gboolean [gst_edgesink_start](#) (`GstBaseSink *basesink`)
start processing of edgesink
- static gboolean [gst_edgesink_stop](#) (`GstBaseSink *basesink`)
Stop processing of edgesink.
- static `GstFlowReturn` [gst_edgesink_render](#) (`GstBaseSink *basesink`, `GstBuffer *buffer`)
render buffer, send buffer
- static gboolean [gst_edgesink_set_caps](#) (`GstBaseSink *basesink`, `GstCaps *caps`)
An implementation of the set_caps vmethod in GstBaseSinkClass.
- static `gchar *` [gst_edgesink_get_host](#) (`GstEdgeSink *self`)
getter for the 'host' property.
- static void [gst_edgesink_set_host](#) (`GstEdgeSink *self`, `const gchar *host`)
setter for the 'host' property.
- static `guint16` [gst_edgesink_get_port](#) (`GstEdgeSink *self`)
getter for the 'port' property.
- static void [gst_edgesink_set_port](#) (`GstEdgeSink *self`, `const guint16 port`)
setter for the 'port' property.
- static `nns_edge_connect_type_e` [gst_edgesink_get_connect_type](#) (`GstEdgeSink *self`)
getter for the 'connect_type' property.
- static void [gst_edgesink_set_connect_type](#) (`GstEdgeSink *self`, `const nns_edge_connect_type_e connect_type`)
setter for the 'connect_type' property.
- static void [gst_edgesink_class_init](#) (`GstEdgeSinkClass *klass`)
initialize the class
- static void [gst_edgesink_init](#) (`GstEdgeSink *self`)
initialize the new element
- static `int` [_nns_edge_event_cb](#) (`nns_edge_event_h event_h`, `void *user_data`)
nnstreamer-edge event callback.
- static gboolean [_wait_connection](#) (`GstEdgeSink *sink`)
If wait-connection is enabled, wait for connection until the connection is established or timeout occurs. Otherwise, return immediately.

Variables

- static GstStaticPadTemplate [sinktemplate](#)
the capabilities of the inputs.

9.10.1 Detailed Description

Publish incoming streams.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

01 Aug 2022

Author

Yechan Choi yechan9.choi@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.10.2 Macro Definition Documentation

9.10.2.1 DEFAULT_MQTT_HOST

```
#define DEFAULT_MQTT_HOST "127.0.0.1"
```

Definition at line 49 of file `edge_sink.c`.

9.10.2.2 DEFAULT_MQTT_PORT

```
#define DEFAULT_MQTT_PORT 1883
```

Definition at line 50 of file `edge_sink.c`.

9.10.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_edgesink_debug
```

Definition at line 20 of file edge_sink.c.

9.10.2.4 gst_edgesink_parent_class

```
#define gst_edgesink_parent_class parent_class
```

Definition at line 52 of file edge_sink.c.

9.10.3 Enumeration Type Documentation

9.10.3.1 anonymous enum

anonymous enum

edgesink properties

Enumerator

PROP_0	
PROP_HOST	
PROP_PORT	
PROP_DEST_HOST	
PROP_DEST_PORT	
PROP_CONNECT_TYPE	
PROP_TOPIC	
PROP_WAIT_CONNECTION	
PROP_CONNECTION_TIMEOUT	
PROP_CUSTOM_LIB	
PROP_LAST	

Definition at line 33 of file edge_sink.c.

9.10.4 Function Documentation

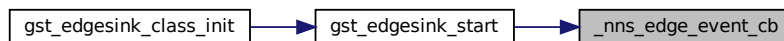
9.10.4.1 `_nns_edge_event_cb()`

```
static int _nns_edge_event_cb (
    nns_edge_event_h event_h,
    void * user_data ) [static]
```

nnstreamer-edge event callback.

Definition at line 306 of file `edge_sink.c`.

Here is the caller graph for this function:



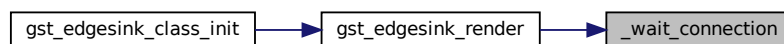
9.10.4.2 `_wait_connection()`

```
static gboolean _wait_connection (
    GstEdgeSink * sink ) [static]
```

If wait-connection is enabled, wait for connection until the connection is established or timeout occurs. Otherwise, return immediately.

Definition at line 400 of file `edge_sink.c`.

Here is the caller graph for this function:



9.10.4.3 `G_DEFINE_TYPE()`

```
G_DEFINE_TYPE (
    GstEdgeSink ,
    gst_edgesink ,
    GST_TYPE_BASE_SINK )
```

9.10.4.4 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_edgesink_debug )
```

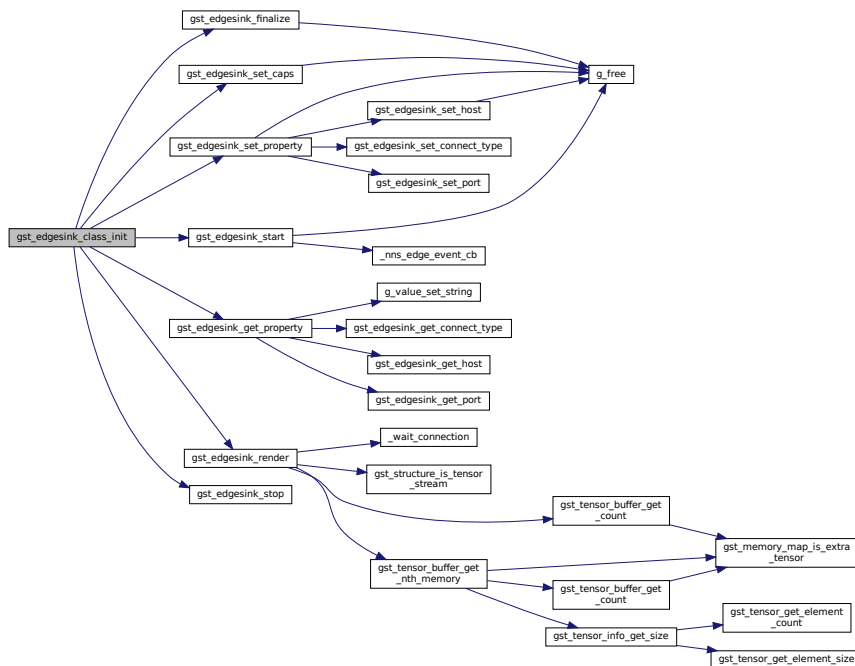
9.10.4.5 gst_edgesink_class_init()

```
static void gst_edgesink_class_init (
    GstEdgeSinkClass * klass ) [static]
```

initialize the class

Definition at line 84 of file edge_sink.c.

Here is the call graph for this function:



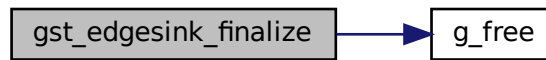
9.10.4.6 gst_edgesink_finalize()

```
static void gst_edgesink_finalize (
    GObject * object ) [static]
```

finalize the object

Definition at line 275 of file edge_sink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



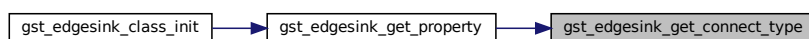
9.10.4.7 `gst_edgesink_get_connect_type()`

```
static nns_edge_connect_type_e gst_edgesink_get_connect_type (  
    GstEdgeSink * self ) [static]
```

getter for the 'connect_type' property.

Definition at line 589 of file `edge_sink.c`.

Here is the caller graph for this function:



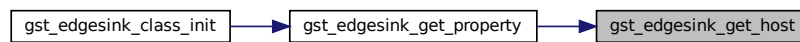
9.10.4.8 `gst_edgesink_get_host()`

```
static gchar * gst_edgesink_get_host (  
    GstEdgeSink * self ) [static]
```

getter for the 'host' property.

Definition at line 551 of file `edge_sink.c`.

Here is the caller graph for this function:



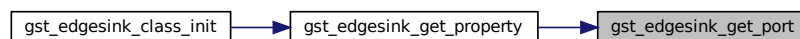
9.10.4.9 `gst_edgesink_get_port()`

```
static guint16 gst_edgesink_get_port (  
    GstEdgeSink * self ) [static]
```

getter for the 'port' property.

Definition at line 571 of file `edge_sink.c`.

Here is the caller graph for this function:



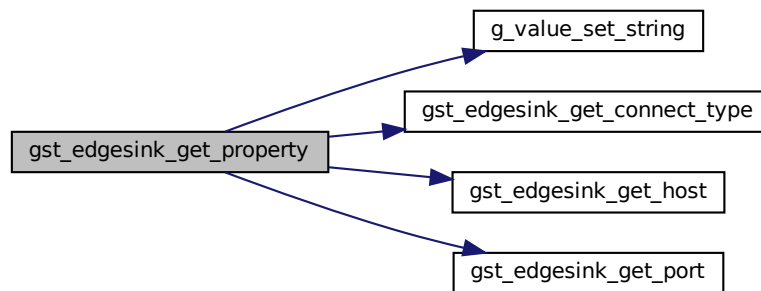
9.10.4.10 `gst_edgesink_get_property()`

```
static void gst_edgesink_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

get property

Definition at line 232 of file `edge_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.10.4.11 `gst_edgesink_init()`

```
static void gst_edgesink_init (
    GstEdgeSink * self ) [static]
```

initialize the new element

Definition at line 158 of file `edge_sink.c`.

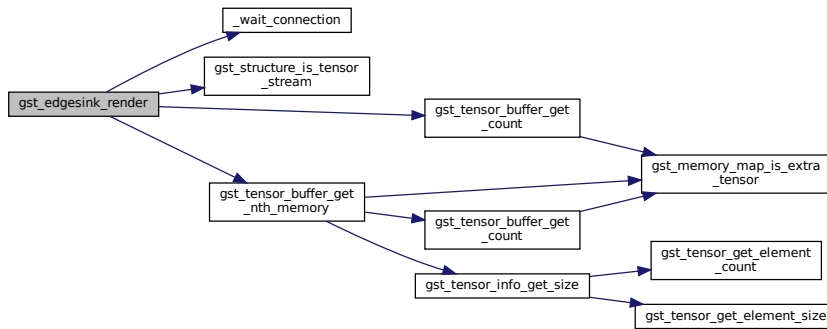
9.10.4.12 `gst_edgesink_render()`

```
static GstFlowReturn gst_edgesink_render (
    GstBaseSink * basesink,
    GstBuffer * buffer ) [static]
```

render buffer, send buffer

Definition at line 450 of file `edge_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.10.4.13 gst_edgesink_set_caps()

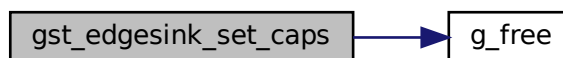
```

static gboolean gst_edgesink_set_caps (
    GstBaseSink * basesink,
    GstCaps * caps ) [static]
  
```

An implementation of the set_caps vmethod in GstBaseSinkClass.

Definition at line 524 of file edge_sink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.10.4.14 `gst_edgesink_set_connect_type()`

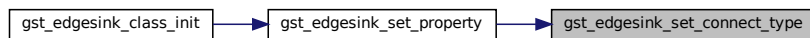
```

static void gst_edgesink_set_connect_type (
    GstEdgeSink * self,
    const nns_edge_connect_type_e connect_type ) [static]
  
```

setter for the 'connect_type' property.

Definition at line 598 of file `edge_sink.c`.

Here is the caller graph for this function:



9.10.4.15 `gst_edgesink_set_host()`

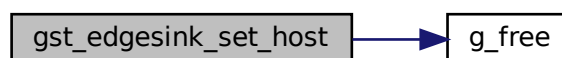
```

static void gst_edgesink_set_host (
    GstEdgeSink * self,
    const gchar * host ) [static]
  
```

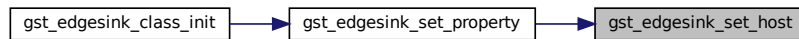
setter for the 'host' property.

Definition at line 560 of file `edge_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.10.4.16 `gst_edgesink_set_port()`

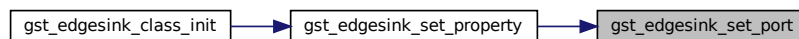
```

static void gst_edgesink_set_port (
    GstEdgeSink * self,
    const guint16 port ) [static]
  
```

setter for the 'port' property.

Definition at line 580 of file `edge_sink.c`.

Here is the caller graph for this function:



9.10.4.17 `gst_edgesink_set_property()`

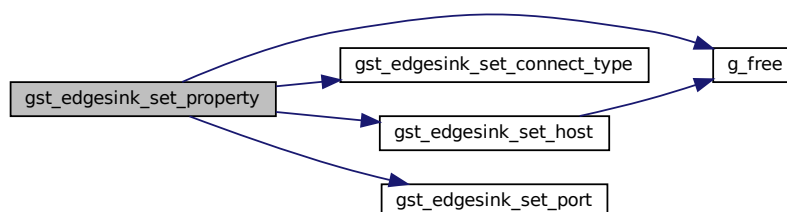
```

static void gst_edgesink_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

set property

Definition at line 178 of file `edge_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



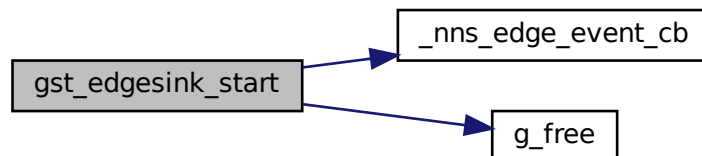
9.10.4.18 `gst_edgesink_start()`

```
static gboolean gst_edgesink_start (  
    GstBaseSink * basesink ) [static]
```

start processing of edgesink

Definition at line 338 of file `edge_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.10.4.19 `gst_edgesink_stop()`

```
static gboolean gst_edgesink_stop (  
    GstBaseSink * basesink ) [static]
```

Stop processing of edgesink.

Definition at line 432 of file `edge_sink.c`.

Here is the caller graph for this function:



9.10.5 Variable Documentation

9.10.5.1 `sinktemplate`

```
GstStaticPadTemplate sinktemplate [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS_ANY)
```

the capabilities of the inputs.

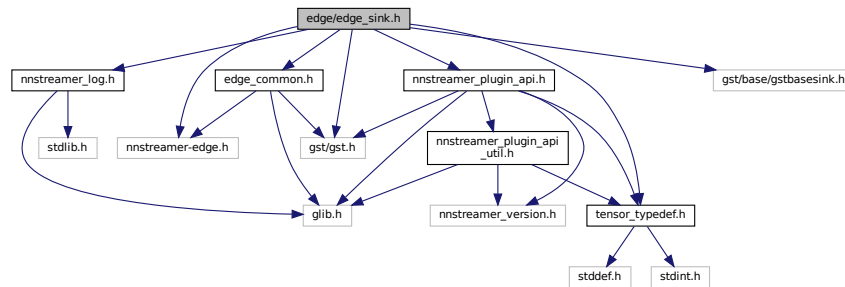
Definition at line 25 of file `edge_sink.c`.

9.11 edge/edge_sink.h File Reference

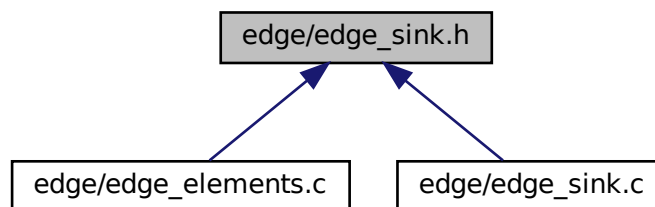
Publish incoming streams.

```
#include <gst/gst.h>  
#include <gst/base/gstbasesink.h>  
#include "edge_common.h"  
#include <nnstreamer-edge.h>  
#include "nnstreamer_log.h"  
#include "nnstreamer_plugin_api.h"
```

```
#include "tensor_typedef.h"
Include dependency graph for edge_sink.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstEdgeSink](#)
GstEdgeSink data structure.
- struct [_GstEdgeSinkClass](#)
GstEdgeSinkClass data structure.

Macros

- #define [GST_TYPE_EDGESINK](#) ([gst_edgesink_get_type\(\)](#))
- #define [GST_EDGESINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_EDGESINK, GstEdgeSink\)](#))
- #define [GST_EDGESINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_EDGESINK, GstEdgeSinkClass\)](#))
- #define [GST_IS_EDGESINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_EDGESINK\)](#))
- #define [GST_IS_EDGESINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_EDGESINK\)](#))
- #define [GST_EDGESINK_CAST\(obj\)](#) ([\(\(GstEdgeSink *\) \(obj\)\)](#))

Typedefs

- typedef struct [_GstEdgeSink](#) [GstEdgeSink](#)
- typedef struct [_GstEdgeSinkClass](#) [GstEdgeSinkClass](#)

Functions

- GType [gst_edgesink_get_type](#) (void)

9.11.1 Detailed Description

Publish incoming streams.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

01 Aug 2022

Author

Yechan Choi yechan9.choi@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.11.2 Macro Definition Documentation

9.11.2.1 GST_EDGESINK

```
#define GST_EDGESINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_EDGESINK, GstEdgeSink))
```

Definition at line 27 of file edge_sink.h.

9.11.2.2 GST_EDGESINK_CAST

```
#define GST_EDGESINK_CAST(  
    obj ) ((GstEdgeSink *) (obj))
```

Definition at line 35 of file edge_sink.h.

9.11.2.3 GST_EDGESINK_CLASS

```
#define GST_EDGESINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_EDGESINK, GstEdgeSinkClass))
```

Definition at line 29 of file `edge_sink.h`.

9.11.2.4 GST_IS_EDGESINK

```
#define GST_IS_EDGESINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_EDGESINK))
```

Definition at line 31 of file `edge_sink.h`.

9.11.2.5 GST_IS_EDGESINK_CLASS

```
#define GST_IS_EDGESINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_EDGESINK))
```

Definition at line 33 of file `edge_sink.h`.

9.11.2.6 GST_TYPE_EDGESINK

```
#define GST_TYPE_EDGESINK (gst_edgesink_get_type())
```

Definition at line 25 of file `edge_sink.h`.

9.11.3 Typedef Documentation

9.11.3.1 GstEdgeSink

```
typedef struct _GstEdgeSink GstEdgeSink
```

Definition at line 36 of file `edge_sink.h`.

9.11.3.2 GstEdgeSinkClass

```
typedef struct _GstEdgeSinkClass GstEdgeSinkClass
```

Definition at line 37 of file edge_sink.h.

9.11.4 Function Documentation

9.11.4.1 gst_edgesink_get_type()

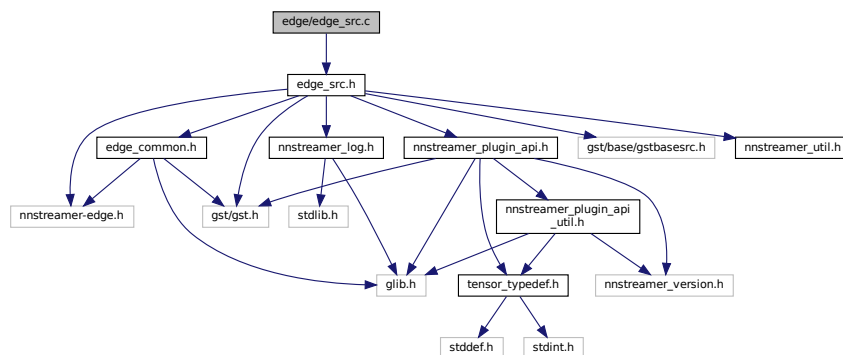
```
GType gst_edgesink_get_type (
    void )
```

9.12 edge/edge_src.c File Reference

Subscribe and push incoming data to the GStreamer pipeline.

```
#include "edge_src.h"
```

Include dependency graph for edge_src.c:



Macros

- #define `GST_CAT_DEFAULT` `gst_edgesrc_debug`
- #define `gst_edgesrc_parent_class` `parent_class`

Enumerations

- enum {
`PROP_0`, `PROP_HOST`, `PROP_PORT`, `PROP_DEST_HOST`,
`PROP_DEST_PORT`, `PROP_CONNECT_TYPE`, `PROP_TOPIC`, `PROP_CUSTOM_LIB`,
`PROP_LAST` }
edgesrc properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_edgesrc_debug](#))
- [G_DEFINE_TYPE](#) ([GstEdgeSrc](#), [gst_edgesrc](#), [GST_TYPE_BASE_SRC](#))
- static void [gst_edgesrc_set_property](#) ([GObject](#) *object, guint prop_id, const [GValue](#) *value, [GParamSpec](#) *pspec)
 - set property*
- static void [gst_edgesrc_get_property](#) ([GObject](#) *object, guint prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
 - get property*
- static void [gst_edgesrc_class_finalize](#) ([GObject](#) *object)
 - finalize the object*
- static gboolean [gst_edgesrc_start](#) ([GstBaseSrc](#) *basesrc)
 - start edgesrc, called when state changed null to ready*
- static gboolean [gst_edgesrc_stop](#) ([GstBaseSrc](#) *basesrc)
 - Stop edgesrc, called when state changed ready to null.*
- static [GstFlowReturn](#) [gst_edgesrc_create](#) ([GstBaseSrc](#) *basesrc, guint64 offset, guint size, [GstBuffer](#) **out_buf)
 - Create a buffer containing the subscribed data.*
- static gchar * [gst_edgesrc_get_dest_host](#) ([GstEdgeSrc](#) *self)
 - getter for the 'host' property.*
- static void [gst_edgesrc_set_dest_host](#) ([GstEdgeSrc](#) *self, const gchar *dest_host)
 - setter for the 'host' property.*
- static guint16 [gst_edgesrc_get_dest_port](#) ([GstEdgeSrc](#) *self)
 - getter for the 'port' property.*
- static void [gst_edgesrc_set_dest_port](#) ([GstEdgeSrc](#) *self, const guint16 dest_port)
 - setter for the 'port' property.*
- static [nns_edge_connect_type_e](#) [gst_edgesrc_get_connect_type](#) ([GstEdgeSrc](#) *self)
 - getter for the 'connect_type' property.*
- static void [gst_edgesrc_set_connect_type](#) ([GstEdgeSrc](#) *self, const [nns_edge_connect_type_e](#) connect_type)
 - setter for the 'connect_type' property.*
- static [GstStateChangeReturn](#) [gst_edgesrc_change_state](#) ([GstElement](#) *element, [GstStateChange](#) transition)
 - Change state of edgesrc.*
- static void [gst_edgesrc_class_init](#) ([GstEdgeSrcClass](#) *klass)
 - initialize the class*
- static void [gst_edgesrc_init](#) ([GstEdgeSrc](#) *self)
 - initialize edgesrc element*
- static int [_nns_edge_event_cb](#) ([nns_edge_event_h](#) event_h, void *user_data)
 - nnstreamer-edge event callback.*

Variables

- static [GstStaticPadTemplate](#) [srctemplate](#)
 - the capabilities of the outputs*

9.12.1 Detailed Description

Subscribe and push incoming data to the GStreamer pipeline.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

02 Aug 2022

Author

Yechan Choi yechan9.choi@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.12.2 Macro Definition Documentation

9.12.2.1 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_edgesrc_debug
```

Definition at line 20 of file edge_src.c.

9.12.2.2 gst_edgesrc_parent_class

```
#define gst_edgesrc_parent_class parent_class
```

Definition at line 45 of file edge_src.c.

9.12.3 Enumeration Type Documentation

9.12.3.1 anonymous enum

```
anonymous enum
```

edgesrc properties

Enumerator

PROP_0	
PROP_HOST	
PROP_PORT	
PROP_DEST_HOST	
PROP_DEST_PORT	
PROP_CONNECT_TYPE	
PROP_TOPIC	
PROP_CUSTOM_LIB	
PROP_LAST	

Definition at line 31 of file edge_src.c.

9.12.4 Function Documentation

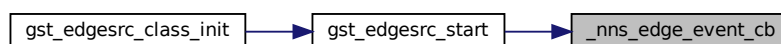
9.12.4.1 `_nns_edge_event_cb()`

```
static int _nns_edge_event_cb (
    nns_edge_event_h event_h,
    void * user_data ) [static]
```

nnstreamer-edge event callback.

Definition at line 305 of file edge_src.c.

Here is the caller graph for this function:



9.12.4.2 `G_DEFINE_TYPE()`

```
G_DEFINE_TYPE (
    GstEdgeSrc ,
    gst_edgesrc ,
    GST_TYPE_BASE_SRC )
```

9.12.4.3 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_edgesrc_debug )
```

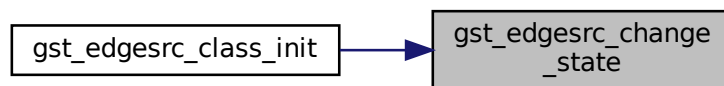
9.12.4.4 gst_edgesrc_change_state()

```
static GstStateChangeReturn gst_edgesrc_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

Change state of edgesrc.

Definition at line 273 of file edge_src.c.

Here is the caller graph for this function:



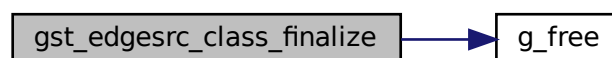
9.12.4.5 gst_edgesrc_class_finalize()

```
static void gst_edgesrc_class_finalize (
    GObject * object ) [static]
```

finalize the object

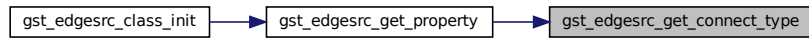
Definition at line 239 of file edge_src.c.

Here is the call graph for this function:



Definition at line 553 of file edge_src.c.

Here is the caller graph for this function:



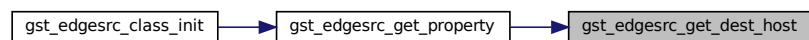
9.12.4.9 `gst_edgesrc_get_dest_host()`

```
static gchar * gst_edgesrc_get_dest_host (  
    GstEdgeSrc * self ) [static]
```

getter for the 'host' property.

Definition at line 516 of file edge_src.c.

Here is the caller graph for this function:



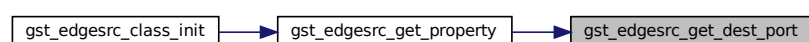
9.12.4.10 `gst_edgesrc_get_dest_port()`

```
static guint16 gst_edgesrc_get_dest_port (  
    GstEdgeSrc * self ) [static]
```

getter for the 'port' property.

Definition at line 535 of file edge_src.c.

Here is the caller graph for this function:



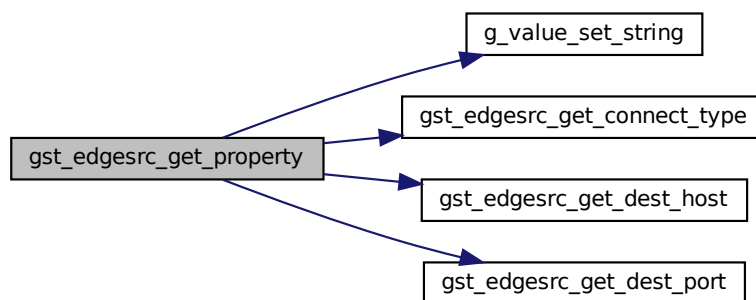
9.12.4.11 `gst_edgesrc_get_property()`

```
static void gst_edgesrc_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

get property

Definition at line 202 of file `edge_src.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.12.4.12 `gst_edgesrc_init()`

```
static void gst_edgesrc_init (  
    GstEdgeSrc * self ) [static]
```

initialize edgesrc element

Definition at line 139 of file `edge_src.c`.

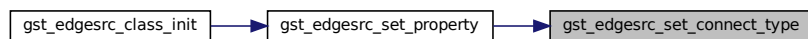
9.12.4.13 `gst_edgesrc_set_connect_type()`

```
static void gst_edgesrc_set_connect_type (  
    GstEdgeSrc * self,  
    const nns_edge_connect_type_e connect_type ) [static]
```

setter for the 'connect_type' property.

Definition at line 562 of file `edge_src.c`.

Here is the caller graph for this function:



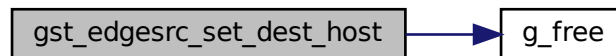
9.12.4.14 `gst_edgesrc_set_dest_host()`

```
static void gst_edgesrc_set_dest_host (  
    GstEdgeSrc * self,  
    const gchar * dest_host ) [static]
```

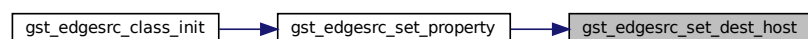
setter for the 'host' property.

Definition at line 525 of file `edge_src.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



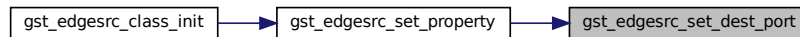
9.12.4.15 `gst_edgesrc_set_dest_port()`

```
static void gst_edgesrc_set_dest_port (
    GstEdgeSrc * self,
    const guint16 dest_port ) [static]
```

setter for the 'port' property.

Definition at line 544 of file `edge_src.c`.

Here is the caller graph for this function:



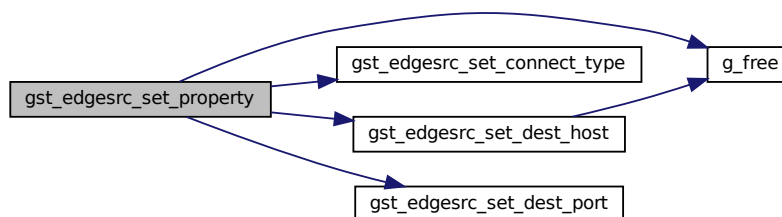
9.12.4.16 `gst_edgesrc_set_property()`

```
static void gst_edgesrc_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

set property

Definition at line 159 of file `edge_src.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



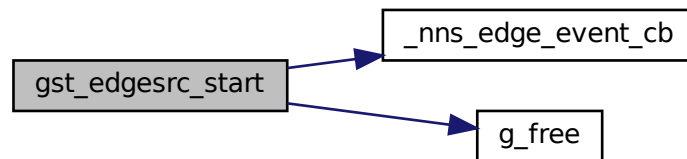
9.12.4.17 `gst_edgesrc_start()`

```
static gboolean gst_edgesrc_start (  
    GstBaseSrc * basesrc ) [static]
```

start edgesrc, called when state changed null to ready

Definition at line 342 of file `edge_src.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.12.4.18 `gst_edgesrc_stop()`

```
static gboolean gst_edgesrc_stop (  
    GstBaseSrc * basesrc ) [static]
```

Stop edgesrc, called when state changed ready to null.

Definition at line 403 of file `edge_src.c`.

Here is the caller graph for this function:



9.12.5 Variable Documentation

9.12.5.1 srctemplate

```
GstStaticPadTemplate srctemplate [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC, GST_PAD_ALWAYS, GST_STATIC_CAPS_ANY)
```

the capabilities of the outputs

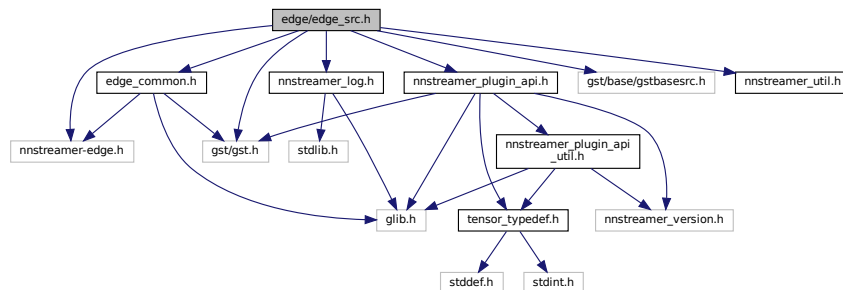
Definition at line 25 of file edge_src.c.

9.13 edge/edge_src.h File Reference

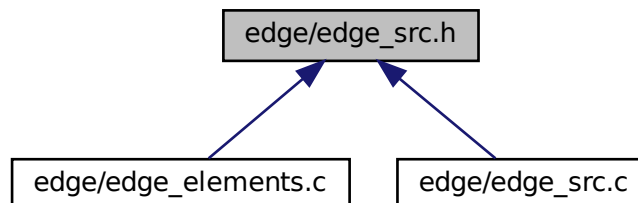
Subscribe and push incoming data to the GStreamer pipeline.

```
#include <gst/gst.h>
#include <gst/base/gstbasesrc.h>
#include "edge_common.h"
#include <nnstreamer-edge.h>
#include "nnstreamer_log.h"
#include "nnstreamer_plugin_api.h"
#include "nnstreamer_util.h"
```

Include dependency graph for edge_src.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstEdgeSrc](#)
GstEdgeSrc data structure.
- struct [_GstEdgeSrcClass](#)
GstEdgeSrcClass data structure.

Macros

- #define [GST_TYPE_EDGESRC](#) ([gst_edgesrc_get_type](#)())
- #define [GST_EDGESRC](#)(obj) (G_TYPE_CHECK_INSTANCE_CAST((obj), [GST_TYPE_EDGESRC](#), [GstEdgeSrc](#)))
- #define [GST_EDGESRC_CLASS](#)(klass) (G_TYPE_CHECK_CLASS_CAST((klass), [GST_TYPE_EDGESRC](#), [GstEdgeSrcClass](#)))
- #define [GST_IS_EDGESRC](#)(obj) (G_TYPE_CHECK_INSTANCE_TYPE((obj), [GST_TYPE_EDGESRC](#)))
- #define [GST_IS_EDGESRC_CLASS](#)(klass) (G_TYPE_CHECK_CLASS_TYPE((klass), [GST_TYPE_EDGESRC](#)))
- #define [GST_EDGESRC_CAST](#)(obj) (([GstEdgeSrc](#) *) (obj))

Typedefs

- typedef struct [_GstEdgeSrc](#) [GstEdgeSrc](#)
- typedef struct [_GstEdgeSrcClass](#) [GstEdgeSrcClass](#)

Functions

- GType [gst_edgesrc_get_type](#) (void)

9.13.1 Detailed Description

Subscribe and push incoming data to the GStreamer pipeline.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

02 Aug 2022

Author

Yechan Choi yechan9.choi@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.13.2 Macro Definition Documentation

9.13.2.1 GST_EDGESRC

```
#define GST_EDGESRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_EDGESRC, GstEdgeSrc))
```

Definition at line 27 of file edge_src.h.

9.13.2.2 GST_EDGESRC_CAST

```
#define GST_EDGESRC_CAST(  
    obj ) ((GstEdgeSrc *) (obj))
```

Definition at line 35 of file edge_src.h.

9.13.2.3 GST_EDGESRC_CLASS

```
#define GST_EDGESRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_EDGESRC, GstEdgeSrcClass))
```

Definition at line 29 of file edge_src.h.

9.13.2.4 GST_IS_EDGESRC

```
#define GST_IS_EDGESRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_EDGESRC))
```

Definition at line 31 of file edge_src.h.

9.13.2.5 GST_IS_EDGESRC_CLASS

```
#define GST_IS_EDGESRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_EDGESRC))
```

Definition at line 33 of file edge_src.h.

9.13.2.6 GST_TYPE_EDGESRC

```
#define GST_TYPE_EDGESRC (gst_edgesrc_get_type())
```

Definition at line 25 of file edge_src.h.

9.13.3 Typedef Documentation

9.13.3.1 GstEdgeSrc

```
typedef struct _GstEdgeSrc GstEdgeSrc
```

Definition at line 36 of file edge_src.h.

9.13.3.2 GstEdgeSrcClass

```
typedef struct _GstEdgeSrcClass GstEdgeSrcClass
```

Definition at line 37 of file edge_src.h.

9.13.4 Function Documentation

9.13.4.1 gst_edgesrc_get_type()

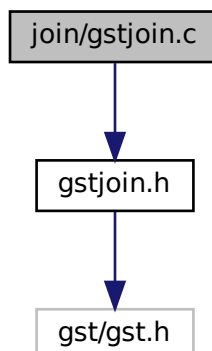
```
GType gst_edgesrc_get_type (  
    void )
```

9.14 join/gstjoin.c File Reference

Select the out that arrived first among the input streams.

```
#include "gstjoin.h"
```

Include dependency graph for gstjoin.c:



Classes

- struct [_GstJoinPad](#)
GstJoinPad data structure.
- struct [_GstJoinPadClass](#)
_GstJoinPadClass data structure

Macros

- #define [GST_CAT_DEFAULT](#) join_debug
- #define [GST_JOIN_GET_LOCK](#)(sel) (&((GstJoin*)(sel))->lock)
- #define [GST_JOIN_GET_COND](#)(sel) (&((GstJoin*)(sel))->cond)
- #define [GST_JOIN_LOCK](#)(sel) (g_mutex_lock ([GST_JOIN_GET_LOCK](#)(sel)))
- #define [GST_JOIN_UNLOCK](#)(sel) (g_mutex_unlock ([GST_JOIN_GET_LOCK](#)(sel)))
- #define [GST_JOIN_WAIT](#)(sel)
- #define [GST_TYPE_JOIN_PAD](#) (gst_join_pad_get_type())
- #define [GST_JOIN_PAD](#)(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj), [GST_TYPE_JOIN_PAD](#), [GstJoinPad](#)))
- #define [GST_JOIN_PAD_CLASS](#)(klass) (G_TYPE_CHECK_CLASS_CAST ((klass), [GST_TYPE_JOIN_PAD](#), [GstJoinPadClass](#)))
- #define [GST_IS_JOIN_PAD](#)(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), [GST_TYPE_JOIN_PAD](#)))
- #define [GST_IS_JOIN_PAD_CLASS](#)(klass) (G_TYPE_CHECK_CLASS_TYPE ((klass), [GST_TYPE_JOIN_PAD](#)))
- #define [GST_JOIN_PAD_CAST](#)(obj) (([GstJoinPad *](#))(obj))
- #define [gst_join_parent_class](#) parent_class
- #define [PACKAGE](#) "join"

Typedefs

- typedef struct [_GstJoinPad](#) [GstJoinPad](#)
- typedef struct [_GstJoinPadClass](#) [GstJoinPadClass](#)

Enumerations

- enum { [PROP_0](#), [PROP_N_PADS](#), [PROP_ACTIVE_PAD](#) }

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) (join_debug)
- static [GstPad *](#) [gst_join_get_active_sinkpad](#) ([GstJoin *](#)sel)
Get or create the active sinkpad.
- static [GstPad *](#) [gst_join_get_linked_pad](#) ([GstJoin *](#)sel, [GstPad *](#)pad, gboolean strict)
Get linked pad.
- static gboolean [gst_join_set_active_pad](#) ([GstJoin *](#)self, [GstPad *](#)pad)
set active sink pad.
- GType [gst_join_pad_get_type](#) (void)
- static void [gst_join_pad_finalize](#) (GObject *object)
finalize the join pad
- static void [gst_join_pad_reset](#) ([GstJoinPad *](#)pad)
Clear and reset join pad.
- static gboolean [gst_join_pad_event](#) ([GstPad *](#)pad, GObject *parent, [GstEvent *](#)event)

- event function for sink pad*
- static gboolean [gst_join_pad_query](#) (GstPad *pad, GstObject *parent, GstQuery *query)
handlesink sink pad query
- static GstIterator * [gst_join_pad_iterate_linked_pads](#) (GstPad *pad, GstObject *parent)
strictly get the linked pad from the sinkpad.
- static GstFlowReturn [gst_join_pad_chain](#) (GstPad *pad, GstObject *parent, GstBuffer *buf)
Chain function, this function does the actual processing.
- [G_DEFINE_TYPE](#) (GstJoinPad, gst_join_pad, GST_TYPE_PAD)
- static void [gst_join_pad_class_init](#) (GstJoinPadClass *klass)
initialize the join's pad class
- static void [gst_join_pad_init](#) (GstJoinPad *pad)
initialize the join pad
- static gboolean [forward_sticky_events](#) (GstPad *sinkpad, GstEvent **event, gpointer user_data)
forward sticky event
- static void [gst_join_dispose](#) (GObject *object)
dispose function for join element
- static void [gst_join_finalize](#) (GObject *object)
finalize join element.
- static void [gst_join_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
Getter for join properties.
- static GstPad * [gst_join_request_new_pad](#) (GstElement *element, GstPadTemplate *templ, const gchar *unused, const GstCaps *caps)
request new sink pad
- [G_DEFINE_TYPE_WITH_CODE](#) (GstJoin, gst_join, GST_TYPE_ELEMENT, GST_DEBUG_CATEGORY↔
_INIT(join_debug, "join", 0, "An input stream join element"))
- static void [gst_join_class_init](#) (GstJoinClass *klass)
initialize the join's class
- static void [gst_join_init](#) (GstJoin *sel)
initialize the join element
- static gboolean [plugin_init](#) (GstPlugin *plugin)
register this element

Variables

- static GstStaticPadTemplate [gst_join_sink_factory](#)
The capabilities of the inputs.
- static GstStaticPadTemplate [gst_join_src_factory](#)
The capabilities of the outputs.

9.14.1 Detailed Description

Select the out that arrived first among the input streams.

Copyright (C) 2020 Gichan Jang gichan2.jang@samsung.com

Date

10 Nov 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.14.2 Macro Definition Documentation

9.14.2.1 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT join_debug
```

Definition at line 35 of file gstjoin.c.

9.14.2.2 GST_IS_JOIN_PAD

```
#define GST_IS_JOIN_PAD(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_JOIN_PAD))
```

Definition at line 80 of file gstjoin.c.

9.14.2.3 GST_IS_JOIN_PAD_CLASS

```
#define GST_IS_JOIN_PAD_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_JOIN_PAD))
```

Definition at line 82 of file gstjoin.c.

9.14.2.4 GST_JOIN_GET_COND

```
#define GST_JOIN_GET_COND(  
    sel ) (&((GstJoin*)(sel))->cond)
```

Definition at line 38 of file gstjoin.c.

9.14.2.5 GST_JOIN_GET_LOCK

```
#define GST_JOIN_GET_LOCK(  
    sel ) (&((GstJoin*)(sel))->lock)
```

Definition at line 37 of file gstjoin.c.

9.14.2.6 GST_JOIN_LOCK

```
#define GST_JOIN_LOCK(  
    sel ) (g_mutex_lock (GST_JOIN_GET_LOCK(sel)))
```

Definition at line 39 of file gstjoin.c.

9.14.2.7 GST_JOIN_PAD

```
#define GST_JOIN_PAD(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_JOIN_PAD, GstJoinPad))
```

Definition at line 76 of file gstjoin.c.

9.14.2.8 GST_JOIN_PAD_CAST

```
#define GST_JOIN_PAD_CAST(  
    obj ) ((GstJoinPad *) (obj))
```

Definition at line 84 of file gstjoin.c.

9.14.2.9 GST_JOIN_PAD_CLASS

```
#define GST_JOIN_PAD_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_JOIN_PAD, GstJoinPadClass))
```

Definition at line 78 of file gstjoin.c.

9.14.2.10 gst_join_parent_class

```
#define gst_join_parent_class parent_class
```

Definition at line 428 of file gstjoin.c.

9.14.2.11 GST_JOIN_UNLOCK

```
#define GST_JOIN_UNLOCK(  
    sel ) (g_mutex_unlock (GST_JOIN_GET_LOCK(sel)))
```

Definition at line 40 of file gstjoin.c.

9.14.2.12 GST_JOIN_WAIT

```
#define GST_JOIN_WAIT(  
    sel )
```

Value:

```
(g_cond_wait (GST_JOIN_GET_COND(sel), \  
GST_JOIN_GET_LOCK(sel)))
```

Definition at line 41 of file gstjoin.c.

9.14.2.13 GST_TYPE_JOIN_PAD

```
#define GST_TYPE_JOIN_PAD (gst_join_pad_get_type())
```

Definition at line 74 of file gstjoin.c.

9.14.2.14 PACKAGE

```
#define PACKAGE "join"
```

Definition at line 701 of file gstjoin.c.

9.14.3 Typedef Documentation

9.14.3.1 GstJoinPad

```
typedef struct _GstJoinPad GstJoinPad
```

Definition at line 87 of file gstjoin.c.

9.14.3.2 GstJoinPadClass

```
typedef struct _GstJoinPadClass GstJoinPadClass
```

Definition at line 88 of file gstjoin.c.

9.14.4 Enumeration Type Documentation

9.14.4.1 anonymous enum

```
anonymous enum
```

Enumerator

PROP_0	
PROP_N_PADS	
PROP_ACTIVE_PAD	

Definition at line 62 of file gstjoin.c.

9.14.5 Function Documentation

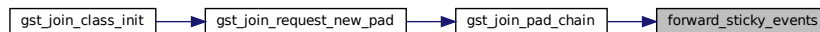
9.14.5.1 forward_sticky_events()

```
static gboolean forward_sticky_events (
    GstPad * sinkpad,
    GstEvent ** event,
    gpointer user_data ) [static]
```

forward sticky event

Definition at line 198 of file gstjoin.c.

Here is the caller graph for this function:



9.14.5.2 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstJoinPad ,
    gst_join_pad ,
    GST_TYPE_PAD )
```

9.14.5.3 G_DEFINE_TYPE_WITH_CODE()

```
G_DEFINE_TYPE_WITH_CODE (
    GstJoin ,
    gst_join ,
    GST_TYPE_ELEMENT ,
    GST_DEBUG_CATEGORY_INIT(join_debug, "join", 0, "An input stream join element") )
```

9.14.5.4 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    join_debug )
```

SECTION:element-join @see_also #GstInputSelector

Note

A join has reduced and changed input-selector's function.

Connect recently arrived buffer from N input streams to the output pad. N streams should not operate at the same time. All capabilities (input stream i and output stream) should be the same. For example, If one sinkpad is receiving buffer, the others should be stopped. <refsect2> <title>Example launch line</title> gst-launch-1.0 ... (input stream 0) ! join.sink_0 \ ... (input stream 1) ! join.sink_1 \ ... \ ... (input stream N) ! join.sink_n \ join name=join ! (arrived input stream) ... </refsect2>

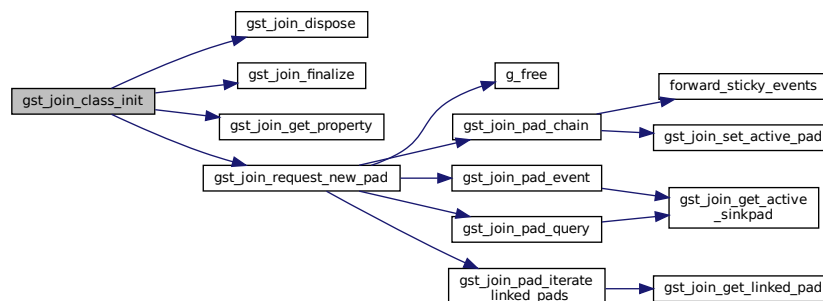
9.14.5.5 gst_join_class_init()

```
static void gst_join_class_init (
    GstJoinClass * klass ) [static]
```

initialize the join's class

Definition at line 437 of file gstjoin.c.

Here is the call graph for this function:



9.14.5.6 `gst_join_dispose()`

```
static void gst_join_dispose (  
    GObject * object ) [static]
```

dispose function for join element

Definition at line 494 of file `gstjoin.c`.

Here is the caller graph for this function:



9.14.5.7 `gst_join_finalize()`

```
static void gst_join_finalize (  
    GObject * object ) [static]
```

finalize join element.

Definition at line 509 of file `gstjoin.c`.

Here is the caller graph for this function:



9.14.5.8 `gst_join_get_active_sinkpad()`

```
static GstPad * gst_join_get_active_sinkpad (
    GstJoin * sel ) [static]
```

Get or create the active sinkpad.

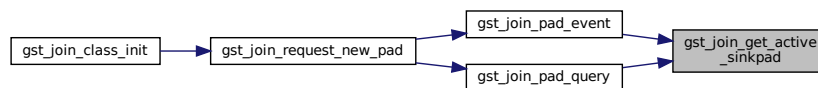
Note

must be called with JOIN_LOCK.

If no pad is currently selected, we return the first usable pad to guarantee consistency

Definition at line 612 of file `gstjoin.c`.

Here is the caller graph for this function:



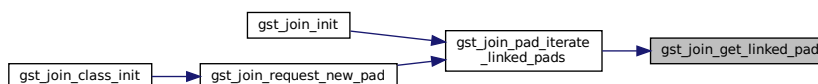
9.14.5.9 `gst_join_get_linked_pad()`

```
static GstPad * gst_join_get_linked_pad (
    GstJoin * sel,
    GstPad * pad,
    gboolean strict ) [static]
```

Get linked pad.

Definition at line 591 of file `gstjoin.c`.

Here is the caller graph for this function:



9.14.5.10 `gst_join_get_property()`

```
static void gst_join_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

Getter for join properties.

Definition at line 565 of file `gstjoin.c`.

Here is the caller graph for this function:



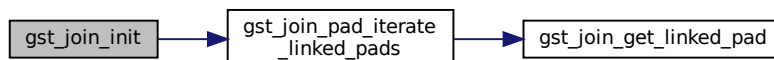
9.14.5.11 `gst_join_init()`

```
static void gst_join_init (
    GstJoin * sel ) [static]
```

initialize the join element

Definition at line 474 of file `gstjoin.c`.

Here is the call graph for this function:



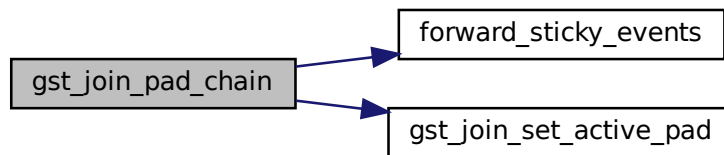
9.14.5.12 `gst_join_pad_chain()`

```
static GstFlowReturn gst_join_pad_chain (  
    GstPad * pad,  
    GstObject * parent,  
    GstBuffer * buf ) [static]
```

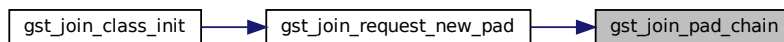
Chain function, this function does the actual processing.

Definition at line 357 of file `gstjoin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.14.5.13 `gst_join_pad_class_init()`

```
static void gst_join_pad_class_init (  
    GstJoinPadClass * klass ) [static]
```

initialize the join's pad class

Definition at line 129 of file `gstjoin.c`.

Here is the call graph for this function:



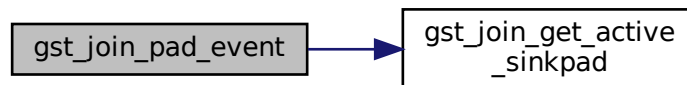
9.14.5.14 `gst_join_pad_event()`

```
static gboolean gst_join_pad_event (  
    GstPad * pad,  
    GstObject * parent,  
    GstEvent * event ) [static]
```

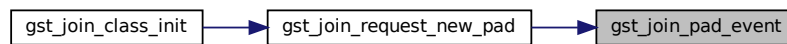
event function for sink pad

Definition at line 233 of file `gstjoin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.14.5.15 `gst_join_pad_finalize()`

```
static void gst_join_pad_finalize (  
    GObject * object ) [static]
```

finalize the join pad

Definition at line 151 of file `gstjoin.c`.

Here is the caller graph for this function:



9.14.5.16 `gst_join_pad_get_type()`

```
GType gst_join_pad_get_type (  
    void )
```

9.14.5.17 `gst_join_pad_init()`

```
static void gst_join_pad_init (  
    GstJoinPad * pad ) [static]
```

initialize the join pad

Definition at line 142 of file `gstjoin.c`.

Here is the call graph for this function:



9.14.5.18 `gst_join_pad_iterate_linked_pads()`

```
static GstIterator * gst_join_pad_iterate_linked_pads (  
    GstPad * pad,  
    GstObject * parent ) [static]
```

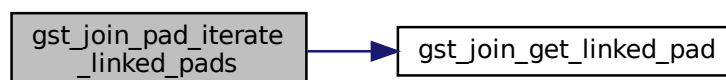
strictly get the linked pad from the sinkpad.

Returns

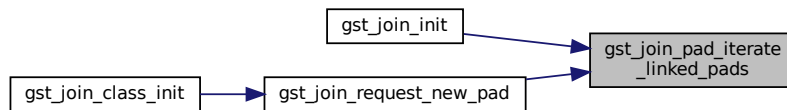
If the pad is active, return the srcpad else return NULL.

Definition at line 173 of file `gstjoin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.14.5.19 `gst_join_pad_query()`

```

static gboolean gst_join_pad_query (
    GstPad * pad,
    GstObject * parent,
    GstQuery * query ) [static]
  
```

handlesink sink pad query

Returns

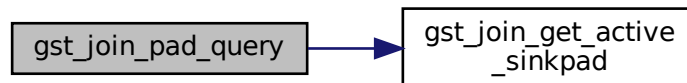
TRUE if the query was performed successfully.

always proxy caps/position/duration/context queries, regardless of active pad or not See https://bugzilla.gnome.org/show_bug.cgi?id=775445

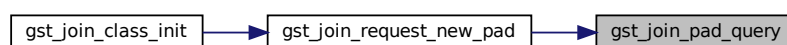
Only do the allocation query for the active sinkpad, after switching a reconfigure event is sent and upstream should reconfigure and do a new allocation query

Definition at line 307 of file `gstjoin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.14.5.20 `gst_join_pad_reset()`

```
static void gst_join_pad_reset (
    GstJoinPad * pad ) [static]
```

Clear and reset join pad.

Note

must be called with the `JOIN_LOCK`

Definition at line 161 of file `gstjoin.c`.

Here is the caller graph for this function:



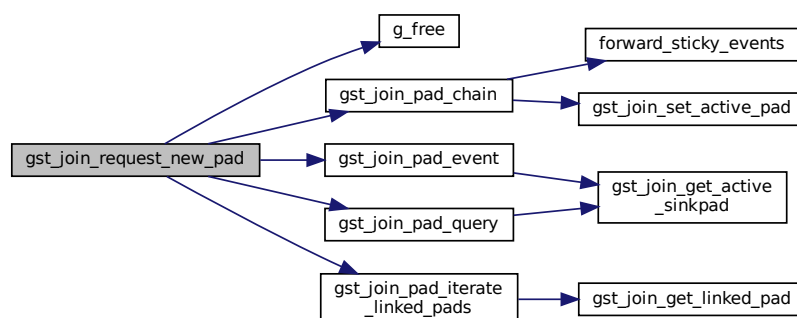
9.14.5.21 `gst_join_request_new_pad()`

```
static GstPad * gst_join_request_new_pad (
    GstElement * element,
    GstPadTemplate * templ,
    const gchar * unused,
    const GstCaps * caps ) [static]
```

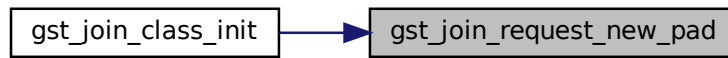
request new sink pad

Definition at line 646 of file `gstjoin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.14.5.22 `gst_join_set_active_pad()`

```

static gboolean gst_join_set_active_pad (
    GstJoin * self,
    GstPad * pad ) [static]
  
```

set active sink pad.

Returns

TRUE when the active pad changed.

Note

this function must be called with the JOIN_LOCK.

Definition at line 525 of file gstjoin.c.

Here is the caller graph for this function:



9.14.5.23 `plugin_init()`

```

static gboolean plugin_init (
    GstPlugin * plugin ) [static]
  
```

register this element

Definition at line 688 of file gstjoin.c.

9.14.6 Variable Documentation

9.14.6.1 gst_join_sink_factory

```
GstStaticPadTemplate gst_join_sink_factory [static]
```

Initial value:

```
=  
GST_STATIC_PAD_TEMPLATE ("sink_%u",  
    GST_PAD_SINK,  
    GST_PAD_REQUEST,  
    GST_STATIC_CAPS_ANY)
```

The capabilities of the inputs.

Definition at line 47 of file gstjoin.c.

9.14.6.2 gst_join_src_factory

```
GstStaticPadTemplate gst_join_src_factory [static]
```

Initial value:

```
=  
GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS_ANY)
```

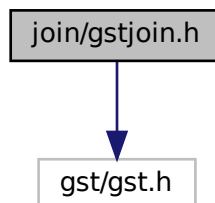
The capabilities of the outputs.

Definition at line 56 of file gstjoin.c.

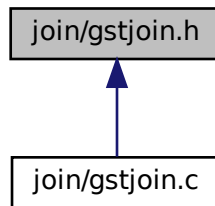
9.15 join/gstjoin.h File Reference

Select the out that arrived first among the input streams.

```
#include <gst/gst.h>  
Include dependency graph for gstjoin.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_GstJoin`](#)
Internal data structure for join instances.
- [struct `_GstJoinClass`](#)
GstJoinClass inherits GstElementClass.

Macros

- `#define GST_TYPE_JOIN (gst_join_get_type())`
- `#define GST_JOIN(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_JOIN, GstJoin))`
- `#define GST_JOIN_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_JOIN, GstJoinClass))`
- `#define GST_IS_JOIN(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_JOIN))`
- `#define GST_IS_JOIN_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_JOIN))`

Typedefs

- `typedef struct _GstJoin GstJoin`
- `typedef struct _GstJoinClass GstJoinClass`

Functions

- `G_GNUC_INTERNAL GType gst_join_get_type (void)`
Get Type function required for gst elements.

9.15.1 Detailed Description

Select the out that arrived first among the input streams.

Copyright (C) 2020 Gichan Jang gichan2.jang@samsung.com

Date

10 Nov 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.15.2 Macro Definition Documentation

9.15.2.1 GST_IS_JOIN

```
#define GST_IS_JOIN(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_JOIN))
```

Definition at line 26 of file `gstjoin.h`.

9.15.2.2 GST_IS_JOIN_CLASS

```
#define GST_IS_JOIN_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_JOIN))
```

Definition at line 28 of file `gstjoin.h`.

9.15.2.3 GST_JOIN

```
#define GST_JOIN(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_JOIN, GstJoin))
```

Definition at line 22 of file `gstjoin.h`.

9.15.2.4 GST_JOIN_CLASS

```
#define GST_JOIN_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_JOIN, GstJoinClass))
```

Definition at line 24 of file gstjoin.h.

9.15.2.5 GST_TYPE_JOIN

```
#define GST_TYPE_JOIN (gst_join_get_type())
```

Definition at line 20 of file gstjoin.h.

9.15.3 Typedef Documentation

9.15.3.1 GstJoin

```
typedef struct _GstJoin GstJoin
```

Definition at line 30 of file gstjoin.h.

9.15.3.2 GstJoinClass

```
typedef struct _GstJoinClass GstJoinClass
```

Definition at line 31 of file gstjoin.h.

9.15.4 Function Documentation

9.15.4.1 gst_join_get_type()

```
G_GNUC_INTERNAL GType gst_join_get_type (  
    void )
```

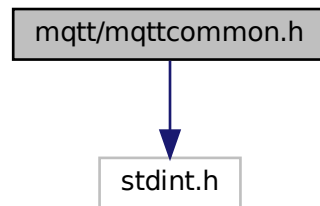
Get Type function required for gst elements.

9.16 mqtt/mqttcommon.h File Reference

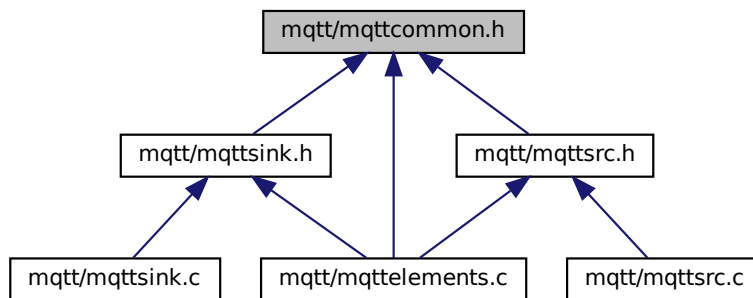
Common macros and utility functions for GStreamer MQTT plugins.

```
#include <stdint.h>
```

Include dependency graph for mqttcommon.h:



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_GstMQTTMessageHdr`](#)

Defined a custom data type, `GstMQTTMessageHdr`.

Macros

- `#define UNUSED(expr) do { (void)(expr); } while (0)`
- `#define GST_MQTT_PACKAGE "GStreamer MQTT Plugins"`
- `#define GST_MQTT_ELEM_NAME_SINK "mqttsink"`
- `#define GST_MQTT_ELEM_NAME_SRC "mqttsrc"`
- `#define GST_MQTT_LEN_MSG_HDR 1024`

- #define `GST_MQTT_MAX_LEN_GST_CAPS_STR` 512
- #define `GST_MQTT_MAX_NUM_MEMS` 16
GST_BUFFER_MEM_MAX in gstreamer/gstbuffer.c is 16. To represent each size of the memory block that the Gst↔ Buffer contains, GST_MQTT_MAX_NUM_MEMS should be 16.
- #define `GST_US_TO_NS_MULTIPLIER` 1000
- #define `DEFAULT_MQTT_CONN_TIMEOUT_SEC` 5

Typedefs

- typedef struct `_GstMQTTMessageHdr` `GstMQTTMessageHdr`
Defined a custom data type, GstMQTTMessageHdr.
- typedef int64_t(* `mqtt_get_unix_epoch`) (uint32_t, char **, uint16_t *)

Functions

- static int64_t `default_mqtt_get_unix_epoch` (uint32_t hnum, char **hnames, uint16_t *hports)
A wrapper function of `g_get_real_time()` to assign it to the function pointer, `mqtt_get_unix_epoch`.

9.16.1 Detailed Description

Common macros and utility functions for GStreamer MQTT plugins.

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

08 Mar 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.16.2 Macro Definition Documentation

9.16.2.1 DEFAULT_MQTT_CONN_TIMEOUT_SEC

```
#define DEFAULT_MQTT_CONN_TIMEOUT_SEC 5
```

Definition at line 40 of file `mqttcommon.h`.

9.16.2.2 GST_MQTT_ELEM_NAME_SINK

```
#define GST_MQTT_ELEM_NAME_SINK "mqttsink"
```

Definition at line 26 of file mqttcommon.h.

9.16.2.3 GST_MQTT_ELEM_NAME_SRC

```
#define GST_MQTT_ELEM_NAME_SRC "mqttsrc"
```

Definition at line 27 of file mqttcommon.h.

9.16.2.4 GST_MQTT_LEN_MSG_HDR

```
#define GST_MQTT_LEN_MSG_HDR 1024
```

Definition at line 29 of file mqttcommon.h.

9.16.2.5 GST_MQTT_MAX_LEN_GST_CAPS_STR

```
#define GST_MQTT_MAX_LEN_GST_CAPS_STR 512
```

Definition at line 30 of file mqttcommon.h.

9.16.2.6 GST_MQTT_MAX_NUM_MEMS

```
#define GST_MQTT_MAX_NUM_MEMS 16
```

GST_BUFFER_MEM_MAX in gstreamer/gstbuffer.c is 16. To represent each size of the memory block that the GstBuffer contains, GST_MQTT_MAX_NUM_MEMS should be 16.

Definition at line 36 of file mqttcommon.h.

9.16.2.7 GST_MQTT_PACKAGE

```
#define GST_MQTT_PACKAGE "GStreamer MQTT Plugins"
```

Definition at line 23 of file mqttcommon.h.

9.16.2.8 GST_US_TO_NS_MULTIPLIER

```
#define GST_US_TO_NS_MULTIPLIER 1000
```

Definition at line 38 of file mqttcommon.h.

9.16.2.9 UNUSED

```
#define UNUSED(  
    expr ) do { (void)(expr); } while (0)
```

Definition at line 19 of file mqttcommon.h.

9.16.3 Typedef Documentation

9.16.3.1 GstMQTTMessageHdr

```
typedef struct _GstMQTTMessageHdr GstMQTTMessageHdr
```

Defined a custom data type, GstMQTTMessageHdr.

GstMQTTMessageHdr contains the information needed to parse the message data at the subscriber side and is prepended to the original message data at the publisher side.

9.16.3.2 mqtt_get_unix_epoch

```
typedef int64_t(* mqtt_get_unix_epoch) (uint32_t, char **, uint16_t *)
```

Definition at line 65 of file mqttcommon.h.

9.16.4 Function Documentation

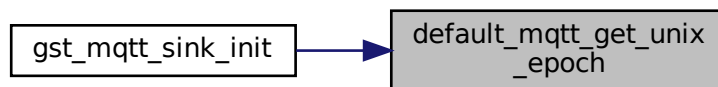
9.16.4.1 default_mqtt_get_unix_epoch()

```
static int64_t default_mqtt_get_unix_epoch (
    uint32_t hnum,
    char ** hnames,
    uint16_t * hports ) [inline], [static]
```

A wrapper function of `g_get_real_time ()` to assign it to the function pointer, `mqtt_get_unix_epoch`.

Definition at line 71 of file `mqttcommon.h`.

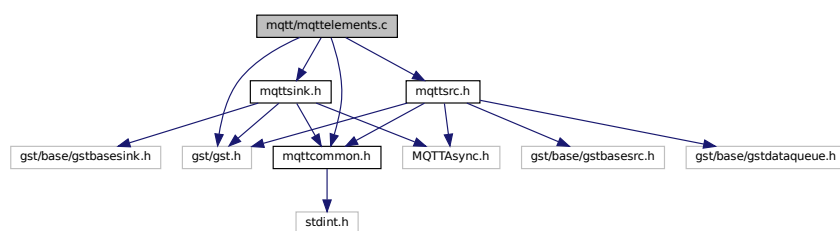
Here is the caller graph for this function:



9.17 mqtt/mqttelements.c File Reference

```
#include <gst/gst.h>
#include "mqttcommon.h"
#include "mqtttsink.h"
#include "mqtttsrc.h"
```

Include dependency graph for `mqttelements.c`:



Macros

- #define `PACKAGE_GST_MQTT_PACKAGE`

Functions

- static gboolean `plugin_init (GstPlugin *plugin)`
The entry point of the GStreamer MQTT plugin.

Macros

- #define [gst_mqtt_sink_parent_class](#) parent_class
- #define [GST_CAT_DEFAULT](#) gst_mqtt_sink_debug

Enumerations

- enum {
[PROP_0](#), [PROP_DEBUG](#), [PROP_MQTT_CLIENT_ID](#), [PROP_MQTT_HOST_ADDRESS](#),
[PROP_MQTT_HOST_PORT](#), [PROP_MQTT_PUB_TOPIC](#), [PROP_MQTT_PUB_WAIT_TIMEOUT](#), [PROP_MQTT_OPT_CLEANS](#),
[PROP_MQTT_OPT_KEEP_ALIVE_INTERVAL](#), [PROP_NUM_BUFFERS](#), [PROP_MAX_MSG_BUF_SIZE](#),
[PROP_MQTT_QOS](#),
[PROP_MQTT_NTP_SYNC](#), [PROP_MQTT_NTP_SRVS](#), [PROP_LAST](#) }
- enum {
[DEFAULT_DEBUG](#) = FALSE, [DEFAULT_NUM_BUFFERS](#) = -1, [DEFAULT_QOS](#) = TRUE, [DEFAULT_SYNC](#)
= FALSE,
[DEFAULT_MQTT_OPT_CLEANSSESSION](#) = TRUE, [DEFAULT_MQTT_OPT_KEEP_ALIVE_INTERVAL](#) =
60, [DEFAULT_MQTT_DISCONNECT_TIMEOUT](#) = G_TIME_SPAN_SECOND * 3, [DEFAULT_MQTT_PUB_WAIT_TIMEOUT](#)
= 1,
[DEFAULT_MAX_MSG_BUF_SIZE](#) = 0, [DEFAULT_MQTT_QOS](#) = 0, [DEFAULT_MQTT_NTP_SYNC](#) =
FALSE, [MAX_LEN_PROP_NTP_SRVS](#) = 4096 }

Functions

- [G_DEFINE_TYPE](#) (GstMqttSink, gst_mqtt_sink, GST_TYPE_BASE_SINK)
- [GST_DEBUG_CATEGORY_STATIC](#) (gst_mqtt_sink_debug)
- static void [gst_mqtt_sink_set_property](#) (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
The setter for the mqttsink's properties.
- static void [gst_mqtt_sink_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
The getter for the mqttsink's properties.
- static void [gst_mqtt_sink_class_finalize](#) (GObject *object)
Finalize GstMqttSinkClass object.
- static GstStateChangeReturn [gst_mqtt_sink_change_state](#) (GstElement *element, GstStateChange transition)
Handle mqttsink's state change.
- static gboolean [gst_mqtt_sink_start](#) (GstBaseSink *basesink)
Start mqttsink, called when state changed null to ready.
- static gboolean [gst_mqtt_sink_stop](#) (GstBaseSink *basesink)
Stop mqttsink, called when state changed ready to null.
- static gboolean [gst_mqtt_sink_query](#) (GstBaseSink *basesink, GstQuery *query)
Perform queries on the element.
- static GstFlowReturn [gst_mqtt_sink_render](#) (GstBaseSink *basesink, GstBuffer *in_buf)
The callback to process each buffer receiving on the sink pad.
- static GstFlowReturn [gst_mqtt_sink_render_list](#) (GstBaseSink *basesink, GstBufferList *list)
The callback to process GstBufferList (instead of a single buffer) on the sink pad.
- static gboolean [gst_mqtt_sink_event](#) (GstBaseSink *basesink, GstEvent *event)
Handle events arriving on the sink pad.
- static gboolean [gst_mqtt_sink_set_caps](#) (GstBaseSink *basesink, GstCaps *caps)
An implementation of the set_caps vmethod in GstBaseSinkClass.
- static gboolean [gst_mqtt_sink_get_debug](#) (GstMqttSink *self)

- Getter for the 'debug' property.*

 - static void `gst_mqtt_sink_set_debug` (`GstMqttSink *self`, const gboolean flag)

Setter for the 'debug' property.
- static gchar * `gst_mqtt_sink_get_client_id` (`GstMqttSink *self`)

Getter for the 'client-id' property.
- static void `gst_mqtt_sink_set_client_id` (`GstMqttSink *self`, const gchar *id)

Setter for the 'client-id' property.
- static gchar * `gst_mqtt_sink_get_host_address` (`GstMqttSink *self`)

Getter for the 'host' property.
- static void `gst_mqtt_sink_set_host_address` (`GstMqttSink *self`, const gchar *addr)

Setter for the 'host' property.
- static gchar * `gst_mqtt_sink_get_host_port` (`GstMqttSink *self`)

Getter for the 'port' property.
- static void `gst_mqtt_sink_set_host_port` (`GstMqttSink *self`, const gchar *port)

Setter for the 'port' property.
- static gchar * `gst_mqtt_sink_get_pub_topic` (`GstMqttSink *self`)

Getter for the 'pub-topic' property.
- static void `gst_mqtt_sink_set_pub_topic` (`GstMqttSink *self`, const gchar *topic)

Setter for the 'pub-topic' property.
- static gulong `gst_mqtt_sink_get_pub_wait_timeout` (`GstMqttSink *self`)

Getter for the 'pub-wait-timeout' property.
- static void `gst_mqtt_sink_set_pub_wait_timeout` (`GstMqttSink *self`, const gulong to)

Setter for the 'pub-wait-timeout' property.
- static gboolean `gst_mqtt_sink_get_opt_cleansession` (`GstMqttSink *self`)

Getter for the 'cleansession' property.
- static void `gst_mqtt_sink_set_opt_cleansession` (`GstMqttSink *self`, const gboolean val)

Setter for the 'cleansession' property.
- static gint `gst_mqtt_sink_get_opt_keep_alive_interval` (`GstMqttSink *self`)

Getter for the 'keep-alive-interval' property.
- static void `gst_mqtt_sink_set_opt_keep_alive_interval` (`GstMqttSink *self`, const gint num)

Setter for the 'keep-alive-interval' property.
- static gsize `gst_mqtt_sink_get_max_msg_buf_size` (`GstMqttSink *self`)

Getter for the 'max-buffer-size' property.
- static void `gst_mqtt_sink_set_max_msg_buf_size` (`GstMqttSink *self`, const gsize size)

Setter for the 'max-buffer-size' property.
- static gint `gst_mqtt_sink_get_num_buffers` (`GstMqttSink *self`)

Getter for the 'num-buffers' property.
- static void `gst_mqtt_sink_set_num_buffers` (`GstMqttSink *self`, const gint num)

Setter for the 'num-buffers' property.
- static gint `gst_mqtt_sink_get_mqtt_qos` (`GstMqttSink *self`)

Getter for the 'mqtt-qos' property.
- static void `gst_mqtt_sink_set_mqtt_qos` (`GstMqttSink *self`, const gint qos)

Setter for the 'mqtt-qos' property.
- static gboolean `gst_mqtt_sink_get_mqtt_ntp_sync` (`GstMqttSink *self`)

Getter for the 'ntp-sync' property.
- static void `gst_mqtt_sink_set_mqtt_ntp_sync` (`GstMqttSink *self`, const gboolean flag)

Setter for the 'ntp-sync' property.
- static gchar * `gst_mqtt_sink_get_mqtt_ntp_srvs` (`GstMqttSink *self`)

Getter for the 'ntp-srvs' property.
- static void `gst_mqtt_sink_set_mqtt_ntp_srvs` (`GstMqttSink *self`, const gchar *pairs)

Setter for the 'ntp-srvs' property.

- static void `cb_mqtt_on_connect` (void *context, MQTTAsync_successData *response)
A callback function corresponding to MQTTAsync_connectOptions's onSuccess. This callback is invoked when the connection between this element and the broker is properly established.
- static void `cb_mqtt_on_connect_failure` (void *context, MQTTAsync_failureData *response)
A callback function corresponding to MQTTAsync_connectOptions's onFailure. This callback is invoked when it is failed to connect to the broker.
- static void `cb_mqtt_on_disconnect` (void *context, MQTTAsync_successData *response)
A callback function corresponding to MQTTAsync_disconnectOptions's onSuccess. Regardless of the MQTTAsync→_disconnect function's result, the pipeline should be stopped after this callback.
- static void `cb_mqtt_on_disconnect_failure` (void *context, MQTTAsync_failureData *response)
A callback function corresponding to MQTTAsync_disconnectOptions's onFailure. Regardless of the MQTTAsync→_disconnect function's result, the pipeline should be stopped after this callback.
- static void `cb_mqtt_on_delivery_complete` (void *context, MQTTAsync_token token)
A callback function to be given to the MQTTAsync_setCallbacks function. This callback is activated when `mqtt_qos` is higher than 0.
- static void `cb_mqtt_on_connection_lost` (void *context, char *cause)
A callback function to be given to the MQTTAsync_setCallbacks function. When the connection between this element and the broker is broken, this callback will be invoked.
- static int `cb_mqtt_on_message_arrived` (void *context, char *topicName, int topicLen, MQTTAsync_message *message)
A callback function to be given to the MQTTAsync_setCallbacks function. In the case of the publisher, this callback is not used.
- static void `cb_mqtt_on_send_success` (void *context, MQTTAsync_successData *response)
A callback function corresponding to MQTTAsync_responseOptions's onSuccess.
- static void `cb_mqtt_on_send_failure` (void *context, MQTTAsync_failureData *response)
A callback function corresponding to MQTTAsync_responseOptions's onFailure.
- static void `gst_mqtt_sink_init` (GstMqttSink *self)
Initialize GstMqttSink object.
- static void `gst_mqtt_sink_class_init` (GstMqttSinkClass *klass)
Initialize GstMqttSinkClass object.
- static void `_put_timestamp_to_msg_buf_hdr` (GstMqttSink *self, GstBuffer *gst_buf, GstMQTTMessageHdr *hdr)
A utility function to set the timestamp information onto the given buffer.
- static gboolean `_mqtt_set_msg_buf_hdr` (GstBuffer *gst_buf, GstMQTTMessageHdr *hdr)
A utility function to set the message header.

Variables

- static GstStaticPadTemplate `sink_pad_template`
- static guint8 `sink_client_id` = 0
- static const gchar `DEFAULT_MQTT_HOST_ADDRESS` [] = "127.0.0.1"
- static const gchar `DEFAULT_MQTT_HOST_PORT` [] = "1883"
- static const gchar `TAG_ERR_MQTTSINK` [] = "ERROR: MQTTSink"
- static const gchar `DEFAULT_MQTT_CLIENT_ID` [] = "\$HOST_\$PID_[0-9][0-9]?\$|^255\$"
- static const gchar `DEFAULT_MQTT_CLIENT_ID_FORMAT` [] = "%s_%u_sink%u"
- static const gchar `DEFAULT_MQTT_PUB_TOPIC` [] = "\$client-id/topic"
- static const gchar `DEFAULT_MQTT_PUB_TOPIC_FORMAT` [] = "%s/topic"
- static const gchar `DEFAULT_MQTT_NTP_SERVERS` [] = "pool.ntp.org:123"

9.18.1 Detailed Description

Register sub-plugins included in libgstmqtt.

Publish incoming data streams as a MQTT topic.

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

09 Mar 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

01 Apr 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.18.2 Macro Definition Documentation

9.18.2.1 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_mqtt_sink_debug
```

Definition at line 42 of file mqttsink.c.

9.18.2.2 gst_mqtt_sink_parent_class

```
#define gst_mqtt_sink_parent_class parent_class
```

Definition at line 38 of file mqttsink.c.

9.18.3 Enumeration Type Documentation

9.18.3.1 anonymous enum

```
anonymous enum
```

Enumerator

DEFAULT_DEBUG	
DEFAULT_NUM_BUFFERS	
DEFAULT_QOS	
DEFAULT_SYNC	
DEFAULT_MQTT_OPT_CLEANSESSION	
DEFAULT_MQTT_OPT_KEEP_ALIVE_INTERVAL	
DEFAULT_MQTT_DISCONNECT_TIMEOUT	
DEFAULT_MQTT_PUB_WAIT_TIMEOUT	
DEFAULT_MAX_MSG_BUF_SIZE	
DEFAULT_MQTT_QOS	
DEFAULT_MQTT_NTP_SYNC	
MAX_LEN_PROP_NTP_SRVS	

Definition at line 65 of file mqttsink.c.

9.18.3.2 anonymous enum

anonymous enum

Enumerator

PROP_0	
PROP_DEBUG	
PROP_MQTT_CLIENT_ID	
PROP_MQTT_HOST_ADDRESS	
PROP_MQTT_HOST_PORT	
PROP_MQTT_PUB_TOPIC	
PROP_MQTT_PUB_WAIT_TIMEOUT	
PROP_MQTT_OPT_CLEANSESSION	
PROP_MQTT_OPT_KEEP_ALIVE_INTERVAL	
PROP_NUM_BUFFERS	
PROP_MAX_MSG_BUF_SIZE	
PROP_MQTT_QOS	
PROP_MQTT_NTP_SYNC	
PROP_MQTT_NTP_SRVS	
PROP_LAST	

Definition at line 44 of file mqttsink.c.

9.18.4 Function Documentation

9.18.4.1 `_mqtt_set_msg_buf_hdr()`

```
static gboolean _mqtt_set_msg_buf_hdr (
    GstBuffer * gst_buf,
    GstMQTTMessageHdr * hdr ) [static]
```

A utility function to set the message header.

Definition at line 725 of file `mqttpsink.c`.

Here is the caller graph for this function:



9.18.4.2 `_put_timestamp_to_msg_buf_hdr()`

```
static void _put_timestamp_to_msg_buf_hdr (
    GstMqttSink * self,
    GstBuffer * gst_buf,
    GstMQTTMessageHdr * hdr ) [static]
```

A utility function to set the timestamp information onto the given buffer.

Definition at line 688 of file `mqttpsink.c`.

Here is the caller graph for this function:



9.18.4.3 `cb_mqtt_on_connect()`

```
static void cb_mqtt_on_connect (
    void * context,
    MQTTAsync_successData * response ) [static]
```

A callback function corresponding to MQTTAsync_connectOptions's onSuccess. This callback is invoked when the connection between this element and the broker is properly established.

Callback function definitions

Definition at line 1267 of file mqttsink.c.

Here is the caller graph for this function:



9.18.4.4 `cb_mqtt_on_connect_failure()`

```
static void cb_mqtt_on_connect_failure (
    void * context,
    MQTTAsync_failureData * response ) [static]
```

A callback function corresponding to MQTTAsync_connectOptions's onFailure. This callback is invoked when it is failed to connect to the broker.

Definition at line 1285 of file mqttsink.c.

Here is the caller graph for this function:



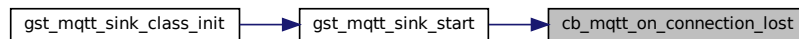
9.18.4.5 `cb_mqtt_on_connection_lost()`

```
static void cb_mqtt_on_connection_lost (
    void * context,
    char * cause ) [static]
```

A callback function to be given to the `MQTTAsync_setCallbacks` function. When the connection between this element and the broker is broken, this callback will be invoked.

Definition at line 1351 of file `mqttpsink.c`.

Here is the caller graph for this function:



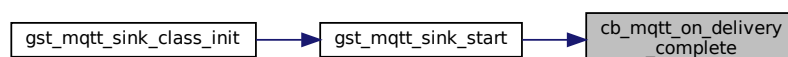
9.18.4.6 `cb_mqtt_on_delivery_complete()`

```
static void cb_mqtt_on_delivery_complete (
    void * context,
    MQTTAsync_token token ) [static]
```

A callback function to be given to the `MQTTAsync_setCallbacks` function. This callback is activated when `mqtt-qos` is higher than 0.

Definition at line 1336 of file `mqttpsink.c`.

Here is the caller graph for this function:



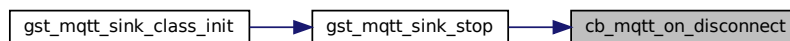
9.18.4.7 `cb_mqtt_on_disconnect()`

```
static void cb_mqtt_on_disconnect (
    void * context,
    MQTTAsync_successData * response ) [static]
```

A callback function corresponding to MQTTAsync_disconnectOptions's onSuccess. Regardless of the MQTTAsync_disconnect function's result, the pipeline should be stopped after this callback.

Definition at line 1303 of file mqttsink.c.

Here is the caller graph for this function:



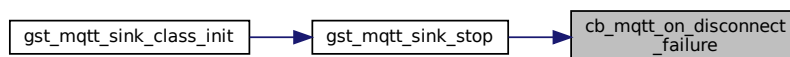
9.18.4.8 `cb_mqtt_on_disconnect_failure()`

```
static void cb_mqtt_on_disconnect_failure (
    void * context,
    MQTTAsync_failureData * response ) [static]
```

A callback function corresponding to MQTTAsync_disconnectOptions's onFailure. Regardless of the MQTTAsync_disconnect function's result, the pipeline should be stopped after this callback.

Definition at line 1320 of file mqttsink.c.

Here is the caller graph for this function:



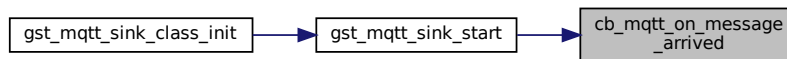
9.18.4.9 `cb_mqtt_on_message_arrived()`

```
static int cb_mqtt_on_message_arrived (  
    void * context,  
    char * topicName,  
    int topicLen,  
    MQTTAsync_message * message ) [static]
```

A callback function to be given to the `MQTTAsync_setCallbacks` function. In the case of the publisher, this callback is not used.

Definition at line 1368 of file `mqttpsink.c`.

Here is the caller graph for this function:



9.18.4.10 `cb_mqtt_on_send_failure()`

```
static void cb_mqtt_on_send_failure (  
    void * context,  
    MQTTAsync_failureData * response ) [static]
```

A callback function corresponding to `MQTTAsync_responseOptions`'s `onFailure`.

Definition at line 1404 of file `mqttpsink.c`.

Here is the caller graph for this function:



9.18.4.11 `cb_mqtt_on_send_success()`

```
static void cb_mqtt_on_send_success (
    void * context,
    MQTTAsync_successData * response ) [static]
```

A callback function corresponding to MQTTAsync_responseOptions's onSuccess.

Definition at line 1384 of file mqttpsink.c.

Here is the caller graph for this function:



9.18.4.12 `G_DEFINE_TYPE()`

```
G_DEFINE_TYPE (
    GstMqttSink ,
    gst_mqtt_sink ,
    GST_TYPE_BASE_SINK )
```

9.18.4.13 `GST_DEBUG_CATEGORY_STATIC()`

```
GST_DEBUG_CATEGORY_STATIC (
    gst_mqtt_sink_debug )
```

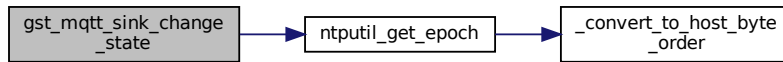
9.18.4.14 `gst_mqtt_sink_change_state()`

```
static GstStateChangeReturn gst_mqtt_sink_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

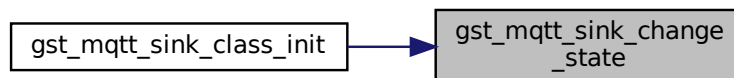
Handle mqttpsink's state change.

Definition at line 491 of file mqttpsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



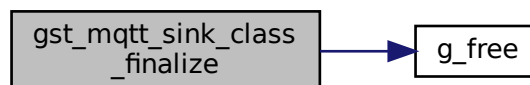
9.18.4.15 `gst_mqtt_sink_class_finalize()`

```
static void gst_mqtt_sink_class_finalize (
    GObject * object ) [static]
```

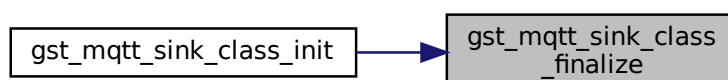
Finalize `GstMqttSinkClass` object.

Definition at line 454 of file `mqttsink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



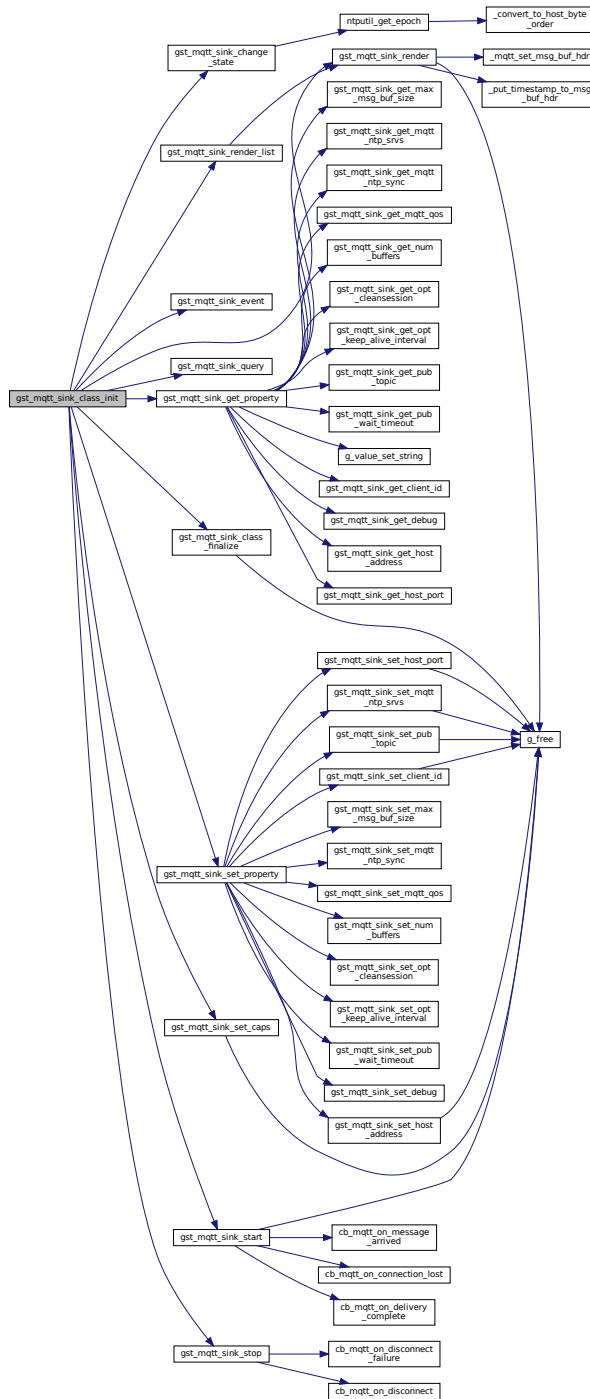
9.18.4.16 `gst_mqtt_sink_class_init()`

```
static void gst_mqtt_sink_class_init (
    GstMqttSinkClass * klass ) [static]
```

Initialize GstMqttSinkClass object.

Definition at line 229 of file mqttsink.c.

Here is the call graph for this function:



9.18.4.17 `gst_mqtt_sink_event()`

```
static gboolean gst_mqtt_sink_event (  
    GstBaseSink * basesink,  
    GstEvent * event ) [static]
```

Handle events arriving on the sink pad.

Definition at line 898 of file `mqttpsink.c`.

Here is the caller graph for this function:



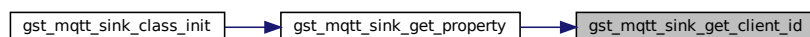
9.18.4.18 `gst_mqtt_sink_get_client_id()`

```
static gchar * gst_mqtt_sink_get_client_id (  
    GstMqttSink * self ) [static]
```

Getter for the 'client-id' property.

Definition at line 976 of file `mqttpsink.c`.

Here is the caller graph for this function:



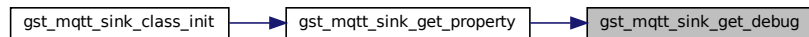
9.18.4.19 `gst_mqtt_sink_get_debug()`

```
static gboolean gst_mqtt_sink_get_debug (  
    GstMqttSink * self ) [static]
```

Getter for the 'debug' property.

Definition at line 958 of file mqttsink.c.

Here is the caller graph for this function:



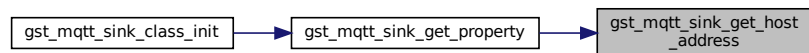
9.18.4.20 `gst_mqtt_sink_get_host_address()`

```
static gchar * gst_mqtt_sink_get_host_address (  
    GstMqttSink * self ) [static]
```

Getter for the 'host' property.

Definition at line 995 of file mqttsink.c.

Here is the caller graph for this function:



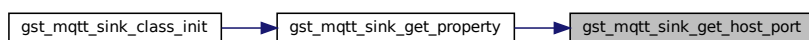
9.18.4.21 `gst_mqtt_sink_get_host_port()`

```
static gchar * gst_mqtt_sink_get_host_port (  
    GstMqttSink * self ) [static]
```

Getter for the 'port' property.

Definition at line 1017 of file mqttsink.c.

Here is the caller graph for this function:



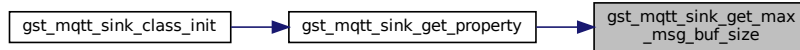
9.18.4.22 `gst_mqtt_sink_get_max_msg_buf_size()`

```
static gsize gst_mqtt_sink_get_max_msg_buf_size (
    GstMqttSink * self ) [static]
```

Getter for the 'max-buffer-size' property.

Definition at line 1109 of file `mqttpsink.c`.

Here is the caller graph for this function:



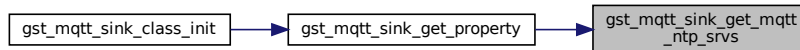
9.18.4.23 `gst_mqtt_sink_get_mqtt_ntp_srvs()`

```
static gchar * gst_mqtt_sink_get_mqtt_ntp_srvs (
    GstMqttSink * self ) [static]
```

Getter for the 'ntp-srvs' property.

Definition at line 1185 of file `mqttpsink.c`.

Here is the caller graph for this function:



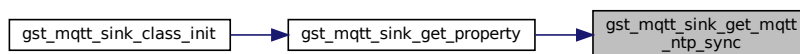
9.18.4.24 `gst_mqtt_sink_get_mqtt_ntp_sync()`

```
static gboolean gst_mqtt_sink_get_mqtt_ntp_sync (
    GstMqttSink * self ) [static]
```

Getter for the 'ntp-sync' property.

Definition at line 1167 of file `mqttpsink.c`.

Here is the caller graph for this function:



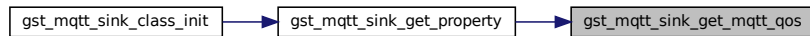
9.18.4.25 `gst_mqtt_sink_get_mqtt_qos()`

```
static gint gst_mqtt_sink_get_mqtt_qos (  
    GstMqttSink * self ) [static]
```

Getter for the 'mqtt-qos' property.

Definition at line 1149 of file mqttsink.c.

Here is the caller graph for this function:



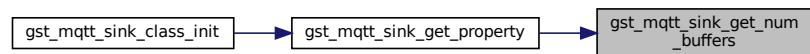
9.18.4.26 `gst_mqtt_sink_get_num_buffers()`

```
static gint gst_mqtt_sink_get_num_buffers (  
    GstMqttSink * self ) [static]
```

Getter for the 'num-buffers' property.

Definition at line 1127 of file mqttsink.c.

Here is the caller graph for this function:



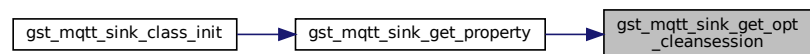
9.18.4.27 `gst_mqtt_sink_get_opt_cleansession()`

```
static gboolean gst_mqtt_sink_get_opt_cleansession (  
    GstMqttSink * self ) [static]
```

Getter for the 'cleansession' property.

Definition at line 1055 of file mqttsink.c.

Here is the caller graph for this function:



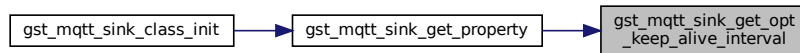
9.18.4.28 `gst_mqtt_sink_get_opt_keep_alive_interval()`

```
static gint gst_mqtt_sink_get_opt_keep_alive_interval (  
    GstMqttSink * self ) [static]
```

Getter for the 'keep-alive-interval' property.

Definition at line 1091 of file `mqttpsink.c`.

Here is the caller graph for this function:



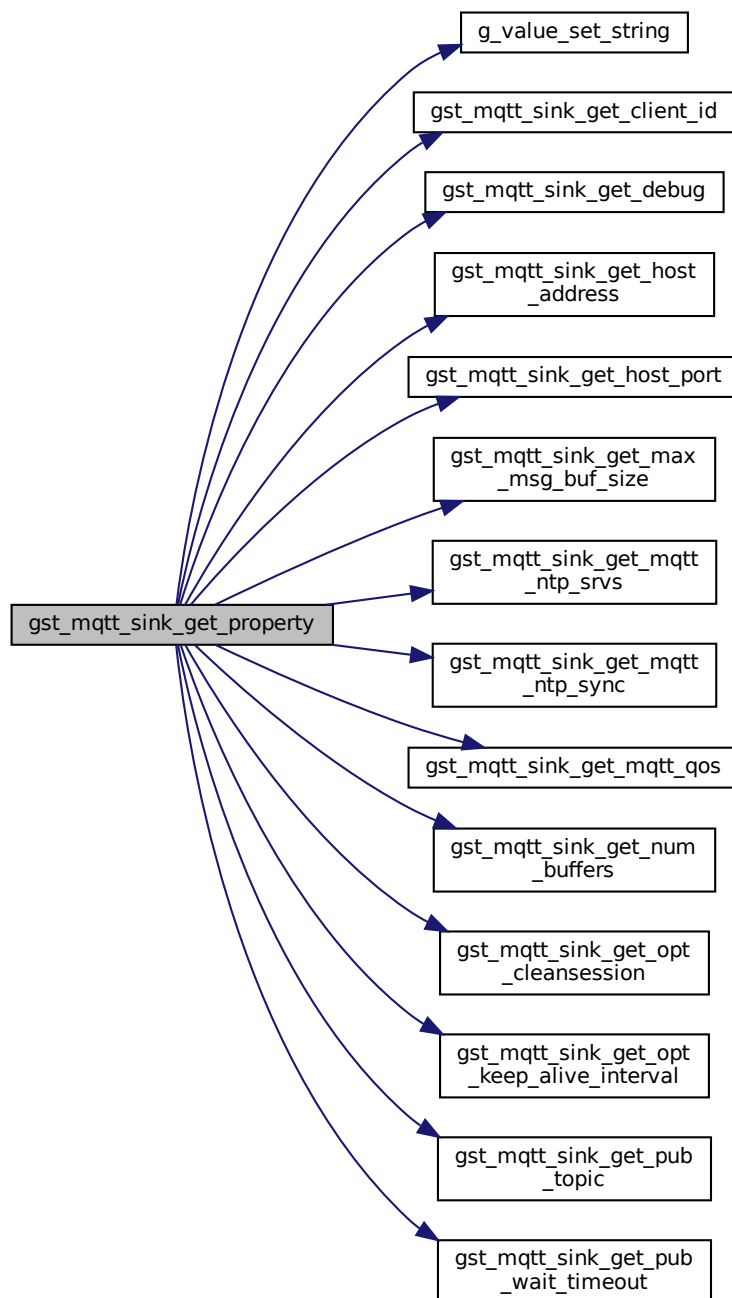
9.18.4.29 `gst_mqtt_sink_get_property()`

```
static void gst_mqtt_sink_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

The getter for the `mqttpsink`'s properties.

Definition at line 399 of file `mqttpsink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



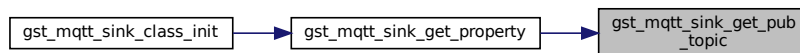
9.18.4.30 `gst_mqtt_sink_get_pub_topic()`

```
static gchar * gst_mqtt_sink_get_pub_topic (
    GstMqttSink * self ) [static]
```

Getter for the 'pub-topic' property.

Definition at line 1036 of file `mqttpsink.c`.

Here is the caller graph for this function:



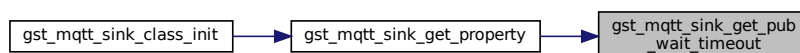
9.18.4.31 `gst_mqtt_sink_get_pub_wait_timeout()`

```
static gulong gst_mqtt_sink_get_pub_wait_timeout (
    GstMqttSink * self ) [static]
```

Getter for the 'pub-wait-timeout' property.

Definition at line 1073 of file `mqttpsink.c`.

Here is the caller graph for this function:



9.18.4.32 `gst_mqtt_sink_init()`

```
static void gst_mqtt_sink_init (  
    GstMqttSink * self ) [static]
```

Initialize GstMqttSink object.

init MQTT related variables

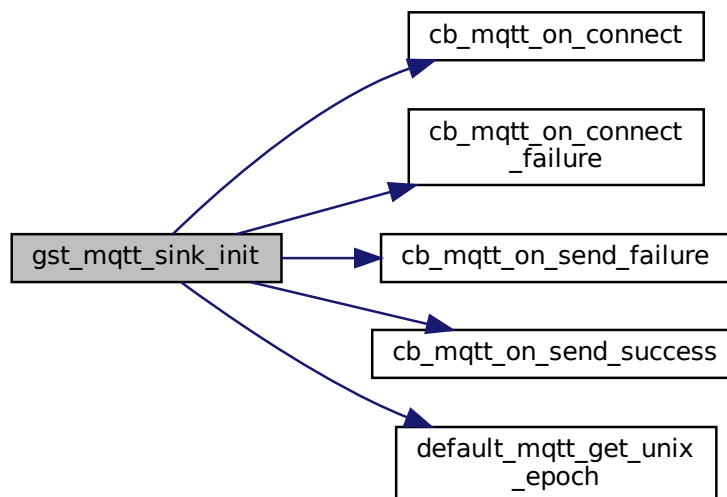
init private variables

init mqttsink properties

init basesink properties

Definition at line 171 of file mqttsink.c.

Here is the call graph for this function:



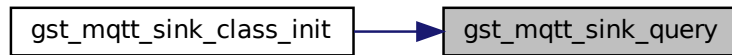
9.18.4.33 `gst_mqtt_sink_query()`

```
static gboolean gst_mqtt_sink_query (  
    GstBaseSink * basesink,  
    GstQuery * query ) [static]
```

Perform queries on the element.

Definition at line 661 of file mqttsink.c.

Here is the caller graph for this function:



9.18.4.34 `gst_mqtt_sink_render()`

```

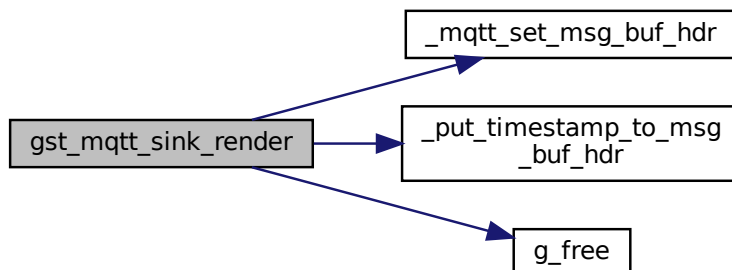
static GstFlowReturn gst_mqtt_sink_render (
    GstBaseSink * basesink,
    GstBuffer * buffer ) [static]
  
```

The callback to process each buffer receiving on the sink pad.

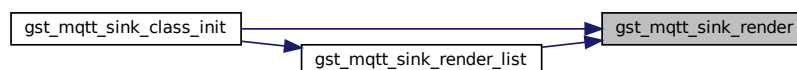
Allocate a message buffer

Definition at line 751 of file mqttsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



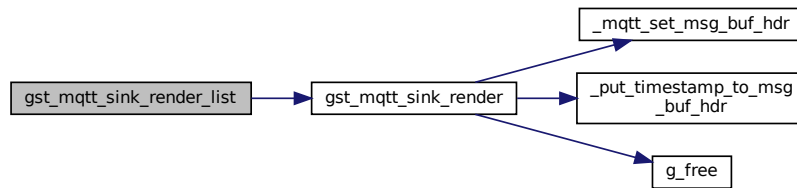
9.18.4.35 `gst_mqtt_sink_render_list()`

```
static GstFlowReturn gst_mqtt_sink_render_list (
    GstBaseSink * basesink,
    GstBufferList * list ) [static]
```

The callback to process `GstBufferList` (instead of a single buffer) on the sink pad.

Definition at line 877 of file `mqttsink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



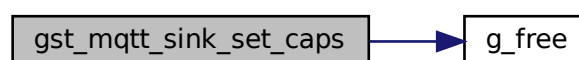
9.18.4.36 `gst_mqtt_sink_set_caps()`

```
static gboolean gst_mqtt_sink_set_caps (
    GstBaseSink * basesink,
    GstCaps * caps ) [static]
```

An implementation of the `set_caps` vmethod in `GstBaseSinkClass`.

Definition at line 924 of file `mqttsink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



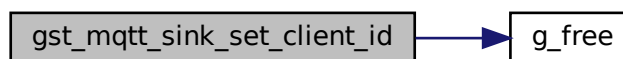
9.18.4.37 `gst_mqtt_sink_set_client_id()`

```
static void gst_mqtt_sink_set_client_id (  
    GstMqttSink * self,  
    const gchar * id ) [static]
```

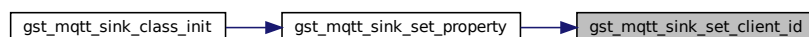
Setter for the 'client-id' property.

Definition at line 985 of file `mqttsink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



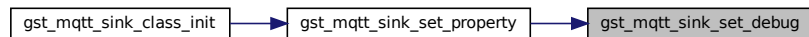
9.18.4.38 `gst_mqtt_sink_set_debug()`

```
static void gst_mqtt_sink_set_debug (
    GstMqttSink * self,
    const gboolean flag ) [static]
```

Setter for the 'debug' property.

Definition at line 967 of file mqttsink.c.

Here is the caller graph for this function:



9.18.4.39 `gst_mqtt_sink_set_host_address()`

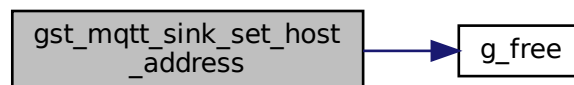
```
static void gst_mqtt_sink_set_host_address (
    GstMqttSink * self,
    const gchar * addr ) [static]
```

Setter for the 'host' property.

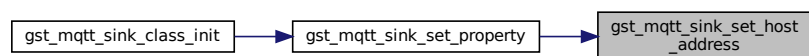
Todo Handle the case where the addr is changed at runtime

Definition at line 1004 of file mqttsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



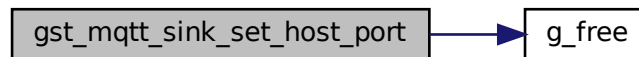
9.18.4.40 `gst_mqtt_sink_set_host_port()`

```
static void gst_mqtt_sink_set_host_port (
    GstMqttSink * self,
    const gchar * port ) [static]
```

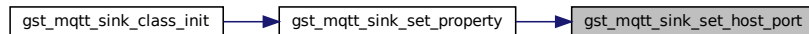
Setter for the 'port' property.

Definition at line 1026 of file `mqttpsink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



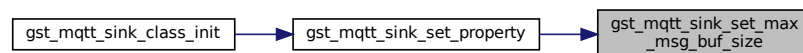
9.18.4.41 `gst_mqtt_sink_set_max_msg_buf_size()`

```
static void gst_mqtt_sink_set_max_msg_buf_size (
    GstMqttSink * self,
    const gsize size ) [static]
```

Setter for the 'max-buffer-size' property.

Definition at line 1118 of file `mqttpsink.c`.

Here is the caller graph for this function:



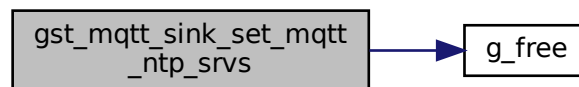
9.18.4.42 `gst_mqtt_sink_set_mqtt_ntp_srvs()`

```
static void gst_mqtt_sink_set_mqtt_ntp_srvs (  
    GstMqttSink * self,  
    const gchar * pairs ) [static]
```

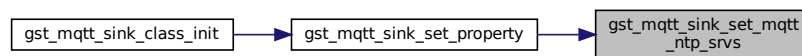
Setter for the 'ntp-srvs' property.

Definition at line 1194 of file mqttsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



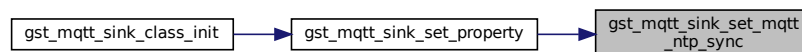
9.18.4.43 `gst_mqtt_sink_set_mqtt_ntp_sync()`

```
static void gst_mqtt_sink_set_mqtt_ntp_sync (  
    GstMqttSink * self,  
    const gboolean flag ) [static]
```

Setter for the 'ntp-sync' property.

Definition at line 1176 of file mqttsink.c.

Here is the caller graph for this function:



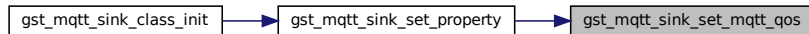
9.18.4.44 `gst_mqtt_sink_set_mqtt_qos()`

```
static void gst_mqtt_sink_set_mqtt_qos (
    GstMqttSink * self,
    const gint qos ) [static]
```

Setter for the 'mqtt-qos' property.

Definition at line 1158 of file `mqttpsink.c`.

Here is the caller graph for this function:



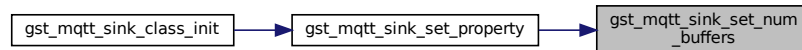
9.18.4.45 `gst_mqtt_sink_set_num_buffers()`

```
static void gst_mqtt_sink_set_num_buffers (
    GstMqttSink * self,
    const gint num ) [static]
```

Setter for the 'num-buffers' property.

Definition at line 1140 of file `mqttpsink.c`.

Here is the caller graph for this function:



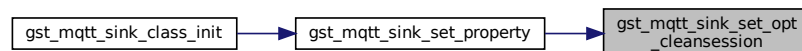
9.18.4.46 `gst_mqtt_sink_set_opt_cleansession()`

```
static void gst_mqtt_sink_set_opt_cleansession (
    GstMqttSink * self,
    const gboolean val ) [static]
```

Setter for the 'cleansession' property.

Definition at line 1064 of file `mqttpsink.c`.

Here is the caller graph for this function:



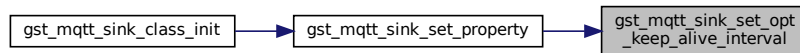
9.18.4.47 `gst_mqtt_sink_set_opt_keep_alive_interval()`

```
static void gst_mqtt_sink_set_opt_keep_alive_interval (
    GstMqttSink * self,
    const gint num ) [static]
```

Setter for the 'keep-alive-interval' property.

Definition at line 1100 of file mqttsink.c.

Here is the caller graph for this function:



9.18.4.48 `gst_mqtt_sink_set_property()`

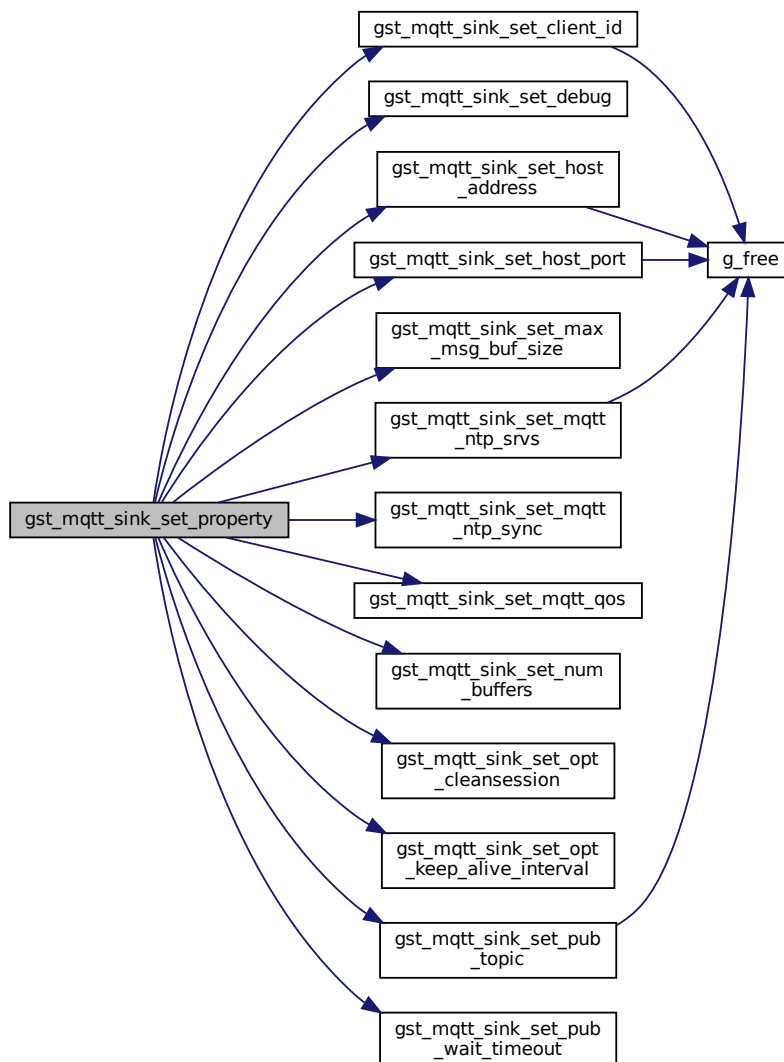
```
static void gst_mqtt_sink_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

The setter for the mqttsink's properties.

Function prototype declarations

Definition at line 344 of file mqttsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



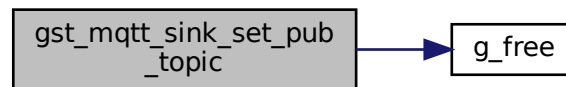
9.18.4.49 `gst_mqtt_sink_set_pub_topic()`

```
static void gst_mqtt_sink_set_pub_topic (  
    GstMqttSink * self,  
    const gchar * topic ) [static]
```

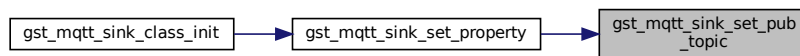
Setter for the 'pub-topic' property.

Definition at line 1045 of file mqttsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



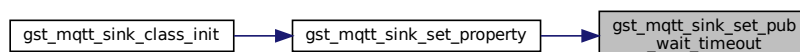
9.18.4.50 `gst_mqtt_sink_set_pub_wait_timeout()`

```
static void gst_mqtt_sink_set_pub_wait_timeout (  
    GstMqttSink * self,  
    const gulong to ) [static]
```

Setter for the 'pub-wait-timeout' property.

Definition at line 1082 of file mqttsink.c.

Here is the caller graph for this function:



9.18.4.51 `gst_mqtt_sink_start()`

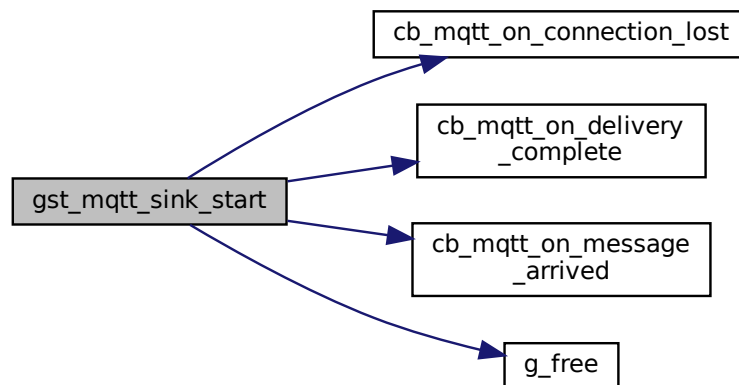
```
static gboolean gst_mqtt_sink_start (
    GstBaseSink * basesink ) [static]
```

Start mqttsink, called when state changed null to ready.

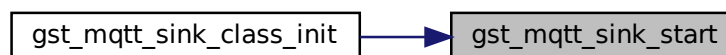
Todo Support other persistence mechanisms MQTTCLIENT_PERSISTENCE_NONE: A memory-based persistence mechanism MQTTCLIENT_PERSISTENCE_DEFAULT: The default file system-based persistence mechanism MQTTCLIENT_PERSISTENCE_USER: An application-specific persistence mechanism

Definition at line 554 of file mqttsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



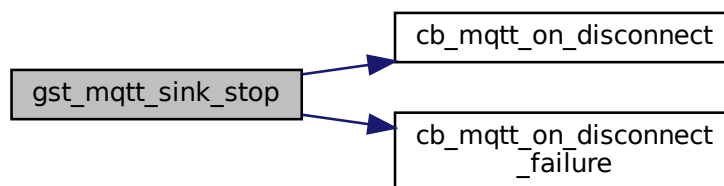
9.18.4.52 `gst_mqtt_sink_stop()`

```
static gboolean gst_mqtt_sink_stop (  
    GstBaseSink * basesink ) [static]
```

Stop mqttsink, called when state changed ready to null.

Definition at line 623 of file mqttsink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.18.5 Variable Documentation

9.18.5.1 `DEFAULT_MQTT_CLIENT_ID`

```
const gchar DEFAULT_MQTT_CLIENT_ID[] = "$HOST_$PID^[0-9][0-9]?|^255$" [static]
```

Definition at line 85 of file mqttsink.c.

9.18.5.2 DEFAULT_MQTT_CLIENT_ID_FORMAT

```
const gchar DEFAULT_MQTT_CLIENT_ID_FORMAT[] = "%s_%u_sink%u" [static]
```

Definition at line 86 of file mqttsink.c.

9.18.5.3 DEFAULT_MQTT_HOST_ADDRESS

```
const gchar DEFAULT_MQTT_HOST_ADDRESS[] = "127.0.0.1" [static]
```

Definition at line 82 of file mqttsink.c.

9.18.5.4 DEFAULT_MQTT_HOST_PORT

```
const gchar DEFAULT_MQTT_HOST_PORT[] = "1883" [static]
```

Definition at line 83 of file mqttsink.c.

9.18.5.5 DEFAULT_MQTT_NTP_SERVERS

```
const gchar DEFAULT_MQTT_NTP_SERVERS[] = "pool.ntp.org:123" [static]
```

Definition at line 89 of file mqttsink.c.

9.18.5.6 DEFAULT_MQTT_PUB_TOPIC

```
const gchar DEFAULT_MQTT_PUB_TOPIC[] = "$client-id/topic" [static]
```

Definition at line 87 of file mqttsink.c.

9.18.5.7 DEFAULT_MQTT_PUB_TOPIC_FORMAT

```
const gchar DEFAULT_MQTT_PUB_TOPIC_FORMAT[] = "%s/topic" [static]
```

Definition at line 88 of file mqttsink.c.

9.18.5.8 sink_client_id

```
guint8 sink_client_id = 0 [static]
```

Definition at line 81 of file mqttsink.c.

9.18.5.9 sink_pad_template

```
GstStaticPadTemplate sink_pad_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK, GST_PAD_ALWAYS, GST_STATIC_CAPS_ANY)
```

Definition at line 35 of file mqttsink.c.

9.18.5.10 TAG_ERR_MQTTSINK

```
const gchar TAG_ERR_MQTTSINK[] = "ERROR: MQTTSink" [static]
```

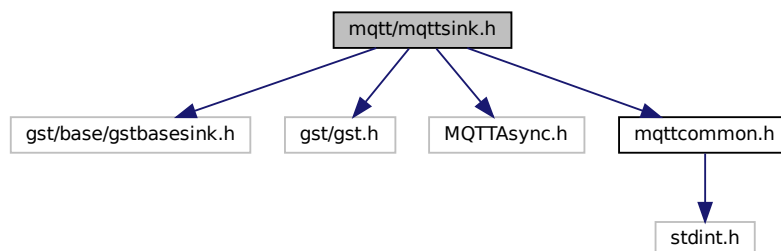
Definition at line 84 of file mqttsink.c.

9.19 mqtt/mqttsink.h File Reference

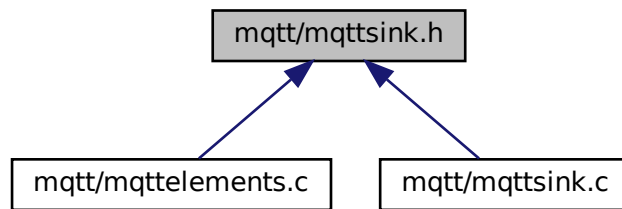
Publish incoming data streams as a MQTT topic.

```
#include <gst/base/gstbasesink.h>  
#include <gst/gst.h>  
#include <MQTTAsync.h>  
#include "mqttcommon.h"
```

Include dependency graph for mqttsink.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstMqttSink](#)
GstMqttSink data structure.
- struct [_GstMqttSinkClass](#)
GstMqttSinkClass data structure.

Macros

- #define [GST_TYPE_MQTT_SINK](#) ([gst_mqtt_sink_get_type\(\)](#))
- #define [GST_MQTT_SINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST](#) ((obj), [GST_TYPE_MQTT_SINK](#), [GstMqttSink](#)))
- #define [GST_IS_MQTT_SINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE](#) ((obj), [GST_TYPE_MQTT_SINK](#)))
- #define [GST_MQTT_SINK_CAST\(obj\)](#) (([GstMqttSink *](#)) obj)
- #define [GST_MQTT_SINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST](#) ((klass), [GST_TYPE_MQTT_SINK](#), [GstMqttSinkClass](#)))
- #define [GST_IS_MQTT_SINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE](#) ((klass), [GST_TYPE_MQTT_SINK](#)))

Typedefs

- typedef struct [_GstMqttSink](#) [GstMqttSink](#)
- typedef struct [_GstMqttSinkClass](#) [GstMqttSinkClass](#)
- typedef enum [_mqtt_sink_state_t](#) [mqtt_sink_state_t](#)
A type definition to indicate the state of this element.

Enumerations

- enum [_mqtt_sink_state_t](#) {
[MQTT_CONNECTION_LOST](#) = -3, [MQTT_CONNECT_FAILURE](#) = -2, [SINK_INITIALIZING](#) = -1,
[SINK_RENDER_STOPPED](#) = 0,
[SINK_RENDER_EOS](#), [SINK_RENDER_ERROR](#), [MQTT_CONNECTED](#), [MQTT_DISCONNECTED](#),
[MQTT_DISCONNECT_FAILED](#) }
A type definition to indicate the state of this element.

Functions

- GType [gst_mqtt_sink_get_type](#) (void)

9.19.1 Detailed Description

Publish incoming data streams as a MQTT topic.

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

08 Mar 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.19.2 Macro Definition Documentation

9.19.2.1 GST_IS_MQTT_SINK

```
#define GST_IS_MQTT_SINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_MQTT_SINK))
```

Definition at line 28 of file mqttsink.h.

9.19.2.2 GST_IS_MQTT_SINK_CLASS

```
#define GST_IS_MQTT_SINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_MQTT_SINK))
```

Definition at line 34 of file mqttsink.h.

9.19.2.3 GST_MQTT_SINK

```
#define GST_MQTT_SINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_MQTT_SINK, GstMqttSink))
```

Definition at line 26 of file `mqttpsink.h`.

9.19.2.4 GST_MQTT_SINK_CAST

```
#define GST_MQTT_SINK_CAST(  
    obj ) ((GstMqttSink *) obj)
```

Definition at line 30 of file `mqttpsink.h`.

9.19.2.5 GST_MQTT_SINK_CLASS

```
#define GST_MQTT_SINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_MQTT_SINK, GstMqttSinkClass))
```

Definition at line 32 of file `mqttpsink.h`.

9.19.2.6 GST_TYPE_MQTT_SINK

```
#define GST_TYPE_MQTT_SINK (gst_mqtt_sink_get_type())
```

Definition at line 24 of file `mqttpsink.h`.

9.19.3 Typedef Documentation

9.19.3.1 GstMqttSink

```
typedef struct _GstMqttSink GstMqttSink
```

Definition at line 37 of file `mqttpsink.h`.

9.19.3.2 GstMqttSinkClass

```
typedef struct _GstMqttSinkClass GstMqttSinkClass
```

Definition at line 38 of file mqttsink.h.

9.19.3.3 mqtt_sink_state_t

```
typedef enum _mqtt_sink_state_t mqtt_sink_state_t
```

A type definition to indicate the state of this element.

9.19.4 Enumeration Type Documentation

9.19.4.1 _mqtt_sink_state_t

```
enum _mqtt_sink_state_t
```

A type definition to indicate the state of this element.

Enumerator

MQTT_CONNECTION_LOST	
MQTT_CONNECT_FAILURE	
SINK_INITIALIZING	
SINK_RENDER_STOPPED	
SINK_RENDER_EOS	
SINK_RENDER_ERROR	
MQTT_CONNECTED	
MQTT_DISCONNECTED	
MQTT_DISCONNECT_FAILED	

Definition at line 43 of file mqttsink.h.

9.19.5 Function Documentation

9.19.5.1 gst_mqtt_sink_get_type()

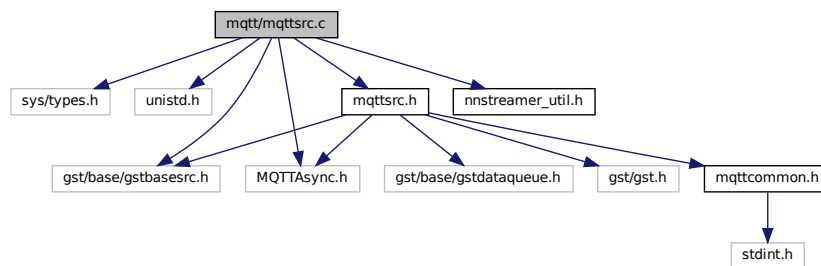
```
GType gst_mqtt_sink_get_type (
    void )
```

9.20 mqtt/mqttsrc.c File Reference

Subscribe a MQTT topic and push incoming data to the GStreamer pipeline.

```
#include <sys/types.h>
#include <unistd.h>
#include <gst/base/gstbasesrc.h>
#include <MQTTAsync.h>
#include <nnstreamer_util.h>
#include "mqttsrc.h"
```

Include dependency graph for mqttsrc.c:



Macros

- #define [gst_mqtt_src_parent_class](#) parent_class
- #define [GST_CAT_DEFAULT](#) gst_mqtt_src_debug

Enumerations

- enum {
[PROP_0](#), [PROP_DEBUG](#), [PROP_IS_LIVE](#), [PROP_MQTT_CLIENT_ID](#),
[PROP_MQTT_HOST_ADDRESS](#), [PROP_MQTT_HOST_PORT](#), [PROP_MQTT_SUB_TOPIC](#), [PROP_MQTT_SUB_TIMEOUT](#),
[PROP_MQTT_OPT_CLEANSSESSION](#), [PROP_MQTT_OPT_KEEP_ALIVE_INTERVAL](#), [PROP_MQTT_QOS](#),
[PROP_LAST](#) }
- enum {
[DEFAULT_DEBUG](#) = FALSE, [DEFAULT_IS_LIVE](#) = TRUE, [DEFAULT_MQTT_OPT_CLEANSSESSION](#) =
TRUE, [DEFAULT_MQTT_OPT_KEEP_ALIVE_INTERVAL](#) = 60,
[DEFAULT_MQTT_SUB_TIMEOUT](#) = 10000000, [DEFAULT_MQTT_SUB_TIMEOUT_MIN](#) = 1000000,
[DEFAULT_MQTT_QOS](#) = 2 }

Functions

- [G_DEFINE_TYPE](#) (GstMqttSrc, gst_mqtt_src, GST_TYPE_BASE_SRC)
- [GST_DEBUG_CATEGORY_STATIC](#) (gst_mqtt_src_debug)
- static void [gst_mqtt_src_set_property](#) (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
The setter for the mqttsrc's properties.
- static void [gst_mqtt_src_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
The getter for the mqttsrc's properties.

- static void [gst_mqtt_src_class_finalize](#) (GObject *object)
Finalize GstMqttSrcClass object.
- static GstStateChangeReturn [gst_mqtt_src_change_state](#) (GstElement *element, GstStateChange transition)
Handle mqttsrc's state change.
- static gboolean [gst_mqtt_src_start](#) (GstBaseSrc *basesrc)
Start mqttsrc, called when state changed null to ready.
- static gboolean [gst_mqtt_src_stop](#) (GstBaseSrc *basesrc)
Stop mqttsrc, called when state changed ready to null.
- static GstCaps * [gst_mqtt_src_get_caps](#) (GstBaseSrc *basesrc, GstCaps *filter)
Get caps of subclass.
- static gboolean [gst_mqtt_src_renegotiate](#) (GstBaseSrc *basesrc)
Do negotiation procedure again if it needed.
- static void [gst_mqtt_src_get_times](#) (GstBaseSrc *basesrc, GstBuffer *buffer, GstClockTime *start, GstClockTime *end)
Return the time information of the given buffer.
- static gboolean [gst_mqtt_src_is_seekable](#) (GstBaseSrc *basesrc)
Check if source supports seeking.
- static GstFlowReturn [gst_mqtt_src_create](#) (GstBaseSrc *basesrc, guint64 offset, guint size, GstBuffer **buf)
Create a buffer containing the subscribed data.
- static gboolean [gst_mqtt_src_query](#) (GstBaseSrc *basesrc, GstQuery *query)
An implementation of the GstBaseSrc vmethod that handles queries.
- static gboolean [gst_mqtt_src_get_debug](#) (GstMqttSrc *self)
Getter for the 'debug' property.
- static void [gst_mqtt_src_set_debug](#) (GstMqttSrc *self, const gboolean flag)
Setter for the 'debug' property.
- static gboolean [gst_mqtt_src_get_is_live](#) (GstMqttSrc *self)
Getter for the 'is-live' property.
- static void [gst_mqtt_src_set_is_live](#) (GstMqttSrc *self, const gboolean flag)
Setter for the 'is-live' property.
- static gchar * [gst_mqtt_src_get_client_id](#) (GstMqttSrc *self)
Getter for the 'client-id' property.
- static void [gst_mqtt_src_set_client_id](#) (GstMqttSrc *self, const gchar *id)
Setter for the 'client-id' property.
- static gchar * [gst_mqtt_src_get_host_address](#) (GstMqttSrc *self)
Getter for the 'host' property.
- static void [gst_mqtt_src_set_host_address](#) (GstMqttSrc *self, const gchar *addr)
Setter for the 'host' property.
- static gchar * [gst_mqtt_src_get_host_port](#) (GstMqttSrc *self)
Getter for the 'port' property.
- static void [gst_mqtt_src_set_host_port](#) (GstMqttSrc *self, const gchar *port)
Setter for the 'port' property.
- static gint64 [gst_mqtt_src_get_sub_timeout](#) (GstMqttSrc *self)
Getter for the 'sub-timeout' property.
- static void [gst_mqtt_src_set_sub_timeout](#) (GstMqttSrc *self, const gint64 t)
Setter for the 'sub-timeout' property.
- static gchar * [gst_mqtt_src_get_sub_topic](#) (GstMqttSrc *self)
Getter for the 'sub-topic' property.
- static void [gst_mqtt_src_set_sub_topic](#) (GstMqttSrc *self, const gchar *topic)
Setter for the 'sub-topic' property.
- static gboolean [gst_mqtt_src_get_opt_cleansession](#) (GstMqttSrc *self)

- Getter for the 'cleansession' property.*

 - static void `gst_mqtt_src_set_opt_cleansession` (`GstMqttSrc *self`, `const gboolean val`)

Setter for the 'cleansession' property.

 - static gint `gst_mqtt_src_get_opt_keep_alive_interval` (`GstMqttSrc *self`)

Getter for the 'keep-alive-interval' property.

 - static void `gst_mqtt_src_set_opt_keep_alive_interval` (`GstMqttSrc *self`, `const gint num`)

Setter for the 'keep-alive-interval' property.

 - static gint `gst_mqtt_src_get_mqtt_qos` (`GstMqttSrc *self`)

Getter for the 'mqtt-qos' property.

 - static void `gst_mqtt_src_set_mqtt_qos` (`GstMqttSrc *self`, `const gint qos`)

Setter for the 'mqtt-qos' property.

 - static void `cb_mqtt_on_connection_lost` (`void *context`, `char *cause`)

A callback to handle the connection lost to the broker.

 - static int `cb_mqtt_on_message_arrived` (`void *context`, `char *topic_name`, `int topic_len`, `MQTTAsync_↔ message *message`)

A callback to handle the arrived message.

 - static void `cb_mqtt_on_connect` (`void *context`, `MQTTAsync_successData *response`)

A callback invoked when the connection is established.

 - static void `cb_mqtt_on_connect_failure` (`void *context`, `MQTTAsync_failureData *response`)

A callback invoked when it is failed to connect to the broker.

 - static void `cb_mqtt_on_subscribe` (`void *context`, `MQTTAsync_successData *response`)

MQTTAsync_responseOptions's onSuccess callback for MQTTAsync_subscribe ()

 - static void `cb_mqtt_on_subscribe_failure` (`void *context`, `MQTTAsync_failureData *response`)

MQTTAsync_responseOptions's onFailure callback for MQTTAsync_subscribe ()

 - static void `cb_mqtt_on_unsubscribe` (`void *context`, `MQTTAsync_successData *response`)

MQTTAsync_responseOptions's onSuccess callback for MQTTAsync_unsubscribe ()

 - static void `cb_mqtt_on_unsubscribe_failure` (`void *context`, `MQTTAsync_failureData *response`)

MQTTAsync_responseOptions's onFailure callback for MQTTAsync_unsubscribe ()

 - static void `cb_memory_wrapped_destroy` (`void *p`)

A callback invoked when destroying the GstMemory which wrapped the arrived message.

 - static `GstMQTTMessageHdr * _extract_mqtt_msg_hdr_from` (`GstMemory *mem`, `GstMemory **hdr_mem`, `GstMapInfo *hdr_map_info`)

A utility function to extract header information from a received message.

 - static void `_put_timestamp_on_gst_buf` (`GstMqttSrc *self`, `GstMQTTMessageHdr *hdr`, `GstBuffer *buf`)

A utility function to put the timestamp information onto a GstBuffer-typed buffer using the given packet header.

 - static gboolean `_subscribe` (`GstMqttSrc *self`)

A helper function to properly invoke MQTTAsync_subscribe ()

 - static gboolean `_unsubscribe` (`GstMqttSrc *self`)

A wrapper function that calls MQTTAsync_unsubscribe ()

 - static gboolean `_is_gst_buffer_timestamp_valid` (`GstBuffer *buf`)

A utility function to check whether the timestamp marked by _put_timestamp_on_gst_buf () is valid or not.

 - static void `gst_mqtt_src_init` (`GstMqttSrc *self`)

Initialize GstMqttSrc object.

 - static void `gst_mqtt_src_class_init` (`GstMqttSrcClass *klass`)

Initialize GstMqttSrcClass object.

Variables

- static GstStaticPadTemplate [src_pad_template](#)
- static guint8 [src_client_id](#) = 0
- static const gchar [DEFAULT_MQTT_HOST_ADDRESS](#) [] = "127.0.0.1"
- static const gchar [DEFAULT_MQTT_HOST_PORT](#) [] = "1883"
- static const gchar [TAG_ERR_MQTTSRC](#) [] = "ERROR: MQTTSrc"
- static const gchar [DEFAULT_MQTT_CLIENT_ID](#) []
- static const gchar [DEFAULT_MQTT_CLIENT_ID_FORMAT](#) [] = "%s_%u_src%u"

9.20.1 Detailed Description

Subscribe a MQTT topic and push incoming data to the GStreamer pipeline.

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

08 Mar 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.20.2 Macro Definition Documentation

9.20.2.1 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_mqtt_src_debug
```

Definition at line 38 of file mqttsrc.c.

9.20.2.2 gst_mqtt_src_parent_class

```
#define gst_mqtt_src_parent_class parent_class
```

Definition at line 34 of file mqttsrc.c.

9.20.3 Enumeration Type Documentation

9.20.3.1 anonymous enum

```
anonymous enum
```

Enumerator

PROP_0	
PROP_DEBUG	
PROP_IS_LIVE	
PROP_MQTT_CLIENT_ID	
PROP_MQTT_HOST_ADDRESS	
PROP_MQTT_HOST_PORT	
PROP_MQTT_SUB_TOPIC	
PROP_MQTT_SUB_TIMEOUT	
PROP_MQTT_OPT_CLEANSESSION	
PROP_MQTT_OPT_KEEP_ALIVE_INTERVAL	
PROP_MQTT_QOS	
PROP_LAST	

Definition at line 40 of file mqttsrc.c.

9.20.3.2 anonymous enum

anonymous enum

Enumerator

DEFAULT_DEBUG	
DEFAULT_IS_LIVE	
DEFAULT_MQTT_OPT_CLEANSESSION	
DEFAULT_MQTT_OPT_KEEP_ALIVE_INTERVAL	
DEFAULT_MQTT_SUB_TIMEOUT	
DEFAULT_MQTT_SUB_TIMEOUT_MIN	
DEFAULT_MQTT_QOS	

Definition at line 58 of file mqttsrc.c.

9.20.4 Function Documentation

9.20.4.1 `_extract_mqtt_msg_hdr_from()`

```
static GstMQTTMessageHdr * _extract_mqtt_msg_hdr_from (
    GstMemory * mem,
    GstMemory ** hdr_mem,
    GstMapInfo * hdr_map_info ) [static]
```

A utility function to extract header information from a received message.

Definition at line 1347 of file mqttsrc.c.

Here is the caller graph for this function:



9.20.4.2 _is_gst_buffer_timestamp_valid()

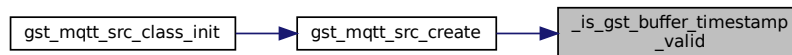
```

static gboolean _is_gst_buffer_timestamp_valid (
    GstBuffer * buf ) [inline], [static]
  
```

A utility function to check whether the timestamp marked by `_put_timestamp_on_gst_buf ()` is valid or not.

Definition at line 156 of file mqttsrc.c.

Here is the caller graph for this function:



9.20.4.3 _put_timestamp_on_gst_buf()

```

static void _put_timestamp_on_gst_buf (
    GstMqttSrc * self,
    GstMQTTMessageHdr * hdr,
    GstBuffer * buf ) [static]
  
```

A utility function to put the timestamp information onto a `GstBuffer`-typed buffer using the given packet header.

Definition at line 1366 of file mqttsrc.c.

Here is the caller graph for this function:



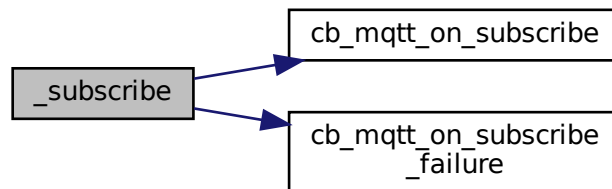
9.20.4.4 `_subscribe()`

```
static gboolean _subscribe (  
    GstMqttSrc * self ) [static]
```

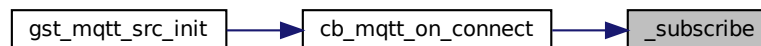
A helper function to properly invoke `MQTTAsync_subscribe ()`

Definition at line 1308 of file `mqttsrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



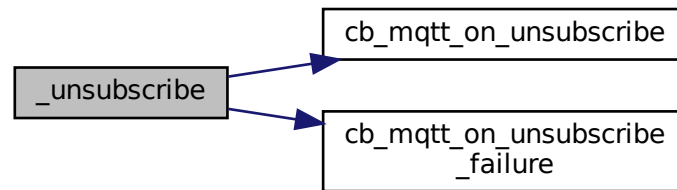
9.20.4.5 `_unsubscribe()`

```
static gboolean _unsubscribe (  
    GstMqttSrc * self ) [static]
```

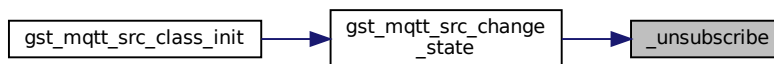
A wrapper function that calls `MQTTAsync_unsubscribe ()`

Definition at line 1328 of file `mqttsrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.20.4.6 `cb_memory_wrapped_destroy()`

```
static void cb_memory_wrapped_destroy (  
    void * p ) [static]
```

A callback invoked when destroying the `GstMemory` which wrapped the arrived message.

Definition at line 1179 of file `mqttsrc.c`.

Here is the caller graph for this function:



9.20.4.7 `cb_mqtt_on_connect()`

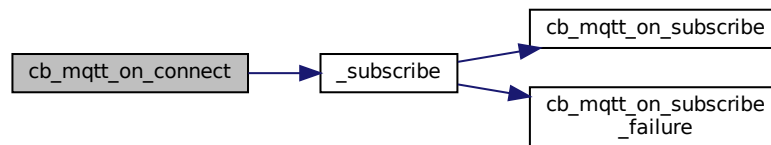
```
static void cb_mqtt_on_connect (
    void * context,
    MQTTAsync_successData * response ) [static]
```

A callback invoked when the connection is established.

GstFlowReturn is an enum type. It is possible to use int here

Definition at line 1190 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



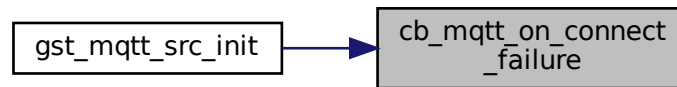
9.20.4.8 `cb_mqtt_on_connect_failure()`

```
static void cb_mqtt_on_connect_failure (
    void * context,
    MQTTAsync_failureData * response ) [static]
```

A callback invoked when it is failed to connect to the broker.

Definition at line 1223 of file mqttsrc.c.

Here is the caller graph for this function:



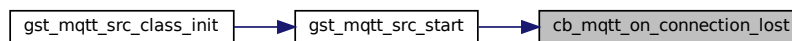
9.20.4.9 `cb_mqtt_on_connection_lost()`

```
static void cb_mqtt_on_connection_lost (  
    void * context,  
    char * cause ) [static]
```

A callback to handle the connection lost to the broker.

Definition at line 1053 of file `mqttsrc.c`.

Here is the caller graph for this function:



9.20.4.10 `cb_mqtt_on_message_arrived()`

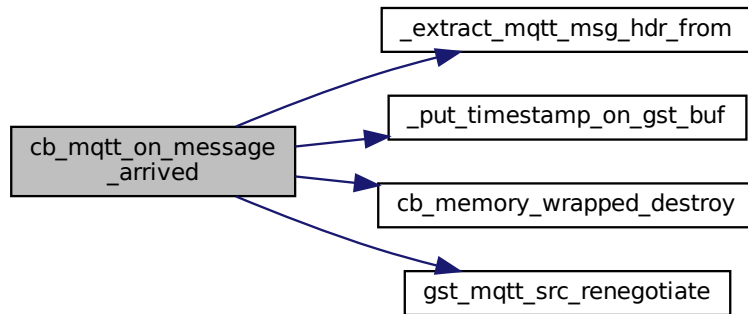
```
static int cb_mqtt_on_message_arrived (  
    void * context,  
    char * topic_name,  
    int topic_len,  
    MQTTAsync_message * message ) [static]
```

A callback to handle the arrived message.

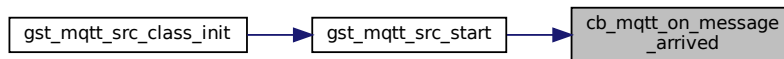
Timestamp synchronization

Definition at line 1075 of file `mqttsrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.20.4.11 cb_mqtt_on_subscribe()

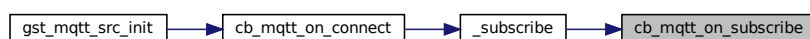
```

static void cb_mqtt_on_subscribe (
    void * context,
    MQTTAsync_successData * response ) [static]
  
```

MQTTAsync_responseOptions's onSuccess callback for MQTTAsync_subscribe ()

Definition at line 1242 of file mqtsrc.c.

Here is the caller graph for this function:



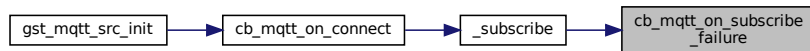
9.20.4.12 cb_mqtt_on_subscribe_failure()

```
static void cb_mqtt_on_subscribe_failure (
    void * context,
    MQTTAsync_failureData * response ) [static]
```

MQTTAsync_responseOptions's onFailure callback for MQTTAsync_subscribe ()

Definition at line 1257 of file mqttsrc.c.

Here is the caller graph for this function:

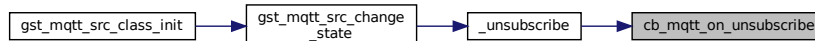
**9.20.4.13 cb_mqtt_on_unsubscribe()**

```
static void cb_mqtt_on_unsubscribe (
    void * context,
    MQTTAsync_successData * response ) [static]
```

MQTTAsync_responseOptions's onSuccess callback for MQTTAsync_unsubscribe ()

Definition at line 1275 of file mqttsrc.c.

Here is the caller graph for this function:

**9.20.4.14 cb_mqtt_on_unsubscribe_failure()**

```
static void cb_mqtt_on_unsubscribe_failure (
    void * context,
    MQTTAsync_failureData * response ) [static]
```

MQTTAsync_responseOptions's onFailure callback for MQTTAsync_unsubscribe ()

Definition at line 1290 of file mqttsrc.c.

Here is the caller graph for this function:



9.20.4.15 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstMqttSrc ,
    gst_mqtt_src ,
    GST_TYPE_BASE_SRC )
```

9.20.4.16 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_mqtt_src_debug )
```

9.20.4.17 gst_mqtt_src_change_state()

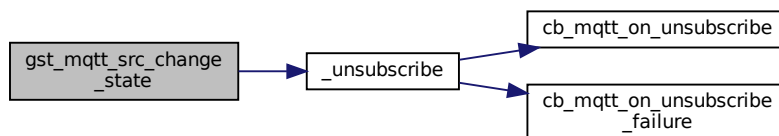
```
static GstStateChangeReturn gst_mqtt_src_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

Handle mqttsrc's state change.

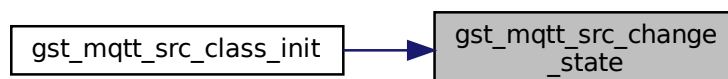
This handles the case when the state is changed to PLAYING again

Definition at line 440 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



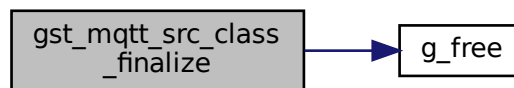
9.20.4.18 `gst_mqtt_src_class_finalize()`

```
static void gst_mqtt_src_class_finalize (  
    GObject * object ) [static]
```

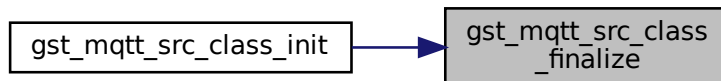
Finalize GstMqttSrcClass object.

Definition at line 408 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



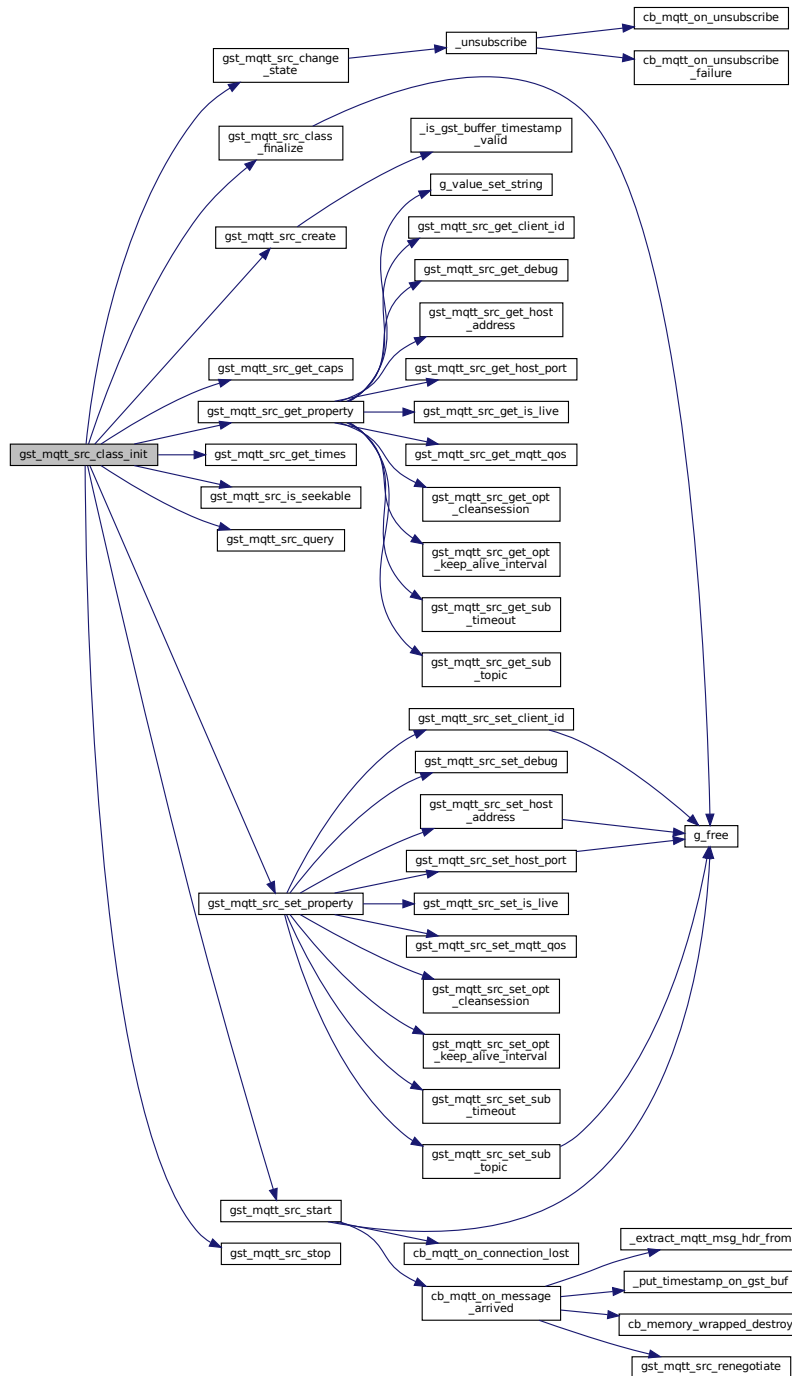
9.20.4.19 `gst_mqtt_src_class_init()`

```
static void gst_mqtt_src_class_init (  
    GstMqttSrcClass * klass ) [static]
```

Initialize GstMqttSrcClass object.

Definition at line 222 of file mqttsrc.c.

Here is the call graph for this function:



9.20.4.20 gst_mqtt_src_create()

```
static GstFlowReturn gst_mqtt_src_create (
    GstBaseSrc * basesrc,
```

```
guint64 offset,  
guint size,  
GstBuffer ** buf ) [static]
```

Create a buffer containing the subscribed data.

Todo DEFAULT_MQTT_SUB_TIMEOUT_MIN is too long

This buffer is coming from the past. Drop it.

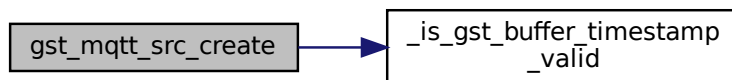
Update latency

Todo If the difference between new latency and old latency, `gst_element_post_message (GST_ELEMENT_CAST (self), gst_message_new_latency (GST_OBJECT_CAST (self)))`; is needed.

Todo : Send EoS here

Definition at line 710 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.20.4.21 `gst_mqtt_src_get_caps()`

```
static GstCaps * gst_mqtt_src_get_caps (
    GstBaseSrc * basesrc,
    GstCaps * filter ) [static]
```

Get caps of subclass.

Definition at line 601 of file `mqttsrc.c`.

Here is the caller graph for this function:



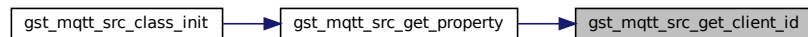
9.20.4.22 `gst_mqtt_src_get_client_id()`

```
static gchar * gst_mqtt_src_get_client_id (
    GstMqttSrc * self ) [static]
```

Getter for the 'client-id' property.

Definition at line 902 of file `mqttsrc.c`.

Here is the caller graph for this function:



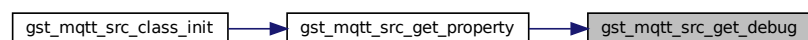
9.20.4.23 `gst_mqtt_src_get_debug()`

```
static gboolean gst_mqtt_src_get_debug (
    GstMqttSrc * self ) [static]
```

Getter for the 'debug' property.

Definition at line 865 of file `mqttsrc.c`.

Here is the caller graph for this function:



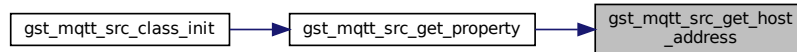
9.20.4.24 `gst_mqtt_src_get_host_address()`

```
static gchar * gst_mqtt_src_get_host_address (  
    GstMqttSrc * self ) [static]
```

Getter for the 'host' property.

Definition at line 921 of file mqttsrc.c.

Here is the caller graph for this function:



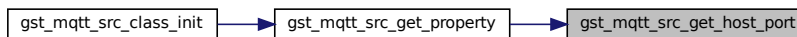
9.20.4.25 `gst_mqtt_src_get_host_port()`

```
static gchar * gst_mqtt_src_get_host_port (  
    GstMqttSrc * self ) [static]
```

Getter for the 'port' property.

Definition at line 943 of file mqttsrc.c.

Here is the caller graph for this function:



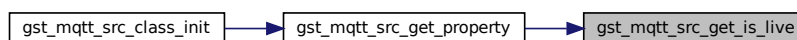
9.20.4.26 `gst_mqtt_src_get_is_live()`

```
static gboolean gst_mqtt_src_get_is_live (  
    GstMqttSrc * self ) [static]
```

Getter for the 'is-live' property.

Definition at line 883 of file mqttsrc.c.

Here is the caller graph for this function:



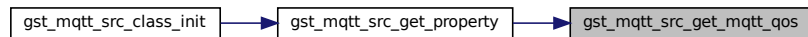
9.20.4.27 `gst_mqtt_src_get_mqtt_qos()`

```
static gint gst_mqtt_src_get_mqtt_qos (
    GstMqttSrc * self ) [static]
```

Getter for the 'mqtt-qos' property.

Definition at line 1035 of file `mqttsrc.c`.

Here is the caller graph for this function:



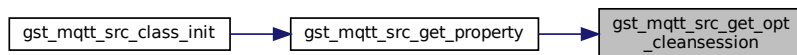
9.20.4.28 `gst_mqtt_src_get_opt_cleansession()`

```
static gboolean gst_mqtt_src_get_opt_cleansession (
    GstMqttSrc * self ) [static]
```

Getter for the 'cleansession' property.

Definition at line 999 of file `mqttsrc.c`.

Here is the caller graph for this function:



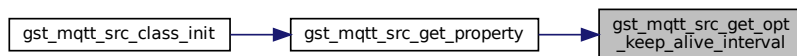
9.20.4.29 `gst_mqtt_src_get_opt_keep_alive_interval()`

```
static gint gst_mqtt_src_get_opt_keep_alive_interval (
    GstMqttSrc * self ) [static]
```

Getter for the 'keep-alive-interval' property.

Definition at line 1017 of file `mqttsrc.c`.

Here is the caller graph for this function:



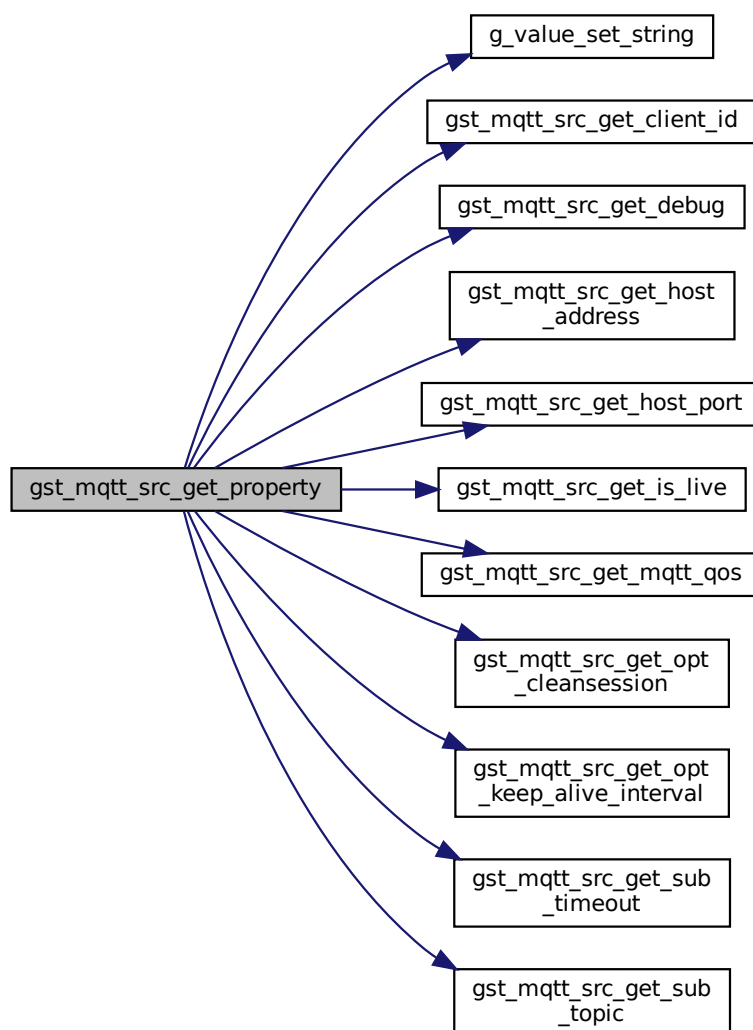
9.20.4.30 `gst_mqtt_src_get_property()`

```
static void gst_mqtt_src_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

The getter for the mqttsrc's properties.

Definition at line 362 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.20.4.31 `gst_mqtt_src_get_sub_timeout()`

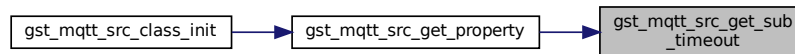
```

static gint64 gst_mqtt_src_get_sub_timeout (
    GstMqttSrc * self ) [static]
  
```

Getter for the 'sub-timeout' property.

Definition at line 962 of file `mqttsrc.c`.

Here is the caller graph for this function:



9.20.4.32 `gst_mqtt_src_get_sub_topic()`

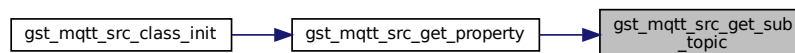
```

static gchar * gst_mqtt_src_get_sub_topic (
    GstMqttSrc * self ) [static]
  
```

Getter for the 'sub-topic' property.

Definition at line 980 of file `mqttsrc.c`.

Here is the caller graph for this function:



9.20.4.33 `gst_mqtt_src_get_times()`

```
static void gst_mqtt_src_get_times (
    GstBaseSrc * basesrc,
    GstBuffer * buffer,
    GstClockTime * start,
    GstClockTime * end ) [static]
```

Return the time information of the given buffer.

Definition at line 674 of file mqttsrc.c.

Here is the caller graph for this function:



9.20.4.34 `gst_mqtt_src_init()`

```
static void gst_mqtt_src_init (
    GstMqttSrc * self ) [static]
```

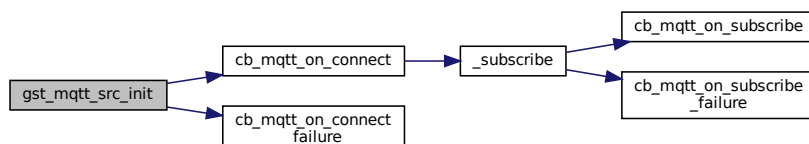
Initialize GstMqttSrc object.

Function definitions init mqttsrc properties

init private member variables

Definition at line 169 of file mqttsrc.c.

Here is the call graph for this function:



9.20.4.35 `gst_mqtt_src_is_seekable()`

```
static gboolean gst_mqtt_src_is_seekable (  
    GstBaseSrc * basesrc ) [static]
```

Check if source supports seeking.

Note

Seeking is not supported since this element handles live subscription data.

Definition at line 700 of file `mqttsrc.c`.

Here is the caller graph for this function:



9.20.4.36 `gst_mqtt_src_query()`

```
static gboolean gst_mqtt_src_query (  
    GstBaseSrc * basesrc,  
    GstQuery * query ) [static]
```

An implementation of the `GstBaseSrc` vmethod that handles queries.

The second argument of `gst_query_set_latency` should be always `TRUE`.

Definition at line 819 of file `mqttsrc.c`.

Here is the caller graph for this function:



9.20.4.37 `gst_mqtt_src_renegotiate()`

```
static gboolean gst_mqtt_src_renegotiate (  
    GstBaseSrc * basesrc ) [static]
```

Do negotiation procedure again if it needed.

Definition at line 624 of file mqttsrc.c.

Here is the caller graph for this function:



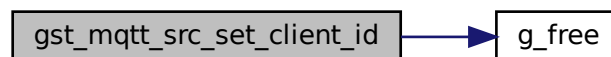
9.20.4.38 `gst_mqtt_src_set_client_id()`

```
static void gst_mqtt_src_set_client_id (  
    GstMqttSrc * self,  
    const gchar * id ) [static]
```

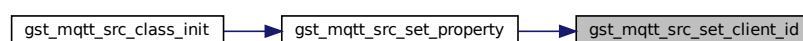
Setter for the 'client-id' property.

Definition at line 911 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



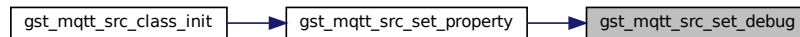
9.20.4.39 `gst_mqtt_src_set_debug()`

```
static void gst_mqtt_src_set_debug (
    GstMqttSrc * self,
    const gboolean flag ) [static]
```

Setter for the 'debug' property.

Definition at line 874 of file `mqttsrc.c`.

Here is the caller graph for this function:



9.20.4.40 `gst_mqtt_src_set_host_address()`

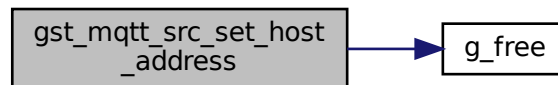
```
static void gst_mqtt_src_set_host_address (
    GstMqttSrc * self,
    const gchar * addr ) [static]
```

Setter for the 'host' property.

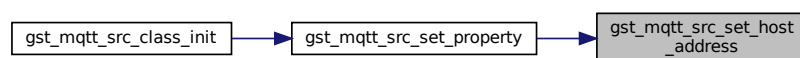
Todo Handle the case where the `addr` is changed at runtime

Definition at line 930 of file `mqttsrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



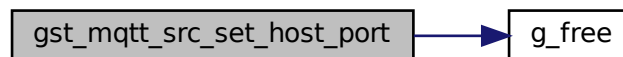
9.20.4.41 `gst_mqtt_src_set_host_port()`

```
static void gst_mqtt_src_set_host_port (
    GstMqttSrc * self,
    const gchar * port ) [static]
```

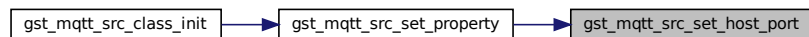
Setter for the 'port' property.

Definition at line 952 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



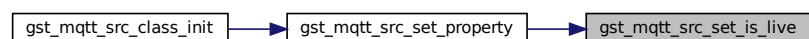
9.20.4.42 `gst_mqtt_src_set_is_live()`

```
static void gst_mqtt_src_set_is_live (
    GstMqttSrc * self,
    const gboolean flag ) [static]
```

Setter for the 'is-live' property.

Definition at line 892 of file mqttsrc.c.

Here is the caller graph for this function:



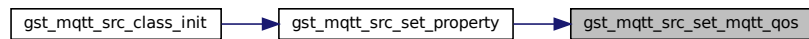
9.20.4.43 `gst_mqtt_src_set_mqtt_qos()`

```
static void gst_mqtt_src_set_mqtt_qos (
    GstMqttSrc * self,
    const gint qos ) [static]
```

Setter for the 'mqtt-qos' property.

Definition at line 1044 of file `mqttsrc.c`.

Here is the caller graph for this function:



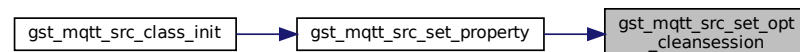
9.20.4.44 `gst_mqtt_src_set_opt_cleansession()`

```
static void gst_mqtt_src_set_opt_cleansession (
    GstMqttSrc * self,
    const gboolean val ) [static]
```

Setter for the 'cleansession' property.

Definition at line 1008 of file `mqttsrc.c`.

Here is the caller graph for this function:



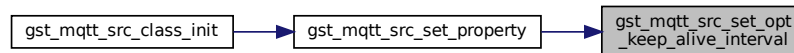
9.20.4.45 `gst_mqtt_src_set_opt_keep_alive_interval()`

```
static void gst_mqtt_src_set_opt_keep_alive_interval (
    GstMqttSrc * self,
    const gint num ) [static]
```

Setter for the 'keep-alive-interval' property.

Definition at line 1026 of file mqttsrc.c.

Here is the caller graph for this function:



9.20.4.46 `gst_mqtt_src_set_property()`

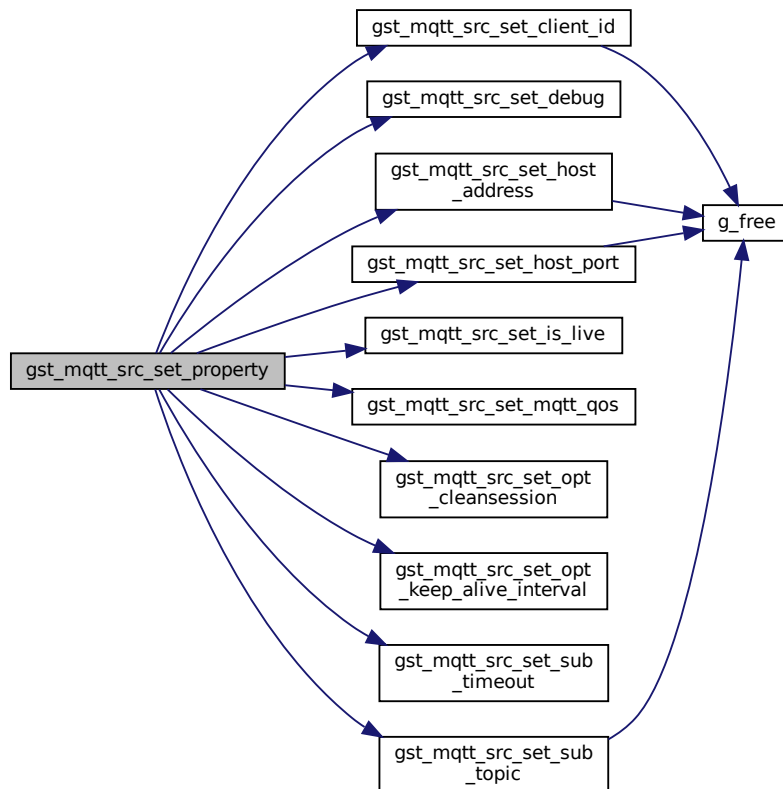
```
static void gst_mqtt_src_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

The setter for the mqttsrc's properties.

Function prototype declarations

Definition at line 316 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.20.4.47 `gst_mqtt_src_set_sub_timeout()`

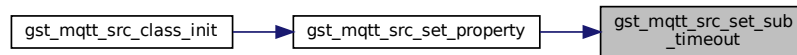
```

static void gst_mqtt_src_set_sub_timeout (
    GstMqttSrc * self,
    const gint64 t ) [static]
  
```

Setter for the 'sub-timeout' property.

Definition at line 971 of file mqttsrc.c.

Here is the caller graph for this function:



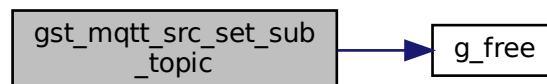
9.20.4.48 `gst_mqtt_src_set_sub_topic()`

```
static void gst_mqtt_src_set_sub_topic (  
    GstMqttSrc * self,  
    const gchar * topic ) [static]
```

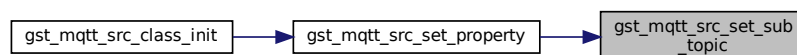
Setter for the 'sub-topic' property.

Definition at line 989 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.20.4.49 `gst_mqtt_src_start()`

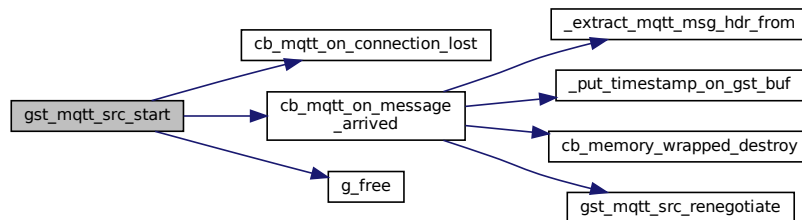
```
static gboolean gst_mqtt_src_start (
    GstBaseSrc * basesrc ) [static]
```

Start mqttsrc, called when state changed null to ready.

Todo Support other persistence mechanisms MQTTCLIENT_PERSISTENCE_NONE: A memory-based persistence mechanism MQTTCLIENT_PERSISTENCE_DEFAULT: The default file system-based persistence mechanism MQTTCLIENT_PERSISTENCE_USER: An application-specific persistence mechanism

Definition at line 522 of file mqttsrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



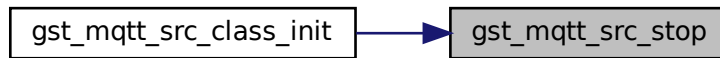
9.20.4.50 `gst_mqtt_src_stop()`

```
static gboolean gst_mqtt_src_stop (
    GstBaseSrc * basesrc ) [static]
```

Stop mqttsrc, called when state changed ready to null.

Definition at line 583 of file mqttsrc.c.

Here is the caller graph for this function:



9.20.5 Variable Documentation

9.20.5.1 DEFAULT_MQTT_CLIENT_ID

```
const gchar DEFAULT_MQTT_CLIENT_ID[] [static]
```

Initial value:

```
=  
"$HOSTNAME_$PID^[0-9][0-9]?$|^255$"
```

Definition at line 73 of file mqttsrc.c.

9.20.5.2 DEFAULT_MQTT_CLIENT_ID_FORMAT

```
const gchar DEFAULT_MQTT_CLIENT_ID_FORMAT[] = "%s_%u_src%" [static]
```

Definition at line 75 of file mqttsrc.c.

9.20.5.3 DEFAULT_MQTT_HOST_ADDRESS

```
const gchar DEFAULT_MQTT_HOST_ADDRESS[] = "127.0.0.1" [static]
```

Definition at line 70 of file mqttsrc.c.

9.20.5.4 DEFAULT_MQTT_HOST_PORT

```
const gchar DEFAULT_MQTT_HOST_PORT[] = "1883" [static]
```

Definition at line 71 of file mqttsrc.c.

9.20.5.5 src_client_id

```
guint8 src_client_id = 0 [static]
```

Definition at line 69 of file mqttsrc.c.

9.20.5.6 src_pad_template

```
GstStaticPadTemplate src_pad_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC, GST_PAD_ALWAYS, GST_STATIC_CAPS_ANY)
```

Definition at line 31 of file mqttsrc.c.

9.20.5.7 TAG_ERR_MQTTSRC

```
const gchar TAG_ERR_MQTTSRC[] = "ERROR: MQTTSrc" [static]
```

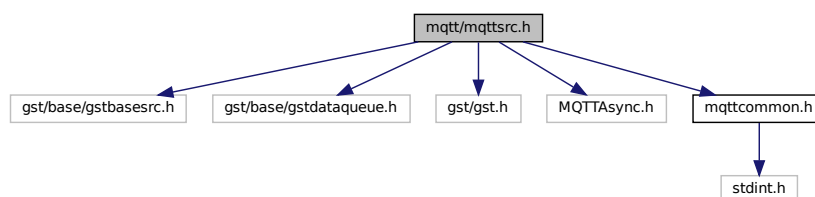
Definition at line 72 of file mqttsrc.c.

9.21 mqtt/mqttsrc.h File Reference

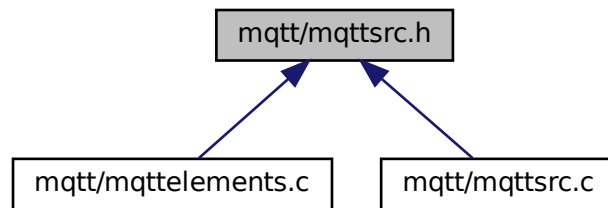
Subscribe a MQTT topic and push incoming data to the GStreamer pipeline.

```
#include <gst/base/gstbasesrc.h>
#include <gst/base/gstdataqueue.h>
#include <gst/gst.h>
#include <MQTTAsync.h>
#include "mqttcommon.h"
```

Include dependency graph for mqttsrc.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstMqttSrc](#)
GstMqttSrc data structure.
- struct [_GstMqttSrcClass](#)
GstMqttSrcClass data structure.

Macros

- #define [GST_TYPE_MQTT_SRC](#) ([gst_mqtt_src_get_type\(\)](#))
- #define [GST_MQTT_SRC\(obj\)](#) (G_TYPE_CHECK_INSTANCE_CAST ((obj), [GST_TYPE_MQTT_SRC](#), [GstMqttSrc](#)))
- #define [GST_IS_MQTT_SRC\(obj\)](#) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), [GST_TYPE_MQTT_SRC](#)))
- #define [GST_MQTT_SRC_CAST\(obj\)](#) (([GstMqttSrc *](#)) obj)
- #define [GST_MQTT_SRC_CLASS\(klass\)](#) (G_TYPE_CHECK_CLASS_CAST ((klass), [GST_TYPE_MQTT_SRC](#), [GstMqttSrcClass](#)))
- #define [GST_IS_MQTT_SRC_CLASS\(klass\)](#) (G_TYPE_CHECK_CLASS_TYPE ((klass), [GST_TYPE_MQTT_SRC](#)))

Typedefs

- typedef struct [_GstMqttSrc](#) [GstMqttSrc](#)
- typedef struct [_GstMqttSrcClass](#) [GstMqttSrcClass](#)

Functions

- GType [gst_mqtt_src_get_type](#) (void)

9.21.1 Detailed Description

Subscribe a MQTT topic and push incoming data to the GStreamer pipeline.

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

08 Mar 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.21.2 Macro Definition Documentation

9.21.2.1 GST_IS_MQTT_SRC

```
#define GST_IS_MQTT_SRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_MQTT_SRC))
```

Definition at line 29 of file mqttsrc.h.

9.21.2.2 GST_IS_MQTT_SRC_CLASS

```
#define GST_IS_MQTT_SRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_MQTT_SRC))
```

Definition at line 35 of file mqttsrc.h.

9.21.2.3 GST_MQTT_SRC

```
#define GST_MQTT_SRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_MQTT_SRC, GstMqttSrc))
```

Definition at line 27 of file mqttsrc.h.

9.21.2.4 GST_MQTT_SRC_CAST

```
#define GST_MQTT_SRC_CAST(  
    obj ) ((GstMqttSrc *) obj)
```

Definition at line 31 of file mqttsrc.h.

9.21.2.5 GST_MQTT_SRC_CLASS

```
#define GST_MQTT_SRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_MQTT_SRC, GstMqttSrcClass))
```

Definition at line 33 of file mqttsrc.h.

9.21.2.6 GST_TYPE_MQTT_SRC

```
#define GST_TYPE_MQTT_SRC (gst_mqtt_src_get_type())
```

Definition at line 25 of file mqttsrc.h.

9.21.3 Typedef Documentation

9.21.3.1 GstMqttSrc

```
typedef struct _GstMqttSrc GstMqttSrc
```

Definition at line 38 of file mqttsrc.h.

9.21.3.2 GstMqttSrcClass

```
typedef struct _GstMqttSrcClass GstMqttSrcClass
```

Definition at line 39 of file mqttsrc.h.

9.21.4 Function Documentation

9.21.4.1 `gst_mqtt_src_get_type()`

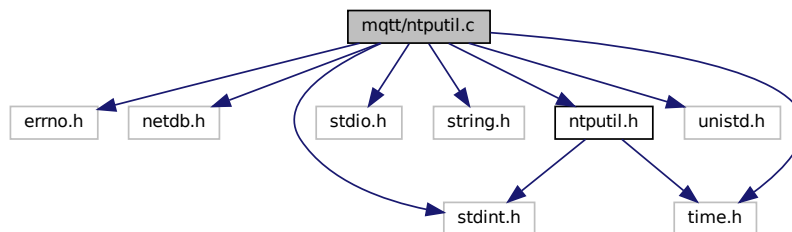
```
GType gst_mqtt_src_get_type (
    void )
```

9.22 `mqtt/nputil.c` File Reference

NTP utility functions.

```
#include <errno.h>
#include <netdb.h>
#include <stdint.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "nputil.h"
```

Include dependency graph for `nputil.c`:



Classes

- [struct `_ntp_timestamp_t`](#)
A custom data type to represent NTP timestamp format.
- [struct `_ntp_packet_t`](#)
A custom data type to represent NTP packet header format.

Typedefs

- [typedef struct `_ntp_timestamp_t` `ntp_timestamp_t`](#)
A custom data type to represent NTP timestamp format.
- [typedef struct `_ntp_packet_t` `ntp_packet_t`](#)
A custom data type to represent NTP packet header format.

Functions

- [uint32_t `_convert_to_host_byte_order`](#) (uint32_t in)
Wrapper function of `ntohl`.
- [int64_t `nputil_get_epoch`](#) (uint32_t hnums, char **hnames, uint16_t *ports)
Get NTP timestamps from the given or public NTP servers.

Variables

- const uint64_t `NTPUTIL_TIMESTAMP_DELTA` = 2208988800ULL
- const double `NTPUTIL_MAX_FRAC_DOUBLE` = 4294967295.0L
- const int64_t `NTPUTIL_SEC_TO_USEC_MULTIPLIER` = 1000000
- const char `NTPUTIL_DEFAULT_HNAME` [] = "pool.ntp.org"
- const uint16_t `NTPUTIL_DEFAULT_PORT` = 123

9.22.1 Detailed Description

NTP utility functions.

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

16 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

Todo Need to support caching and polling timer mechanism

9.22.2 Typedef Documentation

9.22.2.1 ntp_packet_t

```
typedef struct _ntp_packet_t ntp_packet_t
```

A custom data type to represent NTP packet header format.

NTP Packet Header Format (<https://www.ietf.org/rfc/rfc5905.txt> p.18) 0 1 2 3 0 1 2 3 4 5 6 7 8
 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 +--+ |LI
 | VN |Mode | Stratum | Poll | Precision | +--+ | Root
 Delay | +--+ | Root Dispersion | +--+--+--+--+--+
 +--+ | Reference ID | +--+--+--+--+--+--+--+--+--+--+
 +--+ | |

- Reference Timestamp (64) + | | +--+ | |
- Origin Timestamp (64) + | | +--+ | |
- Receive Timestamp (64) + | | +--+ | |
- Transmit Timestamp (64) + | | +--+ | | . . .
 Extension Field 1 (variable) . . . | | +--+ | |
 . . . Extension Field 2 (variable) . . . | | +--+
 +-+ | Key Identifier | +--+ | | | dgst (128) | | |
 +--+

9.22.2.2 ntp_timestamp_t

```
typedef struct _ntp_timestamp_t ntp_timestamp_t
```

A custom data type to represent NTP timestamp format.

NTP Timestamp Format (<https://www.ietf.org/rfc/rfc5905.txt> p.12)

```

0 1 2 3 0 1 2 3 4 5 6 7 8
9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 ++++++
| Seconds | ++++++ | Fraction | ++++++
+++++
```

9.22.3 Function Documentation

9.22.3.1 _convert_to_host_byte_order()

```
uint32_t _convert_to_host_byte_order (
    uint32_t in )
```

Wrapper function of ntohl.

Converting network byte order to host byte order.

Definition at line 125 of file ntputil.c.

Here is the caller graph for this function:



9.22.3.2 ntputil_get_epoch()

```
int64_t ntputil_get_epoch (
    uint32_t hnums,
    char ** hnames,
    uint16_t * ports )
```

Get NTP timestamps from the given or public NTP servers.

Parameters

in	<i>hnums</i>	A number of hostname and port pairs. If 0 is given, the NTP server pool will be used.
in	<i>hnames</i>	A list of hostname
in	<i>ports</i>	A list of port

Returns

an Unix epoch time as microseconds on success, negative values on error

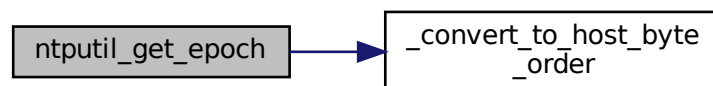
Note

`ntp_timestamp_t recv_ts` in `ntp_packet_t` means the timestamp as the packet left the NTP server. 'sec' corresponds to the seconds passed since 1900 and 'frac' is needed to convert seconds to smaller units of a second such as microseconds. Note that the bit/byte order of those data should be converted to the host's endianness.

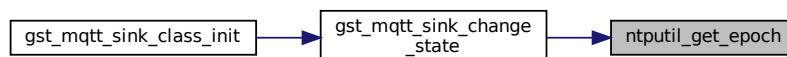
NTP uses an epoch of January 1, 1900 while the Unix epoch is the number of seconds that have elapsed since January 1, 1970. For this reason, we subtract 70 years worth of seconds from the seconds since 1900

Definition at line 140 of file `nputil.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.22.4 Variable Documentation****9.22.4.1 NPUTIL_DEFAULT_HNAME**

```
const char NPUTIL_DEFAULT_HNAME[] = "pool.ntp.org"
```

Definition at line 118 of file `nputil.c`.

9.22.4.2 NTPUTIL_DEFAULT_PORT

```
const uint16_t NTPUTIL_DEFAULT_PORT = 123
```

Definition at line 119 of file ntputil.c.

9.22.4.3 NTPUTIL_MAX_FRAC_DOUBLE

```
const double NTPUTIL_MAX_FRAC_DOUBLE = 4294967295.0L
```

Definition at line 116 of file ntputil.c.

9.22.4.4 NTPUTIL_SEC_TO_USEC_MULTIPLIER

```
const int64_t NTPUTIL_SEC_TO_USEC_MULTIPLIER = 1000000
```

Definition at line 117 of file ntputil.c.

9.22.4.5 NTPUTIL_TIMESTAMP_DELTA

```
const uint64_t NTPUTIL_TIMESTAMP_DELTA = 2208988800ULL
```

Definition at line 115 of file ntputil.c.

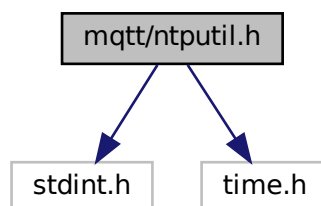
9.23 mqtt/ntputil.h File Reference

A header file of NTP utility functions.

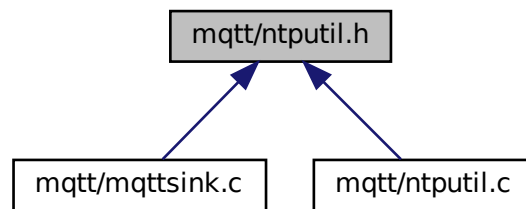
```
#include <stdint.h>
```

```
#include <time.h>
```

Include dependency graph for ntputil.h:



This graph shows which files directly or indirectly include this file:



Functions

- `int64_t nputil_get_epoch` (`uint32_t hnums`, `char **hnames`, `uint16_t *ports`)
Get NTP timestamps from the given or public NTP servers.
- `uint32_t _convert_to_host_byte_order` (`uint32_t in`)
Converting network byte order to host byte order.

9.23.1 Detailed Description

A header file of NTP utility functions.

Copyright (C) 2021 Wook Song wook16.song@samsung.com

Date

28 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.23.2 Function Documentation

9.23.2.1 `_convert_to_host_byte_order()`

```
uint32_t _convert_to_host_byte_order (
    uint32_t in )
```

Converting network byte order to host byte order.

Note

There is a problem in mocking `ntohl` in GMock, so the wrapper function is temporarily used.

Converting network byte order to host byte order.

Definition at line 125 of file `ntputil.c`.

Here is the caller graph for this function:



9.23.2.2 `ntputil_get_epoch()`

```
int64_t ntputil_get_epoch (
    uint32_t hnums,
    char ** hnames,
    uint16_t * ports )
```

Get NTP timestamps from the given or public NTP servers.

Parameters

in	<i>hnums</i>	A number of hostname and port pairs. If 0 is given, the NTP server pool will be used.
in	<i>hnames</i>	A list of hostname
in	<i>ports</i>	A list of port

Returns

an Unix epoch time as microseconds on success, negative values on error

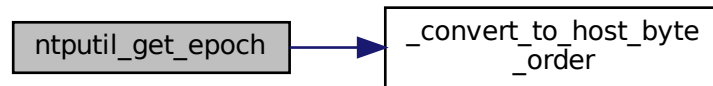
Note

`ntp_timestamp_t` `recv_ts` in `ntp_packet_t` means the timestamp as the packet left the NTP server. 'sec' corresponds to the seconds passed since 1900 and 'frac' is needed to convert seconds to smaller units of a second such as microseconds. Note that the bit/byte order of those data should be converted to the host's endianness.

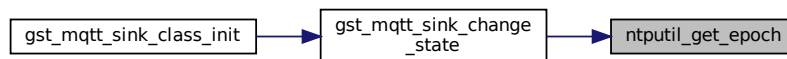
NTP uses an epoch of January 1, 1900 while the Unix epoch is the number of seconds that have elapsed since January 1, 1970. For this reason, we subtract 70 years worth of seconds from the seconds since 1900

Definition at line 140 of file ntputil.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.24 nntstreamer/elements/gsttensor_aggregator.c File Reference

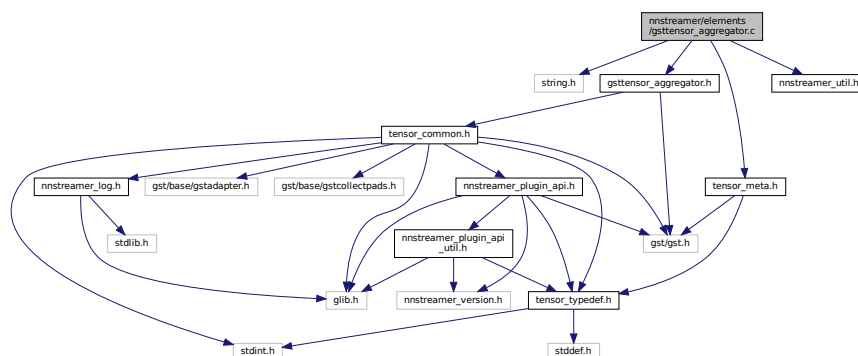
GStreamer plugin to aggregate tensor stream.

```

#include <string.h>
#include "gsttensor_aggregator.h"
#include "tensor_meta.h"
#include <nntstreamer_util.h>

```

Include dependency graph for `gsttensor_aggregator.c`:



Macros

- `#define DBG (!self->silent)`
Macro for debug mode.
- `#define silent_debug_config(self, c, msg)`
- `#define GST_CAT_DEFAULT gst_tensor_aggregator_debug`
- `#define DEFAULT_SILENT TRUE`
Flag to print minimized log.
- `#define DEFAULT_FRAMES_IN 1`
The number of frames in input buffer.
- `#define DEFAULT_FRAMES_OUT 1`
The number of frames in output buffer.
- `#define DEFAULT_FRAMES_FLUSH 0`
The number of frames to flush.
- `#define DEFAULT_FRAMES_DIMENSION (NNS_TENSOR_RANK_LIMIT - 1)`
The dimension index of frames in configured tensor.
- `#define DEFAULT_CONCAT TRUE`
Flag to concatenate output buffer.
- `#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_WITH_NUM ("1")`
Template caps string for pads.
- `#define gst_tensor_aggregator_parent_class parent_class`

Enumerations

- enum {
 PROP_0, PROP_FRAMES_IN, PROP_FRAMES_OUT, PROP_FRAMES_FLUSH,
 PROP_FRAMES_DIMENSION, PROP_CONCAT, PROP_SILENT }
tensor_aggregator properties

Functions

- `GST_DEBUG_CATEGORY_STATIC (gst_tensor_aggregator_debug)`
- `G_DEFINE_TYPE (GstTensorAggregator, gst_tensor_aggregator, GST_TYPE_ELEMENT)`
- static void `gst_tensor_aggregator_finalize` (GObject *object)
Function to finalize instance.
- static void `gst_tensor_aggregator_set_property` (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
Setter for tensor_aggregator properties.
- static void `gst_tensor_aggregator_get_property` (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
Getter for tensor_aggregator properties.
- static gboolean `gst_tensor_aggregator_sink_event` (GstPad *pad, GstObject *parent, GstEvent *event)
This function handles sink events.
- static gboolean `gst_tensor_aggregator_sink_query` (GstPad *pad, GstObject *parent, GstQuery *query)
This function handles sink pad query.
- static gboolean `gst_tensor_aggregator_src_query` (GstPad *pad, GstObject *parent, GstQuery *query)
This function handles src pad query.
- static GstFlowReturn `gst_tensor_aggregator_chain` (GstPad *pad, GstObject *parent, GstBuffer *buf)
Chain function, this function does the actual processing.
- static GstStateChangeReturn `gst_tensor_aggregator_change_state` (GstElement *element, GstStateChange transition)

- Called to perform state change.*

 - static void [gst_tensor_aggregator_reset](#) ([GstTensorAggregator](#) *self)

Clear and reset data.

 - static [GstCaps](#) * [gst_tensor_aggregator_query_caps](#) ([GstTensorAggregator](#) *self, [GstPad](#) *pad, [GstCaps](#) *filter)

Get pad caps for caps negotiation.

 - static gboolean [gst_tensor_aggregator_parse_caps](#) ([GstTensorAggregator](#) *self, const [GstCaps](#) *caps)

Parse caps and set tensor info.

 - static void [gst_tensor_aggregator_class_init](#) ([GstTensorAggregatorClass](#) *klass)

Initialize the tensor_aggregator's class.

 - static void [gst_tensor_aggregator_init](#) ([GstTensorAggregator](#) *self)

Initialize tensor_aggregator element.

 - static [GstAdapter](#) * [gst_tensor_aggregator_get_adapter](#) ([GstTensorAggregator](#) *self, [GstBuffer](#) *buf)

Internal function to get adapter.

 - static gboolean [gst_tensor_aggregator_check_concat_axis](#) ([GstTensorAggregator](#) *self, const [GstTensorInfo](#) *info)

Check tensor dimension and axis to concatenate data.

 - static gboolean [gst_tensor_aggregator_concat](#) ([GstTensorAggregator](#) *self, [GstBuffer](#) *outbuf, const [GstTensorInfo](#) *info)

Change the data in buffer with given axis.

 - static [GstFlowReturn](#) [gst_tensor_aggregator_push](#) ([GstTensorAggregator](#) *self, [GstBuffer](#) *outbuf, gsize frame_size)

Push the buffer to source pad. (Concatenate the buffer if needed)

Variables

- static [GstStaticPadTemplate](#) [sink_template](#)

Template for sink pad.

 - static [GstStaticPadTemplate](#) [src_template](#)

Template for src pad.

9.24.1 Detailed Description

GStreamer plugin to aggregate tensor stream.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details. SECTION:element-tensor_aggregator

Date

29 August 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung jy1210.jung@samsung.com

Bug No known bugs except for NYI items

9.24.2 Macro Definition Documentation

9.24.2.1 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_WITH_NUM ("1")
```

Template caps string for pads.

Definition at line 106 of file gsttensor_aggregator.c.

9.24.2.2 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

Definition at line 42 of file gsttensor_aggregator.c.

9.24.2.3 DEFAULT_CONCAT

```
#define DEFAULT_CONCAT TRUE
```

Flag to concatenate output buffer.

Definition at line 101 of file gsttensor_aggregator.c.

9.24.2.4 DEFAULT_FRAMES_DIMENSION

```
#define DEFAULT_FRAMES_DIMENSION (NNS_TENSOR_RANK_LIMIT - 1)
```

The dimension index of frames in configured tensor.

Definition at line 96 of file gsttensor_aggregator.c.

9.24.2.5 DEFAULT_FRAMES_FLUSH

```
#define DEFAULT_FRAMES_FLUSH 0
```

The number of frames to flush.

Definition at line 91 of file gsttensor_aggregator.c.

9.24.2.6 DEFAULT_FRAMES_IN

```
#define DEFAULT_FRAMES_IN 1
```

The number of frames in input buffer.

Definition at line 81 of file gsttensor_aggregator.c.

9.24.2.7 DEFAULT_FRAMES_OUT

```
#define DEFAULT_FRAMES_OUT 1
```

The number of frames in output buffer.

Definition at line 86 of file gsttensor_aggregator.c.

9.24.2.8 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 76 of file gsttensor_aggregator.c.

9.24.2.9 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_aggregator_debug
```

Definition at line 57 of file gsttensor_aggregator.c.

9.24.2.10 gst_tensor_aggregator_parent_class

```
#define gst_tensor_aggregator_parent_class parent_class
```

Definition at line 124 of file gsttensor_aggregator.c.

9.24.2.11 silent_debug_config

```
#define silent_debug_config(
    self,
    c,
    msg )
```

Value:

```
do { \
  if (DBG) { \
    if (c) { \
      gchar *dim_str; \
      dim_str = gst_tensor_get_dimension_string ((c)->info.info[0].dimension); \
      GST_DEBUG_OBJECT (self, msg " type=%d dim=%s rate=%d/%d", (c)->info.info[0].type, dim_str,
        (c)->rate_n, (c)->rate_d); \
      g_free (dim_str); \
    } \
  } \
} while (0)
```

Definition at line 45 of file gsttensor_aggregator.c.

9.24.3 Enumeration Type Documentation

9.24.3.1 anonymous enum

anonymous enum

tensor_aggregator properties

Enumerator

PROP_0	
PROP_FRAMES_IN	
PROP_FRAMES_OUT	
PROP_FRAMES_FLUSH	
PROP_FRAMES_DIMENSION	
PROP_CONCAT	
PROP_SILENT	

Definition at line 62 of file gsttensor_aggregator.c.

9.24.4 Function Documentation

9.24.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorAggregator ,
```



```
gst_tensor_aggregator ,
GST_TYPE_ELEMENT )
```

9.24.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_aggregator_debug )
```

9.24.4.3 gst_tensor_aggregator_chain()

```
static GstFlowReturn gst_tensor_aggregator_chain (
    GstPad * pad,
    GstObject * parent,
    GstBuffer * buf ) [static]
```

Chain function, this function does the actual processing.

push the incoming buffer (do concat if needed)

supposed same duration for incoming buffer

Update timestamp. If frames-in is larger then frames-out, the same timestamp (pts and dts) would be returned.

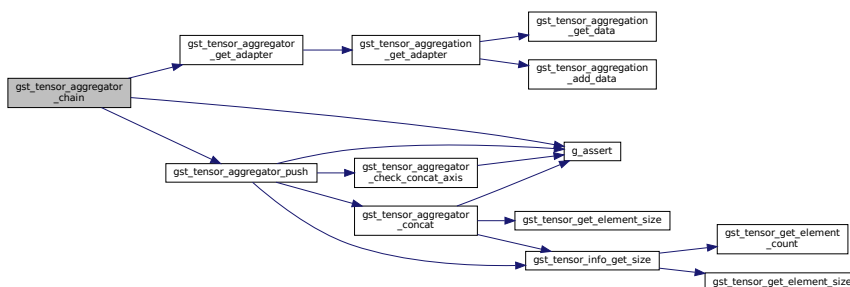
set timestamp

flush data

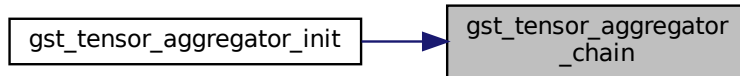
Todo flush data Invalid state, tried to flush large size. We have to determine how to handle this case. (flush the out-size or all available bytes) Now all available bytes in adapter will be flushed.

Definition at line 835 of file gsttensor_aggregator.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.24.4.4 `gst_tensor_aggregator_change_state()`

```

static GstStateChangeReturn gst_tensor_aggregator_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
  
```

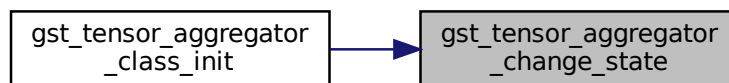
Called to perform state change.

Definition at line 947 of file `gsttensor_aggregator.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.24.4.5 `gst_tensor_aggregator_check_concat_axis()`

```

static gboolean gst_tensor_aggregator_check_concat_axis (
    GstTensorAggregator * self,
    const GstTensorInfo * info ) [static]
  
```

Check tensor dimension and axis to concatenate data.

Parameters

<i>self</i>	this pointer to GstTensorAggregator
<i>info</i>	tensor info for one frame

Returns

True if needed to concatenate

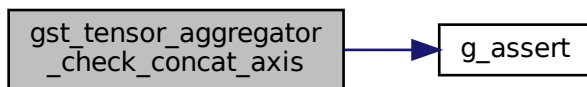
Internal error. Caller should've checked it

Check condition to concatenate data.

concatenate data

Definition at line 540 of file gsttensor_aggregator.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.24.4.6 gst_tensor_aggregator_class_init()

```

static void gst_tensor_aggregator_class_init (
    GstTensorAggregatorClass * klass ) [static]
  
```

Initialize the tensor_aggregator's class.

GstTensorAggregator::frames-in:

The number of frames in incoming buffer. GstTensorAggregator itself cannot get the frames in buffer. (buffer is a single tensor instance) GstTensorAggregator calculates the size of single frame with this property.

GstTensorAggregator::frames-out:

The number of frames in outgoing buffer. (buffer is a single tensor instance) GstTensorAggregator calculates the size of outgoing frames and pushes a buffer to source pad.

GstTensorAggregator::frames-flush:

The number of frames to flush. GstTensorAggregator flushes the bytes (N frames) in GstAdapter after pushing a buffer. If set 0 (default value), all outgoing frames will be flushed.

GstTensorAggregator::frames-dim:

The dimension index of frames in tensor. If frames-in and frames-out are different, GstTensorAggregator has to change the dimension of tensor. With this property, GstTensorAggregator changes the out-caps.

GstTensorAggregator::concat:

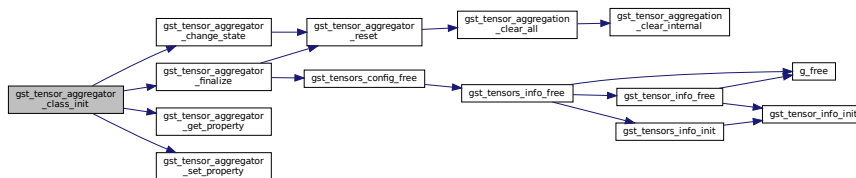
The flag to concatenate output buffer. If concat is true and frames-out is larger than 1, GstTensorAggregator will concatenate the output buffer with the axis frames-dim.

GstTensorAggregator::silent:

The flag to enable/disable debugging messages.

Definition at line 155 of file gsttensor_aggregator.c.

Here is the call graph for this function:



9.24.4.7 gst_tensor_aggregator_concat()

```

static gboolean gst_tensor_aggregator_concat (
    GstTensorAggregator * self,
    GstBuffer * outbuf,
    const GstTensorInfo * info ) [static]
  
```

Change the data in buffer with given axis.

Parameters

<i>self</i>	this pointer to GstTensorAggregator
<i>outbuf</i>	buffer to be concatenated
<i>info</i>	tensor info for one frame

Internal error

Concatenate output buffer with given axis (frames-dim) If frames-dim is equal to (NNS_TENSOR_RANK_LIMIT - 1), nothing to do. (In this case, this function will not be called. See `gst_tensor_aggregator_check_concat_axis ()`)

Ex1) concatenate 2 frames with dimension 3:4:2:1

```
frame 1 [[ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] ] ]
```

```
frame 2 [ [ [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ]
```

```
1-1. result with frames-dim 3 (3:4:2:2) [ [ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] ] ] [ [ [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ]
```

```
1-2. result with frames-dim 2 (3:4:4:1) [ [ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] ] ] [ [ [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ]
```

```
1-3. result with frames-dim 1 (3:8:2:1) [ [ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ]
```

```
1-4. result with frames-dim 0 (6:4:2:1) [ [ [1101 1102 1103 2101 2102 2103] [1104 1105 1106 2104 2105 2106] [1107 1108 1109 2107 2108 2109] [1110 1111 1112 2110 2111 2112] ] [ [1113 1114 1115 2113 2114 2115] [1116 1117 1118 2116 2117 2118] [1119 1120 1121 2119 2120 2121] [1122 1123 1124 2122 2123 2124] ] ]
```

Ex2) concatenate 2 frames with dimension 3:4:2:2

```
frame 1 [[ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] ] ] [ [ [1201 1202 1203] [1204 1205 1206] [1207 1208 1209] [1210 1211 1212] ] [ [1213 1214 1215] [1216 1217 1218] [1219 1220 1221] [1222 1223 1224] ] ]
```

```
frame 2 [ [ [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ] [ [ [2201 2202 2203] [2204 2205 2206] [2207 2208 2209] [2210 2211 2212] ] [ [2213 2214 2215] [2216 2217 2218] [2219 2220 2221] [2222 2223 2224] ] ]
```

```
2-1. result with frames-dim 3 (3:4:2:4) [ [ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] ] ] [ [ [1201 1202 1203] [1204 1205 1206] [1207 1208 1209] [1210 1211 1212] ] [ [1213 1214 1215] [1216 1217 1218] [1219 1220 1221] [1222 1223 1224] ] ] [ [ [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ] [ [ [2201 2202 2203] [2204 2205 2206] [2207 2208 2209] [2210 2211 2212] ] [ [2213 2214 2215] [2216 2217 2218] [2219 2220 2221] [2222 2223 2224] ] ]
```

```
2-2. result with frames-dim 2 (3:4:4:2) [ [ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] ] ] [ [ [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ] [ [ [1201 1202 1203] [1204 1205 1206] [1207 1208 1209] [1210 1211 1212] ] [ [1213 1214 1215] [1216 1217 1218] [1219 1220 1221] [1222 1223 1224] ] ] [ [ [2201 2202 2203] [2204 2205 2206] [2207 2208 2209] [2210 2211 2212] ] [ [2213 2214 2215] [2216 2217 2218] [2219 2220 2221] [2222 2223 2224] ] ]
```

```
2-3. result with frames-dim 1 (3:8:2:2) [ [ [1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112] ] [ [1113 1114 1115] [1116 1117 1118] [1119 1120 1121] [1122 1123 1124] [2113 2114 2115] [2116 2117 2118] [2119 2120 2121] [2122 2123 2124] ] ] [ [ [1201 1202 1203] [1204 1205 1206] [1207 1208 1209] [1210 1211 1212] [2201 2202 2203] [2204 2205 2206] [2207 2208 2209] [2210 2211 2212] ] [ [1213 1214 1215] [1216 1217 1218] [1219 1220 1221] [1222 1223 1224] [2213 2214 2215] [2216 2217 2218] [2219 2220 2221] [2222 2223 2224] ] ]
```

2-4. result with frames-dim 0 (6:4:2:2) [[[1101 1102 1103 2101 2102 2103] [1104 1105 1106 2104 2105 2106] [1107 1108 1109 2107 2108 2109] [1110 1111 1112 2110 2111 2112]] [[1113 1114 1115 2113 2114 2115] [1116 1117 1118 2116 2117 2118] [1119 1120 1121 2119 2120 2121] [1122 1123 1124 2122 2123 2124]]] [[[1201 1202 1203 2201 2202 2203] [1204 1205 1206 2204 2205 2206] [1207 1208 1209 2207 2208 2209] [1210 1211 1212 2210 2211 2212]] [[1213 1214 1215 2213 2214 2215] [1216 1217 1218 2216 2217 2218] [1219 1220 1221 2219 2220 2221] [1222 1223 1224 2222 2223 2224]]]]

Ex3) concatenate 2 frames with dimension 3:4:1:1

frame 1 [[[1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112]]]

frame 2 [[[2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112]]]

3-1. result with frames-dim 3 (3:4:1:2) [[[1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112]]] [[[2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112]]]

3-2. result with frames-dim 2 (3:4:2:1) [[[1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112]] [[2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112]]]

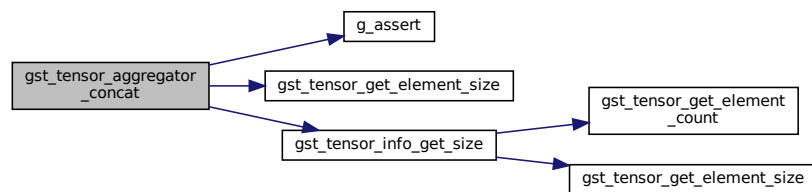
3-3. result with frames-dim 1 (3:8:1:1) [[[1101 1102 1103] [1104 1105 1106] [1107 1108 1109] [1110 1111 1112] [2101 2102 2103] [2104 2105 2106] [2107 2108 2109] [2110 2111 2112]]]

3-4. result with frames-dim 0 (6:4:1:1) [[[1101 1102 1103 2101 2102 2103] [1104 1105 1106 2104 2105 2106] [1107 1108 1109 2107 2108 2109] [1110 1111 1112 2110 2111 2112]]]

get block size

Definition at line 569 of file gsttensor_aggregator.c.

Here is the call graph for this function:



Here is the caller graph for this function:



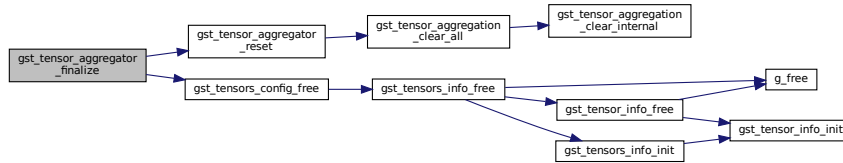
9.24.4.8 `gst_tensor_aggregator_finalize()`

```
static void gst_tensor_aggregator_finalize (
    GObject * object ) [static]
```

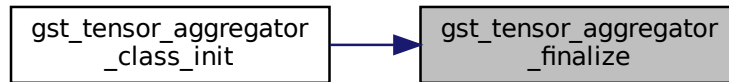
Function to finalize instance.

Definition at line 294 of file `gsttensor_aggregator.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



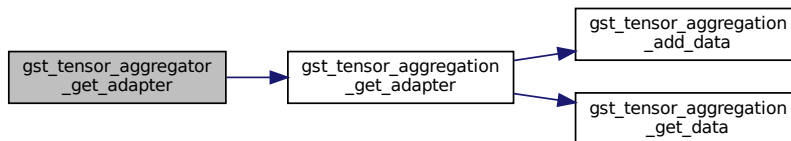
9.24.4.9 `gst_tensor_aggregator_get_adapter()`

```
static GstAdapter* gst_tensor_aggregator_get_adapter (
    GstTensorAggregator * self,
    GstBuffer * buf ) [static]
```

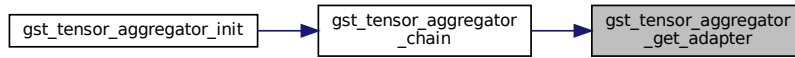
Internal function to get adapter.

Definition at line 521 of file `gsttensor_aggregator.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.24.4.10 `gst_tensor_aggregator_get_property()`

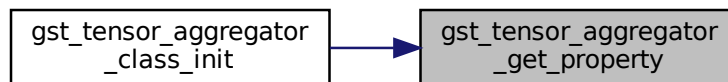
```

static void gst_tensor_aggregator_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
  
```

Getter for `tensor_aggregator` properties.

Definition at line 349 of file `gsttensor_aggregator.c`.

Here is the caller graph for this function:



9.24.4.11 `gst_tensor_aggregator_init()`

```

static void gst_tensor_aggregator_init (
    GstTensorAggregator * self ) [static]
  
```

Initialize `tensor_aggregator` element.

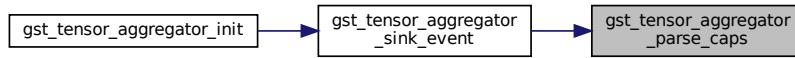
setup sink pad

setup src pad

init properties

Definition at line 254 of file `gsttensor_aggregator.c`.

Here is the caller graph for this function:



9.24.4.13 gst_tensor_aggregator_push()

```

static GstFlowReturn gst_tensor_aggregator_push (
    GstTensorAggregator * self,
    GstBuffer * outbuf,
    gsize frame_size ) [static]
  
```

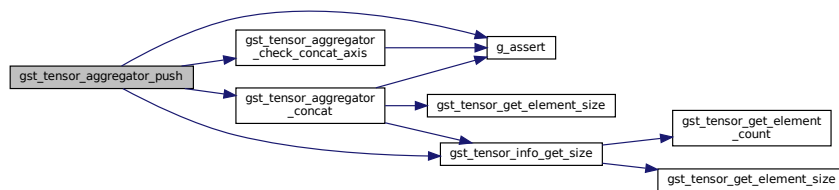
Push the buffer to source pad. (Concatenate the buffer if needed)

tensor info for one frame

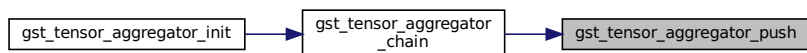
change data in buffer with given axis

Definition at line 805 of file gsttensor_aggregator.c.

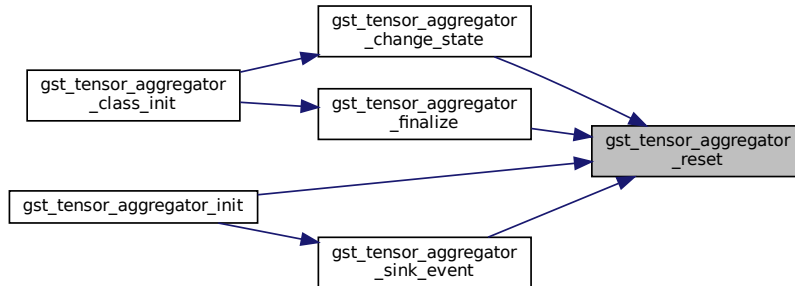
Here is the call graph for this function:



Here is the caller graph for this function:



Here is the caller graph for this function:



9.24.4.16 `gst_tensor_aggregator_set_property()`

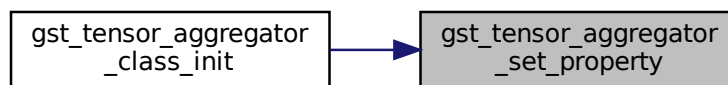
```

static void gst_tensor_aggregator_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

Setter for `tensor_aggregator` properties.

Definition at line 313 of file `gsttensor_aggregator.c`.

Here is the caller graph for this function:



9.24.4.17 `gst_tensor_aggregator_sink_event()`

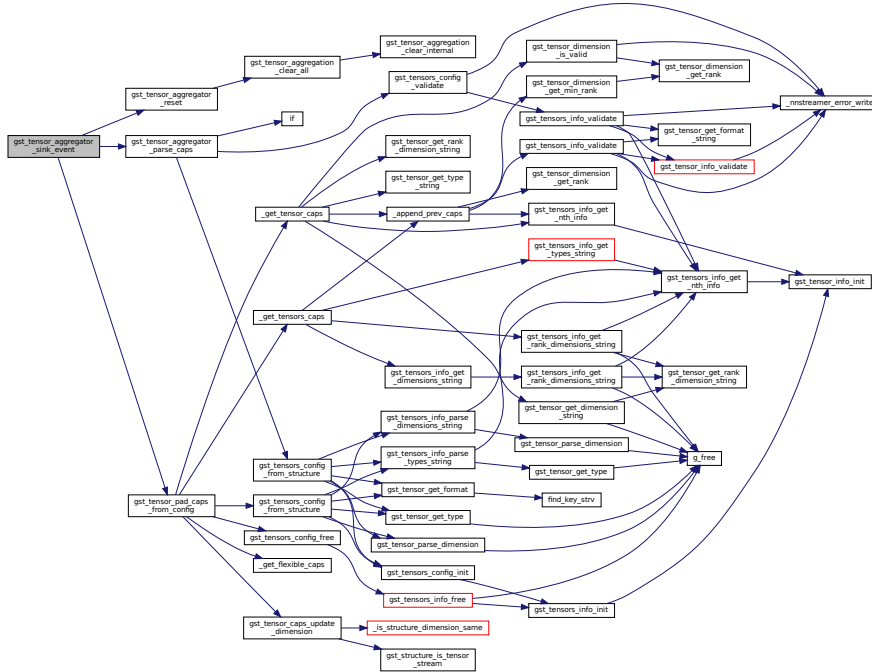
```

static gboolean gst_tensor_aggregator_sink_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
  
```

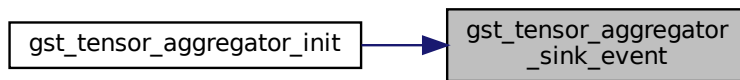
This function handles sink events.

Definition at line 385 of file gsttensor_aggregator.c.

Here is the call graph for this function:



Here is the caller graph for this function:



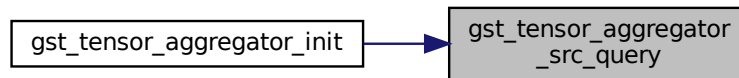
9.24.4.18 gst_tensor_aggregator_sink_query()

```
static gboolean gst_tensor_aggregator_sink_query (
    GstPad * pad,
    GstObject * parent,
    GstQuery * query ) [static]
```

This function handles sink pad query.

Definition at line 434 of file gsttensor_aggregator.c.

Here is the caller graph for this function:



9.24.5 Variable Documentation

9.24.5.1 sink_template

```
GstStaticPadTemplate sink_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

Template for sink pad.

Definition at line 111 of file `gsttensor_aggregator.c`.

9.24.5.2 src_template

```
GstStaticPadTemplate src_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

Template for src pad.

Definition at line 119 of file `gsttensor_aggregator.c`.

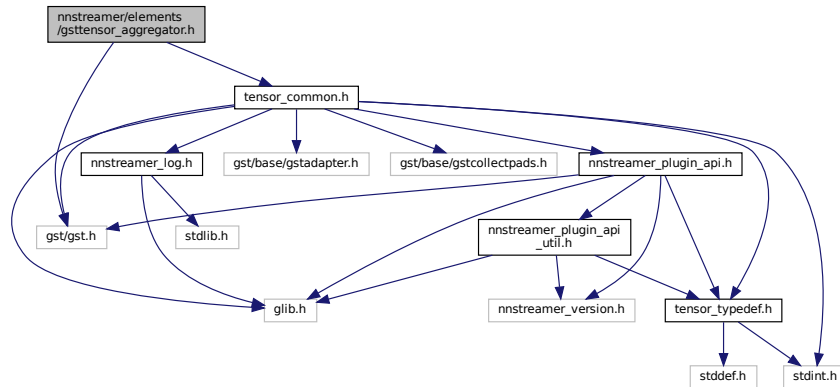
9.25 nnstreamer/elements/gsttensor_aggregator.h File Reference

GStreamer plugin to aggregate tensor stream.

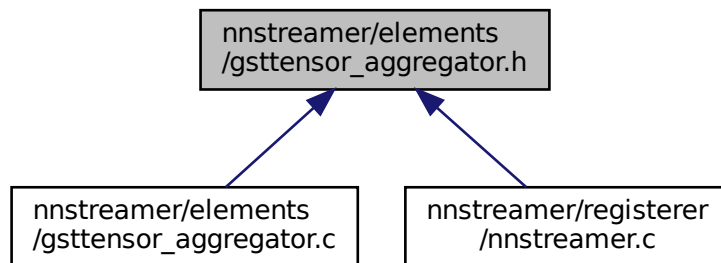
```
#include <gst/gst.h>
```

```
#include <tensor_common.h>
```

Include dependency graph for `gsttensor_aggregator.h`:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorAggregator](#)
GstTensorAggregator data structure.
- struct [_GstTensorAggregatorClass](#)
GstTensorAggregatorClass data structure.

Macros

- #define [GST_TYPE_TENSOR_AGGREGATOR](#) ([gst_tensor_aggregator_get_type\(\)](#))
- #define [GST_TENSOR_AGGREGATOR](#)(obj) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj),[GST_TYPE_TENSOR_AGGREGATOR](#)))
- #define [GST_TENSOR_AGGREGATOR_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_CAST](#)((klass),[GST_TYPE_TENSOR_AGGREGATOR](#)))
- #define [GST_IS_TENSOR_AGGREGATOR](#)(obj) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_AGGREGATOR](#)))
- #define [GST_IS_TENSOR_AGGREGATOR_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_AGGREGATOR](#)))

Typedefs

- typedef struct [_GstTensorAggregator](#) [GstTensorAggregator](#)
- typedef struct [_GstTensorAggregatorClass](#) [GstTensorAggregatorClass](#)

Functions

- GType [gst_tensor_aggregator_get_type](#) (void)
Function to get type of tensor_aggregator.

9.25.1 Detailed Description

GStreamer plugin to aggregate tensor stream.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

29 August 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung jy1210.jung@samsung.com

Bug No known bugs except for NYI items

9.25.2 Macro Definition Documentation

9.25.2.1 GST_IS_TENSOR_AGGREGATOR

```
#define GST_IS_TENSOR_AGGREGATOR(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_AGGREGATOR))
```

Definition at line 41 of file `gsttensor_aggregator.h`.

9.25.2.2 GST_IS_TENSOR_AGGREGATOR_CLASS

```
#define GST_IS_TENSOR_AGGREGATOR_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_AGGREGATOR))
```

Definition at line 43 of file gsttensor_aggregator.h.

9.25.2.3 GST_TENSOR_AGGREGATOR

```
#define GST_TENSOR_AGGREGATOR(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_AGGREGATOR, GstTensorAggregator))
```

Definition at line 37 of file gsttensor_aggregator.h.

9.25.2.4 GST_TENSOR_AGGREGATOR_CLASS

```
#define GST_TENSOR_AGGREGATOR_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_AGGREGATOR, GstTensorAggregatorClass))
```

Definition at line 39 of file gsttensor_aggregator.h.

9.25.2.5 GST_TYPE_TENSOR_AGGREGATOR

```
#define GST_TYPE_TENSOR_AGGREGATOR (gst_tensor_aggregator_get_type())
```

Definition at line 35 of file gsttensor_aggregator.h.

9.25.3 Typedef Documentation

9.25.3.1 GstTensorAggregator

```
typedef struct _GstTensorAggregator GstTensorAggregator
```

Definition at line 46 of file gsttensor_aggregator.h.

9.25.3.2 GstTensorAggregatorClass

```
typedef struct _GstTensorAggregatorClass GstTensorAggregatorClass
```

Definition at line 47 of file gsttensor_aggregator.h.

9.25.4 Function Documentation

9.25.4.1 gst_tensor_aggregator_get_type()

```
GType gst_tensor_aggregator_get_type (
    void )
```

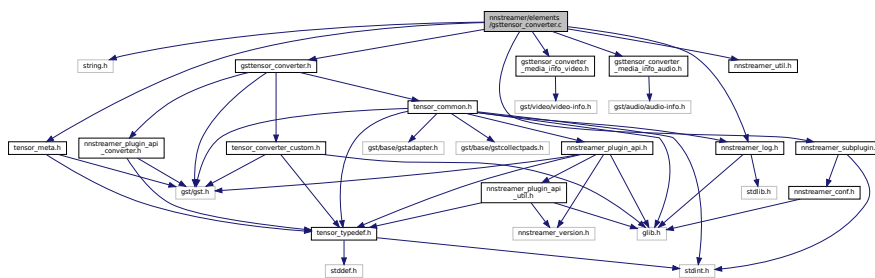
Function to get type of tensor_aggregator.

9.26 nntstreamer/elements/gsttensor_converter.c File Reference

GStreamer plugin to convert media types to tensors (as a filter for other general neural network filters)

```
#include <string.h>
#include "gsttensor_converter.h"
#include "tensor_meta.h"
#include "gsttensor_converter_media_info_video.h"
#include "gsttensor_converter_media_info_audio.h"
#include <nnstreamer_log.h>
#include <nnstreamer_subplugin.h>
#include <nnstreamer_util.h>
```

Include dependency graph for gsttensor_converter.c:



Macros

- #define `TEXT_CAPS_STR` "text/x-raw, format = (string) utf8"
Caps string for text input.
- #define `append_text_caps_template`(caps) `gst_caps_append` (caps, `gst_caps_from_string` (`TEXT_CAPS_STR`))
- #define `OCTET_CAPS_STR` "application/octet-stream"
Caps string for binary stream.
- #define `append_octet_caps_template`(caps) `gst_caps_append` (caps, `gst_caps_from_string` (`OCTET_CAPS_STR`))
- #define `append_flex_tensor_caps_template`(caps) `gst_caps_append` (caps, `gst_caps_from_string` (`GST_TENSORS_FLEX_CAP_DEFAULT`))
Macro to append template caps for flexible tensor.
- #define `DBG` (!self->silent)
Macro for debug mode.
- #define `silent_debug_timestamp`(self, buf)
- #define `GST_CAT_DEFAULT` `gst_tensor_converter_debug`
- #define `STRING_CUSTOM_MODE`(self)
- #define `DEFAULT_SET_TIMESTAMP` TRUE
Flag to set timestamp when received a buffer with invalid timestamp.
- #define `DEFAULT_SILENT` TRUE
Flag to print minimized log.
- #define `DEFAULT_FRAMES_PER_TENSOR` 1
Frames in output tensor.
- #define `gst_tensor_converter_parent_class` `parent_class`

Enumerations

- enum {
`PROP_0`, `PROP_INPUT_DIMENSION`, `PROP_INPUT_TYPE`, `PROP_FRAMES_PER_TENSOR`,
`PROP_SET_TIMESTAMP`, `PROP_SUBPLUGINS`, `PROP_SILENT`, `PROP_MODE` }
tensor_converter properties

Functions

- `GST_DEBUG_CATEGORY_STATIC` (`gst_tensor_converter_debug`)
- `G_DEFINE_TYPE` (`GstTensorConverter`, `gst_tensor_converter`, `GST_TYPE_ELEMENT`)
- static void `gst_tensor_converter_finalize` (`GObject` *object)
Function to finalize instance.
- static void `gst_tensor_converter_set_property` (`GObject` *object, `guint` prop_id, `const GValue` *value, `GParamSpec` *pspec)
Setter for tensor_converter properties.
- static void `gst_tensor_converter_get_property` (`GObject` *object, `guint` prop_id, `GValue` *value, `GParamSpec` *pspec)
Getter for tensor_converter properties.
- static gboolean `gst_tensor_converter_sink_event` (`GstPad` *pad, `GstObject` *parent, `GstEvent` *event)
This function handles sink event.
- static gboolean `gst_tensor_converter_sink_query` (`GstPad` *pad, `GstObject` *parent, `GstQuery` *query)
This function handles sink pad query.
- static gboolean `gst_tensor_converter_src_query` (`GstPad` *pad, `GstObject` *parent, `GstQuery` *query)
This function handles src pad query.
- static `GstFlowReturn` `gst_tensor_converter_chain` (`GstPad` *pad, `GstObject` *parent, `GstBuffer` *buf)
Chain function, this function does the actual processing.

- static GstStateChangeReturn [gst_tensor_converter_change_state](#) (GstElement *element, GstStateChange transition)
Called to perform state change.
- static void [gst_tensor_converter_reset](#) (GstTensorConverter *self)
Clear and reset data.
- static GstCaps * [gst_tensor_converter_query_caps](#) (GstTensorConverter *self, GstPad *pad, GstCaps *filter)
Get pad caps for caps negotiation.
- static gboolean [gst_tensor_converter_parse_caps](#) (GstTensorConverter *self, const GstCaps *caps)
Parse caps and set tensors info.
- static void [gst_tensor_converter_update_caps](#) (GstTensorConverter *self)
Update src pad caps from tensors config.
- static const NNStreamerExternalConverter * [findExternalConverter](#) (const char *media_type)
Internal static function to find registered subplugins.
- static void [gst_tensor_converter_class_init](#) (GstTensorConverterClass *klass)
Initialize the tensor_converter's class.
- static void [gst_tensor_converter_init](#) (GstTensorConverter *self)
Initialize tensor_converter element.
- static GstAdapter * [gst_tensor_converter_get_adapter](#) (GstTensorConverter *self, GstBuffer *buf)
Internal function to get adapter.
- static void [_gst_tensor_converter_chain_segment](#) (GstTensorConverter *self, gsize frame_size)
Chain function's private routine.
- static void [_gst_tensor_converter_chain_timestamp](#) (GstTensorConverter *self, GstBuffer *inbuf, guint frames_in)
Chain function's private routine.
- static GstBuffer * [_gst_tensor_converter_chain_octet](#) (GstTensorConverter *self, GstBuffer *buf)
Chain function's private routine to process octet stream.
- static GstBuffer * [_gst_tensor_converter_chain_flex_tensor](#) (GstTensorConverter *self, GstBuffer *buf)
Chain function's private routine to process flex tensor.
- static GstFlowReturn [_gst_tensor_converter_chain_push](#) (GstTensorConverter *self, GstBuffer *buf)
Chain function's private routine to push buffer into src pad.
- static GstFlowReturn [_gst_tensor_converter_chain_chunk](#) (GstTensorConverter *self, GstBuffer *inbuf, guint frames_in, guint frames_out, gsize frame_size)
Chain function's private routine to push multiple buffers.
- static void [gst_tensor_converter_get_format_list](#) (GValue *list,...)
Get supported format list.
- static gboolean [gst_tensor_converter_video_stride](#) (GstVideoFormat format, gint width)
Determine if we need zero-padding.
- static gboolean [gst_tensor_converter_parse_video](#) (GstTensorConverter *self, const GstCaps *caps, GstTensorsConfig *config)
Set the tensors config structure from video info (internal static function)
- static gboolean [gst_tensor_converter_parse_audio](#) (GstTensorConverter *self, const GstCaps *caps, GstTensorsConfig *config)
Set the tensors config structure from audio info (internal static function)
- static gboolean [gst_tensor_converter_parse_text](#) (GstTensorConverter *self, GstTensorsConfig *config, const GstStructure *structure)
Set the tensors config structure from text info (internal static function)
- static gboolean [gst_tensor_converter_parse_octet](#) (GstTensorConverter *self, GstTensorsConfig *config, const GstStructure *structure)
Set the tensors configs structure from octet stream (internal static function)
- static gboolean [gst_tensor_converter_parse_tensor](#) (GstTensorConverter *self, GstTensorsConfig *config, const GstStructure *structure)

- Set the tensors configs structure from flex tensor stream (internal static function)*

 - static gboolean `gst_tensor_converter_parse_custom` (`GstTensorConverter *self`, `GstTensorsConfig *config`, `const GstCaps *caps`)
- Set the tensors config structure from caps (internal static function for custom mode)*

 - static `GstCaps *` `gst_tensor_converter_get_possible_media_caps` (`GstTensorConverter *self`)
- Get possible media-caps from downstream element.*

 - const `NNStreamerExternalConverter *` `nnstreamer_converter_find` (`const char *name`)
- Find converter sub-plugin with the name.*

 - static gboolean `nnstreamer_converter_validate` (`const NNStreamerExternalConverter *converter`)
- Validate converter sub-plugin's data.*

 - int `registerExternalConverter` (`NNStreamerExternalConverter *ex`)
- Converter's external subplugins should call this at init.*

 - void `unregisterExternalConverter` (`const char *name`)
- Converter's external subplugins should call this at exit.*

 - void `nnstreamer_converter_set_custom_property_desc` (`const char *name`, `const char *prop`,...)
- set custom property description for tensor converter sub-plugin*

 - int `nnstreamer_converter_custom_register` (`const gchar *name`, `tensor_converter_custom` func, `void *data`)
- Registers a callback for tensor_converter custom condition.*

 - int `nnstreamer_converter_custom_unregister` (`const gchar *name`)
- Unregisters a callback for tensor_converter custom condition.*

9.26.1 Detailed Description

GStreamer plugin to convert media types to tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

26 Mar 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

Todo For flatbuffers, support other/tensors with properties

Subplugins are not tested, yet.

9.26.2 Macro Definition Documentation

9.26.2.1 `append_flex_tensor_caps_template`

```
#define append_flex_tensor_caps_template(  
    caps ) gst_caps_append (caps, gst_caps_from_string (GST_TENSORS_FLEX_CAP_DEFAULT))
```

Macro to append template caps for flexible tensor.

Definition at line 85 of file `gsttensor_converter.c`.

9.26.2.2 `append_octet_caps_template`

```
#define append_octet_caps_template(  
    caps ) gst_caps_append (caps, gst_caps_from_string (OCTET_CAPS_STR))
```

Definition at line 79 of file `gsttensor_converter.c`.

9.26.2.3 `append_text_caps_template`

```
#define append_text_caps_template(  
    caps ) gst_caps_append (caps, gst_caps_from_string (TEXT_CAPS_STR))
```

Definition at line 71 of file `gsttensor_converter.c`.

9.26.2.4 `DBG`

```
#define DBG (!self->silent)
```

Macro for debug mode.

Definition at line 92 of file `gsttensor_converter.c`.

9.26.2.5 `DEFAULT_FRAMES_PER_TENSOR`

```
#define DEFAULT_FRAMES_PER_TENSOR 1
```

Frames in output tensor.

Definition at line 142 of file `gsttensor_converter.c`.

9.26.2.6 DEFAULT_SET_TIMESTAMP

```
#define DEFAULT_SET_TIMESTAMP TRUE
```

Flag to set timestamp when received a buffer with invalid timestamp.

Definition at line 132 of file gsttensor_converter.c.

9.26.2.7 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 137 of file gsttensor_converter.c.

9.26.2.8 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_converter_debug
```

Definition at line 104 of file gsttensor_converter.c.

9.26.2.9 gst_tensor_converter_parent_class

```
#define gst_tensor_converter_parent_class parent_class
```

Definition at line 144 of file gsttensor_converter.c.

9.26.2.10 OCTET_CAPS_STR

```
#define OCTET_CAPS_STR "application/octet-stream"
```

Caps string for binary stream.

Definition at line 77 of file gsttensor_converter.c.

9.26.2.11 silent_debug_timestamp

```
#define silent_debug_timestamp(
    self,
    buf )
```

Value:

```
do { \
  if (DBG) { \
    GST_DEBUG_OBJECT (self, "pts = %" GST_TIME_FORMAT, GST_TIME_ARGS (GST_BUFFER_PTS (buf))); \
    GST_DEBUG_OBJECT (self, "dts = %" GST_TIME_FORMAT, GST_TIME_ARGS (GST_BUFFER_DTS (buf))); \
    GST_DEBUG_OBJECT (self, "duration = %" GST_TIME_FORMAT "\n", GST_TIME_ARGS (GST_BUFFER_DURATION (buf))); \
  } \
} while (0)
```

Definition at line 95 of file gsttensor_converter.c.

9.26.2.12 STRING_CUSTOM_MODE

```
#define STRING_CUSTOM_MODE(
    self )
```

Value:

```
((self)->mode == _CONVERTER_MODE_CUSTOM_CODE) ? \
  "custom_code (function)" : \
  ((self)->mode == _CONVERTER_MODE_CUSTOM_SCRIPT) ? \
  "custom_script (py)" : \
  "unknown custom mode (internal error!)"
```

Definition at line 106 of file gsttensor_converter.c.

9.26.2.13 TEXT_CAPS_STR

```
#define TEXT_CAPS_STR "text/x-raw, format = (string) utf8"
```

Caps string for text input.

SECTION:element-tensor_converter

A filter that converts media stream to tensor stream for NN frameworks. The output is always in the format of other/tensor or other/tensors.

```
<refsect2> <title>Example launch line</title> |[ gst-launch-1.0 videotestsrc ! video/x-raw,format=R↵
GB,width=640,height=480 ! tensor_converter ! tensor_sink ]| </refsect2>
```

Definition at line 69 of file gsttensor_converter.c.

9.26.3 Enumeration Type Documentation

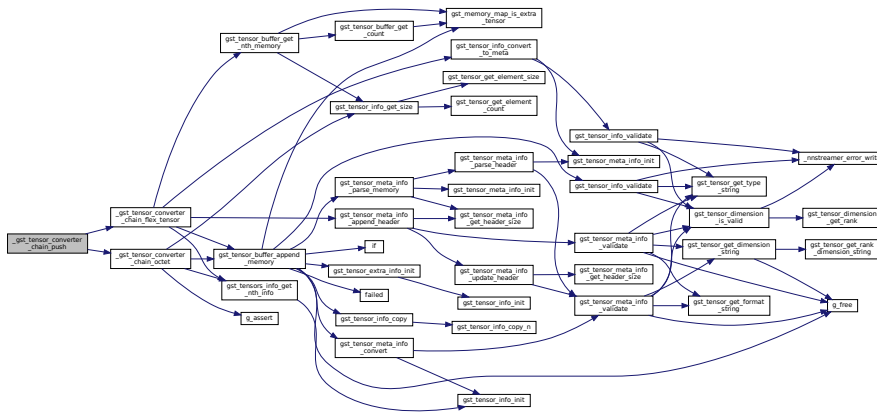
9.26.3.1 anonymous enum

anonymous enum

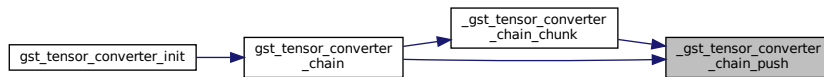
tensor_converter properties

Todo For flatbuffers, support other/tensors.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.5 _gst_tensor_converter_chain_segment()

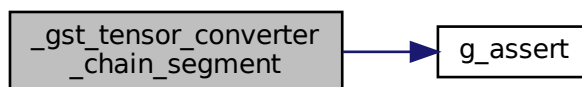
```
static void _gst_tensor_converter_chain_segment (
    GstTensorConverter * self,
    gsize frame_size ) [static]
```

Chain function's private routine.

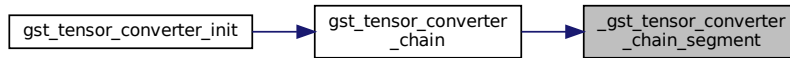
This is an internal logic error.

Definition at line 754 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.6 _gst_tensor_converter_chain_timestamp()

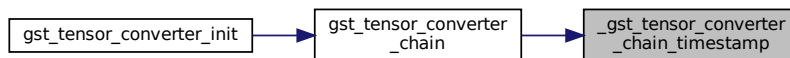
```

static void _gst_tensor_converter_chain_timestamp (
    GstTensorConverter * self,
    GstBuffer * inbuf,
    guint frames_in ) [static]
  
```

Chain function's private routine.

Definition at line 787 of file gsttensor_converter.c.

Here is the caller graph for this function:



9.26.4.7 findExternalConverter()

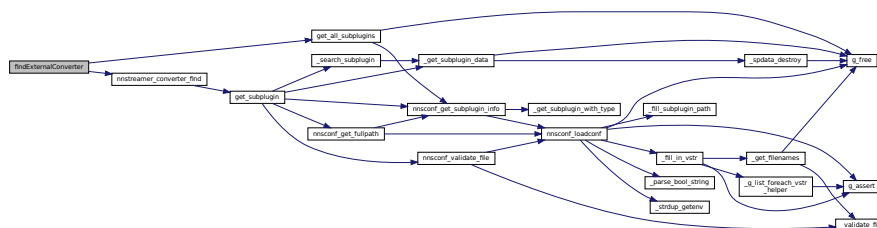
```

static const NNStreamerExternalConverter * findExternalConverter (
    const char * media_type_name ) [static]
  
```

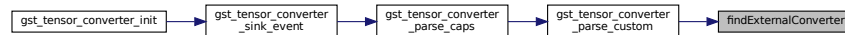
Internal static function to find registered subplugins.

Definition at line 2408 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.8 G_DEFINE_TYPE()

```

G_DEFINE_TYPE (
    GstTensorConverter ,
    gst_tensor_converter ,
    GST_TYPE_ELEMENT )
  
```

9.26.4.9 GST_DEBUG_CATEGORY_STATIC()

```

GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_converter_debug )
  
```

9.26.4.10 gst_tensor_converter_chain()

```

static GstFlowReturn gst_tensor_converter_chain (
    GstPad * pad,
    GstObject * parent,
    GstBuffer * buf ) [static]
  
```

Chain function, this function does the actual processing.

This is an internal logic error.

Supposed 1 frame in buffer (default). Update frame size for each media type.

type * colorspace * width * height

supposed 1 frame in buffer

Refer: <https://gstreamer.freedesktop.org/documentation/design/mediatype-video-raw.html>

Internal logic error!

copy timestamps

copy timestamps

9.26.4.11 `gst_tensor_converter_change_state()`

```
static GstStateChangeReturn gst_tensor_converter_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

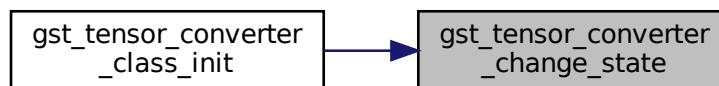
Called to perform state change.

Definition at line 1336 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.12 `gst_tensor_converter_class_init()`

```
static void gst_tensor_converter_class_init (
    GstTensorConverterClass * klass ) [static]
```

Initialize the `tensor_converter`'s class.

`GstTensorConverter::input-dim`:

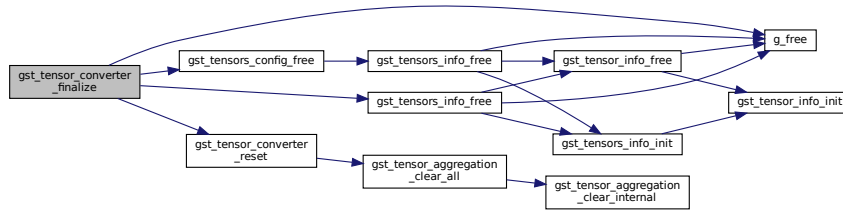
Input tensor dimension from inner array. Generally this property is used to set tensor configuration for byte-stream (application/octet-stream). When setting this property and input media type is video or audio stream, `GstTensorConverter` will compare the media info with this. (If it is different, it will be failed.)

`GstTensorConverter::input-type`:

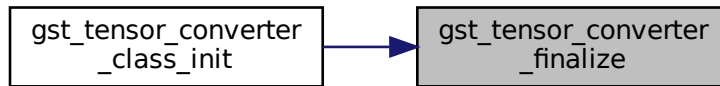
Type of each element of the input tensor. Generally this property is used to set tensor configuration for byte-stream (application/octet-stream). When setting this property and input media type is video or audio stream, `GstTensorConverter` will compare the media info with this. (If it is different, it will be failed.)

`GstTensorConverter::frames-per-tensor`:

Here is the call graph for this function:



Here is the caller graph for this function:



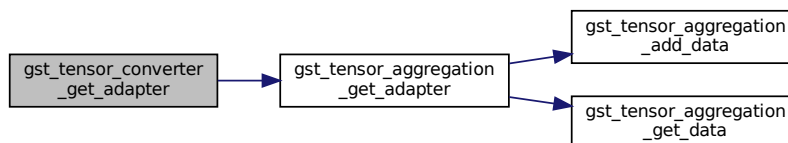
9.26.4.14 gst_tensor_converter_get_adapter()

```
static GstAdapter* gst_tensor_converter_get_adapter (
    GstTensorConverter * self,
    GstBuffer * buf ) [static]
```

Internal function to get adapter.

Definition at line 706 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.15 `gst_tensor_converter_get_format_list()`

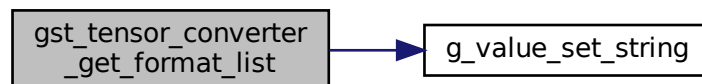
```

static void gst_tensor_converter_get_format_list (
    GValue * list,
    ... ) [static]
  
```

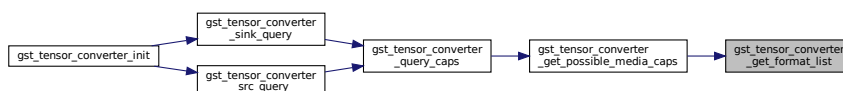
Get supported format list.

Definition at line 1385 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



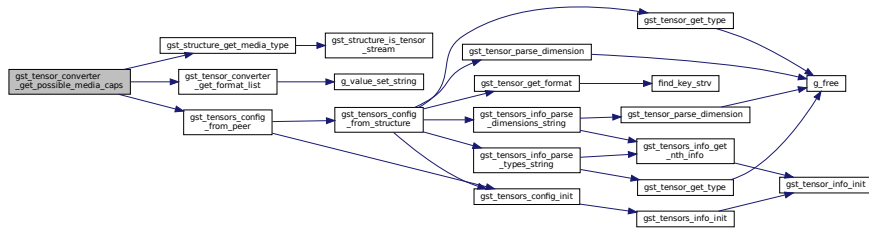
9.26.4.16 `gst_tensor_converter_get_possible_media_caps()`

```
static GstCaps* gst_tensor_converter_get_possible_media_caps (
    GstTensorConverter * self ) [static]
```

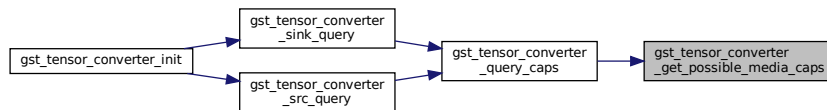
Get possible media-caps from downstream element.

Definition at line 1972 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



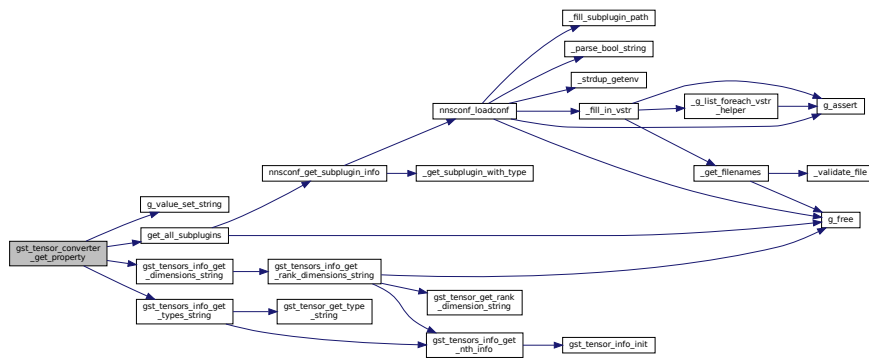
9.26.4.17 `gst_tensor_converter_get_property()`

```
static void gst_tensor_converter_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

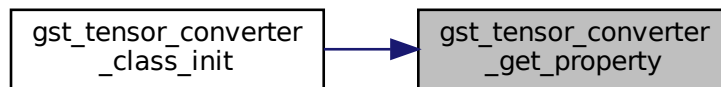
Getter for `tensor_converter` properties.

Definition at line 514 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.18 gst_tensor_converter_init()

```
static void gst_tensor_converter_init (
    GstTensorConverter * self ) [static]
```

Initialize tensor_converter element.

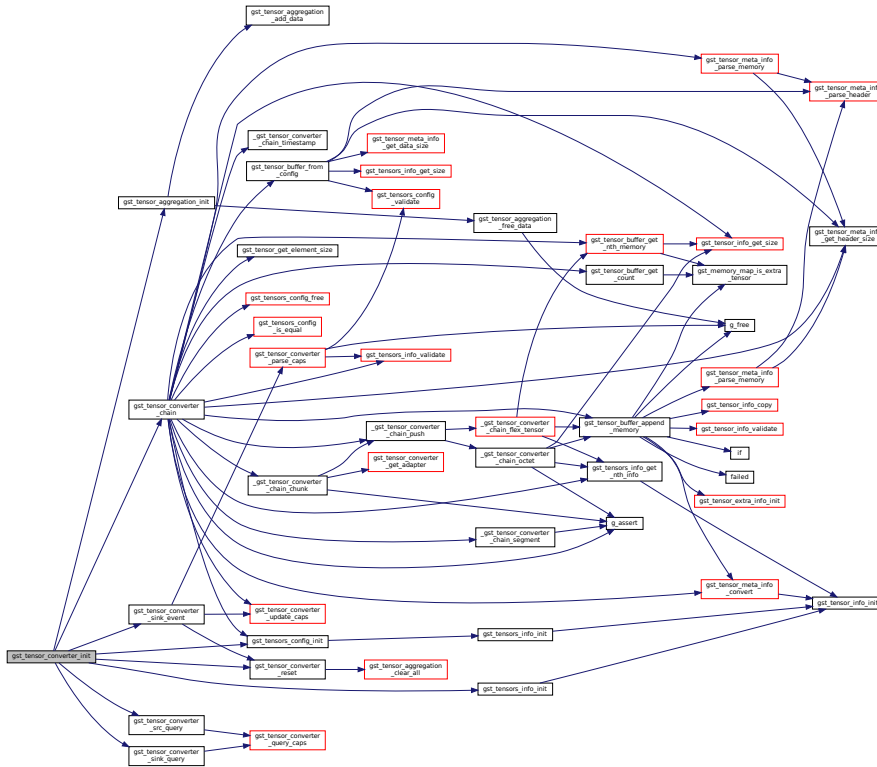
setup sink pad

setup src pad

init properties

Definition at line 331 of file gsttensor_converter.c.

Here is the call graph for this function:



9.26.4.19 gst_tensor_converter_parse_audio()

```
static gboolean gst_tensor_converter_parse_audio (
    GstTensorConverter * self,
    const GstCaps * caps,
    GstTensorsConfig * config ) [static]
```

Set the tensors config structure from audio info (internal static function)

Parameters

<i>self</i>	this pointer to GstTensorConverter
<i>caps</i>	caps for media stream
<i>config</i>	tensors config structure to be filled

Note

Change dimension if tensor contains N frames.

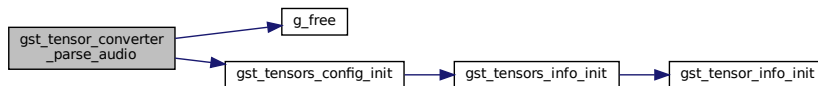
Returns

TRUE if supported type

Refer: https://www.tensorflow.org/api_docs/python/tf/summary/audio A 3-D float32 Tensor of shape [batch_size, frames, channels] or a 2-D float32 Tensor of shape [batch_size, frames].

Definition at line 1582 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.20 `gst_tensor_converter_parse_caps()`

```

static gboolean gst_tensor_converter_parse_caps (
    GstTensorConverter * self,
    const GstCaps * caps ) [static]
  
```

Parse caps and set tensors info.

dimension index of frames in configured tensors

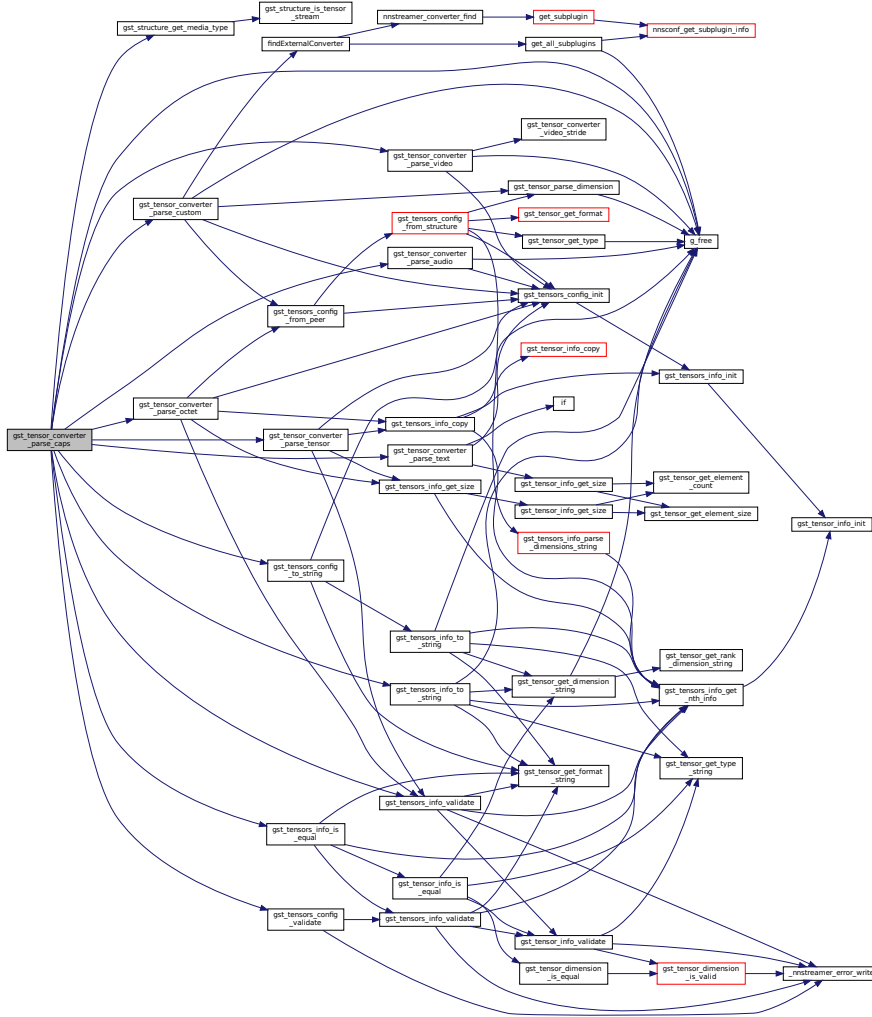
set the number of frames in dimension

not fully configured. the resulting config is weird.

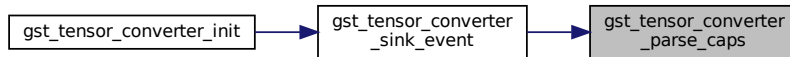
compare tensor info

Definition at line 2190 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.21 gst_tensor_converter_parse_custom()

```
static gboolean gst_tensor_converter_parse_custom (
    GstTensorConverter * self,
    GstTensorsConfig * config,
    const GstCaps * caps ) [static]
```

Set the tensors config structure from caps (internal static function for custom mode)

Parameters

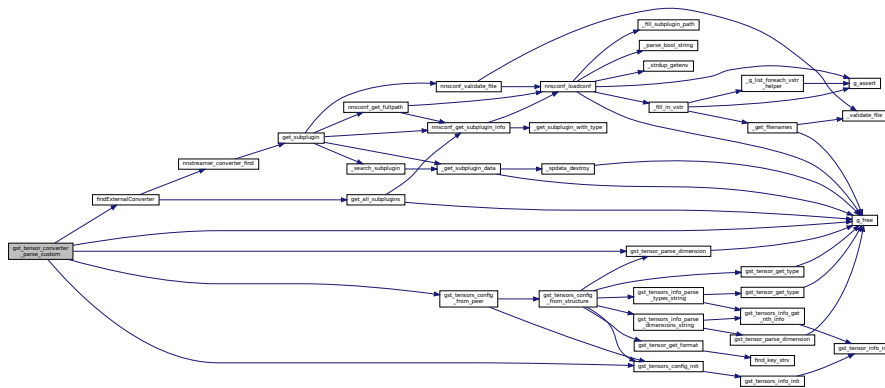
<i>self</i>	this pointer to GstTensorConverter
<i>config</i>	tensors config structure to be filled
<i>caps</i>	incoming caps

Returns

TRUE if supported type

Definition at line 1889 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.22 gst_tensor_converter_parse_octet()

```

static gboolean gst_tensor_converter_parse_octet (
    GstTensorConverter * self,
    GstTensorsConfig * config,
    const GstStructure * structure ) [static]
  
```

Set the tensors configs structure from octet stream (internal static function)

Parameters

<i>self</i>	this pointer to GstTensorConverter
<i>config</i>	tensors config structure to be filled
<i>structure</i>	caps structure

Note

Change tensors dimension and type.

Returns

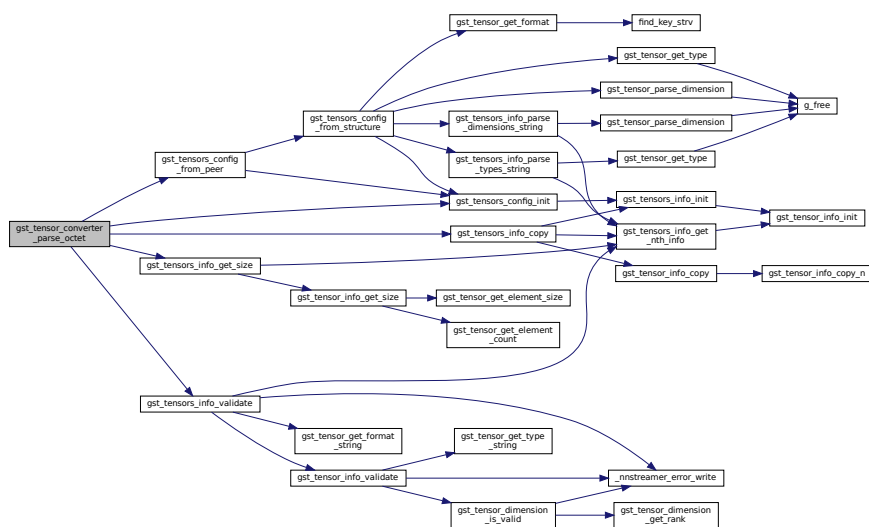
TRUE if supported type

Failure case when octet-stream has multi tensors and multi frames.

Raw byte-stream (application/octet-stream) We cannot get the exact tensors info from caps. All tensors info should be updated. If output is flexible, dimension should be updated in chain function with buffer size. (data format for tensor: [size])

Definition at line 1741 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.23 `gst_tensor_converter_parse_tensor()`

```
static gboolean gst_tensor_converter_parse_tensor (
    GstTensorConverter * self,
    GstTensorsConfig * config,
    const GstStructure * structure ) [static]
```

Set the tensors configs structure from fliex tensor stream (internal static function)

Parameters

<i>self</i>	this pointer to GstTensorConverter
<i>config</i>	tensors config structure to be filled
<i>structure</i>	caps structure

Note

Change tensors dimension and type.

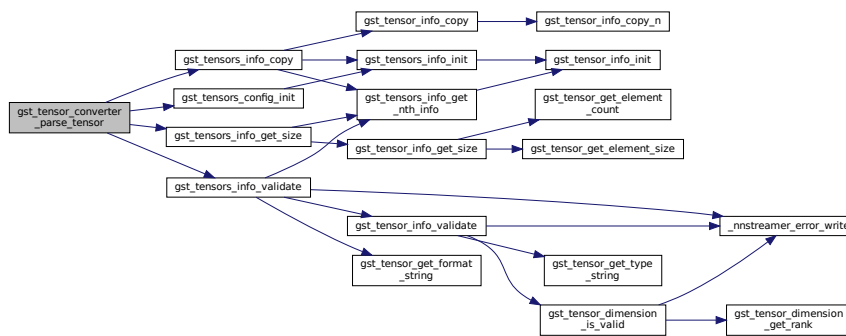
Returns

TRUE if supported type

We cannot get the exact tensors info from caps. All tensors info should be updated in chain function. (data format for tensor: [size])

Definition at line 1834 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.24 gst_tensor_converter_parse_text()

```

static gboolean gst_tensor_converter_parse_text (
    GstTensorConverter * self,
    GstTensorsConfig * config,
    const GstStructure * structure ) [static]
  
```

Set the tensors config structure from text info (internal static function)

Parameters

<i>self</i>	this pointer to GstTensorConverter
<i>config</i>	tensors config structure to be filled
<i>structure</i>	caps structure

Note

Change dimension if tensors contains N frames.

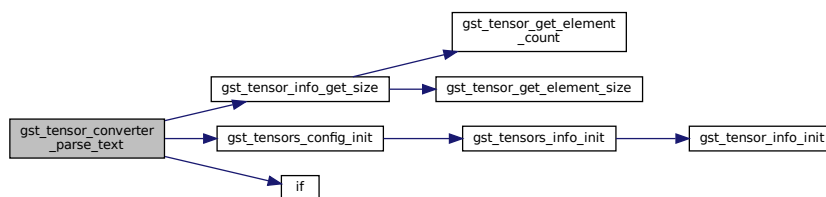
Returns

TRUE if supported type

Refer: https://www.tensorflow.org/api_docs/python/tf/summary/text A string-type Tensor

Definition at line 1670 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.25 gst_tensor_converter_parse_video()

```

static gboolean gst_tensor_converter_parse_video (
    GstTensorConverter * self,
    const GstCaps * caps,
    GstTensorsConfig * config ) [static]
  
```

Set the tensors config structure from video info (internal static function)

Parameters

<i>self</i>	this pointer to GstTensorConverter
<i>caps</i>	caps for media stream
<i>config</i>	tensors config structure to be filled

Note

Change dimension if tensor contains N frames.

Returns

TRUE if supported type

Refer: https://www.tensorflow.org/api_docs/python/tf/summary/image A 4-D uint8 or float32 Tensor of shape [batch_size, height, width, channels] where channels is 1, 3, or 4.

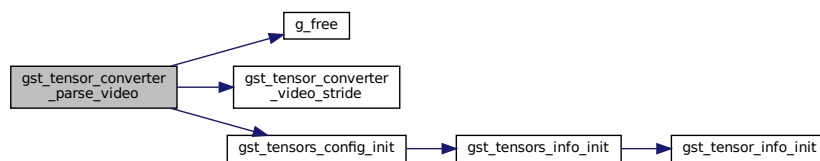
Emit Warning if RSTRIDE = RU4 (3BPP) && Width % 4 > 0

Todo Add more conditions!

Todo need rewrite.

Definition at line 1444 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



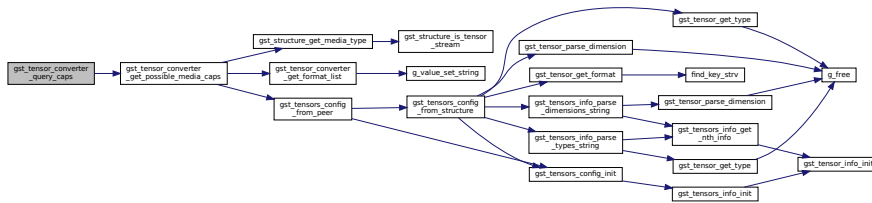
9.26.4.26 `gst_tensor_converter_query_caps()`

```
static GstCaps * gst_tensor_converter_query_caps (
    GstTensorConverter * self,
    GstPad * pad,
    GstCaps * filter ) [static]
```

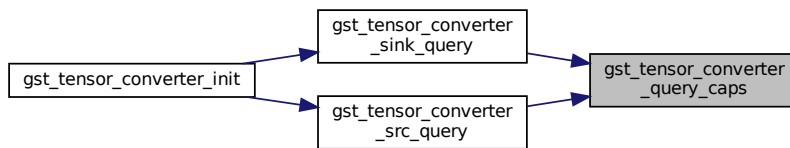
Get pad caps for caps negotiation.

Definition at line 2144 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



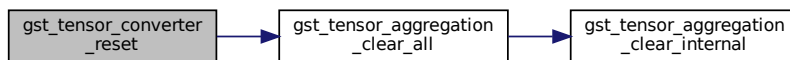
9.26.4.27 `gst_tensor_converter_reset()`

```
static void gst_tensor_converter_reset (
    GstTensorConverter * self ) [static]
```

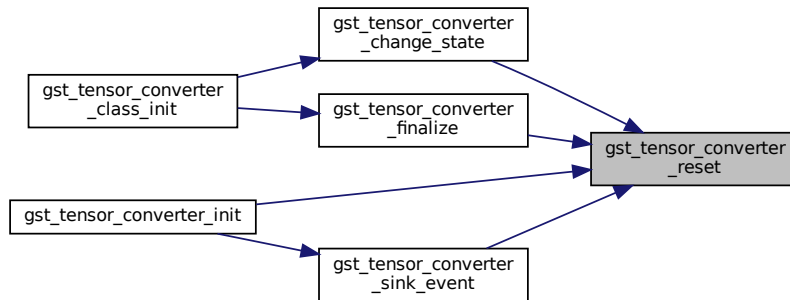
Clear and reset data.

Definition at line 1369 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.28 `gst_tensor_converter_set_property()`

```

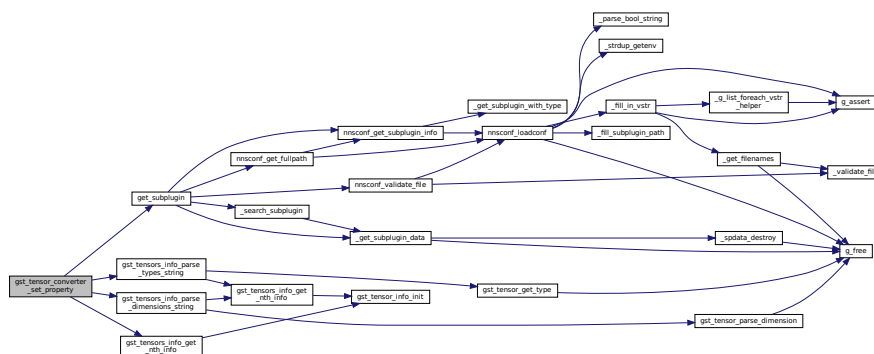
static void gst_tensor_converter_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

Setter for tensor_converter properties.

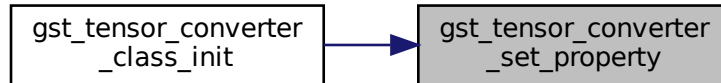
Todo detects framework based on the script extension

Definition at line 406 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.26.4.29 gst_tensor_converter_sink_event()

```
static gboolean gst_tensor_converter_sink_event (  
    GstPad * pad,  
    GObject * parent,  
    GstEvent * event ) [static]
```

This function handles sink event.

Definition at line 586 of file gsttensor_converter.c.

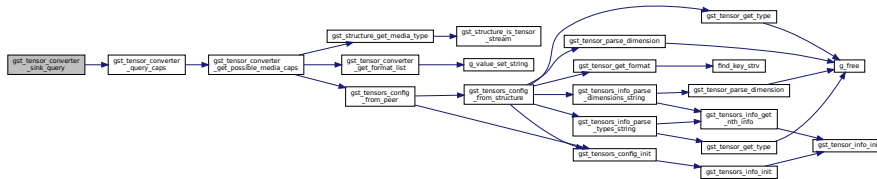
9.26.4.30 `gst_tensor_converter_sink_query()`

```
static gboolean gst_tensor_converter_sink_query (
    GstPad * pad,
    GstObject * parent,
    GstQuery * query ) [static]
```

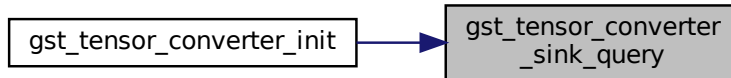
This function handles sink pad query.

Definition at line 654 of file `gsttensor_converter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



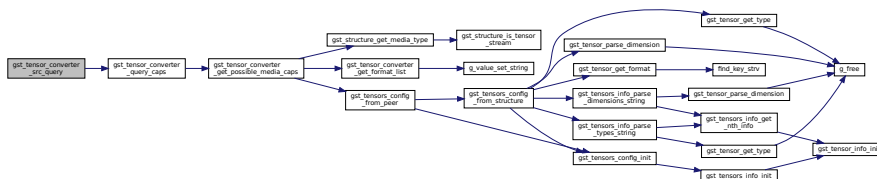
9.26.4.31 `gst_tensor_converter_src_query()`

```
static gboolean gst_tensor_converter_src_query (
    GstPad * pad,
    GstObject * parent,
    GstQuery * query ) [static]
```

This function handles src pad query.

Definition at line 722 of file `gsttensor_converter.c`.

Here is the call graph for this function:



9.26.4.33 `gst_tensor_converter_video_stride()`

```
static gboolean gst_tensor_converter_video_stride (
    GstVideoFormat format,
    gint width ) [static]
```

Determine if we need zero-padding.

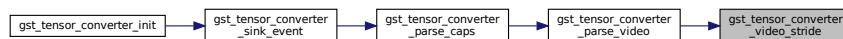
Returns

TRUE if we need to add (or remove) stride per row from the stream data.

Todo The actual list is much longer, fill them. (read <https://gstreamer.freedesktop.org/documentation/design/mediatype-video-raw.html>)

Definition at line 1409 of file `gsttensor_converter.c`.

Here is the caller graph for this function:



9.26.4.34 `nntstreamer_converter_custom_register()`

```
int nntstreamer_converter_custom_register (
    const gchar * name,
    tensor_converter_custom func,
    void * data )
```

Registers a callback for `tensor_converter` custom condition.

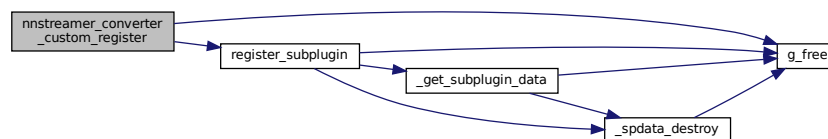
Register the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 2473 of file `gsttensor_converter.c`.

Here is the call graph for this function:



9.26.4.35 nnstreamer_converter_custom_unregister()

```
int nnstreamer_converter_custom_unregister (
    const gchar * name )
```

Unregisters a callback for tensor_converter custom condition.

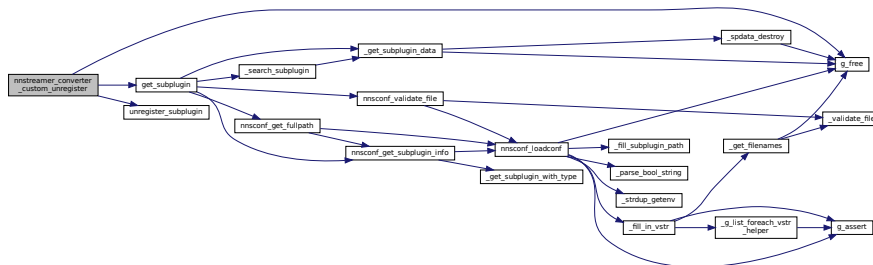
Unregister the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 2500 of file gsttensor_converter.c.

Here is the call graph for this function:



9.26.4.36 nnstreamer_converter_find()

```
const NNStreamerExternalConverter* nnstreamer_converter_find (
    const char * name )
```

Find converter sub-plugin with the name.

Parameters

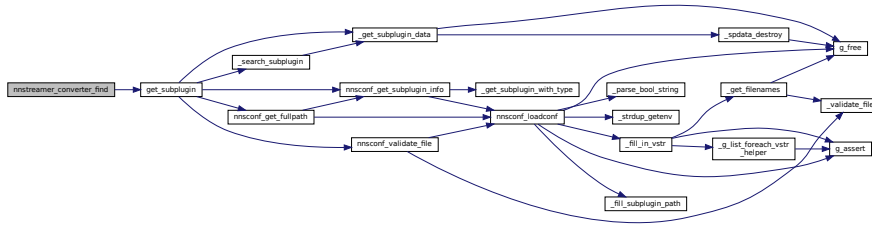
in	<i>name</i>	The name of converter sub-plugin.
----	-------------	-----------------------------------

Returns

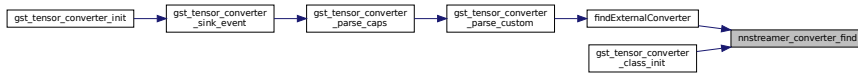
NULL if not found or the sub-plugin object has an error.

Definition at line 2360 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



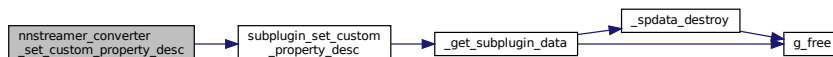
9.26.4.37 nnstreamer_converter_set_custom_property_desc()

```
void nnstreamer_converter_set_custom_property_desc (
    const char * name,
    const char * prop,
    ... )
```

set custom property description for tensor converter sub-plugin

Definition at line 2457 of file `gsttensor_converter.c`.

Here is the call graph for this function:



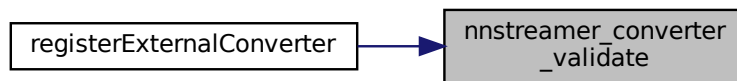
9.26.4.38 nnstreamer_converter_validate()

```
static gboolean nnstreamer_converter_validate (
    const NNStreamerExternalConverter * converter ) [static]
```

Validate converter sub-plugin's data.

Definition at line 2369 of file gsttensor_converter.c.

Here is the caller graph for this function:



9.26.4.39 registerExternalConverter()

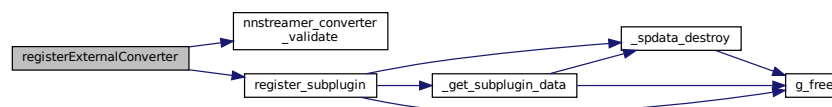
```
int registerExternalConverter (
    NNStreamerExternalConverter * ex )
```

Converter's external subplugins should call this at init.

Converter's sub-plugin should call this function to register itself.

Definition at line 2389 of file gsttensor_converter.c.

Here is the call graph for this function:



9.26.4.40 unregisterExternalConverter()

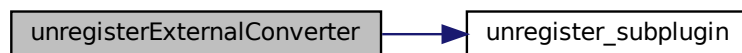
```
void unregisterExternalConverter (
    const char * name )
```

Converter's external subplugins should call this at exit.

Converter's sub-plugin may call this to unregister itself.

Definition at line 2399 of file gsttensor_converter.c.

Here is the call graph for this function:

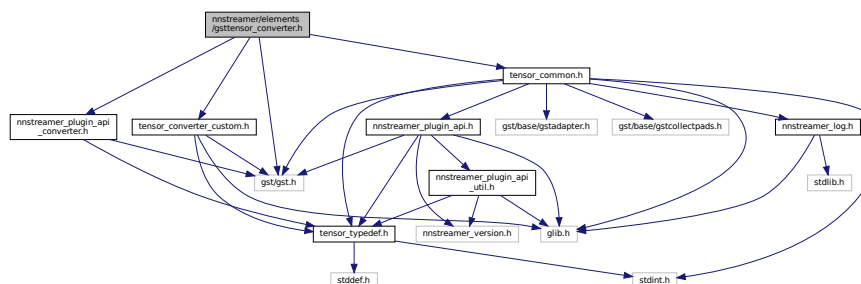


9.27 nnstreamer/elements/gsttensor_converter.h File Reference

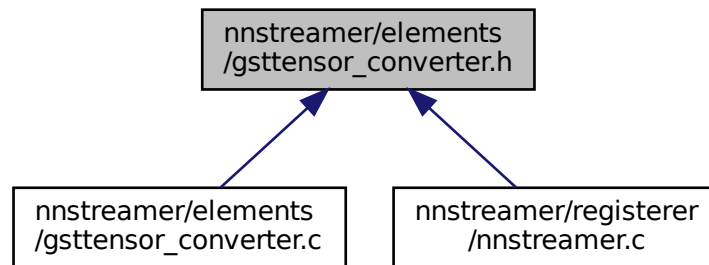
GStreamer plugin to convert media types to tensors (as a filter for other general neural network filters)

```
#include <gst/gst.h>
#include <tensor_common.h>
#include "nnstreamer_plugin_api_converter.h"
#include "tensor_converter_custom.h"
```

Include dependency graph for `gsttensor_converter.h`:



This graph shows which files directly or indirectly include this file:



Classes

- struct [converter_custom_cb_s](#)
- struct [_GstTensorConverter](#)
Internal data structure for tensor_converter instances.
- struct [_GstTensorConverterClass](#)
GstTensorConverterClass data structure.

Macros

- #define [GST_TYPE_TENSOR_CONVERTER](#) ([gst_tensor_converter_get_type](#)())
- #define [GST_TENSOR_CONVERTER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj),[GST_TYPE_TENSOR_CONVERTER](#)))
- #define [GST_TENSOR_CONVERTER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST](#)((klass),[GST_TYPE_TENSOR_CONVERTER](#)))
- #define [GST_IS_TENSOR_CONVERTER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_CONVERTER](#)))
- #define [GST_IS_TENSOR_CONVERTER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_CONVERTER](#)))

Typedefs

- typedef struct [_GstTensorConverter](#) [GstTensorConverter](#)
- typedef struct [_GstTensorConverterClass](#) [GstTensorConverterClass](#)

Enumerations

- enum [tensor_converter_mode](#) { [_CONVERTER_MODE_NONE](#) = 0, [_CONVERTER_MODE_CUSTOM_CODE](#) = 1, [_CONVERTER_MODE_CUSTOM_SCRIPT](#) = 2 }
- tensor converter mode*

Functions

- GType [gst_tensor_converter_get_type](#) (void)
Get Type function required for gst elements.

9.27.1 Detailed Description

GStreamer plugin to convert media types to tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

26 Mar 2018

```
Be careful: this filter assumes that the user has attached
other GST converters as a preprocessor for this filter so that
the incoming buffer is nicely aligned in the array of
uint8[height][width][RGB]. Note that if rstride=RU4, you need
to add the case in "remove_stride_padding_per_row".
```

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.27.2 Macro Definition Documentation

9.27.2.1 GST_IS_TENSOR_CONVERTER

```
#define GST_IS_TENSOR_CONVERTER(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_CONVERTER))
```

Definition at line 50 of file `gsttensor_converter.h`.

9.27.2.2 GST_IS_TENSOR_CONVERTER_CLASS

```
#define GST_IS_TENSOR_CONVERTER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_CONVERTER))
```

Definition at line 52 of file gsttensor_converter.h.

9.27.2.3 GST_TENSOR_CONVERTER

```
#define GST_TENSOR_CONVERTER(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_CONVERTER, GstTensorConverter))
```

Definition at line 46 of file gsttensor_converter.h.

9.27.2.4 GST_TENSOR_CONVERTER_CLASS

```
#define GST_TENSOR_CONVERTER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_CONVERTER, GstTensorConverterClass))
```

Definition at line 48 of file gsttensor_converter.h.

9.27.2.5 GST_TYPE_TENSOR_CONVERTER

```
#define GST_TYPE_TENSOR_CONVERTER (gst_tensor_converter_get_type())
```

Definition at line 44 of file gsttensor_converter.h.

9.27.3 Typedef Documentation

9.27.3.1 GstTensorConverter

```
typedef struct _GstTensorConverter GstTensorConverter
```

Definition at line 55 of file gsttensor_converter.h.

9.27.3.2 GstTensorConverterClass

```
typedef struct _GstTensorConverterClass GstTensorConverterClass
```

Definition at line 56 of file gsttensor_converter.h.

9.27.4 Enumeration Type Documentation

9.27.4.1 tensor_converter_mode

```
enum tensor_converter_mode
```

tensor converter mode

Enumerator

<code>_CONVERTER_MODE_NONE</code>	Normal mode (default)
<code>_CONVERTER_MODE_CUSTOM_CODE</code>	Custom mode (callback type)
<code>_CONVERTER_MODE_CUSTOM_SCRIPT</code>	Custom mode (script type)

Definition at line 66 of file `gsttensor_converter.h`.

9.27.5 Function Documentation

9.27.5.1 `gst_tensor_converter_get_type()`

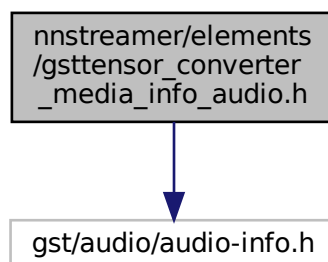
```
GType gst_tensor_converter_get_type (  
    void )
```

Get Type function required for gst elements.

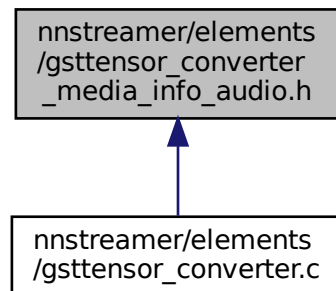
9.28 nntstreamer/elements/gsttensor_converter_media_info_audio.h File Reference

Define collection of media type and functions to parse media info for audio support.

```
#include <gst/audio/audio-info.h>  
Include dependency graph for gsttensor_converter_media_info_audio.h:
```



This graph shows which files directly or indirectly include this file:



Macros

- #define `AUDIO_CAPS_STR`
Caps string for supported audio format.
- #define `append_audio_caps_template`(caps) `gst_caps_append` (caps, `gst_caps_from_string` (`AUDIO_CAPS_STR`))
- #define `is_audio_supported`(...) `TRUE`

9.28.1 Detailed Description

Define collection of media type and functions to parse media info for audio support.

NNStreamer media type definition for tensor-converter Copyright (C) 2019 Jijoong Moon jijoong.moon@samsung.com

Date

26 Mar 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.28.2 Macro Definition Documentation

9.28.2.1 append_audio_caps_template

```
#define append_audio_caps_template(  
    caps ) gst_caps_append (caps, gst_caps_from_string (AUDIO_CAPS_STR))
```

Definition at line 32 of file gstreamer_converter_media_info_audio.h.

9.28.2.2 AUDIO_CAPS_STR

```
#define AUDIO_CAPS_STR
```

Value:

```
GST_AUDIO_CAPS_MAKE (("{ S8, U8, S16LE, S16BE, U16LE, U16BE, S32LE, S32BE, U32LE, U32BE, F32LE, F32BE,  
    F64LE, F64BE }") \  
    ", layout = (string) interleaved"
```

Caps string for supported audio format.

Definition at line 28 of file gstreamer_converter_media_info_audio.h.

9.28.2.3 is_audio_supported

```
#define is_audio_supported(  
    ... ) TRUE
```

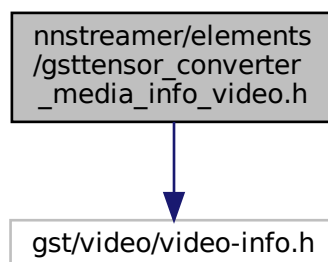
Definition at line 35 of file gstreamer_converter_media_info_audio.h.

9.29 nntstreamer/elements/gsttensor_converter_media_info_video.h File Reference

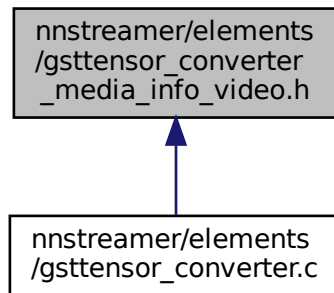
Define collection of media type and functions to parse media info for video support.

```
#include <gst/video/video-info.h>
```

Include dependency graph for gstreamer_converter_media_info_video.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define `NNS_VIDEO_FORMAT` "{ RGB, BGR, RGBx, BGRx, xRGB, xBGR, RGBA, BGRA, ARGB, ABGR, GRAY8, GRAY16_BE, GRAY16_LE }"
Caps string for supported video format.
- #define `VIDEO_CAPS_STR`
- #define `append_video_caps_template`(caps) `gst_caps_append` (caps, `gst_caps_from_string` (`VIDEO_CAPS_STR`))
- #define `is_video_supported`(...) `TRUE`

9.29.1 Detailed Description

Define collection of media type and functions to parse media info for video support.

NNStreamer media type definition for tensor-converter Copyright (C) 2019 Jijoong Moon jijoong.moon@samsung.com

Date

26 Mar 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.29.2 Macro Definition Documentation

9.29.2.1 append_video_caps_template

```
#define append_video_caps_template(  
    caps ) gst_caps_append (caps, gst_caps_from_string (VIDEO_CAPS_STR))
```

Definition at line 38 of file gsttensor_converter_media_info_video.h.

9.29.2.2 is_video_supported

```
#define is_video_supported(  
    ... ) TRUE
```

Definition at line 41 of file gsttensor_converter_media_info_video.h.

9.29.2.3 NNS_VIDEO_FORMAT

```
#define NNS_VIDEO_FORMAT "{ RGB, BGR, RGBx, BGRx, xRGB, xBGR, RGBA, BGRA, ARGB, ABGR, GRAY8,  
GRAY16_BE, GRAY16_LE }"
```

Caps string for supported video format.

Definition at line 31 of file gsttensor_converter_media_info_video.h.

9.29.2.4 VIDEO_CAPS_STR

```
#define VIDEO_CAPS_STR
```

Value:

```
GST_VIDEO_CAPS_MAKE (NNS_VIDEO_FORMAT) \  
    ", interlace-mode = (string) progressive"
```

Definition at line 34 of file gsttensor_converter_media_info_video.h.

9.30 nntstreamer/elements/gsttensor_converter_media_no_audio.h File Reference

Define collection of media type and functions to parse media info for audio if there is no audio support.

Macros

- #define `append_audio_caps_template`(caps)
- #define `is_audio_supported`(...) FALSE
- #define `GstAudioInfo` gsize
- #define `gst_audio_info_init`(i) `memset (i, 0, sizeof (GstAudioInfo))`
- #define `gst_audio_info_from_caps`(...) FALSE
- #define `gst_audio_format_to_string`(...) "Unknown"
- #define `GST_AUDIO_INFO_FORMAT`(...) `GST_AUDIO_FORMAT_UNKNOWN`
- #define `GST_AUDIO_INFO_CHANNELS`(...) 0
- #define `GST_AUDIO_INFO_RATE`(...) 0
- #define `GST_AUDIO_INFO_BPF`(...) 0

Enumerations

- enum `GstAudioFormat` {
 `GST_AUDIO_FORMAT_UNKNOWN`, `GST_AUDIO_FORMAT_S8`, `GST_AUDIO_FORMAT_U8`, `GST_AUDIO_FORMAT_S16`,
 `GST_AUDIO_FORMAT_U16`, `GST_AUDIO_FORMAT_S32`, `GST_AUDIO_FORMAT_U32`, `GST_AUDIO_FORMAT_F32`,
 `GST_AUDIO_FORMAT_F64` }

9.30.1 Detailed Description

Define collection of media type and functions to parse media info for audio if there is no audio support.

NNStreamer media type definition for tensor-converter Copyright (C) 2019 Jijoong Moon [jijoong.↔
moon@samsung.com](mailto:jijoong.moon@samsung.com)

Date

26 Mar 2019

See also

<https://github.com/nstreamer/nstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.30.2 Macro Definition Documentation

9.30.2.1 `append_audio_caps_template`

```
#define append_audio_caps_template(  
    caps )
```

Definition at line 23 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.2 `gst_audio_format_to_string`

```
#define gst_audio_format_to_string(  
    ... ) "Unknown"
```

Definition at line 42 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.3 `GST_AUDIO_INFO_BPF`

```
#define GST_AUDIO_INFO_BPF(  
    ... ) 0
```

Definition at line 47 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.4 `GST_AUDIO_INFO_CHANNELS`

```
#define GST_AUDIO_INFO_CHANNELS(  
    ... ) 0
```

Definition at line 45 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.5 `GST_AUDIO_INFO_FORMAT`

```
#define GST_AUDIO_INFO_FORMAT(  
    ... ) GST\_AUDIO\_FORMAT\_UNKNOWN
```

Definition at line 44 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.6 `gst_audio_info_from_caps`

```
#define gst_audio_info_from_caps(  
    ... ) FALSE
```

Definition at line 41 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.7 `gst_audio_info_init`

```
#define gst_audio_info_init(
    i ) memset (i, 0, sizeof (GstAudioInfo))
```

Definition at line 40 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.8 `GST_AUDIO_INFO_RATE`

```
#define GST_AUDIO_INFO_RATE(
    ... ) 0
```

Definition at line 46 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.9 `GstAudioInfo`

```
#define GstAudioInfo gsize
```

Definition at line 26 of file `gsttensor_converter_media_no_audio.h`.

9.30.2.10 `is_audio_supported`

```
#define is_audio_supported(
    ... ) FALSE
```

Definition at line 24 of file `gsttensor_converter_media_no_audio.h`.

9.30.3 Enumeration Type Documentation

9.30.3.1 `GstAudioFormat`

```
enum GstAudioFormat
```

Enumerator

<code>GST_AUDIO_FORMAT_UNKNOWN</code>	
<code>GST_AUDIO_FORMAT_S8</code>	
<code>GST_AUDIO_FORMAT_U8</code>	
<code>GST_AUDIO_FORMAT_S16</code>	
<code>GST_AUDIO_FORMAT_U16</code>	
<code>GST_AUDIO_FORMAT_S32</code>	
<code>GST_AUDIO_FORMAT_U32</code>	
<code>GST_AUDIO_FORMAT_F32</code>	
<code>GST_AUDIO_FORMAT_F64</code>	

Definition at line 28 of file gsttensor_converter_media_no_audio.h.

9.31 nnstreamer/elements/gsttensor_converter_media_no_video.h File Reference

Define collection of media type and functions to parse media info for video if there is no video support.

Macros

- #define [append_video_caps_template](#)(caps)
- #define [is_video_supported](#)(...) FALSE
- #define [GstVideoInfo](#) gsize
- #define [gst_video_info_init](#)(i) memset (i, 0, sizeof (GstVideoInfo))
- #define [gst_video_info_from_caps](#)(...) FALSE
- #define [gst_video_format_to_string](#)(...) "Unknown"
- #define [GST_VIDEO_INFO_FORMAT](#)(...) GST_VIDEO_FORMAT_UNKNOWN
- #define [GST_VIDEO_INFO_WIDTH](#)(...) 0
- #define [GST_VIDEO_INFO_HEIGHT](#)(...) 0
- #define [GST_VIDEO_INFO_SIZE](#)(...) 0
- #define [GST_VIDEO_INFO_FPS_N](#)(...) 0
- #define [GST_VIDEO_INFO_FPS_D](#)(...) 1

Enumerations

- enum [GstVideoFormat](#) {
[GST_VIDEO_FORMAT_UNKNOWN](#), [GST_VIDEO_FORMAT_GRAY8](#), [GST_VIDEO_FORMAT_RGB](#),
[GST_VIDEO_FORMAT_BGR](#),
[GST_VIDEO_FORMAT_RGBx](#), [GST_VIDEO_FORMAT_BGRx](#), [GST_VIDEO_FORMAT_xRGB](#), [GST_VIDEO_FORMAT_xBG](#),
[GST_VIDEO_FORMAT_RGBA](#), [GST_VIDEO_FORMAT_BGRA](#), [GST_VIDEO_FORMAT_ARGB](#), [GST_VIDEO_FORMAT_AB](#),
[GST_VIDEO_FORMAT_I420](#), [GST_VIDEO_FORMAT_GRAY16_BE](#), [GST_VIDEO_FORMAT_GRAY16_LE](#)
 }

9.31.1 Detailed Description

Define collection of media type and functions to parse media info for video if there is no video support.

NNStreamer media type definition for tensor-converter Copyright (C) 2019 Jijoong Moon jijoong.moon@samsung.com

Date

26 Mar 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.31.2 Macro Definition Documentation

9.31.2.1 `append_video_caps_template`

```
#define append_video_caps_template(  
    caps )
```

Definition at line 23 of file `gsttensor_converter_media_no_video.h`.

9.31.2.2 `gst_video_format_to_string`

```
#define gst_video_format_to_string(  
    ... ) "Unknown"
```

Definition at line 48 of file `gsttensor_converter_media_no_video.h`.

9.31.2.3 `GST_VIDEO_INFO_FORMAT`

```
#define GST_VIDEO_INFO_FORMAT(  
    ... ) GST_VIDEO_FORMAT_UNKNOWN
```

Definition at line 50 of file `gsttensor_converter_media_no_video.h`.

9.31.2.4 `GST_VIDEO_INFO_FPS_D`

```
#define GST_VIDEO_INFO_FPS_D(  
    ... ) 1
```

Definition at line 55 of file `gsttensor_converter_media_no_video.h`.

9.31.2.5 `GST_VIDEO_INFO_FPS_N`

```
#define GST_VIDEO_INFO_FPS_N(  
    ... ) 0
```

Definition at line 54 of file `gsttensor_converter_media_no_video.h`.

9.31.2.6 `gst_video_info_from_caps`

```
#define gst_video_info_from_caps(  
    ... ) FALSE
```

Definition at line 47 of file `gsttensor_converter_media_no_video.h`.

9.31.2.7 `GST_VIDEO_INFO_HEIGHT`

```
#define GST_VIDEO_INFO_HEIGHT(  
    ... ) 0
```

Definition at line 52 of file `gsttensor_converter_media_no_video.h`.

9.31.2.8 `gst_video_info_init`

```
#define gst_video_info_init(  
    i ) memset (i, 0, sizeof (GstVideoInfo))
```

Definition at line 46 of file `gsttensor_converter_media_no_video.h`.

9.31.2.9 `GST_VIDEO_INFO_SIZE`

```
#define GST_VIDEO_INFO_SIZE(  
    ... ) 0
```

Definition at line 53 of file `gsttensor_converter_media_no_video.h`.

9.31.2.10 `GST_VIDEO_INFO_WIDTH`

```
#define GST_VIDEO_INFO_WIDTH(  
    ... ) 0
```

Definition at line 51 of file `gsttensor_converter_media_no_video.h`.

9.31.2.11 `GstVideoInfo`

```
#define GstVideoInfo gsize
```

Definition at line 26 of file `gsttensor_converter_media_no_video.h`.

9.31.2.12 is_video_supported

```
#define is_video_supported(  
    ... ) FALSE
```

Definition at line 24 of file gsttensor_converter_media_no_video.h.

9.31.3 Enumeration Type Documentation

9.31.3.1 GstVideoFormat

```
enum GstVideoFormat
```

Enumerator

GST_VIDEO_FORMAT_UNKNOWN	
GST_VIDEO_FORMAT_GRAY8	
GST_VIDEO_FORMAT_RGB	
GST_VIDEO_FORMAT_BGR	
GST_VIDEO_FORMAT_RGBx	
GST_VIDEO_FORMAT_BGRx	
GST_VIDEO_FORMAT_xRGB	
GST_VIDEO_FORMAT_xBGR	
GST_VIDEO_FORMAT_RGBA	
GST_VIDEO_FORMAT_BGRA	
GST_VIDEO_FORMAT_ARGB	
GST_VIDEO_FORMAT_ABGR	
GST_VIDEO_FORMAT_I420	
GST_VIDEO_FORMAT_GRAY16_BE	
GST_VIDEO_FORMAT_GRAY16_LE	

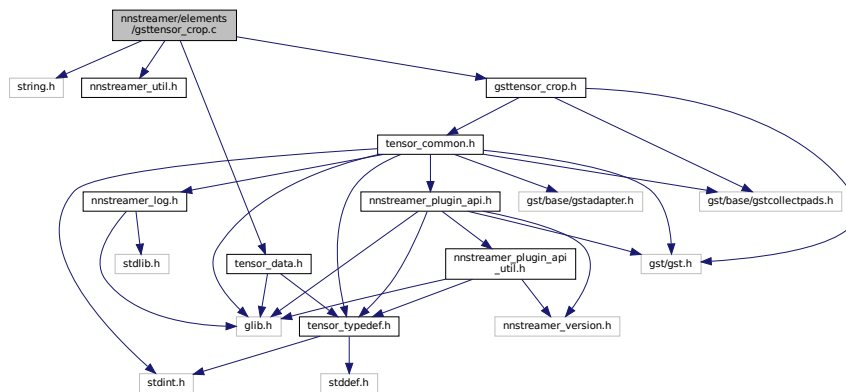
Definition at line 28 of file gsttensor_converter_media_no_video.h.

9.32 nntstreamer/elements/gsttensor_crop.c File Reference

GStreamer element to crop the regions of incoming tensor.

```
#include <string.h>  
#include <nntstreamer_util.h>  
#include "gsttensor_crop.h"  
#include "tensor_data.h"
```


Include dependency graph for `gsttensor_crop.c`:



Classes

- struct [tensor_region_s](#)
Internal data structure to describe tensor region.
- struct [tensor_crop_info_s](#)
Internal data structure to describe cropping tensor data.

Macros

- #define [GST_CAT_DEFAULT](#) `gst_tensor_crop_debug`
- #define [DEFAULT_SILENT](#) `TRUE`
Flag to print minimized log.
- #define [DEFAULT_LATENESS](#) `(-1)`
Default value to compare timestamp of raw and info buffer, in milliseconds (-1 means no synchronization).
- #define [gst_tensor_crop_parent_class](#) `parent_class`

Enumerations

- enum { [PROP_0](#), [PROP_LATENESS](#), [PROP_SILENT](#) }
tensor_crop properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) (`gst_tensor_crop_debug`)
- [G_DEFINE_TYPE](#) (`GstTensorCrop`, `gst_tensor_crop`, `GST_TYPE_ELEMENT`)
- static void [gst_tensor_crop_finalize](#) (`GObject *object`)
Function to finalize instance.
- static void [gst_tensor_crop_set_property](#) (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
Setter for tensor_crop properties.
- static void [gst_tensor_crop_get_property](#) (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)

- Getter for tensor_crop properties.*

 - static GstStateChangeReturn [gst_tensor_crop_change_state](#) (GstElement *element, GstStateChange transition)

Handle state transition.
- static gboolean [gst_tensor_crop_src_event](#) (GstPad *pad, GstObject *parent, GstEvent *event)

Handle event on src pad.
- static gboolean [gst_tensor_crop_sink_event](#) (GstCollectPads *pads, GstCollectData *data, GstEvent *event, gpointer user_data)

Handle event on sink pad.
- static GstFlowReturn [gst_tensor_crop_collected](#) (GstCollectPads *pads, gpointer user_data)

Chain function called when the buffer is available on all of the collect pads.
- static void [gst_tensor_crop_class_init](#) (GstTensorCropClass *klass)

Initialize the tensor_crop's class.
- static void [gst_tensor_crop_pad_reset](#) (GstTensorCropPadData *cpad)

Clear and reset old pad data.
- static void [gst_tensor_crop_reset](#) (GstTensorCrop *self)

Clear and reset old data in tensor_crop.
- static void [gst_tensor_crop_init](#) (GstTensorCrop *self)

Initialize tensor_crop element.
- static GstFlowReturn [gst_tensor_crop_negotiate](#) (GstTensorCrop *self)

Set pad caps if not negotiated.
- static gboolean [gst_tensor_crop_prepare_out_meta](#) (GstTensorCrop *self, gpointer buffer, GstTensorMetaInfo *meta, GstTensorInfo *info, gboolean *is_flexible)

Internal function to prepare output meta info.
- static gboolean [gst_tensor_crop_get_crop_info](#) (GstTensorCrop *self, GstBuffer *info, tensor_crop_info_s *cinfo)

Internal function to parse buffer and fill crop info.
- static GstBuffer * [gst_tensor_crop_do_cropping](#) (GstTensorCrop *self, GstBuffer *raw, tensor_crop_info_s *cinfo)

Internal function to crop incoming buffer.
- static GstFlowReturn [gst_tensor_crop_chain](#) (GstTensorCrop *self, GstCollectData *data_raw, GstCollectData *data_info)

Internal function to transform the input buffer.

Variables

- static GstStaticPadTemplate [raw_template](#)

Template for sink pad (raw data).
- static GstStaticPadTemplate [info_template](#)

Template for sink pad (crop info).
- static GstStaticPadTemplate [src_template](#)

Template for src pad.

9.32.1 Detailed Description

GStreamer element to crop the regions of incoming tensor.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

10 May 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung [jy1210.jung@samsung.com](mailto: jy1210.jung@samsung.com)

Bug No known bugs except for NYI items

9.32.2 Macro Definition Documentation

9.32.2.1 DEFAULT_LATENESS

```
#define DEFAULT_LATENESS (-1)
```

Default value to compare timestamp of raw and info buffer, in milliseconds (-1 means no synchronization).

Definition at line 89 of file gsttensor_crop.c.

9.32.2.2 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 84 of file gsttensor_crop.c.

9.32.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_crop_debug
```

Definition at line 69 of file gsttensor_crop.c.

9.32.2.4 gst_tensor_crop_parent_class

```
#define gst_tensor_crop_parent_class parent_class
```

Definition at line 116 of file gsttensor_crop.c.

9.32.3 Enumeration Type Documentation

9.32.3.1 anonymous enum

anonymous enum

tensor_crop properties

Enumerator

PROP_0	
PROP_LATENESS	
PROP_SILENT	

Definition at line 74 of file gsttensor_crop.c.

9.32.4 Function Documentation

9.32.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorCrop ,
    gst_tensor_crop ,
    GST_TYPE_ELEMENT )
```

9.32.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_crop_debug )
```

9.32.4.3 gst_tensor_crop_chain()

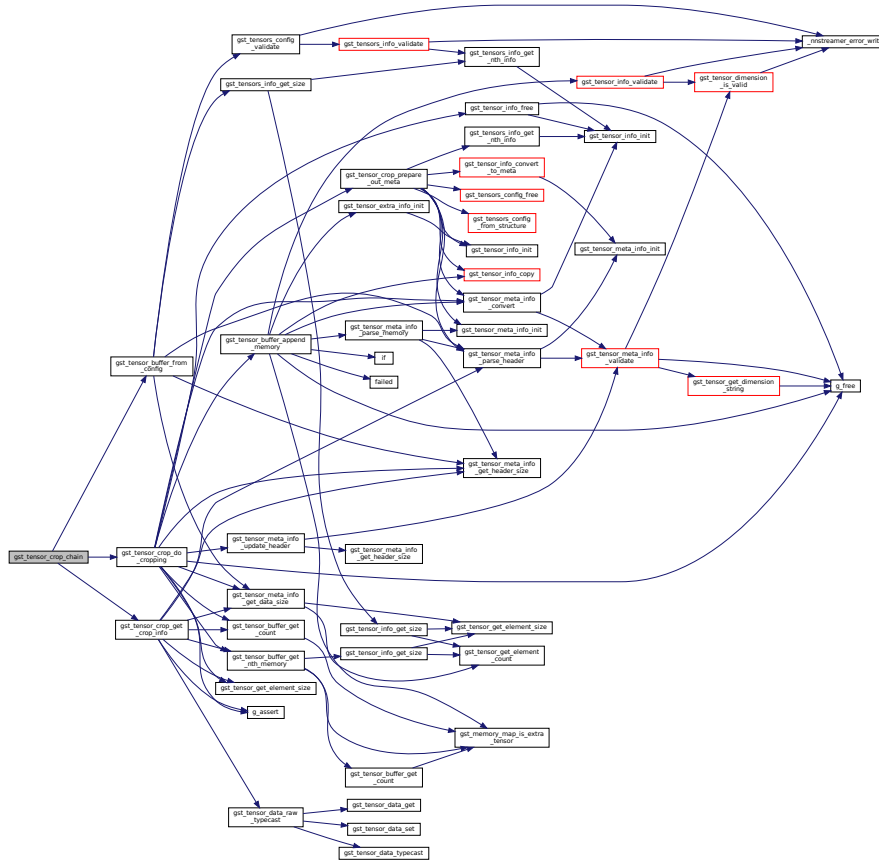
```
static GstFlowReturn gst_tensor_crop_chain (
    GstTensorCrop * self,
    GstCollectData * data_raw,
    GstCollectData * data_info ) [static]
```

Internal function to transform the input buffer.

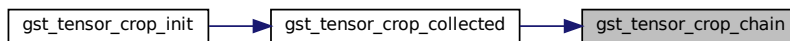
The case when raw and info have different timestamp. Compare timestamp and if time diff is less than lateness, crop raw buffer.

Definition at line 725 of file gsttensor_crop.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.4 gst_tensor_crop_change_state()

```
static GstStateChangeReturn gst_tensor_crop_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

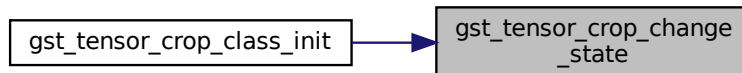
Handle state transition.

Definition at line 333 of file gstreamer/gsttensor_crop.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.5 `gst_tensor_crop_class_init()`

```
static void gst_tensor_crop_class_init (
    GstTensorCropClass * klass ) [static]
```

Initialize the `tensor_crop`'s class.

GstTensorCrop::lateness:

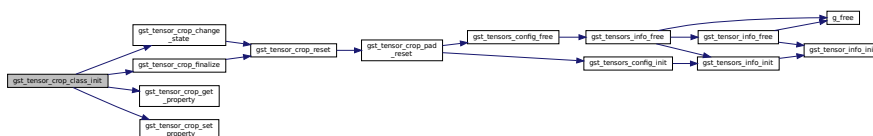
The time difference between raw and info buffer, in milliseconds (-1 means no synchronization). If raw and info buffers on the pads have different timestamp and time-diff is larger than 'lateness', tensor-crop will drop old buffer and wait for next buffers.

GstTensorCrop::silent:

The flag to enable/disable debugging messages.

Definition at line 137 of file `gsttensor_crop.c`.

Here is the call graph for this function:



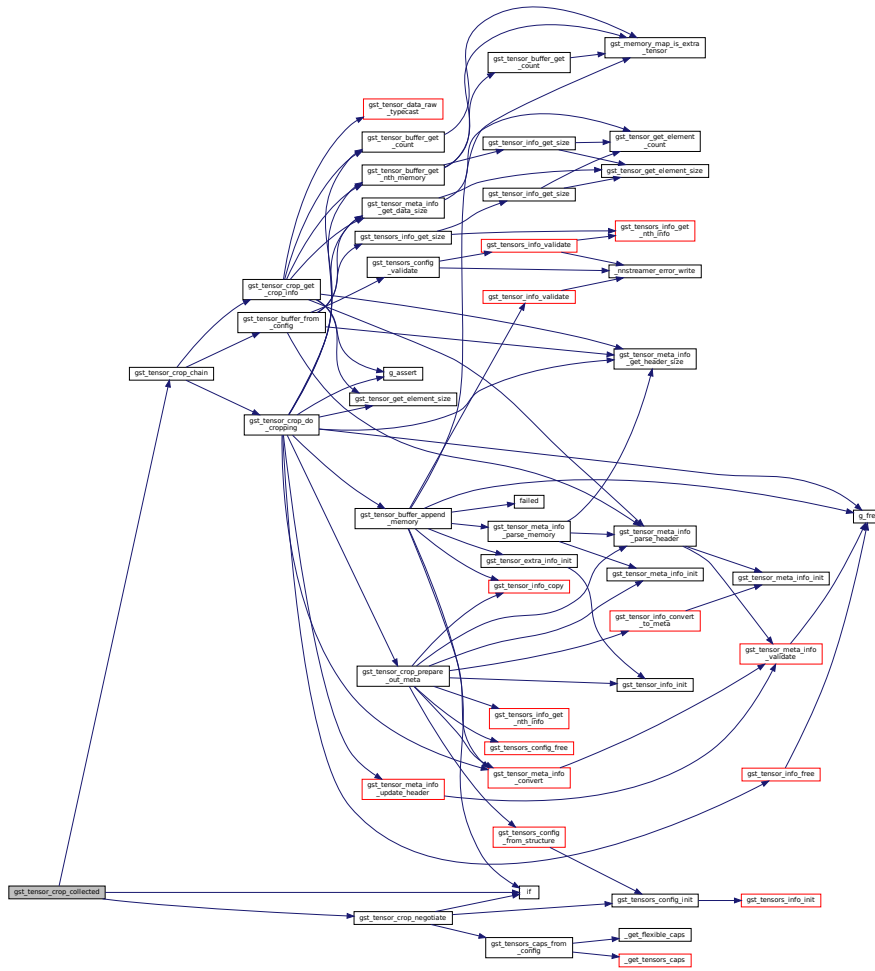
9.32.4.6 gst_tensor_crop_collected()

```
static GstFlowReturn gst_tensor_crop_collected (
    GstCollectPads * pads,
    gpointer user_data ) [static]
```

Chain function called when the buffer is available on all of the collect pads.

Definition at line 812 of file gsttensor_crop.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.8 `gst_tensor_crop_finalize()`

```
static void gst_tensor_crop_finalize (
    GObject * object ) [static]
```

Function to finalize instance.

Definition at line 265 of file `gsttensor_crop.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.9 `gst_tensor_crop_get_crop_info()`

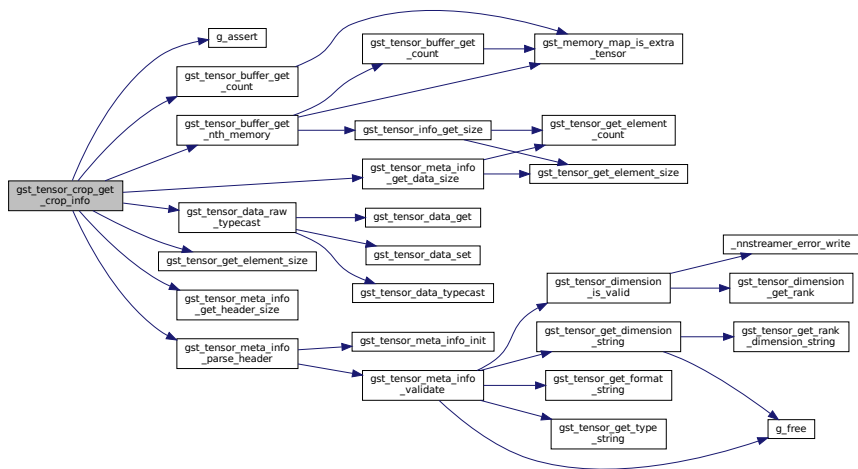
```
static gboolean gst_tensor_crop_get_crop_info (
    GstTensorCrop * self,
    GstBuffer * info,
    tensor_crop_info_s * cinfo ) [static]
```

Internal function to parse buffer and fill crop info.

Todo Add various mode to crop tensor. Now tensor-crop handles NHWC data format only.

Definition at line 547 of file `gsttensor_crop.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.10 gst_tensor_crop_get_property()

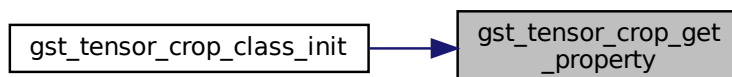
```

static void gst_tensor_crop_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
  
```

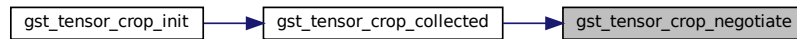
Getter for tensor_crop properties.

Definition at line 309 of file gsttensor_crop.c.

Here is the caller graph for this function:



Here is the caller graph for this function:



9.32.4.13 gst_tensor_crop_pad_reset()

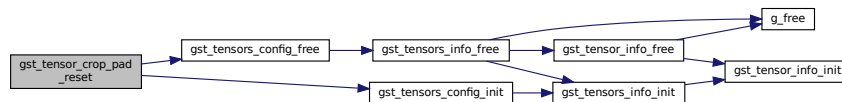
```

static void gst_tensor_crop_pad_reset (
    GstTensorCropPadData * cpad ) [static]
  
```

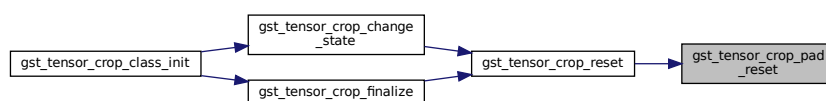
Clear and reset old pad data.

Definition at line 195 of file gsttensor_crop.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.14 gst_tensor_crop_prepare_out_meta()

```

static gboolean gst_tensor_crop_prepare_out_meta (
    GstTensorCrop * self,
    gpointer buffer,
    GstTensorMetaInfo * meta,
    GstTensorInfo * info,
    gboolean * is_flexible ) [static]
  
```

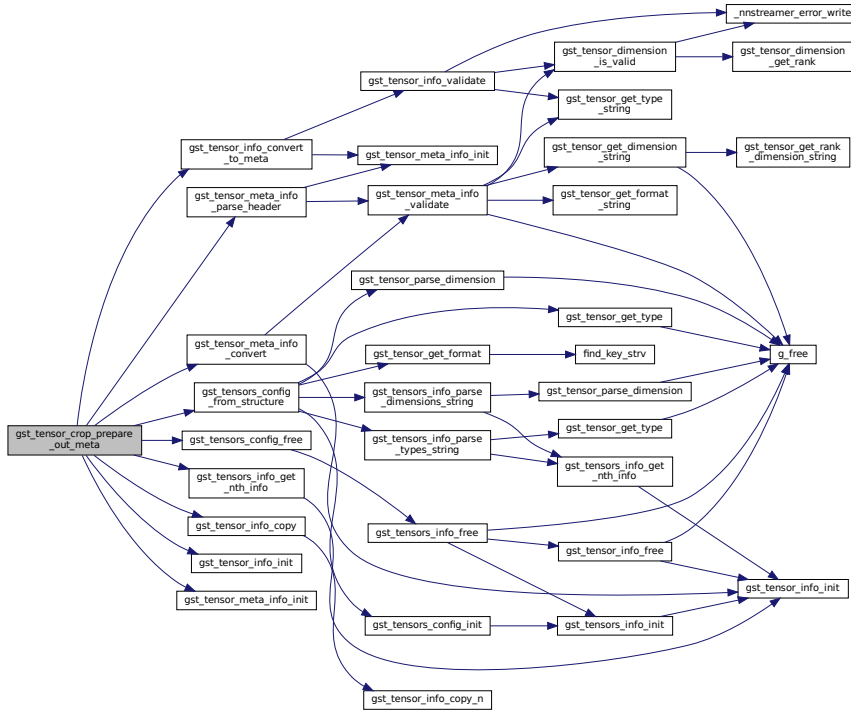
Internal function to prepare output meta info.

Note

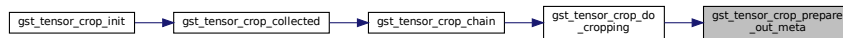
tensor-crop handles single tensor. Parse first one.

Definition at line 497 of file gstreamer/gsttensor_crop.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.15 gst_tensor_crop_reset()

```
static void gst_tensor_crop_reset (
    GstTensorCrop * self ) [static]
```

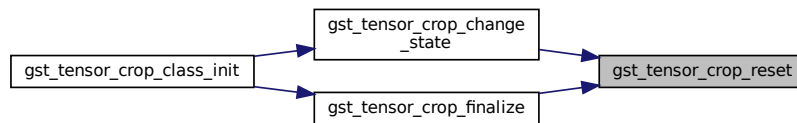
Clear and reset old data in tensor_crop.

Definition at line 205 of file gstreamer/gsttensor_crop.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.32.4.16 `gst_tensor_crop_set_property()`

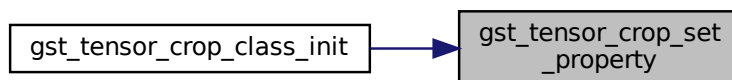
```

static void gst_tensor_crop_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

Setter for `tensor_crop` properties.

Definition at line 285 of file `gsttensor_crop.c`.

Here is the caller graph for this function:



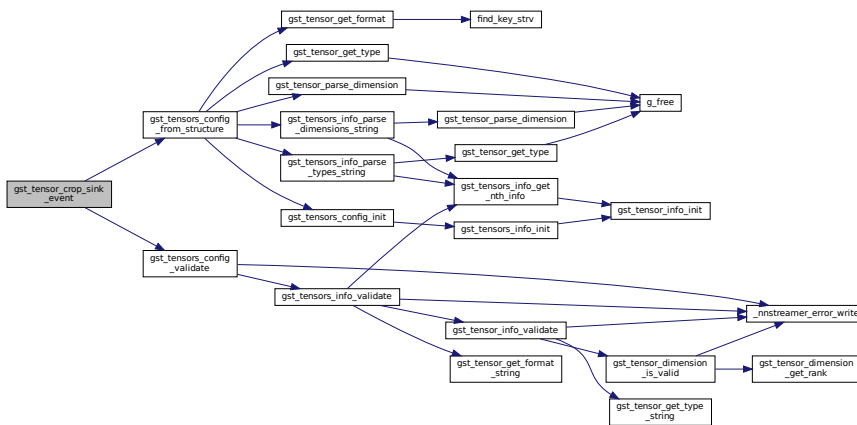
9.32.4.17 `gst_tensor_crop_sink_event()`

```
static gboolean gst_tensor_crop_sink_event (
    GstCollectPads * pads,
    GstCollectData * data,
    GstEvent * event,
    gpointer user_data ) [static]
```

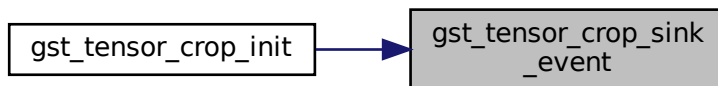
Handle event on sink pad.

Definition at line 388 of file `gsttensor_crop.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



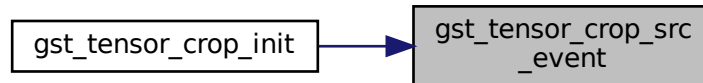
9.32.4.18 `gst_tensor_crop_src_event()`

```
static gboolean gst_tensor_crop_src_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
```

Handle event on src pad.

Definition at line 368 of file gsttensor_crop.c.

Here is the caller graph for this function:



9.32.5 Variable Documentation

9.32.5.1 info_template

```
GstStaticPadTemplate info_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("info",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (GST_TENSORS_FLEX_CAP_DEFAULT))
```

Template for sink pad (crop info).

Definition at line 103 of file gsttensor_crop.c.

9.32.5.2 raw_template

```
GstStaticPadTemplate raw_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("raw",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (GST_TENSOR_CAP_DEFAULT ";"
    GST_TENSORS_CAP_MAKE ("{ static, flexible }")))
```

Template for sink pad (raw data).

Definition at line 94 of file gsttensor_crop.c.

9.32.5.3 src_template

```
GstStaticPadTemplate src_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (GST_TENSORS_FLEX_CAP_DEFAULT))
```

Template for src pad.

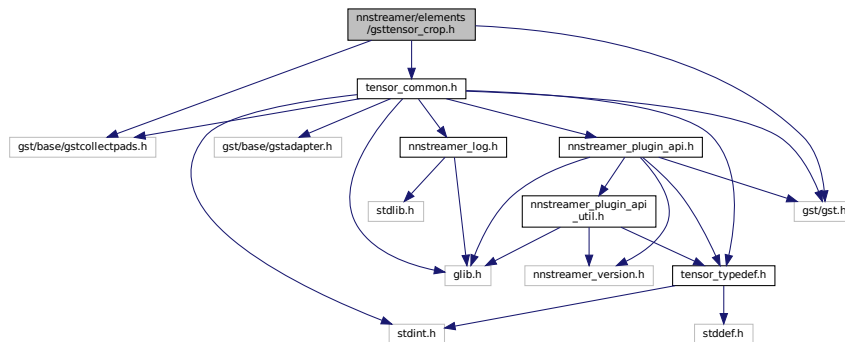
Definition at line 111 of file gsttensor_crop.c.

9.33 nnstreamer/elements/gsttensor_crop.h File Reference

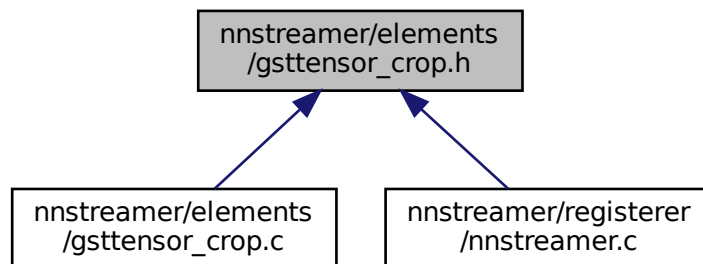
GStreamer element to crop the regions of incoming tensor.

```
#include <gst/gst.h>
#include <gst/base/gstcollectpads.h>
#include <tensor_common.h>
```

Include dependency graph for gsttensor_crop.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [GstTensorCropPadData](#)
GstTensorCrop pad data.
- struct [_GstTensorCrop](#)
GstTensorCrop data structure.
- struct [_GstTensorCropClass](#)
GstTensorCropClass data structure.

Macros

- #define [GST_TYPE_TENSOR_CROP](#) ([gst_tensor_crop_get_type\(\)](#))
- #define [GST_TENSOR_CROP](#)(obj) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj), [GST_TYPE_TENSOR_CROP](#), [GstTensorCrop](#)))
- #define [GST_TENSOR_CROP_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_CAST](#)((klass), [GST_TYPE_TENSOR_CROP](#), [GstTensorCropClass](#)))
- #define [GST_IS_TENSOR_CROP](#)(obj) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj), [GST_TYPE_TENSOR_CROP](#)))
- #define [GST_IS_TENSOR_CROP_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass), [GST_TYPE_TENSOR_CROP](#)))

Typedefs

- typedef struct [_GstTensorCrop](#) [GstTensorCrop](#)
- typedef struct [_GstTensorCropClass](#) [GstTensorCropClass](#)

Functions

- GType [gst_tensor_crop_get_type](#) (void)
Function to get type of tensor_crop.

9.33.1 Detailed Description

GStreamer element to crop the regions of incoming tensor.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

10 May 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung [jy1210.jung@samsung.com](mailto: jy1210.jung@samsung.com)

Bug No known bugs except for NYI items

9.33.2 Macro Definition Documentation

9.33.2.1 GST_IS_TENSOR_CROP

```
#define GST_IS_TENSOR_CROP(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_CROP))
```

Definition at line 28 of file `gsttensor_crop.h`.

9.33.2.2 GST_IS_TENSOR_CROP_CLASS

```
#define GST_IS_TENSOR_CROP_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_CROP))
```

Definition at line 30 of file `gsttensor_crop.h`.

9.33.2.3 GST_TENSOR_CROP

```
#define GST_TENSOR_CROP(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_CROP, GstTensorCrop))
```

Definition at line 24 of file `gsttensor_crop.h`.

9.33.2.4 GST_TENSOR_CROP_CLASS

```
#define GST_TENSOR_CROP_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_CROP, GstTensorCropClass))
```

Definition at line 26 of file `gsttensor_crop.h`.

9.33.2.5 GST_TYPE_TENSOR_CROP

```
#define GST_TYPE_TENSOR_CROP (gst_tensor_crop_get_type())
```

Definition at line 22 of file `gsttensor_crop.h`.

9.33.3 Typedef Documentation

9.33.3.1 GstTensorCrop

```
typedef struct _GstTensorCrop GstTensorCrop
```

Definition at line 33 of file gsttensor_crop.h.

9.33.3.2 GstTensorCropClass

```
typedef struct _GstTensorCropClass GstTensorCropClass
```

Definition at line 34 of file gsttensor_crop.h.

9.33.4 Function Documentation

9.33.4.1 gst_tensor_crop_get_type()

```
GType gst_tensor_crop_get_type (
    void )
```

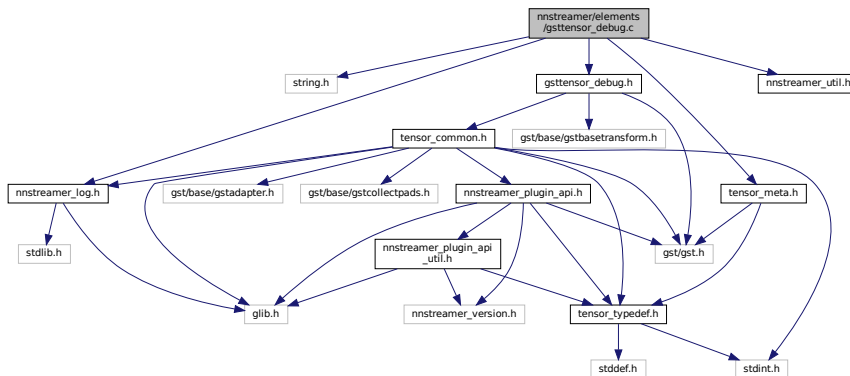
Function to get type of tensor_crop.

9.34 nnstreamer/elements/gsttensor_debug.c File Reference

GStreamer plugin to help debug tensor streams.

```
#include <string.h>
#include <nnstreamer_log.h>
#include <nnstreamer_util.h>
#include "gsttensor_debug.h"
#include "tensor_meta.h"
```

Include dependency graph for gsttensor_debug.c:



Macros

- #define `DBG` (`!self->silent`)
Macro for debug mode.
- #define `GST_CAT_DEFAULT` `gst_tensor_debug_debug`
- #define `CAPS_STRING` `GST_TENSORS_CAP_MAKE(GST_TENSOR_FORMAT_ALL)`
- #define `C_FLAGS`(v) `((guint) v)`
- #define `TENSOR_DEBUG_TYPE_OUTPUT_FLAGS` (`tensor_debug_output_flags_get_type()`)
- #define `DEFAULT_TENSOR_DEBUG_OUTPUT_FLAGS` (`TDBG_OUTPUT_CONSOLE_I`)
- #define `TENSOR_DEBUG_TYPE_CAPS` (`tensor_debug_cap_get_type()`)
- #define `DEFAULT_TENSOR_DEBUG_CAP` (`TDBG_CAP_SHOW_UPDATE_F`)
- #define `TENSOR_DEBUG_TYPE_META_FLAGS` (`tensor_debug_meta_flags_get_type()`)
- #define `DEFAULT_TENSOR_DEBUG_META_FLAGS` (`TDBG_META_DISABLED`)
- #define `DEFAULT_SILENT` `TRUE`
Flag to print minimized log.
- #define `gst_tensor_debug_parent_class` `parent_class`

Enumerations

- enum {
 `PROP_0`, `PROP_SILENT`, `PROP_OUTPUT`, `PROP_CAP`,
 `PROP_META` }
tensor_debug properties

Functions

- `GST_DEBUG_CATEGORY_STATIC` (`gst_tensor_debug_debug`)
- static GType `tensor_debug_output_flags_get_type` (void)
Flags for output_mode of GstTensorDebug.
- static GType `tensor_debug_cap_get_type` (void)
Enums for cap_mode of GstTensorDebug.
- static GType `tensor_debug_meta_flags_get_type` (void)
Flags for meta_mode of GstTensorDebug.
- `G_DEFINE_TYPE` (`GstTensorDebug`, `gst_tensor_debug`, `GST_TYPE_BASE_TRANSFORM`)
- static void `gst_tensor_debug_set_property` (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
Setter for tensor_debug properties.
- static void `gst_tensor_debug_get_property` (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)
Getter for tensor_debug properties.
- static void `gst_tensor_debug_finalize` (`GObject *object`)
Function to finalize instance.
- static `GstFlowReturn` `gst_tensor_debug_transform_ip` (`GstBaseTransform *trans`, `GstBuffer *buffer`)
in-place transform
- static `GstCaps *` `gst_tensor_debug_fixate_caps` (`GstBaseTransform *trans`, `GstPadDirection direction`, `GstCaps *caps`, `GstCaps *othercaps`)
fixate caps. required vmethod of GstBaseTransform.
- static `gboolean` `gst_tensor_debug_set_caps` (`GstBaseTransform *trans`, `GstCaps *in_caps`, `GstCaps *out_caps`)
set caps. required vmethod of GstBaseTransform.
- static void `gst_tensor_debug_class_init` (`GstTensorDebugClass *klass`)
Initialize the tensor_debug's class.
- static void `gst_tensor_debug_init` (`GstTensorDebug *self`)
Initialize tensor_debug element.
- static void `_gst_tensor_debug_output` (`GstTensorDebug *self`, `GstBuffer *buffer`)
The core function that provides debug output based on the contents.

Variables

- static GstStaticPadTemplate [sink_factory](#)
The capabilities of the inputs.
- static GstStaticPadTemplate [src_factory](#)
The capabilities of the outputs.

9.34.1 Detailed Description

GStreamer plugin to help debug tensor streams.

GStreamer/NNStreamer tensor_debug Copyright (C) 2022 MyungJoo Ham myungjoo.ham@samsung.com

Date

23 Sep 2022

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.34.2 Macro Definition Documentation

9.34.2.1 C_FLAGS

```
#define C_FLAGS(  
    v ) ((guint) v)
```

Definition at line 88 of file gsttensor_debug.c.

9.34.2.2 CAPS_STRING

```
#define CAPS_STRING GST\_TENSORS\_CAP\_MAKE(GST\_TENSOR\_FORMAT\_ALL)
```

This is a new element created after the obsolescence of other/tensor. Use other/tensors if you want to use tensor_debug

Definition at line 58 of file gsttensor_debug.c.

9.34.2.3 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

SECTION:element-tensor_debug

A filter that generates debug messages for developer at the insertion point of the given pipeline. An application writer using an nntstreamer pipeline can use tensor_debug to debug or get profile information in their applications.

Note that this does not support other/tensor, but only supports other/tensors.

```
<refsect2> <title>Example launch line</title> |[ gst-launch-1.0 videotestsrc ! video/x-raw,format=R↔GB,width=640,height=480 ! tensor_converter ! tensor_debug output-method=console-info capability=always ! tensor_sink ]| </refsect2>
```

Definition at line 48 of file gsttensor_debug.c.

9.34.2.4 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 191 of file gsttensor_debug.c.

9.34.2.5 DEFAULT_TENSOR_DEBUG_CAP

```
#define DEFAULT_TENSOR_DEBUG_CAP (TDBG_CAP_SHOW_UPDATE_F)
```

Definition at line 161 of file gsttensor_debug.c.

9.34.2.6 DEFAULT_TENSOR_DEBUG_META_FLAGS

```
#define DEFAULT_TENSOR_DEBUG_META_FLAGS (TDBG_META_DISABLED)
```

Definition at line 186 of file gsttensor_debug.c.

9.34.2.7 DEFAULT_TENSOR_DEBUG_OUTPUT_FLAGS

```
#define DEFAULT_TENSOR_DEBUG_OUTPUT_FLAGS (TDBG_OUTPUT_CONSOLE_I)
```

Definition at line 135 of file gsttensor_debug.c.

9.34.2.8 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_debug_debug
```

Definition at line 52 of file gsttensor_debug.c.

9.34.2.9 gst_tensor_debug_parent_class

```
#define gst_tensor_debug_parent_class parent_class
```

Definition at line 193 of file gsttensor_debug.c.

9.34.2.10 TENSOR_DEBUG_TYPE_CAPS

```
#define TENSOR_DEBUG_TYPE_CAPS (tensor_debug_cap_get_type())
```

Definition at line 137 of file gsttensor_debug.c.

9.34.2.11 TENSOR_DEBUG_TYPE_META_FLAGS

```
#define TENSOR_DEBUG_TYPE_META_FLAGS (tensor_debug_meta_flags_get_type())
```

Definition at line 163 of file gsttensor_debug.c.

9.34.2.12 TENSOR_DEBUG_TYPE_OUTPUT_FLAGS

```
#define TENSOR_DEBUG_TYPE_OUTPUT_FLAGS (tensor_debug_output_flags_get_type())
```

Definition at line 90 of file gsttensor_debug.c.

9.34.3 Enumeration Type Documentation

9.34.3.1 anonymous enum

```
anonymous enum
```

tensor_debug properties

Enumerator

PROP_0	
PROP_SILENT	
PROP_OUTPUT	
PROP_CAP	
PROP_META	

Definition at line 79 of file gsttensor_debug.c.

9.34.4 Function Documentation

9.34.4.1 `_gst_tensor_debug_output()`

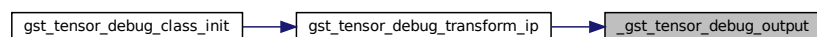
```
static void _gst_tensor_debug_output (
    GstTensorDebug * self,
    GstBuffer * buffer ) [static]
```

The core function that provides debug output based on the contents.

Todo NYI: do the debug task

Definition at line 397 of file gsttensor_debug.c.

Here is the caller graph for this function:



9.34.4.2 `G_DEFINE_TYPE()`

```
G_DEFINE_TYPE (
    GstTensorDebug ,
    gst_tensor_debug ,
    GST_TYPE_BASE_TRANSFORM )
```

9.34.4.3 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_debug_debug )
```

9.34.4.4 gst_tensor_debug_class_init()

```
static void gst_tensor_debug_class_init (
    GstTensorDebugClass * klass ) [static]
```

Initialize the tensor_debug's class.

[GstTensorDebug::silent](#):

The flag to enable/disable debugging messages.

[GstTensorDebug::output](#):

The combination of enums configuring output methods.

Todo check the behavior of name and nick (output methods vs output)

[GstTensorDebug::cap](#):

The logging preference of the stream capability (GSTCAP).

[GstTensorDebug::meta](#):

The logging preference of in-stream metadata (GSTMETA).

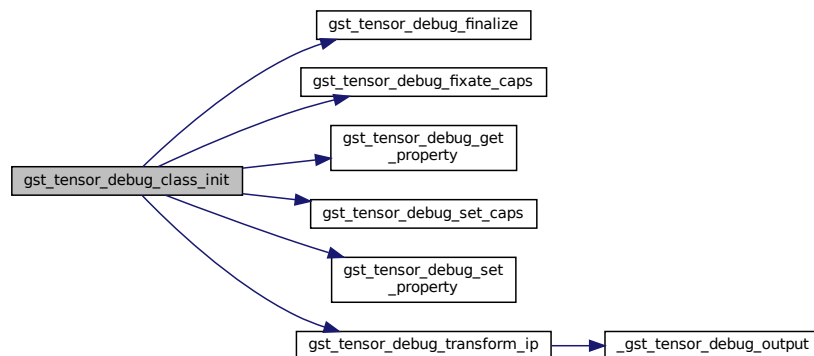
This won't modify the contents!

call transform_ip although it's passthrough

Note. Without transform_caps and with passthrough_on_same_caps = TRUE, This element is not allowed to touch the contents, but can inspect the contents with transform_ip by setting transform_ip_on_passthrough.

Definition at line 215 of file gsttensor_debug.c.

Here is the call graph for this function:



9.34.4.5 `gst_tensor_debug_finalize()`

```
static void gst_tensor_debug_finalize (  
    GObject * object ) [static]
```

Function to finalize instance.

Definition at line 327 of file `gsttensor_debug.c`.

Here is the caller graph for this function:



9.34.4.6 `gst_tensor_debug_fixate_caps()`

```
static GstCaps * gst_tensor_debug_fixate_caps (  
    GstBaseTransform * trans,  
    GstPadDirection direction,  
    GstCaps * caps,  
    GstCaps * othercaps ) [static]
```

fixate caps. required vmethod of `GstBaseTransform`.

Definition at line 421 of file `gsttensor_debug.c`.

Here is the caller graph for this function:



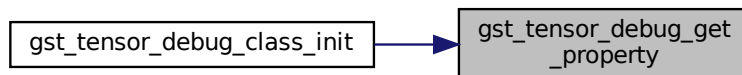
9.34.4.7 `gst_tensor_debug_get_property()`

```
static void gst_tensor_debug_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

Getter for `tensor_debug` properties.

Definition at line 368 of file `gsttensor_debug.c`.

Here is the caller graph for this function:



9.34.4.8 `gst_tensor_debug_init()`

```
static void gst_tensor_debug_init (
    GstTensorDebug * self ) [static]
```

Initialize `tensor_debug` element.

init properties

Definition at line 313 of file `gsttensor_debug.c`.

9.34.4.9 `gst_tensor_debug_set_caps()`

```
static gboolean gst_tensor_debug_set_caps (
    GstBaseTransform * trans,
    GstCaps * incaps,
    GstCaps * outcaps ) [static]
```

set caps. required vmethod of `GstBaseTransform`.

Definition at line 435 of file `gsttensor_debug.c`.

Here is the caller graph for this function:



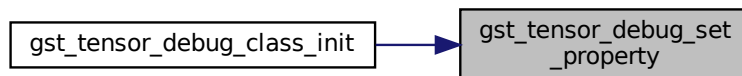
9.34.4.10 `gst_tensor_debug_set_property()`

```
static void gst_tensor_debug_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

Setter for `tensor_debug` properties.

Definition at line 336 of file `gsttensor_debug.c`.

Here is the caller graph for this function:



9.34.4.11 `gst_tensor_debug_transform_ip()`

```
static GstFlowReturn gst_tensor_debug_transform_ip (
    GstBaseTransform * trans,
    GstBuffer * buffer ) [static]
```

in-place transform

Definition at line 408 of file `gsttensor_debug.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.34.4.12 `tensor_debug_cap_get_type()`

```
static GType tensor_debug_cap_get_type (  
    void ) [static]
```

Enums for `cap_mode` of `GstTensorDebug`.

Definition at line 142 of file `gsttensor_debug.c`.

9.34.4.13 `tensor_debug_meta_flags_get_type()`

```
static GType tensor_debug_meta_flags_get_type (  
    void ) [static]
```

Flags for `meta_mode` of `GstTensorDebug`.

Definition at line 168 of file `gsttensor_debug.c`.

9.34.4.14 `tensor_debug_output_flags_get_type()`

```
static GType tensor_debug_output_flags_get_type (  
    void ) [static]
```

Flags for `output_mode` of `GstTensorDebug`.

Definition at line 95 of file `gsttensor_debug.c`.

9.34.5 Variable Documentation

9.34.5.1 `sink_factory`

```
GstStaticPadTemplate sink_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the inputs.

Definition at line 63 of file `gsttensor_debug.c`.

9.34.5.2 src_factory

```
GstStaticPadTemplate src_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (CAPS_STRING))
```

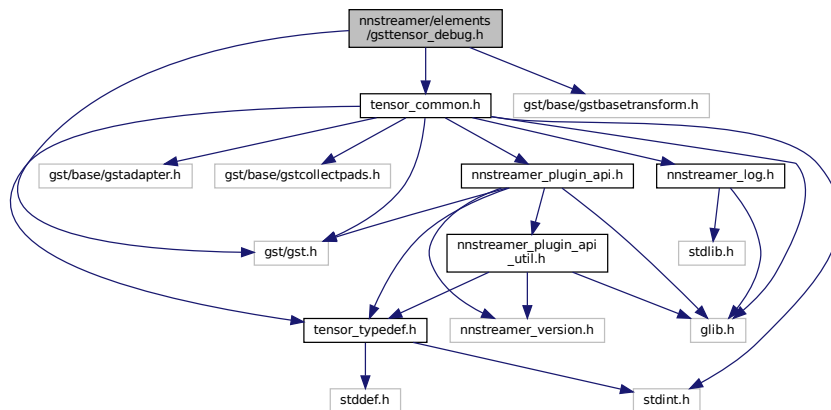
The capabilities of the outputs.

Definition at line 71 of file gsttensor_debug.c.

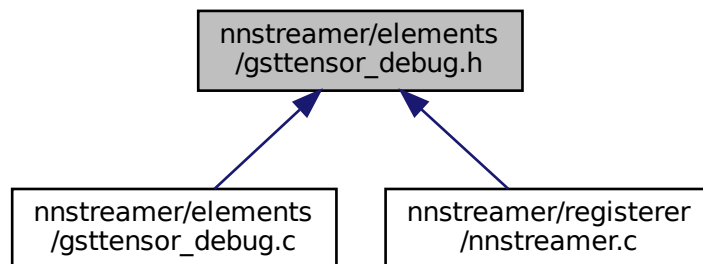
9.35 nnstreamer/elements/gsttensor_debug.h File Reference

GStreamer plugin to help debug tensor streams.

```
#include <gst/gst.h>
#include <gst/base/gstbasetransform.h>
#include <tensor_common.h>
Include dependency graph for gsttensor_debug.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorDebug](#)
Internal data structure for tensor_debug instances.
- struct [_GstTensorDebugClass](#)
GstTensorDebugClass data structure.

Macros

- #define [GST_TYPE_TENSOR_DEBUG](#) ([gst_tensor_debug_get_type](#)())
- #define [GST_TENSOR_DEBUG](#)(obj) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj),[GST_TYPE_TENSOR_DEBUG](#),[GstTensorD](#)
- #define [GST_TENSOR_DEBUG_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_CAST](#)((klass),[GST_TYPE_TENSOR_DEBUG](#),[Gs](#)
- #define [GST_IS_TENSOR_DEBUG](#)(obj) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_DEBUG](#)))
- #define [GST_IS_TENSOR_DEBUG_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_DEBUG](#)
- #define [GST_TENSOR_DEBUG_CAST](#)(obj) (([GstTensorDebug](#) *) (obj))

Typedefs

- typedef struct [_GstTensorDebug](#) [GstTensorDebug](#)
- typedef struct [_GstTensorDebugClass](#) [GstTensorDebugClass](#)

Enumerations

- enum [tdbg_output_mode](#) {
[TDBG_OUTPUT_DISABLED](#) = 0x00, [TDBG_OUTPUT_CONSOLE_I](#) = 0x11, [TDBG_OUTPUT_CONSOLE_W](#)
= 0x12, [TDBG_OUTPUT_CONSOLE_E](#) = 0x13,
[TDBG_OUTPUT_GSTDBG_I](#) = 0x24, [TDBG_OUTPUT_GSTDBG_W](#) = 0x28, [TDBG_OUTPUT_GSTDBG_E](#)
= 0x2C, [TDBG_OUTPUT_CIRCULARBUF](#) = 0x100,
[TDBG_OUTPUT_FILEWRITE](#) = 0x200 }
Property "OUTPUT" specification.
- enum [tdbg_cap_mode](#) { [TDBG_CAP_DISABLED](#) = 0, [TDBG_CAP_SHOW_UPDATE](#) = 1, [TDBG_CAP_SHOW_UPDATE_F](#)
= 2, [TDBG_CAP_SHOW_ALWAYS](#) = 3 }
Property "CAP" specification.
- enum [tdbg_meta_mode](#) { [TDBG_META_DISABLED](#) = 0x0, [TDBG_META_TIMESTAMP](#) = 0x1,
[TDBG_META_QUERYSERVER](#) = 0x2 }
Property "META" specification.

Functions

- GType [gst_tensor_debug_get_type](#) (void)
Get Type function required for gst elements.

9.35.1 Detailed Description

GStreamer plugin to help debug tensor streams.

GStreamer/NNStreamer tensor_debug Copyright (C) 2022 MyungJoo Ham myungjoo.ham@samsung.com

Date

23 Sep 2022

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.35.2 Macro Definition Documentation

9.35.2.1 GST_IS_TENSOR_DEBUG

```
#define GST_IS_TENSOR_DEBUG(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_DEBUG))
```

Definition at line 35 of file gstreamer_debug.h.

9.35.2.2 GST_IS_TENSOR_DEBUG_CLASS

```
#define GST_IS_TENSOR_DEBUG_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_DEBUG))
```

Definition at line 37 of file gstreamer_debug.h.

9.35.2.3 GST_TENSOR_DEBUG

```
#define GST_TENSOR_DEBUG(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_DEBUG, GstTensorDebug))
```

Definition at line 31 of file gstreamer_debug.h.

9.35.2.4 GST_TENSOR_DEBUG_CAST

```
#define GST_TENSOR_DEBUG_CAST(  
    obj ) ((GstTensorDebug *) (obj))
```

Definition at line 39 of file gsttensor_debug.h.

9.35.2.5 GST_TENSOR_DEBUG_CLASS

```
#define GST_TENSOR_DEBUG_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_DEBUG, GstTensorDebugClass))
```

Definition at line 33 of file gsttensor_debug.h.

9.35.2.6 GST_TYPE_TENSOR_DEBUG

```
#define GST_TYPE_TENSOR_DEBUG (gst_tensor_debug_get_type())
```

Definition at line 29 of file gsttensor_debug.h.

9.35.3 Typedef Documentation

9.35.3.1 GstTensorDebug

```
typedef struct _GstTensorDebug GstTensorDebug
```

Definition at line 41 of file gsttensor_debug.h.

9.35.3.2 GstTensorDebugClass

```
typedef struct _GstTensorDebugClass GstTensorDebugClass
```

Definition at line 42 of file gsttensor_debug.h.

9.35.4 Enumeration Type Documentation

9.35.4.1 tdbg_cap_mode

```
enum tdbg_cap_mode
```

Property "CAP" specification.

Enumerator

TDBG_CAP_DISABLED	No output for tensor-capability info
TDBG_CAP_SHOW_UPDATE	Output tensor-capability if there is a change in gstcap
TDBG_CAP_SHOW_UPDATE↔ _F	Output tensor-capability if there is a change in dimensions even if gstcap has no changes. This happens with format=flexible or sparse
TDBG_CAP_SHOW_ALWAYS	

Definition at line 63 of file gsttensor_debug.h.

9.35.4.2 tdbg_meta_mode

enum [tdbg_meta_mode](#)

Property "META" specification.

Enumerator

TDBG_META_DISABLED	Don't bring up metadata of a stream
TDBG_META_TIMESTAMP	Enable timestamp info
TDBG_META_QUERYSERVER	Enable tensor-query-server related info

Definition at line 74 of file gsttensor_debug.h.

9.35.4.3 tdbg_output_mode

enum [tdbg_output_mode](#)

Property "OUTPUT" specification.

Enumerator

TDBG_OUTPUT_DISABLED	disable output.
TDBG_OUTPUT_CONSOLE_I	console/debug info
TDBG_OUTPUT_CONSOLE_W	console/debug warn
TDBG_OUTPUT_CONSOLE_E	console/debug error (non fatal)
TDBG_OUTPUT_GSTDBG_I	gst-debug info
TDBG_OUTPUT_GSTDBG_W	gst-debug warn
TDBG_OUTPUT_GSTDBG_E	gst-debug error (non fatal)
TDBG_OUTPUT_CIRCULARBUF	in circular buffer for later retrievals. (Todo NYI NOT_SUPPORTED)
TDBG_OUTPUT_FILEWRITE	Write to a file. (Todo NYI NOT_SUPPORTED)

Enumerations

- enum {
[PROP_0](#), [PROP_SILENT](#), [PROP_MODE](#), [PROP_MODE_OPTION1](#),
[PROP_MODE_OPTION2](#), [PROP_MODE_OPTION3](#), [PROP_MODE_OPTION4](#), [PROP_MODE_OPTION5](#),
[PROP_MODE_OPTION6](#), [PROP_MODE_OPTION7](#), [PROP_MODE_OPTION8](#), [PROP_MODE_OPTION9](#),
[PROP_SUBPLUGINS](#), [PROP_CONFIG](#) }

Properties.

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensordec_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorDecoder](#), [gst_tensordec](#), [GST_TYPE_BASE_TRANSFORM](#))
- static void [gst_tensordec_set_property](#) ([GObject](#) *object, guint prop_id, const [GValue](#) *value, [GParamSpec](#) *pspec)
Set property (GObject vmethod)
- static void [gst_tensordec_get_property](#) ([GObject](#) *object, guint prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
Get property (GObject vmethod)
- static void [gst_tensordec_class_finalize](#) ([GObject](#) *object)
Finalize instance (GObject vmethod)
- static [GstFlowReturn](#) [gst_tensordec_transform](#) ([GstBaseTransform](#) *trans, [GstBuffer](#) *inbuf, [GstBuffer](#) *outbuf)
non-ip transform. required vmethod for BaseTransform class.
- static [GstCaps](#) * [gst_tensordec_transform_caps](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [GstCaps](#) *filter)
configure tensor-srcpad cap from "proposed" cap.
- static [GstCaps](#) * [gst_tensordec_fixate_caps](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [GstCaps](#) *othercaps)
fixate caps. required vmethod of BaseTransform
- static gboolean [gst_tensordec_set_caps](#) ([GstBaseTransform](#) *trans, [GstCaps](#) *incaps, [GstCaps](#) *outcaps)
set caps. required vmethod of BaseTransform
- static gboolean [gst_tensordec_transform_size](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, gsize size, [GstCaps](#) *othercaps, gsize *othersize)
Tell the framework the required size of buffer based on the info of the other side pad. optional vmethod of BaseTransform.
- static gboolean [nntstreamer_decoder_validate](#) (const [GstTensorDecoderDef](#) *decoder)
Validate decoder sub-plugin's data.
- int [nntstreamer_decoder_probe](#) ([GstTensorDecoderDef](#) *decoder)
Decoder's sub-plugin should call this function to register itself.
- void [nntstreamer_decoder_exit](#) (const char *name)
Decoder's sub-plugin may call this to unregister itself.
- const [GstTensorDecoderDef](#) * [nntstreamer_decoder_find](#) (const char *name)
Find decoder sub-plugin with the name.
- void [nntstreamer_decoder_set_custom_property_desc](#) (const char *name, const char *prop,...)
set custom property description for tensor decoder sub-plugin
- static [GstCaps](#) * [gst_tensordec_media_caps_from_tensor](#) ([GstTensorDecoder](#) *self, const [GstTensorsConfig](#) *config)
Get media caps from tensor config.
- static [GstCaps](#) * [gst_tensordec_media_caps_from_structure](#) ([GstTensorDecoder](#) *self, const [GstStructure](#) *structure)
Parse structure and return media caps.

- static gboolean [gst_tensordec_check_consistency](#) ([GstTensorDecoder](#) *self, [GstTensorsConfig](#) *config)
 - Check tensor config is consistent.*
- static void [gst_tensordec_class_init](#) ([GstTensorDecoderClass](#) *klass)
 - initialize the tensordec's class*
- static void [gst_tensordec_init](#) ([GstTensorDecoder](#) *self)
 - initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure*
- static gboolean [gst_tensordec_process_plugin_options](#) ([GstTensorDecoder](#) *self, guint [opnum](#))
 - Process plugin (self->decoder) with given options if available.*
- [g_free](#) (self->[option](#)[([opnum](#)) - 1])
 - opnum: *
- if ([!gst_tensordec_process_plugin_options](#)(self, ([opnum](#)) - 1)) [GST_ERROR_OBJECT](#)(self)
- Configuring [option](#) for tensor decoder failed ([option](#) %d=%s)"
- [g_value_set_string](#) (value, self->[option](#)[[opnum](#) - 1])
 - opnum: *
- static gboolean [gst_tensordec_configure](#) ([GstTensorDecoder](#) *self, const [GstCaps](#) *in_caps, const [GstCaps](#) *out_caps)
 - Configure tensor metadata from sink caps.*
- int [nnstreamer_decoder_custom_register](#) (const gchar *name, [tensor_decoder_custom](#) func, void *data)
 - Registers a callback for tensor_decoder custom condition.*
- int [nnstreamer_decoder_custom_unregister](#) (const gchar *name)
 - Unregisters a callback for tensor_decoder custom condition.*

Variables

- static [GstStaticPadTemplate](#) [sink_factory](#)
 - The capabilities of the inputs.*
- static [GstStaticPadTemplate](#) [src_factory](#)
 - The capabilities of the outputs.*
- self [option](#) [([opnum](#)) - 1] = [g_value_dup_string](#) (value)
- Configuring [option](#) for tensor decoder [opnum](#)

9.36.1 Detailed Description

GStreamer plugin to convert tensors (as a filter for other general neural network filters) to other media types.

GStreamer / NNStreamer tensor_decoder main Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

26 Mar 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jiyoong.moon@samsung.com

Bug `gst_tensordec_transform_size ()` may be incorrect if direction is SINK.

If `configured = TRUE`, it holds TRUE until exit. What if configuration changes in run-time?

9.36.2 Macro Definition Documentation

9.36.2.1 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_DEFAULT ";" GST_TENSORS_FLEX_CAP_DEFAULT
```

Support multi-tensor along with single-tensor as the input.

Definition at line 89 of file `gsttensor_decoder.c`.

9.36.2.2 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

SECTION:element-tensordec

A filter that converts tensor stream for NN frameworks to media stream. The input is always in the format of other/tensor

```
<refsect2> <title>Example launch line</title> |[ gst-launch -v -m fakesink ! tensor_decoder ! fakesrc silent=TRUE ]| </refsect2>
```

Definition at line 54 of file `gsttensor_decoder.c`.

9.36.2.3 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 84 of file `gsttensor_decoder.c`.

9.36.2.4 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensordec_debug
```

Definition at line 58 of file gsttensor_decoder.c.

9.36.2.5 gst_tensor_decoder_clean_plugin

```
#define gst_tensor_decoder_clean_plugin(  
    self )
```

Value:

```
do { \
    if (self->decoder) { \
        if (self->decoder->exit) \
            self->decoder->exit (&self->plugin_data); \
        else \
            g_free (self->plugin_data); \
        self->plugin_data = NULL; \
    } \
} while (0)
```

Macro to clean sub-plugin data.

Definition at line 200 of file gsttensor_decoder.c.

9.36.2.6 gst_tensordec_parent_class

```
#define gst_tensordec_parent_class parent_class
```

Definition at line 107 of file gsttensor_decoder.c.

9.36.2.7 PROP_MODE_OPTION

```
#define PROP_MODE_OPTION(  
    opnum ) case PROP_MODE_OPTION
```

A macro to process incoming per-mode option.

Parameters

in	<i>opnum</i>	The option number (1 to TensorDecMaxOpNum)
----	--------------	--

Definition at line 457 of file gsttensor_decoder.c.

9.36.2.8 PROP_READ_OPTION

```
#define PROP_READ_OPTION(  
    opnum ) case PROP_MODE_OPTION
```

A macro to read per-mode option.

Parameters

<i>in</i>	<i>opnum</i>	The option number (1 to TensorDecMaxOpNum)
-----------	--------------	--

Definition at line 556 of file gsttensor_decoder.c.

9.36.3 Enumeration Type Documentation

9.36.3.1 anonymous enum

anonymous enum

Properties.

Enumerator

PROP_0	
PROP_SILENT	
PROP_MODE	
PROP_MODE_OPTION1	
PROP_MODE_OPTION2	
PROP_MODE_OPTION3	
PROP_MODE_OPTION4	
PROP_MODE_OPTION5	
PROP_MODE_OPTION6	
PROP_MODE_OPTION7	
PROP_MODE_OPTION8	
PROP_MODE_OPTION9	
PROP_SUBPLUGINS	
PROP_CONFIG	

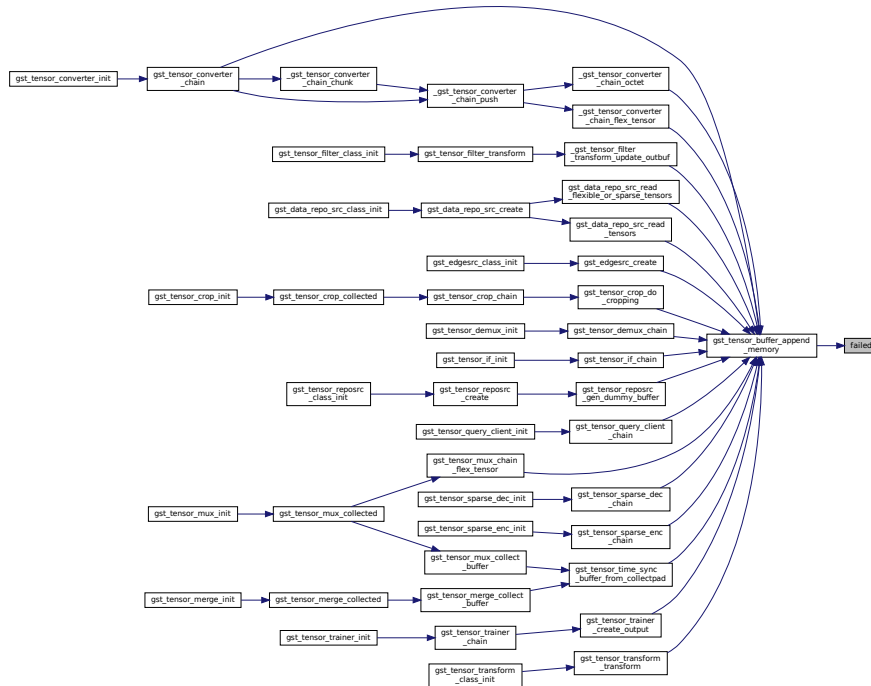
Definition at line 63 of file gsttensor_decoder.c.

9.36.4 Function Documentation

9.36.4.1 failed()

Configuring `option` for tensor decoder failed (
`option % d = %s)`

Here is the caller graph for this function:



9.36.4.2 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorDecoder ,
    gst_tensordec ,
    GST_TYPE_BASE_TRANSFORM )
```

9.36.4.3 g_free()

```
g_free (
    self-> option[(opnum) - 1] )
```

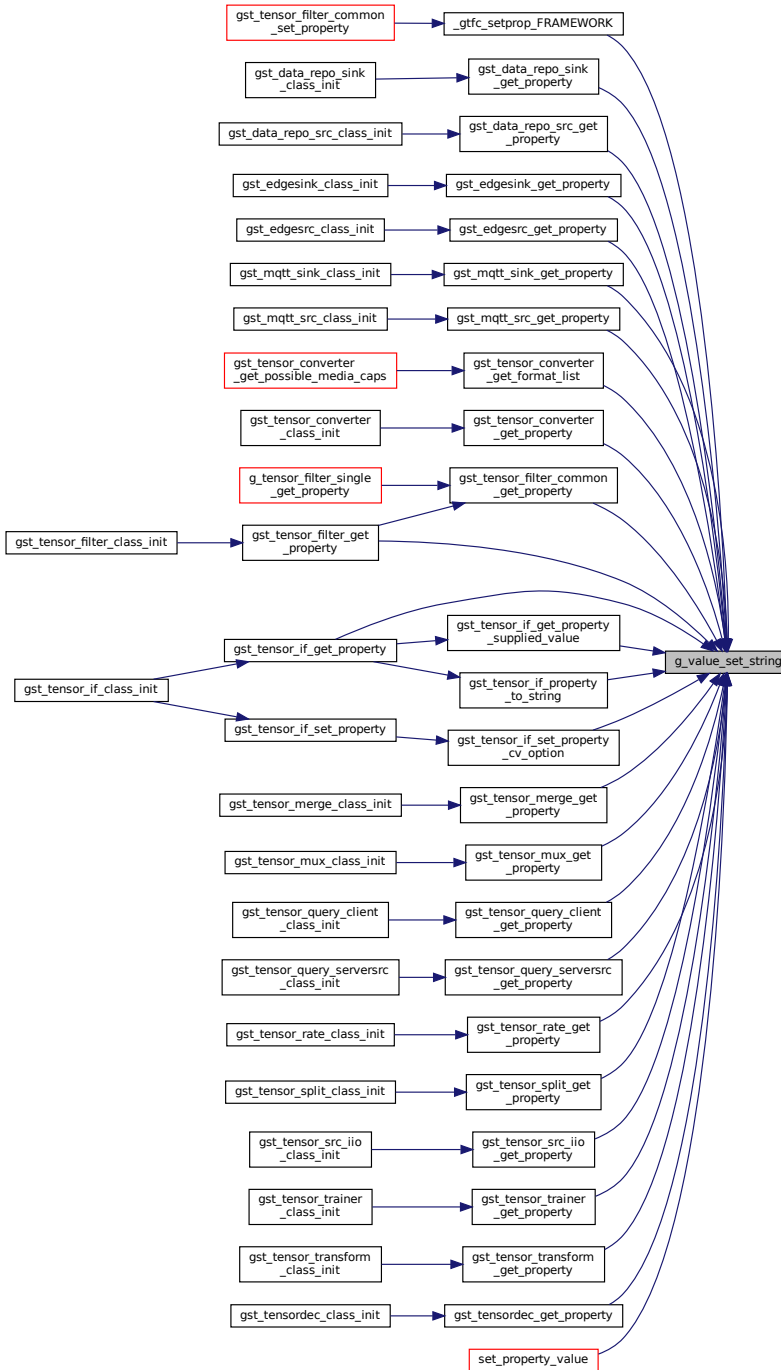
opnum: \

9.36.4.4 g_value_set_string()

```
g_value_set_string (
    value ,
    self-> option[opnum - 1] )
```

opnum: \

Here is the caller graph for this function:



9.36.4.5 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensordec_debug )
```

9.36.4.6 gst_tensordec_check_consistency()

```
static gboolean gst_tensordec_check_consistency (
    GstTensorDecoder * self,
    GstTensorsConfig * config ) [static]
```

Check tensor config is consistent.

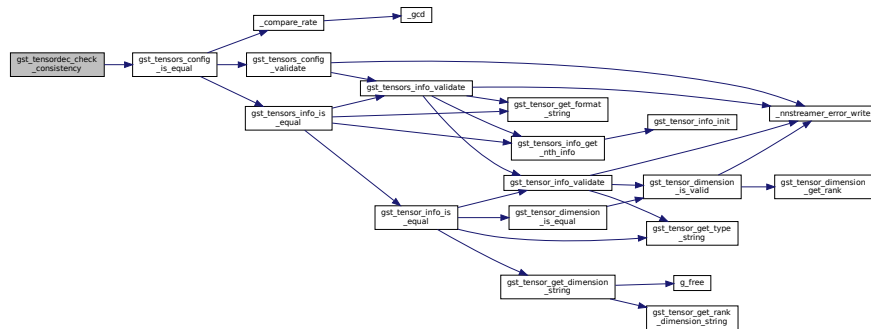
Parameters

<i>self</i>	"this" pointer to check consistency
<i>t_info</i>	newly configured tensor metadata

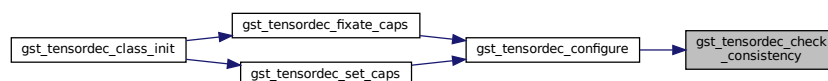
not configured yet

Definition at line 269 of file gsttensor_decoder.c.

Here is the call graph for this function:



Here is the caller graph for this function:



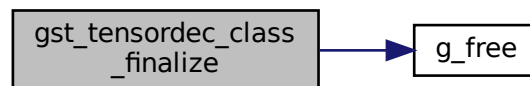
9.36.4.7 `gst_tensordec_class_finalize()`

```
static void gst_tensordec_class_finalize (  
    GObject * object ) [static]
```

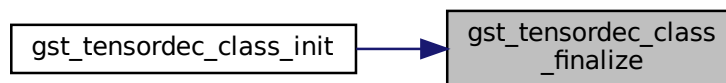
Finalize instance (GObject vmethod)

Definition at line 618 of file `gsttensor_decoder.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.36.4.8 `gst_tensordec_class_init()`

```
static void gst_tensordec_class_init (  
    GstTensorDecoderClass * klass ) [static]
```

initialize the tensordec's class

Refer: <https://gstreamer.freedesktop.org/documentation/design/element-transform.html>

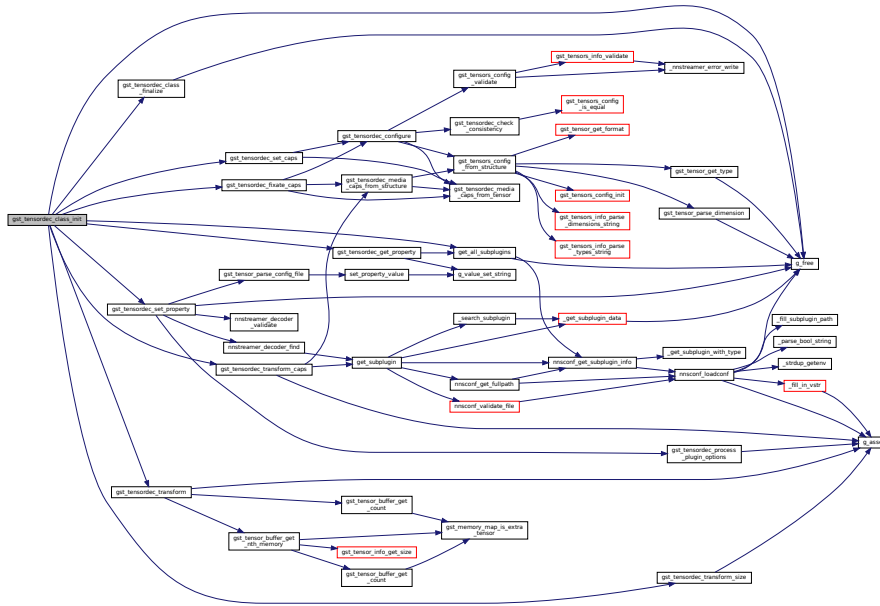
Processing units

Negotiation units

Allocation units

Definition at line 287 of file `gsttensor_decoder.c`.

Here is the call graph for this function:



9.36.4.9 gst_tensordec_configure()

```
static gboolean gst_tensordec_configure (
    GstTensorDecoder * self,
    const GstCaps * in_caps,
    const GstCaps * out_caps ) [static]
```

Configure tensor metadata from sink caps.

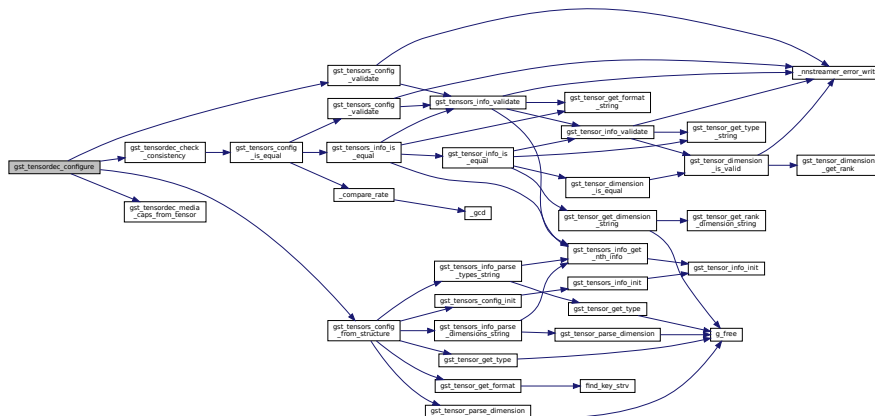
This caps is coming from tensor

If previous input configuration is set and is not compatible with incoming caps, get possible media caps from sub-plugin and change input configuration.

Check if outcaps is compatible with new caps

Definition at line 641 of file gsttensor_decoder.c.

Here is the call graph for this function:



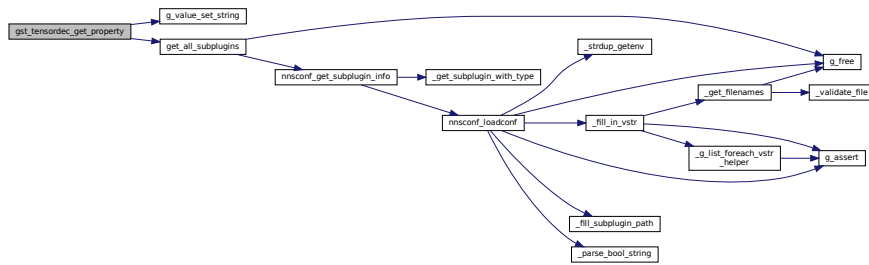
9.36.4.11 `gst_tensordec_get_property()`

```
static break void gst_tensordec_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

Get property (GObject vmethod)

Definition at line 565 of file `gsttensor_decoder.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.36.4.12 `gst_tensordec_init()`

```
static void gst_tensordec_init (
    GstTensorDecoder * self ) [static]
```

initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure

Definition at line 416 of file `gsttensor_decoder.c`.

Here is the call graph for this function:



9.36.4.13 `gst_tensordec_media_caps_from_structure()`

```
static GstCaps* gst_tensordec_media_caps_from_structure (
    GstTensorDecoder * self,
    const GstStructure * structure ) [static]
```

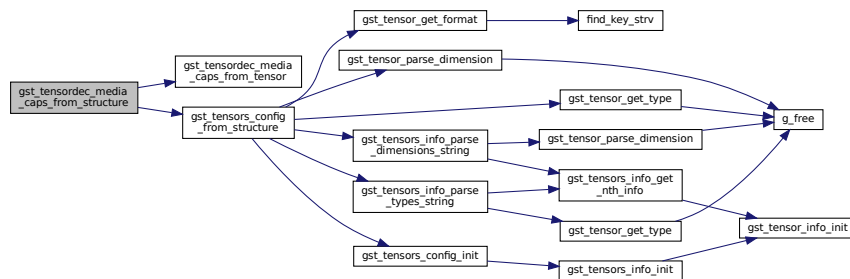
Parse structure and return media caps.

Parameters

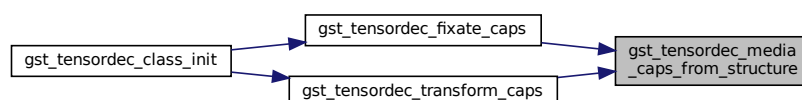
<i>self</i>	"this" pointer
<i>structure</i>	structure to be interpreted

Definition at line 245 of file `gsttensor_decoder.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.36.4.14 `gst_tensordec_media_caps_from_tensor()`

```
static GstCaps* gst_tensordec_media_caps_from_tensor (
    GstTensorDecoder * self,
    const GstTensorsConfig * config ) [static]
```

Get media caps from tensor config.

Parameters

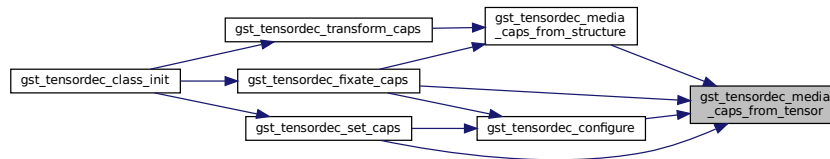
<i>self</i>	"this" pointer
<i>config</i>	tensor config info

Returns

caps for media type

Definition at line 217 of file gsttensor_decoder.c.

Here is the caller graph for this function:



9.36.4.15 gst_tensordec_process_plugin_options()

```

static gboolean gst_tensordec_process_plugin_options (
    GstTensorDecoder * self,
    guint opnum ) [static]
  
```

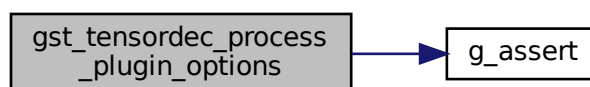
Process plugin (self->decoder) with given options if available.

Return values

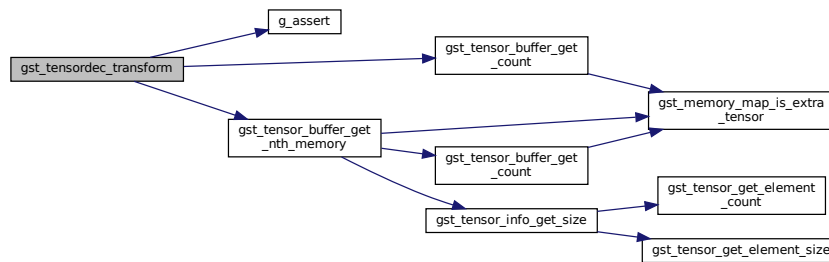
<i>FALSE</i>	if error. TRUE if OK (or SKIP)
--------------	--------------------------------

Definition at line 440 of file gsttensor_decoder.c.

Here is the call graph for this function:



Here is the call graph for this function:



Here is the caller graph for this function:



9.36.4.19 gst_tensordec_transform_caps()

```

static GstCaps * gst_tensordec_transform_caps (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    GstCaps * filter ) [static]
  
```

configure tensor-srcpad cap from "proposed" cap.

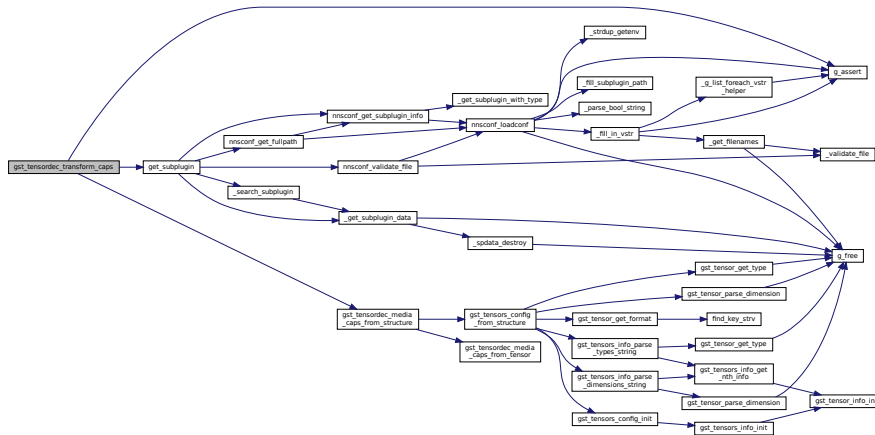
@trans ("this" pointer) @direction (why do we need this?) @caps sinkpad cap @filter this element's cap (don't know specifically.) caps = sinkpad (other/tensor) return = srcpad (media)

caps = srcpad (media) return = sinkpad (other/tensor)

Todo We may do more specific actions here

Definition at line 786 of file gsttensor_decoder.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.36.4.20 gst_tensordec_transform_size()

```
static gboolean gst_tensordec_transform_size (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    gsize size,
    GstCaps * othercaps,
    gsize * othersize ) [static]
```

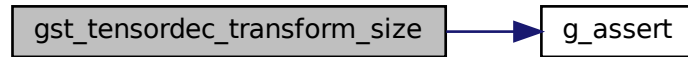
Tell the framework the required size of buffer based on the info of the other side pad. optional vmethod of Base↔ Transform.

This is called when non-ip mode is used.

Todo If direction = SRC, you may need different interpretation!

Definition at line 945 of file gsttensor_decoder.c.

Here is the call graph for this function:



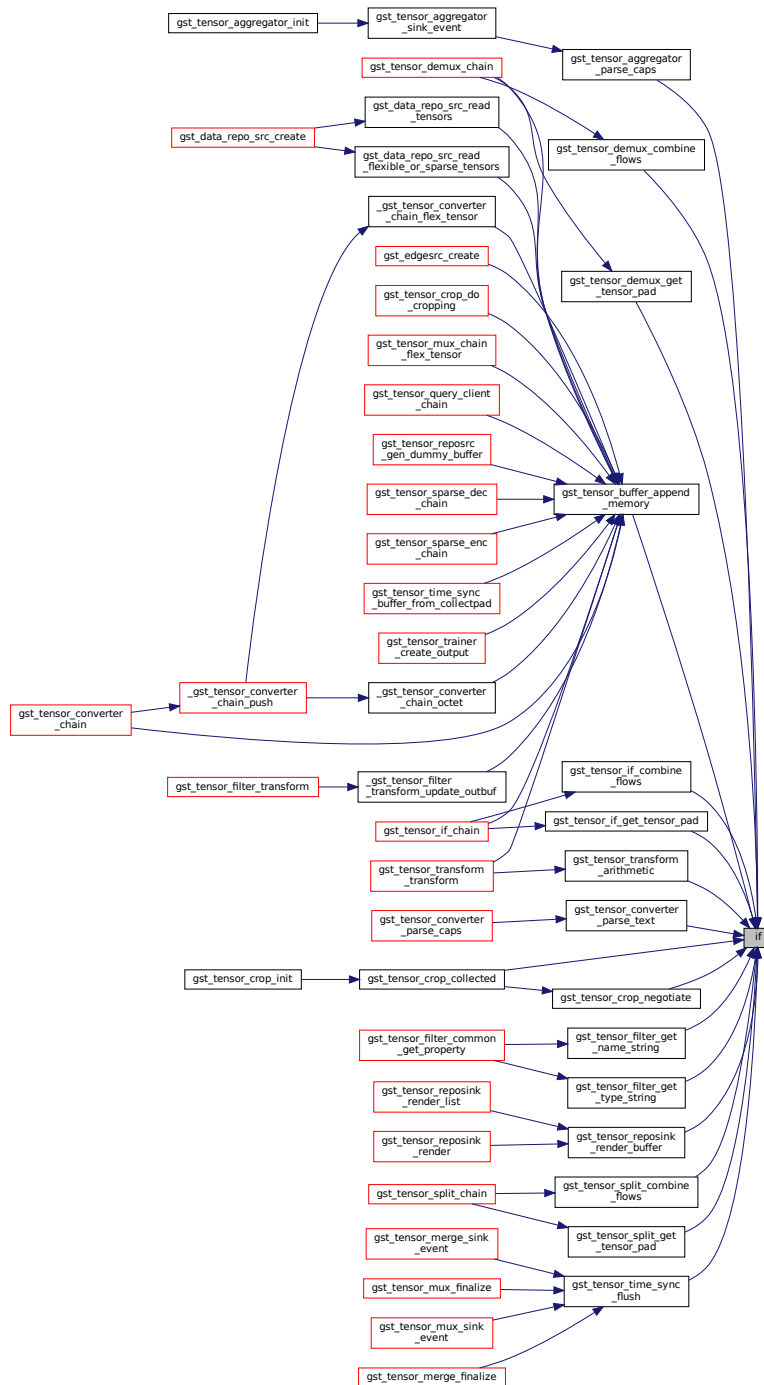
Here is the caller graph for this function:



9.36.4.21 if()

```
if (
    ! gst_tensordec_process_plugin_optionsself, (opnum) - 1 )
```

Here is the caller graph for this function:



9.36.4.22 nnstreamer_decoder_custom_register()

```
int nnstreamer_decoder_custom_register (
    const gchar * name,
```



```

    tensor_decoder_custom func,
    void * data )

```

Registers a callback for tensor_decoder custom condition.

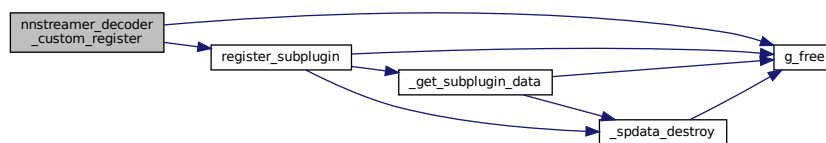
Register the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 972 of file gsttensor_decoder.c.

Here is the call graph for this function:



9.36.4.23 nntstreamer_decoder_custom_unregister()

```

int nntstreamer_decoder_custom_unregister (
    const gchar * name )

```

Unregisters a callback for tensor_decoder custom condition.

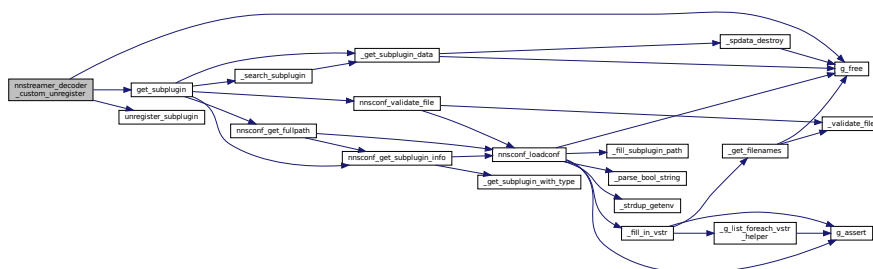
Unregister the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 998 of file gsttensor_decoder.c.

Here is the call graph for this function:



9.36.4.24 `nnstreamer_decoder_exit()`

```
void nnstreamer_decoder_exit (  
    const char * name )
```

Decoder's sub-plugin may call this to unregister itself.

Parameters

in	<i>name</i>	The name of decoder sub-plugin.
----	-------------	---------------------------------

Definition at line 166 of file gsttensor_decoder.c.

Here is the call graph for this function:



9.36.4.25 nnstreamer_decoder_find()

```

const GstTensorDecoderDef* nnstreamer_decoder_find (
    const char * name )
  
```

Find decoder sub-plugin with the name.

Parameters

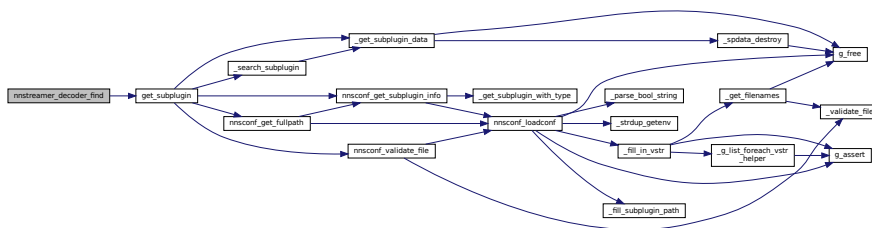
in	<i>name</i>	The name of decoder sub-plugin.
----	-------------	---------------------------------

Returns

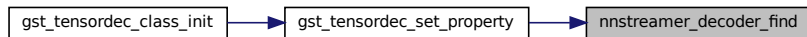
NULL if not found or the sub-plugin object has an error.

Definition at line 177 of file gsttensor_decoder.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.36.4.26 nnstreamer_decoder_probe()

```
int nnstreamer_decoder_probe (
    GstTensorDecoderDef * decoder )
```

Decoder's sub-plugin should call this function to register itself.

Parameters

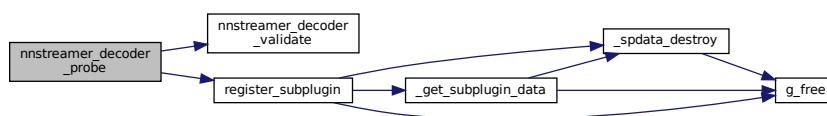
in	<i>decoder</i>	Decoder sub-plugin to be registered.
----	----------------	--------------------------------------

Returns

TRUE if registered. FALSE is failed or duplicated.

Definition at line 155 of file gsttensor_decoder.c.

Here is the call graph for this function:



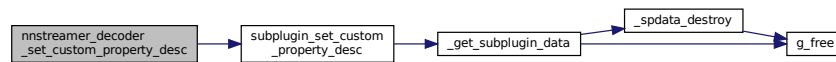
9.36.4.27 nnstreamer_decoder_set_custom_property_desc()

```
void nnstreamer_decoder_set_custom_property_desc (
    const char * name,
    const char * prop,
    ... )
```

set custom property description for tensor decoder sub-plugin

Definition at line 186 of file gsttensor_decoder.c.

Here is the call graph for this function:



9.36.4.28 nnstreamer_decoder_validate()

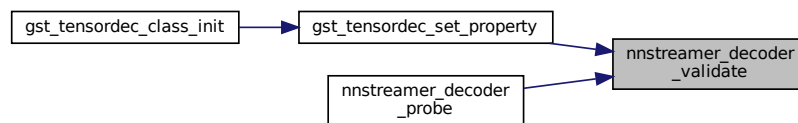
```

static gboolean nnstreamer_decoder_validate (
    const GstTensorDecoderDef * decoder ) [static]
  
```

Validate decoder sub-plugin's data.

Definition at line 134 of file gsttensor_decoder.c.

Here is the caller graph for this function:



9.36.5 Variable Documentation

9.36.5.1 opnum

Configuring `option` for tensor decoder `opnum`

Definition at line 463 of file gsttensor_decoder.c.

9.36.5.2 option

Configuring `option` for tensor decoder `self option = g_value_dup_string (value)`

Definition at line 460 of file gsttensor_decoder.c.

9.36.5.3 sink_factory

```
GstStaticPadTemplate sink_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the inputs.

Definition at line 94 of file gsttensor_decoder.c.

9.36.5.4 src_factory

```
GstStaticPadTemplate src_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS ("ANY"))
```

The capabilities of the outputs.

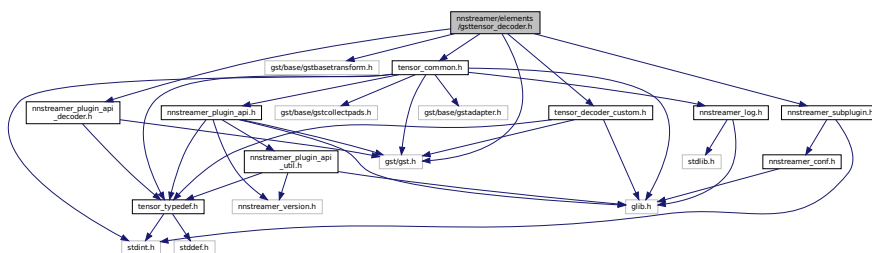
Definition at line 102 of file gsttensor_decoder.c.

9.37 nnstreamer/elements/gsttensor_decoder.h File Reference

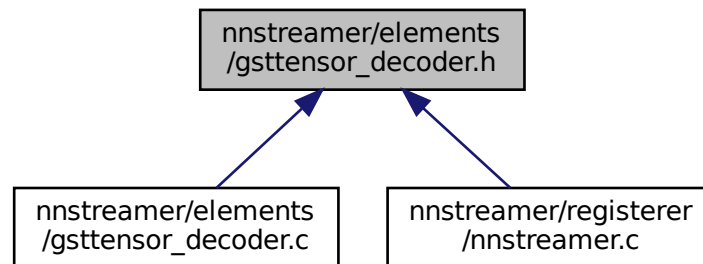
GStreamer plugin to convert tensors to media types.

```
#include <gst/gst.h>
#include <gst/base/gstbasetransform.h>
#include "tensor_common.h"
#include "nnstreamer_subplugin.h"
#include "nnstreamer_plugin_api_decoder.h"
#include "tensor_decoder_custom.h"
```

Include dependency graph for gsttensor_decoder.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [decoder_custom_cb_s](#)
- struct [_GstTensorDecoder](#)
Internal data structure for tensordec instances.
- struct [_GstTensorDecoderClass](#)
GstTensorDecoderClass inherits GstBaseTransformClass.

Macros

- #define [GST_TYPE_TENSOR_DECODER](#) ([gst_tensordec_get_type\(\)](#))
- #define [GST_TENSOR_DECODER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_DECODER, GstTensorDecoder\)](#))
- #define [GST_TENSOR_DECODER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_DECODER, _GstTensorDecoderClass\)](#))
- #define [GST_IS_TENSOR_DECODER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_DECODER\)](#))
- #define [GST_IS_TENSOR_DECODER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_DECODER\)](#))
- #define [GST_TENSOR_DECODER_CAST\(obj\)](#) ([\(\(GstTensorDecoder *\) \(obj\)\)](#))
- #define [TensorDecMaxOpNum](#) (9)

Typedefs

- typedef struct [_GstTensorDecoder](#) [GstTensorDecoder](#)
- typedef struct [_GstTensorDecoderClass](#) [GstTensorDecoderClass](#)

Functions

- GType [gst_tensordec_get_type](#) (void)
Get Type function required for gst elements.

9.37.1 Detailed Description

GStreamer plugin to convert tensors to media types.

GStreamer / NNStreamer tensor_decoder header Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

26 Mar 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.37.2 Macro Definition Documentation

9.37.2.1 GST_IS_TENSOR_DECODER

```
#define GST_IS_TENSOR_DECODER(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_DECODER))
```

Definition at line 48 of file gsttensor_decoder.h.

9.37.2.2 GST_IS_TENSOR_DECODER_CLASS

```
#define GST_IS_TENSOR_DECODER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_DECODER))
```

Definition at line 50 of file gsttensor_decoder.h.

9.37.2.3 GST_TENSOR_DECODER

```
#define GST_TENSOR_DECODER(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_DECODER, GstTensorDecoder))
```

Definition at line 44 of file gsttensor_decoder.h.

9.37.2.4 GST_TENSOR_DECODER_CAST

```
#define GST_TENSOR_DECODER_CAST(  
    obj ) ((GstTensorDecoder *) (obj))
```

Definition at line 52 of file gsttensor_decoder.h.

9.37.2.5 GST_TENSOR_DECODER_CLASS

```
#define GST_TENSOR_DECODER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_DECODER, GstTensorDecoderClass))
```

Definition at line 46 of file gsttensor_decoder.h.

9.37.2.6 GST_TYPE_TENSOR_DECODER

```
#define GST_TYPE_TENSOR_DECODER (gst_tensordec_get_type())
```

Definition at line 42 of file gsttensor_decoder.h.

9.37.2.7 TensorDecMaxOpNum

```
#define TensorDecMaxOpNum (9)
```

Definition at line 62 of file gsttensor_decoder.h.

9.37.3 Typedef Documentation

9.37.3.1 GstTensorDecoder

```
typedef struct _GstTensorDecoder GstTensorDecoder
```

Definition at line 54 of file gsttensor_decoder.h.

9.37.3.2 GstTensorDecoderClass

```
typedef struct _GstTensorDecoderClass GstTensorDecoderClass
```

Definition at line 55 of file gsttensor_decoder.h.

9.37.4 Function Documentation

9.37.4.1 gst_tensordec_get_type()

```
GType gst_tensordec_get_type (
    void )
```

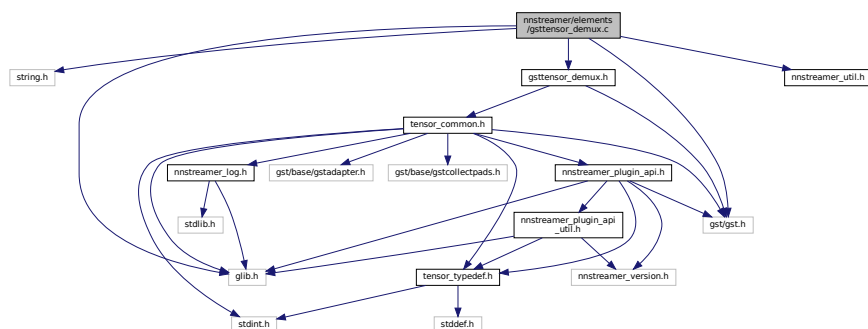
Get Type function required for gst elements.

9.38 nnstreamer/elements/gsttensor_demux.c File Reference

GStreamer plugin to demux tensors (as a filter for other general neural network filters)

```
#include <string.h>
#include <gst/gst.h>
#include <glib.h>
#include <nnstreamer_util.h>
#include "gsttensor_demux.h"
```

Include dependency graph for gsttensor_demux.c:



Macros

- #define `GST_CAT_DEFAULT` `gst_tensor_demux_debug`
- #define `CAPS_STRING_SINK` `GST_TENSORS_CAP_MAKE` ("{ static, flexible }")
Default caps string for sink pad.
- #define `CAPS_STRING_SRC` `GST_TENSOR_CAP_DEFAULT` "," `GST_TENSORS_CAP_MAKE` ("{ static, flexible }")
Default caps string for src pad.
- #define `gst_tensor_demux_parent_class` `parent_class`

Enumerations

- enum { `PROP_0`, `PROP_SILENT`, `PROP_TENSORPICK` }

Functions

- `GST_DEBUG_CATEGORY_STATIC` (`gst_tensor_demux_debug`)
- static `GstFlowReturn` `gst_tensor_demux_chain` (`GstPad *pad`, `GstObject *parent`, `GstBuffer *buf`)
chain function for sink (gst element vmethod)
- static `gboolean` `gst_tensor_demux_event` (`GstPad *pad`, `GstObject *parent`, `GstEvent *event`)
event function for sink (gst element vmethod)
- static `GstStateChangeReturn` `gst_tensor_demux_change_state` (`GstElement *element`, `GstStateChange transition`)
change state (gst element vmethod)
- static void `gst_tensor_demux_set_property` (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
Get property (gst element vmethod)
- static void `gst_tensor_demux_get_property` (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)
Get property (gst element vmethod)
- static void `gst_tensor_demux_dispose` (`GObject *object`)
dispose function for tensor demux (gst element vmethod)
- `G_DEFINE_TYPE` (`GstTensorDemux`, `gst_tensor_demux`, `GST_TYPE_ELEMENT`)
- static void `gst_tensor_demux_class_init` (`GstTensorDemuxClass *klass`)
initialize the tensor_demux's class
- static void `gst_tensor_demux_init` (`GstTensorDemux *tensor_demux`)
initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure
- static void `gst_tensor_demux_remove_src_pads` (`GstTensorDemux *tensor_demux`)
function to remove srcpad list
- static `gboolean` `gst_tensor_demux_parse_caps` (`GstTensorDemux *tensor_demux`, `GstCaps *caps`)
Parse caps and configure tensors info.
- static `gboolean` `gst_tensor_demux_get_tensor_config` (`GstTensorDemux *tensor_demux`, `GstTensorsConfig *config`, `const guint nth`, `const guint total`)
Get tensor config info from configured tensors.
- static `GstTensorPad *` `gst_tensor_demux_get_tensor_pad` (`GstTensorDemux *tensor_demux`, `gboolean *created`, `const guint nth`, `const guint total`)
Checking if the source pad is created and if not, create TensorPad.
- static `GstFlowReturn` `gst_tensor_demux_combine_flows` (`GstTensorDemux *tensor_demux`, `GstTensorPad *pad`, `GstFlowReturn ret`)
Check the status among sources in demux.

Variables

- static GstStaticPadTemplate [src_tmpl](#)
the capabilities of the inputs and outputs. describe the real formats here.
- static GstStaticPadTemplate [sink_tmpl](#)

9.38.1 Detailed Description

GStreamer plugin to demux tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

03 July 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.38.2 Macro Definition Documentation

9.38.2.1 CAPS_STRING_SINK

```
#define CAPS_STRING_SINK GST_TENSORS_CAP_MAKE ("{ static, flexible }")
```

Default caps string for sink pad.

Definition at line 78 of file gsttensor_demux.c.

9.38.2.2 CAPS_STRING_SRC

```
#define CAPS_STRING_SRC GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_MAKE ("{ static, flexible }")
```

Default caps string for src pad.

Definition at line 83 of file gsttensor_demux.c.

9.38.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_demux_debug
```

Definition at line 73 of file gsttensor_demux.c.

9.38.2.4 gst_tensor_demux_parent_class

```
#define gst_tensor_demux_parent_class parent_class
```

Definition at line 119 of file gsttensor_demux.c.

9.38.3 Enumeration Type Documentation

9.38.3.1 anonymous enum

anonymous enum

Enumerator

PROP_0	
PROP_SILENT	
PROP_TENSORPICK	

Definition at line 85 of file gsttensor_demux.c.

9.38.4 Function Documentation

9.38.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorDemux ,
    gst_tensor_demux ,
    GST_TYPE_ELEMENT )
```

9.38.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_demux_debug )
```

SECTION:element-tensor_demux

A Demuxer that demux tensors stream to tensor(s) stream for NN frameworks. The output is always in the format of other/tensor or other/tensors.

```
<refsect2> <title>Example launch line</title> |[ gst-launch-1.0 tensor_mux name=mux ! tensor_demux
name=demux \ filesrc location=testcase01_RGB_100x100.png ! pngdec ! videoscale ! imagefreeze ! video-
convert ! video/x-raw,format=RGB,width=100,height=100,framerate=0/1 ! tensor_converter ! mux.sink_0
\ filesrc location=testcase01_RGB_100x100.png ! pngdec ! videoscale ! imagefreeze ! videoconvert
! video/x-raw,format=RGB,width=100,height=100,framerate=0/1 ! tensor_converter ! mux.sink_1 \ filesrc
location=testcase01_RGB_100x100.png ! pngdec ! videoscale ! imagefreeze ! videoconvert ! video/x-
raw,format=RGB,width=100,height=100,framerate=0/1 ! tensor_converter ! mux.sink_2 \ demux.src_0 ! queue
! filesink location=demux00.log \ demux.src_1 ! queue ! filesink location=demux01.log \ demux.src_2 ! queue !
filesink location=demux02.log ]]
```

```
 |[ gst-launch-1.0 tensor_mux name=mux ! tensor_demux name=demux tensorpick=0,1:2,2+0 \ ... (tensor 0) !
mux.sink_0 \ ... (tensor 1) ! mux.sink_1 \ ... (tensor 2) ! mux.sink_2 \ demux.src_0 ! (tensor 0) ... demux.src_1 !
(tensor 1,2) ... demux.src_2 ! (tensor 2,0) ... ]]
```

</refsect2>

9.38.4.3 gst_tensor_demux_chain()

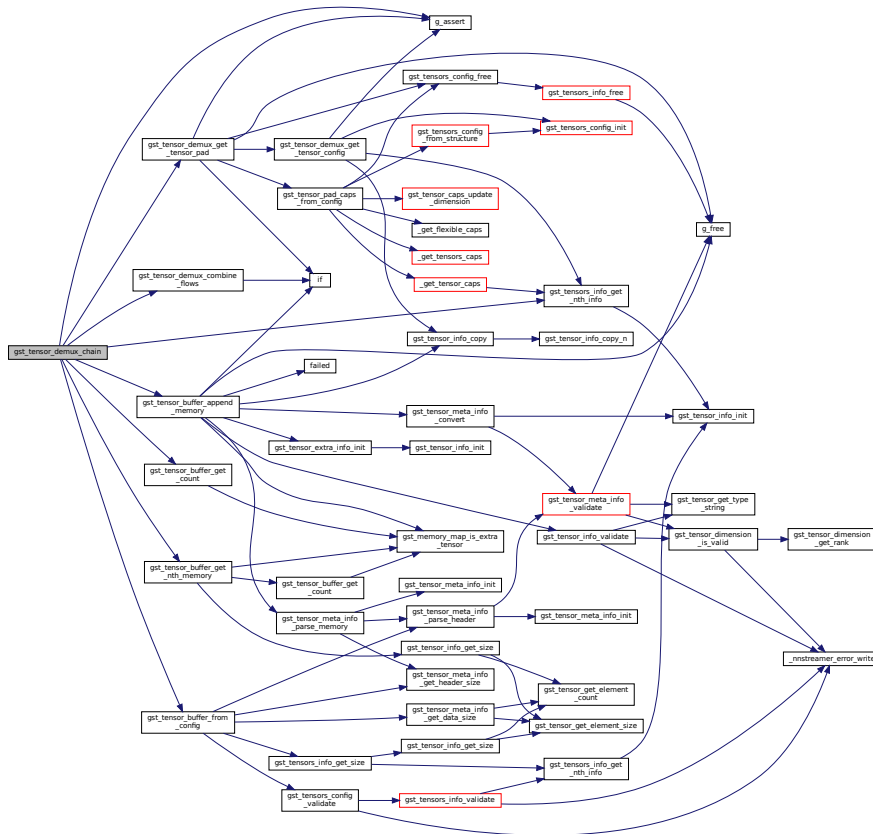
```
static GstFlowReturn gst_tensor_demux_chain (
    GstPad * pad,
    GstObject * parent,
    GstBuffer * buf ) [static]
```

chain function for sink (gst element vmethod)

The number of tensors in the buffer: The number of tensors from caps and gst-buffer should be same when incoming buffer is static tensor. If given buffer is flexible tensor, we cannot get exact number of tensors from config.

Definition at line 476 of file gsttensor_demux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



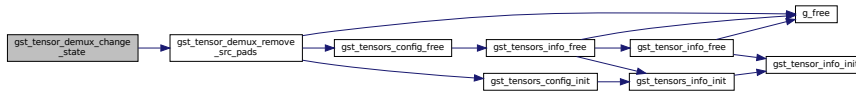
9.38.4.4 gst_tensor_demux_change_state()

```
static GstStateChangeReturn gst_tensor_demux_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

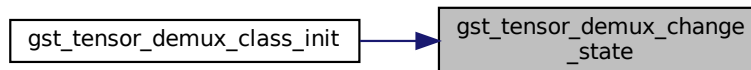
change state (gst element vmethod)

Definition at line 586 of file gsttensor_demux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



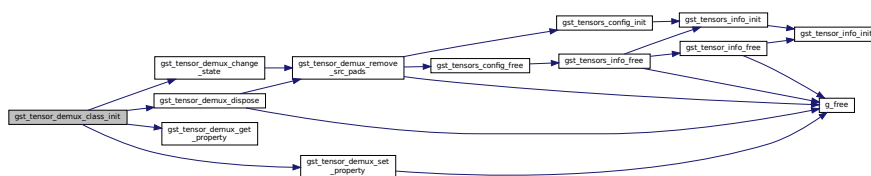
9.38.4.5 gst_tensor_demux_class_init()

```
static void gst_tensor_demux_class_init (
    GstTensorDemuxClass * klass ) [static]
```

initialize the tensor_demux's class

Definition at line 127 of file gsttensor_demux.c.

Here is the call graph for this function:



9.38.4.6 gst_tensor_demux_combine_flows()

```
static GstFlowReturn gst_tensor_demux_combine_flows (
    GstTensorDemux * tensor_demux,
    GstTensorPad * pad,
    GstFlowReturn ret ) [static]
```

Check the status among sources in demux.

Parameters

<i>tensor_demux</i>	TensorDemux Object
<i>TensorPad</i>	Tensorpad
<i>ret</i>	return status of current pad

Returns

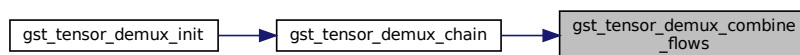
return status after check sources

Definition at line 453 of file gsttensor_demux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



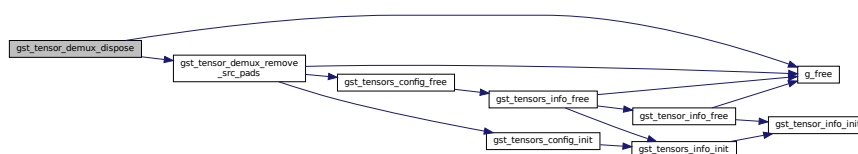
9.38.4.7 gst_tensor_demux_dispose()

```
static void gst_tensor_demux_dispose (
    GObject * object ) [static]
```

dispose function for tensor demux (gst element vmethod)

Definition at line 219 of file gsttensor_demux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.38.4.8 gst_tensor_demux_event()

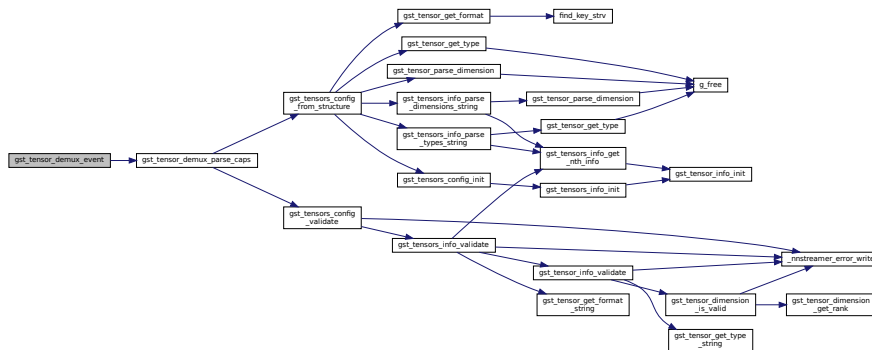
```

static gboolean gst_tensor_demux_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
  
```

event function for sink (gst element vmethod)

Definition at line 252 of file gsttensor_demux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



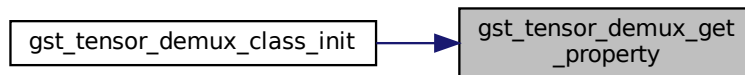
9.38.4.9 `gst_tensor_demux_get_property()`

```
static void gst_tensor_demux_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

Get property (gst element vmethod)

Definition at line 652 of file `gsttensor_demux.c`.

Here is the caller graph for this function:



9.38.4.10 `gst_tensor_demux_get_tensor_config()`

```
static gboolean gst_tensor_demux_get_tensor_config (
    GstTensorDemux * tensor_demux,
    GstTensorsConfig * config,
    const guint nth,
    const guint total ) [static]
```

Get tensor config info from configured tensors.

Parameters

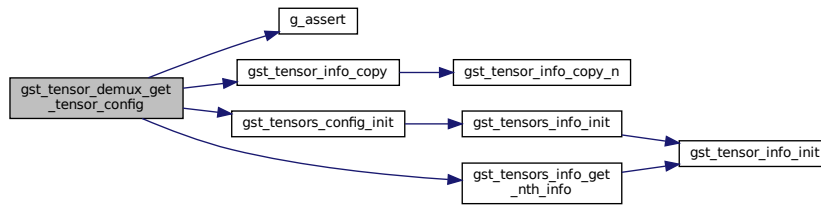
<i>tensor_demux</i>	"this" pointer
<i>config</i>	tensor config to be filled
<i>nth</i>	source ordering
<i>total</i>	number of tensors

Returns

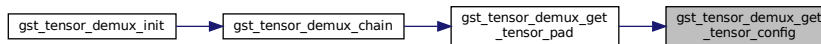
TRUE if successfully configured

Definition at line 290 of file `gsttensor_demux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.38.4.11 gst_tensor_demux_get_tensor_pad()

```

static GstTensorPad* gst_tensor_demux_get_tensor_pad (
    GstTensorDemux * tensor_demux,
    gboolean * created,
    const guint nth,
    const guint total ) [static]
  
```

Checking if the source pad is created and if not, create TensorPad.

Parameters

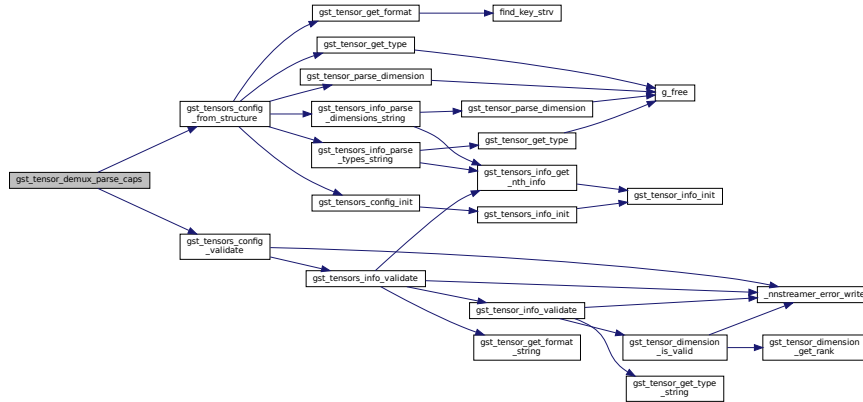
	<i>tensor_demux</i>	TensorDemux Object
out	<i>created</i>	will be updated in this function
	<i>nth</i>	source ordering
	<i>total</i>	number of tensors

Returns

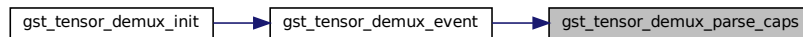
TensorPad if pad is already created, then return created pad. If not return new pad after creation.

Definition at line 349 of file gsttensor_demux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



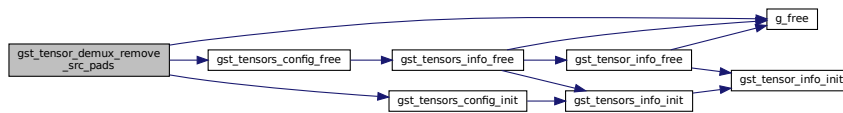
9.38.4.14 gst_tensor_demux_remove_src_pads()

```
static void gst_tensor_demux_remove_src_pads (
    GstTensorDemux * tensor_demux ) [static]
```

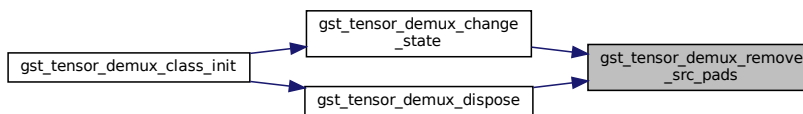
function to remove srcpad list

Definition at line 199 of file `gsttensor_demux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



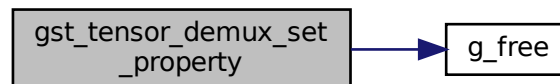
9.38.4.15 `gst_tensor_demux_set_property()`

```
static void gst_tensor_demux_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

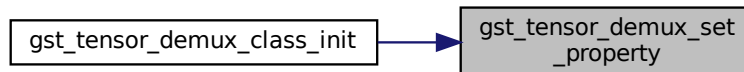
Get property (gst element vmethod)

Definition at line 615 of file `gsttensor_demux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.38.5 Variable Documentation

9.38.5.1 `sink_tmpl`

```
GstStaticPadTemplate sink_tmpl [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
)
```

Definition at line 102 of file `gsttensor_demux.c`.

9.38.5.2 src_tmpl

```
GstStaticPadTemplate src_tmpl [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src_%u",
    GST_PAD_SRC,
    GST_PAD_SOMETIMES,
)
```

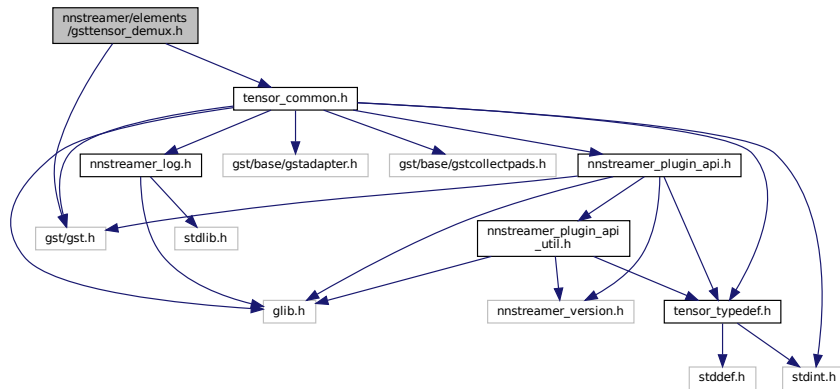
the capabilities of the inputs and outputs. describe the real formats here.

Definition at line 96 of file gsttensor_demux.c.

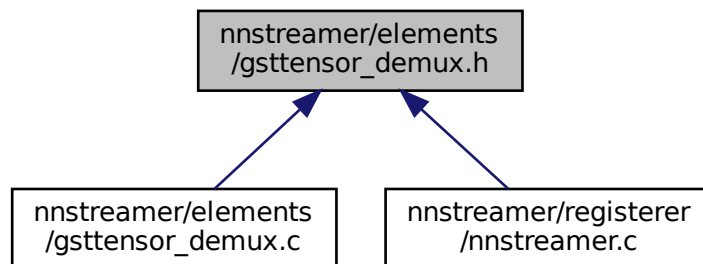
9.39 nnstreamer/elements/gsttensor_demux.h File Reference

GStreamer plugin to demux tensors (as a filter for other general neural network filters)

```
#include <gst/gst.h>
#include <tensor_common.h>
Include dependency graph for gsttensor_demux.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorDemux](#)
Tensor Muxer data structure.
- struct [_GstTensorDemuxClass](#)
GstTensorDeMuxClass inherits GstElementClass.

Macros

- #define [GST_TYPE_TENSOR_DEMUX](#) ([gst_tensor_demux_get_type](#) ())
- #define [GST_TENSOR_DEMUX\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST](#) ((obj), [GST_TYPE_TENSOR_DEMUX](#), [GstTensorDemux](#)))
- #define [GST_TENSOR_DEMUX_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST](#) ((klass), [GST_TYPE_TENSOR_DEMUX](#), [GstTensorDemuxClass](#)))
- #define [GST_TENSOR_DEMUX_GET_CLASS\(obj\)](#) ([G_TYPE_INSTANCE_GET_CLASS](#) ((obj), [GST_TYPE_TENSOR_DEMUX](#), [GstTensorDemuxClass](#)))
- #define [GST_IS_TENSOR_DEMUX\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_DEMUX](#)))
- #define [GST_IS_TENSOR_DEMUX_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_DEMUX](#)))
- #define [GST_TENSOR_DEMUX_CAST\(obj\)](#) (([GstTensorDemux*](#))(obj))

Typedefs

- typedef struct [_GstTensorDemux](#) [GstTensorDemux](#)
- typedef struct [_GstTensorDemuxClass](#) [GstTensorDemuxClass](#)

Functions

- GType [gst_tensor_demux_get_type](#) (void)
Get Type function required for gst elements.

9.39.1 Detailed Description

GStreamer plugin to demux tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

03 July 2018

See also

<https://github.com/nstreamer/nstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.39.2 Macro Definition Documentation

9.39.2.1 GST_IS_TENSOR_DEMUX

```
#define GST_IS_TENSOR_DEMUX(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_DEMUX))
```

Definition at line 40 of file gsttensor_demux.h.

9.39.2.2 GST_IS_TENSOR_DEMUX_CLASS

```
#define GST_IS_TENSOR_DEMUX_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_DEMUX))
```

Definition at line 41 of file gsttensor_demux.h.

9.39.2.3 GST_TENSOR_DEMUX

```
#define GST_TENSOR_DEMUX(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_DEMUX, GstTensorDemux))
```

Definition at line 37 of file gsttensor_demux.h.

9.39.2.4 GST_TENSOR_DEMUX_CAST

```
#define GST_TENSOR_DEMUX_CAST(  
    obj ) ((GstTensorDemux*)(obj))
```

Definition at line 42 of file gsttensor_demux.h.

9.39.2.5 GST_TENSOR_DEMUX_CLASS

```
#define GST_TENSOR_DEMUX_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_DEMUX, GstTensorDemuxClass))
```

Definition at line 38 of file gsttensor_demux.h.

9.39.2.6 GST_TENSOR_DEMUX_GET_CLASS

```
#define GST_TENSOR_DEMUX_GET_CLASS(  
    obj ) (G_TYPE_INSTANCE_GET_CLASS ((obj), GST_TYPE_TENSOR_DEMUX, GstTensorDemuxClass))
```

Definition at line 39 of file gsttensor_demux.h.

9.39.2.7 GST_TYPE_TENSOR_DEMUX

```
#define GST_TYPE_TENSOR_DEMUX (gst_tensor_demux_get_type ())
```

Definition at line 36 of file gsttensor_demux.h.

9.39.3 Typedef Documentation

9.39.3.1 GstTensorDemux

```
typedef struct _GstTensorDemux GstTensorDemux
```

Definition at line 44 of file gsttensor_demux.h.

9.39.3.2 GstTensorDemuxClass

```
typedef struct _GstTensorDemuxClass GstTensorDemuxClass
```

Definition at line 45 of file gsttensor_demux.h.

9.39.4 Function Documentation

9.39.4.1 gst_tensor_demux_get_type()

```
GType gst_tensor_demux_get_type (  
    void )
```

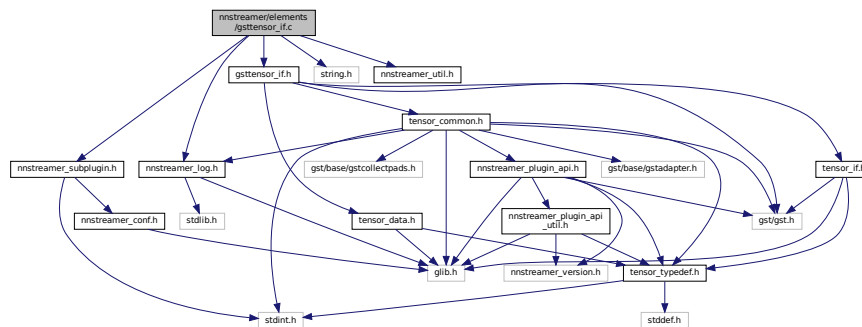
Get Type function required for gst elements.

9.40 nnstreamer/elements/gsttensor_if.c File Reference

GStreamer plugin to control flow based on tensor values.

```
#include <nnstreamer_log.h>
#include <string.h>
#include <nnstreamer_subplugin.h>
#include <nnstreamer_util.h>
#include "gsttensor_if.h"
```

Include dependency graph for `gsttensor_if.c`:



Macros

- `#define` `DBG` (!`tensor_if->`silent)
Macro for debug mode.
- `#define` `GST_CAT_DEFAULT` `gst_tensor_if_debug`
- `#define` `CAPS_STRING` `GST_TENSOR_CAP_DEFAULT` ";" `GST_TENSORS_CAP_DEFAULT`
- `#define` `gst_tensor_if_parent_class` `parent_class`
- `#define` `GST_TYPE_TENSOR_IF_CV` (`gst_tensor_if_cv_get_type` ())
- `#define` `GST_TYPE_TENSOR_IF_OP` (`gst_tensor_if_op_get_type` ())
- `#define` `GST_TYPE_TENSOR_IF_ACT` (`gst_tensor_if_act_get_type` ())
- `#define` `operator_func`(`cv`, `t`, `op`, `sv1`, `sv2`, `ret`)
Macro for operator function.

Enumerations

- enum {
`PROP_0`, `PROP_SILENT`, `PROP_CV`, `PROP_CV_OPTION`,
`PROP_OP`, `PROP_SV`, `PROP_THEN`, `PROP_THEN_OPTION`,
`PROP_ELSE`, `PROP_ELSE_OPTION` }
tensor_if properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_if_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorIf](#), [gst_tensor_if](#), [GST_TYPE_ELEMENT](#))
- static void [gst_tensor_if_set_property](#) ([GObject](#) *object, [guint](#) prop_id, [const GValue](#) *value, [GParamSpec](#) *pspec)
Setter for tensor_if properties.
- static void [gst_tensor_if_get_property](#) ([GObject](#) *object, [guint](#) prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
Getter for tensor_if properties.
- static [GstFlowReturn](#) [gst_tensor_if_chain](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstBuffer](#) *buf)
chain function for sink (gst element vmethod)
- static [gboolean](#) [gst_tensor_if_event](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstEvent](#) *event)
event function for sink (gst element vmethod)
- static void [gst_tensor_if_dispose](#) ([GObject](#) *object)
dispose function for tensor if (gst element vmethod)
- static void [gst_tensor_if_install_properties](#) ([GObjectClass](#) *gobject_class)
Installs all the properties for tensor_if.
- static [GType](#) [gst_tensor_if_cv_get_type](#) ([void](#))
A private function to register GEnumValue array for the 'compared-value' property to a GType and return it.
- static [GType](#) [gst_tensor_if_op_get_type](#) ([void](#))
A private function to register GEnumValue array for the 'operator' property to a GType and return it.
- static [GType](#) [gst_tensor_if_act_get_type](#) ([void](#))
A private function to register GEnumValue array for the 'then' and 'else' properties to a GType and return it.
- static void [gst_tensor_if_class_init](#) ([GstTensorIfClass](#) *klass)
initialize the tensor_if's class (GST Standard)
- static void [gst_tensor_if_init](#) ([GstTensorIf](#) *tensor_if)
initialize the new element (GST Standard) instantiate pads and add them to element set pad callback functions initialize instance structure
- static void [gst_tensor_if_remove_src_pads](#) ([GstTensorIf](#) *tensor_if)
function to remove srcpad list
- static void [gst_tensor_if_set_property_glist](#) ([const GValue](#) *value, [GList](#) **prop_list, [const gchar](#) *delimiters)
Convert GValue to GList according to delimiters.
- static void [gst_tensor_if_set_property_cv_option](#) ([const GValue](#) *value, [GList](#) **prop_list)
Convert GValue to GList for cv option.
- static void [gst_tensor_if_set_property_supplied_value](#) ([const GValue](#) *value, [tensor_if_sv_s](#) *sv, [const gchar](#) *delimiters)
Convert GValue to GList according to delimiters.
- static void [gst_tensor_if_configure_custom_prop](#) ([GstTensorIf](#) *self)
Set custom compared value property.
- static void [gst_tensor_if_property_to_string](#) ([GValue](#) *value, [GList](#) *prop_list, [guint](#) prop_id)
Convert GList to GValue.
- static void [gst_tensor_if_get_property_supplied_value](#) ([GValue](#) *value, [tensor_if_sv_s](#) *sv)
Convert GValue to supplied value according to delimiters.
- static [gboolean](#) [gst_tensor_if_parse_caps](#) ([GstTensorIf](#) *tensor_if, [GstCaps](#) *caps)
Parse caps and configure tensors info.
- static [GstTensorPad](#) * [gst_tensor_if_get_tensor_pad](#) ([GstTensorIf](#) *tensor_if, [GstTensorsConfig](#) *config, [gboolean](#) *created, [guint](#) nth)
Checking if the source pad is created and if not, create TensorPad.
- static [GstFlowReturn](#) [gst_tensor_if_combine_flows](#) ([GstTensorIf](#) *tensor_if, [GstTensorPad](#) *pad, [GstFlow](#)↔
Return ret)
Check the status among sources in if.
- [switch](#) (cv->type)

- static gboolean [gst_tensor_if_get_tensor_average](#) ([GstTensorIf](#) *tensor_if, [GstBuffer](#) *buf, [tensor_data_s](#) *cv, guint nth)
Calculate average value of the nth tensor.
- static gboolean [gst_tensor_if_calculate_cv](#) ([GstTensorIf](#) *tensor_if, [GstBuffer](#) *buf, [tensor_data_s](#) *cv)
Calculate compared value.
- int [nnstreamer_if_custom_register](#) (const gchar *name, [tensor_if_custom](#) func, void *data)
Registers a callback for tensor_if custom condition.
- int [nnstreamer_if_custom_unregister](#) (const gchar *name)
Unregisters a callback for tensor_if custom condition.
- static gboolean [gst_tensor_if_check_condition](#) ([GstTensorIf](#) *tensor_if, [GstBuffer](#) *buf, gboolean *result)
Determining whether a given condition is true or false.

Variables

- static [GstStaticPadTemplate](#) [sink_factory](#)
The capabilities of the inputs.
- static [GstStaticPadTemplate](#) [src_factory](#)
The capabilities of the outputs.
- case [TIFOP_NE](#)
`t == sv1_##t) ? TRUE : FALSE; break; \`
- case [tensor_data_s](#) * cv
- case [tensor_data_s](#) gboolean * result
- [tensor_data_s](#) svtc_1
- [tensor_data_s](#) svtc_2
- [svtc_1](#) type = tensor_if->sv->type
- [svtc_1](#) data = tensor_if->sv->data[0]
- return [TRUE](#)

9.40.1 Detailed Description

GStreamer plugin to control flow based on tensor values.

GStreamer/NNStreamer Tensor-IF Copyright (C) 2020 MyungJoo Ham myungjoo.ham@samsung.com

Date

08 April 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.40.2 Macro Definition Documentation

9.40.2.1 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT " ; " GST_TENSORS_CAP_DEFAULT
```

Definition at line 105 of file gsttensor_if.c.

9.40.2.2 DBG

```
#define DBG (!tensor_if->silent)
```

Macro for debug mode.

SECTION:element-tensor_if

A filter that controls its src-pad based on the values (other/tensor(s)) of its sink-pad. For example, you may skip frames if there is no object detected with high confidence.

The format of statement with tensor-if is: if (Compared_Value OPERATOR Supplied_Value(s)) then THEN else ELSE Compared_Value and Supplied_Value are the operands. Compared_Value is a value from input tensor(s). Supplied_Value is a value from tensor-if properties.

If the given if-condition is simple enough (e.g., if a specific element is between a given range in a tensor frame), it can be expressed as: <refsect2> <title>Example launch line with simple if condition</title> gst-launch ... (some tensor stream) ! tensor_if name=tif compared-value=A_VALUE compared-value-option=3:4:2:5,0 operator=RANGE↔_INCLUSIVE supplied-value=10,100 then=PASSTHROUGH else=TENSORPICK else-option=1 tif.src_0 ! queue ! (tensor(s) stream for TRUE action) ... tif.src_1 ! queue ! (tensor(s) stream for FALSE action) ... </refsect2>

However, if the if-condition is complex and cannot be expressed with tensor-if expressions, you may create a corresponding custom filter with tensor-filter, whose output is other/tensors with an additional tensor that is "1:1:1:1, uint8", which is 1 (true) or 0 (false) as the first tensor of other/tensors and the input tensor/tensors. Then, you can create a pipeline as follows: <refsect2> <title>Example launch line with complex if condition</title> gst-launch ... (some tensor stream) ! tensor_filter framework=custom name=your_code.so ! tensor_if compared-value=A_VALUE compared-value-option=0:0:0:0,0 # 1st tensor's [0][0][0][0]. operator=EQ supplied-value=1 then=PASSTHROUGH # or whatsoever you want else=SKIP # or whatsoever you want ! tensor_demux name=d d.src_0 ! queue ! fakesink # throw away the 1/0 value. d.src_1 ! queue ! do whatever you want here... ... </refsect2>

Definition at line 82 of file gsttensor_if.c.

9.40.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_if_debug
```

Definition at line 103 of file gsttensor_if.c.

9.40.2.4 `gst_tensor_if_parent_class`

```
#define gst_tensor_if_parent_class parent_class
```

Definition at line 122 of file `gsttensor_if.c`.

9.40.2.5 `GST_TYPE_TENSOR_IF_ACT`

```
#define GST_TYPE_TENSOR_IF_ACT (gst_tensor_if_act_get_type ())
```

Definition at line 195 of file `gsttensor_if.c`.

9.40.2.6 `GST_TYPE_TENSOR_IF_CV`

```
#define GST_TYPE_TENSOR_IF_CV (gst_tensor_if_cv_get_type ())
```

Definition at line 139 of file `gsttensor_if.c`.

9.40.2.7 `GST_TYPE_TENSOR_IF_OP`

```
#define GST_TYPE_TENSOR_IF_OP (gst_tensor_if_op_get_type ())
```

Definition at line 163 of file `gsttensor_if.c`.

9.40.2.8 `operator_func`

```
#define operator_func(  
    cv,  
    t,  
    op,  
    sv1,  
    sv2,  
    ret )
```

Value:

```
do { \  
  switch (op) { \  
    case TIFOP_EQ: ret = (cv._
```

Macro for operator function.

Definition at line 813 of file `gsttensor_if.c`.

9.40.3 Enumeration Type Documentation

9.40.3.1 anonymous enum

anonymous enum

tensor_if properties

Enumerator

PROP_0	
PROP_SILENT	
PROP_CV	Compared Value, operand 1 (from input tensor(s))
PROP_CV_OPTION	Compared Value Option
PROP_OP	Operator
PROP_SV	Supplied Value, operand 2 (from the properties)
PROP_THEN	Action if it is TRUE
PROP_THEN_OPTION	Option for TRUE Action
PROP_ELSE	Action if it is FALSE
PROP_ELSE_OPTION	Option for FALSE Action

Definition at line 88 of file gsttensor_if.c.

9.40.4 Function Documentation

9.40.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorIf ,
    gst_tensor_if ,
    GST_TYPE_ELEMENT )
```

9.40.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_if_debug )
```

9.40.4.3 gst_tensor_if_act_get_type()

```
static GType gst_tensor_if_act_get_type (
    void ) [static]
```

A private function to register GEnumValue array for the 'then' and 'else' properties to a GType and return it.

Definition at line 201 of file gsttensor_if.c.

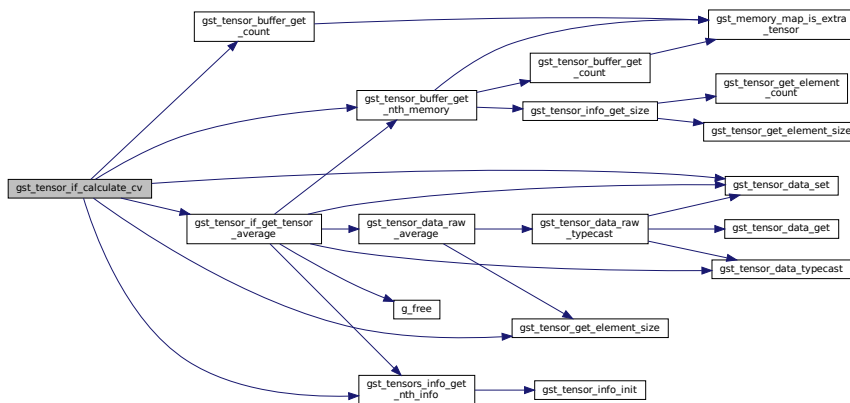
9.40.4.4 `gst_tensor_if_calculate_cv()`

```
static gboolean gst_tensor_if_calculate_cv (
    GstTensorIf * tensor_if,
    GstBuffer * buf,
    tensor_data_s * cv ) [static]
```

Calculate compared value.

Definition at line 939 of file `gsttensor_if.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.40.4.5 `gst_tensor_if_chain()`

```
static GstFlowReturn gst_tensor_if_chain (
    GstPad * pad,
    GstObject * parent,
    GstBuffer * buf ) [static]
```

chain function for sink (gst element vmethod)

Definition at line 1125 of file `gsttensor_if.c`.

Parameters

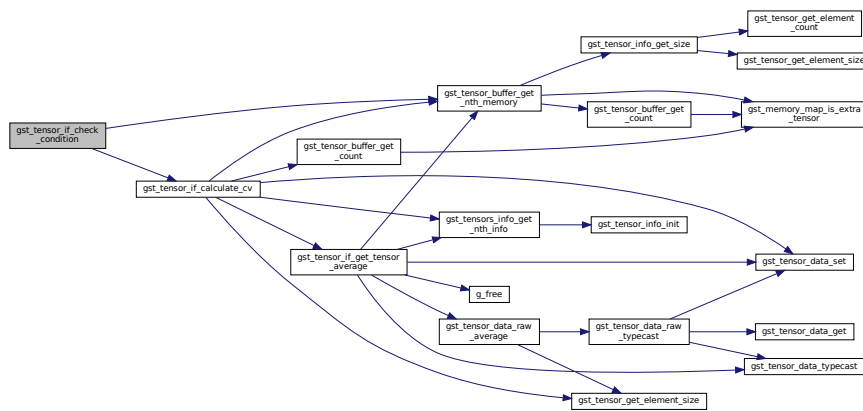
<i>tensor</i> ↔	TensorIf Object
<i>buf</i>	gstbuffer from sink pad

Returns

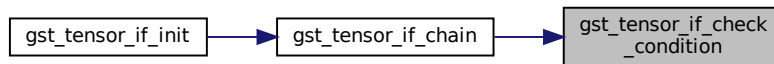
return TRUE if no error

Definition at line 1070 of file gsttensor_if.c.

Here is the call graph for this function:



Here is the caller graph for this function:



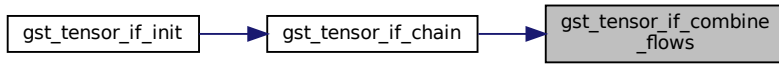
9.40.4.7 gst_tensor_if_class_init()

```
static void gst_tensor_if_class_init (
    GstTensorIfClass * klass ) [static]
```

initialize the tensor_if's class (GST Standard)

Definition at line 222 of file gsttensor_if.c.

Here is the caller graph for this function:



9.40.4.9 gst_tensor_if_configure_custom_prop()

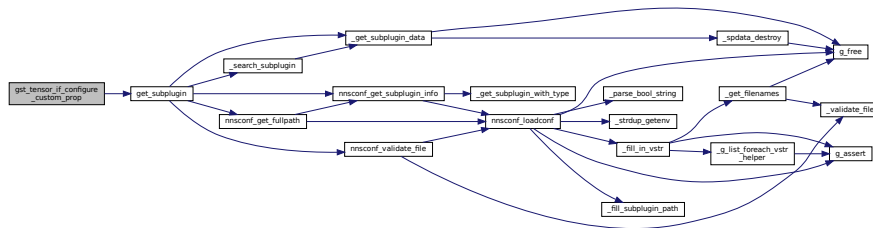
```

static void gst_tensor_if_configure_custom_prop (
    GstTensorIf * self ) [static]
  
```

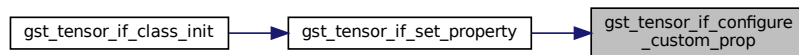
Set custom compared value property.

Definition at line 435 of file gsttensor_if.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.40.4.10 gst_tensor_if_cv_get_type()

```

static GType gst_tensor_if_cv_get_type (
    void ) [static]
  
```

A private function to register GEnumValue array for the 'compared-value' property to a GType and return it.

Definition at line 145 of file gsttensor_if.c.

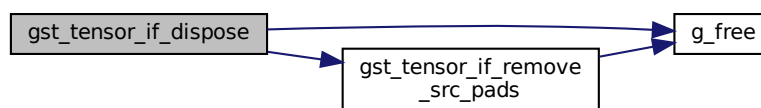
9.40.4.11 `gst_tensor_if_dispose()`

```
static void gst_tensor_if_dispose (
    GObject * object ) [static]
```

dispose function for tensor if (gst element vmethod)

Definition at line 307 of file `gsttensor_if.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.40.4.12 `gst_tensor_if_event()`

```
static gboolean gst_tensor_if_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
```

event function for sink (gst element vmethod)

Definition at line 695 of file `gsttensor_if.c`.

Here is the caller graph for this function:



9.40.4.14 `gst_tensor_if_get_property_supplied_value()`

```

static void gst_tensor_if_get_property_supplied_value (
    GValue * value,
    tensor_if_sv_s * sv ) [static]
  
```

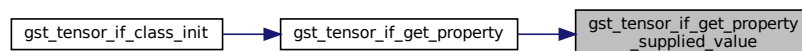
Convert GValue to supplied value according to delimiters.

Definition at line 547 of file `gsttensor_if.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



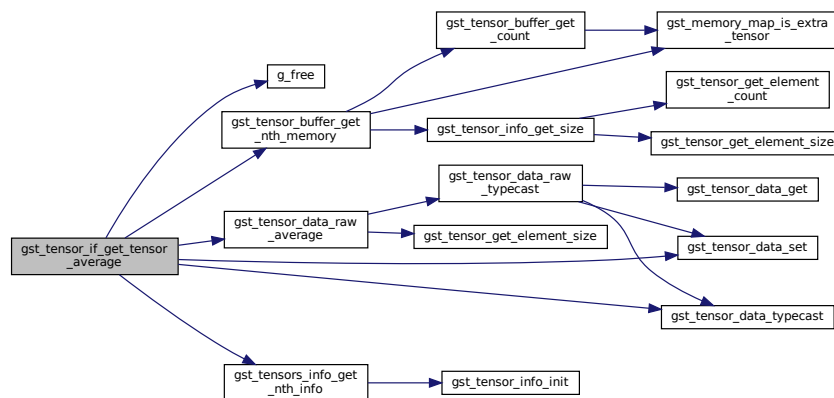
9.40.4.15 `gst_tensor_if_get_tensor_average()`

```
static gboolean gst_tensor_if_get_tensor_average (
    GstTensorIf * tensor_if,
    GstBuffer * buf,
    tensor_data_s * cv,
    guint nth ) [static]
```

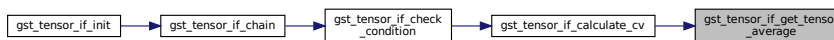
Calculate average value of the nth tensor.

Definition at line 904 of file `gsttensor_if.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

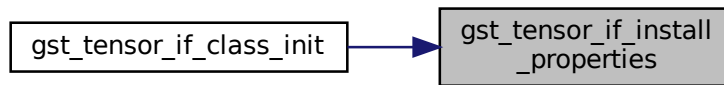


9.40.4.16 `gst_tensor_if_get_tensor_pad()`

```
static GstTensorPad* gst_tensor_if_get_tensor_pad (
    GstTensorIf * tensor_if,
    GstTensorsConfig * config,
    gboolean * created,
    guint nth ) [static]
```

Checking if the source pad is created and if not, create TensorPad.

Here is the caller graph for this function:



9.40.4.19 `gst_tensor_if_op_get_type()`

```
static GType gst_tensor_if_op_get_type (
    void ) [static]
```

A private function to register GEnumValue array for the 'operator' property to a GType and return it.

Definition at line 169 of file `gsttensor_if.c`.

9.40.4.20 `gst_tensor_if_parse_caps()`

```
static gboolean gst_tensor_if_parse_caps (
    GstTensorIf * tensor_if,
    GstCaps * caps ) [static]
```

Parse caps and configure tensors info.

Parameters

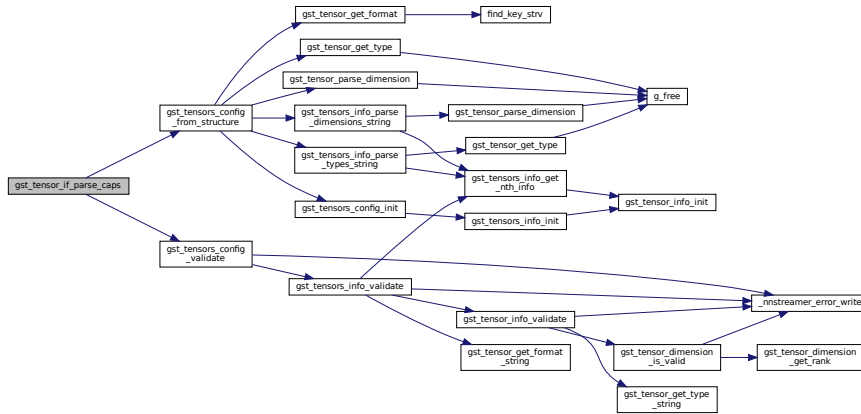
<i>tensor_if</i>	GstTensorIf object
<i>caps</i>	incoming capability

Returns

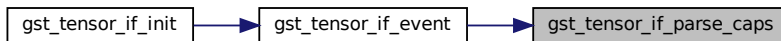
TRUE/FALSE (if successfully configured, return TRUE)

Definition at line 679 of file `gsttensor_if.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.40.4.21 gst_tensor_if_property_to_string()

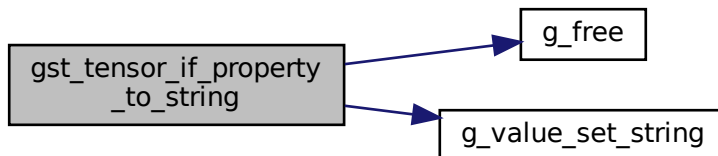
```

static void gst_tensor_if_property_to_string (
    GValue * value,
    GList * prop_list,
    guint prop_id ) [static]
    
```

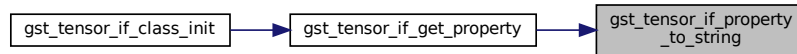
Convert GList to GValue.

Definition at line 503 of file gsttensor_if.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.40.4.22 `gst_tensor_if_remove_src_pads()`

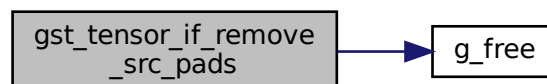
```

static void gst_tensor_if_remove_src_pads (
    GstTensorIf * tensor_if ) [static]
  
```

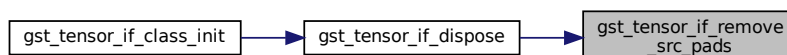
function to remove srcpad list

Definition at line 290 of file gsttensor_if.c.

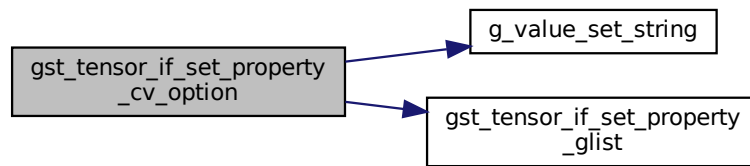
Here is the call graph for this function:



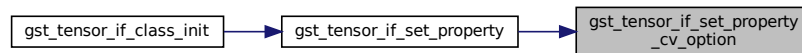
Here is the caller graph for this function:



Here is the call graph for this function:



Here is the caller graph for this function:



9.40.4.25 `gst_tensor_if_set_property_glist()`

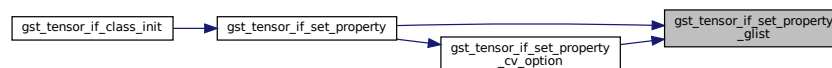
```

static void gst_tensor_if_set_property_glist (
    const GValue * value,
    GList ** prop_list,
    const gchar * delimiters ) [static]
  
```

Convert GValue to GList according to delimiters.

Definition at line 328 of file `gsttensor_if.c`.

Here is the caller graph for this function:



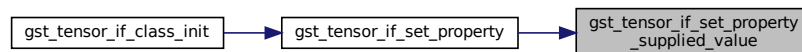
9.40.4.26 `gst_tensor_if_set_property_supplied_value()`

```
static void gst_tensor_if_set_property_supplied_value (
    const GValue * value,
    tensor_if_sv_s * sv,
    const gchar * delimiters ) [static]
```

Convert GValue to GList according to delimiters.

Definition at line 400 of file `gsttensor_if.c`.

Here is the caller graph for this function:



9.40.4.27 `nnstreamer_if_custom_register()`

```
int nnstreamer_if_custom_register (
    const gchar * name,
    tensor_if_custom func,
    void * data )
```

Registers a callback for `tensor_if` custom condition.

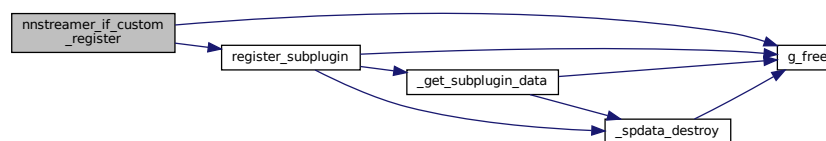
Register the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 1023 of file `gsttensor_if.c`.

Here is the call graph for this function:



9.40.4.28 nnstreamer_if_custom_unregister()

```
int nnstreamer_if_custom_unregister (
    const gchar * name )
```

Unregisters a callback for tensor_if custom condition.

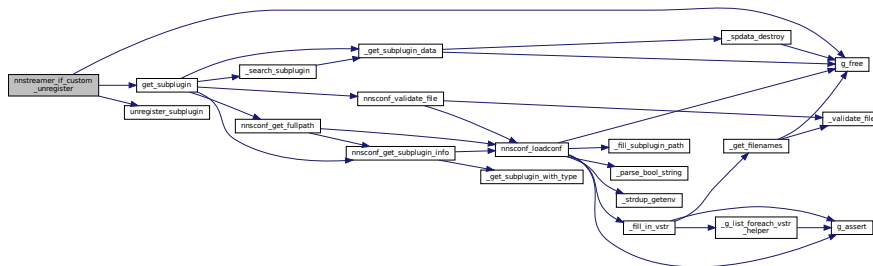
Unregister the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 1049 of file gsttensor_if.c.

Here is the call graph for this function:

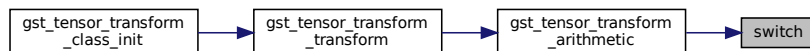


9.40.4.29 switch()

```
switch (
    cv-> type )
```

Definition at line 851 of file gsttensor_if.c.

Here is the caller graph for this function:



9.40.5 Variable Documentation

9.40.5.1 cv

```
case tensor_data_s* cv
```

Definition at line 838 of file gsttensor_if.c.

9.40.5.2 data

```
svtc_2 data = tensor_if->sv->data[0]
```

Definition at line 844 of file gsttensor_if.c.

9.40.5.3 result

```
* result
```

Initial value:

```
{  
    gboolean ret = FALSE
```

Definition at line 839 of file gsttensor_if.c.

9.40.5.4 sink_factory

```
GstStaticPadTemplate sink_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the inputs.

Definition at line 109 of file gsttensor_if.c.

9.40.5.5 src_factory

```
GstStaticPadTemplate src_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_SOMETIMES,  
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the outputs.

Definition at line 117 of file gsttensor_if.c.

9.40.5.6 svtc_1

```
gst_tensor_data_typecast & svtc_1
```

Definition at line 841 of file gsttensor_if.c.

9.40.5.7 svtc_2

```
gst_tensor_data_typecast & svtc_2
```

Definition at line 841 of file gsttensor_if.c.

9.40.5.8 TIFOP_NE

```
case TIFOP_NE
```

```
t == sv1._##t) ? TRUE : FALSE; break; \
```

Definition at line 837 of file gsttensor_if.c.

9.40.5.9 TRUE

```
return TRUE
```

Definition at line 897 of file gsttensor_if.c.

9.40.5.10 type

```
svtc_2 type = tensor_if->sv->type
```

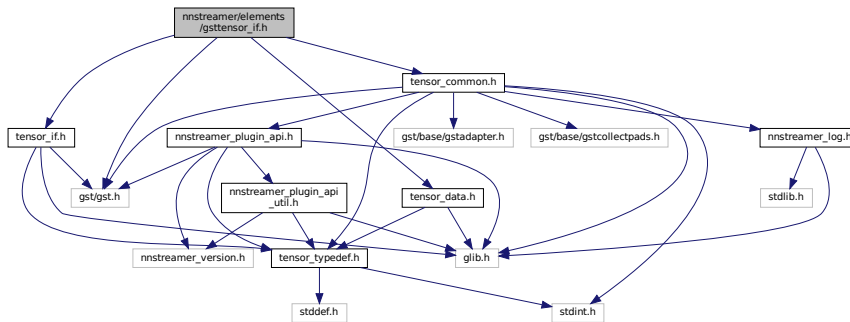
Definition at line 843 of file gsttensor_if.c.

9.41 nnstreamer/elements/gsttensor_if.h File Reference

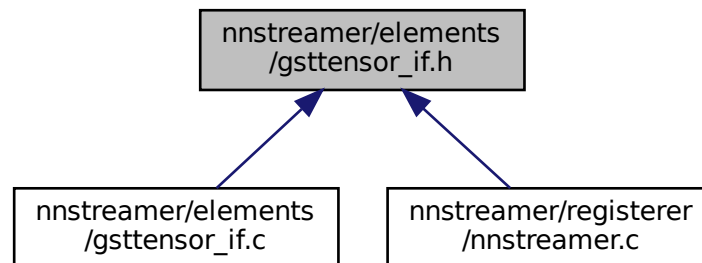
GStreamer plugin to control flow based on tensor values.

```
#include <gst/gst.h>
#include <tensor_common.h>
#include <tensor_data.h>
#include <tensor_if.h>
```

Include dependency graph for gsttensor_if.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [tensor_if_sv_s](#)
Internal data structure for supplied value.
- struct [custom_cb_s](#)
- struct [_GstTensorIf](#)
Tensor If data structure.
- struct [_GstTensorIfClass](#)
GstTensorIfClass inherits GstElementClass.

Macros

- #define `GST_TYPE_TENSOR_IF` (`gst_tensor_if_get_type` ())
- #define `GST_TENSOR_IF`(obj) (`G_TYPE_CHECK_INSTANCE_CAST` ((obj), `GST_TYPE_TENSOR_IF`, `GstTensorIf`))
- #define `GST_TENSOR_IF_CLASS`(klass) (`G_TYPE_CHECK_CLASS_CAST` ((klass), `GST_TYPE_TENSOR_IF`, `GstTensorIfClass`))
- #define `GST_TENSOR_IF_GET_CLASS`(obj) (`G_TYPE_INSTANCE_GET_CLASS` ((obj), `GST_TYPE_TENSOR_IF`, `GstTensorIfClass`))
- #define `GST_IS_TENSOR_IF`(obj) (`G_TYPE_CHECK_INSTANCE_TYPE`((obj),`GST_TYPE_TENSOR_IF`))
- #define `GST_IS_TENSOR_IF_CLASS`(klass) (`G_TYPE_CHECK_CLASS_TYPE`((klass),`GST_TYPE_TENSOR_IF`))
- #define `GST_TENSOR_IF_CAST`(obj) ((`GstTensorIf*`)(obj))

Typedefs

- typedef struct `_GstTensorIf` `GstTensorIf`
- typedef struct `_GstTensorIfClass` `GstTensorIfClass`

Enumerations

- enum `tensor_if_compared_value` {
`TIFCV_A_VALUE` = 0, `TIFCV_TENSOR_TOTAL_VALUE` = 1, `TIFCV_ALL_TENSORS_TOTAL_VALUE` = 2,
`TIFCV_TENSOR_AVERAGE_VALUE` = 3,
`TIFCV_ALL_TENSORS_AVERAGE_VALUE` = 4, `TIFCV_CUSTOM` = 5, `TIFCV_END` }
Compared_Value.
- enum `tensor_if_operator` {
`TIFOP_EQ` = 0, `TIFOP_NE`, `TIFOP_GT`, `TIFOP_GE`,
`TIFOP_LT`, `TIFOP_LE`, `TIFOP_RANGE_INCLUSIVE`, `TIFOP_RANGE_EXCLUSIVE`,
`TIFOP_NOT_IN_RANGE_INCLUSIVE`, `TIFOP_NOT_IN_RANGE_EXCLUSIVE`, `TIFOP_END` }
OPERAND.
- enum `tensor_if_behavior` {
`TIFB_PASSTHROUGH` = 0, `TIFB_SKIP`, `TIFB_FILL_ZERO`, `TIFB_FILL_VALUES`,
`TIFB_FILL_WITH_FILE`, `TIFB_FILL_WITH_FILE_RPT`, `TIFB_REPEAT_PREVIOUS_FRAME`, `TIFB_TENSORPICK`,
`TIFB_END` }
Behaviors that may fit in THEN and ELSE.
- enum `tensor_if_srcpads` { `TIFSP_THEN_PAD` = 0, `TIFSP_ELSE_PAD` }
Which src pad.

Functions

- GType `gst_tensor_if_get_type` (void)
Get Type function required for gst elements.

9.41.1 Detailed Description

GStreamer plugin to control flow based on tensor values.

GStreamer/NNStreamer Tensor-IF Copyright (C) 2020 MyungJoo Ham myungjoo.ham@samsung.com

Date

08 April 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

Todo Add "event/signal" to reload FILL_WITH_FILE* file??? (TBD)

The output dimension is SAME with input dimension.

9.41.2 Macro Definition Documentation

9.41.2.1 GST_IS_TENSOR_IF

```
#define GST_IS_TENSOR_IF(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_IF))
```

Definition at line 33 of file gstreamer_if.h.

9.41.2.2 GST_IS_TENSOR_IF_CLASS

```
#define GST_IS_TENSOR_IF_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_IF))
```

Definition at line 34 of file gstreamer_if.h.

9.41.2.3 GST_TENSOR_IF

```
#define GST_TENSOR_IF(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_TENSOR_IF, GstTensorIf))
```

Definition at line 30 of file gsttensor_if.h.

9.41.2.4 GST_TENSOR_IF_CAST

```
#define GST_TENSOR_IF_CAST(  
    obj ) ((GstTensorIf*)(obj))
```

Definition at line 35 of file gsttensor_if.h.

9.41.2.5 GST_TENSOR_IF_CLASS

```
#define GST_TENSOR_IF_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_TENSOR_IF, GstTensorIfClass))
```

Definition at line 31 of file gsttensor_if.h.

9.41.2.6 GST_TENSOR_IF_GET_CLASS

```
#define GST_TENSOR_IF_GET_CLASS(  
    obj ) (G_TYPE_INSTANCE_GET_CLASS ((obj), GST_TYPE_TENSOR_IF, GstTensorIfClass))
```

Definition at line 32 of file gsttensor_if.h.

9.41.2.7 GST_TYPE_TENSOR_IF

```
#define GST_TYPE_TENSOR_IF (gst_tensor_if_get_type ())
```

Definition at line 29 of file gsttensor_if.h.

9.41.3 Typedef Documentation

9.41.3.1 GstTensorIf

```
typedef struct _GstTensorIf GstTensorIf
```

Definition at line 36 of file gstreamer_if.h.

9.41.3.2 GstTensorIfClass

```
typedef struct _GstTensorIfClass GstTensorIfClass
```

Definition at line 37 of file gstreamer_if.h.

9.41.4 Enumeration Type Documentation

9.41.4.1 tensor_if_behavior

```
enum tensor_if_behavior
```

Behaviors that may fit in THEN and ELSE.

FILL_WITH_FILE, FILL_WITH_FILE_RPT, and REPEAT_PREVIOUS_FRAME caches an output frame and thus, may consume additional memory and incur an additional memcopy.

Enumerator

TIFB_PASSTHROUGH	The input frame becomes the output frame
TIFB_SKIP	Do not generate output frame (frame skip)
TIFB_FILL_ZERO	Fill output frame with zeros
TIFB_FILL_VALUES	Fill output frame with a user given value
TIFB_FILL_WITH_FILE	Fill output frame with a user given file (a raw data of tensor/tensors) If the filesize is smaller, the reset is filled with 0
TIFB_FILL_WITH_FILE_RPT	Fill output frame with a user given file (a raw data of tensor/tensors) If the filesize is smally, the file is repeatedly used
TIFB_REPEAT_PREVIOUS_FRAME	Resend the previous output frame. If this is the first, send ZERO values.
TIFB_TENSORPICK	Choose nth tensor (or tensors) among tensors
TIFB_END	

Definition at line 79 of file gstreamer_if.h.

9.41.4.2 tensor_if_compared_value

```
enum tensor_if_compared_value
```

Compared_Value.

Enumerator

TIFCV_A_VALUE	Decide based on a single scalar value of tensors
TIFCV_TENSOR_TOTAL_VALUE	Decide based on a total (sum) value of a specific tensor
TIFCV_ALL_TENSORS_TOTAL_VALUE	Decide based on a total (sum) value of of tensors or a specific tensor
TIFCV_TENSOR_AVERAGE_VALUE	Decide based on a average value of a specific tensor
TIFCV_ALL_TENSORS_AVERAGE_VALUE	Decide based on a average value of tensors or a specific tensor
TIFCV_CUSTOM	Decide based on a user defined condition
TIFCV_END	

Definition at line 42 of file gsttensor_if.h.

9.41.4.3 tensor_if_operator

```
enum tensor_if_operator
```

OPERAND.

Enumerator

TIFOP_EQ	==
TIFOP_NE	!=
TIFOP_GT	
TIFOP_GE	>=
TIFOP_LT	<
TIFOP_LE	<=
TIFOP_RANGE_INCLUSIVE	in [min, max]
TIFOP_RANGE_EXCLUSIVE	in (min, max)
TIFOP_NOT_IN_RANGE_INCLUSIVE	not in [min, max]
TIFOP_NOT_IN_RANGE_EXCLUSIVE	not in (min, max)
TIFOP_END	

Definition at line 60 of file gsttensor_if.h.

9.41.4.4 tensor_if_srcpads

```
enum tensor_if_srcpads
```

Which src pad.

Enumerator

TIFSP_THEN_PAD	
TIFSP_ELSE_PAD	

Definition at line 96 of file gsttensor_if.h.

9.41.5 Function Documentation

9.41.5.1 gst_tensor_if_get_type()

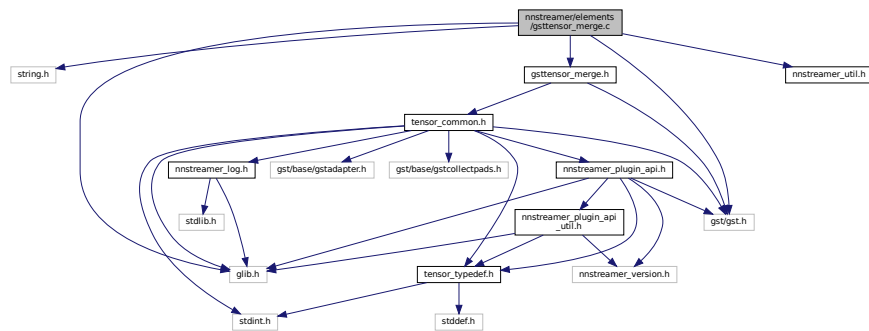
```
GType gst_tensor_if_get_type (
    void )
```

Get Type function required for gst elements.

9.42 nnstreamer/elements/gsttensor_merge.c File Reference

GStreamer plugin to merge tensors (as a filter for other general neural network filters)

```
#include <string.h>
#include <gst/gst.h>
#include <glib.h>
#include <nnstreamer_util.h>
#include "gsttensor_merge.h"
Include dependency graph for gsttensor_merge.c:
```



Macros

- #define `GST_CAT_DEFAULT` `gst_tensor_merge_debug`
- #define `DBG` (`!tensor_merge->silent`)
Macro for debug mode.
- #define `CAPS_STRING` `GST_TENSOR_CAP_DEFAULT` `","` `GST_TENSORS_CAP_WITH_NUM` `("1")`
Template caps string.
- #define `gst_tensor_merge_parent_class` `parent_class`

Enumerations

- enum {
[PROP_0](#), [PROP_MODE](#), [PROP_OPTION](#), [PROP_SYNC_MODE](#),
[PROP_SYNC_OPTION](#), [PROP_SILENT](#) }

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_merge_debug](#))
- static gboolean [gst_tensor_merge_src_event](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstEvent](#) *event)
src event vmethod
- static [GstPad](#) * [gst_tensor_merge_request_new_pad](#) ([GstElement](#) *element, [GstPadTemplate](#) *templ, const [gchar](#) *req_name, const [GstCaps](#) *caps)
making new request pad (gst element vmethod)
- static [GstStateChangeReturn](#) [gst_tensor_merge_change_state](#) ([GstElement](#) *element, [GstStateChange](#) transition)
change state (gst element vmethod)
- static gboolean [gst_tensor_merge_sink_event](#) ([GstCollectPads](#) *pads, [GstCollectData](#) *data, [GstEvent](#) *event, [GstTensorMerge](#) *tensor_merge)
sink event vmethod
- static [GstFlowReturn](#) [gst_tensor_merge_collected](#) ([GstCollectPads](#) *pads, [GstTensorMerge](#) *tensor_merge)
Gst Collect Pads Function which is called once collect pads done.
- static void [gst_tensor_merge_set_property](#) ([GObject](#) *object, guint prop_id, const [GValue](#) *value, [GParamSpec](#) *pspec)
Get property (gst element vmethod)
- static void [gst_tensor_merge_get_property](#) ([GObject](#) *object, guint prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
Get property (gst element vmethod)
- static void [gst_tensor_merge_finalize](#) ([GObject](#) *object)
finalize vmethod
- [G_DEFINE_TYPE](#) ([GstTensorMerge](#), [gst_tensor_merge](#), [GST_TYPE_ELEMENT](#))
- static void [gst_tensor_merge_class_init](#) ([GstTensorMergeClass](#) *klass)
initialize the tensor_merge's class
- static void [gst_tensor_merge_init](#) ([GstTensorMerge](#) *tensor_merge)
initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure
- static [tensor_merge_mode](#) [gst_tensor_merge_get_mode](#) (const [gchar](#) *str)
Get the corresponding mode from the string value.
- static gboolean [gst_tensor_merge_get_merged_config](#) ([GstTensorMerge](#) *tensor_merge, const [GstTensorsConfig](#) *in_config, [GstTensorsConfig](#) *out_config)
Generate out TensorsConfig with in TensorsConfig.
- static gboolean [gst_tensor_merge_collect_buffer](#) ([GstTensorMerge](#) *tensor_merge, [GstBuffer](#) *tensors_buf, gboolean *is_eos)
Looping to generate output buffer for srcpad.
- static [GstFlowReturn](#) [gst_tensor_merge_generate_mem](#) ([GstTensorMerge](#) *tensor_merge, [GstBuffer](#) *tensors_buf, [GstBuffer](#) *tensor_buf)
Generate Output GstMemory.
- static gboolean [gst_tensor_merge_set_src_caps](#) ([GstTensorMerge](#) *tensor_merge)
Set src pad caps if src pad is not negotiated.
- static void [gst_tensor_merge_send_segment_event](#) ([GstTensorMerge](#) *tensor_merge, [GstClockTime](#) pts, [GstClockTime](#) dts)
Send segment event if necessary.
- static void [gst_tensor_merge_ready_to_paused](#) ([GstTensorMerge](#) *tensor_merge)
Ready --> Pasuse State Change.
- static gboolean [gst_tensor_merge_set_option_data](#) ([GstTensorMerge](#) *tensor_merge)
Setup internal data (data_ in GstTensor_Merge)*

Variables

- static const gchar * [gst_tensor_merge_mode_string](#) []
- static const gchar * [gst_tensor_merge_linear_string](#) []
- static GstStaticPadTemplate [src_tmpl](#)
the capabilities of the inputs and outputs. describe the real formats here.
- static GstStaticPadTemplate [sink_tmpl](#)

9.42.1 Detailed Description

GStreamer plugin to merge tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

03 July 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.42.2 Macro Definition Documentation

9.42.2.1 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_WITH_NUM ("1")
```

Template caps string.

Definition at line 102 of file `gsttensor_merge.c`.

9.42.2.2 DBG

```
#define DBG (!tensor_merge->silent)
```

Macro for debug mode.

Definition at line 73 of file gsttensor_merge.c.

9.42.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_merge_debug
```

Definition at line 67 of file gsttensor_merge.c.

9.42.2.4 gst_tensor_merge_parent_class

```
#define gst_tensor_merge_parent_class parent_class
```

Definition at line 135 of file gsttensor_merge.c.

9.42.3 Enumeration Type Documentation

9.42.3.1 anonymous enum

anonymous enum

Enumerator

PROP_0	
PROP_MODE	
PROP_OPTION	
PROP_SYNC_MODE	
PROP_SYNC_OPTION	
PROP_SILENT	

Definition at line 76 of file gsttensor_merge.c.

9.42.4 Function Documentation

9.42.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorMerge ,
    gst_tensor_merge ,
    GST_TYPE_ELEMENT )
```

9.42.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_merge_debug )
```

SECTION:element-tensormerge

A Merger that merge tensor stream to tensor stream for NN frameworks. The output is always in the format of other/tensor

```
<refsect2> <title>Example launch line</title> |[ gst-launch -v -m tensor_merge name=merge ! fakesink
filesrc location=b.png ! pngdec ! videoscale ! imagefreeze ! videoconvert ! video/x-raw,format=R↵
GB,width=100,height=100,framerate=0/1 ! tensor_converter ! merge.sink_0 filesrc location=b.png ! pngdec
! videoscale ! imagefreeze ! videoconvert ! video/x-raw,format=RGB,width=100,height=100,framerate=0/1 !
tensor_converter ! merge.sink_1 filesrc location=b.png ! pngdec ! videoscale ! imagefreeze ! videoconvert !
video/x-raw,format=RGB,width=100,height=100,framerate=0/1 ! tensor_converter ! merge.sink_2 ]|
```

```
 |[ gst-launch -v -m tensor_merge name=merge ! filesink location=merge.log multifilesrc location="testsequence↵
_%1d.png" index=0 caps="image/png, framerate=(fraction)30/1" ! pngdec ! tensor_converter ! merge.sink_↵
0 multifilesrc location="testsequence_%1d.png" index=0 caps="image/png, framerate=(fraction)30/1" ! pngdec !
tensor_converter ! merge.sink_1 multifilesrc location="testsequence_%1d.png" index=0 caps="image/png, framerate↵
=(fraction)30/1" ! pngdec ! tensor_converter ! merge.sink_2
```

```
</refsect2>
```

9.42.4.3 gst_tensor_merge_change_state()

```
static GstStateChangeReturn gst_tensor_merge_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

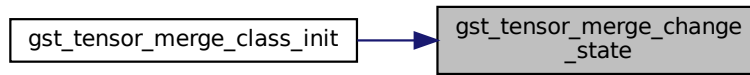
change state (gst element vmethod)

Definition at line 776 of file gstreamer_merge.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.4 `gst_tensor_merge_class_init()`

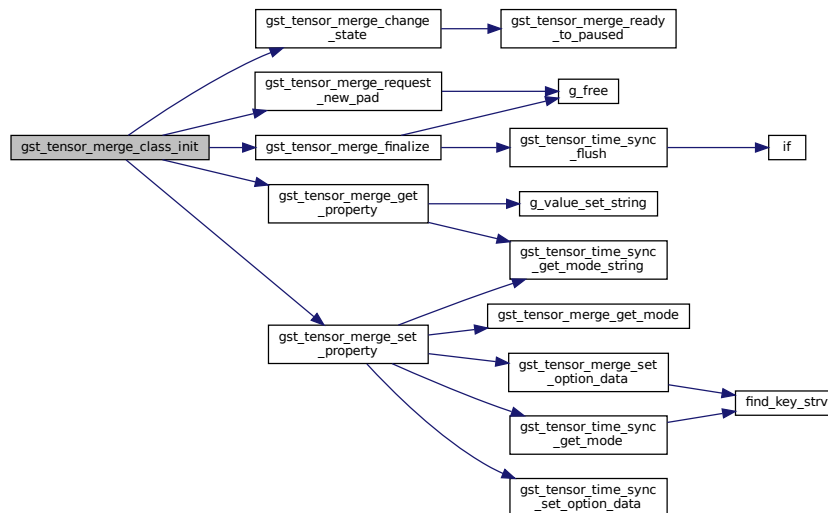
```

static void gst_tensor_merge_class_init (
    GstTensorMergeClass * klass ) [static]
  
```

initialize the tensor_merge's class

Definition at line 142 of file `gsttensor_merge.c`.

Here is the call graph for this function:



9.42.4.5 `gst_tensor_merge_collect_buffer()`

```

static gboolean gst_tensor_merge_collect_buffer (
    GstTensorMerge * tensor_merge,
    GstBuffer * tensors_buf,
    gboolean * is_eos ) [static]
  
```

Looping to generate output buffer for srcpad.

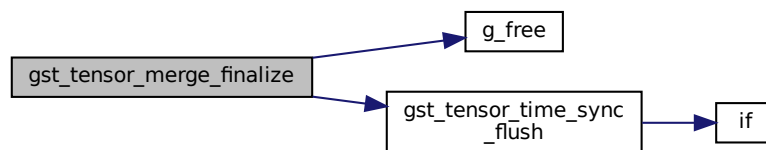
9.42.4.7 `gst_tensor_merge_finalize()`

```
static void gst_tensor_merge_finalize (
    GObject * object ) [static]
```

finalize vmethod

Definition at line 258 of file `gsttensor_merge.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.8 `gst_tensor_merge_generate_mem()`

```
static GstFlowReturn gst_tensor_merge_generate_mem (
    GstTensorMerge * tensor_merge,
    GstBuffer * tensors_buf,
    GstBuffer * tensor_buf ) [static]
```

Generate Output GstMemory.

Parameters

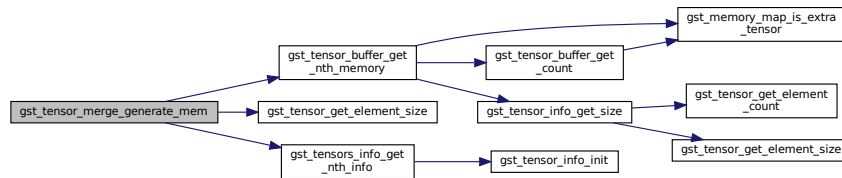
<i>tensor_merge</i>	tensor merger
<i>tensors_buf</i>	collected tensors buffer
<i>tensor_buf</i>	output tensor buffer

Returns

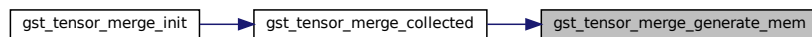
boolean

Definition at line 475 of file gsttensor_merge.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.42.4.9 gst_tensor_merge_get_merged_config()**

```

static gboolean gst_tensor_merge_get_merged_config (
    GstTensorMerge * tensor_merge,
    const GstTensorsConfig * in_config,
    GstTensorsConfig * out_config ) [static]
  
```

Generate out TensorsConfig with in TensorsConfig.

Parameters

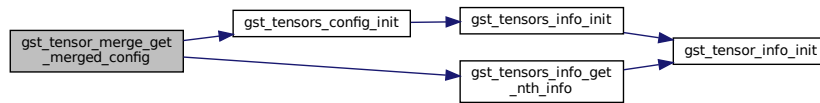
<i>tensor_merge</i>	tensor merger
<i>in_config</i>	in tensors config data (multi tensors)
<i>out_config</i>	out tensors config data (single tensor)

Returns

true / false

Definition at line 379 of file gsttensor_merge.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.10 gst_tensor_merge_get_mode()

```
static tensor\_merge\_mode gst_tensor_merge_get_mode (
    const gchar * str ) [static]
```

Get the corresponding mode from the string value.

Parameters

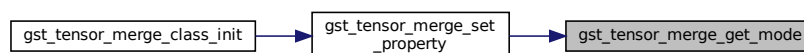
<code>in</code>	<code>str</code>	The string value for the mode
-----------------	------------------	-------------------------------

Returns

corresponding mode for the string. GTT_END for errors

Definition at line 244 of file `gsttensor_merge.c`.

Here is the caller graph for this function:



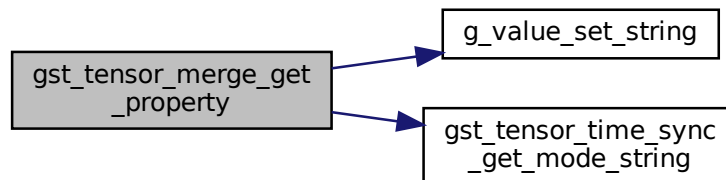
9.42.4.11 `gst_tensor_merge_get_property()`

```
static void gst_tensor_merge_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

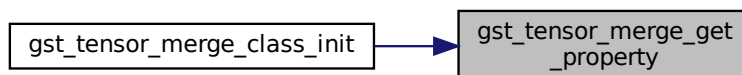
Get property (gst element vmethod)

Definition at line 894 of file `gsttensor_merge.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



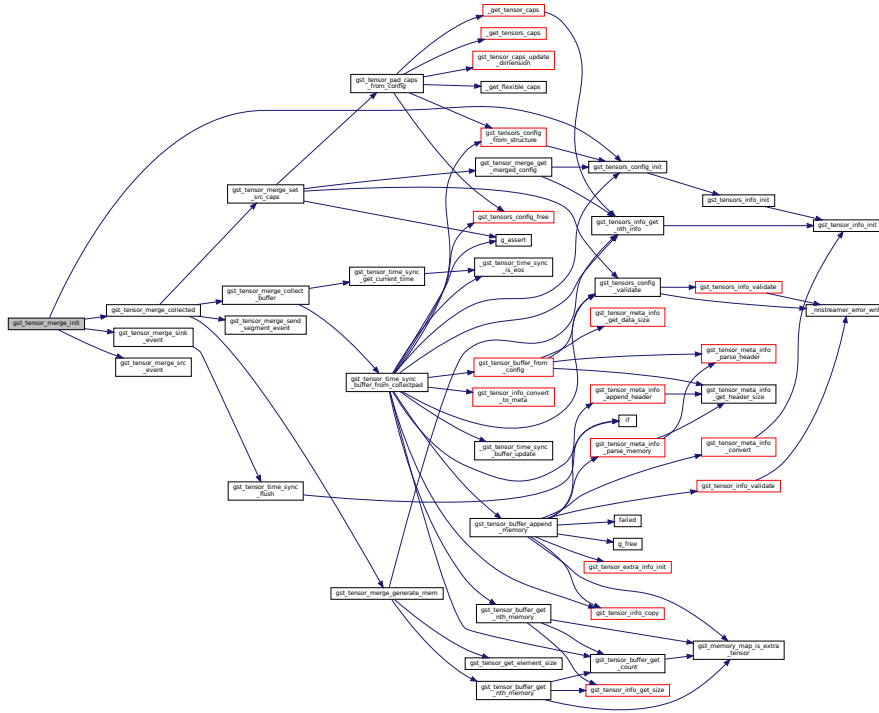
9.42.4.12 `gst_tensor_merge_init()`

```
static void gst_tensor_merge_init (
    GstTensorMerge * tensor_merge ) [static]
```

initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure

Definition at line 209 of file `gsttensor_merge.c`.

Here is the call graph for this function:



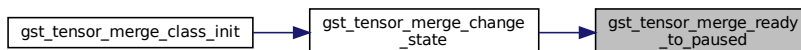
9.42.4.13 gst_tensor_merge_ready_to_paused()

```
static void gst_tensor_merge_ready_to_paused (
    GstTensorMerge * tensor_merge ) [static]
```

Ready --> Pasuse State Change.

Definition at line 764 of file gsttensor_merge.c.

Here is the caller graph for this function:



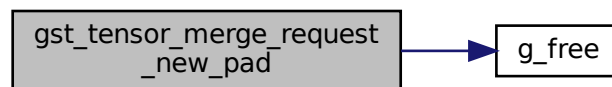
9.42.4.14 `gst_tensor_merge_request_new_pad()`

```
static GstPad * gst_tensor_merge_request_new_pad (
    GstElement * element,
    GstPadTemplate * templ,
    const gchar * name,
    const GstCaps * caps ) [static]
```

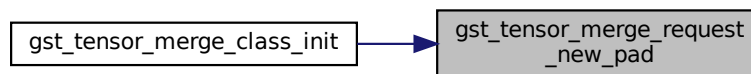
making new request pad (gst element vmethod)

Definition at line 287 of file `gsttensor_merge.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



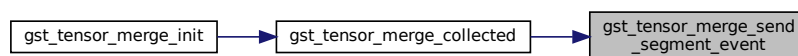
9.42.4.15 `gst_tensor_merge_send_segment_event()`

```
static void gst_tensor_merge_send_segment_event (
    GstTensorMerge * tensor_merge,
    GstClockTime pts,
    GstClockTime dts ) [static]
```

Send segment event if necessary.

Definition at line 666 of file `gsttensor_merge.c`.

Here is the caller graph for this function:



9.42.4.16 `gst_tensor_merge_set_option_data()`

```
static gboolean gst_tensor_merge_set_option_data (
    GstTensorMerge * tensor_merge ) [static]
```

Setup internal data (data_* in GstTensor_Merge)

Parameters

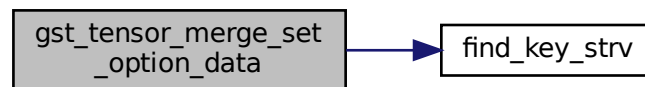
<i>[in/out]</i>	filter "this" pointer. mode & option MUST BE set already.
-----------------	---

Return values

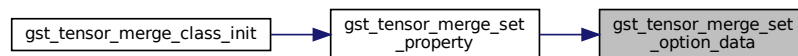
<i>TRUE</i>	if ok or not configured, yet.
<i>FALSE</i>	if given input is configured invalid.

Definition at line 812 of file gsttensor_merge.c.

Here is the call graph for this function:



Here is the caller graph for this function:



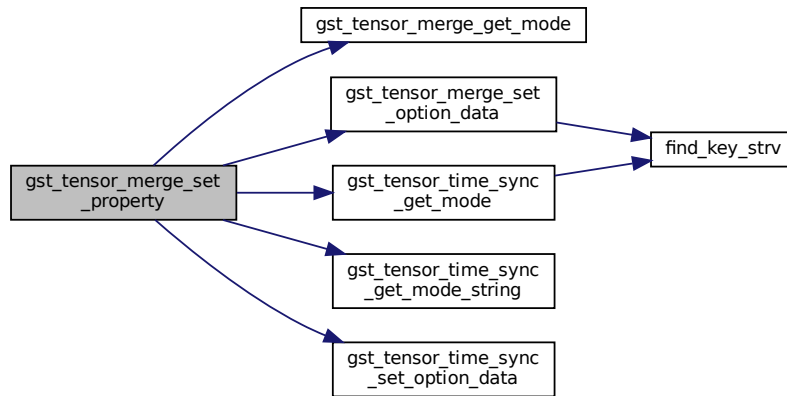
9.42.4.17 `gst_tensor_merge_set_property()`

```
static void gst_tensor_merge_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

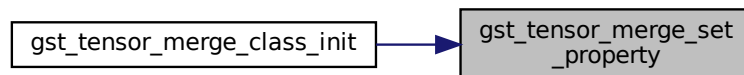
Get property (gst element vmethod)

Definition at line 841 of file gsttensor_merge.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.18 `gst_tensor_merge_set_src_caps()`

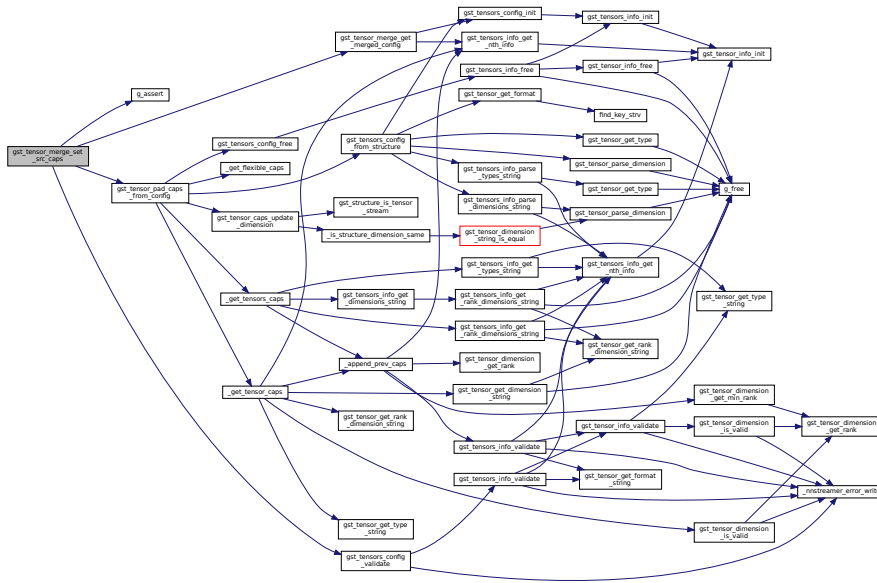
```
static gboolean gst_tensor_merge_set_src_caps (
    GstTensorMerge * tensor_merge ) [static]
```

Set src pad caps if src pad is not negotiated.

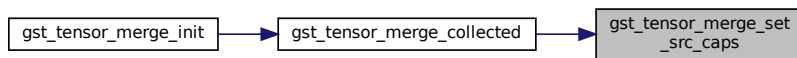
Internal Logic Error?

Definition at line 632 of file gsttensor_merge.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.19 gst_tensor_merge_sink_event()

```
static gboolean gst_tensor_merge_sink_event (
    GstCollectPads * pads,
    GstCollectData * data,
    GstEvent * event,
    GstTensorMerge * tensor_merge ) [static]
```

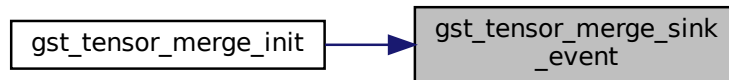
sink event vmethod

Definition at line 353 of file gsttensor_merge.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.42.4.20 `gst_tensor_merge_src_event()`

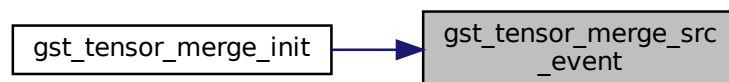
```

static gboolean gst_tensor_merge_src_event (
    GstPad * pad,
    GObject * parent,
    GstEvent * event ) [static]
  
```

src event vmethod

Definition at line 334 of file `gsttensor_merge.c`.

Here is the caller graph for this function:



9.42.5 Variable Documentation

9.42.5.1 `gst_tensor_merge_linear_string`

```

const gchar* gst_tensor_merge_linear_string[] [static]
  
```

Initial value:

```

= {
    [LINEAR_FIRST] = "0",
    [LINEAR_SECOND] = "1",
    [LINEAR_THIRD] = "2",
    [LINEAR_FOURTH] = "3",
    [LINEAR_END] = NULL,
}
  
```

Definition at line 91 of file `gsttensor_merge.c`.

9.42.5.2 `gst_tensor_merge_mode_string`

```
const gchar* gst_tensor_merge_mode_string[] [static]
```

Initial value:

```
= {  
    [GTT_LINEAR] = "linear",  
    [GTT_END] = "error",  
}
```

Definition at line 86 of file `gsttensor_merge.c`.

9.42.5.3 `sink_tmpl`

```
GstStaticPadTemplate sink_tmpl [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink_%u",  
    GST_PAD_SINK,  
    GST_PAD_REQUEST,  
    GST_STATIC_CAPS (CAPS_STRING))
```

Definition at line 113 of file `gsttensor_merge.c`.

9.42.5.4 `src_tmpl`

```
GstStaticPadTemplate src_tmpl [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

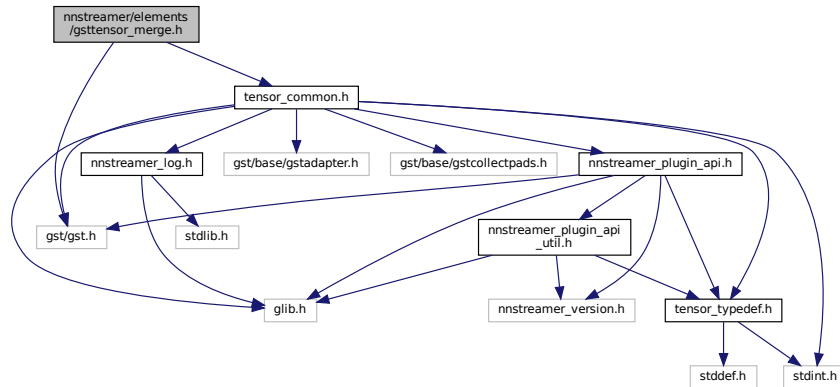
the capabilities of the inputs and outputs. describe the real formats here.

Definition at line 108 of file `gsttensor_merge.c`.

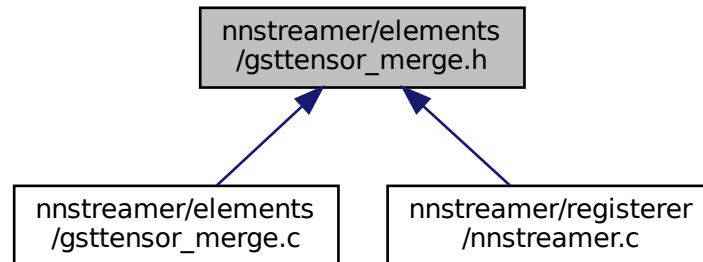
9.43 nnstreamer/elements/gsttensor_merge.h File Reference

GStreamer plugin to merge tensors (as a filter for other general neural network filters)

```
#include <gst/gst.h>
#include <tensor_common.h>
Include dependency graph for gsttensor_merge.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_tensor_merge_linear](#)
Internal data structure for linear mode.
- struct [_GstTensorMerge](#)
Tensor Merge data structure.
- struct [_GstTensorMergeClass](#)
GstTensorMergeClass inherits GstElementClass.

Macros

- #define `GST_TYPE_TENSOR_MERGE` (`gst_tensor_merge_get_type` ())
- #define `GST_TENSOR_MERGE(obj)` (`G_TYPE_CHECK_INSTANCE_CAST` ((obj), `GST_TYPE_TENSOR_MERGE`, `GstTensorMerge`))
- #define `GST_TENSOR_MERGE_CLASS(klass)` (`G_TYPE_CHECK_CLASS_CAST` ((klass), `GST_TYPE_TENSOR_MERGE`, `GstTensorMergeClass`))
- #define `GST_TENSOR_MERGE_GET_CLASS(obj)` (`G_TYPE_INSTANCE_GET_CLASS` ((obj), `GST_TYPE_TENSOR_MERGE`, `GstTensorMergeClass`))
- #define `GST_IS_TENSOR_MERGE(obj)` (`G_TYPE_CHECK_INSTANCE_TYPE`((obj),`GST_TYPE_TENSOR_MERGE`))
- #define `GST_IS_TENSOR_MERGE_CLASS(klass)` (`G_TYPE_CHECK_CLASS_TYPE`((klass),`GST_TYPE_TENSOR_MERGE`))
- #define `GST_TENSOR_MERGE_CAST(obj)` ((`GstTensorMerge*`)(obj))

Typedefs

- typedef struct `_GstTensorMerge` `GstTensorMerge`
- typedef struct `_GstTensorMergeClass` `GstTensorMergeClass`
- typedef struct `_tensor_merge_linear` `tensor_merge_linear`
Internal data structure for linear mode.

Enumerations

- enum `tensor_merge_mode` { `GTT_LINEAR` = 0, `GTT_END` }
- enum `tensor_merge_linear_mode` { `LINEAR_FIRST` = 0, `LINEAR_SECOND` = 1, `LINEAR_THIRD` = 2, `LINEAR_FOURTH` = 3, `LINEAR_END` }

Functions

- GType `gst_tensor_merge_get_type` (void)
Get Type function required for gst elements.

9.43.1 Detailed Description

GStreamer plugin to merge tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

03 July 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.43.2 Macro Definition Documentation

9.43.2.1 GST_IS_TENSOR_MERGE

```
#define GST_IS_TENSOR_MERGE(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_TENSOR_MERGE))
```

Definition at line 39 of file gsttensor_merge.h.

9.43.2.2 GST_IS_TENSOR_MERGE_CLASS

```
#define GST_IS_TENSOR_MERGE_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_TENSOR_MERGE))
```

Definition at line 40 of file gsttensor_merge.h.

9.43.2.3 GST_TENSOR_MERGE

```
#define GST_TENSOR_MERGE(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_TENSOR_MERGE, GstTensorMerge))
```

Definition at line 36 of file gsttensor_merge.h.

9.43.2.4 GST_TENSOR_MERGE_CAST

```
#define GST_TENSOR_MERGE_CAST(  
    obj ) ((GstTensorMerge*)(obj))
```

Definition at line 41 of file gsttensor_merge.h.

9.43.2.5 GST_TENSOR_MERGE_CLASS

```
#define GST_TENSOR_MERGE_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_TENSOR_MERGE, GstTensorMergeClass))
```

Definition at line 37 of file gsttensor_merge.h.

9.43.2.6 GST_TENSOR_MERGE_GET_CLASS

```
#define GST_TENSOR_MERGE_GET_CLASS(  
    obj ) (G_TYPE_INSTANCE_GET_CLASS ((obj), GST_TYPE_TENSOR_MERGE, GstTensorMergeClass))
```

Definition at line 38 of file gsttensor_merge.h.

9.43.2.7 GST_TYPE_TENSOR_MERGE

```
#define GST_TYPE_TENSOR_MERGE (gst_tensor_merge_get_type ())
```

Definition at line 35 of file gsttensor_merge.h.

9.43.3 Typedef Documentation

9.43.3.1 GstTensorMerge

```
typedef struct _GstTensorMerge GstTensorMerge
```

Definition at line 42 of file gsttensor_merge.h.

9.43.3.2 GstTensorMergeClass

```
typedef struct _GstTensorMergeClass GstTensorMergeClass
```

Definition at line 43 of file gsttensor_merge.h.

9.43.3.3 tensor_merge_linear

```
typedef struct _tensor_merge_linear tensor_merge_linear
```

Internal data structure for linear mode.

9.43.4 Enumeration Type Documentation

9.43.4.1 tensor_merge_linear_mode

```
enum tensor_merge_linear_mode
```

Enumerator

LINEAR_FIRST	
LINEAR_SECOND	
LINEAR_THIRD	
LINEAR_FOURTH	
LINEAR_END	

Definition at line 51 of file gsttensor_merge.h.

9.43.4.2 tensor_merge_mode

enum `tensor_merge_mode`

Enumerator

GTT_LINEAR	
GTT_END	

Definition at line 45 of file gsttensor_merge.h.

9.43.5 Function Documentation**9.43.5.1 gst_tensor_merge_get_type()**

```
GType gst_tensor_merge_get_type (  
    void )
```

Get Type function required for gst elements.

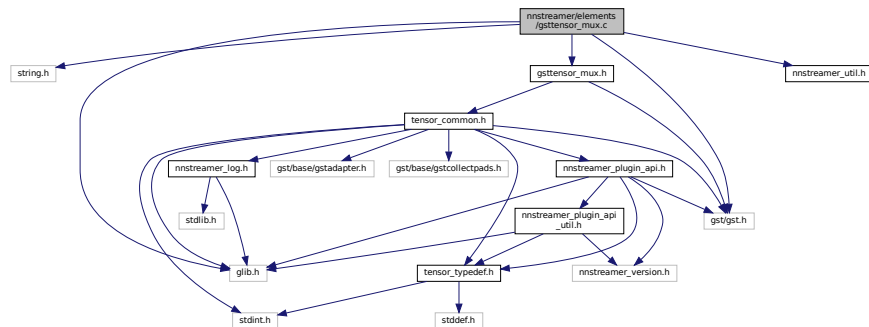
9.44 nnstreamer/elements/gsttensor_mux.c File Reference

GStreamer plugin to mux tensors (as a filter for other general neural network filters)

```
#include <string.h>  
#include <gst/gst.h>  
#include <glib.h>  
#include <nnstreamer_util.h>
```

```
#include "gsttensor_mux.h"
```

Include dependency graph for `gsttensor_mux.c`:



Macros

- `#define` [GST_CAT_DEFAULT](#) `gst_tensor_mux_debug`
- `#define` [DBG](#) (`!tensor_mux->silent`)
Macro for debug mode.
- `#define` [CAPS_STRING_SINK](#) `GST_TENSOR_CAP_DEFAULT` `","` `GST_TENSORS_CAP_MAKE` (`"{ static, flexible }"`)
Default caps string for sink pad.
- `#define` [CAPS_STRING_SRC](#) `GST_TENSORS_CAP_MAKE` (`"{ static, flexible }"`)
Default caps string for src pad.
- `#define` [gst_tensor_mux_parent_class](#) `parent_class`

Enumerations

- `enum` { [PROP_0](#), [PROP_SILENT](#), [PROP_SYNC_MODE](#), [PROP_SYNC_OPTION](#) }

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) (`gst_tensor_mux_debug`)
- static gboolean [gst_tensor_mux_src_event](#) (`GstPad *pad`, `GstObject *parent`, `GstEvent *event`)
src event vmethod
- static `GstPad *` [gst_tensor_mux_request_new_pad](#) (`GstElement *element`, `GstPadTemplate *templ`, `const gchar *req_name`, `const GstCaps *caps`)
making new request pad (gst element vmethod)
- static `GstStateChangeReturn` [gst_tensor_mux_change_state](#) (`GstElement *element`, `GstStateChange` transition)
change state (gst element vmethod)
- static gboolean [gst_tensor_mux_sink_event](#) (`GstCollectPads *pads`, `GstCollectData *data`, `GstEvent *event`, [GstTensorMux](#) *`tensor_mux`)
sink event vmethod
- static `GstFlowReturn` [gst_tensor_mux_collected](#) (`GstCollectPads *pads`, [GstTensorMux](#) *`tensor_mux`)
Gst Collect Pads Function which is called once collect pads done.
- static `GstFlowReturn` [gst_tensor_mux_do_clip](#) (`GstCollectPads *pads`, `GstCollectData *data`, `GstBuffer *buffer`, `GstBuffer **out`, [GstTensorMux](#) *`tensor_mux`)

- Gst Clip Pads Function which is called right after a buffer is received for each pad.*
- static void [gst_tensor_mux_set_property](#) (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
 - Get property (gst element vmethod)*
 - static void [gst_tensor_mux_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
 - Get property (gst element vmethod)*
 - static void [gst_tensor_mux_finalize](#) (GObject *object)
 - finalize vmethod*
 - [G_DEFINE_TYPE](#) (GstTensorMux, gst_tensor_mux, GST_TYPE_ELEMENT)
 - static void [gst_tensor_mux_class_init](#) (GstTensorMuxClass *klass)
 - initialize the tensor_mux's class*
 - static void [gst_tensor_mux_init](#) (GstTensorMux *tensor_mux)
 - initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure*
 - static void [gst_tensor_mux_set_waiting](#) (GstTensorMux *tensor_mux, gboolean waiting)
 - set pads waiting property*
 - static gboolean [gst_tensor_mux_collect_buffer](#) (GstTensorMux *tensor_mux, GstBuffer *tensors_buf, gboolean *is_eos)
 - Looping to generate output buffer for srcpad.*
 - static gboolean [gst_tensor_mux_set_src_caps](#) (GstTensorMux *tensor_mux)
 - Set src pad caps if src pad is not negotiated.*
 - static void [gst_tensor_mux_send_segment_event](#) (GstTensorMux *tensor_mux, GstClockTime pts, GstClockTime dts)
 - Create a new segment event if necessary.*
 - static GstBuffer * [gst_tensor_mux_chain_flex_tensor](#) (GstTensorMux *tensor_mux, GstBuffer *buf)
 - Process flex tensor.*
 - static void [gst_tensor_mux_ready_to_paused](#) (GstTensorMux *tensor_mux)
 - Ready --> Pasuse State Change.*

Variables

- static GstStaticPadTemplate [src_tmpl](#)
 - the capabilities of the inputs and outputs. describe the real formats here.*
- static GstStaticPadTemplate [sink_tmpl](#)

9.44.1 Detailed Description

GStreamer plugin to mux tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

03 July 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.44.2 Macro Definition Documentation

9.44.2.1 CAPS_STRING_SINK

```
#define CAPS_STRING_SINK GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_MAKE ("{ static, flexible }")
```

Default caps string for sink pad.

Definition at line 89 of file gsttensor_mux.c.

9.44.2.2 CAPS_STRING_SRC

```
#define CAPS_STRING_SRC GST_TENSORS_CAP_MAKE ("{ static, flexible }")
```

Default caps string for src pad.

Definition at line 94 of file gsttensor_mux.c.

9.44.2.3 DBG

```
#define DBG (!tensor_mux->silent)
```

Macro for debug mode.

Definition at line 75 of file gsttensor_mux.c.

9.44.2.4 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_mux_debug
```

Definition at line 69 of file gsttensor_mux.c.

9.44.2.5 gst_tensor_mux_parent_class

```
#define gst_tensor_mux_parent_class parent_class
```

Definition at line 132 of file gsttensor_mux.c.

9.44.3 Enumeration Type Documentation

9.44.3.1 anonymous enum

anonymous enum

Enumerator

PROP_0	
PROP_SILENT	
PROP_SYNC_MODE	
PROP_SYNC_OPTION	

Definition at line 78 of file gsttensor_mux.c.

9.44.4 Function Documentation

9.44.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorMux ,
    gst_tensor_mux ,
    GST_TYPE_ELEMENT )
```


9.44.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_mux_debug )
```

SECTION:element-tensormux

A Muxer that merge tensor stream to tensors stream for NN frameworks. The output is always in the format of other/tensors

```
<refsect2> <title>Example launch line</title> |[ gst-launch -v -m \ filesrc location=b.png ! pngdec ! videoscale ! imagefreeze ! videoconvert ! video/x-raw,format=RGB,width=100,height=100,framerate=0/1 ! tensor_converter ! mux.sink_0 \ filesrc location=b.png ! pngdec ! videoscale ! imagefreeze ! videoconvert ! video/x-raw,format=RGB,width=100,height=100,framerate=0/1 ! tensor_converter ! mux.sink_1 \ filesrc location=b.png ! pngdec ! videoscale ! imagefreeze ! videoconvert ! video/x-raw,format=RGB,width=100,height=100,framerate=0/1 ! tensor_converter ! mux.sink_2 \ tensor_mux name=mux ! fakesink ]|
```

```
[| gst-launch -v -m \ multifilesrc location="testsequence_%1d.png" index=0 caps="image/png, framerate=(fraction)30/1" ! pngdec ! tensor_converter ! mux.sink_0 \ multifilesrc location="testsequence_%1d.png" index=0 caps="image/png, framerate=(fraction)30/1" ! pngdec ! tensor_converter ! mux.sink_1 \ multifilesrc location="testsequence_%1d.png" index=0 caps="image/png, framerate=(fraction)30/1" ! pngdec ! tensor_converter ! mux.sink_2 \ tensor_mux name=mux ! filesink location=mux.log ]| </refsect2>
```

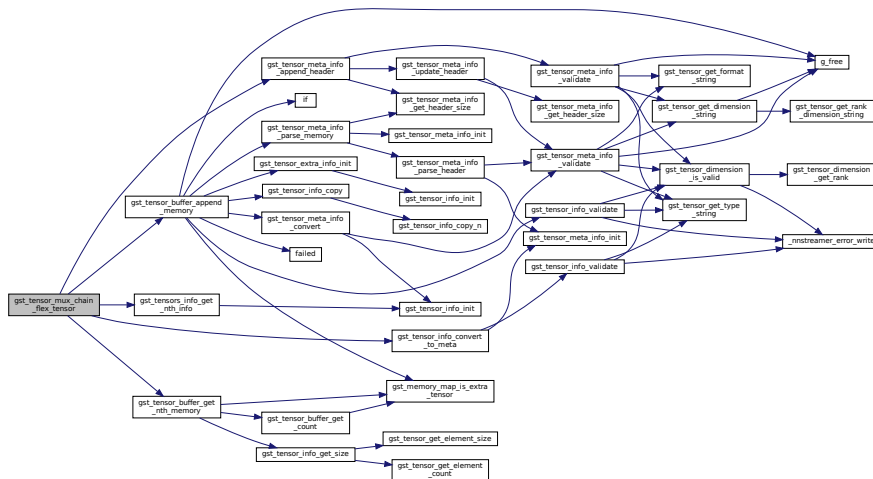
9.44.4.3 gst_tensor_mux_chain_flex_tensor()

```
static GstBuffer* gst_tensor_mux_chain_flex_tensor (
    GstTensorMux * tensor_mux,
    GstBuffer * buf ) [static]
```

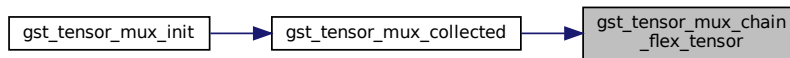
Process flex tensor.

Definition at line 446 of file gsttensor_mux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.44.4.4 `gst_tensor_mux_change_state()`

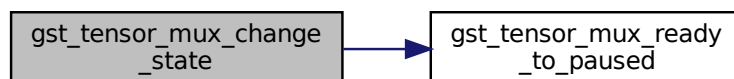
```

static GstStateChangeReturn gst_tensor_mux_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
  
```

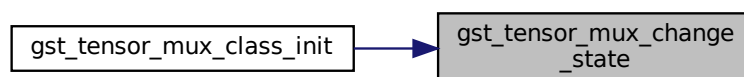
change state (gst element vmethod)

Definition at line 584 of file `gsttensor_mux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



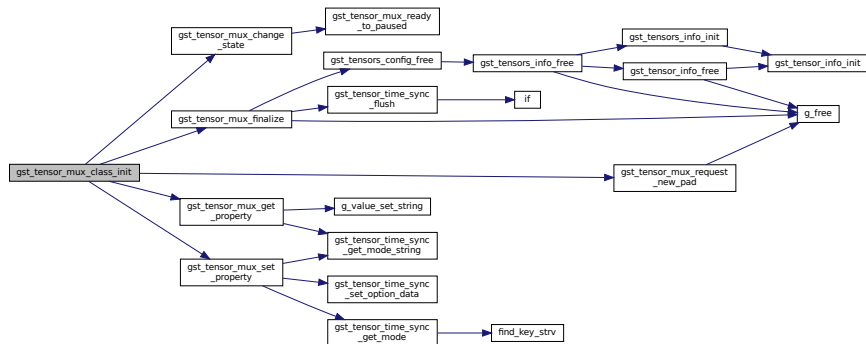
9.44.4.5 `gst_tensor_mux_class_init()`

```
static void gst_tensor_mux_class_init (
    GstTensorMuxClass * klass ) [static]
```

initialize the tensor_mux's class

Definition at line 139 of file `gsttensor_mux.c`.

Here is the call graph for this function:



9.44.4.6 `gst_tensor_mux_collect_buffer()`

```
static gboolean gst_tensor_mux_collect_buffer (
    GstTensorMux * tensor_mux,
    GstBuffer * tensors_buf,
    gboolean * is_eos ) [static]
```

Looping to generate output buffer for srcpad.

Parameters

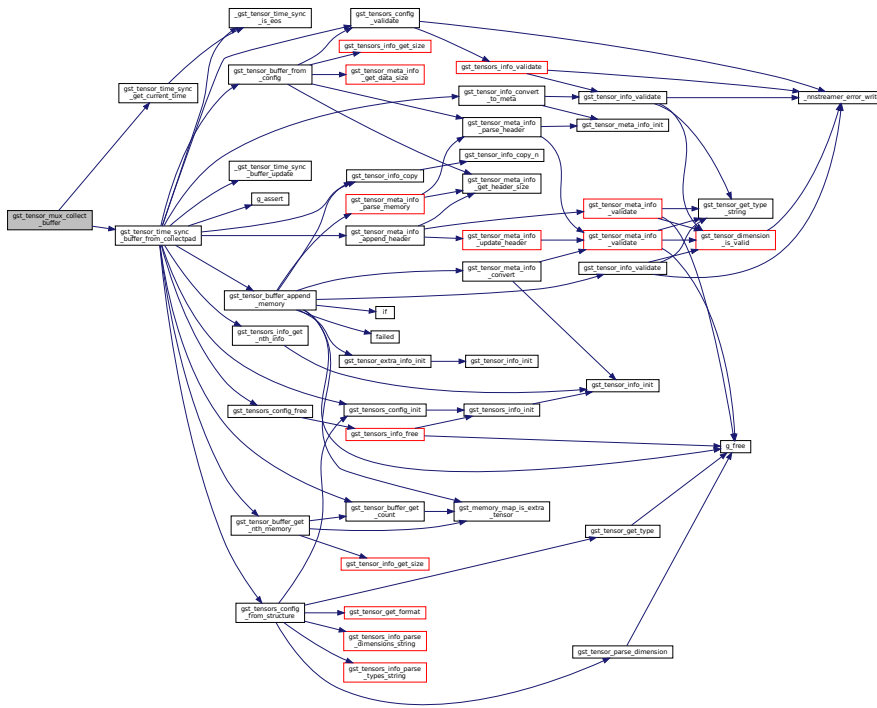
<i>tensor_mux</i>	tensor muxer
<i>tensors_buf</i>	output buffer for srcpad
<i>is_eos</i>	boolean EOS (End of Stream)

Returns

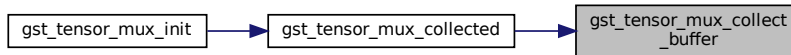
TRUE to push buffer to src pad

Definition at line 368 of file `gsttensor_mux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.44.4.7 gst_tensor_mux_collected()

```
static GstFlowReturn gst_tensor_mux_collected (
    GstCollectPads * pads,
    GstTensorMux * tensor_mux ) [static]
```

Gst Collect Pads Function which is called once collect pads done.

Parameters

<i>pads</i>	GstCollectPads
<i>tensor_mux</i>	Muxer

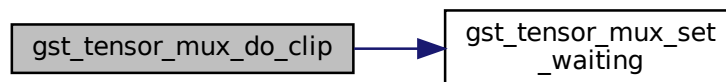
9.44.4.8 `gst_tensor_mux_do_clip()`

```
static GstFlowReturn gst_tensor_mux_do_clip (
    GstCollectPads * pads,
    GstCollectData * data,
    GstBuffer * buffer,
    GstBuffer ** out,
    GstTensorMux * tensor_mux ) [static]
```

Gst Clip Pads Function which is called right after a buffer is received for each pad.

Definition at line 558 of file `gsttensor_mux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



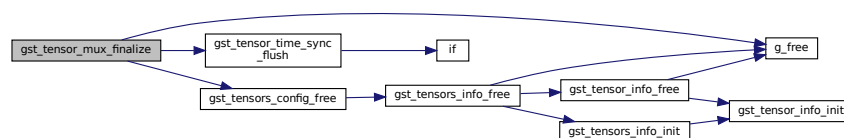
9.44.4.9 `gst_tensor_mux_finalize()`

```
static void gst_tensor_mux_finalize (
    GObject * object ) [static]
```

finalize vmethod

Definition at line 227 of file `gsttensor_mux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



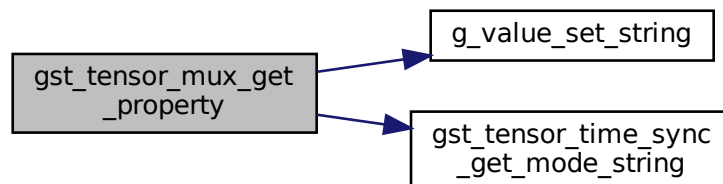
9.44.4.10 `gst_tensor_mux_get_property()`

```
static void gst_tensor_mux_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

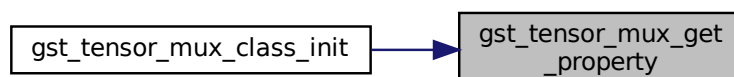
Get property (gst element vmethod)

Definition at line 650 of file `gsttensor_mux.c`.

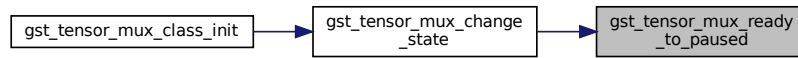
Here is the call graph for this function:



Here is the caller graph for this function:



Here is the caller graph for this function:



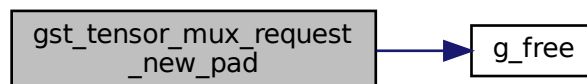
9.44.4.13 gst_tensor_mux_request_new_pad()

```
static GstPad * gst_tensor_mux_request_new_pad (  
    GstElement * element,  
    GstPadTemplate * templ,  
    const gchar * name,  
    const GstCaps * caps ) [static]
```

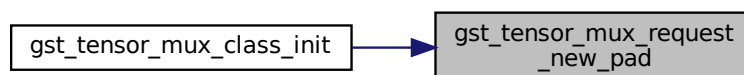
making new request pad (gst element vmethod)

Definition at line 253 of file gsttensor_mux.c.

Here is the call graph for this function:



Here is the caller graph for this function:



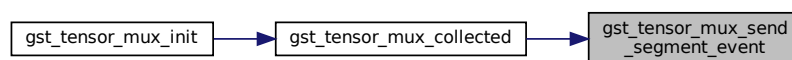
9.44.4.14 `gst_tensor_mux_send_segment_event()`

```
static void gst_tensor_mux_send_segment_event (
    GstTensorMux * tensor_mux,
    GstClockTime pts,
    GstClockTime dts ) [static]
```

Create a new segment event if necessary.

Definition at line 422 of file `gsttensor_mux.c`.

Here is the caller graph for this function:



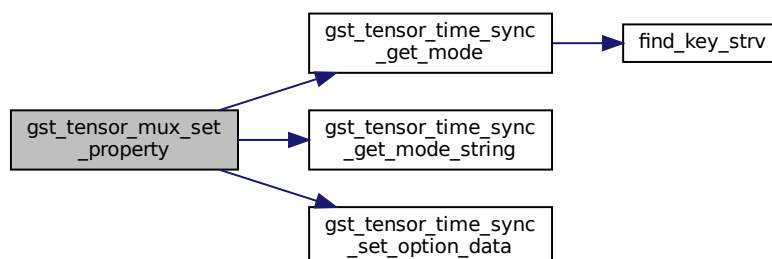
9.44.4.15 `gst_tensor_mux_set_property()`

```
static void gst_tensor_mux_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

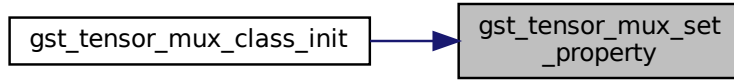
Get property (gst element vmethod)

Definition at line 617 of file `gsttensor_mux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.44.4.16 `gst_tensor_mux_set_src_caps()`

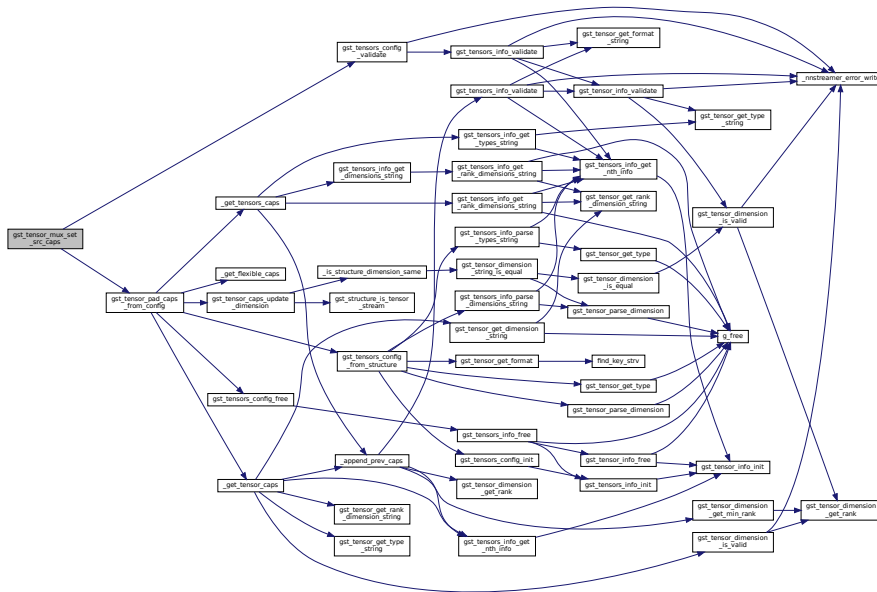
```

static gboolean gst_tensor_mux_set_src_caps (
    GstTensorMux * tensor_mux ) [static]
    
```

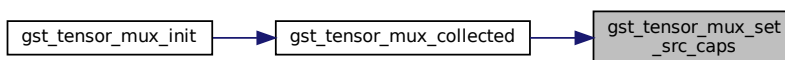
Set src pad caps if src pad is not negotiated.

Definition at line 393 of file `gsttensor_mux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



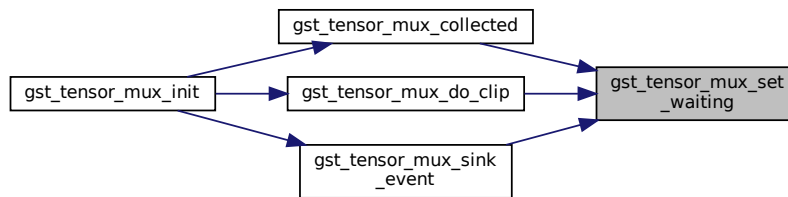
9.44.4.17 `gst_tensor_mux_set_waiting()`

```
static void gst_tensor_mux_set_waiting (
    GstTensorMux * tensor_mux,
    gboolean waiting ) [static]
```

set pads waiting property

Definition at line 322 of file `gsttensor_mux.c`.

Here is the caller graph for this function:



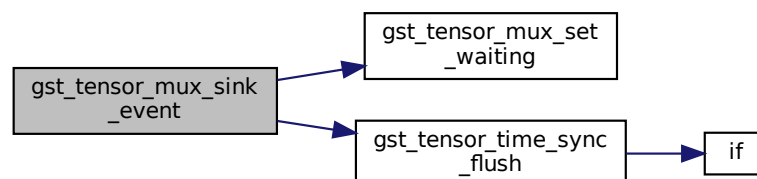
9.44.4.18 `gst_tensor_mux_sink_event()`

```
static gboolean gst_tensor_mux_sink_event (
    GstCollectPads * pads,
    GstCollectData * data,
    GstEvent * event,
    GstTensorMux * tensor_mux ) [static]
```

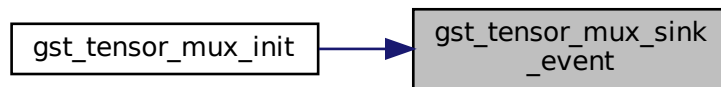
sink event vmethod

Definition at line 339 of file `gsttensor_mux.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.44.4.19 `gst_tensor_mux_src_event()`

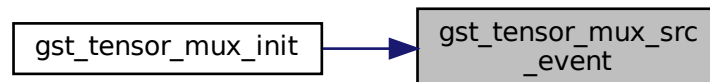
```

static gboolean gst_tensor_mux_src_event (
    GstPad * pad,
    GObject * parent,
    GstEvent * event ) [static]
  
```

src event vmethod

Definition at line 303 of file `gsttensor_mux.c`.

Here is the caller graph for this function:



9.44.5 Variable Documentation

9.44.5.1 `sink_tmpl`

```
GstStaticPadTemplate sink_tmpl [static]
```

Initial value:

```

= GST_STATIC_PAD_TEMPLATE ("sink_%u",
    GST_PAD_SINK,
    GST_PAD_REQUEST,
)
  
```

Definition at line 106 of file `gsttensor_mux.c`.

9.44.5.2 src_tmpl

```
GstStaticPadTemplate src_tmpl [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
)
```

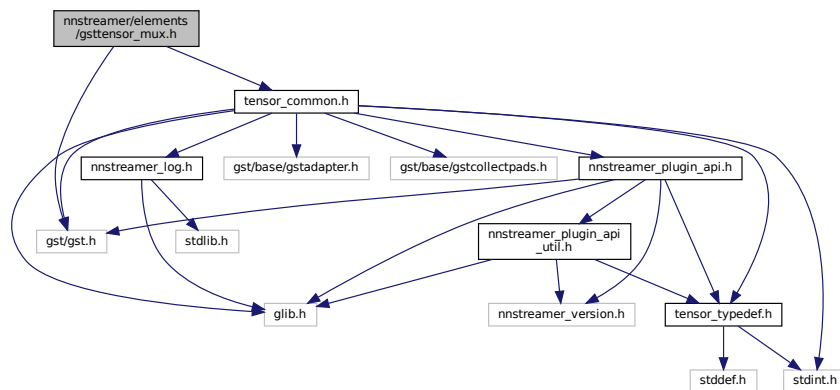
the capabilities of the inputs and outputs. describe the real formats here.

Definition at line 100 of file gsttensor_mux.c.

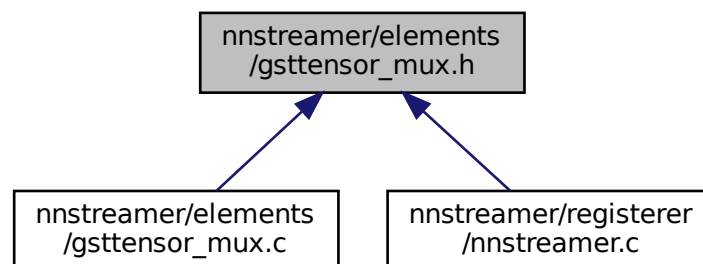
9.45 nnstreamer/elements/gsttensor_mux.h File Reference

GStreamer plugin to mux tensors (as a filter for other general neural network filters)

```
#include <gst/gst.h>
#include <tensor_common.h>
Include dependency graph for gsttensor_mux.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorMux](#)
Tensor Muxer data structure.
- struct [_GstTensorMuxClass](#)
GstTensorMuxClass inherits GstElementClass.

Macros

- #define [GST_TYPE_TENSOR_MUX](#) ([gst_tensor_mux_get_type](#) ())
- #define [GST_TENSOR_MUX\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST](#) ((obj), [GST_TYPE_TENSOR_MUX](#), [GstTensorMux](#)))
- #define [GST_TENSOR_MUX_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST](#) ((klass), [GST_TYPE_TENSOR_MUX](#), [GstTensorMuxClass](#)))
- #define [GST_TENSOR_MUX_GET_CLASS\(obj\)](#) ([G_TYPE_INSTANCE_GET_CLASS](#) ((obj), [GST_TYPE_TENSOR_MUX](#), [GstTensorMuxClass](#)))
- #define [GST_IS_TENSOR_MUX\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_MUX](#)))
- #define [GST_IS_TENSOR_MUX_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_MUX](#)))
- #define [GST_TENSOR_MUX_CAST\(obj\)](#) (([GstTensorMux*](#))(obj))

Typedefs

- typedef struct [_GstTensorMux](#) [GstTensorMux](#)
- typedef struct [_GstTensorMuxClass](#) [GstTensorMuxClass](#)

Functions

- GType [gst_tensor_mux_get_type](#) (void)
Get Type function required for gst elements.

9.45.1 Detailed Description

GStreamer plugin to mux tensors (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

03 July 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.45.2 Macro Definition Documentation

9.45.2.1 GST_IS_TENSOR_MUX

```
#define GST_IS_TENSOR_MUX(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_TENSOR_MUX))
```

Definition at line 39 of file gsttensor_mux.h.

9.45.2.2 GST_IS_TENSOR_MUX_CLASS

```
#define GST_IS_TENSOR_MUX_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_TENSOR_MUX))
```

Definition at line 40 of file gsttensor_mux.h.

9.45.2.3 GST_TENSOR_MUX

```
#define GST_TENSOR_MUX(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_TENSOR_MUX, GstTensorMux))
```

Definition at line 36 of file gsttensor_mux.h.

9.45.2.4 GST_TENSOR_MUX_CAST

```
#define GST_TENSOR_MUX_CAST(  
    obj ) ((GstTensorMux*) (obj))
```

Definition at line 41 of file gsttensor_mux.h.

9.45.2.5 GST_TENSOR_MUX_CLASS

```
#define GST_TENSOR_MUX_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_TENSOR_MUX, GstTensorMuxClass))
```

Definition at line 37 of file gsttensor_mux.h.

9.45.2.6 GST_TENSOR_MUX_GET_CLASS

```
#define GST_TENSOR_MUX_GET_CLASS(  
    obj ) (G_TYPE_INSTANCE_GET_CLASS ((obj), GST_TYPE_TENSOR_MUX, GstTensorMuxClass))
```

Definition at line 38 of file gsttensor_mux.h.

9.45.2.7 GST_TYPE_TENSOR_MUX

```
#define GST_TYPE_TENSOR_MUX (gst_tensor_mux_get_type ())
```

Definition at line 35 of file gsttensor_mux.h.

9.45.3 Typedef Documentation

9.45.3.1 GstTensorMux

```
typedef struct _GstTensorMux GstTensorMux
```

Definition at line 42 of file gsttensor_mux.h.

9.45.3.2 GstTensorMuxClass

```
typedef struct _GstTensorMuxClass GstTensorMuxClass
```

Definition at line 43 of file gsttensor_mux.h.

9.45.4 Function Documentation

9.45.4.1 gst_tensor_mux_get_type()

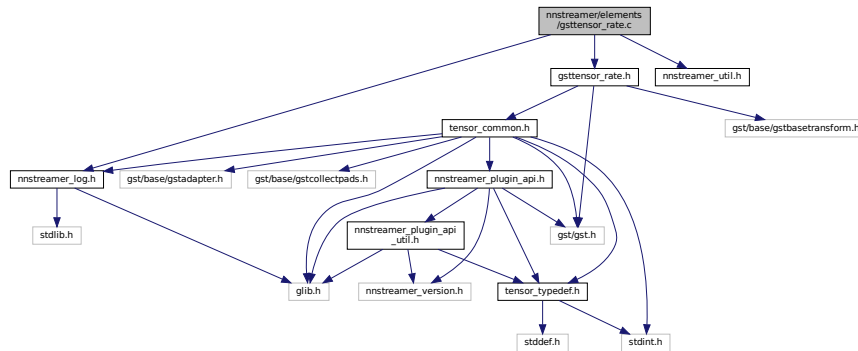
```
GType gst_tensor_mux_get_type (  
    void )
```

Get Type function required for gst elements.

9.46 nnstreamer/elements/gsttensor_rate.c File Reference

GStreamer plugin to adjust tensor rate.

```
#include <nnstreamer_log.h>
#include <nnstreamer_util.h>
#include "gsttensor_rate.h"
Include dependency graph for gsttensor_rate.c:
```



Macros

- `#define DBG (!self->silent)`
Macro for debug mode.
- `#define ABSDIFF(a, b) (((a) > (b)) ? (a) - (b) : (b) - (a))`
- `#define GST_CAT_DEFAULT gst_tensor_rate_debug`
- `#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_DEFAULT`
- `#define GST_TENSOR_RATE_SCALED_TIME(self, count)`
- `#define DEFAULT_SILENT TRUE`
default parameters
- `#define DEFAULT_THROTTLE TRUE`
- `#define gst_tensor_rate_parent_class parent_class`
- `#define THROTTLE_DELAY_RATIO (0.999)`
- `#define MAGIC_LIMIT 25`

Enumerations

- enum {
`PROP_0, PROP_IN, PROP_OUT, PROP_DUP,`
`PROP_DROP, PROP_SILENT, PROP_THROTTLE, PROP_FRAMERATE }`
tensor_rate properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) (`gst_tensor_rate_debug`)
- [G_DEFINE_TYPE](#) (`GstTensorRate`, `gst_tensor_rate`, `GST_TYPE_BASE_TRANSFORM`)
- static void [gst_tensor_rate_set_property](#) (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
Setter for tensor_rate properties.
- static void [gst_tensor_rate_get_property](#) (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)
Getter for tensor_rate properties.
- static void [gst_tensor_rate_finalize](#) (`GObject *object`)
Function to finalize instance. (GST Standard)
- static `GstFlowReturn` [gst_tensor_rate_transform_ip](#) (`GstBaseTransform *trans`, `GstBuffer *buffer`)
in-place transform
- static `GstCaps *` [gst_tensor_rate_transform_caps](#) (`GstBaseTransform *trans`, `GstPadDirection direction`, `GstCaps *caps`, `GstCaps *filter`)
configure tensor-srcpad cap from "proposed" cap. (GST Standard)
- static `GstCaps *` [gst_tensor_rate_fixate_caps](#) (`GstBaseTransform *trans`, `GstPadDirection direction`, `GstCaps *caps`, `GstCaps *othercaps`)
fixate caps. required vmethod of GstBaseTransform.
- static `gboolean` [gst_tensor_rate_set_caps](#) (`GstBaseTransform *trans`, `GstCaps *in_caps`, `GstCaps *out_caps`)
set caps. required vmethod of GstBaseTransform.
- static void [gst_tensor_rate_swap_prev](#) (`GstTensorRate *self`, `GstBuffer *buffer`, `gint64 time`)
swap a previous buffer
- static `GstFlowReturn` [gst_tensor_rate_flush_prev](#) (`GstTensorRate *self`, `gboolean duplicate`, `GstClockTime next_intime`)
flush the oldest buffer
- static void [gst_tensor_rate_notify_drop](#) (`GstTensorRate *self`)
notify a frame drop event
- static void [gst_tensor_rate_notify_duplicate](#) (`GstTensorRate *self`)
notify a frame duplicate event
- static `gboolean` [gst_tensor_rate_start](#) (`GstBaseTransform *trans`)
Called when the element starts processing. optional vmethod of BaseTransform.
- static `gboolean` [gst_tensor_rate_stop](#) (`GstBaseTransform *trans`)
Called when the element stops processing. optional vmethod of BaseTransform.
- static `gboolean` [gst_tensor_rate_sink_event](#) (`GstBaseTransform *trans`, `GstEvent *event`)
Event handler for sink pad of tensor rate.
- static void [gst_tensor_rate_install_properties](#) (`GObjectClass *object_class`)
Installs all the properties for tensor_rate.
- static void [gst_tensor_rate_class_init](#) (`GstTensorRateClass *klass`)
initialize the tensor_rate's class (GST Standard)
- static `GstFlowReturn` [gst_tensor_rate_push_buffer](#) (`GstTensorRate *self`, `GstBuffer *outbuf`, `gboolean duplicate`, `GstClockTime next_intime`)
push the buffer to src pad
- static void [gst_tensor_rate_reset](#) (`GstTensorRate *self`)
reset variables of the element (GST Standard)
- static void [gst_tensor_rate_init](#) (`GstTensorRate *self`)
initialize the new element (GST Standard)
- static void [gst_tensor_rate_send_qos_throttle](#) (`GstTensorRate *self`, `GstClockTime timestamp`)
send throttling qos event to upstream elements

Variables

- static GstStaticPadTemplate [sink_factory](#)
The capabilities of the inputs.
- static GstStaticPadTemplate [src_factory](#)
The capabilities of the outputs.
- static GParamSpec * [pspec_drop](#) = NULL
- static GParamSpec * [pspec_duplicate](#) = NULL

9.46.1 Detailed Description

GStreamer plugin to adjust tensor rate.

GStreamer/NNStreamer Tensor-Rate Copyright (C) 2020 Dongju Chae dongju.chae@samsung.com

Date

24 Sep 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Dongju Chae dongju.chae@samsung.com

Bug No known bugs except for NYI items

9.46.2 Macro Definition Documentation

9.46.2.1 ABSDIFF

```
#define ABSDIFF(  
    a,  
    b ) ((a) > (b)) ? (a) - (b) : (b) - (a)
```

Definition at line 60 of file gsttensor_rate.c.

9.46.2.2 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT "; " GST_TENSORS_CAP_DEFAULT
```

Definition at line 66 of file gsttensor_rate.c.

9.46.2.3 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

SECTION:element-tensor_rate

This element controls a frame rate of tensor streams in the pipeline.

Basically, this element takes an incoming stream of tensor frames, and produces an adjusted stream that matches the source pad's framerate. The adjustment is performed by dropping and duplicating tensor frames. By default the element will simply negotiate the same framerate on its source and sink pad.

Also, when 'throttle' property is set, it propagates a specified frame-rate to upstream elements by sending qos events, which prevents unnecessary data from upstream elements.

```
<refsect2> <title>Example launch line with tensor rate</title> gst-launch-1.0 videotestsrc ! video/x-raw,width=640,height=480,framerate=15/1 ! tensor_converter ! tensor_rate framerate=10/1 throttle=true ! tensor_←_decoder mode=direct_video ! videoconvert ! autovideosink </refsect2>
```

Definition at line 56 of file gsttensor_rate.c.

9.46.2.4 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

default parameters

Definition at line 73 of file gsttensor_rate.c.

9.46.2.5 DEFAULT_THROTTLE

```
#define DEFAULT_THROTTLE TRUE
```

Definition at line 74 of file gsttensor_rate.c.

9.46.2.6 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_rate_debug
```

Definition at line 64 of file gsttensor_rate.c.

9.46.2.7 `gst_tensor_rate_parent_class`

```
#define gst_tensor_rate_parent_class parent_class
```

Definition at line 110 of file `gsttensor_rate.c`.

9.46.2.8 `GST_TENSOR_RATE_SCALED_TIME`

```
#define GST_TENSOR_RATE_SCALED_TIME(  
    self,  
    count )
```

Value:

```
gst_util_uint64_scale (count,\  
    self->to_rate_denominator * GST_SECOND, self->to_rate_numerator)
```

Definition at line 68 of file `gsttensor_rate.c`.

9.46.2.9 `MAGIC_LIMIT`

```
#define MAGIC_LIMIT 25
```

Definition at line 773 of file `gsttensor_rate.c`.

9.46.2.10 `THROTTLE_DELAY_RATIO`

```
#define THROTTLE_DELAY_RATIO (0.999)
```

Definition at line 449 of file `gsttensor_rate.c`.

9.46.3 Enumeration Type Documentation

9.46.3.1 anonymous enum

anonymous enum

`tensor_rate` properties

Enumerator

PROP_0	
PROP_IN	
PROP_OUT	
PROP_DUP	
PROP_DROP	
PROP_SILENT	
PROP_THROTTLE	
PROP_FRAMERATE	

Definition at line 79 of file gsttensor_rate.c.

9.46.4 Function Documentation

9.46.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorRate ,
    gst_tensor_rate ,
    GST_TYPE_BASE_TRANSFORM )
```

9.46.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_rate_debug )
```

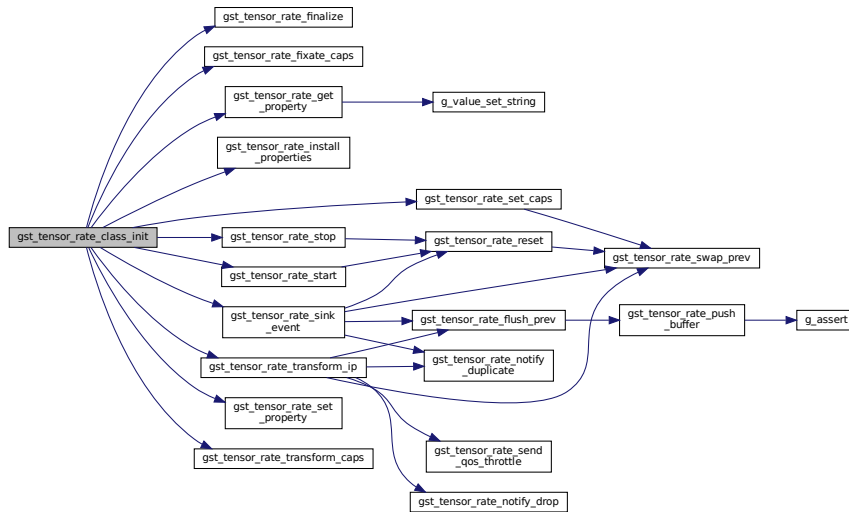
9.46.4.3 gst_tensor_rate_class_init()

```
static void gst_tensor_rate_class_init (
    GstTensorRateClass * klass ) [static]
```

initialize the tensor_rate's class (GST Standard)

Definition at line 148 of file gsttensor_rate.c.

Here is the call graph for this function:



9.46.4.4 `gst_tensor_rate_finalize()`

```
static void gst_tensor_rate_finalize (
    GObject * object ) [static]
```

Function to finalize instance. (GST Standard)

Definition at line 341 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.5 `gst_tensor_rate_fixate_caps()`

```
static GstCaps * gst_tensor_rate_fixate_caps (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    GstCaps * othercaps ) [static]
```

fixate caps. required vmethod of GstBaseTransform.

Definition at line 680 of file gsttensor_rate.c.

Here is the caller graph for this function:



9.46.4.6 `gst_tensor_rate_flush_prev()`

```
static GstFlowReturn gst_tensor_rate_flush_prev (
    GstTensorRate * self,
    gboolean duplicate,
    GstClockTime next_intime ) [static]
```

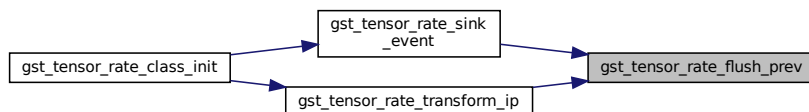
flush the oldest buffer

Definition at line 259 of file gsttensor_rate.c.

Here is the call graph for this function:



Here is the caller graph for this function:



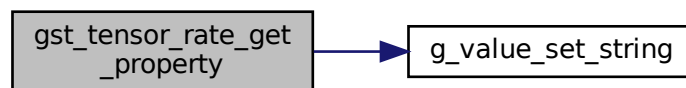
9.46.4.7 `gst_tensor_rate_get_property()`

```
static void gst_tensor_rate_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

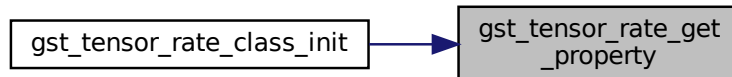
Getter for `tensor_rate` properties.

Definition at line 407 of file `gsttensor_rate.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



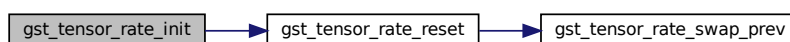
9.46.4.8 `gst_tensor_rate_init()`

```
static void gst_tensor_rate_init (
    GstTensorRate * self ) [static]
```

initialize the new element (GST Standard)

Definition at line 317 of file `gsttensor_rate.c`.

Here is the call graph for this function:



9.46.4.9 `gst_tensor_rate_install_properties()`

```
static void gst_tensor_rate_install_properties (
    GObjectClass * object_class ) [static]
```

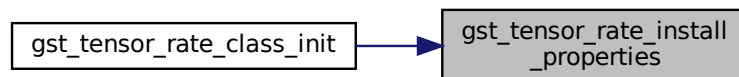
Installs all the properties for `tensor_rate`.

Parameters

in	<i>object_class</i>	Glib object class whose properties will be set
----	---------------------	--

Definition at line 953 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.10 `gst_tensor_rate_notify_drop()`

```
static void gst_tensor_rate_notify_drop (
    GstTensorRate * self ) [static]
```

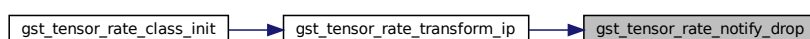
notify a frame drop event

Parameters

in	<i>self</i>	"this" pointer
----	-------------	----------------

Definition at line 758 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.11 `gst_tensor_rate_notify_duplicate()`

```
static void gst_tensor_rate_notify_duplicate (
    GstTensorRate * self ) [static]
```

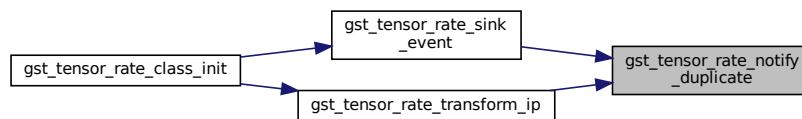
notify a frame duplicate event

Parameters

in	<i>self</i>	"this" pointer
----	-------------	----------------

Definition at line 768 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.12 `gst_tensor_rate_push_buffer()`

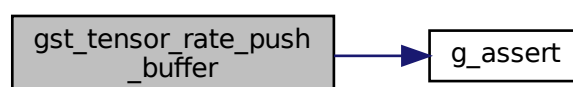
```
static GstFlowReturn gst_tensor_rate_push_buffer (
    GstTensorRate * self,
    GstBuffer * outbuf,
    gboolean duplicate,
    GstClockTime next_intime ) [static]
```

push the buffer to src pad

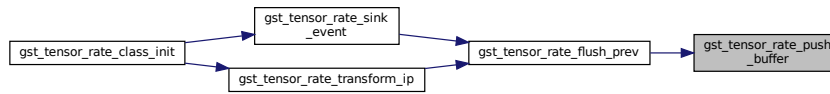
There must always be a valid duration on `prevbuf` if `rate > 0`, it is ensured in the `transform_ip` function

Definition at line 202 of file `gsttensor_rate.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



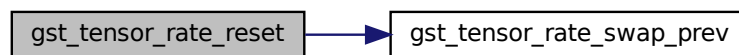
9.46.4.13 gst_tensor_rate_reset()

```
static void gst_tensor_rate_reset (
    GstTensorRate * self ) [static]
```

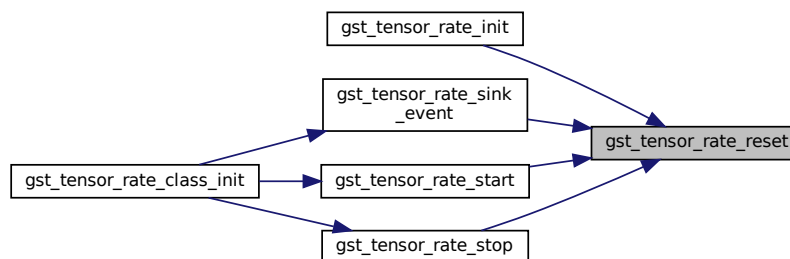
reset variables of the element (GST Standard)

Definition at line 295 of file gsttensor_rate.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.46.4.14 `gst_tensor_rate_send_qos_throttle()`

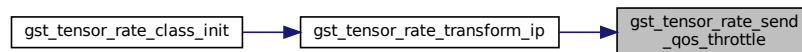
```
static void gst_tensor_rate_send_qos_throttle (
    GstTensorRate * self,
    GstClockTime timestamp ) [static]
```

send throttling qos event to upstream elements

unused

Definition at line 455 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.15 `gst_tensor_rate_set_caps()`

```
static gboolean gst_tensor_rate_set_caps (
    GstBaseTransform * trans,
    GstCaps * incaps,
    GstCaps * outcaps ) [static]
```

set caps. required vmethod of `GstBaseTransform`.

After a `setcaps`, our caps may have changed. In that case, we can't use the old buffer, if there was one (it might have different dimensions)

Definition at line 705 of file `gsttensor_rate.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



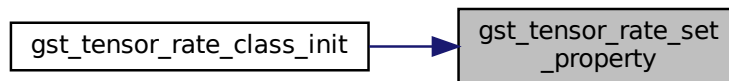
9.46.4.16 `gst_tensor_rate_set_property()`

```
static void gst_tensor_rate_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

Setter for `tensor_rate` properties.

Definition at line 350 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.17 `gst_tensor_rate_sink_event()`

```
static gboolean gst_tensor_rate_sink_event (
    GstBaseTransform * trans,
    GstEvent * event ) [static]
```

Event handler for sink pad of tensor rate.

Parameters

in	<i>trans</i>	"this" pointer
in	<i>event</i>	a passed event object

Returns

TRUE if there is no error.

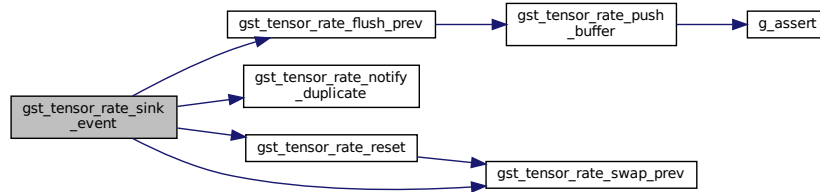
fill up to the end of current segment, or only send out the stored buffer if there is no specific stop. regardless, prevent going loopy in strange cases

fill up to the end of current segment, or only send out the stored buffer if there is no specific stop. regardless, prevent going loopy in strange cases

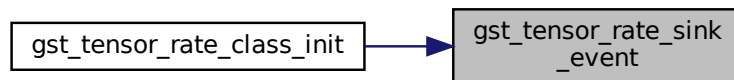
Output at least one frame but if the buffer duration is valid, output enough frames to use the complete buffer duration

Definition at line 781 of file `gsttensor_rate.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.46.4.18 gst_tensor_rate_start()

```

static gboolean gst_tensor_rate_start (
    GstBaseTransform * trans ) [static]
  
```

Called when the element starts processing. optional vmethod of BaseTransform.

Parameters

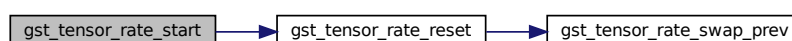
in	<i>trans</i>	"this" pointer
----	--------------	----------------

Returns

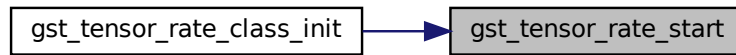
TRUE if there is no error.

Definition at line 928 of file gsttensor_rate.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.46.4.19 gst_tensor_rate_stop()

```

static gboolean gst_tensor_rate_stop (
    GstBaseTransform * trans ) [static]
  
```

Called when the element stops processing. optional vmethod of BaseTransform.

Parameters

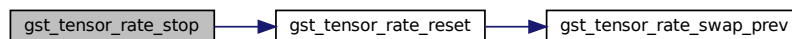
in	<i>trans</i>	"this" pointer
----	--------------	----------------

Returns

TRUE if there is no error.

Definition at line 941 of file gsttensor_rate.c.

Here is the call graph for this function:



Here is the caller graph for this function:



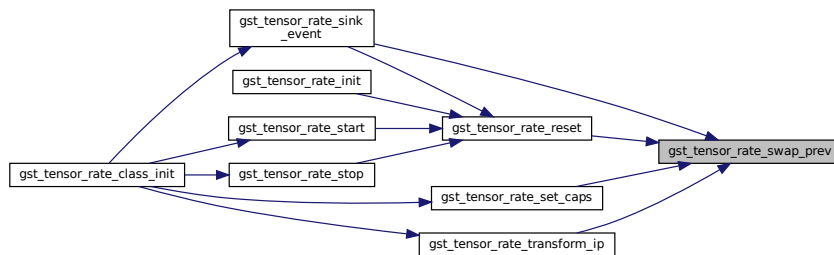
9.46.4.20 `gst_tensor_rate_swap_prev()`

```
static void gst_tensor_rate_swap_prev (
    GstTensorRate * self,
    GstBuffer * buffer,
    gint64 time ) [static]
```

swap a previous buffer

Definition at line 280 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.21 `gst_tensor_rate_transform_caps()`

```
static GstCaps * gst_tensor_rate_transform_caps (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    GstCaps * filter ) [static]
```

configure tensor-srcpad cap from "proposed" cap. (GST Standard)

@trans ("this" pointer) @direction (why do we need this?) @caps sinkpad cap (if direction GST_PAD_SINK) @filter this element's cap (don't know specifically.)

Be careful not to fix/set caps at this stage. Negotiation not completed yet.

Definition at line 634 of file `gsttensor_rate.c`.

Here is the caller graph for this function:



9.46.4.22 `gst_tensor_rate_transform_ip()`

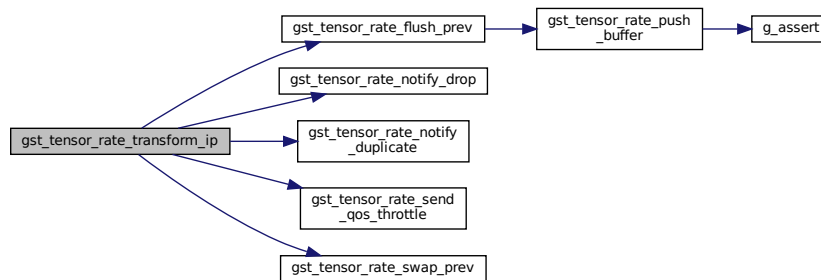
```
static GstFlowReturn gst_tensor_rate_transform_ip (
    GstBaseTransform * trans,
    GstBuffer * buffer ) [static]
```

in-place transform

continue while the first one was the best, if they were equal avoid going into an infinite loop

Definition at line 477 of file `gsttensor_rate.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.46.5 Variable Documentation

9.46.5.1 `pspec_drop`

```
GParamSpec* pspec_drop = NULL [static]
```

Definition at line 107 of file `gsttensor_rate.c`.

9.46.5.2 pspec_duplicate

```
GParamSpec* pspec_duplicate = NULL [static]
```

Definition at line 108 of file gsttensor_rate.c.

9.46.5.3 sink_factory

```
GstStaticPadTemplate sink_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the inputs.

Definition at line 94 of file gsttensor_rate.c.

9.46.5.4 src_factory

```
GstStaticPadTemplate src_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the outputs.

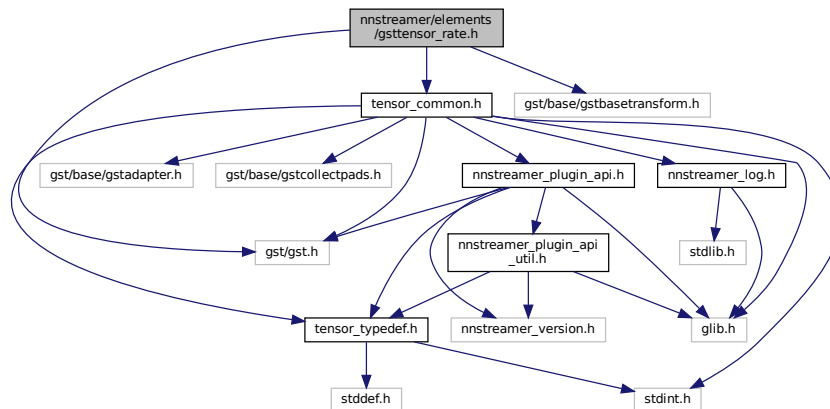
Definition at line 102 of file gsttensor_rate.c.

9.47 nnstreamer/elements/gsttensor_rate.h File Reference

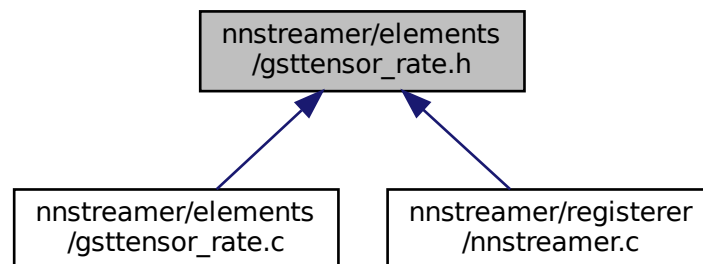
GStreamer plugin to adjust tensor rate.

```
#include <gst/gst.h>  
#include <gst/base/gstbasetransform.h>
```

```
#include <tensor_common.h>
Include dependency graph for gsttensor_rate.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorRate](#)
Tensor Rate data structure.
- struct [_GstTensorRateClass](#)
GstTensorRateClass inherits GstElementClass.

Macros

- `#define GST_TYPE_TENSOR_RATE (gst_tensor_rate_get_type ())`
- `#define GST_TENSOR_RATE(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_TENSOR_RATE, GstTensorRate))`
- `#define GST_TENSOR_RATE_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_TENSOR_RATE, GstTensorRateClass))`

- #define `GST_TENSOR_RATE_GET_CLASS(obj)` (`G_TYPE_INSTANCE_GET_CLASS((obj), GST_TYPE_TENSOR_RATE, GstTensorRateClass)`)
- #define `GST_IS_TENSOR_RATE(obj)` (`G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_RATE)`)
- #define `GST_IS_TENSOR_RATE_CLASS(klass)` (`G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_RATE)`)
- #define `GST_TENSOR_RATE_CAST(obj)` (`((GstTensorRate*)(obj))`)

Typedefs

- typedef struct `_GstTensorRate` `GstTensorRate`
- typedef struct `_GstTensorRateClass` `GstTensorRateClass`

Functions

- GType `gst_tensor_rate_get_type` (void)
Get Type function required for gst elements.

9.47.1 Detailed Description

GStreamer plugin to adjust tensor rate.

GStreamer/NNStreamer Tensor-Rate Copyright (C) 2020 Dongju Chae dongju.chae@samsung.com

Date

24 Sep 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Dongju Chae dongju.chae@samsung.com

Bug No known bugs except for NYI items

9.47.2 Macro Definition Documentation

9.47.2.1 GST_IS_TENSOR_RATE

```
#define GST_IS_TENSOR_RATE(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_RATE))
```

Definition at line 28 of file `gsttensor_rate.h`.

9.47.2.2 GST_IS_TENSOR_RATE_CLASS

```
#define GST_IS_TENSOR_RATE_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_RATE))
```

Definition at line 29 of file gsttensor_rate.h.

9.47.2.3 GST_TENSOR_RATE

```
#define GST_TENSOR_RATE(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_TENSOR_RATE, GstTensorRate))
```

Definition at line 25 of file gsttensor_rate.h.

9.47.2.4 GST_TENSOR_RATE_CAST

```
#define GST_TENSOR_RATE_CAST(  
    obj ) ((GstTensorRate*)(obj))
```

Definition at line 30 of file gsttensor_rate.h.

9.47.2.5 GST_TENSOR_RATE_CLASS

```
#define GST_TENSOR_RATE_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_TENSOR_RATE, GstTensorRateClass))
```

Definition at line 26 of file gsttensor_rate.h.

9.47.2.6 GST_TENSOR_RATE_GET_CLASS

```
#define GST_TENSOR_RATE_GET_CLASS(  
    obj ) (G_TYPE_INSTANCE_GET_CLASS ((obj), GST_TYPE_TENSOR_RATE, GstTensorRateClass))
```

Definition at line 27 of file gsttensor_rate.h.

9.47.2.7 GST_TYPE_TENSOR_RATE

```
#define GST_TYPE_TENSOR_RATE (gst_tensor_rate_get_type ())
```

Definition at line 24 of file gsttensor_rate.h.

9.47.3 Typedef Documentation

9.47.3.1 GstTensorRate

```
typedef struct _GstTensorRate GstTensorRate
```

Definition at line 31 of file gsttensor_rate.h.

9.47.3.2 GstTensorRateClass

```
typedef struct _GstTensorRateClass GstTensorRateClass
```

Definition at line 32 of file gsttensor_rate.h.

9.47.4 Function Documentation

9.47.4.1 gst_tensor_rate_get_type()

```
GType gst_tensor_rate_get_type (
    void )
```

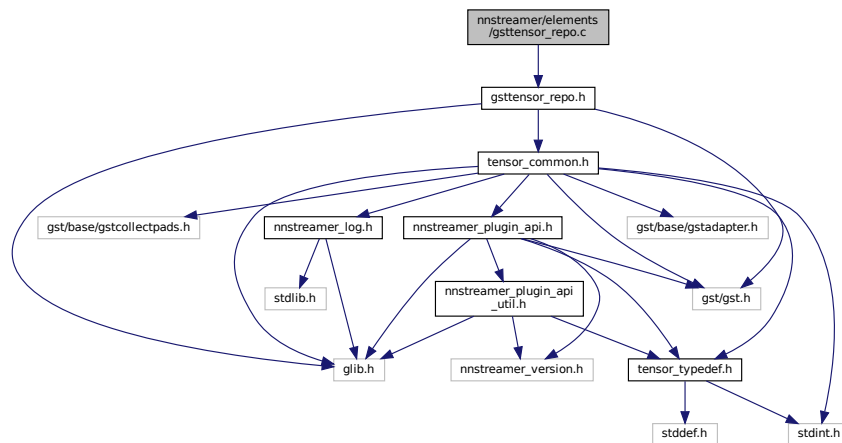
Get Type function required for gst elements.

9.48 nnstreamer/elements/gsttensor_repo.c File Reference

tensor repo file for NNStreamer, the GStreamer plugin for neural networks

```
#include "gsttensor_repo.h"
```

Include dependency graph for gsttensor_repo.c:



Macros

- #define `DBG FALSE`
- #define `GST_REPO_LOCK()` (`g_mutex_lock(&_repo.repo_lock)`)
Macro for Lock & Cond.
- #define `GST_REPO_UNLOCK()` (`g_mutex_unlock(&_repo.repo_lock)`)
- #define `GST_REPO_WAIT()` (`g_cond_wait(&_repo.repo_cond, &_repo.repo_lock)`)
- #define `GST_REPO_BROADCAST()` (`g_cond_broadcast (&_repo.repo_cond)`)

Functions

- static void `gst_tensor_repo_release_repodata` (gpointer `data`)
Internal function to release repo data.
- `GstTensorRepoData *` `gst_tensor_repo_get_repodata` (guint `nth`)
Getter to get nth GstTensorRepoData.
- gboolean `gst_tensor_repo_set_changed` (guint `o_nth`, guint `nth`, gboolean `is_sink`)
Set the changing status of repo.
- gboolean `gst_tensor_repo_add_repodata` (guint `nth`, gboolean `is_sink`)
Add GstTensorRepoData into repo.
- gboolean `gst_tensor_repo_set_buffer` (guint `nth`, `GstBuffer *``buffer`, `GstCaps *``caps`)
Push GstBuffer into repo.
- gboolean `gst_tensor_repo_check_eos` (guint `nth`)
Check EOS (End-of-Stream) of slot.
- gboolean `gst_tensor_repo_check_changed` (guint `nth`, guint `*newid`, gboolean `is_sink`)
Check repo data is changed.
- gboolean `gst_tensor_repo_set_eos` (guint `nth`)
Set EOS (End-of-Stream) of slot.
- `GstBuffer *` `gst_tensor_repo_get_buffer` (guint `nth`, gboolean `*eos`, guint `*newid`, `GstCaps **caps`)
Get GstTensorRepoData from repo.
- gboolean `gst_tensor_repo_remove_repodata` (guint `nth`)
Remove nth GstTensorRepoData from GstTensorRepo.
- void `gst_tensor_repo_init` (void)
GstTensorRepo initialization.
- gboolean `gst_tensor_repo_wait` (void)
Wait for finish of initialization.

Variables

- static `GstTensorRepo _repo` = {`.num_data` = 0,`.initialized` = `FALSE` }
tensor repo global variable with init.

9.48.1 Detailed Description

tensor repo file for NNStreamer, the GStreamer plugin for neural networks

NNStreamer Tensor Repo Header's Contents Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

17 Nov 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.48.2 Macro Definition Documentation

9.48.2.1 DBG

```
#define DBG FALSE
```

Definition at line 29 of file gsttensor_repo.c.

9.48.2.2 GST_REPO_BROADCAST

```
#define GST_REPO_BROADCAST( ) (g_cond_broadcast (&_repo.repo_cond))
```

Definition at line 43 of file gsttensor_repo.c.

9.48.2.3 GST_REPO_LOCK

```
#define GST_REPO_LOCK( ) (g_mutex_lock(&_repo.repo_lock))
```

Macro for Lock & Cond.

Definition at line 40 of file gsttensor_repo.c.

9.48.2.4 GST_REPO_UNLOCK

```
#define GST_REPO_UNLOCK( ) (g_mutex_unlock(&_repo.repo_lock))
```

Definition at line 41 of file gsttensor_repo.c.

9.48.2.5 GST_REPO_WAIT

```
#define GST_REPO_WAIT( ) (g_cond_wait(&_repo.repo_cond, &_repo.repo_lock))
```

Definition at line 42 of file gsttensor_repo.c.

9.48.3 Function Documentation

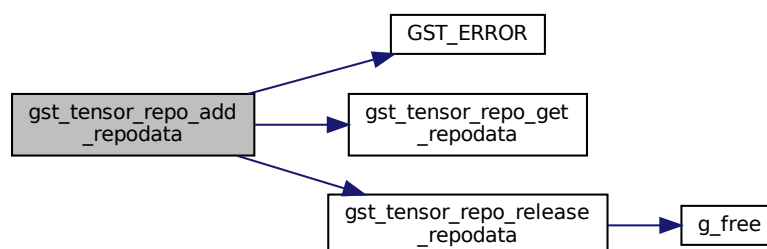
9.48.3.1 gst_tensor_repo_add_repodata()

```
gboolean gst_tensor_repo_add_repodata (  
    guint nth,  
    gboolean is_sink )
```

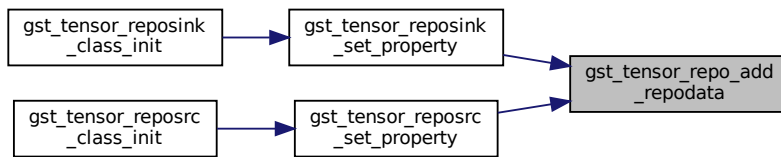
Add [GstTensorRepoData](#) into repo.

Definition at line 129 of file gsttensor_repo.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.48.3.2 `gst_tensor_repo_check_changed()`

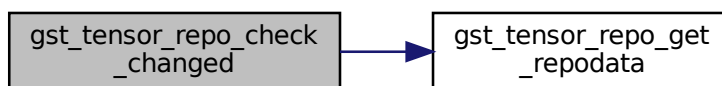
```

gboolean gst_tensor_repo_check_changed (
    guint nth,
    guint * newid,
    gboolean is_sink )
  
```

Check repo data is changed.

Definition at line 255 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



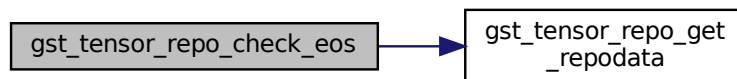
9.48.3.3 `gst_tensor_repo_check_eos()`

```
gboolean gst_tensor_repo_check_eos (
    guint nth )
```

Check EOS (End-of-Stream) of slot.

Definition at line 236 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



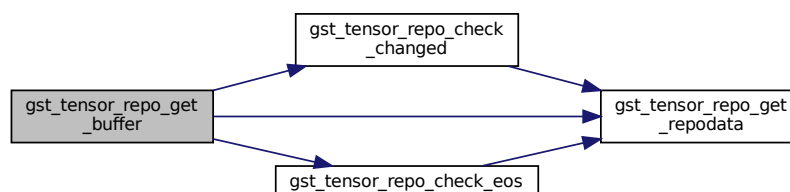
9.48.3.4 `gst_tensor_repo_get_buffer()`

```
GstBuffer* gst_tensor_repo_get_buffer (
    guint nth,
    gboolean * eos,
    guint * newid,
    GstCaps ** caps )
```

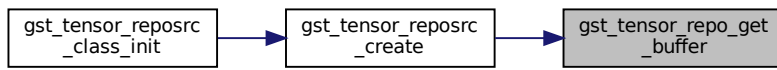
Get [GstTensorRepoData](#) from `repo`.

Definition at line 309 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



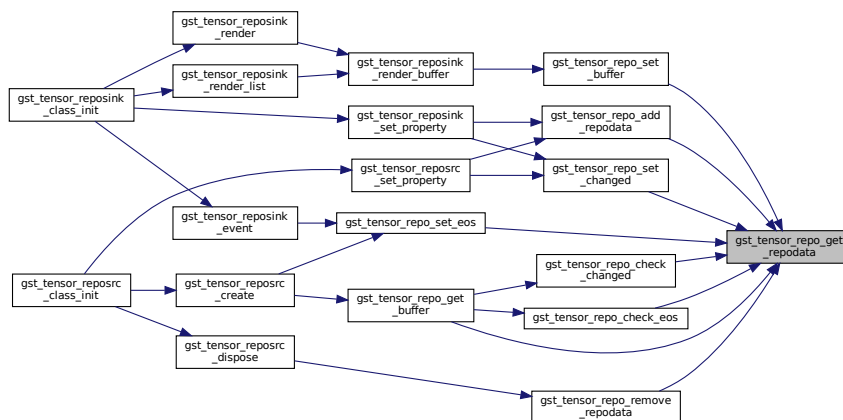
9.48.3.5 `gst_tensor_repo_get_repodata()`

```
GstTensorRepoData* gst_tensor_repo_get_repodata (
    guint nth )
```

Getter to get nth [GstTensorRepoData](#).

Definition at line 74 of file `gsttensor_repo.c`.

Here is the caller graph for this function:



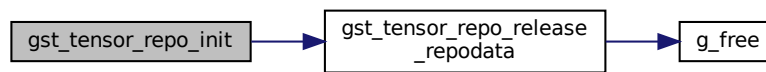
9.48.3.6 `gst_tensor_repo_init()`

```
void gst_tensor_repo_init (
    void )
```

[GstTensorRepo](#) initialization.

Definition at line 386 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



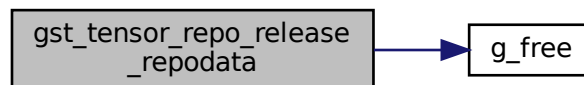
9.48.3.7 `gst_tensor_repo_release_repdata()`

```
static void gst_tensor_repo_release_repdata (
    gpointer data ) [static]
```

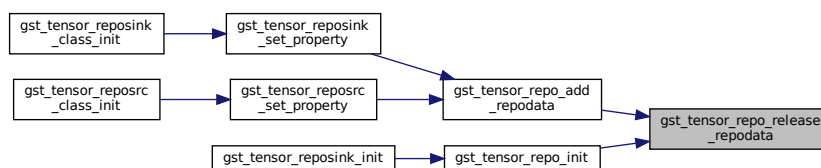
Internal function to release repo data.

Definition at line 49 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.48.3.8 `gst_tensor_repo_remove_repodata()`

```
gboolean gst_tensor_repo_remove_repodata (
    guint nth )
```

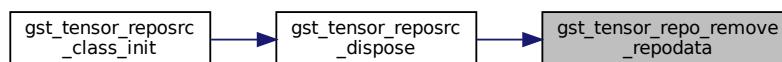
Remove nth [GstTensorRepoData](#) from [GstTensorRepo](#).

Definition at line 357 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



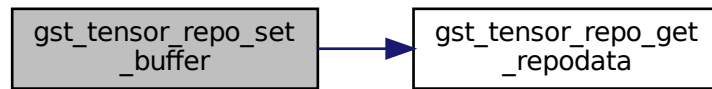
9.48.3.9 `gst_tensor_repo_set_buffer()`

```
gboolean gst_tensor_repo_set_buffer (
    guint nth,
    GstBuffer * buffer,
    GstCaps * caps )
```

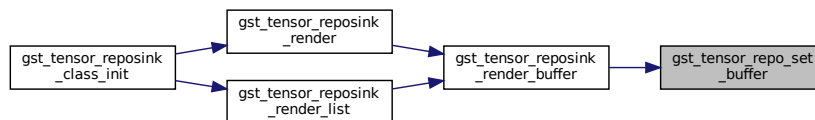
Push `GstBuffer` into repo.

Definition at line 193 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.48.3.10 `gst_tensor_repo_set_changed()`

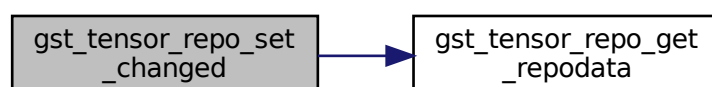
```

gboolean gst_tensor_repo_set_changed (
    guint o_nth,
    guint nth,
    gboolean is_sink )
  
```

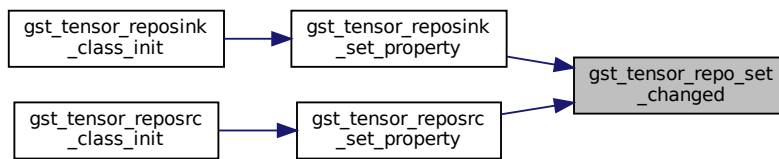
Set the changing status of repo.

Definition at line 91 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



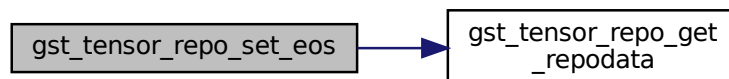
9.48.3.11 `gst_tensor_repo_set_eos()`

```
gboolean gst_tensor_repo_set_eos (
    guint nth )
```

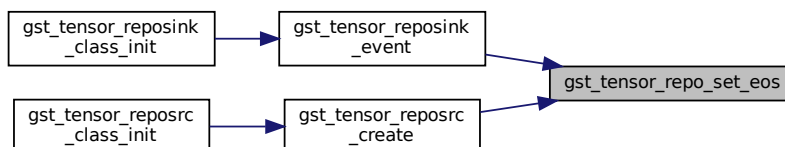
Set EOS (End-of-Stream) of slot.

Definition at line 287 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.48.3.12 gst_tensor_repo_wait()

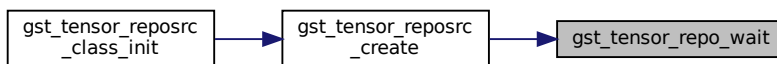
```
gboolean gst_tensor_repo_wait (
    void )
```

Wait for finish of initialization.

Wait for the repo initialization.

Definition at line 406 of file gsttensor_repo.c.

Here is the caller graph for this function:



9.48.4 Variable Documentation

9.48.4.1 _repo

```
GstTensorRepo _repo = { .num_data = 0, .initialized = FALSE } [static]
```

tensor repo global variable with init.

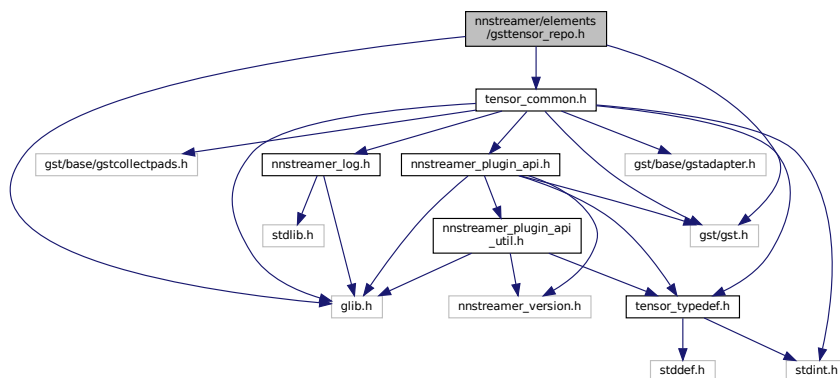
Definition at line 35 of file gsttensor_repo.c.

9.49 nnstreamer/elements/gsttensor_repo.h File Reference

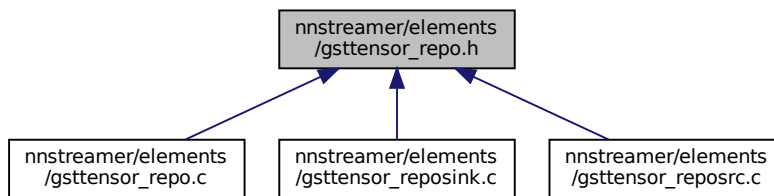
tensor repo header file for NNStreamer, the GStreamer plugin for neural networks

```
#include <glib.h>
#include <gst/gst.h>
#include "tensor_common.h"
```

Include dependency graph for gsttensor_repo.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [GstTensorRepoData](#)
GstTensorRepo internal data structure.
- struct [GstTensorRepo](#)
GstTensorRepo data structure.

Functions

- [GstTensorRepoData *](#) [gst_tensor_repo_get_repodata](#) (guint nth)
Getter to get nth GstTensorRepoData.
- gboolean [gst_tensor_repo_add_repodata](#) (guint myid, gboolean is_sink)
Add GstTensorRepoData into repo.
- gboolean [gst_tensor_repo_set_buffer](#) (guint nth, GstBuffer *buffer, GstCaps *caps)
Push GstBuffer into repo.
- gboolean [gst_tensor_repo_check_eos](#) (guint nth)
Check EOS (End-of-Stream) of slot.
- gboolean [gst_tensor_repo_set_eos](#) (guint nth)
Set EOS (End-of-Stream) of slot.
- gboolean [gst_tensor_repo_set_changed](#) (guint o_nth, guint nth, gboolean is_sink)
Set the changing status of repo.
- GstBuffer * [gst_tensor_repo_get_buffer](#) (guint nth, gboolean *eos, guint *newid, GstCaps **caps)
Get GstTensorRepoData from repo.
- gboolean [gst_tensor_repo_check_changed](#) (guint nth, guint *newid, gboolean is_sink)
Check repo data is changed.
- gboolean [gst_tensor_repo_remove_repodata](#) (guint nth)
Remove nth GstTensorRepoData from GstTensorRepo.
- void [gst_tensor_repo_init](#) (void)
GstTensorRepo initialization.
- gboolean [gst_tensor_repo_wait](#) (void)
Wait for the repo initialization.

9.49.1 Detailed Description

tensor repo header file for NNStreamer, the GStreamer plugin for neural networks

NNStreamer Tensor Repo Header Copyright (C) 2018 Jijoong Moon ji joong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

17 Nov 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon ji joong.moon@samsung.com

Bug No known bugs except for NYI items

9.49.2 Function Documentation

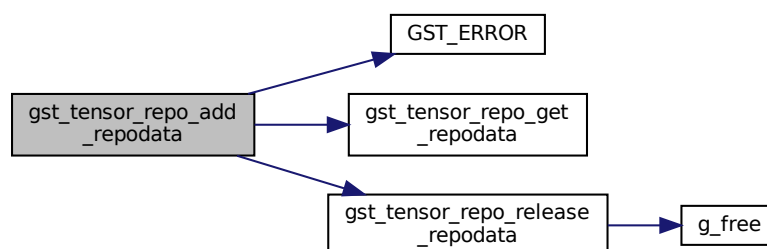
9.49.2.1 `gst_tensor_repo_add_repodata()`

```
gboolean gst_tensor_repo_add_repodata (  
    guint myid,  
    gboolean is_sink )
```

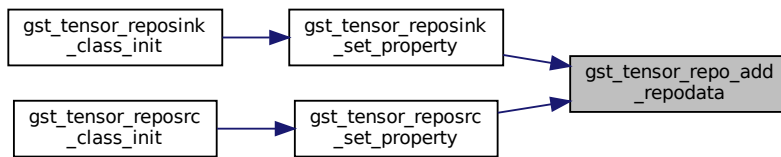
Add `GstTensorRepoData` into repo.

Definition at line 129 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.49.2.2 `gst_tensor_repo_check_changed()`

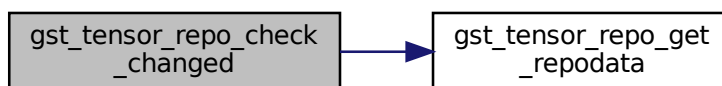
```

gboolean gst_tensor_repo_check_changed (
    guint nth,
    guint * newid,
    gboolean is_sink )
  
```

Check repo data is changed.

Definition at line 255 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



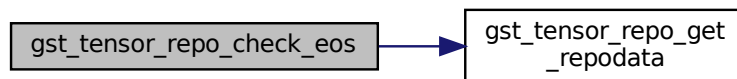
9.49.2.3 `gst_tensor_repo_check_eos()`

```
gboolean gst_tensor_repo_check_eos (
    guint nth )
```

Check EOS (End-of-Stream) of slot.

Definition at line 236 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



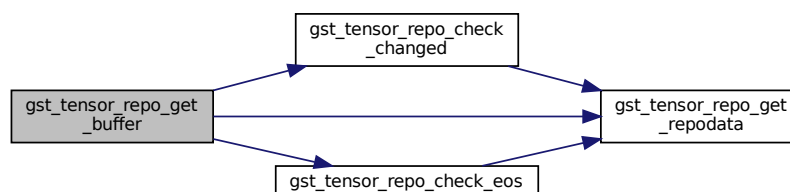
9.49.2.4 `gst_tensor_repo_get_buffer()`

```
GstBuffer* gst_tensor_repo_get_buffer (
    guint nth,
    gboolean * eos,
    guint * newid,
    GstCaps ** caps )
```

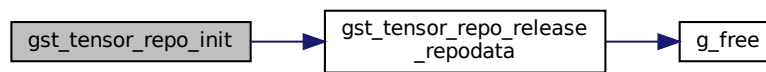
Get [GstTensorRepoData](#) from `repo`.

Definition at line 309 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the call graph for this function:



Here is the caller graph for this function:



9.49.2.7 `gst_tensor_repo_remove_repdata()`

```
gboolean gst_tensor_repo_remove_repdata (
    guint nth )
```

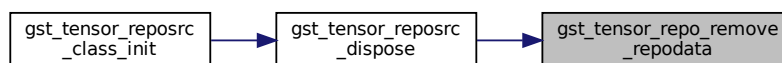
Remove `nth` [GstTensorRepoData](#) from [GstTensorRepo](#).

Definition at line 357 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



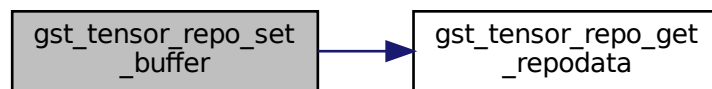
9.49.2.8 `gst_tensor_repo_set_buffer()`

```
gboolean gst_tensor_repo_set_buffer (
    guint nth,
    GstBuffer * buffer,
    GstCaps * caps )
```

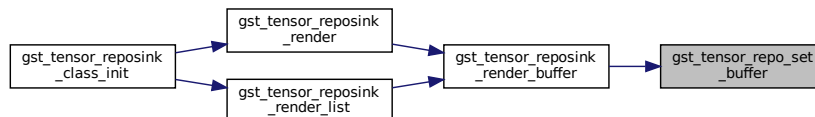
Push GstBuffer into repo.

Definition at line 193 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



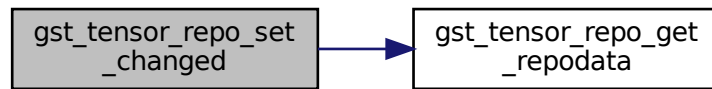
9.49.2.9 `gst_tensor_repo_set_changed()`

```
gboolean gst_tensor_repo_set_changed (
    guint o_nth,
    guint nth,
    gboolean is_sink )
```

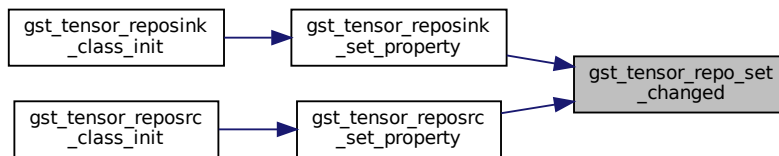
Set the changing status of repo.

Definition at line 91 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.49.2.10 `gst_tensor_repo_set_eos()`

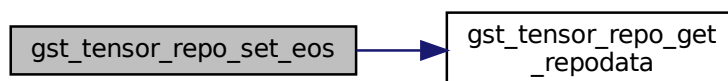
```

gboolean gst_tensor_repo_set_eos (
    guint nth )
  
```

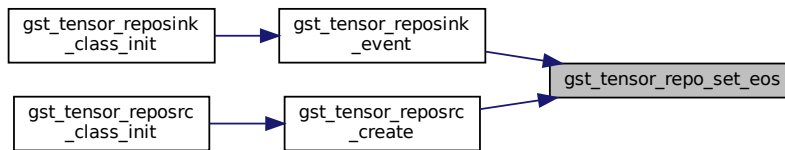
Set EOS (End-of-Stream) of slot.

Definition at line 287 of file `gsttensor_repo.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.49.2.11 gst_tensor_repo_wait()

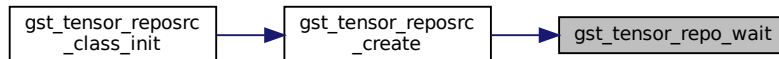
```
gboolean gst_tensor_repo_wait (
    void )
```

Wait for the repo initialization.

Wait for the repo initialization.

Definition at line 406 of file gsttensor_repo.c.

Here is the caller graph for this function:

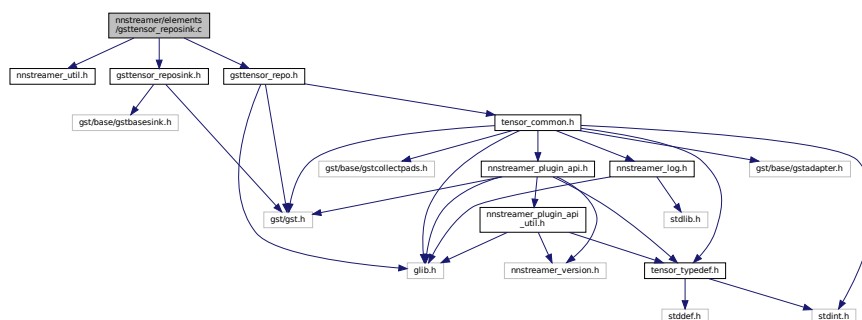


9.50 nnstreamer/elements/gsttensor_reposink.c File Reference

GStreamer plugin to handle tensor repository.

```
#include <nnstreamer_util.h>
#include "gsttensor_repo.h"
#include "gsttensor_reposink.h"
```

Include dependency graph for gsttensor_reposink.c:



Macros

- #define [GST_CAT_DEFAULT](#) `gst_tensor_reposink_debug`
- #define [DEFAULT_SIGNAL_RATE](#) `0`
- #define [DEFAULT_SILENT](#) `TRUE`
- #define [DEFAULT_QOS](#) `TRUE`
- #define [DEFAULT_INDEX](#) `0`
- #define [gst_tensor_reposink_parent_class](#) `parent_class`

Enumerations

- enum { [PROP_0](#), [PROP_SIGNAL_RATE](#), [PROP_SLOT](#), [PROP_SILENT](#) }
tensor_reposink properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) (`gst_tensor_reposink_debug`)
- static void [gst_tensor_reposink_set_property](#) (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
set property vmethod
- static void [gst_tensor_reposink_get_property](#) (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)
get property vmethod
- static void [gst_tensor_reposink_dispose](#) (`GObject *object`)
dispose vmethod implementation
- static gboolean [gst_tensor_reposink_start](#) (`GstBaseSink *sink`)
start vmethod implementation
- static gboolean [gst_tensor_reposink_stop](#) (`GstBaseSink *sink`)
stop vmethod implementation
- static gboolean [gst_tensor_reposink_event](#) (`GstBaseSink *sink`, `GstEvent *event`)
Handle events.
- static gboolean [gst_tensor_reposink_query](#) (`GstBaseSink *sink`, `GstQuery *query`)
query vmethod implementation
- static `GstFlowReturn` [gst_tensor_reposink_render](#) (`GstBaseSink *sink`, `GstBuffer *buffer`)
render vmethod implementation
- static `GstFlowReturn` [gst_tensor_reposink_render_list](#) (`GstBaseSink *sink`, `GstBufferList *buffer_list`)
render list vmethod implementation
- static gboolean [gst_tensor_reposink_set_caps](#) (`GstBaseSink *sink`, `GstCaps *caps`)
set_caps vmethod implementation
- static `GstCaps *` [gst_tensor_reposink_get_caps](#) (`GstBaseSink *sink`, `GstCaps *filter`)
get_caps vmethod implementation
- [G_DEFINE_TYPE](#) (`GstTensorRepoSink`, `gst_tensor_reposink`, `GST_TYPE_BASE_SINK`)
- static void [gst_tensor_reposink_class_init](#) (`GstTensorRepoSinkClass *klass`)
class initialization of tensor_reposink
- static void [gst_tensor_reposink_init](#) (`GstTensorRepoSink *self`)
initialization of tensor_reposink
- static gboolean [gst_tensor_reposink_render_buffer](#) (`GstTensorRepoSink *self`, `GstBuffer *buffer`)
Push GstBuffer.

9.50.1 Detailed Description

GStreamer plugin to handle tensor repository.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details. SECTION: element-tensor_reposink

Set element to handle tensor repo

Date

19 Nov 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.50.2 Macro Definition Documentation

9.50.2.1 DEFAULT_INDEX

```
#define DEFAULT_INDEX 0
```

Definition at line 56 of file gsttensor_reposink.c.

9.50.2.2 DEFAULT_QOS

```
#define DEFAULT_QOS TRUE
```

Definition at line 55 of file gsttensor_reposink.c.

9.50.2.3 DEFAULT_SIGNAL_RATE

```
#define DEFAULT_SIGNAL_RATE 0
```

Definition at line 53 of file gsttensor_reposink.c.

9.50.2.4 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Definition at line 54 of file gsttensor_reposink.c.

9.50.2.5 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_reposink_debug
```

Definition at line 40 of file gsttensor_reposink.c.

9.50.2.6 gst_tensor_reposink_parent_class

```
#define gst_tensor_reposink_parent_class parent_class
```

Definition at line 79 of file gsttensor_reposink.c.

9.50.3 Enumeration Type Documentation

9.50.3.1 anonymous enum

anonymous enum

tensor_reposink properties

Enumerator

PROP_0	
PROP_SIGNAL_RATE	
PROP_SLOT	
PROP_SILENT	

Definition at line 45 of file gsttensor_reposink.c.

9.50.4 Function Documentation

9.50.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorRepoSink ,
    gst_tensor_reposink ,
    GST_TYPE_BASE_SINK )
```

9.50.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_reposink_debug )
```

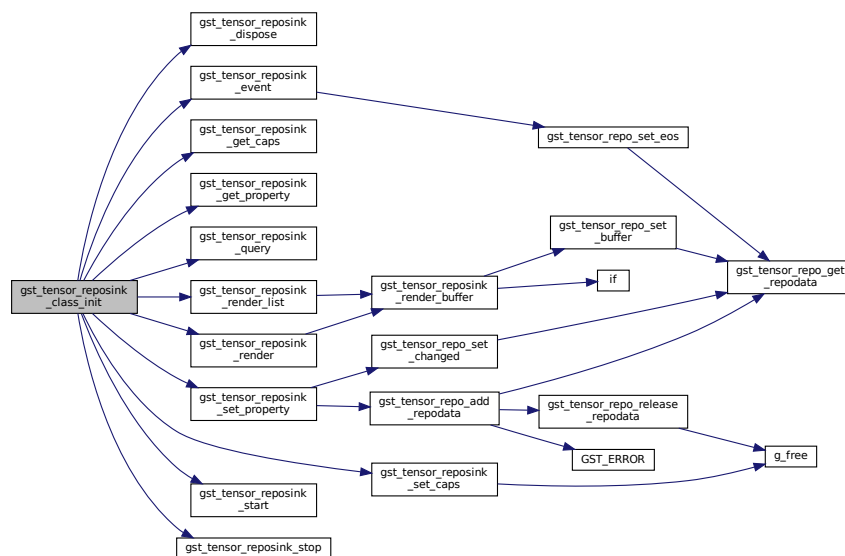
9.50.4.3 gst_tensor_reposink_class_init()

```
static void gst_tensor_reposink_class_init (
    GstTensorRepoSinkClass * klass ) [static]
```

class initialization of tensor_reposink

Definition at line 86 of file gsttensor_reposink.c.

Here is the call graph for this function:



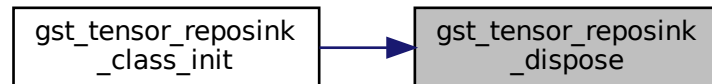
9.50.4.4 `gst_tensor_reposink_dispose()`

```
static void gst_tensor_reposink_dispose (  
    GObject * object ) [static]
```

dispose vmethod implementation

Definition at line 240 of file `gsttensor_reposink.c`.

Here is the caller graph for this function:



9.50.4.5 `gst_tensor_reposink_event()`

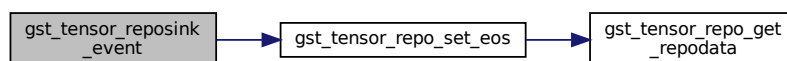
```
static gboolean gst_tensor_reposink_event (  
    GstBaseSink * sink,  
    GstEvent * event ) [static]
```

Handle events.

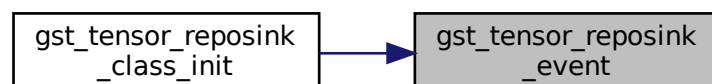
GstBaseSink method implementation.

Definition at line 278 of file `gsttensor_reposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



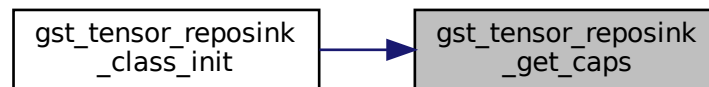
9.50.4.6 `gst_tensor_reposink_get_caps()`

```
static GstCaps * gst_tensor_reposink_get_caps (  
    GstBaseSink * sink,  
    GstCaps * filter ) [static]
```

`get_caps` vmethod implementation

Definition at line 448 of file `gsttensor_reposink.c`.

Here is the caller graph for this function:



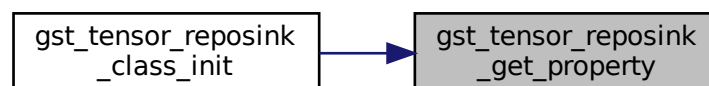
9.50.4.7 `gst_tensor_reposink_get_property()`

```
static void gst_tensor_reposink_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

`get property` vmethod

Definition at line 213 of file `gsttensor_reposink.c`.

Here is the caller graph for this function:



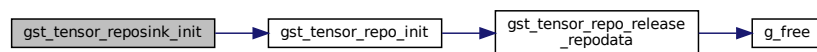
9.50.4.8 `gst_tensor_reposink_init()`

```
static void gst_tensor_reposink_init (  
    GstTensorRepoSink * self ) [static]
```

initialization of tensor_reposink

Definition at line 148 of file `gsttensor_reposink.c`.

Here is the call graph for this function:



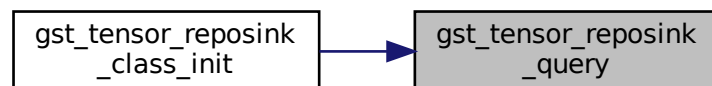
9.50.4.9 `gst_tensor_reposink_query()`

```
static gboolean gst_tensor_reposink_query (  
    GstBaseSink * sink,  
    GstQuery * query ) [static]
```

query vmethod implementation

Definition at line 303 of file `gsttensor_reposink.c`.

Here is the caller graph for this function:



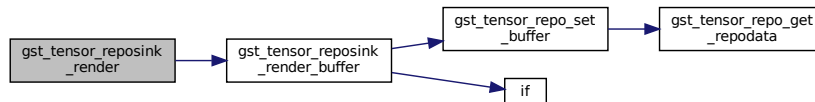
9.50.4.10 `gst_tensor_reposink_render()`

```
static GstFlowReturn gst_tensor_reposink_render (
    GstBaseSink * sink,
    GstBuffer * buffer ) [static]
```

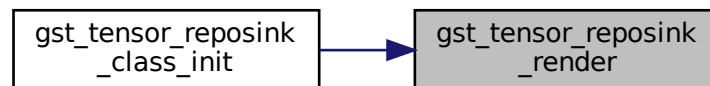
render vmethod implementation

Definition at line 377 of file `gsttensor_reposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



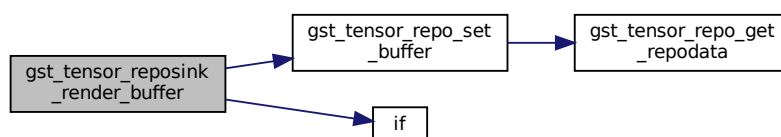
9.50.4.11 `gst_tensor_reposink_render_buffer()`

```
static gboolean gst_tensor_reposink_render_buffer (
    GstTensorRepoSink * self,
    GstBuffer * buffer ) [static]
```

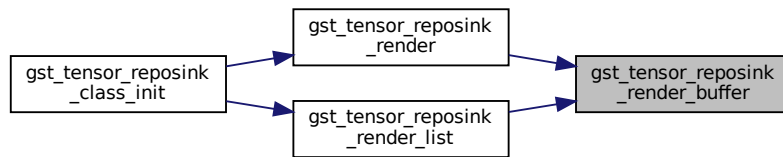
Push GstBuffer.

Definition at line 330 of file `gsttensor_reposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.50.4.12 `gst_tensor_reposink_render_list()`

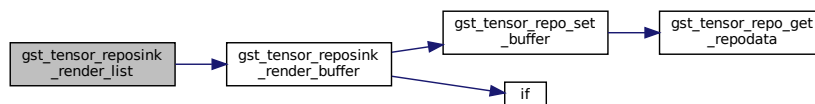
```

static GstFlowReturn gst_tensor_reposink_render_list (
    GstBaseSink * sink,
    GstBufferList * buffer_list ) [static]
  
```

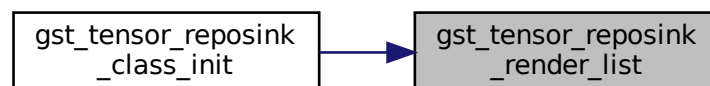
render list vmethod implementation

Definition at line 392 of file `gsttensor_reposink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



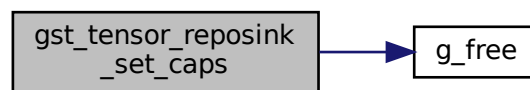
9.50.4.13 `gst_tensor_reposink_set_caps()`

```
static gboolean gst_tensor_reposink_set_caps (  
    GstBaseSink * sink,  
    GstCaps * caps ) [static]
```

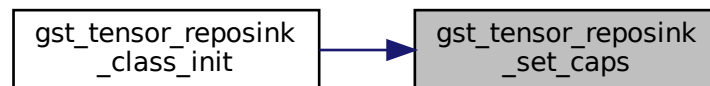
set_caps vmethod implementation

Definition at line 416 of file gsttensor_reposink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



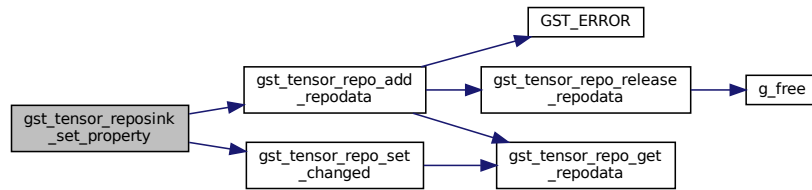
9.50.4.14 `gst_tensor_reposink_set_property()`

```
static void gst_tensor_reposink_set_property (  
    GObject * object,  
    guint prop_id,  
    const GValue * value,  
    GParamSpec * pspec ) [static]
```

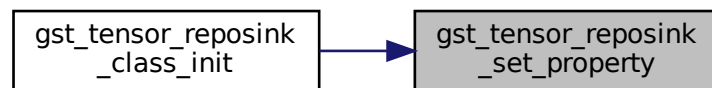
set property vmethod

Definition at line 175 of file gsttensor_reposink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



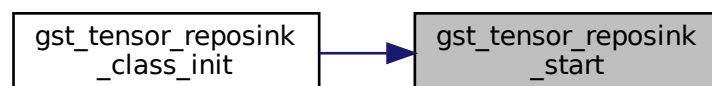
9.50.4.15 gst_tensor_reposink_start()

```
static gboolean gst_tensor_reposink_start (
    GstBaseSink * sink ) [static]
```

start vmethod implementation

Definition at line 256 of file `gsttensor_reposink.c`.

Here is the caller graph for this function:



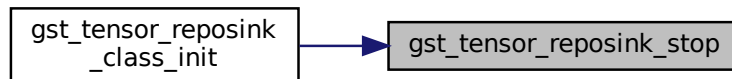
9.50.4.16 `gst_tensor_reposink_stop()`

```
static gboolean gst_tensor_reposink_stop (  
    GstBaseSink * sink ) [static]
```

stop vmethod implementation

Definition at line 266 of file `gsttensor_reposink.c`.

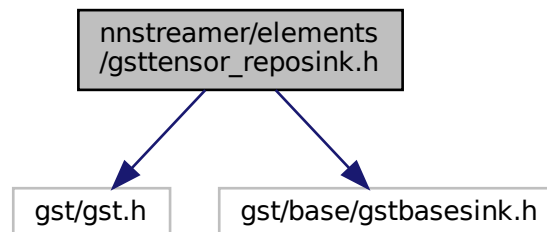
Here is the caller graph for this function:



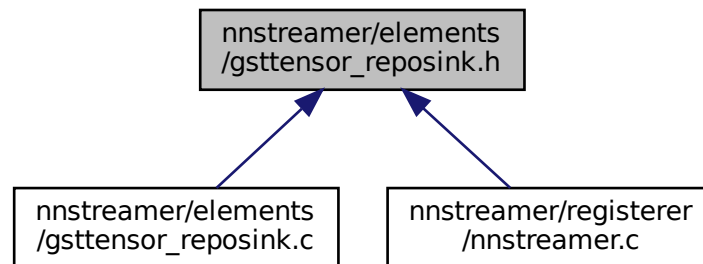
9.51 `nnstreamer/elements/gsttensor_reposink.h` File Reference

GStreamer plugin to handle tensor repository.

```
#include <gst/gst.h>  
#include <gst/base/gstbasesink.h>  
Include dependency graph for gsttensor_reposink.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorRepoSink](#)
GstTensorRepoSink data structure.
- struct [_GstTensorRepoSinkClass](#)
GstTensorRepoSinkClass data structure.

Macros

- #define [GST_TYPE_TENSOR_REPOSINK](#) ([gst_tensor_reposink_get_type\(\)](#))
- #define [GST_TENSOR_REPOSINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_REPOSINK, GstTensorRepoSink\)](#))
- #define [GST_TENSOR_REPOSINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_REPOSINK, GstTensorRepoSinkClass\)](#))
- #define [GST_IS_TENSOR_REPOSINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_REPOSINK\)](#))
- #define [GST_IS_TENSOR_REPOSINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_REPOSINK\)](#))

Typedefs

- typedef struct [_GstTensorRepoSink](#) [GstTensorRepoSink](#)
- typedef struct [_GstTensorRepoSinkClass](#) [GstTensorRepoSinkClass](#)

Functions

- GType [gst_tensor_reposink_get_type](#) (void)
Function to get type of tensor_reposink.

9.51.1 Detailed Description

GStreamer plugin to handle tensor repository.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

19 Nov 2018

See also

<https://github.com/nstreamer/nstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.51.2 Macro Definition Documentation

9.51.2.1 GST_IS_TENSOR_REPOSINK

```
#define GST_IS_TENSOR_REPOSINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_REPOSINK))
```

Definition at line 41 of file gsttensor_reposink.h.

9.51.2.2 GST_IS_TENSOR_REPOSINK_CLASS

```
#define GST_IS_TENSOR_REPOSINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_REPOSINK))
```

Definition at line 43 of file gsttensor_reposink.h.

9.51.2.3 GST_TENSOR_REPOSINK

```
#define GST_TENSOR_REPOSINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_REPOSINK, GstTensorRepoSink))
```

Definition at line 37 of file `gsttensor_reposink.h`.

9.51.2.4 GST_TENSOR_REPOSINK_CLASS

```
#define GST_TENSOR_REPOSINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_REPOSINK, GstTensorRepoSinkClass))
```

Definition at line 39 of file `gsttensor_reposink.h`.

9.51.2.5 GST_TYPE_TENSOR_REPOSINK

```
#define GST_TYPE_TENSOR_REPOSINK (gst_tensor_reposink_get_type())
```

Definition at line 35 of file `gsttensor_reposink.h`.

9.51.3 Typedef Documentation

9.51.3.1 GstTensorRepoSink

```
typedef struct _GstTensorRepoSink GstTensorRepoSink
```

Definition at line 46 of file `gsttensor_reposink.h`.

9.51.3.2 GstTensorRepoSinkClass

```
typedef struct _GstTensorRepoSinkClass GstTensorRepoSinkClass
```

Definition at line 47 of file `gsttensor_reposink.h`.

9.51.4 Function Documentation

9.51.4.1 `gst_tensor_reposink_get_type()`

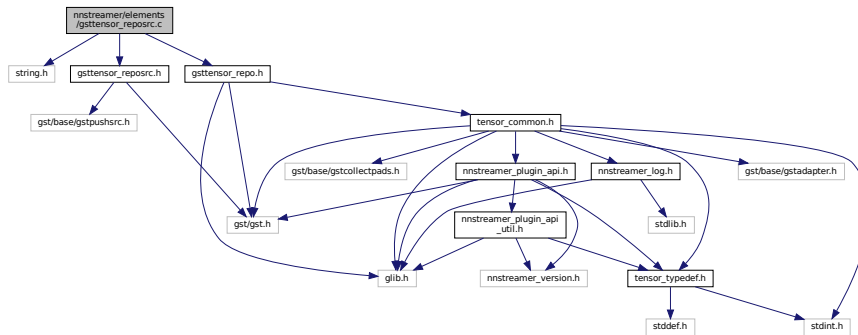
```
GType gst_tensor_reposink_get_type (
    void )
```

Function to get type of `tensor_reposink`.

9.52 `nnstreamer/elements/gsttensor_reposrc.c` File Reference

GStreamer plugin to handle tensor repository.

```
#include <string.h>
#include "gsttensor_repo.h"
#include "gsttensor_reposrc.h"
Include dependency graph for gsttensor_reposrc.c:
```



Macros

- `#define GST_CAT_DEFAULT` `gst_tensor_reposrc_debug`
- `#define CAPS_STRING` `GST_TENSOR_CAP_DEFAULT` `;` `" GST_TENSORS_CAP_DEFAULT`
- `#define DEFAULT_SILENT` `TRUE`
- `#define DEFAULT_INDEX` `0`
- `#define INVALID_INDEX` `G_MAXUINT`
- `#define gst_tensor_reposrc_parent_class` `parent_class`

Enumerations

- enum { `PROP_0`, `PROP_CAPS`, `PROP_SLOT_ID`, `PROP_SILENT` }
- tensor_reposrc properties*

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) (`gst_tensor_reposrc_debug`)
- static void [gst_tensor_reposrc_set_property](#) (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
set property of tensor_reposrc
- static void [gst_tensor_reposrc_get_property](#) (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)
get property of tensor_reposrc
- static void [gst_tensor_reposrc_dispose](#) (`GObject *object`)
object dispose of tensor_reposrc
- static `GstCaps *` [gst_tensor_reposrc_getcaps](#) (`GstBaseSrc *src`, `GstCaps *filter`)
get cap of tensor_reposrc
- static `GstFlowReturn` [gst_tensor_reposrc_create](#) (`GstPushSrc *src`, `GstBuffer **buffer`)
create func of tensor_reposrc
- [G_DEFINE_TYPE](#) (`GstTensorRepoSrc`, `gst_tensor_reposrc`, `GST_TYPE_PUSH_SRC`)
- static void [gst_tensor_reposrc_class_init](#) (`GstTensorRepoSrcClass *klass`)
class initialization of tensor_reposrc
- static void [gst_tensor_reposrc_init](#) (`GstTensorRepoSrc *self`)
object initialization of tensor_reposrc
- static `GstBuffer *` [gst_tensor_reposrc_gen_dummy_buffer](#) (`GstTensorRepoSrc *self`)
create dummy buffer for initialization

9.52.1 Detailed Description

GStreamer plugin to handle tensor repository.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details. SECTION: element-tensor_reposrc

Pop element to handle tensor repo

Date

19 Nov 2018

See also

<https://github.com/nntstreamer/nntstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.52.2 Macro Definition Documentation

9.52.2.1 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT "; " GST_TENSORS_CAP_DEFAULT
```

Definition at line 41 of file gsttensor_reposrc.c.

9.52.2.2 DEFAULT_INDEX

```
#define DEFAULT_INDEX 0
```

Definition at line 55 of file gsttensor_reposrc.c.

9.52.2.3 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Definition at line 54 of file gsttensor_reposrc.c.

9.52.2.4 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_reposrc_debug
```

Definition at line 40 of file gsttensor_reposrc.c.

9.52.2.5 gst_tensor_reposrc_parent_class

```
#define gst_tensor_reposrc_parent_class parent_class
```

Definition at line 67 of file gsttensor_reposrc.c.

9.52.2.6 INVALID_INDEX

```
#define INVALID_INDEX G_MAXUINT
```

Definition at line 56 of file gsttensor_reposrc.c.

9.52.3 Enumeration Type Documentation

9.52.3.1 anonymous enum

anonymous enum

tensor_reposrc properties

Enumerator

PROP_0	
PROP_CAPS	
PROP_SLOT_ID	
PROP_SILENT	

Definition at line 46 of file gsttensor_reposrc.c.

9.52.4 Function Documentation

9.52.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorRepoSrc ,
    gst_tensor_reposrc ,
    GST_TYPE_PUSH_SRC )
```

9.52.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_reposrc_debug )
```

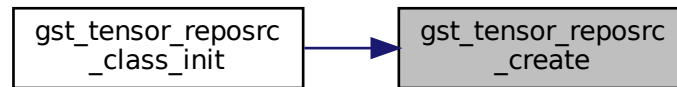
9.52.4.3 gst_tensor_reposrc_class_init()

```
static void gst_tensor_reposrc_class_init (
    GstTensorRepoSrcClass * klass ) [static]
```

class initialization of tensor_reposrc

Definition at line 74 of file gsttensor_reposrc.c.

Here is the caller graph for this function:



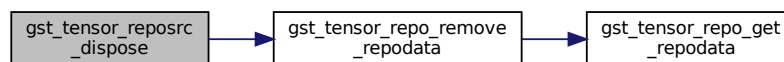
9.52.4.5 `gst_tensor_reposrc_dispose()`

```
static void gst_tensor_reposrc_dispose (  
    GObject * object ) [static]
```

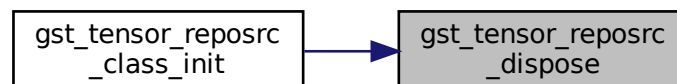
object dispose of `tensor_reposrc`

Definition at line 141 of file `gsttensor_reposrc.c`.

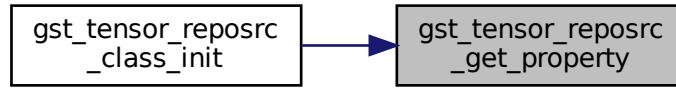
Here is the call graph for this function:



Here is the caller graph for this function:



Here is the caller graph for this function:



9.52.4.8 gst_tensor_reposrc_getcaps()

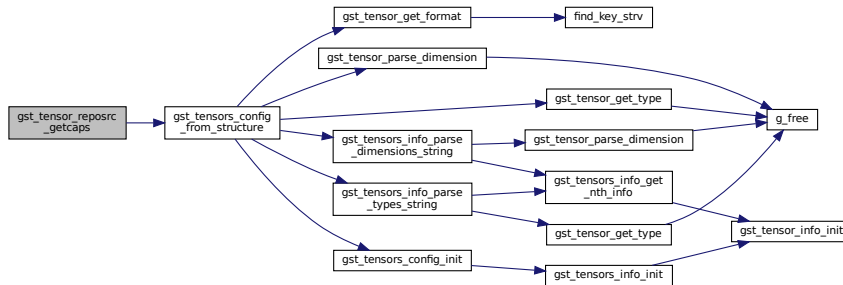
```

static GstCaps * gst_tensor_reposrc_getcaps (
    GstBaseSrc * src,
    GstCaps * filter ) [static]
  
```

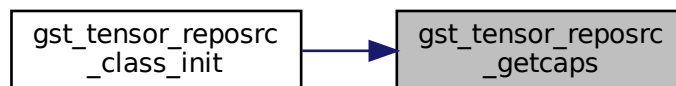
get cap of tensor_reposrc

Definition at line 160 of file gsttensor_reposrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.52.4.9 `gst_tensor_reposrc_init()`

```
static void gst_tensor_reposrc_init (
    GstTensorRepoSrc * self ) [static]
```

object initialization of `tensor_reposrc`

Definition at line 126 of file `gsttensor_reposrc.c`.

Here is the call graph for this function:



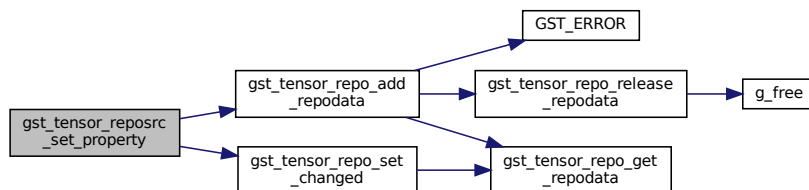
9.52.4.10 `gst_tensor_reposrc_set_property()`

```
static void gst_tensor_reposrc_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

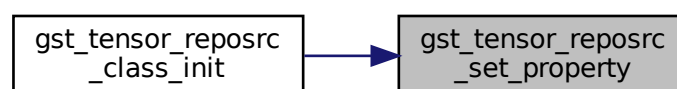
set property of `tensor_reposrc`

Definition at line 201 of file `gsttensor_reposrc.c`.

Here is the call graph for this function:



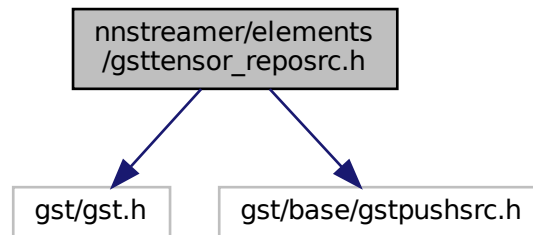
Here is the caller graph for this function:



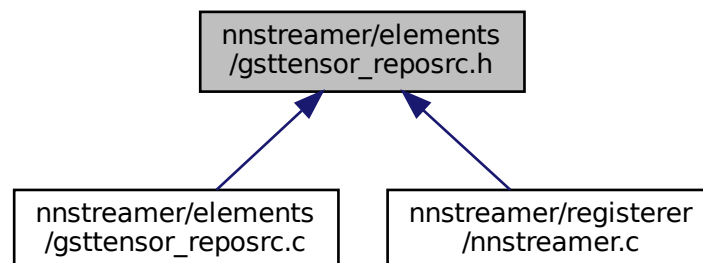
9.53 nnstreamer/elements/gsttensor_reposrc.h File Reference

GStreamer plugin to handle tensor repository.

```
#include <gst/gst.h>
#include <gst/base/gstpushsrc.h>
Include dependency graph for gsttensor_reposrc.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_GstTensorRepoSrc`](#)
GstTensorRepoSrc data structure.
- [struct `_GstTensorRepoSrcClass`](#)
GstTensorRepoSrcClass data structure.

Macros

- [#define `GST_TYPE_TENSOR_REPOSRC`](#) (`gst_tensor_reposrc_get_type()`)
- [#define `GST_TENSOR_REPOSRC\(obj\)`](#) (`G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_REPOSRC, GstT`
- [#define `GST_TENSOR_REPOSRC_CLASS\(klass\)`](#) (`G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_REPOS`
- [#define `GST_IS_TENSOR_REPOSRC\(obj\)`](#) (`G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_REPOSRC)`)
- [#define `GST_IS_TENSOR_REPOSRC_CLASS\(klass\)`](#) (`G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_REP`

Typedefs

- typedef struct [_GstTensorRepoSrc](#) [GstTensorRepoSrc](#)
- typedef struct [_GstTensorRepoSrcClass](#) [GstTensorRepoSrcClass](#)

Functions

- GType [gst_tensor_reposrc_get_type](#) (void)
Function to get type of tensor_reposrc.

9.53.1 Detailed Description

GStreamer plugin to handle tensor repository.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

19 Nov 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.53.2 Macro Definition Documentation

9.53.2.1 GST_IS_TENSOR_REPOSRC

```
#define GST_IS_TENSOR_REPOSRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_REPOSRC))
```

Definition at line 41 of file `gsttensor_reposrc.h`.

9.53.2.2 GST_IS_TENSOR_REPOSRC_CLASS

```
#define GST_IS_TENSOR_REPOSRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_REPOSRC))
```

Definition at line 43 of file `gsttensor_reposrc.h`.

9.53.2.3 GST_TENSOR_REPOSRC

```
#define GST_TENSOR_REPOSRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_REPOSRC, GstTensorRepoSrc))
```

Definition at line 37 of file `gsttensor_reposrc.h`.

9.53.2.4 GST_TENSOR_REPOSRC_CLASS

```
#define GST_TENSOR_REPOSRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_REPOSRC, GstTensorRepoSrcClass))
```

Definition at line 39 of file `gsttensor_reposrc.h`.

9.53.2.5 GST_TYPE_TENSOR_REPOSRC

```
#define GST_TYPE_TENSOR_REPOSRC (gst_tensor_reposrc_get_type())
```

Definition at line 35 of file `gsttensor_reposrc.h`.

9.53.3 Typedef Documentation

9.53.3.1 GstTensorRepoSrc

```
typedef struct _GstTensorRepoSrc GstTensorRepoSrc
```

Definition at line 46 of file `gsttensor_reposrc.h`.

9.53.3.2 GstTensorRepoSrcClass

```
typedef struct _GstTensorRepoSrcClass GstTensorRepoSrcClass
```

Definition at line 47 of file gsttensor_reposrc.h.

9.53.4 Function Documentation

9.53.4.1 gst_tensor_reposrc_get_type()

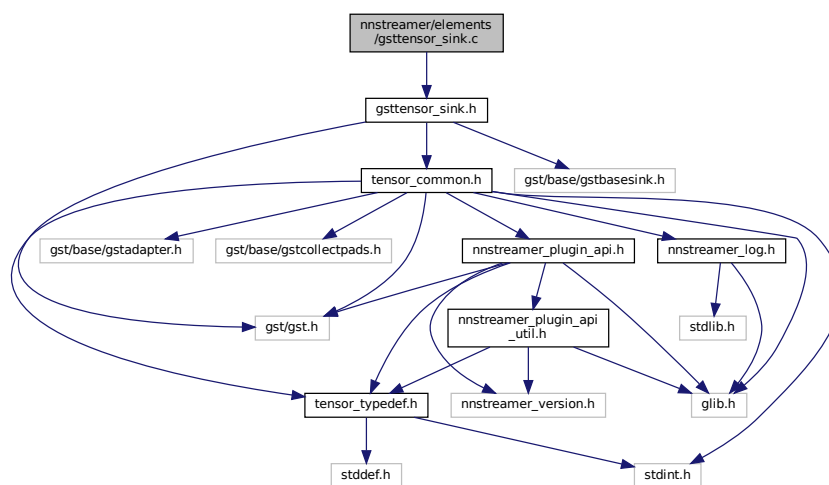
```
GType gst_tensor_reposrc_get_type (
    void )
```

Function to get type of tensor_reposrc.

9.54 nnstreamer/elements/gsttensor_sink.c File Reference

GStreamer plugin to handle tensor stream.

```
#include "gsttensor_sink.h"
Include dependency graph for gsttensor_sink.c:
```



Macros

- #define `DBG` (!self->silent)
Macro for debug mode.
- #define `silent_debug_timestamp`(self, buf)
- #define `GST_CAT_DEFAULT` `gst_tensor_sink_debug`
- #define `DEFAULT_EMIT_SIGNAL` `TRUE`
Flag to emit signals.
- #define `DEFAULT_SIGNAL_RATE` `0`
New data signals per second.
- #define `DEFAULT_SILENT` `TRUE`
Flag to print minimized log.
- #define `DEFAULT_QOS` `TRUE`
Flag for qos event.
- #define `DEFAULT_SYNC` `FALSE`
Flag to synchronize on the clock (Default FALSE). It may be delayed with `tensor_filter` element, to invoke neural network model. See `GstBaseSink::sync` property for more details.
- #define `gst_tensor_sink_parent_class` `parent_class`

Enumerations

- enum { `SIGNAL_NEW_DATA`, `SIGNAL_STREAM_START`, `SIGNAL_EOS`, `LAST_SIGNAL` }
tensor_sink signals.
- enum { `PROP_0`, `PROP_SIGNAL_RATE`, `PROP_EMIT_SIGNAL`, `PROP_SILENT` }
tensor_sink properties.

Functions

- `GST_DEBUG_CATEGORY_STATIC` (`gst_tensor_sink_debug`)
- static void `gst_tensor_sink_set_property` (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
Setter for tensor_sink properties.
- static void `gst_tensor_sink_get_property` (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
Getter for tensor_sink properties.
- static void `gst_tensor_sink_finalize` (GObject *object)
Function to finalize instance.
- static gboolean `gst_tensor_sink_event` (GstBaseSink *sink, GstEvent *event)
Handle events.
- static gboolean `gst_tensor_sink_query` (GstBaseSink *sink, GstQuery *query)
Handle queries.
- static GstFlowReturn `gst_tensor_sink_render` (GstBaseSink *sink, GstBuffer *buffer)
Handle buffer.
- static GstFlowReturn `gst_tensor_sink_render_list` (GstBaseSink *sink, GstBufferList *buffer_list)
Handle list of buffers.
- static void `gst_tensor_sink_render_buffer` (GstTensorSink *self, GstBuffer *buffer)
Handle buffer data.
- static void `gst_tensor_sink_set_last_render_time` (GstTensorSink *self, GstClockTime now)
Setter for value last_render_time.
- static GstClockTime `gst_tensor_sink_get_last_render_time` (GstTensorSink *self)

- Getter for value last_render_time.*
- static void `gst_tensor_sink_set_signal_rate` (`GstTensorSink *self`, `guint rate`)
 - Setter for value signal_rate.*
- static `guint gst_tensor_sink_get_signal_rate` (`GstTensorSink *self`)
 - Getter for value signal_rate.*
- static void `gst_tensor_sink_set_emit_signal` (`GstTensorSink *self`, `gboolean emit`)
 - Setter for flag emit_signal.*
- static `gboolean gst_tensor_sink_get_emit_signal` (`GstTensorSink *self`)
 - Getter for flag emit_signal.*
- static void `gst_tensor_sink_set_silent` (`GstTensorSink *self`, `gboolean silent`)
 - Setter for flag silent.*
- static `gboolean gst_tensor_sink_get_silent` (`GstTensorSink *self`)
 - Getter for flag silent.*
- `G_DEFINE_TYPE` (`GstTensorSink`, `gst_tensor_sink`, `GST_TYPE_BASE_SINK`)
- static void `gst_tensor_sink_class_init` (`GstTensorSinkClass *klass`)
 - Initialize tensor_sink class.*
- static void `gst_tensor_sink_init` (`GstTensorSink *self`)
 - Initialize tensor_sink element.*

Variables

- static `guint _tensor_sink_signals [LAST_SIGNAL] = { 0 }`
 - Variable for signal ids.*

9.54.1 Detailed Description

GStreamer plugin to handle tensor stream.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details. SECTION:element-tensor_sink

Sink element to handle tensor stream

Date

15 June 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung [jy1210.jung@samsung.com](mailto: jy1210.jung@samsung.com)

Bug No known bugs except for NYI items

9.54.2 Macro Definition Documentation

9.54.2.1 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

Definition at line 41 of file gsttensor_sink.c.

9.54.2.2 DEFAULT_EMIT_SIGNAL

```
#define DEFAULT_EMIT_SIGNAL TRUE
```

Flag to emit signals.

Definition at line 80 of file gsttensor_sink.c.

9.54.2.3 DEFAULT_QOS

```
#define DEFAULT_QOS TRUE
```

Flag for qos event.

See GstBaseSink::qos property for more details.

Definition at line 97 of file gsttensor_sink.c.

9.54.2.4 DEFAULT_SIGNAL_RATE

```
#define DEFAULT_SIGNAL_RATE 0
```

New data signals per second.

Definition at line 85 of file gsttensor_sink.c.

9.54.2.5 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 90 of file gsttensor_sink.c.

9.54.2.6 DEFAULT_SYNC

```
#define DEFAULT_SYNC FALSE
```

Flag to synchronize on the clock (Default FALSE). It may be delayed with `tensor_filter` element, to invoke neural network model. See `GstBaseSink::sync` property for more details.

Definition at line 104 of file gsttensor_sink.c.

9.54.2.7 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_sink_debug
```

Definition at line 53 of file gsttensor_sink.c.

9.54.2.8 gst_tensor_sink_parent_class

```
#define gst_tensor_sink_parent_class parent_class
```

Definition at line 140 of file gsttensor_sink.c.

9.54.2.9 silent_debug_timestamp

```
#define silent_debug_timestamp(  
    self,  
    buf )
```

Value:

```
do { \
  if (DBG) { \
    GST_DEBUG_OBJECT (self, "pts = %" GST_TIME_FORMAT, GST_TIME_ARGS (GST_BUFFER_PTS (buf))); \
    GST_DEBUG_OBJECT (self, "dts = %" GST_TIME_FORMAT, GST_TIME_ARGS (GST_BUFFER_DTS (buf))); \
    GST_DEBUG_OBJECT (self, "duration = %" GST_TIME_FORMAT "\n", GST_TIME_ARGS (GST_BUFFER_DURATION (buf))); \
  } \
} while (0)
```

Definition at line 44 of file gsttensor_sink.c.

9.54.3 Enumeration Type Documentation

9.54.3.1 anonymous enum

anonymous enum

tensor_sink signals.

Enumerator

SIGNAL_NEW_DATA	
SIGNAL_STREAM_START	
SIGNAL_EOS	
LAST_SIGNAL	

Definition at line 58 of file gsttensor_sink.c.

9.54.3.2 anonymous enum

anonymous enum

tensor_sink properties.

Enumerator

PROP_0	
PROP_SIGNAL_RATE	
PROP_EMIT_SIGNAL	
PROP_SILENT	

Definition at line 69 of file gsttensor_sink.c.

9.54.4 Function Documentation**9.54.4.1 G_DEFINE_TYPE()**

```
G_DEFINE_TYPE (
    GstTensorSink ,
    gst_tensor_sink ,
    GST_TYPE_BASE_SINK )
```

9.54.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_sink_debug )
```

9.54.4.3 `gst_tensor_sink_class_init()`

```
static void gst_tensor_sink_class_init (
    GstTensorSinkClass * klass ) [static]
```

Initialize `tensor_sink` class.

GObject methods

GstTensorSink::signal-rate:

The number of new data signals per second (Default 0 for unlimited, MAX 500) If signal-rate is larger than 0, GstTensorSink calculates the time to emit a signal with this property. If set 0 (default value), all the received buffers will be passed to the application.

Please note that this property does not guarantee the periodic signals. This means if GstTensorSink cannot get the buffers in time, it will pass all the buffers. (working like default 0)

GstTensorSink::emit-signal:

The flag to emit the signals for new data, stream start, and eos.

GstTensorSink::silent:

The flag to enable/disable debugging messages.

GstTensorSink::new-data:

Signal to get the buffer from GstTensorSink.

GstTensorSink::stream-start:

Signal to indicate the start of a new stream. Optional. An application can use this signal to detect the start of a new stream, instead of the message `GST_MESSAGE_STREAM_START` from pipeline.

GstTensorSink::eos:

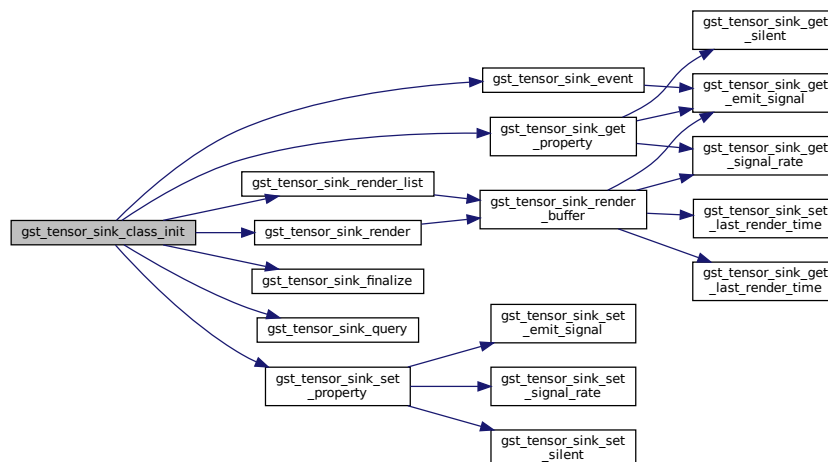
Signal to indicate the end-of-stream. Optional. An application can use this signal to detect the EOS (end-of-stream), instead of the message `GST_MESSAGE_EOS` from pipeline.

pad template

GstBaseSink methods

Definition at line 147 of file `gsttensor_sink.c`.

Here is the call graph for this function:



9.54.4.4 `gst_tensor_sink_event()`

```
static gboolean gst_tensor_sink_event (  
    GstBaseSink * sink,  
    GstEvent * event ) [static]
```

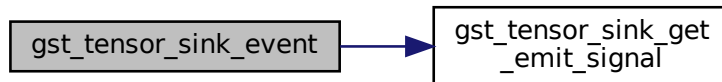
Handle events.

GstBaseSink method implementation

GstBaseSink method implementation.

Definition at line 365 of file `gsttensor_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.54.4.5 `gst_tensor_sink_finalize()`

```
static void gst_tensor_sink_finalize (  
    GObject * object ) [static]
```

Function to finalize instance.

GObject method implementation.

Definition at line 348 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.4.6 `gst_tensor_sink_get_emit_signal()`

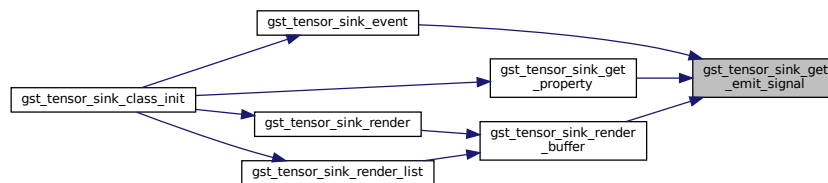
```

static gboolean gst_tensor_sink_get_emit_signal (
    GstTensorSink * self ) [static]
  
```

Getter for flag `emit_signal`.

Definition at line 610 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.4.7 `gst_tensor_sink_get_last_render_time()`

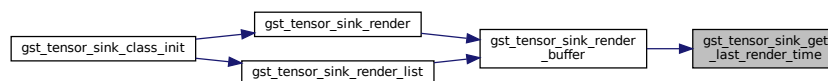
```

static GstClockTime gst_tensor_sink_get_last_render_time (
    GstTensorSink * self ) [static]
  
```

Getter for value `last_render_time`.

Definition at line 548 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.4.8 `gst_tensor_sink_get_property()`

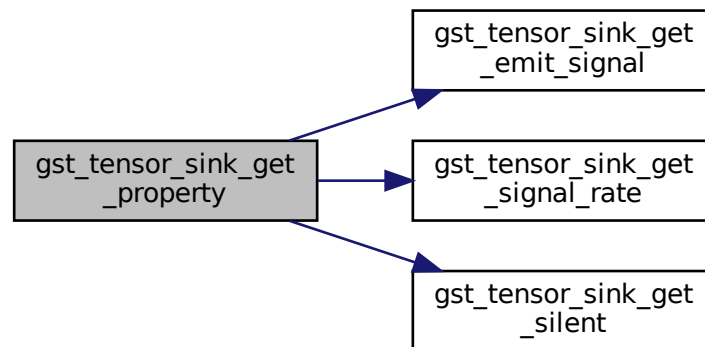
```
static void gst_tensor_sink_get_property (  
    GObject * object,  
    guint prop_id,  
    GValue * value,  
    GParamSpec * pspec ) [static]
```

Getter for tensor_sink properties.

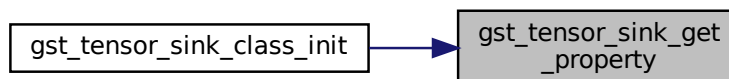
GObject method implementation.

Definition at line 316 of file gsttensor_sink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



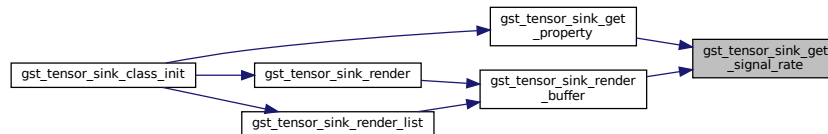
9.54.4.9 `gst_tensor_sink_get_signal_rate()`

```
static guint gst_tensor_sink_get_signal_rate (
    GstTensorSink * self ) [static]
```

Getter for value `signal_rate`.

Definition at line 579 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



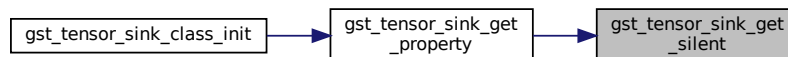
9.54.4.10 `gst_tensor_sink_get_silent()`

```
static gboolean gst_tensor_sink_get_silent (
    GstTensorSink * self ) [static]
```

Getter for flag `silent`.

Definition at line 639 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.4.11 `gst_tensor_sink_init()`

```
static void gst_tensor_sink_init (
    GstTensorSink * self ) [static]
```

Initialize `tensor_sink` element.

init properties

enable qos

Definition at line 257 of file `gsttensor_sink.c`.

9.54.4.12 `gst_tensor_sink_query()`

```
static gboolean gst_tensor_sink_query (  
    GstBaseSink * sink,  
    GstQuery * query ) [static]
```

Handle queries.

GstBaseSink method implementation. tensor sink does not support seeking

Definition at line 406 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.4.13 `gst_tensor_sink_render()`

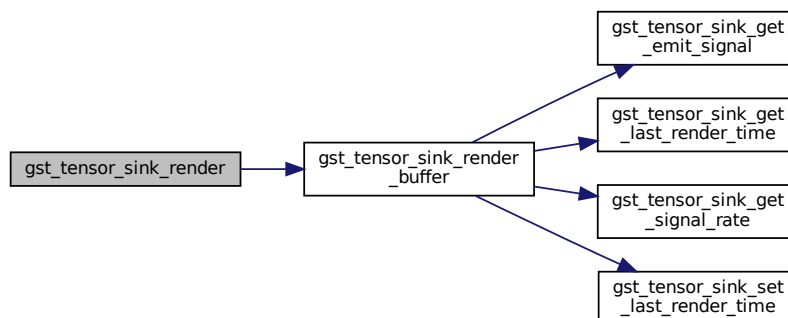
```
static GstFlowReturn gst_tensor_sink_render (  
    GstBaseSink * sink,  
    GstBuffer * buffer ) [static]
```

Handle buffer.

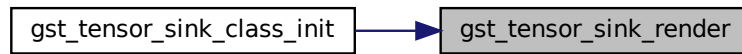
GstBaseSink method implementation.

Definition at line 438 of file `gsttensor_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.54.4.14 `gst_tensor_sink_render_buffer()`

```

static void gst_tensor_sink_render_buffer (
    GstTensorSink * self,
    GstBuffer * buffer ) [static]
  
```

Handle buffer data.

internal functions

Returns

None

Parameters

<i>self</i>	pointer to GstTensorSink
<i>buffer</i>	pointer to GstBuffer to be handled

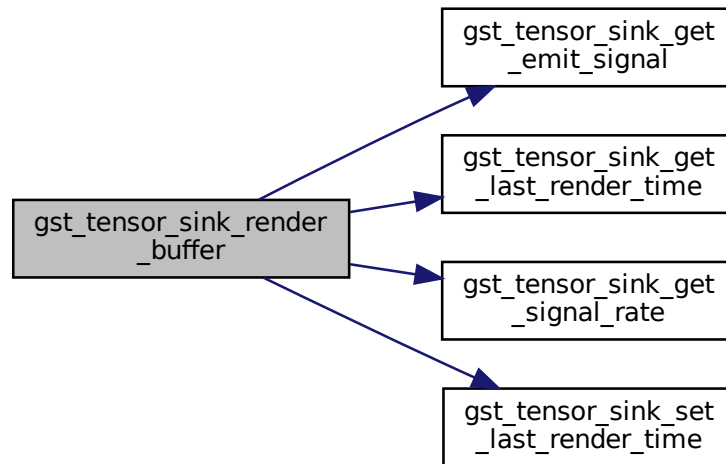
time for next signal

send data after render time, or firstly received buffer

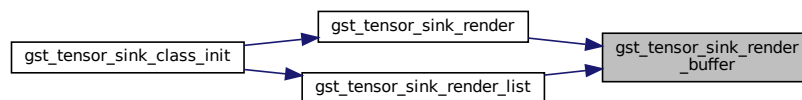
send data if signal rate is 0

Definition at line 479 of file gsttensor_sink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.54.4.15 `gst_tensor_sink_render_list()`

```

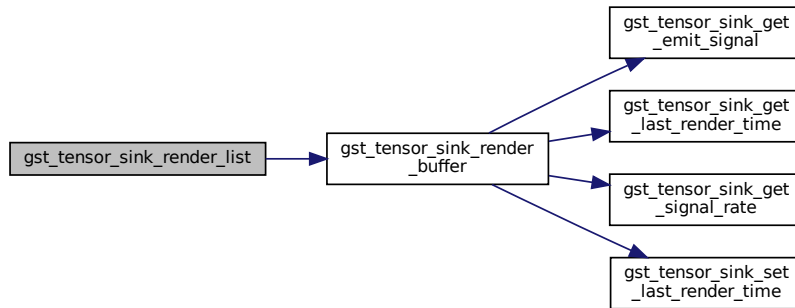
static GstFlowReturn gst_tensor_sink_render_list (
    GstBaseSink * sink,
    GstBufferList * buffer_list ) [static]
  
```

Handle list of buffers.

GstBaseSink method implementation.

Definition at line 454 of file `gsttensor_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.54.4.16 `gst_tensor_sink_set_emit_signal()`

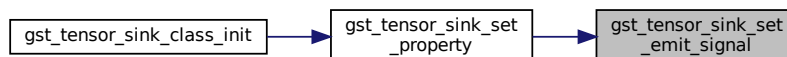
```

static void gst_tensor_sink_set_emit_signal (
    GstTensorSink * self,
    gboolean emit ) [static]
  
```

Setter for flag `emit_signal`.

Definition at line 596 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



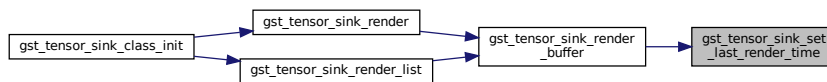
9.54.4.17 `gst_tensor_sink_set_last_render_time()`

```
static void gst_tensor_sink_set_last_render_time (
    GstTensorSink * self,
    GstClockTime now ) [static]
```

Setter for value `last_render_time`.

Definition at line 535 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.4.18 `gst_tensor_sink_set_property()`

```
static void gst_tensor_sink_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

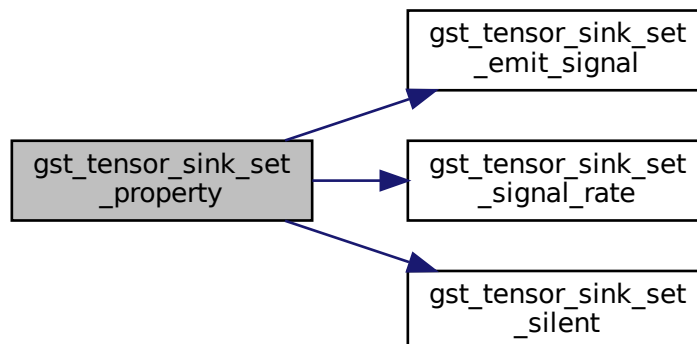
Setter for `tensor_sink` properties.

GObject method implementation

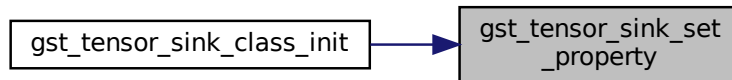
GObject method implementation.

Definition at line 284 of file `gsttensor_sink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.54.4.19 `gst_tensor_sink_set_signal_rate()`

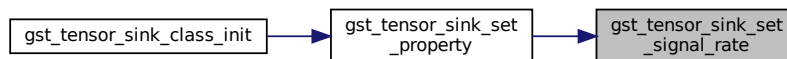
```

static void gst_tensor_sink_set_signal_rate (
    GstTensorSink * self,
    guint rate ) [static]
  
```

Setter for value `signal_rate`.

Definition at line 565 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.4.20 `gst_tensor_sink_set_silent()`

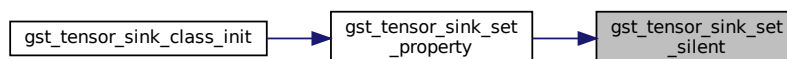
```

static void gst_tensor_sink_set_silent (
    GstTensorSink * self,
    gboolean silent ) [static]
  
```

Setter for flag `silent`.

Definition at line 627 of file `gsttensor_sink.c`.

Here is the caller graph for this function:



9.54.5 Variable Documentation

9.54.5.1 _tensor_sink_signals

```
guint _tensor_sink_signals[LAST\_SIGNAL] = { 0 } [static]
```

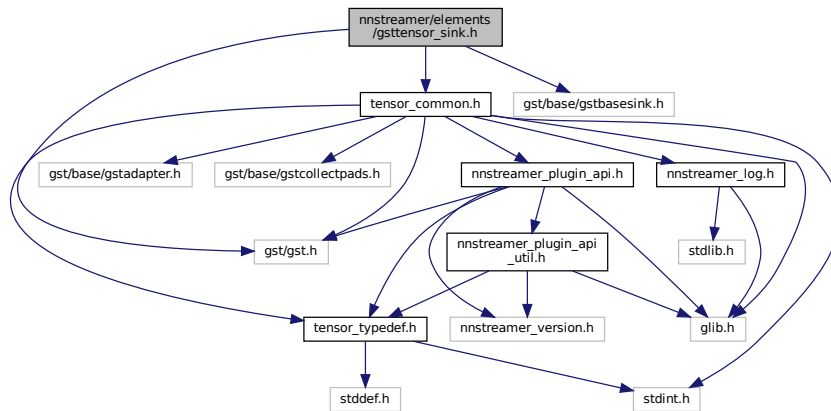
Variable for signal ids.

Definition at line 109 of file gsttensor_sink.c.

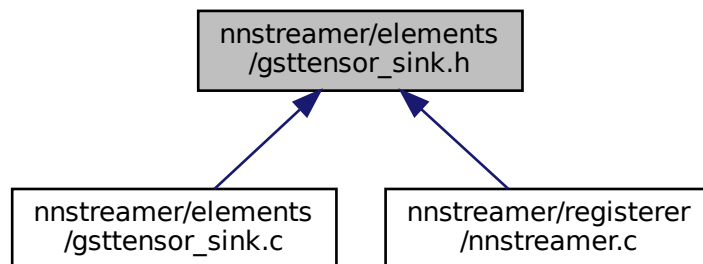
9.55 nnstreamer/elements/gsttensor_sink.h File Reference

GStreamer plugin to handle tensor stream.

```
#include <gst/gst.h>
#include <gst/base/gstbasesink.h>
#include <tensor_common.h>
Include dependency graph for gsttensor_sink.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorSink](#)
GstTensorSink data structure.
- struct [_GstTensorSinkClass](#)
GstTensorSinkClass data structure.

Macros

- #define [GST_TYPE_TENSOR_SINK](#) ([gst_tensor_sink_get_type\(\)](#))
- #define [GST_TENSOR_SINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_SINK, GstTensorSink\)](#))
- #define [GST_TENSOR_SINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_SINK, GstTensorSinkClass\)](#))
- #define [GST_IS_TENSOR_SINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_SINK\)](#))
- #define [GST_IS_TENSOR_SINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_SINK\)](#))

Typedefs

- typedef struct [_GstTensorSink](#) [GstTensorSink](#)
- typedef struct [_GstTensorSinkClass](#) [GstTensorSinkClass](#)

Functions

- GType [gst_tensor_sink_get_type](#) (void)
Function to get type of tensor_sink.

9.55.1 Detailed Description

GStreamer plugin to handle tensor stream.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Samsung Electronics Co., Ltd.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

15 June 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung jy1210.jung@samsung.com

Bug No known bugs except for NYI items

9.55.2 Macro Definition Documentation

9.55.2.1 GST_IS_TENSOR_SINK

```
#define GST_IS_TENSOR_SINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_SINK))
```

Definition at line 42 of file gsttensor_sink.h.

9.55.2.2 GST_IS_TENSOR_SINK_CLASS

```
#define GST_IS_TENSOR_SINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_SINK))
```

Definition at line 44 of file gsttensor_sink.h.

9.55.2.3 GST_TENSOR_SINK

```
#define GST_TENSOR_SINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_SINK, GstTensorSink))
```

Definition at line 38 of file gsttensor_sink.h.

9.55.2.4 GST_TENSOR_SINK_CLASS

```
#define GST_TENSOR_SINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_SINK, GstTensorSinkClass))
```

Definition at line 40 of file gsttensor_sink.h.

9.55.2.5 GST_TYPE_TENSOR_SINK

```
#define GST_TYPE_TENSOR_SINK (gst_tensor_sink_get_type())
```

Definition at line 36 of file gsttensor_sink.h.

9.55.3 Typedef Documentation

9.55.3.1 GstTensorSink

```
typedef struct _GstTensorSink GstTensorSink
```

Definition at line 47 of file gsttensor_sink.h.

9.55.3.2 GstTensorSinkClass

```
typedef struct _GstTensorSinkClass GstTensorSinkClass
```

Definition at line 48 of file gsttensor_sink.h.

9.55.4 Function Documentation

9.55.4.1 gst_tensor_sink_get_type()

```
GType gst_tensor_sink_get_type (
    void )
```

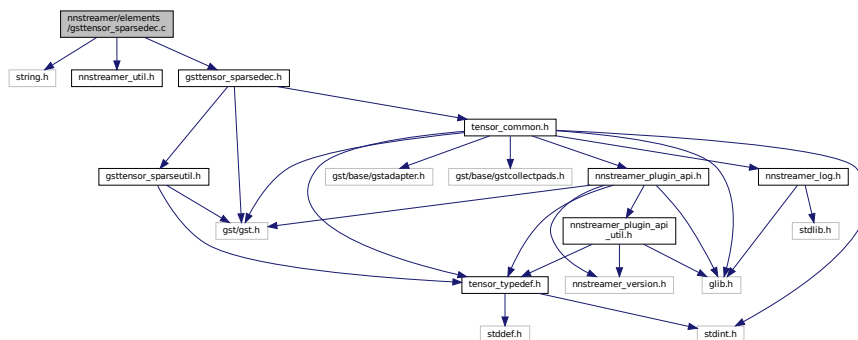
Function to get type of tensor_sink.

9.56 nnstreamer/elements/gsttensor_sparsedec.c File Reference

GStreamer element to decode sparse tensors into dense tensors.

```
#include <string.h>
#include <nnstreamer_util.h>
#include "gsttensor_sparsedec.h"
```

Include dependency graph for gsttensor_sparsedec.c:



Macros

- #define [DBG](#) (!self->silent)
Macro for debug mode.
- #define [GST_CAT_DEFAULT](#) [gst_tensor_sparse_dec_debug](#)
- #define [DEFAULT_SILENT](#) [TRUE](#)
Flag to print minimized log.
- #define [gst_tensor_sparse_dec_parent_class](#) [parent_class](#)

Enumerations

- enum { [PROP_0](#), [PROP_SILENT](#) }
tensor_sparse_dec properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_sparse_dec_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorSparseDec](#), [gst_tensor_sparse_dec](#), [GST_TYPE_ELEMENT](#))
- static void [gst_tensor_sparse_dec_finalize](#) ([GObject](#) *object)
Function to finalize instance.
- static void [gst_tensor_sparse_dec_set_property](#) ([GObject](#) *object, [guint](#) prop_id, [const GValue](#) *value, [GParamSpec](#) *pspec)
Setter for tensor_sparse_dec properties.
- static void [gst_tensor_sparse_dec_get_property](#) ([GObject](#) *object, [guint](#) prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
Getter for tensor_sparse_dec properties.
- static [GstFlowReturn](#) [gst_tensor_sparse_dec_chain](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstBuffer](#) *buf)
Internal function to transform the input buffer.
- static [gboolean](#) [gst_tensor_sparse_dec_sink_event](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstEvent](#) *event)
This function handles sink pad event.
- static [gboolean](#) [gst_tensor_sparse_dec_sink_query](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstQuery](#) *query)
This function handles sink pad query.
- static void [gst_tensor_sparse_dec_class_init](#) ([GstTensorSparseDecClass](#) *klass)
Initialize the tensor_sparse's class.
- static void [gst_tensor_sparse_dec_init](#) ([GstTensorSparseDec](#) *self)
Initialize tensor_sparse_dec element.
- static [GstCaps](#) * [gst_tensor_sparse_dec_query_caps](#) ([GstTensorSparseDec](#) *self, [GstPad](#) *pad, [GstCaps](#) *filter)
Get pad caps for caps negotiation.

Variables

- static [GstStaticPadTemplate](#) [sink_template](#)
Template for sink pad.
- static [GstStaticPadTemplate](#) [src_template](#)
Template for src pad.

9.56.1 Detailed Description

GStreamer element to decode sparse tensors into dense tensors.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

27 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Yongjoo Ahn yongjool.ahn@samsung.com

Bug No known bugs except for NYI items

9.56.2 Macro Definition Documentation

9.56.2.1 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

SECTION:element-tensor_sparse_dec

tensor_sparse_dec is a GStreamer element to decode incoming sparse tensor into static (dense) format.

The input is always in the format of other/tensors,format=sparse. The output is always in the format of other/tensors,format=static.

Please see also tensor_sparse_enc.

```
<refsect2> <title>Example launch line</title> |[ gst-launch-1.0 ... ! other/tensors,format=static ! \ tensor_↔
sparse_enc ! other/tensors,format=sparse ! \ tensor_sparse_dec ! tensor_sink ]| </refsect2>
```

Definition at line 45 of file gsttensor_sparsedec.c.

9.56.2.2 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 63 of file gsttensor_sparsedec.c.

9.56.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_sparse_dec_debug
```

Definition at line 49 of file gsttensor_sparsedec.c.

9.56.2.4 gst_tensor_sparse_dec_parent_class

```
#define gst_tensor_sparse_dec_parent_class parent_class
```

Definition at line 81 of file gsttensor_sparsedec.c.

9.56.3 Enumeration Type Documentation

9.56.3.1 anonymous enum

anonymous enum

tensor_sparse_dec properties

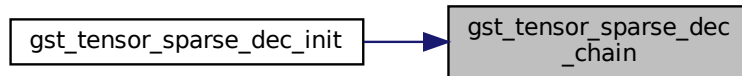
Enumerator

PROP_0	
PROP_SILENT	

Definition at line 54 of file gsttensor_sparsedec.c.

9.56.4 Function Documentation

Here is the caller graph for this function:



9.56.4.4 gst_tensor_sparse_dec_class_init()

```

static void gst_tensor_sparse_dec_class_init (
    GstTensorSparseDecClass * klass ) [static]
  
```

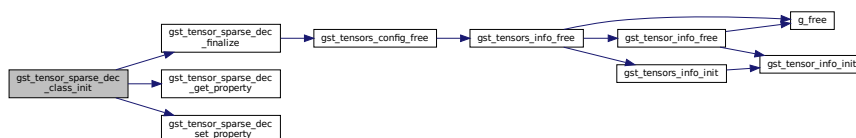
Initialize the tensor_sparse's class.

[GstTensorSparseDec::silent](#):

The flag to enable/disable debugging messages.

Definition at line 101 of file `gsttensor_sparsedec.c`.

Here is the call graph for this function:



9.56.4.5 gst_tensor_sparse_dec_finalize()

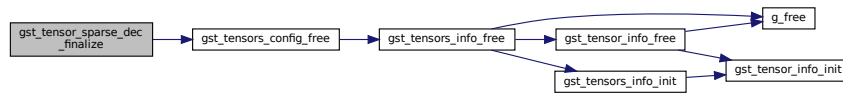
```

static void gst_tensor_sparse_dec_finalize (
    GObject * object ) [static]
  
```

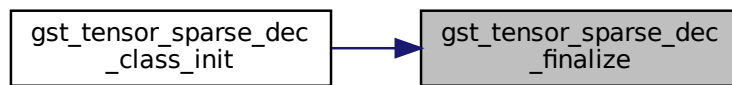
Function to finalize instance.

Definition at line 172 of file `gsttensor_sparsedec.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.56.4.6 `gst_tensor_sparse_dec_get_property()`

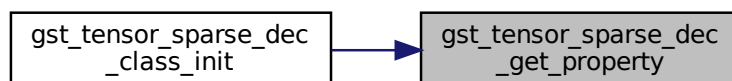
```

static void gst_tensor_sparse_dec_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
  
```

Getter for `tensor_sparse_dec` properties.

Definition at line 208 of file `gsttensor_sparsedec.c`.

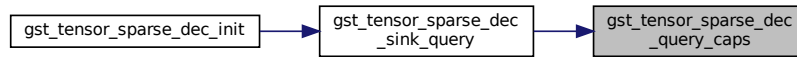
Here is the caller graph for this function:



pad don't have current caps. use the template caps

Definition at line 229 of file gsttensor_sparsedec.c.

Here is the caller graph for this function:



9.56.4.9 `gst_tensor_sparse_dec_set_property()`

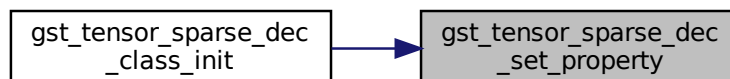
```

static void gst_tensor_sparse_dec_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

Setter for `tensor_sparse_dec` properties.

Definition at line 187 of file gsttensor_sparsedec.c.

Here is the caller graph for this function:



9.56.4.10 `gst_tensor_sparse_dec_sink_event()`

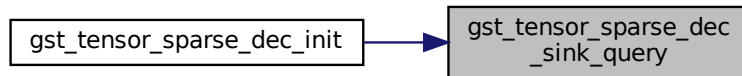
```

static gboolean gst_tensor_sparse_dec_sink_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
  
```

This function handles sink pad event.

Definition at line 313 of file gsttensor_sparsedec.c.

Here is the caller graph for this function:



9.56.5 Variable Documentation

9.56.5.1 sink_template

```
GstStaticPadTemplate sink_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (GST_TENSORS_SPARSE_CAP_DEFAULT))
```

Template for sink pad.

Definition at line 68 of file `gsttensor_sparsedec.c`.

9.56.5.2 src_template

```
GstStaticPadTemplate src_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (GST_TENSORS_CAP_DEFAULT))
```

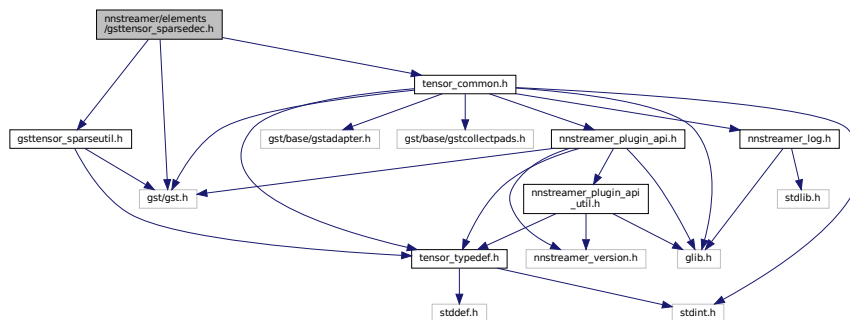
Template for src pad.

Definition at line 76 of file `gsttensor_sparsedec.c`.

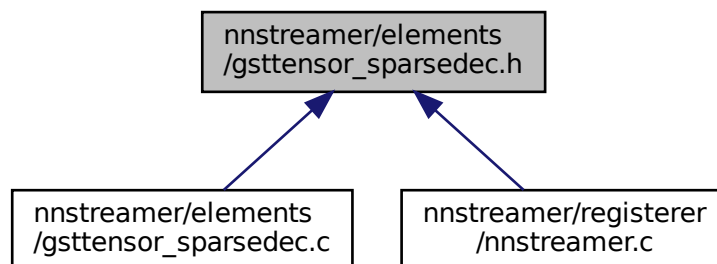
9.57 nnstreamer/elements/gsttensor_sparsedec.h File Reference

GStreamer element to decode sparse tensors into dense tensors.

```
#include <gst/gst.h>
#include <tensor_common.h>
#include "gsttensor_sparseutil.h"
Include dependency graph for gsttensor_sparsedec.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorSparseDec](#)
GstTensorSparseDec data structure.
- struct [_GstTensorSparseDecClass](#)
GstTensorSparseClass data structure.

Macros

- #define [GST_TYPE_TENSOR_SPARSE_DEC](#) ([gst_tensor_sparse_dec_get_type\(\)](#))
- #define [GST_TENSOR_SPARSE_DEC\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_SPARSE_DEC, G_TENSOR_SPARSE_DEC\)](#))
- #define [GST_TENSOR_SPARSE_DEC_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_SPARSE_DEC, G_TENSOR_SPARSE_DEC_CLASS\)](#))
- #define [GST_IS_TENSOR_SPARSE_DEC\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_SPARSE_DEC\)](#))
- #define [GST_IS_TENSOR_SPARSE_DEC_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_SPARSE_DEC\)](#))

Typedefs

- typedef struct [_GstTensorSparseDec](#) [GstTensorSparseDec](#)
- typedef struct [_GstTensorSparseDecClass](#) [GstTensorSparseDecClass](#)

Functions

- GType [gst_tensor_sparse_dec_get_type](#) (void)
Function to get type of tensor_sparse.

9.57.1 Detailed Description

GStreamer element to decode sparse tensors into dense tensors.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

27 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Yongjoo Ahn yongjool.ahn@samsung.com

Bug No known bugs except for NYI items

9.57.2 Macro Definition Documentation

9.57.2.1 GST_IS_TENSOR_SPARSE_DEC

```
#define GST_IS_TENSOR_SPARSE_DEC(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_SPARSE_DEC))
```

Definition at line 28 of file `gsttensor_sparsedec.h`.

9.57.2.2 GST_IS_TENSOR_SPARSE_DEC_CLASS

```
#define GST_IS_TENSOR_SPARSE_DEC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_SPARSE_DEC))
```

Definition at line 30 of file `gsttensor_sparsedec.h`.

9.57.2.3 GST_TENSOR_SPARSE_DEC

```
#define GST_TENSOR_SPARSE_DEC(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_SPARSE_DEC, GstTensorSparseDec))
```

Definition at line 24 of file `gsttensor_sparsedec.h`.

9.57.2.4 GST_TENSOR_SPARSE_DEC_CLASS

```
#define GST_TENSOR_SPARSE_DEC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_SPARSE_DEC, GstTensorSparseDecClass))
```

Definition at line 26 of file `gsttensor_sparsedec.h`.

9.57.2.5 GST_TYPE_TENSOR_SPARSE_DEC

```
#define GST_TYPE_TENSOR_SPARSE_DEC (gst_tensor_sparse_dec_get_type())
```

Definition at line 22 of file `gsttensor_sparsedec.h`.

9.57.3 Typedef Documentation

9.57.3.1 GstTensorSparseDec

```
typedef struct _GstTensorSparseDec GstTensorSparseDec
```

Definition at line 33 of file `gsttensor_sparsedec.h`.

9.57.3.2 GstTensorSparseDecClass

```
typedef struct _GstTensorSparseDecClass GstTensorSparseDecClass
```

Definition at line 34 of file `gsttensor_sparsedec.h`.

9.57.4 Function Documentation

9.57.4.1 `gst_tensor_sparse_dec_get_type()`

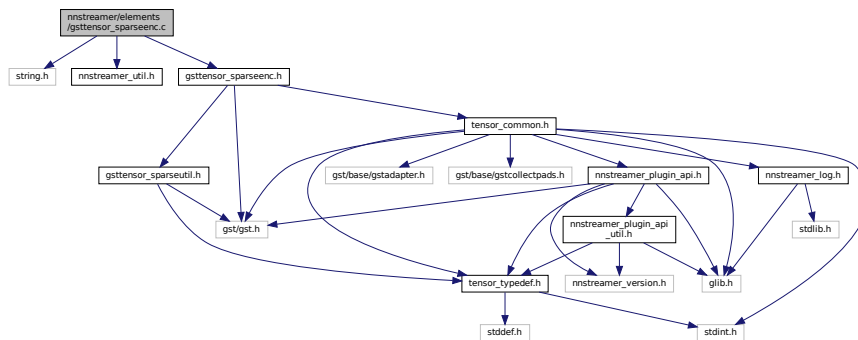
```
GType gst_tensor_sparse_dec_get_type (
    void )
```

Function to get type of `tensor_sparse`.

9.58 `nnstreamer/elements/gsttensor_sparseenc.c` File Reference

GStreamer element to encode dense tensors into sparse tensors.

```
#include <string.h>
#include <nnstreamer_util.h>
#include "gsttensor_sparseenc.h"
Include dependency graph for gsttensor_sparseenc.c:
```



Macros

- `#define DBG (!self->silent)`
Macro for debug mode.
- `#define GST_CAT_DEFAULT gst_tensor_sparse_enc_debug`
- `#define DEFAULT_SILENT TRUE`
Flag to print minimized log.
- `#define gst_tensor_sparse_enc_parent_class parent_class`

Enumerations

- enum { `PROP_0`, `PROP_SILENT` }
tensor_sparse_enc properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_sparse_enc_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorSparseEnc](#), [gst_tensor_sparse_enc](#), [GST_TYPE_ELEMENT](#))
- static void [gst_tensor_sparse_enc_finalize](#) ([GObject](#) *object)

Function to finalize instance.
- static void [gst_tensor_sparse_enc_set_property](#) ([GObject](#) *object, guint prop_id, const [GValue](#) *value, [GParamSpec](#) *pspec)

Setter for tensor_sparse_enc properties.
- static void [gst_tensor_sparse_enc_get_property](#) ([GObject](#) *object, guint prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)

Getter for tensor_sparse_enc properties.
- static [GstFlowReturn](#) [gst_tensor_sparse_enc_chain](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstBuffer](#) *buf)

Internal function to transform the input buffer.
- static [GstCaps](#) * [gst_tensor_sparse_enc_query_caps](#) ([GstTensorSparseEnc](#) *self, [GstPad](#) *pad, [GstCaps](#) *filter)

Get pad caps for caps negotiation.
- static gboolean [gst_tensor_sparse_enc_sink_event](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstEvent](#) *event)

This function handles sink pad event.
- static gboolean [gst_tensor_sparse_enc_sink_query](#) ([GstPad](#) *pad, [GstObject](#) *parent, [GstQuery](#) *query)

This function handles sink pad query.
- static void [gst_tensor_sparse_enc_class_init](#) ([GstTensorSparseEncClass](#) *klass)

Initialize the tensor_sparse's class.
- static void [gst_tensor_sparse_enc_init](#) ([GstTensorSparseEnc](#) *self)

Initialize tensor_sparse_enc element.
- static gboolean [gst_tensor_sparse_enc_parse_caps](#) ([GstTensorSparseEnc](#) *self, const [GstCaps](#) *caps)

Parse caps and set tensors config.

Variables

- static [GstStaticPadTemplate](#) [sink_template](#)

Template for sink pad.
- static [GstStaticPadTemplate](#) [src_template](#)

Template for src pad.

9.58.1 Detailed Description

GStreamer element to encode dense tensors into sparse tensors.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

27 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Yongjoo Ahn yongjool.ahn@samsung.com

Bug No known bugs except for NYI items

9.58.2 Macro Definition Documentation

9.58.2.1 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

SECTION:element-tensor_sparse_enc

tensor_sparse_enc is a GStreamer element to encode incoming tensor into sparse format.

The input is always in the format of other/tensors,format=static. The output is always in the format of other/tensors,format=sparse.

Please see also tensor_sparse_dec.

```
<refsect2> <title>Example launch line</title> |[ gst-launch-1.0 ... ! other/tensors,format=static ! \ tensor_↔
sparse_enc ! tensor_sink ]| </refsect2>
```

Definition at line 44 of file gsttensor_sparseenc.c.

9.58.2.2 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Flag to print minimized log.

Definition at line 62 of file gsttensor_sparseenc.c.

9.58.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_sparse_enc_debug
```

Definition at line 48 of file gsttensor_sparseenc.c.

9.58.2.4 gst_tensor_sparse_enc_parent_class

```
#define gst_tensor_sparse_enc_parent_class parent_class
```

Definition at line 80 of file gsttensor_sparseenc.c.

9.58.3 Enumeration Type Documentation

9.58.3.1 anonymous enum

```
anonymous enum
```

tensor_sparse_enc properties

Enumerator

PROP_0	
PROP_SILENT	

Definition at line 53 of file gsttensor_sparseenc.c.

9.58.4 Function Documentation

9.58.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorSparseEnc ,
    gst_tensor_sparse_enc ,
    GST_TYPE_ELEMENT )
```

9.58.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_sparse_enc_debug )
```

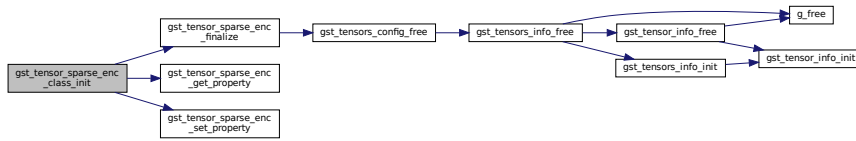
9.58.4.3 gst_tensor_sparse_enc_chain()

```
static GstFlowReturn gst_tensor_sparse_enc_chain (
    GstPad * pad,
    GstObject * parent,
    GstBuffer * buf ) [static]
```

Internal function to transform the input buffer.

Definition at line 372 of file gsttensor_sparseenc.c.

Here is the call graph for this function:



9.58.4.5 gst_tensor_sparse_enc_finalize()

```
static void gst_tensor_sparse_enc_finalize (
    GObject * object ) [static]
```

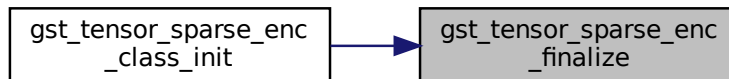
Function to finalize instance.

Definition at line 172 of file gsttensor_sparseenc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.58.4.8 gst_tensor_sparse_enc_parse_caps()

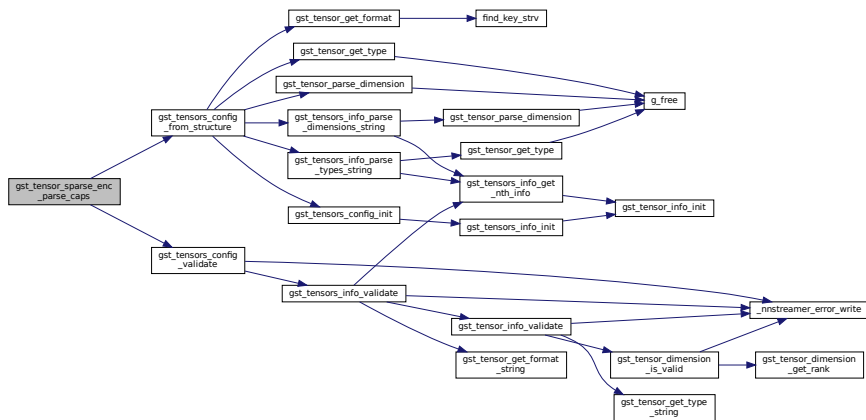
```
static gboolean gst_tensor_sparse_enc_parse_caps (
    GstTensorSparseEnc * self,
    const GstCaps * caps ) [static]
```

Parse caps and set tensors config.

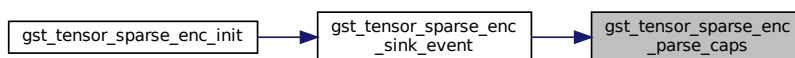
not fully configured

Definition at line 228 of file gstreamer_sparseenc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.58.4.9 gst_tensor_sparse_enc_query_caps()

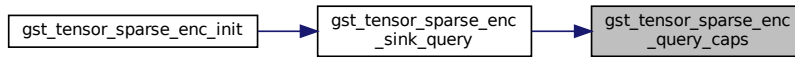
```
static GstCaps * gst_tensor_sparse_enc_query_caps (
    GstTensorSparseEnc * self,
    GstPad * pad,
    GstCaps * filter ) [static]
```

Get pad caps for caps negotiation.

pad don't have current caps. use the template caps

Definition at line 286 of file gsttensor_sparseenc.c.

Here is the caller graph for this function:



9.58.4.10 `gst_tensor_sparse_enc_set_property()`

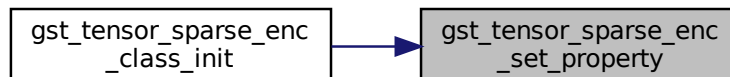
```

static void gst_tensor_sparse_enc_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

Setter for `tensor_sparse_enc` properties.

Definition at line 186 of file gsttensor_sparseenc.c.

Here is the caller graph for this function:



9.58.4.11 `gst_tensor_sparse_enc_sink_event()`

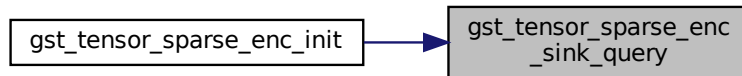
```

static gboolean gst_tensor_sparse_enc_sink_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
  
```

This function handles sink pad event.

Definition at line 254 of file gsttensor_sparseenc.c.

Here is the caller graph for this function:



9.58.5 Variable Documentation

9.58.5.1 sink_template

```
GstStaticPadTemplate sink_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (GST_TENSORS_CAP_DEFAULT))
```

Template for sink pad.

Definition at line 67 of file `gsttensor_sparseenc.c`.

9.58.5.2 src_template

```
GstStaticPadTemplate src_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (GST_TENSORS_SPARSE_CAP_DEFAULT))
```

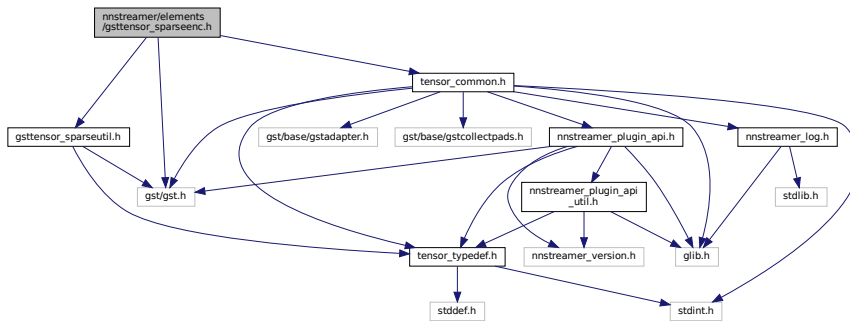
Template for src pad.

Definition at line 75 of file `gsttensor_sparseenc.c`.

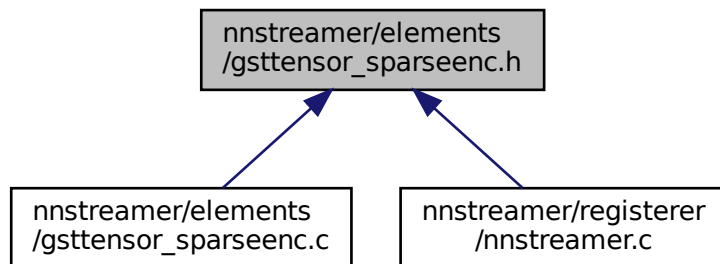
9.59 nnstreamer/elements/gsttensor_sparseenc.h File Reference

GStreamer element to encode sparse tensors into dense tensors.

```
#include <gst/gst.h>
#include <tensor_common.h>
#include "gsttensor_sparseutil.h"
Include dependency graph for gsttensor_sparseenc.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorSparseEnc](#)
GstTensorSparseEnc data structure.
- struct [_GstTensorSparseEncClass](#)
GstTensorSparseClass data structure.

Macros

- #define [GST_TYPE_TENSOR_SPARSE_ENC](#) ([gst_tensor_sparse_enc_get_type\(\)](#))
- #define [GST_TENSOR_SPARSE_ENC\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_SPARSE_ENC, G_TENSOR_SPARSE_ENC\)](#))
- #define [GST_TENSOR_SPARSE_ENC_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_SPARSE_ENC, G_TENSOR_SPARSE_ENC_CLASS\)](#))
- #define [GST_IS_TENSOR_SPARSE_ENC\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_SPARSE_ENC\)](#))
- #define [GST_IS_TENSOR_SPARSE_ENC_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_SPARSE_ENC\)](#))

Typedefs

- typedef struct [_GstTensorSparseEnc](#) [GstTensorSparseEnc](#)
- typedef struct [_GstTensorSparseEncClass](#) [GstTensorSparseEncClass](#)

Functions

- GType [gst_tensor_sparse_enc_get_type](#) (void)
Function to get type of tensor_sparse.

9.59.1 Detailed Description

GStreamer element to encode sparse tensors into dense tensors.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

27 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Yongjoo Ahn yongjool.ahn@samsung.com

Bug No known bugs except for NYI items

9.59.2 Macro Definition Documentation

9.59.2.1 GST_IS_TENSOR_SPARSE_ENC

```
#define GST_IS_TENSOR_SPARSE_ENC(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_SPARSE_ENC))
```

Definition at line 28 of file `gsttensor_sparseenc.h`.

9.59.2.2 GST_IS_TENSOR_SPARSE_ENC_CLASS

```
#define GST_IS_TENSOR_SPARSE_ENC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_SPARSE_ENC))
```

Definition at line 30 of file `gsttensor_sparseenc.h`.

9.59.2.3 GST_TENSOR_SPARSE_ENC

```
#define GST_TENSOR_SPARSE_ENC(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_SPARSE_ENC, GstTensorSparseEnc))
```

Definition at line 24 of file `gsttensor_sparseenc.h`.

9.59.2.4 GST_TENSOR_SPARSE_ENC_CLASS

```
#define GST_TENSOR_SPARSE_ENC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_SPARSE_ENC, GstTensorSparseEncClass))
```

Definition at line 26 of file `gsttensor_sparseenc.h`.

9.59.2.5 GST_TYPE_TENSOR_SPARSE_ENC

```
#define GST_TYPE_TENSOR_SPARSE_ENC (gst_tensor_sparse_enc_get_type())
```

Definition at line 22 of file `gsttensor_sparseenc.h`.

9.59.3 Typedef Documentation

9.59.3.1 GstTensorSparseEnc

```
typedef struct _GstTensorSparseEnc GstTensorSparseEnc
```

Definition at line 33 of file `gsttensor_sparseenc.h`.

9.59.3.2 GstTensorSparseEncClass

```
typedef struct _GstTensorSparseEncClass GstTensorSparseEncClass
```

Definition at line 34 of file `gsttensor_sparseenc.h`.

9.59.4 Function Documentation

9.59.4.1 `gst_tensor_sparse_enc_get_type()`

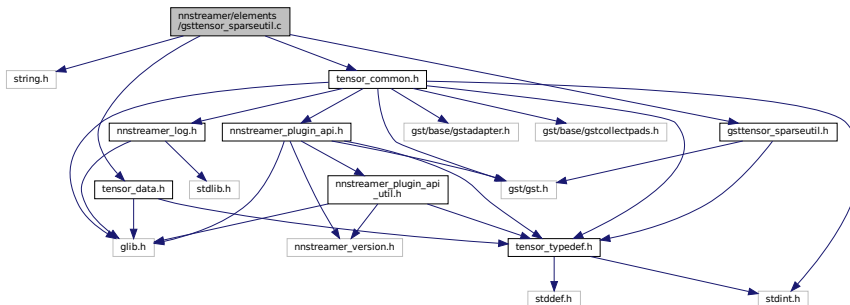
```
GType gst_tensor_sparse_enc_get_type (
    void )
```

Function to get type of `tensor_sparse`.

9.60 `nnstreamer/elements/gsttensor_sparseutil.c` File Reference

Util functions for `tensor_sparse` encoder and decoder.

```
#include <string.h>
#include <tensor_common.h>
#include <tensor_data.h>
#include "gsttensor_sparseutil.h"
Include dependency graph for gsttensor_sparseutil.c:
```



Functions

- `GstMemory * gst_tensor_sparse_to_dense (GstTensorMetaInfo *meta, GstMemory *mem)`
Make dense tensor with input sparse tensor.
- `GstMemory * gst_tensor_sparse_from_dense (GstTensorMetaInfo *meta, GstMemory *mem)`
Make sparse tensor with input dense tensor.

9.60.1 Detailed Description

Util functions for `tensor_sparse` encoder and decoder.

GStreamer / NNStreamer Sparse Tensor support Copyright (C) 2021 Yongjoo Ahn yongjool.ahn@samsung.com

Date

27 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Yongjoo Ahn yongjool.ahn@samsung.com

Bug No known bugs except for NYI items

9.60.2 Function Documentation

9.60.2.1 `gst_tensor_sparse_from_dense()`

```
GstMemory* gst_tensor_sparse_from_dense (
    GstTensorMetaInfo * meta,
    GstMemory * mem )
```

Make sparse tensor with input dense tensor.

Parameters

<code>in, out</code>	<code>meta</code>	tensor meta structure to be updated
<code>in</code>	<code>mem</code>	gst-memory of dense tensor data

Returns

pointer of `GstMemory` with sparse tensor data or `NULL` on error. Caller should handle this newly allocated memory.

alloc maximum possible size of memory

Consider using macro to reduce loc and readability

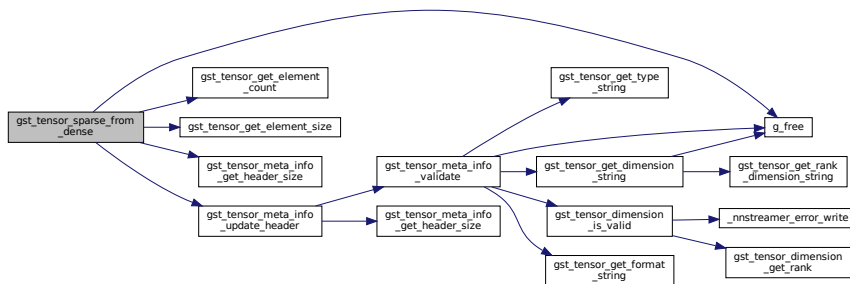
update meta nnz info

write to output buffer

add meta info header

Definition at line 116 of file `gsttensor_sparseutil.c`.

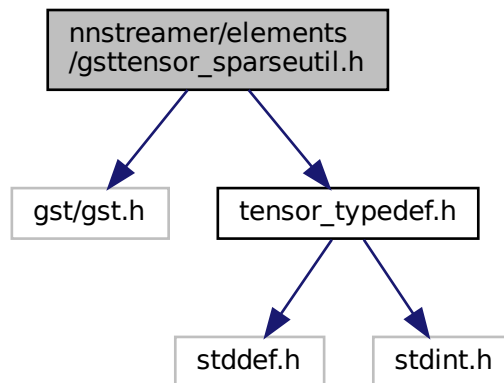
Here is the call graph for this function:



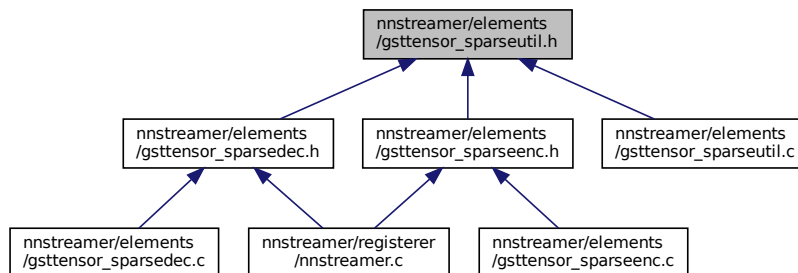
9.61 nnstreamer/elements/gsttensor_sparseutil.h File Reference

Util functions for tensor_sparse encoder and decoder.

```
#include <gst/gst.h>
#include <tensor_typedef.h>
Include dependency graph for gsttensor_sparseutil.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- `G_BEGIN_DECLS GstMemory * gst_tensor_sparse_to_dense (GstTensorMetaInfo *meta, GstMemory *mem)`
Make dense tensor with input sparse tensor.
- `GstMemory * gst_tensor_sparse_from_dense (GstTensorMetaInfo *meta, GstMemory *mem)`
Make sparse tensor with input dense tensor.

9.61.1 Detailed Description

Util functions for tensor_sparse encoder and decoder.

GStreamer / NNStreamer Sparse Tensor support Copyright (C) 2021 Yongjoo Ahn yongjoo1.ahn@samsung.com

Date

06 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Yongjoo Ahn yongjoo1.ahn@samsung.com

Bug No known bugs except for NYI items

9.61.2 Function Documentation

9.61.2.1 `gst_tensor_sparse_from_dense()`

```
GstMemory* gst_tensor_sparse_from_dense (
    GstTensorMetaInfo * meta,
    GstMemory * mem )
```

Make sparse tensor with input dense tensor.

Parameters

<code>in, out</code>	<code>meta</code>	tensor meta structure to be updated
<code>in</code>	<code>mem</code>	gst-memory of dense tensor data

Returns

pointer of GstMemory with sparse tensor data or NULL on error. Caller should handle this newly allocated memory.

alloc maximum possible size of memory

Consider using macro to reduce loc and readability

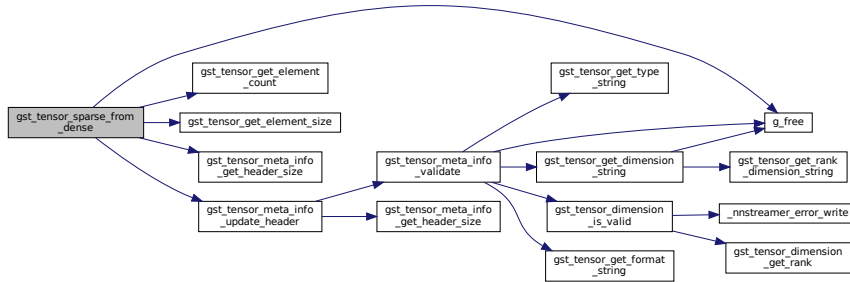
update meta nnz info

write to output buffer

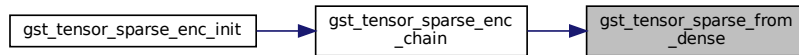
add meta info header

Definition at line 116 of file gsttensor_sparseutil.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.61.2.2 gst_tensor_sparse_to_dense()

```

G_BEGIN_DECLS GstMemory* gst_tensor_sparse_to_dense (
    GstTensorMetaInfo * meta,
    GstMemory * mem )
    
```

Make dense tensor with input sparse tensor.

Parameters

in, out	<i>meta</i>	tensor meta structure to be updated
in	<i>mem</i>	gst-memory of sparse tensor data

Returns

pointer of GstMemory with dense tensor data or NULL on error. Caller should handle this newly allocated memory.

Definition at line 27 of file gsttensor_sparseutil.c.

Enumerations

- enum { [PROP_0](#), [PROP_SILENT](#), [PROP_TENSORPICK](#), [PROP_TENSORSEG](#) }

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_split_debug](#))
- static GstFlowReturn [gst_tensor_split_chain](#) (GstPad *pad, GstObject *parent, GstBuffer *buf)
chain function for sink (gst element vmethod)
- static gboolean [gst_tensor_split_event](#) (GstPad *pad, GstObject *parent, GstEvent *event)
event function for sink (gst element vmethod)
- static GstStateChangeReturn [gst_tensor_split_change_state](#) (GstElement *element, GstStateChange transition)
change state (gst element vmethod)
- static void [gst_tensor_split_set_property](#) (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
Get property (gst element vmethod)
- static void [gst_tensor_split_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
Get property (gst element vmethod)
- static void [gst_tensor_split_finalize](#) (GObject *object)
finalize function for tensor split (gst element vmethod)
- [G_DEFINE_TYPE](#) (GstTensorSplit, gst_tensor_split, GST_TYPE_ELEMENT)
- static void [gst_tensor_split_class_init](#) (GstTensorSplitClass *klass)
initialize the tensor_split's class
- static void [gst_tensor_split_init](#) (GstTensorSplit *split)
initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure
- static void [gst_tensor_split_remove_src_pads](#) (GstTensorSplit *split)
function to remove srcpad list
- static gboolean [gst_tensor_split_get_capsparam](#) (GstTensorSplit *split, GstCaps *caps)
Set Caps in pad.
- static GstTensorPad * [gst_tensor_split_get_tensor_pad](#) (GstTensorSplit *split, GstBuffer *inbuf, gboolean *created, guint nth)
Checking if the source pad is created and if not, create TensorPad.
- static GstFlowReturn [gst_tensor_split_combine_flows](#) (GstTensorSplit *split, GstTensorPad *pad, GstFlowReturn ret)
Check the status among sources in split.
- static GstMemory * [gst_tensor_split_get_splitted](#) (GstTensorSplit *split, GstBuffer *buffer, gint nth)
Make splitted tensor.
- static void [_clear_tensorseg](#) (tensor_dim **element)
Glib Array Clear Function.

Variables

- static GstStaticPadTemplate [src_tmpl](#)
the capabilities of the inputs and outputs. describe the real formats here.
- static GstStaticPadTemplate [sink_tmpl](#)

9.62.1 Detailed Description

GStreamer plugin to split tensor (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

27 Aug 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.62.2 Macro Definition Documentation

9.62.2.1 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_WITH_NUM ("1")
```

Template caps string.

Definition at line 73 of file gsttensor_split.c.

9.62.2.2 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_split_debug
```

Definition at line 60 of file gsttensor_split.c.

9.62.2.3 `gst_tensor_split_parent_class`

```
#define gst_tensor_split_parent_class parent_class
```

Definition at line 101 of file `gsttensor_split.c`.

9.62.3 Enumeration Type Documentation

9.62.3.1 anonymous enum

anonymous enum

Enumerator

PROP_0	
PROP_SILENT	
PROP_TENSORPICK	
PROP_TENSORSEG	

Definition at line 62 of file `gsttensor_split.c`.

9.62.4 Function Documentation

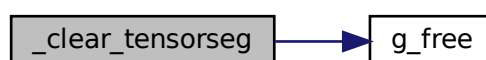
9.62.4.1 `_clear_tensorseg()`

```
static void _clear_tensorseg (  
    tensor_dim ** element ) [static]
```

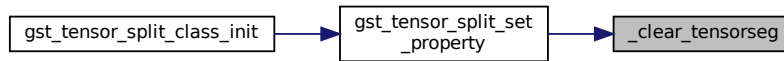
Glib Array Clear Function.

Definition at line 569 of file `gsttensor_split.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.4.2 G_DEFINE_TYPE()

```

G_DEFINE_TYPE (
    GstTensorSplit ,
    gst_tensor_split ,
    GST_TYPE_ELEMENT )
  
```

9.62.4.3 GST_DEBUG_CATEGORY_STATIC()

```

GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_split_debug )
  
```

SECTION:element-tensor_split

A Deuxer that split tensors stream to tensor stream for NN frameworks. The outputs are always in the format of other/tensor.

```

<refsect2> <title>Example launch line</title> |[ gst-launch -v -m filesrc location=testcase_RGB_100x100.png !
pngdec ! videoscale ! imagefreeze ! videoconvert ! video/x-raw,format=RGB,width=100,height=100,framerate=0/1
! tensor_converter ! tensor_split name=split tensorseg=2:100:100,1:100:100 split.src_0 ! queue ! filesink
location=src0.log split.src_1 ! queue ! filesink location=src1.log ]|
  
```

</refsect2>

9.62.4.4 gst_tensor_split_chain()

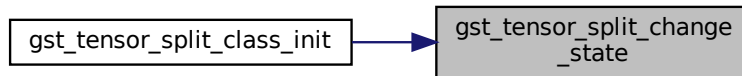
```

static GstFlowReturn gst_tensor_split_chain (
    GstPad * pad,
    GstObject * parent,
    GstBuffer * buf ) [static]
  
```

chain function for sink (gst element vmethod)

Definition at line 459 of file gsttensor_split.c.

Here is the caller graph for this function:



9.62.4.6 gst_tensor_split_class_init()

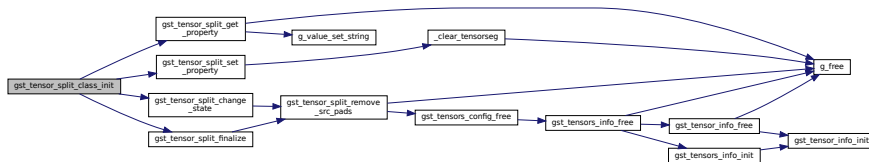
```

static void gst_tensor_split_class_init (
    GstTensorSplitClass * klass ) [static]
  
```

initialize the tensor_split's class

Definition at line 109 of file gsttensor_split.c.

Here is the call graph for this function:



9.62.4.7 gst_tensor_split_combine_flows()

```

static GstFlowReturn gst_tensor_split_combine_flows (
    GstTensorSplit * split,
    GstTensorPad * pad,
    GstFlowReturn ret ) [static]
  
```

Check the status among sources in split.

Parameters

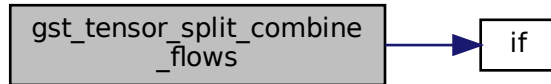
<i>split</i>	TensorSplit Object
<i>TensorPad</i>	Tensorpad
<i>ret</i>	return status of current pad

Returns

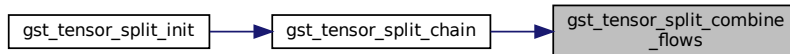
return status after check sources

Definition at line 387 of file gsttensor_split.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.4.8 gst_tensor_split_event()

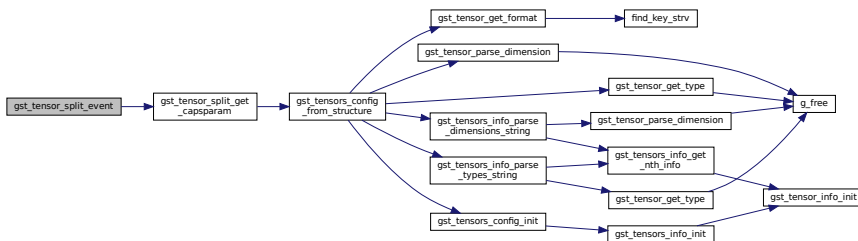
```

static gboolean gst_tensor_split_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
  
```

event function for sink (gst element vmethod)

Definition at line 233 of file gsttensor_split.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.4.9 `gst_tensor_split_finalize()`

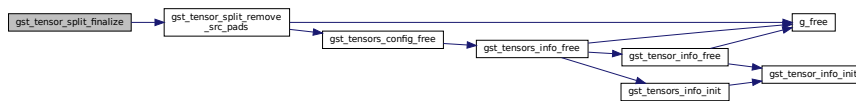
```

static void gst_tensor_split_finalize (
    GObject * object ) [static]
  
```

finalize function for tensor split (gst element vmethod)

Definition at line 202 of file `gsttensor_split.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.4.10 `gst_tensor_split_get_capsparam()`

```

static gboolean gst_tensor_split_get_capsparam (
    GstTensorSplit * split,
    GstCaps * caps ) [static]
  
```

Set Caps in pad.

Parameters

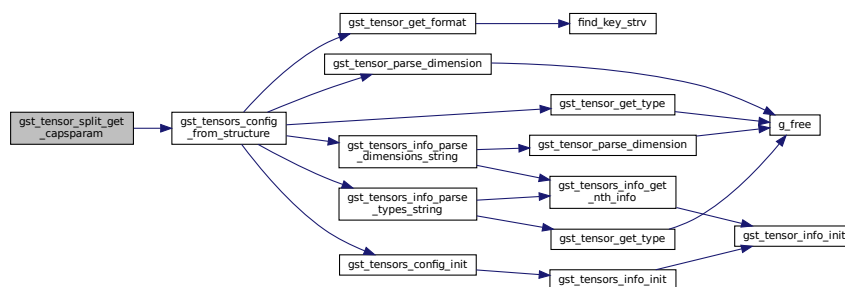
<i>split</i>	GstTensorSplit object
<i>caps</i>	incoming capability

Returns

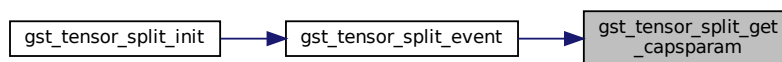
TRUE/FALSE (if successfully generate & set cap, return TRUE)

Definition at line 220 of file gsttensor_split.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.4.11 gst_tensor_split_get_property()

```

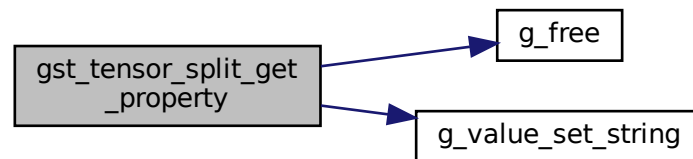
static void gst_tensor_split_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
  
```

Get property (gst element vmethod)

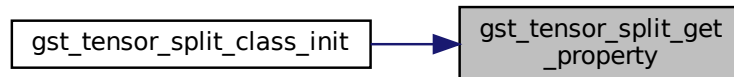
If i = 1, this is previous p. Otherwise, it's previous g_strjoin result.

Definition at line 653 of file gsttensor_split.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.4.12 `gst_tensor_split_get_splitted()`

```

static GstMemory* gst_tensor_split_get_splitted (
    GstTensorSplit * split,
    GstBuffer * buffer,
    gint nth ) [static]
  
```

Make splitted tensor.

Parameters

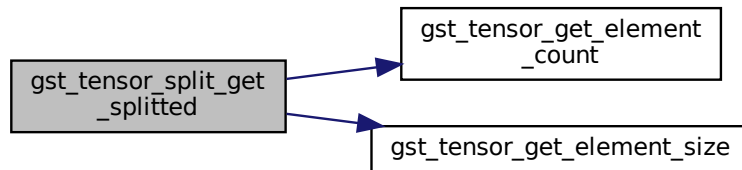
<i>split</i>	TensorSplit object
<i>buffer</i>	gstbuffer form src
<i>nth</i>	orther of tensor

Returns

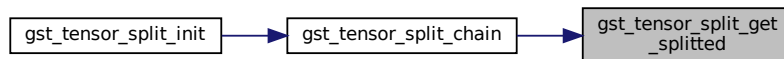
return GstMemory for splitted tensor

Definition at line 415 of file `gsttensor_split.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.4.13 `gst_tensor_split_get_tensor_pad()`

```

static GstTensorPad* gst_tensor_split_get_tensor_pad (
    GstTensorSplit * split,
    GstBuffer * inbuf,
    gboolean * created,
    guint nth ) [static]
  
```

Checking if the source pad is created and if not, create TensorPad.

Parameters

	<i>split</i>	TensorSplit Object
	<i>inbuf</i>	inputbuf GstBuffer Object including GstMeta
out	<i>created</i>	will be updated in this function
	<i>nth</i>	source ordering

Returns

TensorPad if pad is already created, then return created pad. If not return new pad after creation.

Definition at line 276 of file `gsttensor_split.c`.

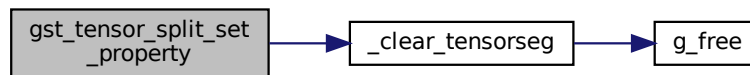
9.62.4.16 `gst_tensor_split_set_property()`

```
static void gst_tensor_split_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

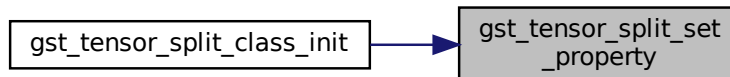
Get property (gst element vmethod)

Definition at line 578 of file `gsttensor_split.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.62.5 Variable Documentation

9.62.5.1 `sink_tmpl`

```
GstStaticPadTemplate sink_tmpl [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (CAPS_STRING))
```

Definition at line 84 of file `gsttensor_split.c`.

9.62.5.2 src_tmpl

```
GstStaticPadTemplate src_tmpl [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src_%u",
    GST_PAD_SRC,
    GST_PAD_SOMETIMES,
    GST_STATIC_CAPS (CAPS_STRING))
```

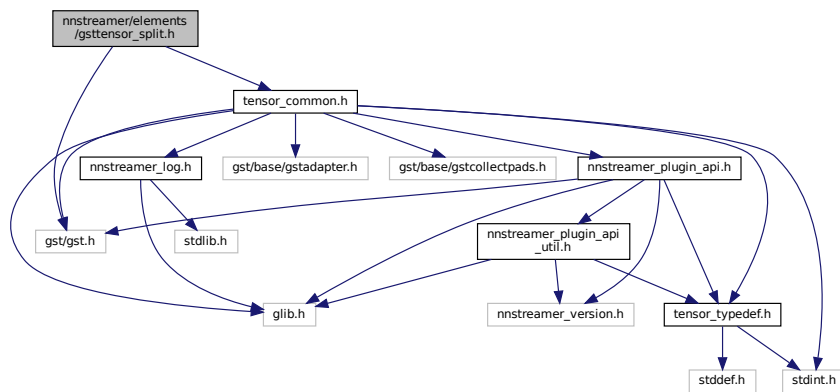
the capabilities of the inputs and outputs. describe the real formats here.

Definition at line 79 of file gsttensor_split.c.

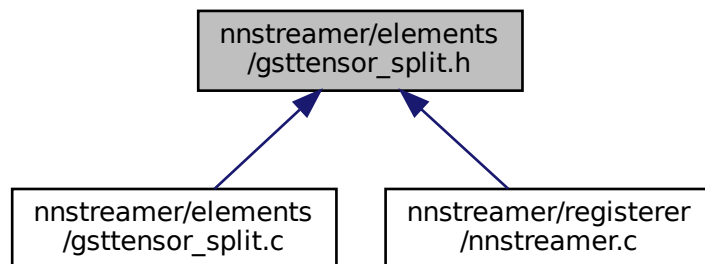
9.63 nnstreamer/elements/gsttensor_split.h File Reference

GStreamer plugin to split tensor (as a filter for other general neural network filters)

```
#include <gst/gst.h>
#include <tensor_common.h>
Include dependency graph for gsttensor_split.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorSplit](#)
Tensor Split data structure.
- struct [_GstTensorSplitClass](#)
GstTensorSplitClass inherits GstElementClass.

Macros

- #define [GST_TYPE_TENSOR_SPLIT](#) ([gst_tensor_split_get_type](#) ())
- #define [GST_TENSOR_SPLIT](#)(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj), [GST_TYPE_TENSOR_SPLIT](#), [GstTensorSplit](#)))
- #define [GST_TENSOR_SPLIT_CLASS](#)(klass) (G_TYPE_CHECK_CLASS_CAST ((klass), [GST_TYPE_TENSOR_SPLIT](#), [GstTensorSplitClass](#)))
- #define [GST_TENSOR_SPLIT_GET_CLASS](#)(obj) (G_TYPE_INSTANCE_GET_CLASS ((obj), [GST_TYPE_TENSOR_SPLIT](#), [GstTensorSplitClass](#)))
- #define [GST_IS_TENSOR_SPLIT](#)(obj) (G_TYPE_CHECK_INSTANCE_TYPE((obj), [GST_TYPE_TENSOR_SPLIT](#)))
- #define [GST_IS_TENSOR_SPLIT_CLASS](#)(klass) (G_TYPE_CHECK_CLASS_TYPE((klass), [GST_TYPE_TENSOR_SPLIT](#)))
- #define [GST_TENSOR_SPLIT_CAST](#)(obj) (([GstTensorSplit*](#))(obj))

Typedefs

- typedef struct [_GstTensorSplit](#) [GstTensorSplit](#)
- typedef struct [_GstTensorSplitClass](#) [GstTensorSplitClass](#)

Functions

- GType [gst_tensor_split_get_type](#) (void)
Get Type function required for gst elements.

9.63.1 Detailed Description

GStreamer plugin to split tensor (as a filter for other general neural network filters)

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 Jijoong Moon jijoong.moon@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

27 Aug 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Jijoong Moon jijoong.moon@samsung.com

Bug No known bugs except for NYI items

9.63.2 Macro Definition Documentation

9.63.2.1 GST_IS_TENSOR_SPLIT

```
#define GST_IS_TENSOR_SPLIT(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_TENSOR_SPLIT))
```

Definition at line 40 of file gsttensor_split.h.

9.63.2.2 GST_IS_TENSOR_SPLIT_CLASS

```
#define GST_IS_TENSOR_SPLIT_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE ((klass), GST_TYPE_TENSOR_SPLIT))
```

Definition at line 41 of file gsttensor_split.h.

9.63.2.3 GST_TENSOR_SPLIT

```
#define GST_TENSOR_SPLIT(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST ((obj), GST_TYPE_TENSOR_SPLIT, GstTensorSplit))
```

Definition at line 37 of file gsttensor_split.h.

9.63.2.4 GST_TENSOR_SPLIT_CAST

```
#define GST_TENSOR_SPLIT_CAST(  
    obj ) ((GstTensorSplit*) (obj))
```

Definition at line 42 of file gsttensor_split.h.

9.63.2.5 GST_TENSOR_SPLIT_CLASS

```
#define GST_TENSOR_SPLIT_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST ((klass), GST_TYPE_TENSOR_SPLIT, GstTensorSplitClass))
```

Definition at line 38 of file gsttensor_split.h.

9.63.2.6 GST_TENSOR_SPLIT_GET_CLASS

```
#define GST_TENSOR_SPLIT_GET_CLASS(  
    obj ) (G_TYPE_INSTANCE_GET_CLASS ((obj), GST_TYPE_TENSOR_SPLIT, GstTensorSplitClass))
```

Definition at line 39 of file gsttensor_split.h.

9.63.2.7 GST_TYPE_TENSOR_SPLIT

```
#define GST_TYPE_TENSOR_SPLIT (gst_tensor_split_get_type ())
```

Definition at line 36 of file gsttensor_split.h.

9.63.3 Typedef Documentation

9.63.3.1 GstTensorSplit

```
typedef struct _GstTensorSplit GstTensorSplit
```

Definition at line 44 of file gsttensor_split.h.

9.63.3.2 GstTensorSplitClass

```
typedef struct _GstTensorSplitClass GstTensorSplitClass
```

Definition at line 45 of file gsttensor_split.h.

9.63.4 Function Documentation

9.63.4.1 gst_tensor_split_get_type()

```
GType gst_tensor_split_get_type (  
    void )
```

Get Type function required for gst elements.

- #define `MIN_FREQUENCY` 0
 - Minimum and maximum operating frequency for the device Frequency 0 chooses the first available frequency supported by device.*
- #define `MAX_FREQUENCY` `G_MAXULONG`
- #define `DEFAULT_FREQUENCY` 0
- #define `MIN_POLL_TIMEOUT` -1
 - Minimum and maximum polling timeout for the buffered reading.*
- #define `MAX_POLL_TIMEOUT` `G_MAXINT`
- #define `DEFAULT_POLL_TIMEOUT` 10000
- #define `DEFAULT_MERGE_CHANNELS` `TRUE`
 - Default behavior on merging channels.*
- #define `DEFAULT_PROP_DEVICE_NUM` -1
 - default trigger and device numbers*
- #define `DEFAULT_PROP_TRIGGER_NUM` -1
- #define `BLOCKSIZE` 1
- #define `DEVICE` "device"
 - IIO devices/triggers.*
- #define `BUFFER` "buffer"
- #define `TRIGGER` "trigger"
- #define `CHANNELS` "scan_elements"
- #define `IIO` "iio:"
- #define `TIMESTAMP` "timestamp"
- #define `DEVICE_PREFIX` `IIO DEVICE`
- #define `TRIGGER_PREFIX` `IIO TRIGGER`
- #define `CURRENT_TRIGGER` "current_trigger"
- #define `EN_SUFFIX` "_en"
 - IIO device channels.*
- #define `INDEX_SUFFIX` "_index"
- #define `TYPE_SUFFIX` "_type"
- #define `SCALE_SUFFIX` "_scale"
- #define `OFFSET_SUFFIX` "_offset"
- #define `NAME_FILE` "name"
 - filenames for IIO devices/triggers characteristics*
- #define `AVAIL_FREQUENCY_FILE` "sampling_frequency_available"
- #define `SAMPLING_FREQUENCY` "sampling_frequency"
- #define `gst_tensor_src_iio_parent_class` `parent_class`

Enumerations

- enum {
 - `PROP_0`, `PROP_MODE`, `PROP_SILENT`, `PROP_BASE_DIRECTORY`,
 - `PROP_DEV_DIRECTORY`, `PROP_DEVICE`, `PROP_DEVICE_NUM`, `PROP_TRIGGER`,
 - `PROP_TRIGGER_NUM`, `PROP_CHANNELS`, `PROP_BUFFER_CAPACITY`, `PROP_FREQUENCY`,
 - `PROP_MERGE_CHANNELS`, `PROP_POLL_TIMEOUT` }
 - tensor_src_iio properties.*

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_src_iio_debug](#))
- [g_assert](#) (`sizeof(DTYPE_UNSIGNED)==sizeof(DTYPE_SIGNED)`)
- [value_unsigned](#) & [if](#) (`prop->is_signed`)
- [PROCESS_SCANNED_DATA](#) (`guint8, gint8`)
- [PROCESS_SCANNED_DATA](#) (`guint16, gint16`)
- [PROCESS_SCANNED_DATA](#) (`guint32, gint32`)
- [PROCESS_SCANNED_DATA](#) (`guint64, gint64`)
- static void [gst_tensor_src_iio_set_property](#) (`GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec`)
set tensor_src_iio properties
- static void [gst_tensor_src_iio_get_property](#) (`GObject *object, guint prop_id, GValue *value, GParamSpec *pspec`)
get tensor_src_iio properties
- static void [gst_tensor_src_iio_finalize](#) (`GObject *object`)
finalize the instance
- static gboolean [gst_tensor_src_iio_start](#) (`GstBaseSrc *src`)
start function, called when state changed null to ready. load the device and init the device resources
- static gboolean [gst_tensor_src_iio_stop](#) (`GstBaseSrc *src`)
stop function, called when state changed ready to null.
- static `GstStateChangeReturn` [gst_tensor_src_iio_change_state](#) (`GstElement *element, GstStateChange transition`)
Perform state change.
- static gboolean [gst_tensor_src_iio_event](#) (`GstBaseSrc *src, GstEvent *event`)
handle events
- static gboolean [gst_tensor_src_iio_set_caps](#) (`GstBaseSrc *src, GstCaps *caps`)
set new caps
- static `GstCaps *` [gst_tensor_src_iio_get_caps](#) (`GstBaseSrc *src, GstCaps *filter`)
get caps of subclass
- static `GstCaps *` [gst_tensor_src_iio_fixate](#) (`GstBaseSrc *src, GstCaps *caps`)
fixate the caps when needed during negotiation
- static gboolean [gst_tensor_src_iio_is_seekable](#) (`GstBaseSrc *src`)
check if source supports seeking
- static `GstFlowReturn` [gst_tensor_src_iio_create](#) (`GstBaseSrc *src, guint64 offset, guint size, GstBuffer **buffer`)
create a buffer with requested size and offset
- static `GstFlowReturn` [gst_tensor_src_iio_fill](#) (`GstBaseSrc *src, guint64 offset, guint size, GstBuffer *buffer`)
fill the buffer with data
- static void [gst_tensor_src_iio_get_times](#) (`GstBaseSrc *basesrc, GstBuffer *buffer, GstClockTime *start, GstClockTime *end`)
returns the time for the buffers
- [G_DEFINE_TYPE](#) (`GstTensorSrcIIO, gst_tensor_src_iio, GST_TYPE_BASE_SRC`)
- static void [gst_tensor_src_iio_class_init](#) (`GstTensorSrcIIOClass *klass`)
initialize the tensor_src_iio class.
- static void [gst_tensor_src_iio_channel_properties_free](#) (`gpointer data`)
delete GstTensorSrcIIODeviceProperties structure
- static void [gst_tensor_src_iio_device_properties_init](#) (`GstTensorSrcIIODeviceProperties *prop`)
initialize GstTensorSrcIIODeviceProperties structure
- static void [gst_tensor_src_iio_init](#) (`GstTensorSrcIIO *self`)
initialize tensor_src_iio element.
- static `gint` [gst_tensor_src_merge_tensor_by_type](#) (`GstTensorInfo *info, guint size, guint dir`)

- merge multiple other/tensor*

 - static gint [gst_tensor_src_iio_get_id_by_name](#) (const gchar *dir_name, const gchar *name, const gchar *prefix)

check if device/trigger with the given name exists
- static gchar * [gst_tensor_src_iio_get_name_by_id](#) (const gchar *dir_name, const gint id, const gchar *prefix)

check if device/trigger with the given id exists
- static gboolean [gst_tensor_src_iio_get_float_from_file](#) (const gchar *dirname, const gchar *name, const gchar *suffix, gfloat *value)

parse float value from the file
- static gboolean [gst_tensor_src_iio_set_channel_type](#) (GstTensorSrcIIOChannelProperties *prop, const gchar *contents)

get type info about the channel from the string
- static gchar * [gst_tensor_src_iio_get_generic_name](#) (const gchar *channel_name)

get generic name for channel from the string
- static gint [gst_tensor_channel_list_sort_cmp](#) (gconstpointer a, gconstpointer b)

compare channels for sort based on their indices
- static void [gst_tensor_channel_list_filter_enabled](#) (gpointer data, gpointer user_data)

compare channels for filtering if enabled
- static gint [gst_tensor_src_iio_get_all_channel_info](#) (GstTensorSrcIIO *self, const gchar *dir_name)

get info about all the channels in the device
- static gint64 [gst_tensor_src_iio_get_available_frequency](#) (const gchar *base_dir, const guint64 frequency)

return sampling frequency given the frequency input from user
- static gboolean [gst_tensor_write_sysfs_string](#) (GstTensorSrcIIO *self, const gchar *file, const gchar *base_dir, const gchar *contents)

write the string in to the file
- static gboolean [gst_tensor_write_sysfs_int](#) (GstTensorSrcIIO *self, const gchar *file, const gchar *base_dir, const gint contents)

write the int in to the file
- static gboolean [gst_tensor_set_all_channels](#) (GstTensorSrcIIO *self, const gint contents)

set value to all the channels
- static guint [gst_tensor_get_size_from_channels](#) (GList *channels)

get the size of the combined data from channels
- static gboolean [gst_tensor_src_iio_create_config](#) (GstTensorSrcIIO *tensor_src_iio)

create the structure for the caps to update the src pad caps
- static gboolean [gst_tensor_src_iio_setup_device_properties](#) (GstTensorSrcIIO *self)

setup device using name/id
- static gboolean [gst_tensor_src_iio_setup_trigger_properties](#) (GstTensorSrcIIO *self)

setup trigger using name/id
- static gboolean [gst_tensor_src_iio_setup_sampling_frequency](#) (GstTensorSrcIIO *self)

setup device sampling frequency
- static gboolean [gst_tensor_src_iio_setup_scan_channels](#) (GstTensorSrcIIO *self)

setup scan channels for the device
- static gboolean [gst_tensor_src_iio_setup_device_buffer](#) (GstTensorSrcIIO *self)

setup device using name/id
- static void [gst_tensor_src_restore_iio_device](#) (GstTensorSrcIIO *self)

restore the iio device to its original device.
- static gboolean [gst_tensor_src_iio_process_scanned_data](#) (GstTensorSrcIIOChannelProperties *prop, gchar *data, gfloat *buffer_map)

process the scanned data from IIO device

Variables

- [GstTensorSrcIOChannelProperties](#) * prop
DTYPE_UNSIGNED (.
- [GstTensorSrcIOChannelProperties](#) DTYPE_UNSIGNED value_unsigned
- else
- return value_float

9.64.1 Detailed Description

GStreamer plugin to capture sensor data as tensor(s)

GStreamer Tensor_Source_IIO Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2019 Parichay Kapoor pk.kapoor@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

27 Feb 2019

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

Bug No known bugs except for NYI items

Todo support specific channels as input
handle timestamp received from device

This is the plugin to capture data from sensors and convert them to tensor format. Current implementation will support accelerators, light and gyro sensors.

9.64.2 Macro Definition Documentation

9.64.2.1 AVAIL_FREQUENCY_FILE

```
#define AVAIL_FREQUENCY_FILE "sampling_frequency_available"
```

Definition at line 244 of file gsttensor_srcio.c.

9.64.2.2 BLOCKSIZE

```
#define BLOCKSIZE 1
```

blocksize for buffer

Definition at line 216 of file gsttensor_srcio.c.

9.64.2.3 BUFFER

```
#define BUFFER "buffer"
```

Definition at line 222 of file gsttensor_srcio.c.

9.64.2.4 CHANNELS

```
#define CHANNELS "scan_elements"
```

Definition at line 224 of file gsttensor_srcio.c.

9.64.2.5 CHANNELS_ENABLED_ALL_CHAR

```
#define CHANNELS_ENABLED_ALL_CHAR "all"
```

Definition at line 160 of file gsttensor_srcio.c.

9.64.2.6 CHANNELS_ENABLED_AUTO_CHAR

```
#define CHANNELS_ENABLED_AUTO_CHAR "auto"
```

iio device channel enabled mode

Definition at line 159 of file gsttensor_srcio.c.

9.64.2.7 CURRENT_TRIGGER

```
#define CURRENT_TRIGGER "current_trigger"
```

Definition at line 229 of file gstreamer_srcii.c.

9.64.2.8 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

SECTION:element-tensor_src_ii

#tensor_src_ii extends #gstbasesrc source element to handle Linux Industrial I/O sensors as input. IIO sources are only supported in buffered mode. Source elements only support push mode scheduling as a live source.

#tensor_src_ii supports configuring the device as well as the trigger via properties. Buffer capacity, frequency and scan channels to be read can be configured before PLAYING the stream. The configuration is supported only in states <= READY. Setting the state back to NULL restores the original configuration of the IIO device. The source can be configured to work with trigger for the source or read the data from the device at regular time intervals. Device name/number is the only necessary configuration needed to run the element (other configuration parameters is optional).

The output caps is either of other/tensor or other/tensors.

Data from various channels can be merged to form 1 other/tensor. Final caps of the src pad is of the following format: <itemizedlist> <listitem>

Dimension 0 : Channel number</listitem>

<listitem>

Dimension 1 : buffer capacity</listitem>

</itemizedlist> Other dimensions are not utilized. The data in the dimension 0 is sorted on the basis of the indexing of the channels provided by the IIO device.

The enabling of buffer for data capture is performed when transitioning from PAUSED to PLAYING state. This leads to automated synchronization handled by gstreamer. Buffer duration and timestamps set by #gstbasesrc remain in sync with linux IIO timestamps.

```
<refsect2> <title>Example launch line</title> |[ gst-launch -v -m tensor_src_ii device-number=0 ! fakesink ]|</refsect2>
```

Definition at line 97 of file gstreamer_srcii.c.

9.64.2.9 DEFAULT_BUFFER_CAPACITY

```
#define DEFAULT_BUFFER_CAPACITY 1
```

Definition at line 185 of file gstreamer_srcii.c.

9.64.2.10 DEFAULT_FREQUENCY

```
#define DEFAULT_FREQUENCY 0
```

Definition at line 193 of file gsttensor_srcio.c.

9.64.2.11 DEFAULT_MERGE_CHANNELS

```
#define DEFAULT_MERGE_CHANNELS TRUE
```

Default behavior on merging channels.

Definition at line 205 of file gsttensor_srcio.c.

9.64.2.12 DEFAULT_OPERATING_CHANNELS_ENABLED

```
#define DEFAULT_OPERATING_CHANNELS_ENABLED CHANNELS_ENABLED_AUTO_CHAR
```

Definition at line 161 of file gsttensor_srcio.c.

9.64.2.13 DEFAULT_OPERATING_MODE

```
#define DEFAULT_OPERATING_MODE MODE_CONTINUOUS
```

Definition at line 168 of file gsttensor_srcio.c.

9.64.2.14 DEFAULT_POLL_TIMEOUT

```
#define DEFAULT_POLL_TIMEOUT 10000
```

Definition at line 200 of file gsttensor_srcio.c.

9.64.2.15 DEFAULT_PROP_BASE_DIRECTORY

```
#define DEFAULT_PROP_BASE_DIRECTORY "/sys/bus/iio/devices"
```

IIO system paths.

Definition at line 153 of file gsttensor_srcio.c.

9.64.2.16 DEFAULT_PROP_DEV_DIRECTORY

```
#define DEFAULT_PROP_DEV_DIRECTORY "/dev"
```

Definition at line 154 of file gstreamer_srcio.c.

9.64.2.17 DEFAULT_PROP_DEVICE_NUM

```
#define DEFAULT_PROP_DEVICE_NUM -1
```

default trigger and device numbers

Definition at line 210 of file gstreamer_srcio.c.

9.64.2.18 DEFAULT_PROP_SILENT

```
#define DEFAULT_PROP_SILENT TRUE
```

Flag to print minimized log.

Definition at line 173 of file gstreamer_srcio.c.

9.64.2.19 DEFAULT_PROP_STRING

```
#define DEFAULT_PROP_STRING NULL
```

Flag for general default value of string.

Definition at line 178 of file gstreamer_srcio.c.

9.64.2.20 DEFAULT_PROP_TRIGGER_NUM

```
#define DEFAULT_PROP_TRIGGER_NUM -1
```

Definition at line 211 of file gstreamer_srcio.c.

9.64.2.21 DEVICE

```
#define DEVICE "device"
```

IIO devices/triggers.

Definition at line 221 of file gsttensor_srciiio.c.

9.64.2.22 DEVICE_PREFIX

```
#define DEVICE_PREFIX IIO DEVICE
```

Definition at line 227 of file gsttensor_srciiio.c.

9.64.2.23 EN_SUFFIX

```
#define EN_SUFFIX "_en"
```

IIO device channels.

Definition at line 234 of file gsttensor_srciiio.c.

9.64.2.24 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_src_iio_debug
```

Definition at line 101 of file gsttensor_srciiio.c.

9.64.2.25 gst_tensor_src_iio_parent_class

```
#define gst_tensor_src_iio_parent_class parent_class
```

internal functions

Definition at line 280 of file gsttensor_srciiio.c.

9.64.2.26 IIO

```
#define IIO "iio:"
```

Definition at line 225 of file gsttensor_srcii.c.

9.64.2.27 INDEX_SUFFIX

```
#define INDEX_SUFFIX "_index"
```

Definition at line 235 of file gsttensor_srcii.c.

9.64.2.28 MAX_BUFFER_CAPACITY

```
#define MAX_BUFFER_CAPACITY G_MAXUINT
```

Definition at line 184 of file gsttensor_srcii.c.

9.64.2.29 MAX_FREQUENCY

```
#define MAX_FREQUENCY G_MAXULONG
```

Definition at line 192 of file gsttensor_srcii.c.

9.64.2.30 MAX_POLL_TIMEOUT

```
#define MAX_POLL_TIMEOUT G_MAXINT
```

Definition at line 199 of file gsttensor_srcii.c.

9.64.2.31 MIN_BUFFER_CAPACITY

```
#define MIN_BUFFER_CAPACITY 1
```

Minimum and maximum buffer length for iio.

Definition at line 183 of file gsttensor_srcii.c.

9.64.2.32 MIN_FREQUENCY

```
#define MIN_FREQUENCY 0
```

Minimum and maximum operating frequency for the device Frequency 0 chooses the first available frequency supported by device.

Definition at line 191 of file gsttensor_srciiio.c.

9.64.2.33 MIN_POLL_TIMEOUT

```
#define MIN_POLL_TIMEOUT -1
```

Minimum and maximum polling timeout for the buffered reading.

Definition at line 198 of file gsttensor_srciiio.c.

9.64.2.34 MODE_CONTINUOUS

```
#define MODE_CONTINUOUS "continuous"
```

Definition at line 167 of file gsttensor_srciiio.c.

9.64.2.35 MODE_ONE_SHOT

```
#define MODE_ONE_SHOT "one-shot"
```

tensor_src_iiio device modes

Definition at line 166 of file gsttensor_srciiio.c.

9.64.2.36 NAME_FILE

```
#define NAME_FILE "name"
```

filenames for IIO devices/triggers characteristics

Definition at line 243 of file gsttensor_srciiio.c.

9.64.2.37 OFFSET_SUFFIX

```
#define OFFSET_SUFFIX "_offset"
```

Definition at line 238 of file gstreamer_srcio.c.

9.64.2.38 PROCESS_SCANNED_DATA

```
#define PROCESS_SCANNED_DATA(  
    DTYPE_UNSIGNED,  
    DTYPE_SIGNED )
```

Value:

```
\  
static gfloat \  
gst_tensor_src_iio_process_scanned_data_from_
```

Macro to generate data processing functions for various types.

Definition at line 106 of file gstreamer_srcio.c.

9.64.2.39 SAMPLING_FREQUENCY

```
#define SAMPLING_FREQUENCY "sampling_frequency"
```

Definition at line 245 of file gstreamer_srcio.c.

9.64.2.40 SCALE_SUFFIX

```
#define SCALE_SUFFIX "_scale"
```

Definition at line 237 of file gstreamer_srcio.c.

9.64.2.41 TIMESTAMP

```
#define TIMESTAMP "timestamp"
```

Definition at line 226 of file gstreamer_srcio.c.

9.64.2.42 TRIGGER

```
#define TRIGGER "trigger"
```

Definition at line 223 of file gsttensor_srcio.c.

9.64.2.43 TRIGGER_PREFIX

```
#define TRIGGER_PREFIX IIO TRIGGER
```

Definition at line 228 of file gsttensor_srcio.c.

9.64.2.44 TYPE_SUFFIX

```
#define TYPE_SUFFIX "_type"
```

Definition at line 236 of file gsttensor_srcio.c.

9.64.3 Enumeration Type Documentation

9.64.3.1 anonymous enum

anonymous enum

tensor_src_iio properties.

Enumerator

PROP_0	
PROP_MODE	
PROP_SILENT	
PROP_BASE_DIRECTORY	
PROP_DEV_DIRECTORY	
PROP_DEVICE	
PROP_DEVICE_NUM	
PROP_TRIGGER	
PROP_TRIGGER_NUM	
PROP_CHANNELS	
PROP_BUFFER_CAPACITY	
PROP_FREQUENCY	
PROP_MERGE_CHANNELS	
PROP_POLL_TIMEOUT	

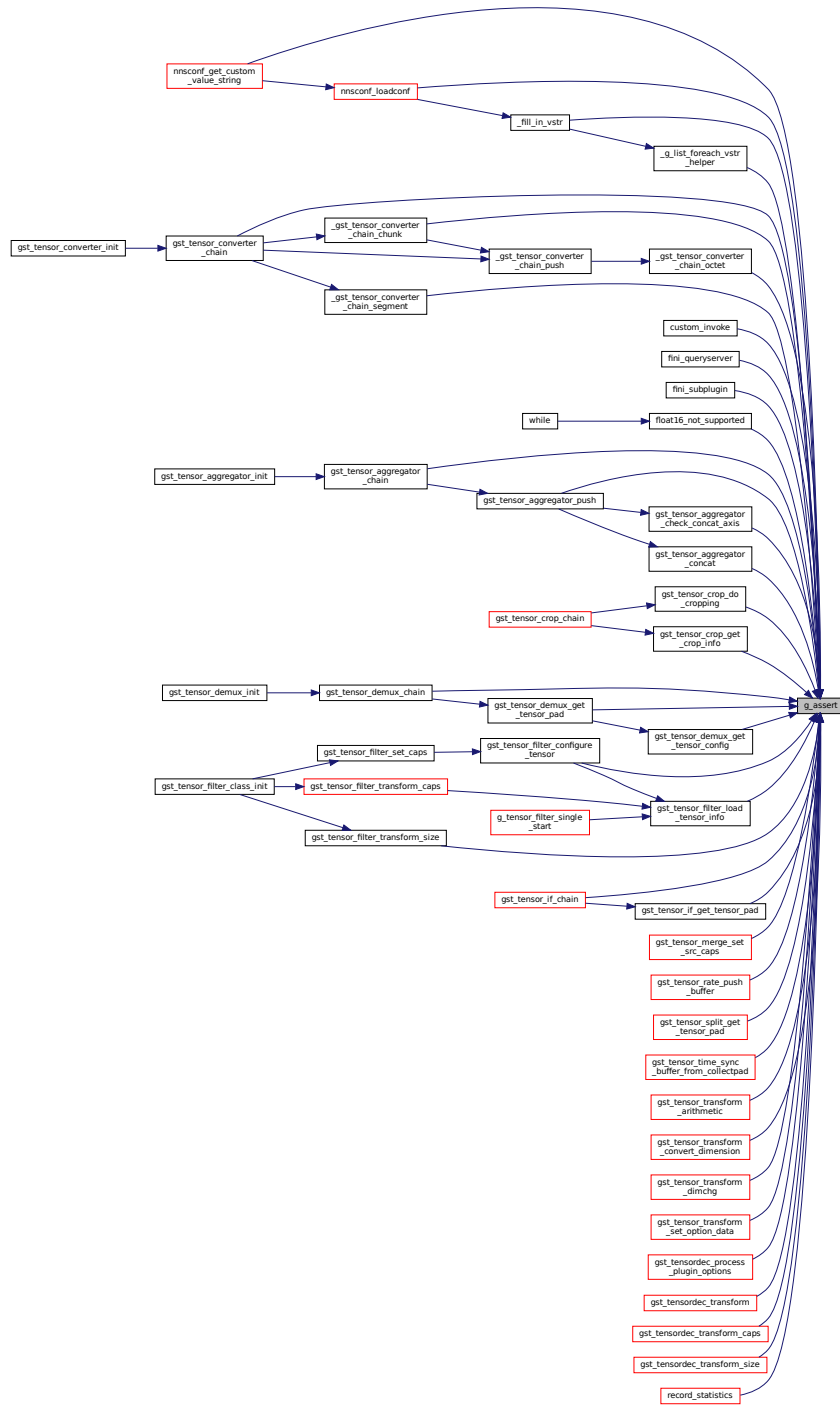
Definition at line 132 of file gsttensor_srcio.c.

9.64.4 Function Documentation

9.64.4.1 g_assert()

```
g_assert (
    sizeof(DTYPE_UNSIGNED) == sizeof(DTYPE_SIGNED) )
```

Here is the caller graph for this function:



9.64.4.2 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorSrcIIO ,
```



```
gst_tensor_src_iio ,
GST_TYPE_BASE_SRC )
```

9.64.4.3 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_src_iio_debug )
```

9.64.4.4 gst_tensor_channel_list_filter_enabled()

```
static void gst_tensor_channel_list_filter_enabled (
    gpointer data,
    gpointer user_data ) [static]
```

compare channels for filtering if enabled

Parameters

in	<i>data</i>	Pointer of the data of the element
	<i>[in/out]</i>	<i>user_data</i> Pointer to the address of the list to be filtered

Definition at line 836 of file gstreamer_srcii.c.

9.64.4.5 gst_tensor_channel_list_sort_cmp()

```
static gint gst_tensor_channel_list_sort_cmp (
    gconstpointer a,
    gconstpointer b ) [static]
```

compare channels for sort based on their indices

Parameters

in	<i>a</i>	First param to be compared
in	<i>b</i>	Second param to be compared

Returns

negative if *ab*

Definition at line 822 of file gstreamer_srcii.c.

9.64.4.6 `gst_tensor_get_size_from_channels()`

```
static guint gst_tensor_get_size_from_channels (
    GList * channels ) [static]
```

get the size of the combined data from channels

Parameters

in	<i>channels</i>	List of all the channels
----	-----------------	--------------------------

Returns

Size of one scan of data combined from all the channels

Also evaluates the location of each channel in the buffer

Definition at line 1499 of file `gsttensor_srciiio.c`.

9.64.4.7 `gst_tensor_set_all_channels()`

```
static gboolean gst_tensor_set_all_channels (
    GstTensorSrcIIO * self,
    const gint contents ) [static]
```

set value to all the channels

Parameters

in	<i>self</i>	Tensor src IIO object
in	<i>contents</i>	Data to be written to the file

Returns

True if write was successful, false if failure on any channel

Definition at line 1469 of file `gsttensor_srciiio.c`.

9.64.4.8 `gst_tensor_src_iiio_change_state()`

```
static GstStateChangeReturn gst_tensor_src_iiio_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

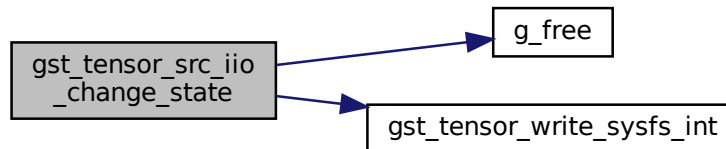
Perform state change.

enable the buffer for the data to be captured

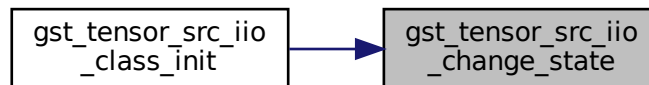
disable the buffer

Definition at line 2228 of file `gsttensor_srcii.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.64.4.9 `gst_tensor_src_iio_channel_properties_free()`

```
static void gst_tensor_src_iio_channel_properties_free (
    gpointer data ) [static]
```

delete GstTensorSrcIIODeviceProperties structure

Parameters

in	<i>data</i>	Data pointer to be freed
----	-------------	--------------------------

Definition at line 411 of file `gsttensor_srcii.c`.

9.64.4.11 `gst_tensor_src_iio_create()`

```
static GstFlowReturn gst_tensor_src_iio_create (
    GstBaseSrc * src,
    guint64 offset,
    guint size,
    GstBuffer ** buffer ) [static]
```

create a buffer with requested size and offset

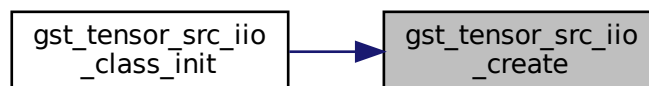
Note

offset, size ignored as the tensor src iio does not support pull mode

all the data, if unermgd should be of the same size

Definition at line 2325 of file `gsttensor_srciiio.c`.

Here is the caller graph for this function:



9.64.4.12 `gst_tensor_src_iio_create_config()`

```
static gboolean gst_tensor_src_iio_create_config (
    GstTensorSrcIIO * tensor_src_iio ) [static]
```

create the structure for the caps to update the src pad caps

Parameters

<i>[in/out]</i>	structure Caps structure which will filled
-----------------	--

Returns

True if structure is created and filled, False for any error

create a bigger array, insert info in it and then merge tensors with same type+size

compile tensor info data

merge info about the tensors with same type

verify the merging of the array

tensors config data

buffer_capacity number of data samples are captured at once, packed together and sent downstream

Definition at line 1527 of file gsttensor_srciiio.c.

9.64.4.13 `gst_tensor_src_iio_device_properties_init()`

```
static void gst_tensor_src_iio_device_properties_init (
    GstTensorSrcIIODeviceProperties * prop ) [static]
```

initialize GstTensorSrcIIODeviceProperties structure

Parameters

in	<i>data</i>	Device properties pointer to be initialized
----	-------------	---

Definition at line 427 of file gsttensor_srciiio.c.

Here is the caller graph for this function:



9.64.4.14 `gst_tensor_src_iio_event()`

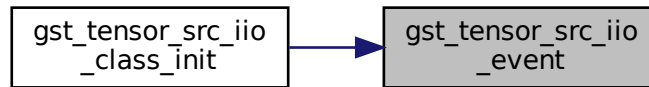
```
static gboolean gst_tensor_src_iio_event (
    GstBaseSrc * src,
    GstEvent * event ) [static]
```

handle events

No events to be handled yet

Definition at line 2132 of file gsttensor_srciiio.c.

Here is the caller graph for this function:



9.64.4.15 `gst_tensor_src_iio_fill()`

```
static GstFlowReturn gst_tensor_src_iio_fill (  
    GstBaseSrc * src,  
    guint64 offset,  
    guint size,  
    GstBuffer * buffer ) [static]
```

fill the buffer with data

Note

ignore offset,size as there is pull mode

buffer timestamp is already handled by gstreamer with gst clock

Only supporting tensors made of 1 tensor for now

get writable buffer

memory to data from file

wait for the data to arrive

sleep for a device tick

using read for non-blocking access

parse the read data

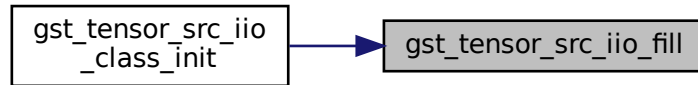
current assumption is that the all data is float and merged to form a 1 dimension data. 2nd dimension comes from buffer capacity.

for other/tensor, only 1 map exist as there is only 1 mem

for other/tensors, multiple maps exist as there are multiple mem

Definition at line 2462 of file `gsttensor_srcii.c`.

Here is the caller graph for this function:



9.64.4.16 `gst_tensor_src_iio_finalize()`

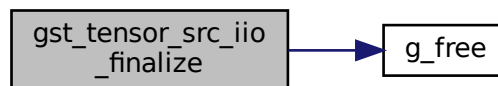
```

static void gst_tensor_src_iio_finalize (
    GObject * object ) [static]
  
```

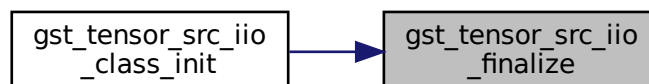
finalize the instance

Definition at line 1358 of file `gsttensor_srcio.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.64.4.17 `gst_tensor_src_iio_fixate()`

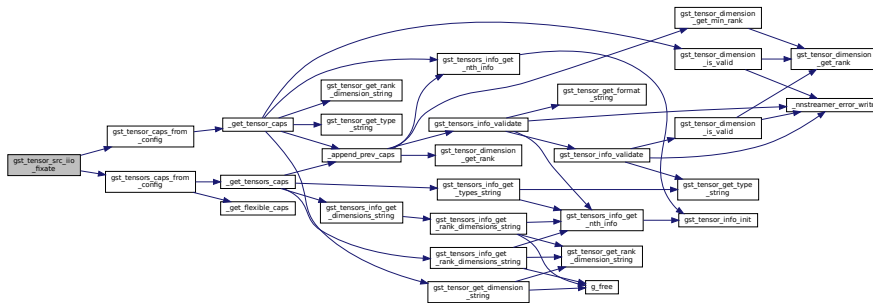
```
static GstCaps * gst_tensor_src_iio_fixate (
    GstBaseSrc * src,
    GstCaps * caps ) [static]
```

fixate the caps when needed during negotiation

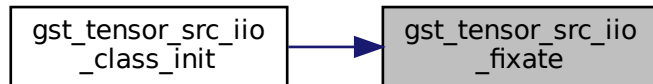
Caps are fixated based on the device source in `_start()`.

Definition at line 2186 of file `gsttensor_srcio.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.64.4.18 `gst_tensor_src_iio_get_all_channel_info()`

```
static gint gst_tensor_src_iio_get_all_channel_info (
    GstTensorSrcIIO * self,
    const gchar * dir_name ) [static]
```

get info about all the channels in the device

Parameters

	<i>[in/out]</i>	self Tensor src IIO object
in	<i>dir_name</i>	Directory name with all the scan elements for device

Returns

>=0 number of enabled channels -1 if any error when scanning channels

check for enable

not enabling and handling buffer timestamps for now

set the name and base_dir

find and set the current state

find and set the index

find and set the type information

if specific type info unavailable, use generic type info

find and setup offset info

find and setup scale info

sort the list with the order of the indices

Definition at line 860 of file gsttensor_srcii.c.

9.64.4.19 gst_tensor_src_ii_get_available_frequency()

```
static gint64 gst_tensor_src_ii_get_available_frequency (
    const gchar * base_dir,
    const guint64 frequency ) [static]
```

return sampling frequency given the frequency input from user

Parameters

in	<i>base_dir</i>	Device base directory (containing sampling freq file)
in	<i>frequency</i>	Frequency specified by user (else 0)

Returns

>0 if OK, represents sampling frequency to be set 0 if sampling frequency file does not exist, dont change anything -1 if any error occurs

get frequency list supported by the device

if the frequency is set 0, set the first available frequency else verify the frequency received from user is supported by the device

Definition at line 1036 of file gsttensor_srcii.c.

9.64.4.20 `gst_tensor_src_iio_get_caps()`

```
static GstCaps * gst_tensor_src_iio_get_caps (
    GstBaseSrc * src,
    GstCaps * filter ) [static]
```

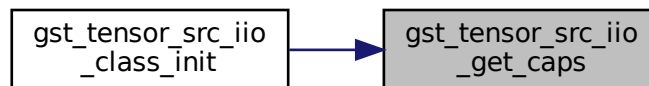
get caps of subclass

Note

`basesrc_get_caps` returns the caps from the `pad_template` however, we set the caps manually and needs to returned here

Definition at line 2160 of file `gsttensor_srcii.c`.

Here is the caller graph for this function:



9.64.4.21 `gst_tensor_src_iio_get_float_from_file()`

```
static gboolean gst_tensor_src_iio_get_float_from_file (
    const gchar * dirname,
    const gchar * name,
    const gchar * suffix,
    gfloat * value ) [static]
```

parse float value from the file

Parameters

in	<i>dirname</i>	Directory containing the file
in	<i>name</i>	Filename of the file
in	<i>suffix</i>	Suffix to be attached to the filename
	<i>[in/out]</i>	value Output value returned via value

Returns

FALSE on errors, else TRUE

Definition at line 677 of file `gsttensor_srcii.c`.

9.64.4.22 `gst_tensor_src_iio_get_generic_name()`

```
static gchar* gst_tensor_src_iio_get_generic_name (
    const gchar * channel_name ) [static]
```

get generic name for channel from the string

Parameters

in	<i>channel_name</i>	Name of the channel with its id embedded in it
----	---------------------	--

Returns

Ptr to the generic name of the channel, caller should free the returned string

Definition at line 799 of file `gsttensor_src_iio.c`.

9.64.4.23 `gst_tensor_src_iio_get_id_by_name()`

```
static gint gst_tensor_src_iio_get_id_by_name (
    const gchar * dir_name,
    const gchar * name,
    const gchar * prefix ) [static]
```

check if device/trigger with the given name exists

Parameters

in	<i>dir_name</i>	Directory containing all the devices
in	<i>name</i>	Name of the device to be found
in	<i>prefix</i>	Prefix to match with the filename of the device

Returns

>=0 if OK, represents device/trigger number -1 if returned with error

check for prefix and the next digit should be a number

Definition at line 576 of file `gsttensor_src_iio.c`.

9.64.4.24 `gst_tensor_src_iio_get_name_by_id()`

```
static gchar* gst_tensor_src_iio_get_name_by_id (
    const gchar * dir_name,
    const gint id,
    const gchar * prefix ) [static]
```

check if device/trigger with the given id exists

Parameters

in	<i>dir_name</i>	Directory containing all the devices
in	<i>id</i>	ID of the device to be found
in	<i>prefix</i>	Prefix to match with the filename of the device

Returns

name on success (owned by caller), else NULL

Definition at line 640 of file `gsttensor_srciiio.c`.

9.64.4.25 `gst_tensor_src_iio_get_property()`

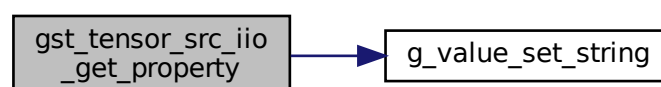
```
static void gst_tensor_src_iio_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

get `tensor_src_iio` properties

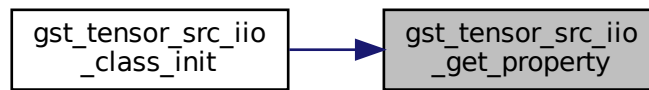
interface of frequency is kept long for outside but `uint64` inside

Definition at line 1264 of file `gsttensor_srciiio.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.64.4.26 `gst_tensor_src_iio_get_times()`

```

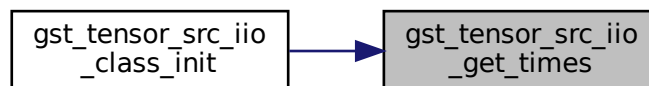
static void gst_tensor_src_iio_get_times (
    GstBaseSrc * basesrc,
    GstBuffer * buffer,
    GstClockTime * start,
    GstClockTime * end ) [static]
  
```

returns the time for the buffers

can't sync using DTS, use PTS

Definition at line 2298 of file `gsttensor_srciiio.c`.

Here is the caller graph for this function:



9.64.4.27 `gst_tensor_src_iio_init()`

```

static void gst_tensor_src_iio_init (
    GstTensorSrcIIO * self ) [static]
  
```

initialize `tensor_src_iio` element.

init properties

format of the source since IIO device as a source is live and operates at a fixed frequency, GST_FORMAT_TIME is used

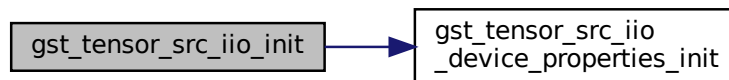
set the source to be live

set the timestamps on each buffer

set async is necessary to make state change async sync state changes does not need calling `_start_complete()` from `_start()`

Definition at line 439 of file `gsttensor_srciiio.c`.

Here is the call graph for this function:



9.64.4.28 `gst_tensor_src_iio_is_seekable()`

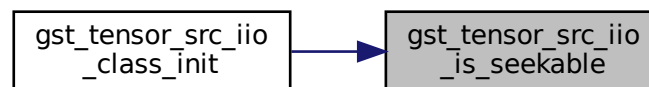
```
static gboolean gst_tensor_src_iio_is_seekable (
    GstBaseSrc * src ) [static]
```

check if source supports seeking

iio sensors are live source without any support for seeking

Definition at line 2287 of file `gsttensor_srciiio.c`.

Here is the caller graph for this function:



9.64.4.29 `gst_tensor_src_iio_process_scanned_data()`

```
static gboolean gst_tensor_src_iio_process_scanned_data (
    GstTensorSrcIIOChannelProperties * prop,
    gchar * data,
    gfloat * buffer_map ) [static]
```

process the scanned data from IIO device

Parameters

in	<i>prop</i>	Properties of one of the enabled channels
in	<i>data</i>	Data read from the IIO device
	<i>[in/out]</i>	buffer_map Gst buffer map to write data to

Returns

FALSE if fail, else TRUE

assumes each data starting point is byte aligned right shift the extra storage bits

right shift the extra storage bits for big endian

mask out the extra storage bits for little endian

follow through

right shift the extra storage bits for big endian

mask out the extra storage bits for little endian

follow through

follow through

follow through

right shift the extra storage bits for big endian

mask out the extra storage bits for little endian

Definition at line 2376 of file gsttensor_srciiio.c.

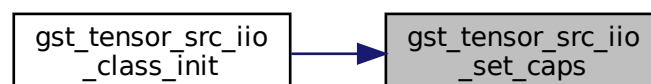
9.64.4.30 `gst_tensor_src_iio_set_caps()`

```
static gboolean gst_tensor_src_iio_set_caps (
    GstBaseSrc * src,
    GstCaps * caps ) [static]
```

set new caps

Definition at line 2142 of file gsttensor_srciiio.c.

Here is the caller graph for this function:



9.64.4.31 `gst_tensor_src_iio_set_channel_type()`

```
static gboolean gst_tensor_src_iio_set_channel_type (
    GstTensorSrcIIOChannelProperties * prop,
    const gchar * contents ) [static]
```

get type info about the channel from the string

Parameters

	<i>[in/out]</i>	prop Channel properties where type info will be set
in	<i>contents</i>	Contains type unparsed information to be set

Returns

True if info was successfully set, false if info is not be parsed correctly @detail The format for the contents is expected to be of format

check endian

verify static parts of the contents

check sign

used bits

verify static parts of the contents

storage bits

verify static parts of the contents

Definition at line 718 of file `gsttensor_srcio.c`.

9.64.4.32 `gst_tensor_src_iio_set_property()`

```
static void gst_tensor_src_iio_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

set `tensor_src_iio` properties

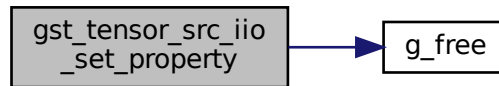
GObject method implementation No support for setting properties in PAUSED/PLAYING state as it needs to reset the device. To change the properties, user should stop the pipeline and set element state to READY/NULL and then change the properties

using direct as we only need to store keys and keys form a unique set

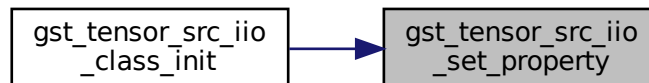
this means val is duplicated. just skip it, then.

Definition at line 1098 of file gsttensor_srcii.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.64.4.33 `gst_tensor_src_ii_setup_device_buffer()`

```
static gboolean gst_tensor_src_ii_setup_device_buffer (
    GstTensorSrcIIo * self ) [static]
```

setup device using name/id

Parameters

<i>[in/out]</i>	self Tensor src ii object
-----------------	---------------------------

Returns

TRUE on success, FALSE on failure

once all these are set, set the buffer related things

open the buffer to read and ready the file descriptor

Definition at line 1913 of file gsttensor_srcii.c.

9.64.4.34 `gst_tensor_src_iiio_setup_device_properties()`

```
static gboolean gst_tensor_src_iiio_setup_device_properties (  
    GstTensorSrcIIIO * self ) [static]
```

setup device using name/id

Parameters

<i>[in/out]</i>	self Tensor src iiio object
-----------------	-----------------------------

Returns

TRUE on success, FALSE on failure

Find the device

Definition at line 1623 of file gsttensor_srciiio.c.

9.64.4.35 `gst_tensor_src_iiio_setup_sampling_frequency()`

```
static gboolean gst_tensor_src_iiio_setup_sampling_frequency (  
    GstTensorSrcIIIO * self ) [static]
```

setup device sampling frequency

Parameters

<i>[in/out]</i>	self Tensor src iiio object
-----------------	-----------------------------

Returns

TRUE on success, FALSE on failure

check if sampling frequency file exists

reset the sampling frequency set by the user if any, as it cant be set

store the default frequency

verify the frequency given by the user if any from the list of available sampling frequencies

if sampling frequency file does not exist, no error

if sampling frequency file does not exist, sampling frequency is first value from the list of available sampling frequencies. So, we can ignore setting it

interface of frequency is kept long for outside but uint64 inside

Definition at line 1735 of file gsttensor_srciiio.c.

9.64.4.36 `gst_tensor_src_iio_setup_scan_channels()`

```
static gboolean gst_tensor_src_iio_setup_scan_channels (  
    GstTensorSrcIIO * self ) [static]
```

setup scan channels for the device

Parameters

<i>[in/out]</i>	self Tensor src iio object
-----------------	----------------------------

Returns

TRUE on success, FALSE on failure

get all the channels that exist and then set enable on them

if enabling all channels failed, disable all channels

enable the custom channels and disable the rest

filter out disabled channels

set fixed caps for the src pad

create tensor_config

Definition at line 1822 of file gsttensor_src_iio.c.

9.64.4.37 `gst_tensor_src_iio_setup_trigger_properties()`

```
static gboolean gst_tensor_src_iio_setup_trigger_properties (  
    GstTensorSrcIIO * self ) [static]
```

setup trigger using name/id

Parameters

<i>[in/out]</i>	self Tensor src iio object
-----------------	----------------------------

Returns

TRUE on success, FALSE on failure

register the trigger

verify if trigger is supported by our device

find if the provided trigger exists

get the default trigger, if any

set the trigger

Definition at line 1659 of file gsttensor_srciiio.c.

9.64.4.38 `gst_tensor_src_iio_start()`

```
static gboolean gst_tensor_src_iio_start (  
    GstBaseSrc * src ) [static]
```

start function, called when state changed null to ready. load the device and init the device resources

GstBaseSrc method implementation load and init resources

no support one shot mode for now

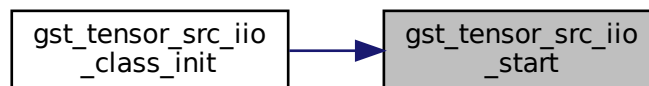
bytes every buffer will be fixed

complete the start of the base src

complete the start of the base src

Definition at line 1977 of file gsttensor_srciiio.c.

Here is the caller graph for this function:



9.64.4.39 `gst_tensor_src_iio_stop()`

```
static gboolean gst_tensor_src_iio_stop (
    GstBaseSrc * src ) [static]
```

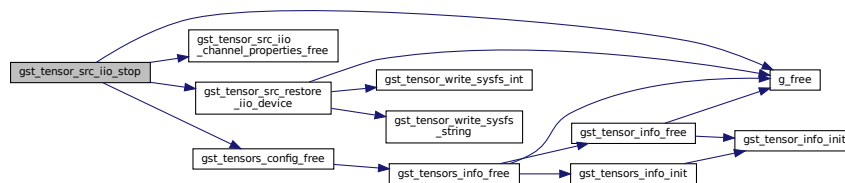
stop function, called when state changed ready to null.

free resources related to the device

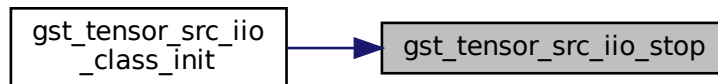
restore the iio device

Definition at line 2097 of file `gsttensor_src_iio.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.64.4.40 `gst_tensor_src_merge_tensor_by_type()`

```
static gint gst_tensor_src_merge_tensor_by_type (
    GstTensorInfo * info,
    guint size,
    guint dir ) [static]
```

merge multiple other/tensor

Note

- they should have matching type and shape to form 1 other/tensors
- extra dimension should be available for other/tensors
- order of merge is stable
- merging into 1 tensor only supported using innermost dimension.

Parameters

	<i>[in/out]</i>	info Tensor info to be merged
<i>in</i>	<i>size</i>	Info array size
<i>in</i>	<i>dir</i>	Innermost/outermost/innermost-outer (0/1/2) available dimension

Returns

>=0 number of valid entries in the info after merge -1 failed due to missing extra dimension/mismatch shape/type

base error control check

verify extra dimension (innermost to outermost)

verify that all the types and shapes match

return original if cant be merged and size within limits

If there are multiple available dimensions to merge along, we use *dir* to choose which the dimension to merge. If there is just 1 dimension, *dir* variable has no effect

No outer dimension available to merge

Now merge into 1 tensor using the selected dimension

Definition at line 491 of file `gsttensor_srciiio.c`.

9.64.4.41 `gst_tensor_src_restore_iio_device()`

```
static void gst_tensor_src_restore_iio_device (
    GstTensorSrcIIO * self ) [static]
```

restore the iio device to its original device.

reset enabled channels

reset `sampling_frequency`

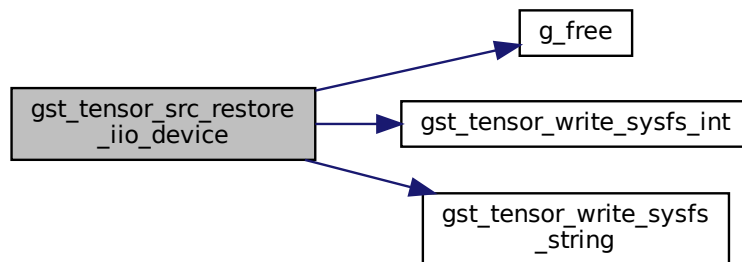
converting to long as setting interface to device

reset `buffer_capacity`

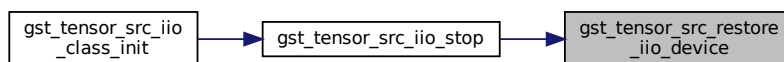
reset current trigger

Definition at line 2049 of file `gsttensor_srciiio.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.64.4.42 `gst_tensor_write_sysfs_int()`

```

static gboolean gst_tensor_write_sysfs_int (
    GstTensorSrcIIO * self,
    const gchar * file,
    const gchar * base_dir,
    const gint contents ) [static]
  
```

write the int in to the file

Parameters

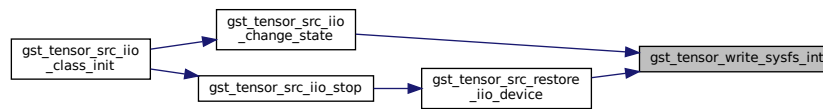
in	<i>self</i>	Tensor src IIO object
in	<i>file</i>	Destination file for the data
in	<i>base_dir</i>	Directory containing the file
in	<i>contents</i>	Data to be written to the file

Returns

True if write was successful, false on failure

Definition at line 1449 of file `gsttensor_srciiio.c`.

Here is the caller graph for this function:



9.64.4.43 gst_tensor_write_sysfs_string()

```

static gboolean gst_tensor_write_sysfs_string (
    GstTensorSrcIIO * self,
    const gchar * file,
    const gchar * base_dir,
    const gchar * contents ) [static]
  
```

write the string in to the file

Parameters

in	<i>self</i>	Tensor src IIO object
in	<i>file</i>	Destination file for the data
in	<i>base_dir</i>	Directory containing the file
in	<i>contents</i>	Data to be written to the file

Returns

True if write was successful, false on failure

Definition at line 1384 of file gsttensor_srciiio.c.

Here is the caller graph for this function:



9.64.4.44 if()

```

value_unsigned& if (
    prop-> is_signed )
  
```

Definition at line 117 of file gsttensor_srciiio.c.

9.64.4.45 PROCESS_SCANNED_DATA() [1/4]

```
PROCESS_SCANNED_DATA (
    guint16 ,
    gint16 )
```

9.64.4.46 PROCESS_SCANNED_DATA() [2/4]

```
PROCESS_SCANNED_DATA (
    guint32 ,
    gint32 )
```

9.64.4.47 PROCESS_SCANNED_DATA() [3/4]

```
PROCESS_SCANNED_DATA (
    guint64 ,
    gint64 )
```

9.64.4.48 PROCESS_SCANNED_DATA() [4/4]

```
PROCESS_SCANNED_DATA (
    guint8 ,
    gint8 )
```

Define data processing functions for various types

9.64.5 Variable Documentation

9.64.5.1 else

else

Initial value:

```
{
    value_float = ((gfloat) value_unsigned + prop->offset) * prop->scale
```

Definition at line 123 of file gsttensor_srcio.c.

9.64.5.2 prop

`GstTensorSrcIIIOChannelProperties*` prop

DTYPE_UNSIGNED (.

Definition at line 110 of file `gsttensor_srciiio.c`.

9.64.5.3 value_float

return `value_float`

Definition at line 126 of file `gsttensor_srciiio.c`.

9.64.5.4 value_unsigned

`value_unsigned`

Initial value:

```
{
    gfloat value_float
```

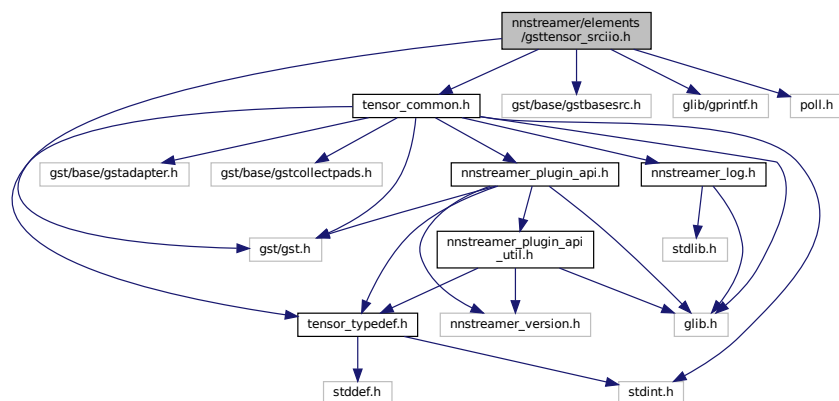
Definition at line 110 of file `gsttensor_srciiio.c`.

9.65 nntstreamer/elements/gsttensor_srciiio.h File Reference

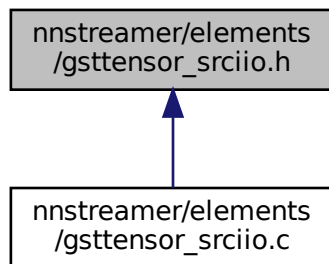
GStreamer plugin to support linux IIO as tensor(s)

```
#include <gst/gst.h>
#include <gst/base/gstbasesrc.h>
#include <glib/gprintf.h>
#include <tensor_common.h>
#include <poll.h>
```

Include dependency graph for `gsttensor_srciiio.h`:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorSrcIIODeviceProperties](#)
GstTensorSrcIIO devices's properties (internal data structure)
- struct [_GstTensorSrcIIOChannelProperties](#)
GstTensorSrcIIO channel's properties (internal data structure)
- struct [_GstTensorSrcIIO](#)
GstTensorSrcIIO data structure.
- struct [_GstTensorSrcIIOClass](#)
GstTensorSrcIIOClass data structure.

Macros

- #define [GST_TYPE_TENSOR_SRC_IIO](#) ([gst_tensor_src_iio_get_type\(\)](#))
- #define [GST_TENSOR_SRC_IIO\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_SRC_IIO, GstTensorSrcIIO\)](#))
- #define [GST_TENSOR_SRC_IIO_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_SRC_IIO, _GstTensorSrcIIOClass\)](#))
- #define [GST_IS_TENSOR_SRC_IIO\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_SRC_IIO\)](#))
- #define [GST_IS_TENSOR_SRC_IIO_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_SRC_IIO\)](#))
- #define [GST_TENSOR_SRC_IIO_CAST\(obj\)](#) ([\(\(GstTensorSrcIIO *\) \(obj\)\)](#))

Typedefs

- typedef struct [_GstTensorSrcIIO](#) [GstTensorSrcIIO](#)
- typedef struct [_GstTensorSrcIIOClass](#) [GstTensorSrcIIOClass](#)
- typedef struct [_GstTensorSrcIIODeviceProperties](#) [GstTensorSrcIIODeviceProperties](#)
GstTensorSrcIIO devices's properties (internal data structure)
- typedef struct [_GstTensorSrcIIOChannelProperties](#) [GstTensorSrcIIOChannelProperties](#)
GstTensorSrcIIO channel's properties (internal data structure)

Enumerations

- enum [channels_enabled_options](#) { [CHANNELS_ENABLED_ALL](#), [CHANNELS_ENABLED_AUTO](#), [CHANNELS_ENABLED_CUSTOM](#) }
iiio device channel enabled mode

Functions

- GType `gst_tensor_src_iio_get_type` (void)
Function to get type of tensor_src_iio.

9.65.1 Detailed Description

GStreamer plugin to support linux IIO as tensor(s)

GStreamer Tensor_Src_IIO Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2019 Parichay Kapoor pk.kapoor@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

26 Feb 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

Bug No known bugs except for NYI items

9.65.2 Macro Definition Documentation

9.65.2.1 GST_IS_TENSOR_SRC_IIO

```
#define GST_IS_TENSOR_SRC_IIO(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_SRC_IIO))
```

Definition at line 43 of file `gsttensor_srciiio.h`.

9.65.2.2 GST_IS_TENSOR_SRC_IIO_CLASS

```
#define GST_IS_TENSOR_SRC_IIO_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_SRC_IIO))
```

Definition at line 45 of file gsttensor_srciiio.h.

9.65.2.3 GST_TENSOR_SRC_IIO

```
#define GST_TENSOR_SRC_IIO(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_SRC_IIO, GstTensorSrcIIO))
```

Definition at line 39 of file gsttensor_srciiio.h.

9.65.2.4 GST_TENSOR_SRC_IIO_CAST

```
#define GST_TENSOR_SRC_IIO_CAST(  
    obj ) ((GstTensorSrcIIO *) (obj))
```

Definition at line 47 of file gsttensor_srciiio.h.

9.65.2.5 GST_TENSOR_SRC_IIO_CLASS

```
#define GST_TENSOR_SRC_IIO_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_SRC_IIO, GstTensorSrcIIOClass))
```

Definition at line 41 of file gsttensor_srciiio.h.

9.65.2.6 GST_TYPE_TENSOR_SRC_IIO

```
#define GST_TYPE_TENSOR_SRC_IIO (gst_tensor_src_iiio_get_type())
```

Definition at line 37 of file gsttensor_srciiio.h.

9.65.3 Typedef Documentation

9.65.3.1 GstTensorSrcIIO

```
typedef struct _GstTensorSrcIIO GstTensorSrcIIO
```

Definition at line 48 of file gstreamer_srciiio.h.

9.65.3.2 GstTensorSrcIIOChannelProperties

```
typedef struct _GstTensorSrcIIOChannelProperties GstTensorSrcIIOChannelProperties
```

GstTensorSrcIIO channel's properties (internal data structure)

9.65.3.3 GstTensorSrcIIOClass

```
typedef struct _GstTensorSrcIIOClass GstTensorSrcIIOClass
```

Definition at line 49 of file gstreamer_srciiio.h.

9.65.3.4 GstTensorSrcIIODeviceProperties

```
typedef struct _GstTensorSrcIIODeviceProperties GstTensorSrcIIODeviceProperties
```

GstTensorSrcIIO devices's properties (internal data structure)

This data structure is used for both device/triggers, as triggers are also iio devices

9.65.4 Enumeration Type Documentation

9.65.4.1 channels_enabled_options

```
enum channels_enabled_options
```

iio device channel enabled mode

Enumerator

CHANNELS_ENABLED_ALL	
CHANNELS_ENABLED_AUTO	
CHANNELS_ENABLED_CUSTOM	

Definition at line 54 of file gsttensor_srcio.h.

9.65.5 Function Documentation

9.65.5.1 gst_tensor_src_iio_get_type()

```
GType gst_tensor_src_iio_get_type (
    void )
```

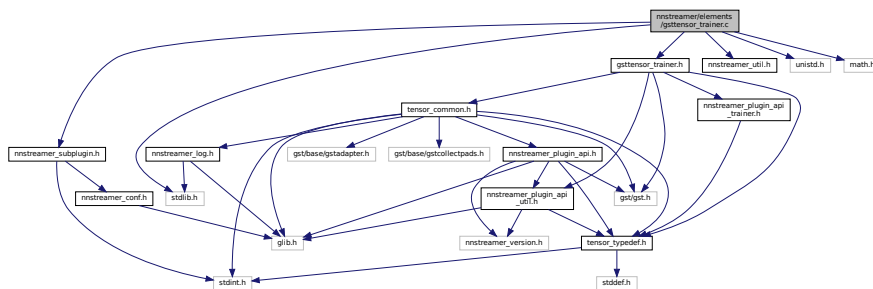
Function to get type of tensor_src_iio.

9.66 nnstreamer/elements/gsttensor_trainer.c File Reference

GStreamer plugin to train tensor data using NN Frameworks.

```
#include <stdlib.h>
#include <nnstreamer_subplugin.h>
#include <nnstreamer_util.h>
#include "gsttensor_trainer.h"
#include <unistd.h>
#include <math.h>
```

Include dependency graph for gsttensor_trainer.c:



Macros

- #define `SINK_CAPS_STRING` `GST_TENSORS_CAP_MAKE ("{ static, flexible }")`
Default caps string for sink.
- #define `SRC_CAPS_STRING` `GST_TENSORS_CAP_MAKE ("{ static}")`
Default caps string for src.
- #define `GST_CAT_DEFAULT` `gst_tensor_trainer_debug`
- #define `gst_tensor_trainer_parent_class` `parent_class`
- #define `MODEL_STATS_SIZE` `4`
- #define `DEFAULT_PROP_INPUT_LIST` `1`
Default framework property value.
- #define `DEFAULT_PROP_LABEL_LIST` `1`
- #define `DEFAULT_PROP_TRAIN_SAMPLES` `0`
- #define `DEFAULT_PROP_VALID_SAMPLES` `0`
- #define `DEFAULT_PROP_EPOCHS` `1`
- #define `DEFAULT_STR_PROP_VALUE` `""`
Default string property value.

Enumerations

- enum { [TRAINING_LOSS](#), [TRAINING_ACCURACY](#), [VALIDATION_LOSS](#), [VALIDATION_ACCURACY](#) }
Statistical from the model being trained An enum value indicates the value stored at the index of the output tensor.
- enum { [PROP_0](#), [PROP_FRAMEWORK](#), [PROP_MODEL_CONFIG](#), [PROP_MODEL_SAVE_PATH](#), [PROP_MODEL_LOAD_PATH](#), [PROP_NUM_INPUTS](#), [PROP_NUM_LABELS](#), [PROP_NUM_TRAINING_SAMPLES](#), [PROP_NUM_VALIDATION_SAMPLES](#), [PROP_EPOCHS](#) }
tensor_trainer properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_trainer_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorTrainer](#), [gst_tensor_trainer](#), [GST_TYPE_ELEMENT](#))
- static void [gst_tensor_trainer_set_property](#) ([GObject](#) *object, guint prop_id, const [GValue](#) *value, [GParamSpec](#) *pspec)
Setter for tensor_trainsink properties.
- static void [gst_tensor_trainer_get_property](#) ([GObject](#) *object, guint prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
Getter tensor_trainsink properties.
- static void [gst_tensor_trainer_finalize](#) ([GObject](#) *object)
Function to finalize instance.
- static gboolean [gst_tensor_trainer_sink_event](#) ([GstPad](#) *sinkpad, [GstObject](#) *parent, [GstEvent](#) *event)
Event handler for sink pad of tensor_trainer.
- static gboolean [gst_tensor_trainer_sink_query](#) ([GstPad](#) *sinkpad, [GstObject](#) *parent, [GstQuery](#) *query)
This function handles sink pad query.
- static gboolean [gst_tensor_trainer_src_query](#) ([GstPad](#) *srcpad, [GstObject](#) *parent, [GstQuery](#) *query)
This function handles src pad query.
- static [GstFlowReturn](#) [gst_tensor_trainer_chain](#) ([GstPad](#) *sinkpad, [GstObject](#) *parent, [GstBuffer](#) *inbuf)
Chain function, this function does the actual processing.
- static [GstCaps](#) * [gst_tensor_trainer_query_caps](#) ([GstTensorTrainer](#) *trainer, [GstPad](#) *pad, [GstCaps](#) *filter)
Get pad caps for caps negotiation.
- static [GstStateChangeReturn](#) [gst_tensor_trainer_change_state](#) ([GstElement](#) *element, [GstStateChange](#) transition)
Change state of tensor_trainsink.
- static void [gst_tensor_trainer_set_prop_framework](#) ([GstTensorTrainer](#) *trainer, const [GValue](#) *value)
Handle "PROP_FRAMEWORK" for set-property.
- static void [gst_tensor_trainer_set_prop_model_config_file_path](#) ([GstTensorTrainer](#) *trainer, const [GValue](#) *value)
Handle "PROP_MODEL_CONFIG" for set-property.
- static void [gst_tensor_trainer_set_model_save_path](#) ([GstTensorTrainer](#) *trainer, const [GValue](#) *value)
Handle "PROP_MODEL_SAVE_PATH" for set-property.
- static void [gst_tensor_trainer_set_model_load_path](#) ([GstTensorTrainer](#) *trainer, const [GValue](#) *value)
Handle "PROP_MODEL_LOAD_PATH" for set-property.
- static gboolean [gst_tensor_trainer_find_framework](#) ([GstTensorTrainer](#) *trainer, const char *name)
Find Trainer sub-plugin with the name.
- static gboolean [gst_tensor_trainer_create_framework](#) ([GstTensorTrainer](#) *trainer)
Create NN framework.
- static gsize [gst_tensor_trainer_get_tensor_size](#) ([GstTensorTrainer](#) *trainer, guint index, gboolean is_input)
Calculate tensor buffer size.
- static gboolean [gst_tensor_trainer_create_model](#) ([GstTensorTrainer](#) *trainer)

- Create model.*

 - static void [gst_tensor_trainer_create_event_notifier](#) ([GstTensorTrainer](#) *trainer)

Create a event notifier.
- static void [gst_tensor_trainer_start_model_training](#) ([GstTensorTrainer](#) *trainer)

Start model training.
- static void [gst_tensor_trainer_stop_model_training](#) ([GstTensorTrainer](#) *trainer)

Stop model training.
- static void [gst_tensor_trainer_set_output_meta](#) ([GstTensorTrainer](#) *trainer)

initialize the output tensor dimension
- static void [gst_tensor_trainer_class_init](#) ([GstTensorTrainerClass](#) *klass)

initialize the tensor_trainer's class
- static void [gst_tensor_trainer_init](#) ([GstTensorTrainer](#) *trainer)

Initialize tensor_trainer.
- static gboolean [gst_tensor_trainer_check_invalid_param](#) ([GstTensorTrainer](#) *trainer)

Check invalid param.
- static gpointer [gst_tensor_trainer_dummy_data_generation_func](#) ([GstTensorTrainer](#) *trainer)

Dummy data generation thread.
- static void [gst_tensor_trainer_wait_for_epoch_completion](#) ([GstTensorTrainer](#) *trainer)

Wait for epoch eompletion.
- static gboolean [gst_tensor_trainer_epochs_is_complete](#) ([GstTensorTrainer](#) *trainer)

Check if current epochs is complete, tensor_trainer wait for one of epochs to complete before getting the results from the subplugin.
- static gboolean [gst_tensor_trainer_check_buffer_drop_conditions](#) ([GstTensorTrainer](#) *trainer)

Check buffer drop conditions. If condition is met, drop the buffer.
- static gboolean [gst_tensor_trainer_check_chain_conditions](#) ([GstTensorTrainer](#) *trainer, guint num_tensors)

Check chain conditions. If all conditions are met, proceed to next step.
- static gsize [gst_tensor_trainer_convert_meta](#) ([GstTensorTrainer](#) *trainer, [GstTensorMetaInfo](#) *meta, [GstTensorInfo](#) *info, void *data)

Convert tensor meta and get the size of tensor header.
- static gboolean [gst_tensor_trainer_push_input](#) ([GstTensorTrainer](#) *trainer, [GstBuffer](#) *inbuf, gboolean in_↔flexible)

Create input tensors from the buffer and push it into trainer fw.
- static gboolean [gst_tensor_trainer_get_model_stats](#) ([GstTensorTrainer](#) *trainer, double *model_stats)

Get the model statistics from the sub-plugin.
- static [GstBuffer](#) * [gst_tensor_trainer_create_output](#) ([GstTensorTrainer](#) *trainer)

Create output tensors.
- static void [gst_tensor_trainer_wait_for_training_completion](#) ([GstTensorTrainer](#) *trainer)

Wait for training completion.
- int [nnstreamer_trainer_probe](#) ([GstTensorTrainerFramework](#) *ttsp)

Trainer's sub-plugin should call this function to register itself.
- int [nnstreamer_trainer_exit](#) ([GstTensorTrainerFramework](#) *ttsp)

Trainer's sub-plugin may call this to unregister itself.
- void [nnstreamer_trainer_notify_event](#) ([GstTensorTrainerEventNotifier](#) *notifier, [GstTensorTrainerEventType](#) type, void *data)

Trainer's sub-plugin may call this to send event.

Variables

- static [GstStaticPadTemplate](#) [sink_template](#)

The capabilities of the sink pad.
- static [GstStaticPadTemplate](#) [src_template](#)

The capabilities of the src pad.

9.66.1 Detailed Description

GStreamer plugin to train tensor data using NN Frameworks.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

20 October 2022

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

```
//! Example launch line [[ gst-launch-1.0 datareposrc location=mnist_trainingSet.dat json=mnist.json start-  
sample-index=3 stop-sample-index=202 epochs=5 ! \ tensor_trainer framework=nntrainer model-config=mnist.ini  
model-save-path=model.bin \ num-inputs=1 num-labels=1 num-training-samples=100 num-validation-samples=100  
epochs=5 ! \ tensor_sink ]]
```

Total number of data to be received is $1000((\text{num-training-samples} + \text{num-validation-samples}) * \text{epochs})$

output tensors : dimensions=1:1:4, types=float64. values are training loss, training accuracy, validation loss and validation accuracy. -INFINITY value is stored if the value fetched from the sub-plugin is not greater than 0.

9.66.2 Macro Definition Documentation

9.66.2.1 DEFAULT_PROP_EPOCHS

```
#define DEFAULT_PROP_EPOCHS 1
```

Definition at line 88 of file gsttensor_trainer.c.

9.66.2.2 DEFAULT_PROP_INPUT_LIST

```
#define DEFAULT_PROP_INPUT_LIST 1
```

Default framework property value.

Definition at line 84 of file gsttensor_trainer.c.

9.66.2.3 DEFAULT_PROP_LABEL_LIST

```
#define DEFAULT_PROP_LABEL_LIST 1
```

Definition at line 85 of file gsttensor_trainer.c.

9.66.2.4 DEFAULT_PROP_TRAIN_SAMPLES

```
#define DEFAULT_PROP_TRAIN_SAMPLES 0
```

Definition at line 86 of file gsttensor_trainer.c.

9.66.2.5 DEFAULT_PROP_VALID_SAMPLES

```
#define DEFAULT_PROP_VALID_SAMPLES 0
```

Definition at line 87 of file gsttensor_trainer.c.

9.66.2.6 DEFAULT_STR_PROP_VALUE

```
#define DEFAULT_STR_PROP_VALUE ""
```

Default string property value.

Definition at line 92 of file gsttensor_trainer.c.

9.66.2.7 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_trainer_debug
```

Definition at line 64 of file gsttensor_trainer.c.

9.66.2.8 gst_tensor_trainer_parent_class

```
#define gst_tensor_trainer_parent_class parent_class
```

Definition at line 65 of file gsttensor_trainer.c.

9.66.2.9 MODEL_STATS_SIZE

```
#define MODEL_STATS_SIZE 4
```

Definition at line 79 of file gsttensor_trainer.c.

9.66.2.10 SINK_CAPS_STRING

```
#define SINK_CAPS_STRING GST_TENSORS_CAP_MAKE ("{ static, flexible }")
```

Default caps string for sink.

Definition at line 40 of file gsttensor_trainer.c.

9.66.2.11 SRC_CAPS_STRING

```
#define SRC_CAPS_STRING GST_TENSORS_CAP_MAKE ("{ static}")
```

Default caps string for src.

Definition at line 45 of file gsttensor_trainer.c.

9.66.3 Enumeration Type Documentation

9.66.3.1 anonymous enum

anonymous enum

Statistical from the model being trained An enum value indicates the value stored at the index of the output tensor.

Enumerator

TRAINING_LOSS	
TRAINING_ACCURACY	
VALIDATION_LOSS	
VALIDATION_ACCURACY	

Definition at line 72 of file gsttensor_trainer.c.

9.66.3.2 anonymous enum

anonymous enum

tensor_trainer properties

Enumerator

PROP_0	
PROP_FRAMEWORK	
PROP_MODEL_CONFIG	
PROP_MODEL_SAVE_PATH	
PROP_MODEL_LOAD_PATH	
PROP_NUM_INPUTS	
PROP_NUM_LABELS	
PROP_NUM_TRAINING_SAMPLES	
PROP_NUM_VALIDATION_SAMPLES	
PROP_EPOCHS	

Definition at line 97 of file gsttensor_trainer.c.

9.66.4 Function Documentation

9.66.4.1 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorTrainer ,
    gst_tensor_trainer ,
    GST_TYPE_ELEMENT )
```

9.66.4.2 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_trainer_debug )
```

9.66.4.3 `gst_tensor_trainer_chain()`

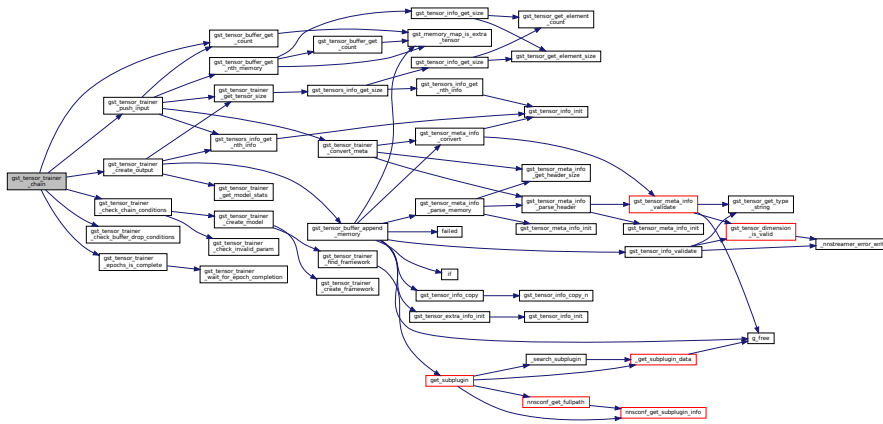
```
static GstFlowReturn gst_tensor_trainer_chain (
    GstPad * sinkpad,
    GstObject * parent,
    GstBuffer * inbuf ) [static]
```

Chain function, this function does the actual processing.

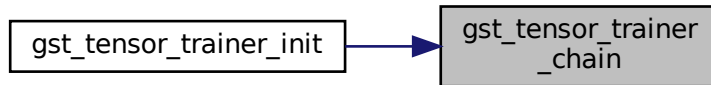
Update result if one of epochs is complete, push one outbuf is necessary to change pipeline state. Scheduling with subplugin does not work.

Definition at line 868 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

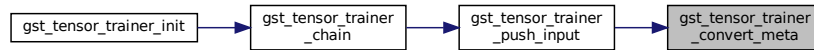


9.66.4.4 `gst_tensor_trainer_change_state()`

```
static GstStateChangeReturn gst_tensor_trainer_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

Change state of `tensor_trainsink`.

Here is the caller graph for this function:



9.66.4.10 `gst_tensor_trainer_create_event_notifier()`

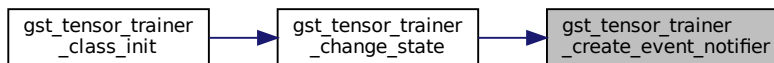
```

static void gst_tensor_trainer_create_event_notifier (
    GstTensorTrainer * trainer ) [static]
  
```

Create a event notifier.

Definition at line 1296 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



9.66.4.11 `gst_tensor_trainer_create_framework()`

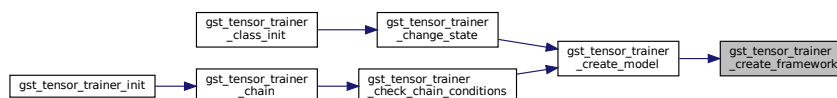
```

static gboolean gst_tensor_trainer_create_framework (
    GstTensorTrainer * trainer ) [static]
  
```

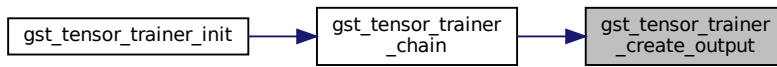
Create NN framework.

Definition at line 1221 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



Here is the caller graph for this function:



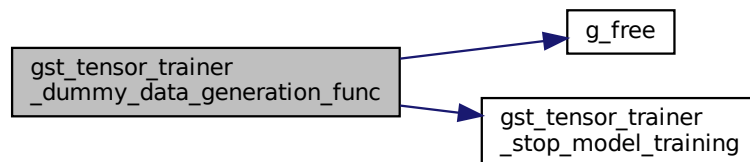
9.66.4.14 gst_tensor_trainer_dummy_data_generation_func()

```
static gpointer gst_tensor_trainer_dummy_data_generation_func (  
    GstTensorTrainer * trainer ) [static]
```

Dummy data generation thread.

Definition at line 471 of file gsttensor_trainer.c.

Here is the call graph for this function:



Here is the caller graph for this function:



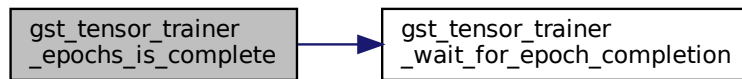
9.66.4.15 `gst_tensor_trainer_epochs_is_complete()`

```
static gboolean gst_tensor_trainer_epochs_is_complete (
    GstTensorTrainer * trainer ) [static]
```

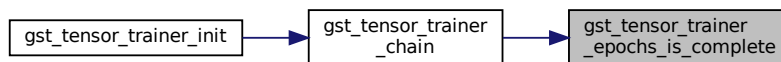
Check if current epochs is complete, tensor_trainer wait for one of epochs to complete before getting the results from the subplugin.

Definition at line 609 of file gsttensor_trainer.c.

Here is the call graph for this function:



Here is the caller graph for this function:



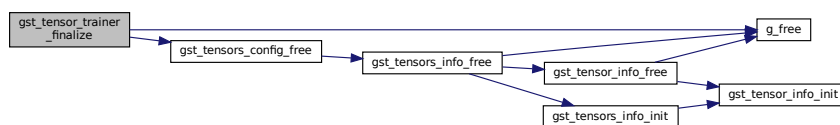
9.66.4.16 `gst_tensor_trainer_finalize()`

```
static void gst_tensor_trainer_finalize (
    GObject * object ) [static]
```

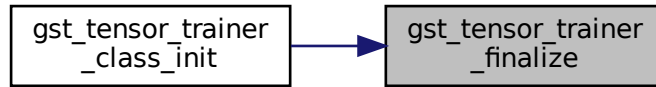
Function to finalize instance.

Definition at line 316 of file gsttensor_trainer.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.66.4.17 `gst_tensor_trainer_find_framework()`

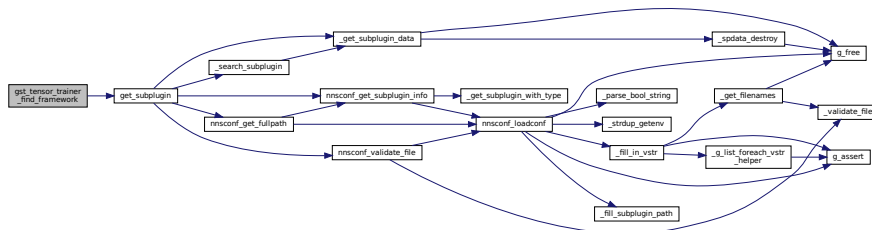
```

static gboolean gst_tensor_trainer_find_framework (
    GstTensorTrainer * trainer,
    const char * name ) [static]
  
```

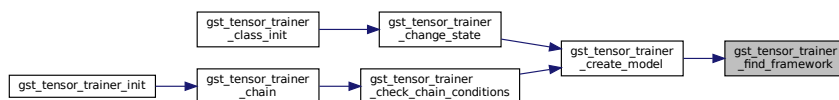
Find Trainer sub-plugin with the name.

Definition at line 1196 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



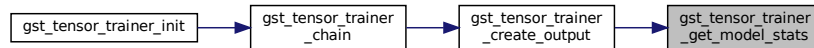
9.66.4.18 `gst_tensor_trainer_get_model_stats()`

```
static gboolean gst_tensor_trainer_get_model_stats (
    GstTensorTrainer * trainer,
    double * model_stats ) [static]
```

Get the model statistics from the sub-plugin.

Definition at line 769 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



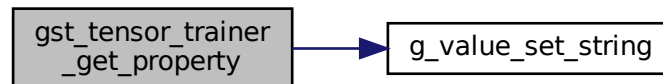
9.66.4.19 `gst_tensor_trainer_get_property()`

```
static void gst_tensor_trainer_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

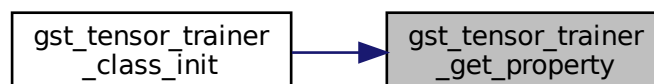
Getter `tensor_trainsink` properties.

Definition at line 395 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



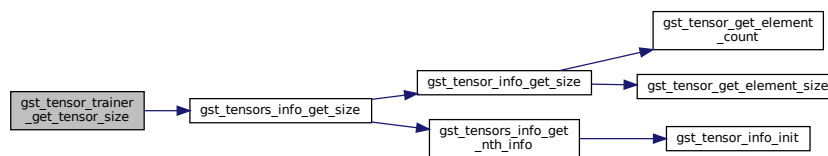
9.66.4.20 `gst_tensor_trainer_get_tensor_size()`

```
gsize gst_tensor_trainer_get_tensor_size (
    GstTensorTrainer * trainer,
    guint index,
    gboolean is_input ) [static]
```

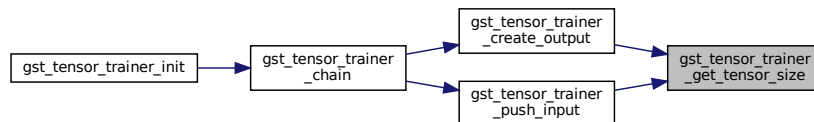
Calculate tensor buffer size.

Definition at line 1250 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.66.4.21 `gst_tensor_trainer_init()`

```
static void gst_tensor_trainer_init (
    GstTensorTrainer * trainer ) [static]
```

Initialize `tensor_trainer`.

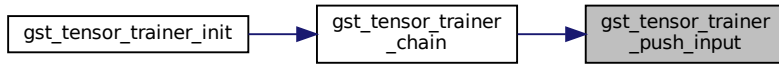
setup sink pad

setup src pad

init properties

Definition at line 262 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



9.66.4.23 gst_tensor_trainer_query_caps()

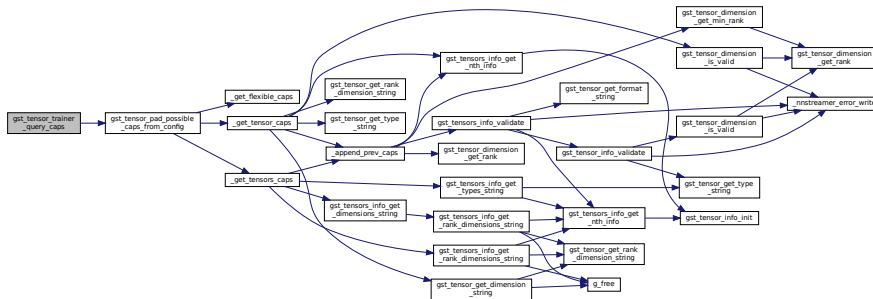
```

static GstCaps * gst_tensor_trainer_query_caps (
    GstTensorTrainer * trainer,
    GstPad * pad,
    GstCaps * filter ) [static]
  
```

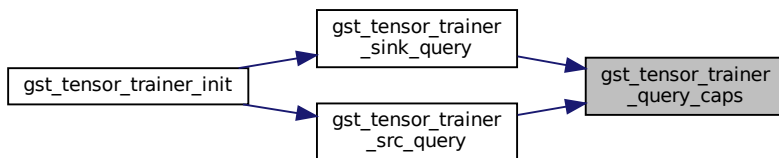
Get pad caps for caps negotiation.

Definition at line 919 of file gstreamer_tensor_trainer.c.

Here is the call graph for this function:



Here is the caller graph for this function:



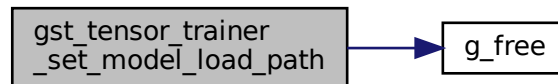
9.66.4.24 `gst_tensor_trainer_set_model_load_path()`

```
static void gst_tensor_trainer_set_model_load_path (
    GstTensorTrainer * trainer,
    const GValue * value ) [static]
```

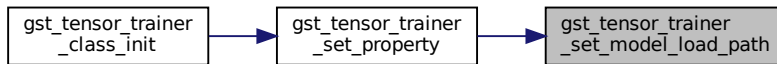
Handle "PROP_MODEL_LOAD_PATH" for set-property.

Definition at line 1183 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



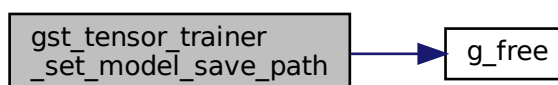
9.66.4.25 `gst_tensor_trainer_set_model_save_path()`

```
static void gst_tensor_trainer_set_model_save_path (
    GstTensorTrainer * trainer,
    const GValue * value ) [static]
```

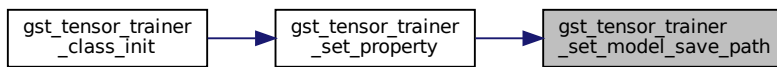
Handle "PROP_MODEL_SAVE_PATH" for set-property.

Definition at line 1170 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.66.4.26 gst_tensor_trainer_set_output_meta()

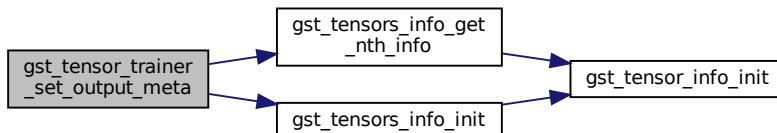
```
static void gst_tensor_trainer_set_output_meta (
    GstTensorTrainer * trainer ) [static]
```

initialize the output tensor dimension

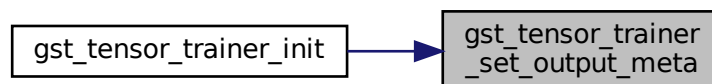
loss, accuracy, val_loss, val_accuracy

Definition at line 1347 of file gsttensor_trainer.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.66.4.27 `gst_tensor_trainer_set_prop_framework()`

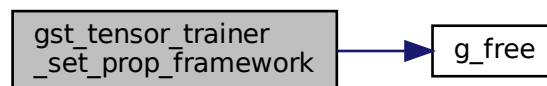
```
static void gst_tensor_trainer_set_prop_framework (
    GstTensorTrainer * trainer,
    const GValue * value ) [static]
```

Handle "PROP_FRAMEWORK" for set-property.

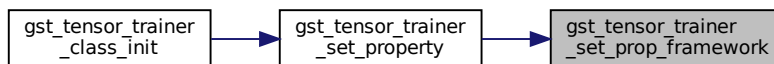
Todo Check valid framework

Definition at line 1143 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



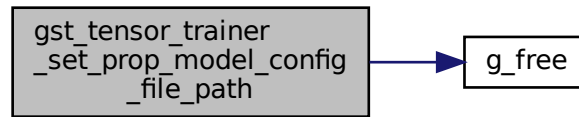
9.66.4.28 `gst_tensor_trainer_set_prop_model_config_file_path()`

```
static void gst_tensor_trainer_set_prop_model_config_file_path (
    GstTensorTrainer * trainer,
    const GValue * value ) [static]
```

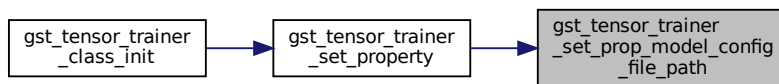
Handle "PROP_MODEL_CONFIG" for set-property.

Definition at line 1157 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



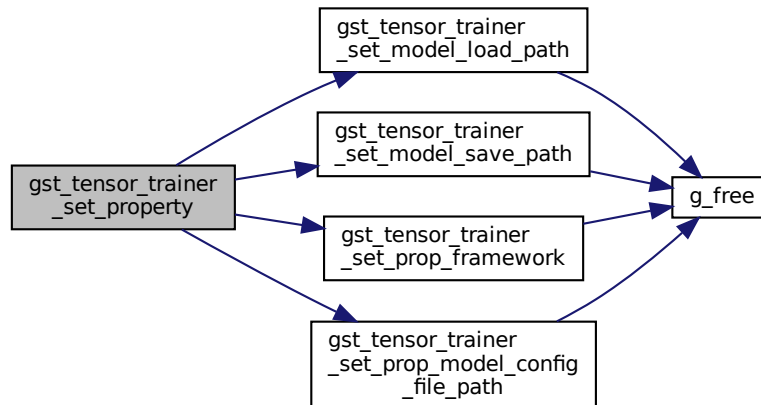
9.66.4.29 `gst_tensor_trainer_set_property()`

```
static void gst_tensor_trainer_set_property (  
    GObject * object,  
    guint prop_id,  
    const GValue * value,  
    GParamSpec * pspec ) [static]
```

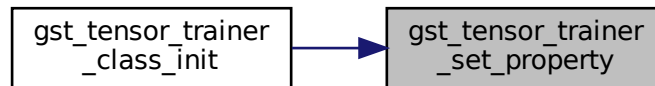
Setter for `tensor_trainsink` properties.

Definition at line 350 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.66.4.30 `gst_tensor_trainer_sink_event()`

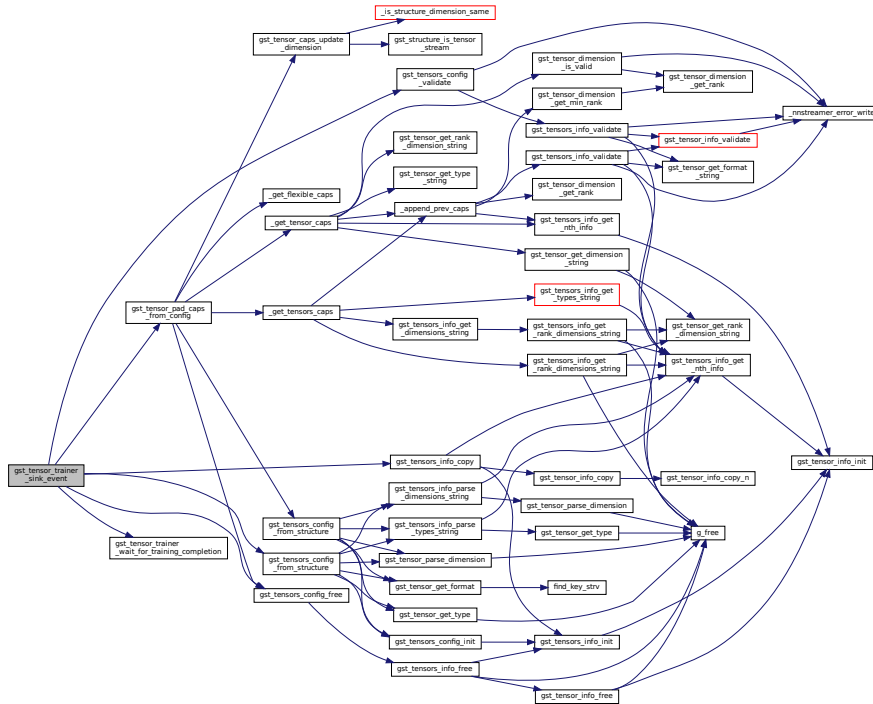
```

static gboolean gst_tensor_trainer_sink_event (
    GstPad * sinkpad,
    GstObject * parent,
    GstEvent * event ) [static]
  
```

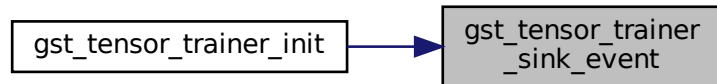
Event handler for sink pad of `tensor_trainer`.

Definition at line 977 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



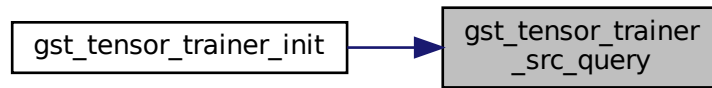
9.66.4.31 gst_tensor_trainer_sink_query()

```
static gboolean gst_tensor_trainer_sink_query (
    GstPad * sinkpad,
    GstObject * parent,
    GstQuery * query ) [static]
```

This function handles sink pad query.

Definition at line 1046 of file gsttensor_trainer.c.

Here is the caller graph for this function:



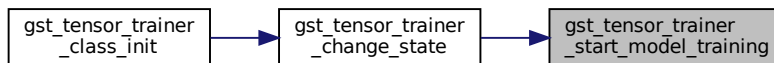
9.66.4.33 `gst_tensor_trainer_start_model_training()`

```
static void gst_tensor_trainer_start_model_training (  
    GstTensorTrainer * trainer ) [static]
```

Start model training.

Definition at line 1308 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



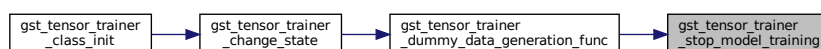
9.66.4.34 `gst_tensor_trainer_stop_model_training()`

```
static void gst_tensor_trainer_stop_model_training (  
    GstTensorTrainer * trainer ) [static]
```

Stop model training.

Definition at line 1328 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



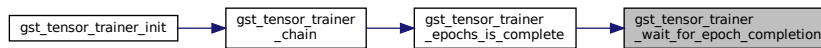
9.66.4.35 `gst_tensor_trainer_wait_for_epoch_completion()`

```
static void gst_tensor_trainer_wait_for_epoch_completion (
    GstTensorTrainer * trainer ) [static]
```

Wait for epoch eompletion.

Definition at line 590 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



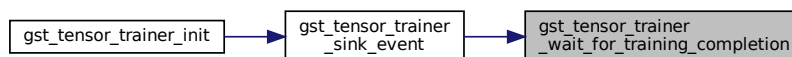
9.66.4.36 `gst_tensor_trainer_wait_for_training_completion()`

```
static void gst_tensor_trainer_wait_for_training_completion (
    GstTensorTrainer * trainer ) [static]
```

Wait for training completion.

Definition at line 955 of file `gsttensor_trainer.c`.

Here is the caller graph for this function:



9.66.4.37 `nnstreamer_trainer_exit()`

```
int nnstreamer_trainer_exit (
    GstTensorTrainerFramework * ttsf )
```

Trainer's sub-plugin may call this to unregister itself.

Parameters

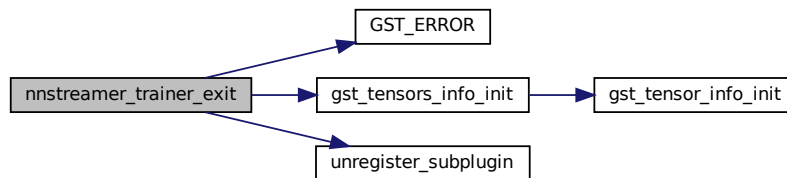
<code>in</code>	<code>ttsf</code>	tensor_trainer sub-plugin to be unregistered.
-----------------	-------------------	---

Returns

TRUE if unregistered. FALSE is failed.

Definition at line 1398 of file gsttensor_trainer.c.

Here is the call graph for this function:

**9.66.4.38 nntstreamer_trainer_notify_event()**

```

void nntstreamer_trainer_notify_event (
    GstTensorTrainerEventNotifier * notifier,
    GstTensorTrainerEventType type,
    void * data )
  
```

Trainer's sub-plugin may call this to send event.

Trainer's sub-plugin call this to notify event.

Parameters

in	<i>notifier</i>	event notifier, sub-plugin must send events with this.
in	<i>type</i>	event type

Definition at line 1425 of file gsttensor_trainer.c.

9.66.4.39 nntstreamer_trainer_probe()

```

int nntstreamer_trainer_probe (
    GstTensorTrainerFramework * ttsp )
  
```

Trainer's sub-plugin should call this function to register itself.

Parameters

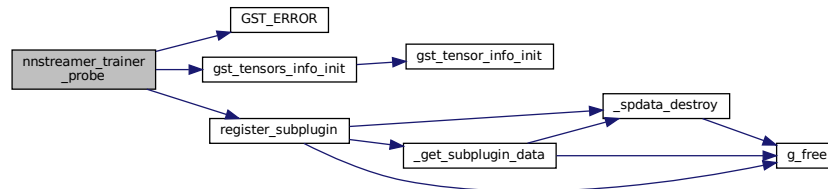
in	<i>ttsp</i>	tensor_trainer sub-plugin to be registered.
----	-------------	---

Returns

TRUE if registered. FALSE is failed or duplicated.

Definition at line 1371 of file gsttensor_trainer.c.

Here is the call graph for this function:

**9.66.5 Variable Documentation****9.66.5.1 sink_template**

```
GstStaticPadTemplate sink_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (SINK_CAPS_STRING))
```

The capabilities of the sink pad.

Definition at line 50 of file gsttensor_trainer.c.

9.66.5.2 src_template

```
GstStaticPadTemplate src_template [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (SRC_CAPS_STRING))
```

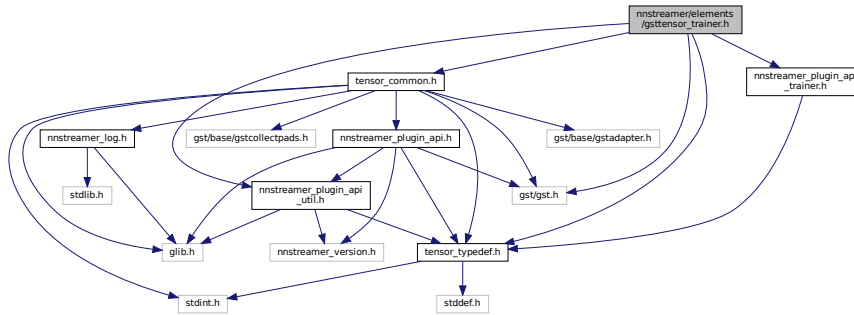
The capabilities of the src pad.

Definition at line 58 of file gsttensor_trainer.c.

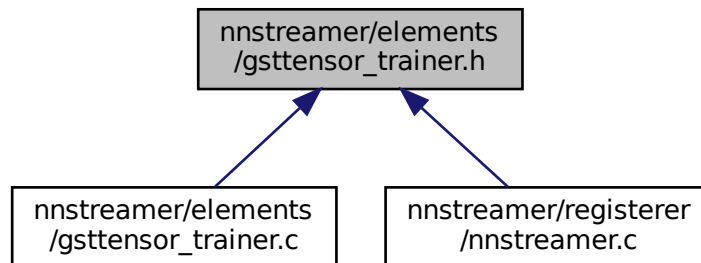
9.67 nnstreamer/elements/gsttensor_trainer.h File Reference

GStreamer plugin to train tensor data using NN Frameworks.

```
#include <gst/gst.h>
#include <tensor_typedef.h>
#include <tensor_common.h>
#include <nnstreamer_plugin_api_util.h>
#include <nnstreamer_plugin_api_trainer.h>
Include dependency graph for gsttensor_trainer.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorTrainer](#)
GstTensorTrainer data structure.
- struct [_GstTensorTrainerClass](#)
GstTensorTrainerClass data structure.

Macros

- #define [GST_TYPE_TENSOR_TRAINER](#) ([gst_tensor_trainer_get_type\(\)](#))
- #define [GST_TENSOR_TRAINER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_TRAINER, GstTensorTrainer\)](#))
- #define [GST_TENSOR_TRAINER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_TRAINER, GstTensorTrainerClass\)](#))
- #define [GST_IS_TENSOR_TRAINER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_TRAINER\)](#))
- #define [GST_IS_TENSOR_TRAINER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_TRAINER\)](#))

Typedefs

- typedef struct [_GstTensorTrainer](#) [GstTensorTrainer](#)
- typedef struct [_GstTensorTrainerClass](#) [GstTensorTrainerClass](#)

Functions

- GType [gst_tensor_trainer_get_type](#) (void)
Function to get type of tensor_trainer.

9.67.1 Detailed Description

GStreamer plugin to train tensor data using NN Frameworks.

Copyright (C) 2022 Samsung Electronics Co., Ltd.

Date

20 October 2022

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

9.67.2 Macro Definition Documentation

9.67.2.1 GST_IS_TENSOR_TRAINER

```
#define GST_IS_TENSOR_TRAINER(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_TRAINER))
```

Definition at line 32 of file `gsttensor_trainer.h`.

9.67.2.2 GST_IS_TENSOR_TRAINER_CLASS

```
#define GST_IS_TENSOR_TRAINER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_TRAINER))
```

Definition at line 34 of file `gsttensor_trainer.h`.

9.67.2.3 GST_TENSOR_TRAINER

```
#define GST_TENSOR_TRAINER(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_TRAINER, GstTensorTrainer))
```

Definition at line 28 of file gstreamer/gsttensor_trainer.h.

9.67.2.4 GST_TENSOR_TRAINER_CLASS

```
#define GST_TENSOR_TRAINER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_TRAINER, GstTensorTrainerClass))
```

Definition at line 30 of file gstreamer/gsttensor_trainer.h.

9.67.2.5 GST_TYPE_TENSOR_TRAINER

```
#define GST_TYPE_TENSOR_TRAINER (gst_tensor_trainer_get_type())
```

Definition at line 26 of file gstreamer/gsttensor_trainer.h.

9.67.3 Typedef Documentation

9.67.3.1 GstTensorTrainer

```
typedef struct _GstTensorTrainer GstTensorTrainer
```

Definition at line 37 of file gstreamer/gsttensor_trainer.h.

9.67.3.2 GstTensorTrainerClass

```
typedef struct _GstTensorTrainerClass GstTensorTrainerClass
```

Definition at line 38 of file gstreamer/gsttensor_trainer.h.

9.67.4 Function Documentation

9.67.4.1 `gst_tensor_trainer_get_type()`

```
GType gst_tensor_trainer_get_type (
    void )
```

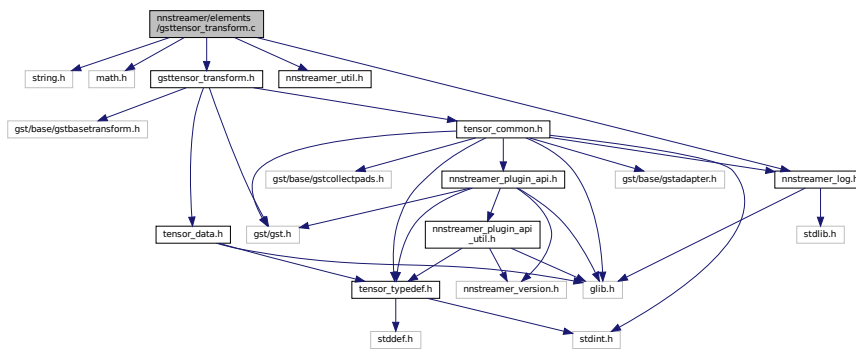
Function to get type of `tensor_trainer`.

9.68 `nnstreamer/elements/gsttensor_transform.c` File Reference

GStreamer plugin to transform tensor dimension or type.

```
#include <string.h>
#include <math.h>
#include <nnstreamer_log.h>
#include <nnstreamer_util.h>
#include "gsttensor_transform.h"
```

Include dependency graph for `gsttensor_transform.c`:



Macros

- `#define DBG` (Ifilter->silent)
Macro for debug mode.
- `#define GST_CAT_DEFAULT` `gst_tensor_transform_debug`
- `#define CAPS_STRING` `GST_TENSOR_CAP_DEFAULT` ",," `GST_TENSORS_CAP_MAKE` ("{ static, flexible }")
- `#define REGEX_DIMCHG_OPTION` `"^([0-9]|1[0-5]):([0-9]|1[0-5])$"`
- `#define REGEX_TYPECAST_OPTION` `"(^([u]?int(8|16|32|64)$|^float(16|32|64)$)"`
- `#define REGEX_TRANSPOSE_OPTION` `"^(?:([0-2]):(?:!\.*\1)){3}$"`
- `#define REGEX_STAND_OPTION` `"^(default|dc-average):([u]?int(8|16|32|64)|float(16|32|64))?(,per-channel:(true|false))?$"`
- `#define REGEX_CLAMP_OPTION`
- `#define REGEX_PADDING_OPTION` `"^((left|right|top|bottom|front|back):(\d)(,)?)+(layout:(NCHW|NH←WC))?$"`
- `#define REGEX_ARITH_OPTION`
- `#define REGEX_ARITH_OPTION_TYPECAST` `"(typecast:([u]?int(8|16|32|64)|float(16|32|64)))"`
- `#define NNS_TENSOR_TRANSPOSE_RANK_LIMIT` (4)
The transpose rank is fixed to 4. This RANK does not affect other/tensors(s)'s NNS_TENSOR_RANK_LIMIT.
- `#define NNS_TENSOR_PADDING_RANK_LIMIT` (3)

The padding rank is fixed to 3. This RANK does not affect other/tensors(s)'s NNS_TENSOR_RANK_LIMIT.

- #define [DEFAULT_ACCELERATION FALSE](#)
Flag to set orc acceleration.
- #define [gst_tensor_transform_parent_class](#) parent_class
- #define [GST_TYPE_TENSOR_TRANSFORM_MODE](#) ([gst_tensor_transform_mode_get_type](#) ())
- #define [_conv_to_f16](#)(intype, o, i, n) do { [float16_not_supported](#) (); } while (0)
- #define [_conv_from_f16](#)(otype, o, i, n) do { [float16_not_supported](#) (); } while (0)
- #define [_op_float16](#)(i, n, v, op) do { [float16_not_supported](#) (); } while (0)
- #define [handle_operator](#)(d, v, oper, vtype)
Macro for operator.
- #define [transposeloop](#)(cl, ck, cj, ci, sl, sk, sj, si, typesize)

Enumerations

- enum {
[PROP_0](#), [PROP_SILENT](#), [PROP_MODE](#), [PROP_OPTION](#),
[PROP_ACCELERATION](#), [PROP_APPLY](#), [PROP_TRANSPOSE_RANK_LIMIT](#) }
tensor_transform properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_transform_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorTransform](#), [gst_tensor_transform](#), [GST_TYPE_BASE_TRANSFORM](#))
- static void [gst_tensor_transform_set_property](#) (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)
Set property (gst element vmethod)
- static void [gst_tensor_transform_get_property](#) (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)
Get property (gst element vmethod)
- static void [gst_tensor_transform_finalize](#) (GObject *object)
Function to finalize instance (gst element vmethod)
- static GstFlowReturn [gst_tensor_transform_transform](#) (GstBaseTransform *trans, GstBuffer *inbuf, GstBuffer *outbuf)
non-ip transform. required vmethod for BaseTransform class.
- static GstCaps * [gst_tensor_transform_transform_caps](#) (GstBaseTransform *trans, GstPadDirection direction, GstCaps *caps, GstCaps *filtercap)
configure srcpad cap from "proposed" cap. (required vmethod for BaseTransform)
- static GstCaps * [gst_tensor_transform_fixate_caps](#) (GstBaseTransform *trans, GstPadDirection direction, GstCaps *caps, GstCaps *othercaps)
fixate caps. required vmethod of BaseTransform
- static gboolean [gst_tensor_transform_set_caps](#) (GstBaseTransform *trans, GstCaps *incaps, GstCaps *outcaps)
set caps. required vmethod of BaseTransform
- static gboolean [gst_tensor_transform_transform_size](#) (GstBaseTransform *trans, GstPadDirection direction, GstCaps *caps, gsize size, GstCaps *othercaps, gsize *othersize)
Tell the framework the required size of buffer based on the info of the other side pad. Note that this is always the same with the input. optional vmethod of BaseTransform.
- static gboolean [gst_tensor_transform_convert_dimension](#) ([GstTensorTransform](#) *filter, GstPadDirection direction, guint idx, const [GstTensorInfo](#) *in_info, [GstTensorInfo](#) *out_info)
Dimension conversion calculation.
- static GType [gst_tensor_transform_mode_get_type](#) (void)

- A private function to register GEnumValue array for the 'mode' property to a GType and return it.*
- static void `gst_tensor_transform_class_init` (`GstTensorTransformClass *klass`)
initialize the tensor_transform's class
 - static void `gst_tensor_transform_init` (`GstTensorTransform *filter`)
initialize the new element (G_DEFINE_TYPE requires this) instantiate pads and add them to element set pad callback functions initialize instance structure
 - static `tensor_transform_operator` `gst_tensor_transform_get_operator` (`const gchar *str`)
Get the corresponding operator from the string value.
 - static `tensor_transform_stand_mode` `gst_tensor_transform_get_stand_mode` (`const gchar *str`)
Get the corresponding mode from the string value.
 - static void `float16_not_supported` (`void`)
Generate error if float16 is required.
 - `while` (0)
 - static gboolean `gst_tensor_transform_set_option_data` (`GstTensorTransform *filter`)
Setup internal data (data_ in GstTensorTransform)*
 - static `GstFlowReturn` `gst_tensor_transform_dimchg` (`GstTensorTransform *filter`, `GstTensorInfo *in_info`, `GstTensorInfo *out_info`, `const uint8_t *inptr`, `uint8_t *outptr`)
subrouting for tensor-transform, "dimchg" case.
 - static `GstFlowReturn` `gst_tensor_transform_typecast` (`GstTensorTransform *filter`, `GstTensorInfo *in_info`, `GstTensorInfo *out_info`, `const uint8_t *inptr`, `uint8_t *outptr`)
subrouting for tensor-transform, "typecast" case.
 - static `GstFlowReturn` `gst_tensor_transform_arithmetic` (`GstTensorTransform *filter`, `GstTensorInfo *in_info`, `GstTensorInfo *out_info`, `const uint8_t *inptr`, `uint8_t *outptr`)
subrouting for tensor-transform, "arithmetic" case.
 - static `GstFlowReturn` `gst_tensor_transform_transpose` (`GstTensorTransform *filter`, `GstTensorInfo *in_info`, `GstTensorInfo *out_info`, `const uint8_t *inptr`, `uint8_t *outptr`)
subrouting for tensor-transform, "transpose" case.
 - static `GstFlowReturn` `gst_tensor_transform_stand` (`GstTensorTransform *filter`, `GstTensorInfo *in_info`, `GstTensorInfo *out_info`, `const uint8_t *inptr`, `uint8_t *outptr`)
subrouting for tensor-transform, "stand" case. : pixel = abs((pixel - average(tensor))/(std(tensor) + val))
 - static `GstFlowReturn` `gst_tensor_transform_clamp` (`GstTensorTransform *filter`, `GstTensorInfo *in_info`, `GstTensorInfo *out_info`, `const uint8_t *inptr`, `uint8_t *outptr`)
subrouting for tensor-transform, "clamp" case. : pixel = if (pixel > max) ? max : if (pixel < min) ? min : pixel
 - static `GstFlowReturn` `gst_tensor_transform_padding` (`GstTensorTransform *filter`, `GstTensorInfo *in_info`, `GstTensorInfo *out_info`, `const uint8_t *inptr`, `uint8_t *outptr`)
subrouting for tensor-transform, "padding" case.
 - static gboolean `gst_tensor_transform_read_caps` (`GstTensorTransform *filter`, `const GstCaps *caps`, `GstTensorsConfig *config`)
Read cap, parse tensor configuration (dim/type) from the cap.

Variables

- static const gchar * `gst_tensor_transform_stand_string` []
- static const gchar * `gst_tensor_transform_operator_string` []
- static `GstStaticPadTemplate` `sink_factory`
The capabilities of the inputs.
- static `GstStaticPadTemplate` `src_factory`
The capabilities of the outputs.
- `break`
*vtype += (v)->data_##vtype; *
- case `GTT_OP_MUL`
- default `__pad0__`
- default Unknown `operator%d", oper)`
- return `FALSE`

9.68.1 Detailed Description

GStreamer plugin to transform tensor dimension or type.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

10 Jul 2018

See also

<https://github.com/nstreamer/nstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug This is NYI.

9.68.2 Macro Definition Documentation

9.68.2.1 `_conv_from_f16`

```
#define _conv_from_f16(  
    otype,  
    o,  
    i,  
    n ) do { float16_not_supported (); } while (0)
```

Definition at line 465 of file `gsttensor_transform.c`.

9.68.2.2 `_conv_to_f16`

```
#define _conv_to_f16(  
    intype,  
    o,  
    i,  
    n ) do { float16_not_supported (); } while (0)
```

Definition at line 464 of file `gsttensor_transform.c`.

9.68.2.3 `_op_float16`

```
#define _op_float16(
    i,
    n,
    v,
    op ) do { float16_not_supported (); } while (0)
```

Definition at line 466 of file `gsttensor_transform.c`.

9.68.2.4 `CAPS_STRING`

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_MAKE ("{ static, flexible }")
```

Definition at line 69 of file `gsttensor_transform.c`.

9.68.2.5 `DBG`

```
#define DBG (!filter->silent)
```

Macro for debug mode.

SECTION:element-tensor_transform

A filter that transforms tensors dimension or type. The input and output is always in the format of other/tensor or other/tensors.

<refsect2> <title>Example launch line</title> |[`gst-launch -v -m fakesrc ! tensor_converter ! tensor_transform mode=dimchg option=0:2 ! fakesink silent=TRUE]` | <title>How to use dimchg</title> |[`option=0:2 # Move 0th dim to 2nd dim. I.e., [a][H][W][C] ==> [a][C][H][W]`] | </refsect2>

Definition at line 64 of file `gsttensor_transform.c`.

9.68.2.6 `DEFAULT_ACCELERATION`

```
#define DEFAULT_ACCELERATION FALSE
```

Flag to set orc acceleration.

Definition at line 115 of file `gsttensor_transform.c`.

9.68.2.7 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_transform_debug
```

Definition at line 68 of file gsttensor_transform.c.

9.68.2.8 gst_tensor_transform_parent_class

```
#define gst_tensor_transform_parent_class parent_class
```

Definition at line 148 of file gsttensor_transform.c.

9.68.2.9 GST_TYPE_TENSOR_TRANSFORM_MODE

```
#define GST_TYPE_TENSOR_TRANSFORM_MODE (gst_tensor_transform_mode_get_type ())
```

Definition at line 176 of file gsttensor_transform.c.

9.68.2.10 handle_operator

```
#define handle_operator(  
    d,  
    v,  
    oper,  
    vtype )
```

Value:

```
do { \  
    switch (oper) { \  
        case GTT_OP_ADD: \  
            (d)->data._
```

Macro for operator.

Definition at line 573 of file gsttensor_transform.c.

9.68.2.11 NNS_TENSOR_PADDING_RANK_LIMIT

```
#define NNS_TENSOR_PADDING_RANK_LIMIT (3)
```

The padding rank is fixed to 3. This RANK does not affect other/tensors(s)'s NNS_TENSOR_RANK_LIMIT.

Definition at line 93 of file gsttensor_transform.c.

9.68.2.12 NNS_TENSOR_TRANSPOSE_RANK_LIMIT

```
#define NNS_TENSOR_TRANSPOSE_RANK_LIMIT (4)
```

The transpose rank is fixed to 4. This RANK does not affect other/tensors(s)'s NNS_TENSOR_RANK_LIMIT.

Definition at line 87 of file gsttensor_transform.c.

9.68.2.13 REGEX_ARITH_OPTION

```
#define REGEX_ARITH_OPTION
```

Value:

```
"^(typecast:([u]?int(8|16|32|64)|float(16|32|64)),)?"\
"(per-channel:(false|true@[0-9]+),)?"\
"(((add|mul|div):([-+]?[0-9]*\\.[0-9]+|[eE]([-+]?[0-9]+)?))+@[0-9]+)?(,|))+$"
```

Definition at line 77 of file gsttensor_transform.c.

9.68.2.14 REGEX_ARITH_OPTION_TYPECAST

```
#define REGEX_ARITH_OPTION_TYPECAST "(typecast:([u]?int(8|16|32|64)|float(16|32|64)))"
```

Definition at line 81 of file gsttensor_transform.c.

9.68.2.15 REGEX_CLAMP_OPTION

```
#define REGEX_CLAMP_OPTION
```

Value:

```
"^((( [-+]?[0-9]*\\.[0-9]+|[eE]([-+]?[0-9]+)?))):"\
"((( [-+]?[0-9]*\\.[0-9]+|[eE]([-+]?[0-9]+)?))$"
```

Definition at line 74 of file gsttensor_transform.c.

9.68.2.16 REGEX_DIMCHG_OPTION

```
#define REGEX_DIMCHG_OPTION "^([0-9]|1[0-5]):([0-9]|1[0-5])$"
```

Definition at line 70 of file gsttensor_transform.c.

9.68.2.17 REGEX_PADDING_OPTION

```
#define REGEX_PADDING_OPTION "^( (left|right|top|bottom|front|back) : (\\d) (,) ?) + (layout : (NCHW|NH↔WC)) ? $"
```

Definition at line 76 of file `gsttensor_transform.c`.

9.68.2.18 REGEX_STAND_OPTION

```
#define REGEX_STAND_OPTION "^(default|dc-average) ( : ([u]?int (8|16|32|64)|float (16|32|64)) ) ? (,per-channel↔ : (true|false)) ? $"
```

Definition at line 73 of file `gsttensor_transform.c`.

9.68.2.19 REGEX_TRANSPOSE_OPTION

```
#define REGEX_TRANSPOSE_OPTION "^(?:([0-2]) : (?!.*\\1)) {3}3 $"
```

Definition at line 72 of file `gsttensor_transform.c`.

9.68.2.20 REGEX_TYPECAST_OPTION

```
#define REGEX_TYPECAST_OPTION " (^ [u]?int (8|16|32|64) $ | ^ float (16|32|64) $ ) "
```

Definition at line 71 of file `gsttensor_transform.c`.

9.68.2.21 transposeloop

```
#define transposeloop(
    cl,
    ck,
    cj,
    ci,
    sl,
    sk,
    sj,
    si,
    typesize )
```

Value:

```
do { \
    size_t i, j, k, l;
    int inidx = 0, outidx=0;
    for (cl=0; cl<sl; cl++)
        for (ci=0; ci<si; ci++)
            for (cj=0; cj<sj; cj++)
                for (ck=0; ck<sk; ck++){
                    const uint8_t *_in; \
                    uint8_t *_out; \
                    outidx = si*sj*sk*cl + sj*sk*ci + sk*cj + ck; \
                    inidx = SK*SJ*SI*1 + SJ*SI*k + SI*j + i; \
                    _in = inptr + inidx * typesize; \
                    _out = outptr + outidx * typesize; \
                    nns_memcpy(_out, _in, typesize); \
                }
    } while (0);
```

Macro to run loop for various data types with transpose

Definition at line 1506 of file `gsttensor_transform.c`.

9.68.3 Enumeration Type Documentation

9.68.3.1 anonymous enum

anonymous enum

tensor_transform properties

Enumerator

PROP_0	
PROP_SILENT	
PROP_MODE	
PROP_OPTION	
PROP_ACCELERATION	
PROP_APPLY	
PROP_TRANSPOSE_RANK_LIMIT	

Definition at line 98 of file gsttensor_transform.c.

9.68.4 Function Documentation

9.68.4.1 float16_not_supported()

```
static void float16_not_supported (
    void ) [static]
```

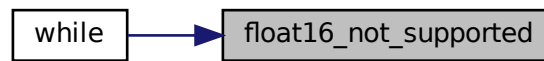
Generate error if float16 is required.

Definition at line 348 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.2 G_DEFINE_TYPE()

```

G_DEFINE_TYPE (
    GstTensorTransform ,
    gst_tensor_transform ,
    GST_TYPE_BASE_TRANSFORM )
  
```

9.68.4.3 GST_DEBUG_CATEGORY_STATIC()

```

GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_transform_debug )
  
```

9.68.4.4 gst_tensor_transform_arithmetic()

```

static GstFlowReturn gst_tensor_transform_arithmetic (
    GstTensorTransform * filter,
    GstTensorInfo * in_info,
    GstTensorInfo * out_info,
    const uint8_t * inptr,
    uint8_t * outptr ) [static]
  
```

subrouting for tensor-transform, "arithmetic" case.

Parameters

	<i>[in/out]</i>	filter "this" pointer
in	<i>in_info</i>	input tensor info
in	<i>out_info</i>	output tensor info
in	<i>inptr</i>	input tensor
out	<i>outptr</i>	output tensor

Returns

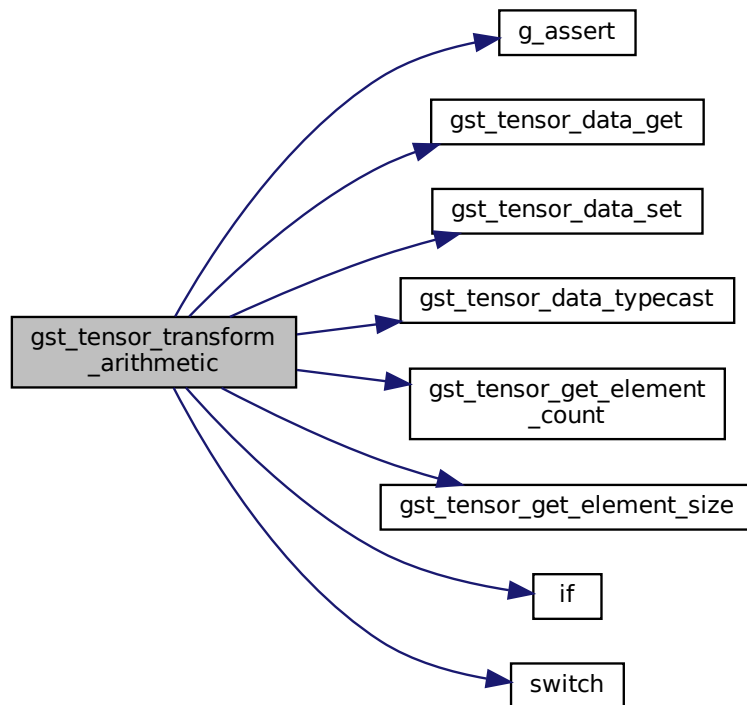
Gst flow status

In case of 3:4:4:1, ch_dim:0 -> #ch: 3, ch_size: 1, ch_offset: 3 ch_dim:1 -> #ch: 4, ch_size: 3, ch_offset: 12
 ch_dim:2 -> #ch: 4, ch_size: 12, ch_offset: 48 ch_dim:3 -> #ch: 1, ch_size: 48, ch_offset: 48 * 4

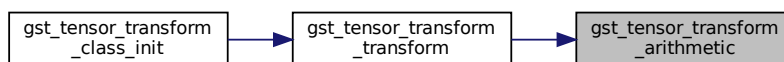
Todo add more options

Definition at line 1330 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.5 `gst_tensor_transform_clamp()`

```
static GstFlowReturn gst_tensor_transform_clamp (
    GstTensorTransform * filter,
    GstTensorInfo * in_info,
    GstTensorInfo * out_info,
    const uint8_t * inptr,
    uint8_t * outptr ) [static]
```

subrouting for tensor-transform, "clamp" case. : pixel = if (pixel > max) ? max : if (pixel < min) ? min : pixel

Parameters

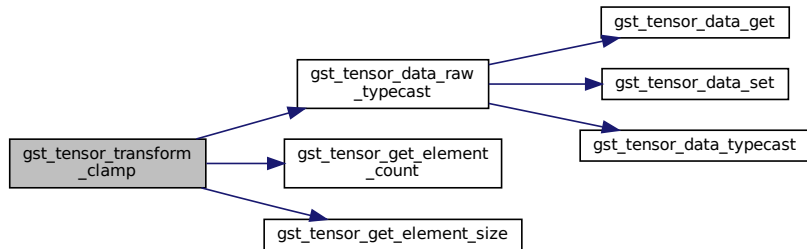
	<i>[in/out]</i>	filter "this" pointer
in	<i>in_info</i>	input tensor info
in	<i>out_info</i>	output tensor info
in	<i>inptr</i>	input tensor
out	<i>outptr</i>	output tensor

Returns

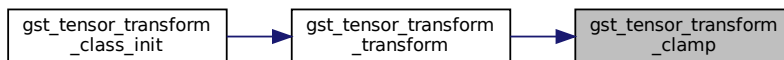
Gst flow status

Definition at line 1725 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



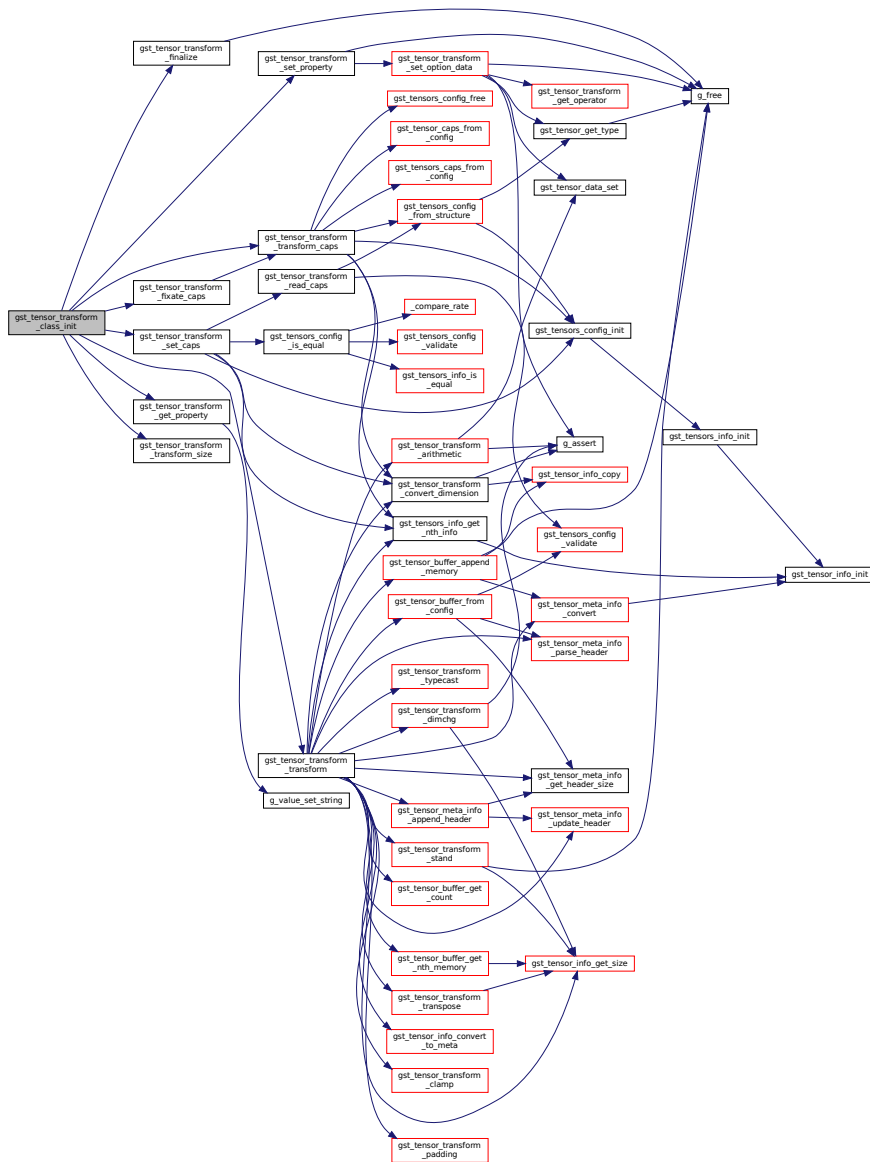
9.68.4.6 gst_tensor_transform_class_init()

```
static void gst_tensor_transform_class_init (
    GstTensorTransformClass * klass ) [static]
```

initialize the tensor_transform's class

Definition at line 224 of file gsttensor_transform.c.

Here is the call graph for this function:



9.68.4.7 `gst_tensor_transform_convert_dimension()`

```
static gboolean gst_tensor_transform_convert_dimension (
    GstTensorTransform * filter,
    GstPadDirection direction,
    guint idx,
    const GstTensorInfo * in_info,
    GstTensorInfo * out_info ) [static]
```

Dimension conversion calculation.

Parameters

in	<i>filter</i>	"this" pointer
in	<i>direction</i>	GST_PAD_SINK if input->output conv
in	<i>idx</i>	index of the input tensors
in	<i>in_info</i>	tensor info structure of source tensor (input if direction is SINK)
out	<i>out_info</i>	tensor info structure of destination tensor (output if direction is SINK)

Returns

TRUE if success

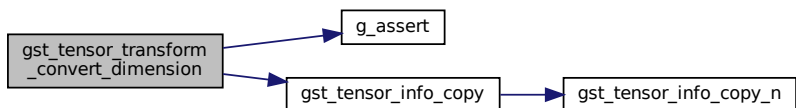
For both directions, dimension does not change

src = SINKPAD / dest = SRCPAD

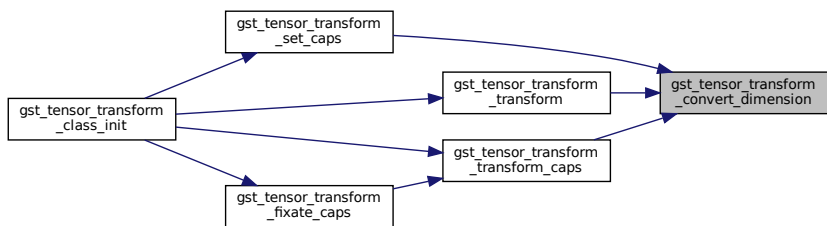
For both directions, dimension does not change

Definition at line 2005 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.8 `gst_tensor_transform_dimchg()`

```
static GstFlowReturn gst_tensor_transform_dimchg (
    GstTensorTransform * filter,
    GstTensorInfo * in_info,
    GstTensorInfo * out_info,
    const uint8_t * inptr,
    uint8_t * outptr ) [static]
```

subrouting for tensor-transform, "dimchg" case.

Parameters

	<i>[in/out]</i>	filter "this" pointer
in	<i>in_info</i>	input tensor info
in	<i>out_info</i>	output tensor info
in	<i>inptr</i>	input tensor
out	<i>outptr</i>	output tensor

Returns

Gst flow status

Useless memcopy. Do not call this or

Todo do "IP" operation

Smaller-loop-ed a to larger-loop-ed b E.g., [N][H][W][c] (c:W:H:N) --> [N][c][H][W] (W:H:c:N)

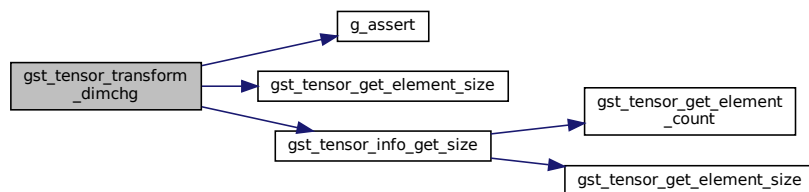
Todo CRITICAL-TODO: Optimize the performance!

Larger-loop-ed a to smaller-loop-ed b E.g., [N][c][H][W] (W:H:c:N) --> [N][H][W][c] (c:W:H:N)

Todo NYI

Definition at line 1197 of file `gsttensor_transform.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



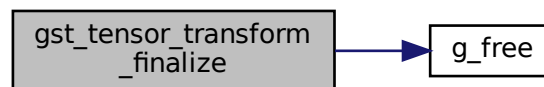
9.68.4.9 `gst_tensor_transform_finalize()`

```
static void gst_tensor_transform_finalize (  
    GObject * object ) [static]
```

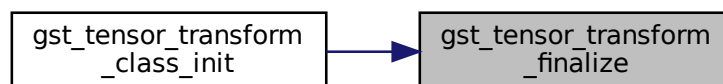
Function to finalize instance (gst element vmethod)

Definition at line 1163 of file `gsttensor_transform.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



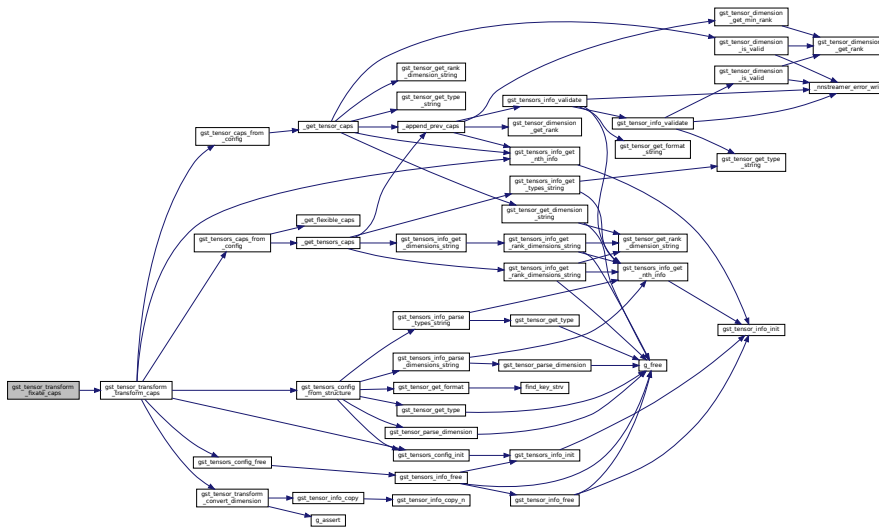
9.68.4.10 `gst_tensor_transform_fixate_caps()`

```
static GstCaps * gst_tensor_transform_fixate_caps (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    GstCaps * othercaps ) [static]
```

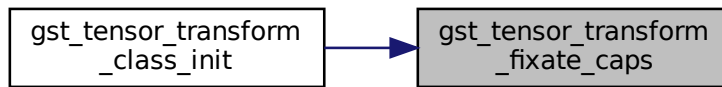
fixate caps. required vmethod of BaseTransform

Definition at line 2222 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.11 `gst_tensor_transform_get_operator()`

```
static tensor_transform_operator gst_tensor_transform_get_operator (
    const gchar * str ) [static]
```

Get the corresponding operator from the string value.

Parameters

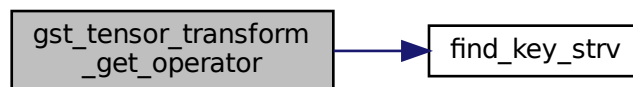
in	<i>str</i>	The string value for the operator
----	------------	-----------------------------------

Returns

corresponding operator for the string (GTT_OP_UNKNOWN for errors)

Definition at line 319 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.12 `gst_tensor_transform_get_property()`

```

static void gst_tensor_transform_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
  
```

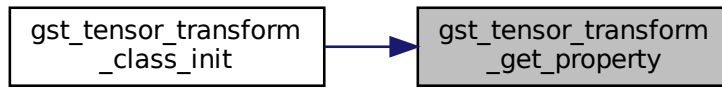
Get property (gst element vmethod)

Definition at line 1107 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.13 `gst_tensor_transform_get_stand_mode()`

```
static tensor_transform_stand_mode gst_tensor_transform_get_stand_mode (
    const gchar * str ) [static]
```

Get the corresponding mode from the string value.

Parameters

<code>in</code>	<code>str</code>	The string value for the mode
-----------------	------------------	-------------------------------

Returns

corresponding mode for the string. `STAND_END` for errors

Definition at line 334 of file `gsttensor_transform.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



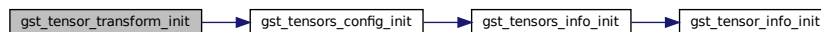
9.68.4.14 `gst_tensor_transform_init()`

```
static void gst_tensor_transform_init (
    GstTensorTransform * filter ) [static]
```

initialize the new element (G_DEFINE_TYPE requires this) instantiate pads and add them to element set pad callback functions initialize instance structure

Definition at line 299 of file `gsttensor_transform.c`.

Here is the call graph for this function:



9.68.4.15 `gst_tensor_transform_mode_get_type()`

```
static GType gst_tensor_transform_mode_get_type (
    void ) [static]
```

A private function to register GEnumValue array for the 'mode' property to a GType and return it.

Definition at line 182 of file `gsttensor_transform.c`.

9.68.4.16 `gst_tensor_transform_padding()`

```
static GstFlowReturn gst_tensor_transform_padding (
    GstTensorTransform * filter,
    GstTensorInfo * in_info,
    GstTensorInfo * out_info,
    const uint8_t * inptr,
    uint8_t * outptr ) [static]
```

subrouting for tensor-transform, "padding" case.

Parameters

	<i>[in/out]</i>	filter "this" pointer
in	<i>in_info</i>	input tensor info
in	<i>out_info</i>	output tensor info
in	<i>inptr</i>	input tensor
out	<i>outptr</i>	output tensor

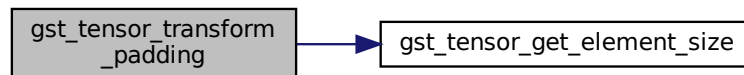
Returns

Gst flow status

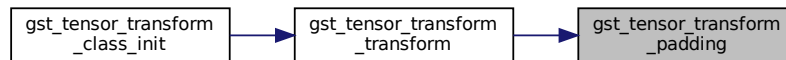
Todo Add constant option instead of using zero padding value

Definition at line 1762 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.68.4.17 gst_tensor_transform_read_caps()**

```

static gboolean gst_tensor_transform_read_caps (
    GstTensorTransform * filter,
    const GstCaps * caps,
    GstTensorsConfig * config ) [static]
  
```

Read cap, parse tensor configuration (dim/type) from the cap.

Parameters

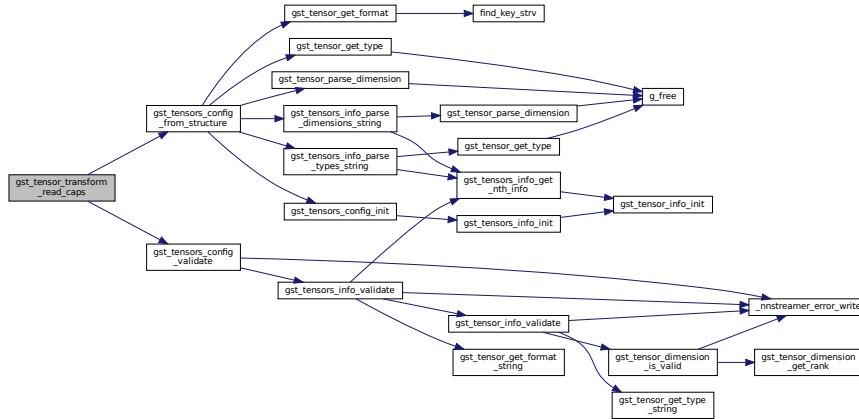
in	<i>filter</i>	"this" pointer
in	<i>caps</i>	The input caps to be read
out	<i>config</i>	configured tensor info

Returns

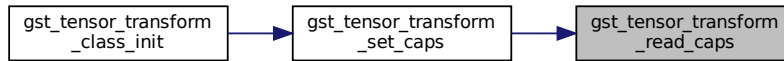
TRUE if successful (both dim/type read). FALSE if not.

Definition at line 1978 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.18 gst_tensor_transform_set_caps()

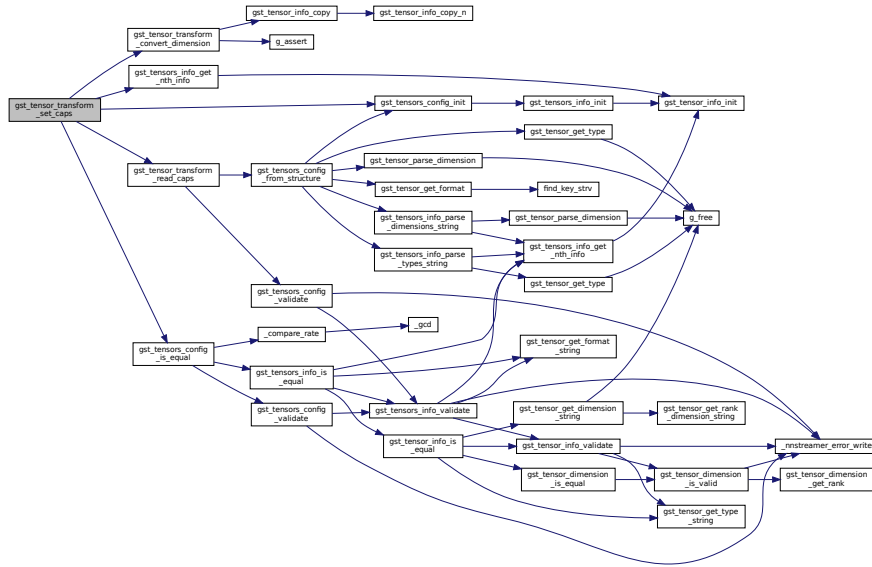
```

static gboolean gst_tensor_transform_set_caps (
    GstBaseTransform * trans,
    GstCaps * incaps,
    GstCaps * outcaps ) [static]
    
```

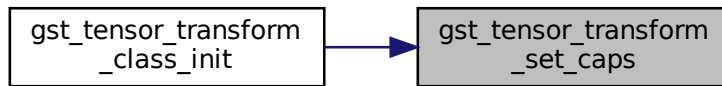
set caps. required vmethod of BaseTransform

Definition at line 2249 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.19 gst_tensor_transform_set_option_data()

```
static gboolean gst_tensor_transform_set_option_data (
    GstTensorTransform * filter ) [static]
```

Setup internal data (data_* in GstTensorTransform)

Parameters

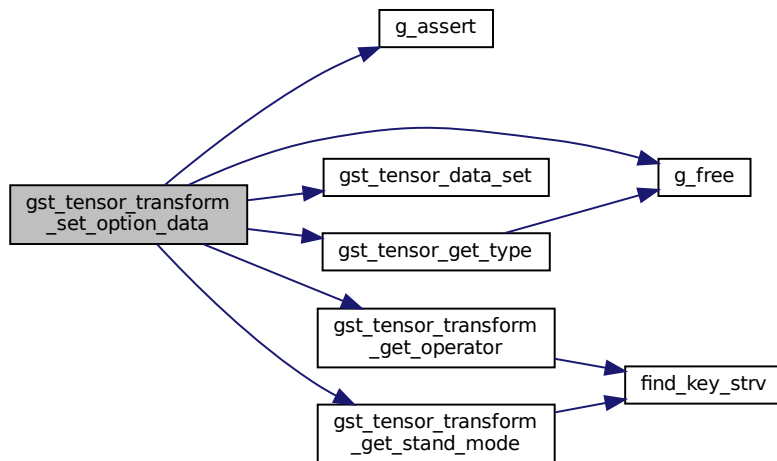
<i>[in/out]</i>	filter "this" pointer. mode & option MUST BE set already.
-----------------	---

Return values

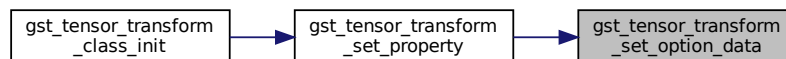
<i>TRUE</i>	if OK or operation-skipped, <i>FALSE</i> if fatal-error.
-------------	--

Definition at line 663 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.20 gst_tensor_transform_set_property()

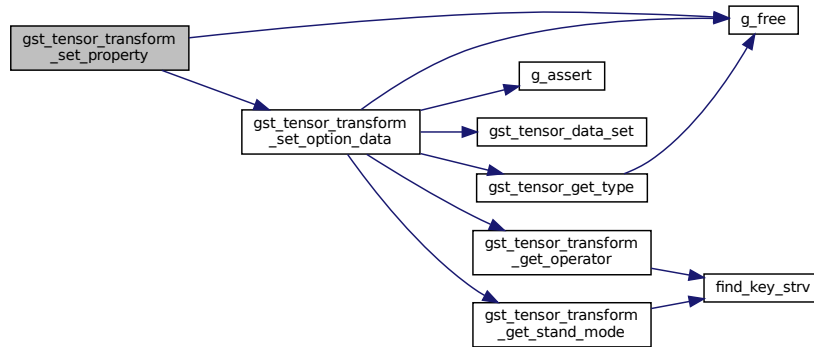
```

static void gst_tensor_transform_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
  
```

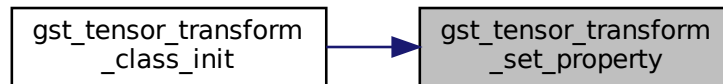
Set property (gst element vmethod)

Definition at line 1040 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.21 gst_tensor_transform_stand()

```

static GstFlowReturn gst_tensor_transform_stand (
    GstTensorTransform * filter,
    GstTensorInfo * in_info,
    GstTensorInfo * out_info,
    const uint8_t * inptr,
    uint8_t * outptr ) [static]
  
```

subrouting for tensor-transform, "stand" case. : $\text{pixel} = \text{abs}((\text{pixel} - \text{average}(\text{tensor})) / (\text{std}(\text{tensor}) + \text{val}))$

Parameters

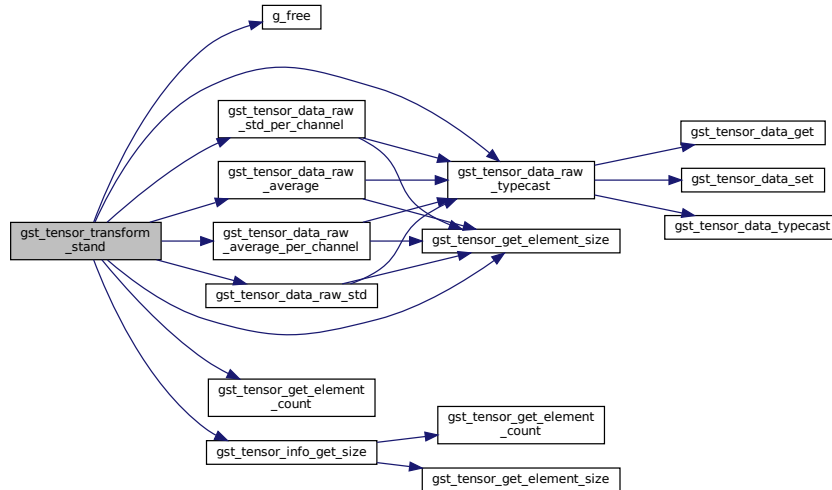
	[in/out]	filter "this" pointer
in	<i>in_info</i>	input tensor info
in	<i>out_info</i>	output tensor info
in	<i>inptr</i>	input tensor
out	<i>outptr</i>	output tensor

Returns

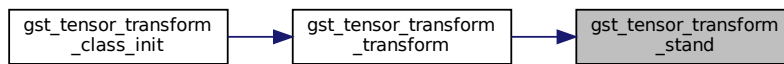
Gst flow status

Definition at line 1605 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.22 gst_tensor_transform_transform()

```
static GstFlowReturn gst_tensor_transform_transform (
    GstBaseTransform * trans,
    GstBuffer * inbuf,
    GstBuffer * outbuf ) [static]
```

non-ip transform. required vmethod for BaseTransform class.

Parameters

	<i>[in/out]</i>	trans "super" pointer
in	<i>inbuf</i>	The input gst buffer
out	<i>outbuf</i>	The output gst buffer

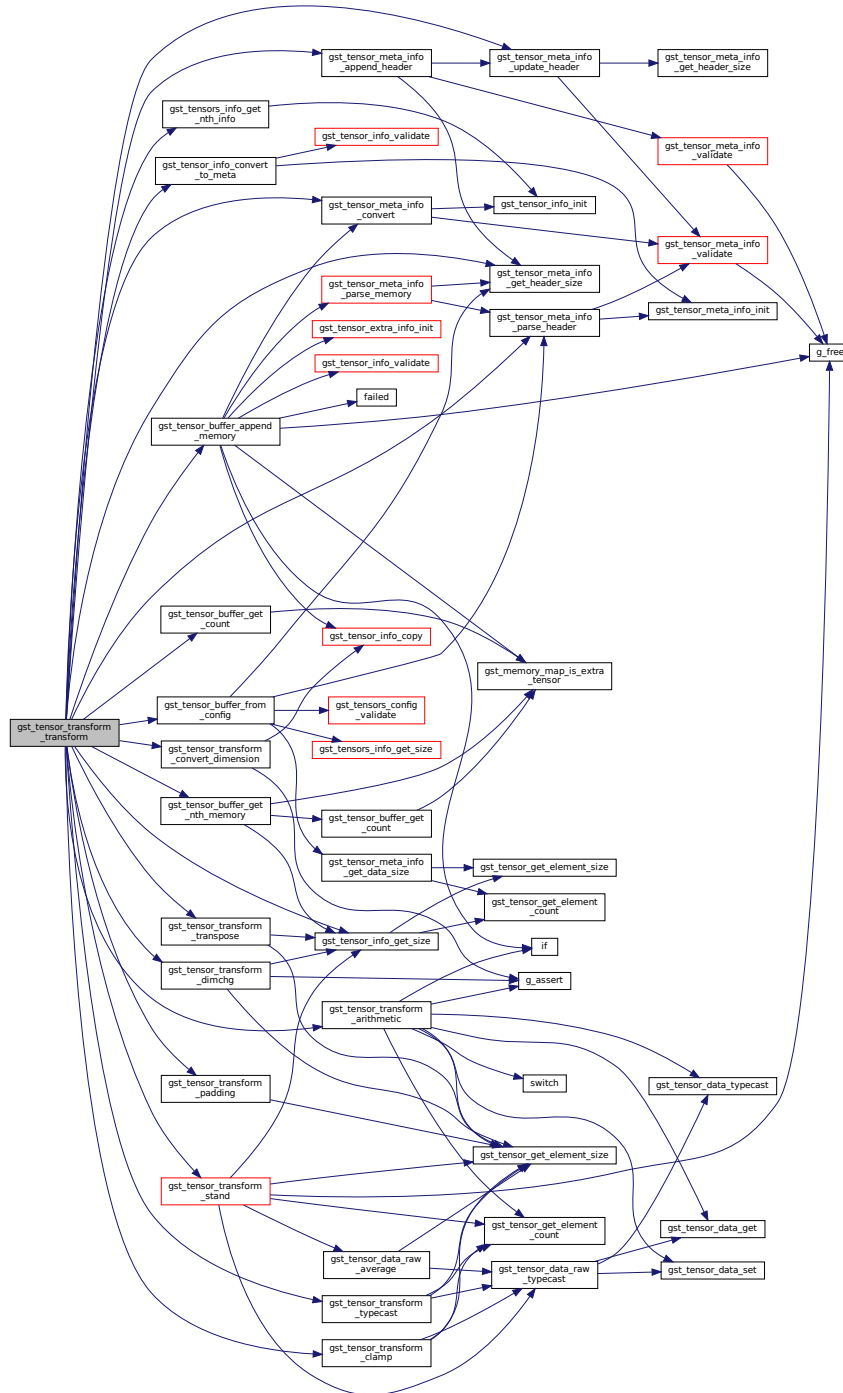
Returns

Gst Flow Status

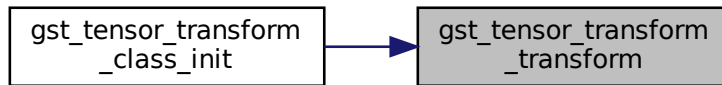
Todo max rank supported in tensor-transform is 4

Definition at line 1815 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.23 `gst_tensor_transform_transform_caps()`

```

static GstCaps * gst_tensor_transform_transform_caps (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    GstCaps * filtercap ) [static]
  
```

configure srcpad cap from "proposed" cap. (required vmethod for BaseTransform)

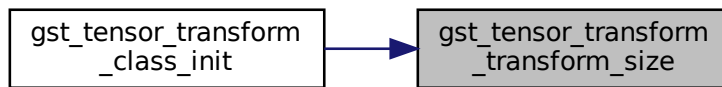
Parameters

<i>trans</i>	("this" pointer)
<i>direction</i>	(why do we need this?)
<i>caps</i>	sinkpad cap
<i>filtercap</i>	this element's cap (don't know specifically.)

Todo Get to know what the heck is @filtercap and use it!

Definition at line 2139 of file `gsttensor_transform.c`.

Here is the caller graph for this function:



9.68.4.25 gst_tensor_transform_transpose()

```

static GstFlowReturn gst_tensor_transform_transpose (
    GstTensorTransform * filter,
    GstTensorInfo * in_info,
    GstTensorInfo * out_info,
    const uint8_t * inptr,
    uint8_t * outptr ) [static]
  
```

subrouting for tensor-transform, "transpose" case.

Parameters

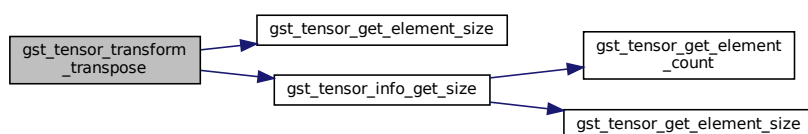
	<i>[in/out]</i>	filter "this" pointer
in	<i>in_info</i>	input tensor info
in	<i>out_info</i>	output tensor info
in	<i>inptr</i>	input tensor
out	<i>outptr</i>	output tensor

Returns

Gst flow status

Definition at line 1533 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.68.4.26 gst_tensor_transform_typecast()

```

static GstFlowReturn gst_tensor_transform_typecast (
    GstTensorTransform * filter,
    GstTensorInfo * in_info,
    GstTensorInfo * out_info,
    const uint8_t * inptr,
    uint8_t * outptr ) [static]
  
```

subrouting for tensor-transform, "typecast" case.

Parameters

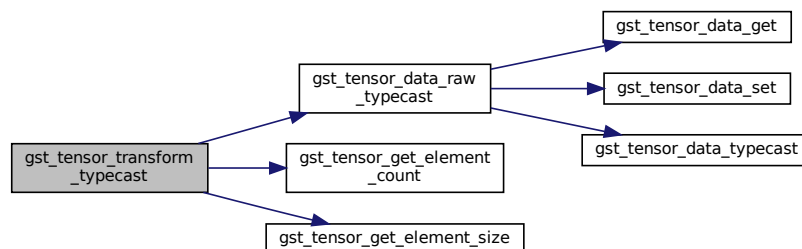
	[in/out]	filter "this" pointer
in	<i>in_info</i>	input tensor info
in	<i>out_info</i>	output tensor info
in	<i>inptr</i>	input tensor
out	<i>outptr</i>	output tensor

Returns

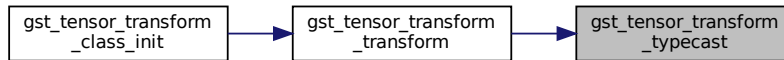
Gst flow status

Definition at line 1292 of file gsttensor_transform.c.

Here is the call graph for this function:



Here is the caller graph for this function:

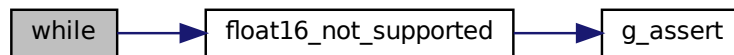


9.68.4.27 while()

```
while (
    0 )
```

Definition at line 592 of file gstreamer_transform.c.

Here is the call graph for this function:



9.68.5 Variable Documentation

9.68.5.1 __pad0__

```
default __pad0__
```

Definition at line 589 of file gstreamer_transform.c.

9.68.5.2 break

```
d data _ break
```

```
vtype += (v)->data.##_vtype; \
```

```
vtype /= (v)->data.##_vtype; \
```

Definition at line 577 of file gstreamer_transform.c.

9.68.5.3 FALSE

```
return FALSE
```

Definition at line 590 of file gsttensor_transform.c.

9.68.5.4 gst_tensor_transform_operator_string

```
const gchar* gst_tensor_transform_operator_string[] [static]
```

Initial value:

```
= {  
  [GTT_OP_TYPECAST] = "typecast",  
  [GTT_OP_ADD] = "add",  
  [GTT_OP_MUL] = "mul",  
  [GTT_OP_DIV] = "div",  
  [GTT_OP_UNKNOWN] = NULL  
}
```

Definition at line 124 of file gsttensor_transform.c.

9.68.5.5 gst_tensor_transform_stand_string

```
const gchar* gst_tensor_transform_stand_string[] [static]
```

Initial value:

```
= {  
  [STAND_DEFAULT] = "default",  
  [STAND_DC_AVERAGE] = "dc-average",  
  [STAND_END] = NULL  
}
```

Definition at line 118 of file gsttensor_transform.c.

9.68.5.6 GTT_OP_MUL

```
case GTT_OP_MUL
```

Definition at line 584 of file gsttensor_transform.c.

9.68.5.7 operator%d", oper)

```
default Unknown operator%d", oper)
```

Definition at line 589 of file gsttensor_transform.c.

9.68.5.8 sink_factory

```
GstStaticPadTemplate sink_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the inputs.

Definition at line 135 of file gsttensor_transform.c.

9.68.5.9 src_factory

```
GstStaticPadTemplate src_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the outputs.

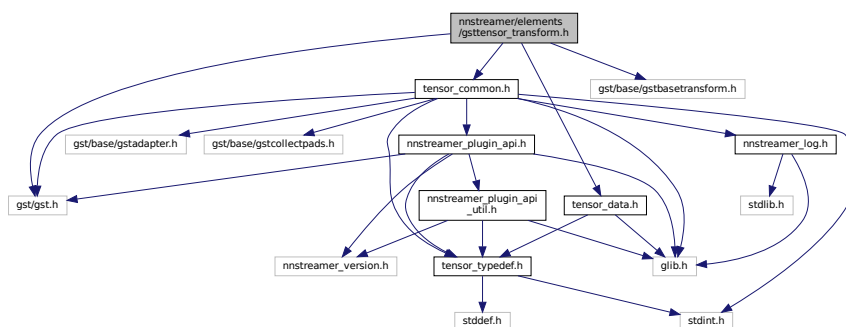
Definition at line 143 of file gsttensor_transform.c.

9.69 nnstreamer/elements/gsttensor_transform.h File Reference

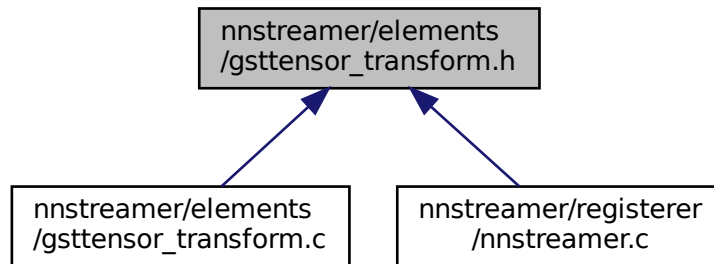
GStreamer plugin to transform tensor dimension or type.

```
#include <gst/gst.h>
#include <gst/base/gstbasetransform.h>
#include <tensor_common.h>
#include <tensor_data.h>
```

Include dependency graph for gsttensor_transform.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_tensor_transform_dimchg](#)
Internal data structure for dimchg mode.
- struct [_tensor_transform_typecast](#)
Internal data structure for typecast mode.
- struct [tensor_transform_operator_s](#)
Internal data structure for operator of arithmetic mode.
- struct [_tensor_transform_arithmetic](#)
Internal data structure for arithmetic mode.
- struct [_tensor_transform_transpose](#)
Internal data structure for transpose mode.
- struct [_tensor_transform_stand](#)
Internal data structure for stand mode.
- struct [_tensor_transform_clamp](#)
Internal data structure for clamp mode.
- struct [_tensor_transform_padding](#)
Internal data structure for padding mode.
- struct [_GstTensorTransform](#)
Internal data structure for tensor_transform instances.
- struct [_GstTensorTransformClass](#)
GstTensorTransformClass inherits GstBaseTransformClass.

Macros

- #define [GST_TYPE_TENSOR_TRANSFORM](#) ([gst_tensor_transform_get_type\(\)](#))
- #define [GST_TENSOR_TRANSFORM](#)(obj) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj),[GST_TYPE_TENSOR_TRANSFORM](#)))
- #define [GST_TENSOR_TRANSFORM_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_CAST](#)((klass),[GST_TYPE_TENSOR_TRANSFORM](#)))
- #define [GST_IS_TENSOR_TRANSFORM](#)(obj) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_TRANSFORM](#)))
- #define [GST_IS_TENSOR_TRANSFORM_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_TRANSFORM](#)))
- #define [GST_TENSOR_TRANSFORM_CAST](#)(obj) (([GstTensorTransform *](#))(obj))

Typedefs

- typedef struct [_GstTensorTransform](#) [GstTensorTransform](#)
- typedef struct [_GstTensorTransformClass](#) [GstTensorTransformClass](#)
- typedef enum [_tensor_transform_mode](#) [tensor_transform_mode](#)
- typedef struct [_tensor_transform_dimchg](#) [tensor_transform_dimchg](#)
Internal data structure for dimchg mode.
- typedef struct [_tensor_transform_typecast](#) [tensor_transform_typecast](#)
Internal data structure for typecast mode.
- typedef struct [_tensor_transform_arithmetic](#) [tensor_transform_arithmetic](#)
Internal data structure for arithmetic mode.
- typedef struct [_tensor_transform_transpose](#) [tensor_transform_transpose](#)
Internal data structure for transpose mode.
- typedef struct [_tensor_transform_stand](#) [tensor_transform_stand](#)
Internal data structure for stand mode.
- typedef struct [_tensor_transform_clamp](#) [tensor_transform_clamp](#)
Internal data structure for clamp mode.
- typedef struct [_tensor_transform_padding](#) [tensor_transform_padding](#)
Internal data structure for padding mode.

Enumerations

- enum [_tensor_transform_mode](#) {
[GTT_DIMCHG](#) = 0, [GTT_TYPECAST](#), [GTT_ARITHMETIC](#), [GTT_TRANSPOSE](#),
[GTT_STAND](#), [GTT_CLAMP](#), [GTT_PADDING](#), [GTT_UNKNOWN](#) = -1 }
- enum [tensor_transform_operator](#) {
[GTT_OP_TYPECAST](#) = 0, [GTT_OP_ADD](#) = 1, [GTT_OP_MUL](#) = 2, [GTT_OP_DIV](#) = 3,
[GTT_OP_UNKNOWN](#) }
- enum [tensor_transform_stand_mode](#) { [STAND_DEFAULT](#) = 0, [STAND_DC_AVERAGE](#) = 1, [STAND_END](#) }
- enum [tensor_transform_padding_axis](#) {
[PADDING_LEFT](#) = 0, [PADDING_RIGHT](#) = 1, [PADDING_TOP](#) = 2, [PADDING_BOTTOM](#) = 3,
[PADDING_FRONT](#) = 4, [PADDING_BACK](#) = 5, [PADDING_END](#) }

Functions

- GType [gst_tensor_transform_get_type](#) (void)
Get Type function required for gst elements.

9.69.1 Detailed Description

GStreamer plugin to transform tensor dimension or type.

GStreamer Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

10 Jul 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs.

9.69.2 Macro Definition Documentation

9.69.2.1 GST_IS_TENSOR_TRANSFORM

```
#define GST_IS_TENSOR_TRANSFORM(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_TRANSFORM))
```

Definition at line 48 of file gsttensor_transform.h.

9.69.2.2 GST_IS_TENSOR_TRANSFORM_CLASS

```
#define GST_IS_TENSOR_TRANSFORM_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_TRANSFORM))
```

Definition at line 50 of file gsttensor_transform.h.

9.69.2.3 GST_TENSOR_TRANSFORM

```
#define GST_TENSOR_TRANSFORM(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_TRANSFORM, GstTensorTransform))
```

Definition at line 44 of file gsttensor_transform.h.

9.69.2.4 GST_TENSOR_TRANSFORM_CAST

```
#define GST_TENSOR_TRANSFORM_CAST(  
    obj ) ((GstTensorTransform *) (obj))
```

Definition at line 52 of file gsttensor_transform.h.

9.69.2.5 GST_TENSOR_TRANSFORM_CLASS

```
#define GST_TENSOR_TRANSFORM_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_TRANSFORM, GstTensorTransformClass))
```

Definition at line 46 of file gsttensor_transform.h.

9.69.2.6 GST_TYPE_TENSOR_TRANSFORM

```
#define GST_TYPE_TENSOR_TRANSFORM (gst_tensor_transform_get_type())
```

Definition at line 42 of file gsttensor_transform.h.

9.69.3 Typedef Documentation

9.69.3.1 GstTensorTransform

```
typedef struct _GstTensorTransform GstTensorTransform
```

Definition at line 54 of file gsttensor_transform.h.

9.69.3.2 GstTensorTransformClass

```
typedef struct _GstTensorTransformClass GstTensorTransformClass
```

Definition at line 55 of file gsttensor_transform.h.

9.69.3.3 `tensor_transform_arithmetic`

```
typedef struct _tensor_transform_arithmetic tensor_transform_arithmetic
```

Internal data structure for arithmetic mode.

9.69.3.4 `tensor_transform_clamp`

```
typedef struct _tensor_transform_clamp tensor_transform_clamp
```

Internal data structure for clamp mode.

9.69.3.5 `tensor_transform_dimchg`

```
typedef struct _tensor_transform_dimchg tensor_transform_dimchg
```

Internal data structure for dimchg mode.

9.69.3.6 `tensor_transform_mode`

```
typedef enum _tensor_transform_mode tensor_transform_mode
```

9.69.3.7 `tensor_transform_padding`

```
typedef struct _tensor_transform_padding tensor_transform_padding
```

Internal data structure for padding mode.

9.69.3.8 `tensor_transform_stand`

```
typedef struct _tensor_transform_stand tensor_transform_stand
```

Internal data structure for stand mode.

9.69.3.9 tensor_transform_transpose

```
typedef struct _tensor_transform_transpose tensor_transform_transpose
```

Internal data structure for transpose mode.

9.69.3.10 tensor_transform_typecast

```
typedef struct _tensor_transform_typecast tensor_transform_typecast
```

Internal data structure for typecast mode.

9.69.4 Enumeration Type Documentation

9.69.4.1 _tensor_transform_mode

```
enum _tensor_transform_mode
```

Enumerator

GTT_DIMCHG	
GTT_TYPECAST	
GTT_ARITHMETIC	
GTT_TRANSPOSE	
GTT_STAND	
GTT_CLAMP	
GTT_PADDING	
GTT_UNKNOWN	

Definition at line 57 of file gsttensor_transform.h.

9.69.4.2 tensor_transform_operator

```
enum tensor_transform_operator
```

Enumerator

GTT_OP_TYPECAST	
GTT_OP_ADD	
GTT_OP_MUL	
GTT_OP_DIV	
GTT_OP_UNKNOWN	

Definition at line 70 of file gsttensor_transform.h.

9.69.4.3 tensor_transform_padding_axis

enum `tensor_transform_padding_axis`

Enumerator

PADDING_LEFT	
PADDING_RIGHT	
PADDING_TOP	
PADDING_BOTTOM	
PADDING_FRONT	
PADDING_BACK	
PADDING_END	

Definition at line 87 of file gsttensor_transform.h.

9.69.4.4 tensor_transform_stand_mode

enum `tensor_transform_stand_mode`

Enumerator

STAND_DEFAULT	
STAND_DC_AVERAGE	
STAND_END	

Definition at line 80 of file gsttensor_transform.h.

9.69.5 Function Documentation

9.69.5.1 gst_tensor_transform_get_type()

```
GType gst_tensor_transform_get_type (
    void )
```

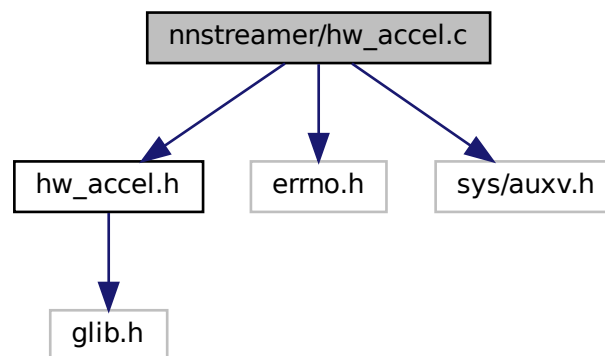
Get Type function required for gst elements.

9.70 nnstreamer/elements/ignore_warning.sh File Reference

9.71 nnstreamer/hw_accel.c File Reference

Common hardware acceleration availability checks.

```
#include <hw_accel.h>
#include <errno.h>
#include <sys/auxv.h>
Include dependency graph for hw_accel.c:
```



Functions

- `gint cpu_neon_accel_available` (void)
Check if neon is supported.

9.71.1 Detailed Description

Common hardware acceleration availability checks.

NNStreamer hardware accelerator availability checking Copyright (C) 2020 Parichay Kapoor pk.kapoor@samsung.com

Date

8 September 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

Bug No known bugs except for NYI items

9.71.2 Function Documentation

9.71.2.1 `cpu_neon_accel_available()`

```
gint cpu_neon_accel_available (
    void )
```

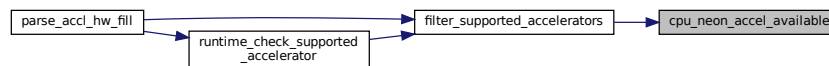
Check if neon is supported.

Return values

0	if supported, else -errno
---	---------------------------

Definition at line 41 of file `hw_accel.c`.

Here is the caller graph for this function:

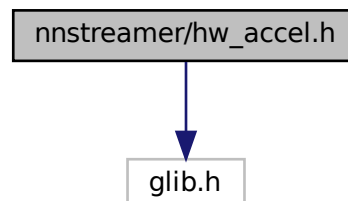


9.72 `nnstreamer/hw_accel.h` File Reference

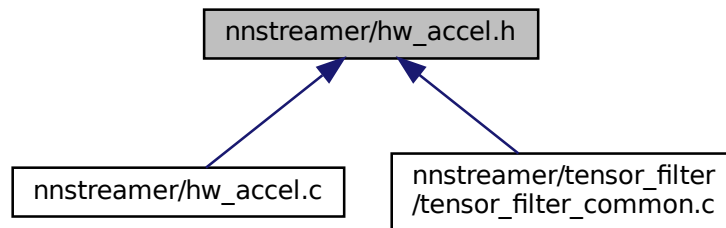
Common hardware acceleration availability header.

```
#include <glib.h>
```

Include dependency graph for `hw_accel.h`:



This graph shows which files directly or indirectly include this file:



Functions

- gint [cpu_neon_accel_available](#) (void)
Check if neon is supported.

9.72.1 Detailed Description

Common hardware acceleration availability header.

NNStreamer hardware accelerator availability checking Copyright (C) 2020 Parichay Kapoor pk.kapoor@samsung.com

Date

8 September 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

Bug No known bugs except for NYI items

9.72.2 Function Documentation

9.72.2.1 [cpu_neon_accel_available\(\)](#)

```
gint cpu_neon_accel_available (  
    void )
```

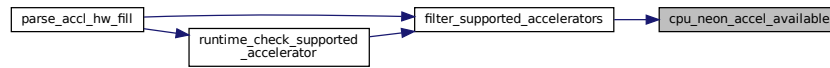
Check if neon is supported.

Return values

0	if supported, else -errno
---	---------------------------

Definition at line 41 of file hw_accel.c.

Here is the caller graph for this function:



9.73 nnstreamer/include/nnstreamer_plugin_api.h File Reference

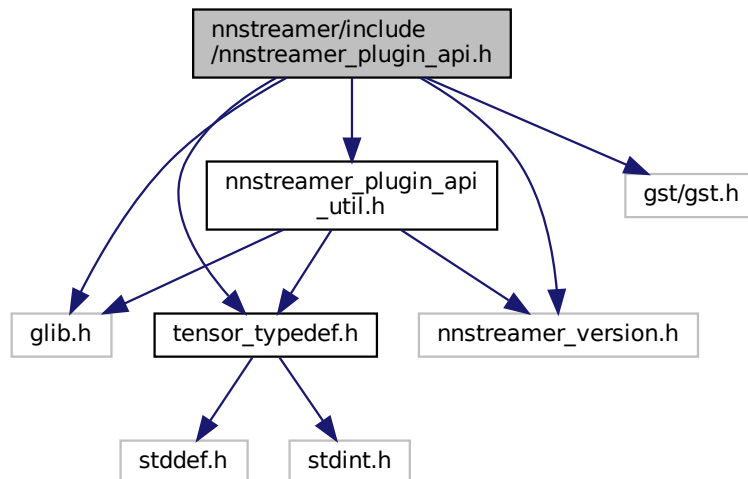
Optional/Additional NNStreamer APIs for sub-plugin writers. (Need Gst devel)

```

#include <glib.h>
#include <gst/gst.h>
#include <tensor_typedef.h>
#include <nnstreamer_version.h>
#include <nnstreamer_plugin_api_util.h>

```

Include dependency graph for nnstreamer_plugin_api.h:



This graph shows which files directly or indirectly include this file:



Functions

- G_BEGIN_DECLS gboolean [gst_structure_is_tensor_stream](#) (const GstStructure *structure)
Check given mimetype is tensor stream.
- [media_type gst_structure_get_media_type](#) (const GstStructure *structure)
Get media type from structure.
- gboolean [gst_tensors_config_from_structure](#) (GstTensorsConfig *config, const GstStructure *structure)
Parse structure and set tensors config (for other/tensors)
- gboolean [gst_tensors_config_from_peer](#) (GstPad *pad, GstTensorsConfig *config, gboolean *is_fixed)
Parse caps from peer pad and set tensors config.
- GstCaps * [gst_tensor_caps_from_config](#) (const GstTensorsConfig *config)
Get tensor caps from tensors config (for other/tensor)
- GstCaps * [gst_tensors_caps_from_config](#) (const GstTensorsConfig *config)
Get caps from tensors config (for other/tensors)
- void [gst_tensor_alloc_init](#) (gsize alignment)
set alignment that default allocator would align to
- gboolean [gst_tensor_meta_info_parse_memory](#) (GstTensorMetaInfo *meta, GstMemory *mem)
Parse memory and fill the tensor meta.
- GstMemory * [gst_tensor_meta_info_append_header](#) (GstTensorMetaInfo *meta, GstMemory *mem)
Append header to memory.
- void [gst_tensor_caps_update_dimension](#) (GstCaps *caps, GstCaps *filter)
Update caps dimension for negotiation.
- gboolean [gst_tensor_caps_can_intersect](#) (GstCaps *caps1, GstCaps *caps2)
Try intersecting @caps1 and @caps2 for tensor stream.
- GstMemory * [gst_tensor_buffer_get_nth_memory](#) (GstBuffer *buffer, const guint index)
Get the nth GstMemory from given buffer.
- gboolean [gst_tensor_buffer_append_memory](#) (GstBuffer *buffer, GstMemory *memory, const GstTensorInfo *info)
Append memory to given buffer.
- guint [gst_tensor_buffer_get_count](#) (GstBuffer *buffer)
Get the number of tensors in the buffer.

9.73.1 Detailed Description

Optional/Additional NNStreamer APIs for sub-plugin writers. (Need Gst devel)

NNStreamer Common API Header for Plug-Ins Copyright (C) 2019 MyungJoo Ham myungjoo.ham@samsung.com Copyright (C) 2019 Wook Song wook16.song@samsung.com

Date

24 Jan 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com and Wook Song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.73.2 Function Documentation

9.73.2.1 `gst_structure_get_media_type()`

```
media_type gst_structure_get_media_type (
    const GstStructure * structure )
```

Get media type from structure.

Parameters

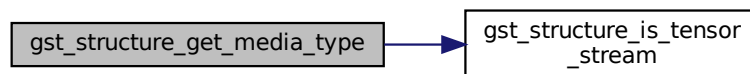
<i>structure</i>	structure to be interpreted
------------------	-----------------------------

Returns

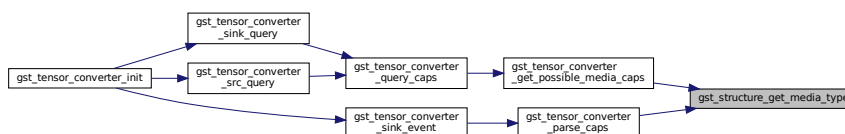
corresponding media type (returns `_NNS_MEDIA_INVALID` for unsupported type)

Definition at line 1001 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

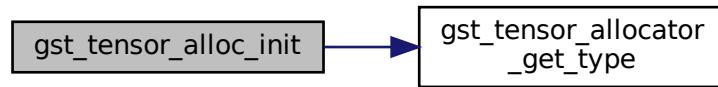


9.73.2.2 `gst_structure_is_tensor_stream()`

```
G_BEGIN_DECLS gboolean gst_structure_is_tensor_stream (
    const GstStructure * structure )
```

Check given mimetype is tensor stream.

Here is the call graph for this function:



9.73.2.4 gst_tensor_buffer_append_memory()

```

gboolean gst_tensor_buffer_append_memory (
    GstBuffer * buffer,
    GstMemory * memory,
    const GstTensorInfo * info )
  
```

Append *memory* to given *buffer*.

Parameters

	<i>[in/out]</i>	buffer GstBuffer to be appended.
in	<i>memory</i>	GstMemory to append. This function takes ownership of this, even if it returns failure.
in	<i>info</i>	GstTensorInfo of given <i>memory</i> .

Returns

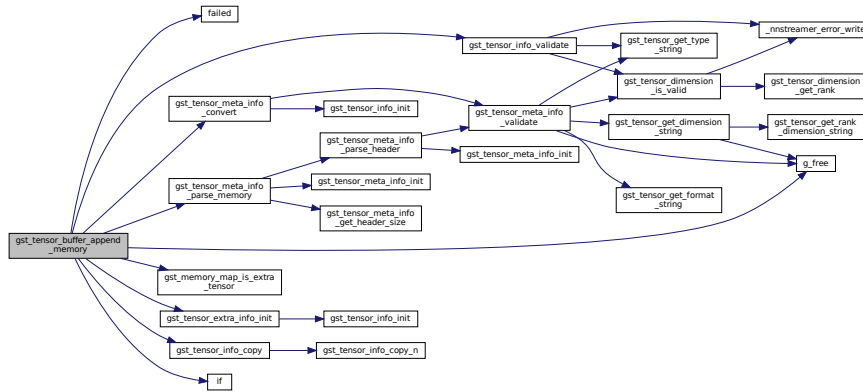
TRUE if successfully appended, otherwise FALSE.

Free the name string, cause it does not freed by gstreamer.

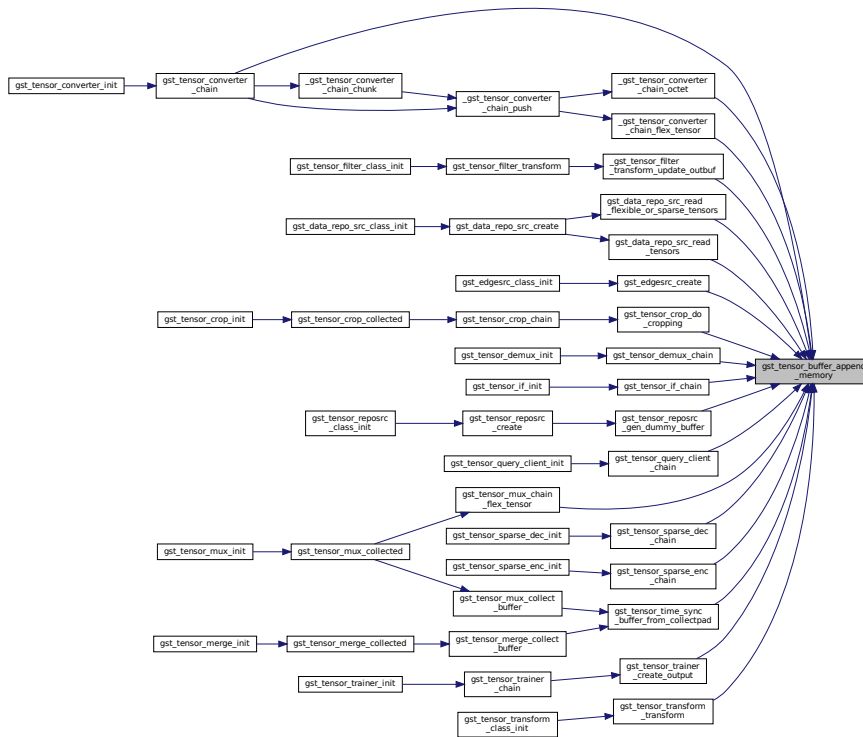
Todo Make custom `gst_allocator` later?

Definition at line 1666 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



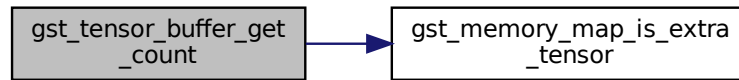
9.73.2.5 `gst_tensor_buffer_get_count()`

```
guint gst_tensor_buffer_get_count (
    GstBuffer * buffer )
```

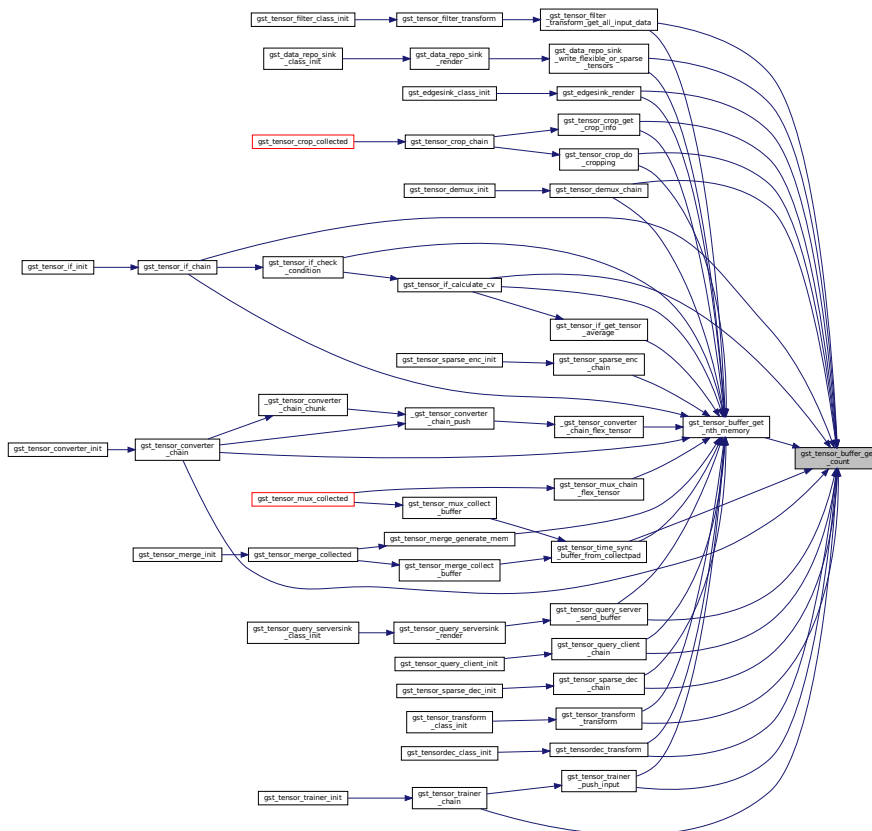
Get the number of tensors in the buffer.

Definition at line 1813 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.6 gst_tensor_buffer_get_nth_memory()

```

GstMemory* gst_tensor_buffer_get_nth_memory (
    GstBuffer * buffer,
    const guint index )
  
```

Get the nth GstMemory from given *buffer*.

Parameters

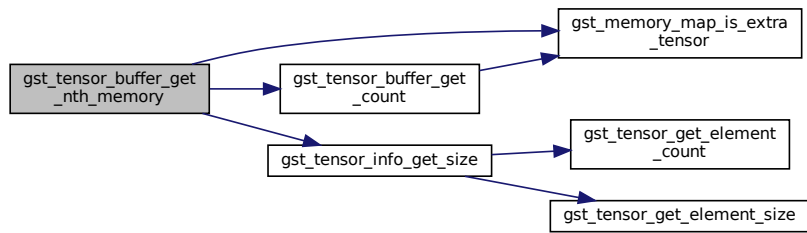
in	<i>buffer</i>	GstBuffer to be parsed.
in	<i>index</i>	Index of GstMemory to be returned.

Returns

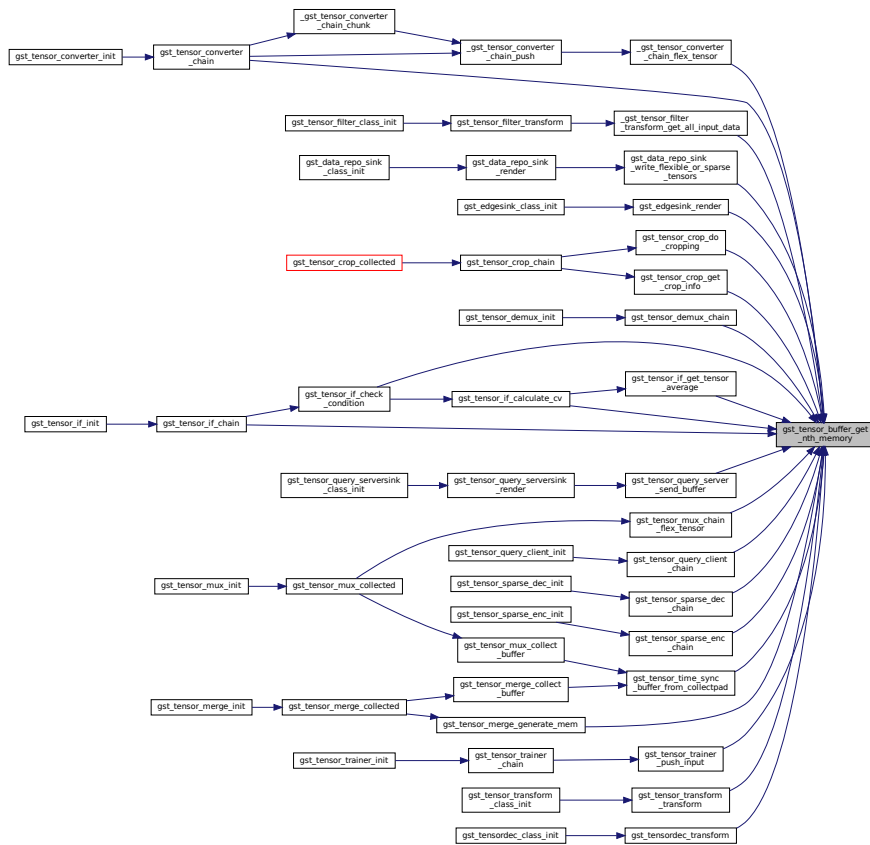
GstMemory if found, otherwise NULL (Caller should free returned memory using `gst_memory_unref()`).

Definition at line 1586 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.7 `gst_tensor_caps_can_intersect()`

```
gboolean gst_tensor_caps_can_intersect (
    GstCaps * caps1,
    GstCaps * caps2 )
```

Try intersecting @caps1 and @caps2 for tensor stream.

Parameters

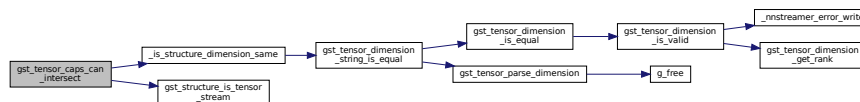
<i>caps1</i>	a GstCaps to intersect
<i>caps2</i>	a GstCaps to intersect

Returns

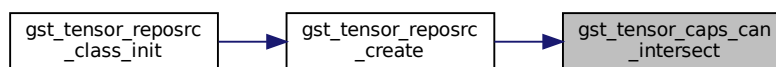
TRUE if intersection would be not empty.

Definition at line 1142 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.8 `gst_tensor_caps_from_config()`

```
GstCaps* gst_tensor_caps_from_config (
    const GstTensorsConfig * config )
```

Get tensor caps from tensors config (for other/tensor)

Parameters

<i>config</i>	tensors config info
---------------	---------------------

Returns

caps for given config

Get tensor caps from tensors config (for other/tensor)

Parameters

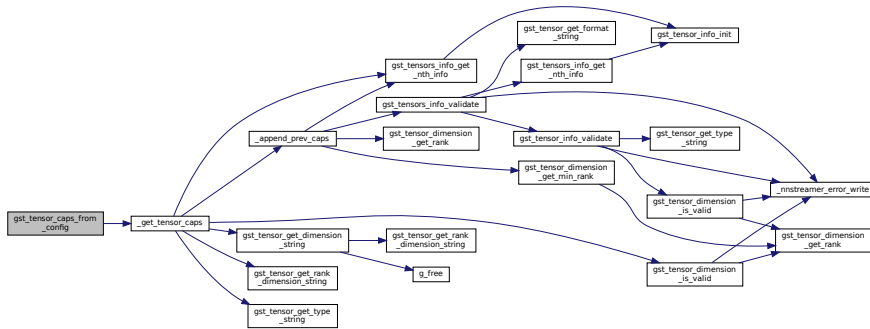
<i>config</i>	tensors config info
---------------	---------------------

Returns

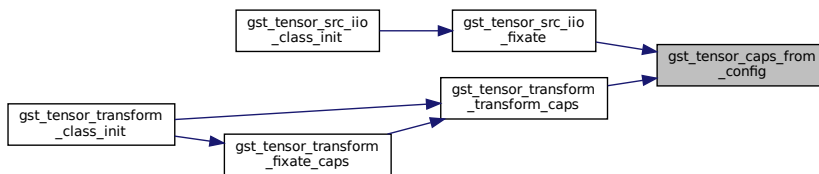
caps for given config

Definition at line 1395 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.9 `gst_tensor_caps_update_dimension()`

```
void gst_tensor_caps_update_dimension (
    GstCaps * caps,
    GstCaps * filter )
```

Update caps dimension for negotiation.

Parameters

<i>caps</i>	caps to compare and update
<i>filter</i>	caps to compare

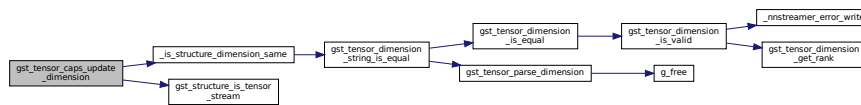
Update caps dimension for negotiation.

Parameters

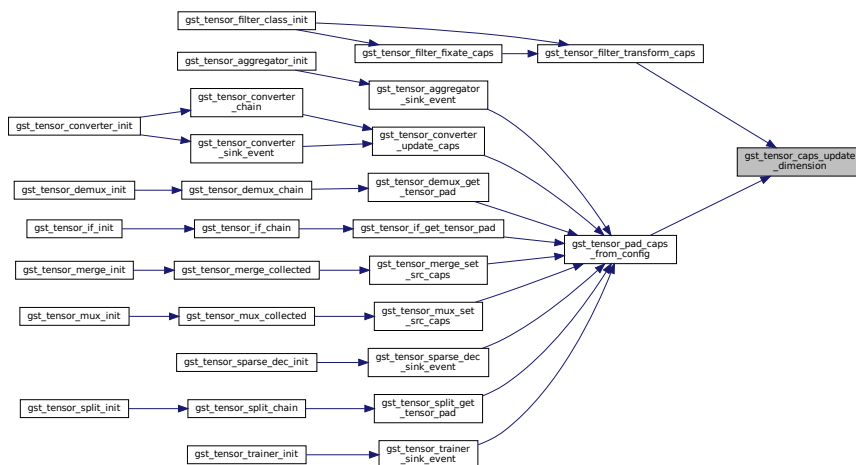
<i>caps</i>	caps to compare and update
<i>filter</i>	caps to compare

Definition at line 1093 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.10 `gst_tensor_meta_info_append_header()`

```
GstMemory* gst_tensor_meta_info_append_header (
    GstTensorMetaInfo * meta,
    GstMemory * mem )
```

Append header to memory.

Parameters

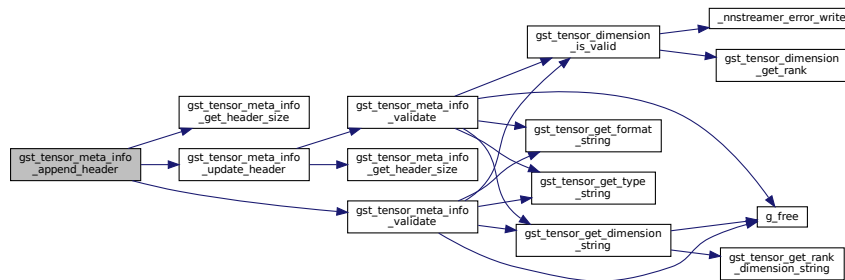
in	<i>meta</i>	tensor meta structure
in	<i>mem</i>	pointer to GstMemory

Returns

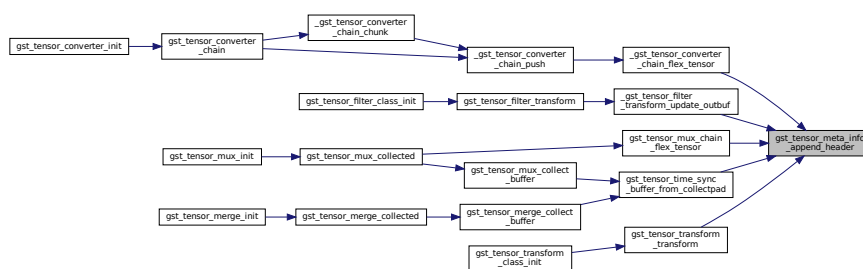
Newly allocated GstMemory (Caller should free returned memory using `gst_memory_unref()`)

Definition at line 1544 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.11 `gst_tensor_meta_info_parse_memory()`

```
gboolean gst_tensor_meta_info_parse_memory (
    GstTensorMetaInfo * meta,
    GstMemory * mem )
```

Parse memory and fill the tensor meta.

Parameters

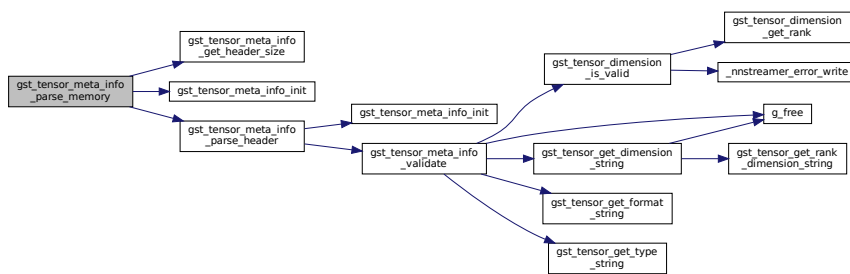
out	meta	tensor meta structure to be filled
in	mem	pointer to GstMemory to be parsed

Returns

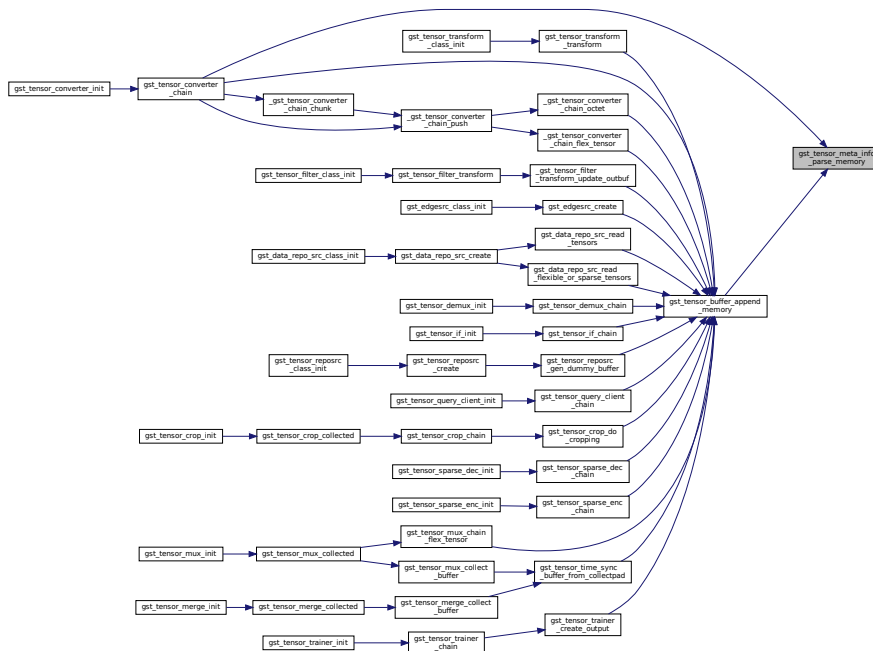
TRUE if successfully set the meta

Definition at line 1509 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.13 `gst_tensors_config_from_peer()`

```
gboolean gst_tensors_config_from_peer (  
    GstPad * pad,  
    GstTensorsConfig * config,  
    gboolean * is_fixed )
```

Parse caps from peer pad and set tensors config.

Parameters

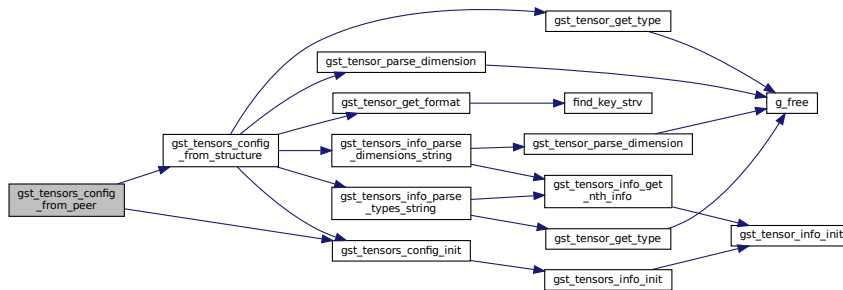
<i>pad</i>	GstPad to get the capabilities
<i>config</i>	tensors config structure to be filled
<i>is_fixed</i>	flag to be updated when peer caps is fixed (not mandatory, do nothing when the param is null)

Returns

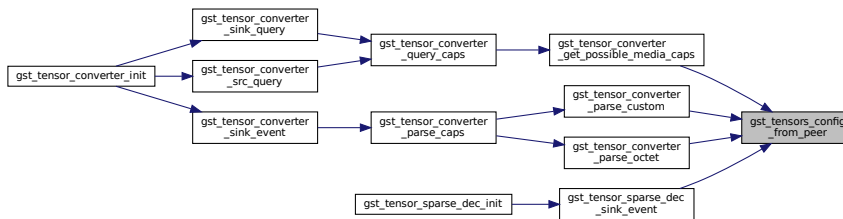
TRUE if successfully configured from peer

Definition at line 1041 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.73.2.14 `gst_tensors_config_from_structure()`

```
gboolean gst_tensors_config_from_structure (
    GstTensorsConfig * config,
    const GstStructure * structure )
```

Parse structure and set tensors config (for other/tensors)

Parameters

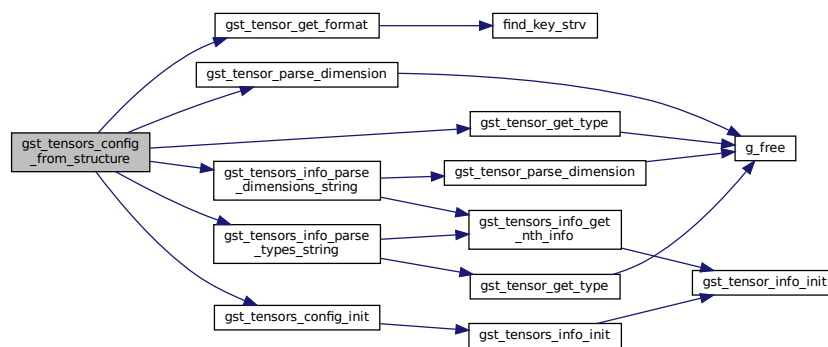
<i>config</i>	tensors config structure to be filled
<i>structure</i>	structure to be interpreted

Returns

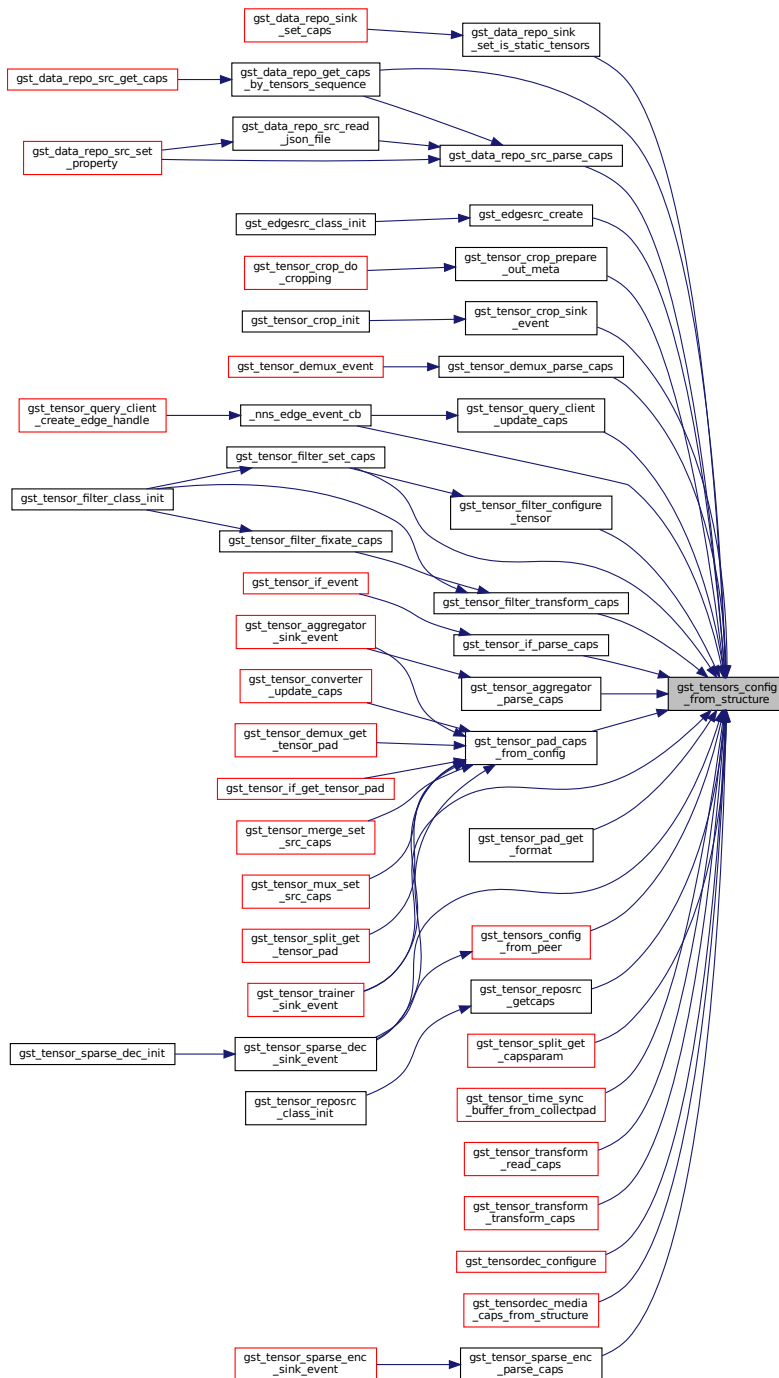
TRUE if no error

Definition at line 1413 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



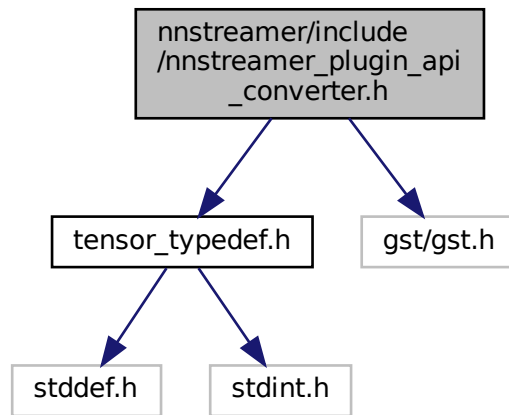
Here is the caller graph for this function:



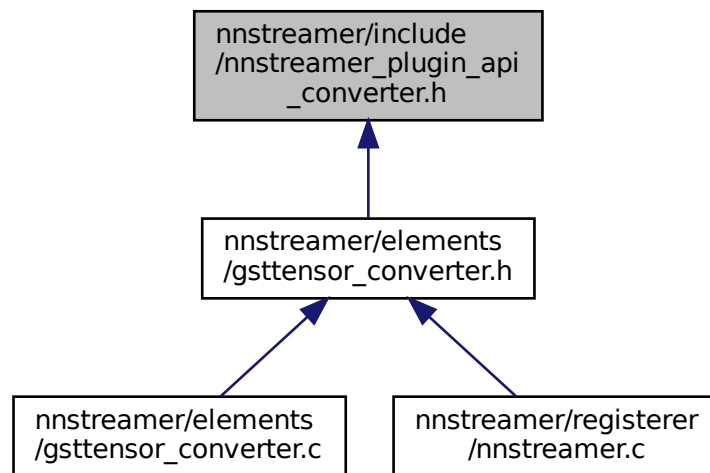
9.74 nnstreamer/include/nnstreamer_plugin_api_converter.h File Reference

Mandatory APIs for NNStreamer Converter sub-plugins (Need Gst Devel)

```
#include "tensor_typedef.h"
#include <gst/gst.h>
Include dependency graph for nnstreamer_plugin_api_converter.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_NNStreamerExternalConverter`](#)
Converter's subplugin implementation.

Typedefs

- typedef struct [_NNStreamerExternalConverter](#) [NNStreamerExternalConverter](#)
Converter's subplugin implementation.

Functions

- const [NNStreamerExternalConverter](#) * [nnstreamer_converter_find](#) (const char *name)
Find converter sub-plugin with the name.
- int [registerExternalConverter](#) ([NNStreamerExternalConverter](#) *ex)
Converter's sub-plugin should call this function to register itself.
- void [unregisterExternalConverter](#) (const char *prefix)
Converter's sub-plugin may call this to unregister itself.
- void [nnstreamer_converter_set_custom_property_desc](#) (const char *name, const char *prop,...)
set custom property description for tensor converter sub-plugin

9.74.1 Detailed Description

Mandatory APIs for NNStreamer Converter sub-plugins (Need Gst Devel)

NNStreamer API for Tensor_Converter Sub-Plugins Copyright (C) 2019 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

09 Dec 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.74.2 Typedef Documentation

9.74.2.1 NNStreamerExternalConverter

```
typedef struct _NNStreamerExternalConverter NNStreamerExternalConverter
```

Converter's subplugin implementation.

9.74.3 Function Documentation

9.74.3.1 nnstreamer_converter_find()

```
const NNStreamerExternalConverter* nnstreamer_converter_find (
    const char * name )
```

Find converter sub-plugin with the name.

Parameters

in	<i>name</i>	The name of converter sub-plugin.
----	-------------	-----------------------------------

Returns

NNStreamerExternalConverter if subplugin is found. NULL if not found or the sub-plugin object has an error.

Parameters

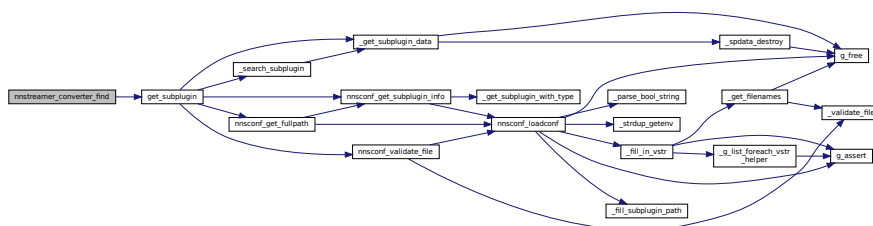
in	<i>name</i>	The name of converter sub-plugin.
----	-------------	-----------------------------------

Returns

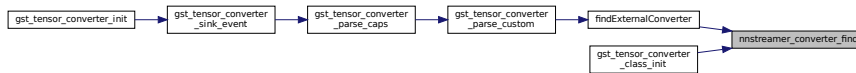
NULL if not found or the sub-plugin object has an error.

Definition at line 2360 of file gsttensor_converter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.74.3.2 nntstreamer_converter_set_custom_property_desc()

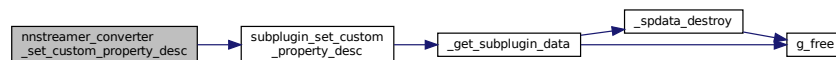
```

void nntstreamer_converter_set_custom_property_desc (
    const char * name,
    const char * prop,
    ... )
  
```

set custom property description for tensor converter sub-plugin

Definition at line 2457 of file gsttensor_converter.c.

Here is the call graph for this function:



9.74.3.3 registerExternalConverter()

```

int registerExternalConverter (
    NNStreamerExternalConverter * ex )
  
```

Converter's sub-plugin should call this function to register itself.

Parameters

in	ex	Converter sub-plugin to be registered.
----	----	--

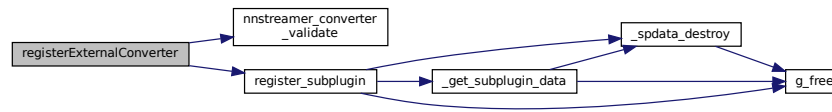
Returns

TRUE if registered. FALSE is failed or duplicated.

Converter's sub-plugin should call this function to register itself.

Definition at line 2389 of file gsttensor_converter.c.

Here is the call graph for this function:



9.74.3.4 unregisterExternalConverter()

```
void unregisterExternalConverter (
    const char * name )
```

Converter's sub-plugin may call this to unregister itself.

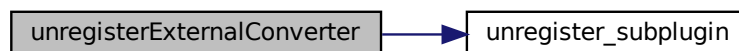
Parameters

in	<i>prefix</i>	The name of converter sub-plugin.
----	---------------	-----------------------------------

Converter's sub-plugin may call this to unregister itself.

Definition at line 2399 of file gsttensor_converter.c.

Here is the call graph for this function:

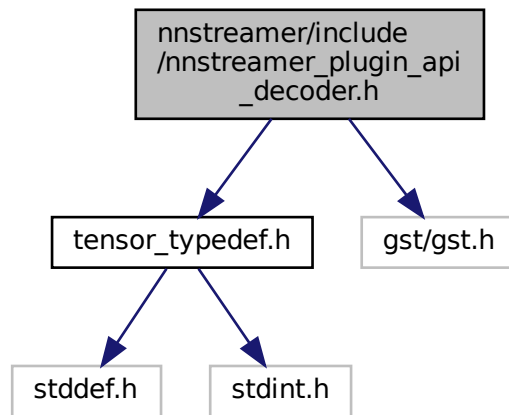


9.75 nnstreamer/include/nnstreamer_plugin_api_decoder.h File Reference

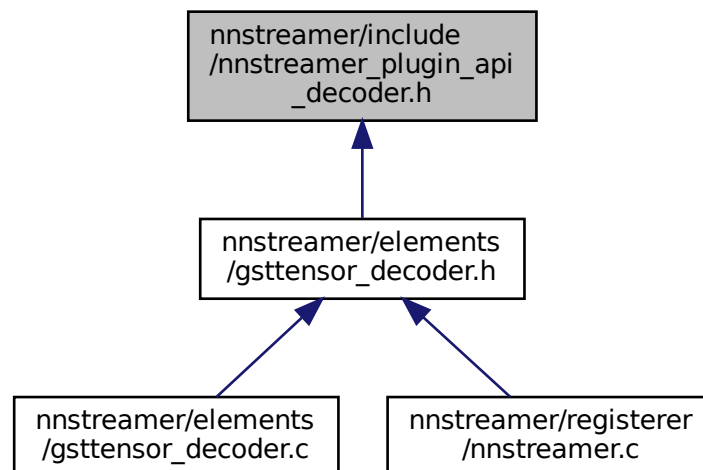
Mandatory APIs for NNStreamer Decoder sub-plugins (Need Gst Devel)

```
#include "tensor_typedef.h"
#include <gst/gst.h>
```

Include dependency graph for nnstreamer_plugin_api_decoder.h:



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_GstTensorDecoderDef`](#)

Decoder definitions for different semantics of tensors This allows developers to create their own decoders.

Typedefs

- typedef struct [_GstTensorDecoderDef](#) [GstTensorDecoderDef](#)
Decoder definitions for different semantics of tensors This allows developers to create their own decoders.

Functions

- int [nnstreamer_decoder_probe](#) ([GstTensorDecoderDef](#) *decoder)
Decoder's sub-plugin should call this function to register itself.
- void [nnstreamer_decoder_exit](#) (const char *name)
Decoder's sub-plugin may call this to unregister itself.
- const [GstTensorDecoderDef](#) * [nnstreamer_decoder_find](#) (const char *name)
Find decoder sub-plugin with the name.
- void [nnstreamer_decoder_set_custom_property_desc](#) (const char *name, const char *prop,...)
set custom property description for tensor decoder sub-plugin

9.75.1 Detailed Description

Mandatory APIs for NNStreamer Decoder sub-plugins (Need Gst Devel)

NNStreamer API for Tensor_Decoder Sub-Plugins Copyright (C) 2019 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

30 Jan 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.75.2 Typedef Documentation

9.75.2.1 GstTensorDecoderDef

```
typedef struct _GstTensorDecoderDef GstTensorDecoderDef
```

Decoder definitions for different semantics of tensors This allows developers to create their own decoders.

9.75.3 Function Documentation

9.75.3.1 nnstreamer_decoder_exit()

```
void nnstreamer_decoder_exit (  
    const char * name )
```

Decoder's sub-plugin may call this to unregister itself.

Parameters

in	<i>name</i>	The name of decoder sub-plugin.
----	-------------	---------------------------------

Definition at line 166 of file gsttensor_decoder.c.

Here is the call graph for this function:



9.75.3.2 nnstreamer_decoder_find()

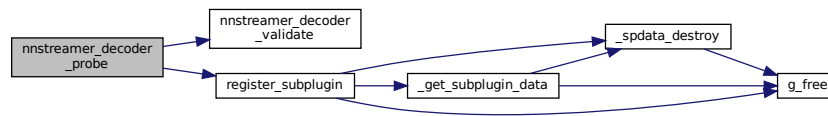
```
const GstTensorDecoderDef* nnstreamer_decoder_find (  
    const char * name )
```

Find decoder sub-plugin with the name.

Parameters

in	<i>name</i>	The name of decoder sub-plugin.
----	-------------	---------------------------------

Here is the call graph for this function:



9.75.3.4 nnstreamer_decoder_set_custom_property_desc()

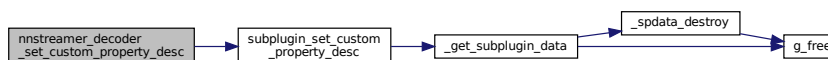
```

void nnstreamer_decoder_set_custom_property_desc (
    const char * name,
    const char * prop,
    ... )
  
```

set custom property description for tensor decoder sub-plugin

Definition at line 186 of file gsttensor_decoder.c.

Here is the call graph for this function:

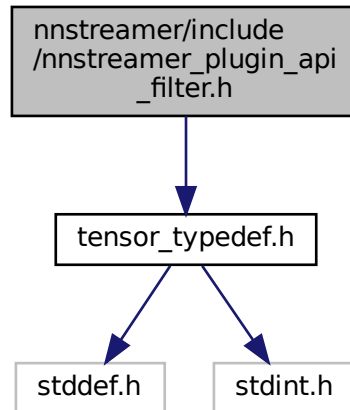


9.76 nnstreamer/include/nnstreamer_plugin_api_filter.h File Reference

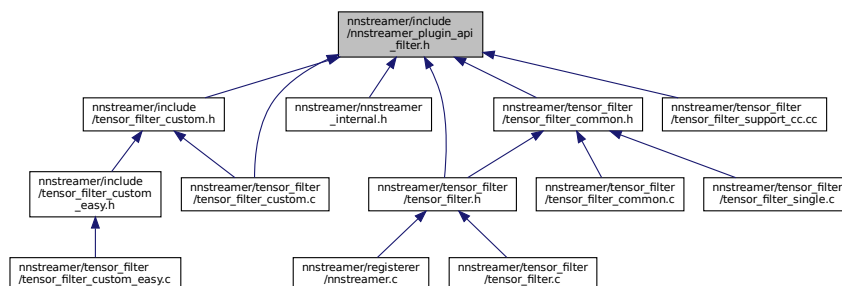
Mandatory APIs for NNStreamer Filter sub-plugins (No External Dependencies)

```
#include "tensor_typedef.h"
```

Include dependency graph for `nnstreamer_plugin_api_filter.h`:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorFilterProperties](#)
GstTensorFilter's properties for NN framework (internal data structure)
- struct [_GstTensorFilterFrameworkStatistics](#)
Structure definition for tensor-filter framework statistics.
- struct [_GstTensorFilterFrameworkInfo](#)
Tensor_Filter Subplugin framework related information.
- struct [_GstTensorFilterFrameworkEventData](#)
User data for the tensor_tilter subplugin related events.
- struct [_GstTensorFilterFramework](#)
Tensor_Filter Subplugin definition.
- struct [parse_accl_args](#)
Accelerator related arguments for parsing.

Macros

- #define [ACCL_NONE_STR](#) "none"
- #define [ACCL_DEFAULT_STR](#) "default"
- #define [ACCL_AUTO_STR](#) "auto"
- #define [ACCL_CPU_STR](#) "cpu"
- #define [ACCL_CPU_SIMD_STR](#) "cpu.simd"
- #define [ACCL_CPU_NEON_STR](#) "cpu.neon"
- #define [ACCL_GPU_STR](#) "gpu"
- #define [ACCL_NPU_STR](#) "npu"
- #define [ACCL_NPU_MOVIDIUS_STR](#) "npu.movidius"
- #define [ACCL_NPU_EDGE_TPU_STR](#) "npu.edgetpu"
- #define [ACCL_NPU_VIVANTE_STR](#) "npu.vivante"
- #define [ACCL_NPU_SRCN_STR](#) "npu.srcn" /** srcn hardware supported by nnfw */
- #define [ACCL_NPU_SLSI_STR](#) "npu.slsi"
- #define [ACCL_NPU_SR_STR](#) "npu.sr"
- #define [GST_TENSOR_FILTER_FRAMEWORK_BASE](#) (0xDEAFDEAD00000000ULL)
- #define [GST_TENSOR_FILTER_FRAMEWORK_V0](#) ([GST_TENSOR_FILTER_FRAMEWORK_BASE](#))
- #define [GST_TENSOR_FILTER_FRAMEWORK_V1](#) ([GST_TENSOR_FILTER_FRAMEWORK_BASE](#) | 0x10000ULL)
- #define [GST_TENSOR_FILTER_API_VERSION_DEFINED](#) (1)
- #define [GST_TENSOR_FILTER_API_VERSION_MIN](#) (0) /* The minimum API version supported (could be obsolete) */
- #define [GST_TENSOR_FILTER_API_VERSION_MAX](#) (1) /* The maximum API version supported (recommended) */
- #define [checkGstTensorFilterFrameworkVersion](#)(value, version) (([GST_TENSOR_FILTER_FRAMEWORK_BASE](#) | ((version) << 16)) == (value & 0xFFFFFFFFFFFFFFFF0000ULL))
Check the value of the version field of GstTensorFilterFramework.
- #define [parse_accl_hw](#)(...) [parse_accl_hw_fill](#)(([parse_accl_args](#)){__VA_ARGS__})
workaround to provide default arguments

Typedefs

- typedef [tensor_layout](#) [tensors_layout](#)[[NNS_TENSOR_SIZE_LIMIT](#)]
- typedef struct [_GstTensorFilterProperties](#) [GstTensorFilterProperties](#)
GstTensorFilter's properties for NN framework (internal data structure)
- typedef struct [_GstTensorFilterFrameworkStatistics](#) [GstTensorFilterFrameworkStatistics](#)
Structure definition for tensor-filter framework statistics.
- typedef struct [_GstTensorFilterFrameworkInfo](#) [GstTensorFilterFrameworkInfo](#)
Tensor_Filter Subplugin framework related information.
- typedef struct [_GstTensorFilterFrameworkEventData](#) [GstTensorFilterFrameworkEventData](#)
User data for the tensor_tilter subplugin related events.
- typedef struct [_GstTensorFilterFramework](#) [GstTensorFilterFramework](#)

Enumerations

- enum `accl_hw` {
`ACCL_NONE = 0`, `ACCL_AUTO = 0x1`, `ACCL_DEFAULT = 0x2`, `ACCL_CPU = 0x1000`,
`ACCL_CPU_SIMD = 0x1100`, `ACCL_CPU_NEON = 0x1100`, `ACCL_GPU = 0x2000`, `ACCL_NPU = 0x4000`,
`ACCL_NPU_MOVIDIUS = 0x4001`, `ACCL_NPU_EDGE_TPU = 0x4002`, `ACCL_NPU_VIVANTE = 0x4003`,
`ACCL_NPU_SRCN = 0x4004`,
`ACCL_NPU_SLSI = 0x4005`, `ACCL_NPU_SR = 0x4100` }
acceleration hw properties.
- enum `event_ops` {
`DESTROY_NOTIFY`, `RELOAD_MODEL`, `CUSTOM_PROP`, `SET_INPUT_PROP`,
`SET_OUTPUT_PROP`, `SET_ACCELERATOR`, `CHECK_HW_AVAILABILITY` }
Tensor_Filter Subplugin related events.
- enum `model_info_ops` { `GET_IN_OUT_INFO`, `SET_INPUT_INFO` }
Tensor_Filter Subplugin's model related info gathering operations.

Functions

- int `nnstreamer_filter_probe` (`GstTensorFilterFramework *tfsp`)
Filter's sub-plugin should call this function to register itself.
- void `nnstreamer_filter_exit` (const char *name)
Filter's sub-plugin may call this to unregister itself.
- const `GstTensorFilterFramework * nnstreamer_filter_find` (const char *name)
Find filter sub-plugin with the name.
- void `nnstreamer_filter_set_custom_property_desc` (const char *name, const char *prop,...)
set custom property description for tensor filter sub-plugin
- `accl_hw get_accl_hw_type` (const char *str)
return accl_hw type from string
- const char * `get_accl_hw_str` (const `accl_hw` key)
return string based on accl_hw type
- `accl_hw parse_accl_hw_fill` (`parse_accl_args` `accl_args`)
parse user given string to extract accelerator based on given regex filling in optional arguments
- void * `nnstreamer_filter_shared_model_get` (void *instance, const char *key)
Get the shared model representation that is already shared and has the same key.
- void * `nnstreamer_filter_shared_model_insert_and_get` (void *instance, char *key, void *interpreter)
Insert the new shared model representation and get the value.
- int `nnstreamer_filter_shared_model_remove` (void *instance, const char *key, void(*free_callback)(void *))
Remove the instance registered at the referred list of shared model table. If referred list is empty, free_callback is executed.
- void `nnstreamer_filter_shared_model_replace` (void *instance, const char *key, void *new_interpreter, void(*replace_callback)(void *, void *), void(*free_callback)(void *))
Helper to reload interpreter for instances that has shared key. replace_callback is called iterating instances in referred list.

9.76.1 Detailed Description

Mandatory APIs for NNStreamer Filter sub-plugins (No External Dependencies)

NNStreamer API for Tensor_Filter Sub-Plugins Copyright (C) 2019 MyungJoo Ham myungjoo.ham@samsung.com

Date

30 Jan 2019

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.76.2 Macro Definition Documentation

9.76.2.1 ACCL_AUTO_STR

```
#define ACCL_AUTO_STR "auto"
```

Definition at line 22 of file nnstreamer_plugin_api_filter.h.

9.76.2.2 ACCL_CPU_NEON_STR

```
#define ACCL_CPU_NEON_STR "cpu.neon"
```

Definition at line 25 of file nnstreamer_plugin_api_filter.h.

9.76.2.3 ACCL_CPU_SIMD_STR

```
#define ACCL_CPU_SIMD_STR "cpu.simd"
```

Definition at line 24 of file nnstreamer_plugin_api_filter.h.

9.76.2.4 ACCL_CPU_STR

```
#define ACCL_CPU_STR "cpu"
```

Definition at line 23 of file nnstreamer_plugin_api_filter.h.

9.76.2.5 ACCL_DEFAULT_STR

```
#define ACCL_DEFAULT_STR "default"
```

Definition at line 21 of file nnstreamer_plugin_api_filter.h.

9.76.2.6 ACCL_GPU_STR

```
#define ACCL_GPU_STR "gpu"
```

Definition at line 26 of file nnstreamer_plugin_api_filter.h.

9.76.2.7 ACCL_NONE_STR

```
#define ACCL_NONE_STR "none"
```

Macros for accelerator types

Definition at line 20 of file nnstreamer_plugin_api_filter.h.

9.76.2.8 ACCL_NPU_EDGE_TPU_STR

```
#define ACCL_NPU_EDGE_TPU_STR "npu.edgetpu"
```

Definition at line 30 of file nnstreamer_plugin_api_filter.h.

9.76.2.9 ACCL_NPU_MOVIDIUS_STR

```
#define ACCL_NPU_MOVIDIUS_STR "npu.movidius"
```

Definition at line 29 of file nnstreamer_plugin_api_filter.h.

9.76.2.10 ACCL_NPU_SLSI_STR

```
#define ACCL_NPU_SLSI_STR "npu.slsi"
```

Definition at line 33 of file nnstreamer_plugin_api_filter.h.

9.76.2.11 ACCL_NPU_SR_STR

```
#define ACCL_NPU_SR_STR "npu.sr"
```

Definition at line 34 of file nnstreamer_plugin_api_filter.h.

9.76.2.12 ACCL_NPU_SRCN_STR

```
#define ACCL_NPU_SRCN_STR "npu.srcn" /** srcn hardware supported by nnfw */
```

Definition at line 32 of file nnstreamer_plugin_api_filter.h.

9.76.2.13 ACCL_NPU_STR

```
#define ACCL_NPU_STR "npu"
```

Todo Define ACCL_DSP_STR

Definition at line 28 of file nnstreamer_plugin_api_filter.h.

9.76.2.14 ACCL_NPU_VIVANTE_STR

```
#define ACCL_NPU_VIVANTE_STR "npu.vivante"
```

Definition at line 31 of file nnstreamer_plugin_api_filter.h.

9.76.2.15 checkGstTensorFilterFrameworkVersion

```
#define checkGstTensorFilterFrameworkVersion(  
    value,  
    version) ((GST_TENSOR_FILTER_FRAMEWORK_BASE | ((version) << 16)) == (value &  
0xFFFFFFFFFFFFFFFF0000ULL))
```

Check the value of the version field of GstTensorFilterFramework.

Definition at line 47 of file nnstreamer_plugin_api_filter.h.

9.76.2.16 GST_TENSOR_FILTER_API_VERSION_DEFINED

```
#define GST_TENSOR_FILTER_API_VERSION_DEFINED (1)
```

Definition at line 40 of file `nnstreamer_plugin_api_filter.h`.

9.76.2.17 GST_TENSOR_FILTER_API_VERSION_MAX

```
#define GST_TENSOR_FILTER_API_VERSION_MAX (1) /* The maximum API version supported (recommended) */
```

Definition at line 42 of file `nnstreamer_plugin_api_filter.h`.

9.76.2.18 GST_TENSOR_FILTER_API_VERSION_MIN

```
#define GST_TENSOR_FILTER_API_VERSION_MIN (0) /* The minimum API version supported (could be obsolete) */
```

Definition at line 41 of file `nnstreamer_plugin_api_filter.h`.

9.76.2.19 GST_TENSOR_FILTER_FRAMEWORK_BASE

```
#define GST_TENSOR_FILTER_FRAMEWORK_BASE (0xDEAFDEAD00000000ULL)
```

Definition at line 36 of file `nnstreamer_plugin_api_filter.h`.

9.76.2.20 GST_TENSOR_FILTER_FRAMEWORK_V0

```
#define GST_TENSOR_FILTER_FRAMEWORK_V0 (GST_TENSOR_FILTER_FRAMEWORK_BASE)
```

Definition at line 37 of file `nnstreamer_plugin_api_filter.h`.

9.76.2.21 GST_TENSOR_FILTER_FRAMEWORK_V1

```
#define GST_TENSOR_FILTER_FRAMEWORK_V1 (GST_TENSOR_FILTER_FRAMEWORK_BASE | 0x10000ULL)
```

Definition at line 38 of file `nnstreamer_plugin_api_filter.h`.

9.76.2.22 parse_accl_hw

```
#define parse_accl_hw(  
    ... ) parse_accl_hw_fill((parse_accl_args) {__VA_ARGS__})
```

workaround to provide default arguments

Definition at line 544 of file nntstreamer_plugin_api_filter.h.

9.76.3 Typedef Documentation

9.76.3.1 GstTensorFilterFramework

```
typedef struct _GstTensorFilterFramework GstTensorFilterFramework
```

Definition at line 240 of file nntstreamer_plugin_api_filter.h.

9.76.3.2 GstTensorFilterFrameworkEventData

```
typedef struct _GstTensorFilterFrameworkEventData GstTensorFilterFrameworkEventData
```

User data for the tensor_tilter subplugin related events.

9.76.3.3 GstTensorFilterFrameworkInfo

```
typedef struct _GstTensorFilterFrameworkInfo GstTensorFilterFrameworkInfo
```

Tensor_Filter Subplugin framework related information.

All the information except the supported accelerator is provided statically. Accelerators can be provided based on static or dynamic check dependent on framework support.

9.76.3.4 GstTensorFilterFrameworkStatistics

```
typedef struct _GstTensorFilterFrameworkStatistics GstTensorFilterFrameworkStatistics
```

Structure definition for tensor-filter framework statistics.

9.76.3.5 GstTensorFilterProperties

```
typedef struct _GstTensorFilterProperties GstTensorFilterProperties
```

GstTensorFilter's properties for NN framework (internal data structure)

Because custom filters of `tensor_filter` may need to access internal data of `GstTensorFilter`, we define this data structure here.

9.76.3.6 tensors_layout

```
typedef tensor_layout tensors_layout[NNS_TENSOR_SIZE_LIMIT]
```

Definition at line 104 of file `nnstreamer_plugin_api_filter.h`.

9.76.4 Enumeration Type Documentation

9.76.4.1 accl_hw

```
enum accl_hw
```

acceleration hw properties.

Enabling acceleration (choosing any accelerator hardware other `ACCL_NONE`) will enable acceleration for the framework dependent on the acceleration supported by the framework.

For example, enabling acceleration with `tf-lite` will enable `NNAPI`. However, with `NNFW` will enable `GPU/NEON` etc.

Appropriate acceleration should be used with each framework. For example, `ACCL_CPU_NEON` is supported with `NNFW` tensor filter. Using `ACCL_NEON` with `pytorch` would result in a warning message, and the accelerator would fallback on `ACCL_AUTO`.

`ACCL_AUTO` automatically chooses the accelerator based on the ones supported by the subplugin framework. However, `ACCL_DEFAULT` would use the accelerator set by the subplugin framework, if any.

Note

Acceleration for a framework can be enabled/disabled in the build. If the acceleration for a framework was disabled in the build, setting the accelerator while use a tensor filter with that framework will have no effect.

Add definition of new accelerators to `accl_hw_get_type()` in `tensor_filter_common.c` as well.

Enumerator

<code>ACCL_NONE</code>	no explicit acceleration no acceleration (defaults to CPU)
<code>ACCL_AUTO</code>	If there is no default config, and device needs to be specified, fallback to <code>ACCL_AUTO</code> choose optimized device automatically
<code>ACCL_DEFAULT</code>	use default device configuration by the framework
<code>ACCL_CPU</code>	Enables acceleration, <code>0xn000</code> any version of that device, <code>0xnxxx</code> : device # <code>xxx-1</code> specify device as CPU, if possible

Enumerator

ACCL_CPU_SIMD	specify device as SIMD in cpu, if possible
ACCL_CPU_NEON	specify device as NEON (alias for SIMD) in cpu, if possible
ACCL_GPU	specify device as GPU, if possible
ACCL_NPU	Todo Define ACCL_DSP specify device as any NPU, if possible
ACCL_NPU_MOVIDIUS	specify device as movidius, if possible
ACCL_NPU_EDGE_TPU	specify device as edge tpu, if possible
ACCL_NPU_VIVANTE	specify device as vivante, if possible
ACCL_NPU_SRCN	specify device as srcn, if possible
ACCL_NPU_SLSI	specify device as S.LSI, if possible
ACCL_NPU_SR	specify device as any SR npu, if possible

Definition at line 80 of file nnstreamer_plugin_api_filter.h.

9.76.4.2 event_ops

enum `event_ops`

Tensor_Filter Subplugin related events.

These are possible set of events that can be supported by the tensor filter subplugin.

Enumerator

DESTROY_NOTIFY	Free the data element allocated in the invoke callback
RELOAD_MODEL	Reloads the subplugin with newly provided model
CUSTOM_PROP	Update the custom properties for the framework
SET_INPUT_PROP	Update input tensor info and layout
SET_OUTPUT_PROP	Update output tensor info and layout
SET_ACCELERATOR	Update accelerator of the subplugin to be used as backend
CHECK_HW_AVAILABILITY	Check the hw availability with custom option

Definition at line 177 of file nnstreamer_plugin_api_filter.h.

9.76.4.3 model_info_ops

enum `model_info_ops`

Tensor_Filter Subplugin's model related info gathering operations.

Enumerator

GET_IN_OUT_INFO	Gets the input and output tensor info
SET_INPUT_INFO	Sets the provided input tensor info, and get updated output tensor info

Definition at line 191 of file nstreamer_plugin_api_filter.h.

9.76.5 Function Documentation

9.76.5.1 get_accl_hw_str()

```
const char* get_accl_hw_str (
    const accl_hw key )
```

return string based on accl_hw type

Parameters

key	The key enum value
-----	--------------------

Returns

Corresponding string. Returns ACCL_NONE_STR if not found.

Note

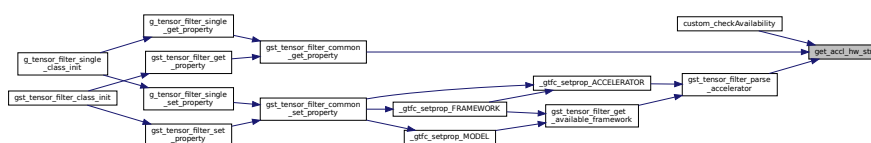
Do not free the returned char *

Definition at line 2598 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.76.5.2 get_accl_hw_type()

```
accl_hw get_accl_hw_type (
    const char * str )
```

return accl_hw type from string

9.76.5.3 nstreamer_filter_exit()

```
void nstreamer_filter_exit (
    const char * name )
```

Filter's sub-plugin may call this to unregister itself.

Parameters

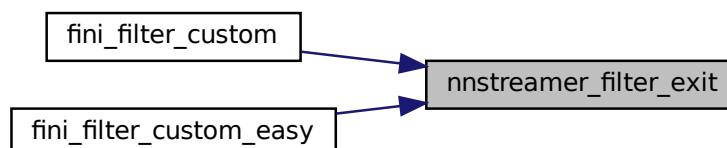
in	<i>name</i>	The name of filter sub-plugin.
----	-------------	--------------------------------

Definition at line 638 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



Returns

TRUE if registered. FALSE is failed or duplicated.

Note

Do not change the subplugins callbacks after probing the filter.

Parameters

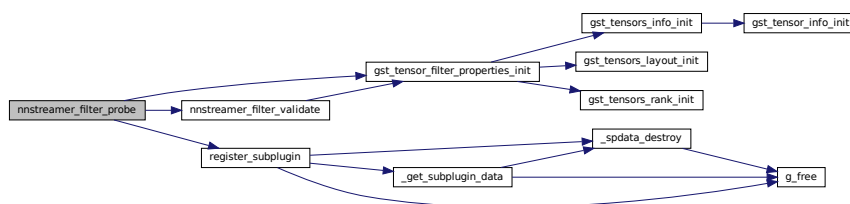
<code>in</code>	<code>tfs</code>	Tensor-Filter Sub-Plugin to be registered.
-----------------	------------------	--

Returns

TRUE if registered. FALSE is failed or duplicated.

Definition at line 611 of file `tensor_filter_common.c`.

Here is the call graph for this function:

**9.76.5.6 nnstreamer_filter_set_custom_property_desc()**

```

void nnstreamer_filter_set_custom_property_desc (
    const char * name,
    const char * prop,
    ... )
  
```

set custom property description for tensor filter sub-plugin

Definition at line 647 of file `tensor_filter_common.c`.

Here is the call graph for this function:



9.76.5.7 nnstreamer_filter_shared_model_get()

```
void* nnstreamer_filter_shared_model_get (
    void * instance,
    const char * key )
```

Get the shared model representation that is already shared and has the same key.

Parameters

in	<i>instance</i>	The instance that is sharing the model representation. It will be registered at the referred list.
in	<i>key</i>	The key to find the matched shared representation.

Returns

The model interpreter. NULL if it does not exist.

Definition at line 2993 of file tensor_filter_common.c.

9.76.5.8 nnstreamer_filter_shared_model_insert_and_get()

```
void* nnstreamer_filter_shared_model_insert_and_get (
    void * instance,
    char * key,
    void * interpreter )
```

Insert the new shared model representation and get the value.

Parameters

in	<i>instance</i>	The instance that is sharing the model representation. It will be registered at the referred list.
in	<i>key</i>	The key for shared model.
in	<i>interpreter</i>	The interpreter to be shared.

Returns

The model interpreter inserted. NULL if it is already inserted.

Internal error case. The interpreter already exists in shared table, do not insert and return null.

Definition at line 3026 of file tensor_filter_common.c.

9.76.5.9 nnstreamer_filter_shared_model_remove()

```
int nnstreamer_filter_shared_model_remove (
    void * instance,
```



```
const char * key,
void(*) (void *) free_callback )
```

Remove the instance registered at the referred list of shared model table. If referred list is empty, `free_callback` is executed.

Parameters

in	<i>instance</i>	The instance that should be removed from the referred list.
in	<i>key</i>	The key to find the shared model.
in	<i>free_callback</i>	The callback function to destroy the interpreter, which takes the interpreter as arg.

Returns

TRUE if the instance is removed. FALSE if failed to remove it.

Definition at line 3081 of file `tensor_filter_common.c`.

9.76.5.10 nntstreamer_filter_shared_model_replace()

```
void nntstreamer_filter_shared_model_replace (
    void * instance,
    const char * key,
    void * new_interpreter,
    void(*) (void *, void *) replace_callback,
    void(*) (void *) free_callback )
```

Helper to reload interpreter for instances that has shared key. `replace_callback` is called iterating instances in referred list.

Parameters

in	<i>instance</i>	The instance that is sharing the model representation.
in	<i>key</i>	The key to find the shared model.
in	<i>interpreter</i>	The new interpreter to replace.
in	<i>replace_callback</i>	The callback function to replace with new interpreter.
in	<i>free_callback</i>	The callback function to destroy the old interpreter.

Definition at line 3128 of file `tensor_filter_common.c`.

9.76.5.11 parse_accl_hw_fill()

```
accl_hw parse_accl_hw_fill (
    parse_accl_args accl_args )
```

parse user given string to extract accelerator based on given regex filling in optional arguments

Note

The order of arguments for calling this function is:

- in_accl: user provided input accelerator string
- sup_accl: list of supported accelerator
- auto_accl: auto accelerator (optional)
- def_accl: default accelerator (optional)

remove unsupported accelerators from this list based on runtime system

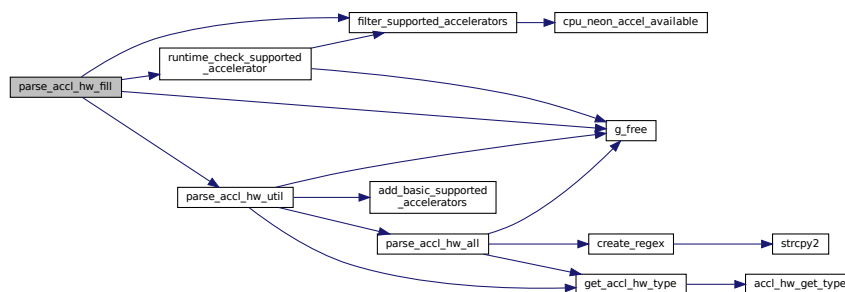
filtered supported accelerators can be empty

update default accelerator if it is not available at runtime

update auto accelerator if it is not available at runtime

Definition at line 2833 of file tensor_filter_common.c.

Here is the call graph for this function:

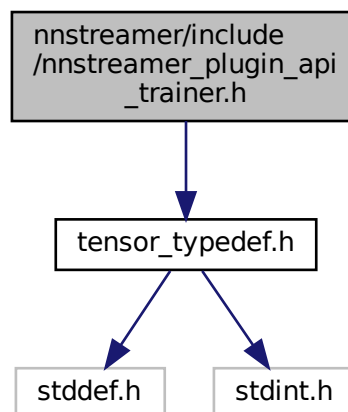


9.77 nnstreamer/include/nnstreamer_plugin_api_trainer.h File Reference

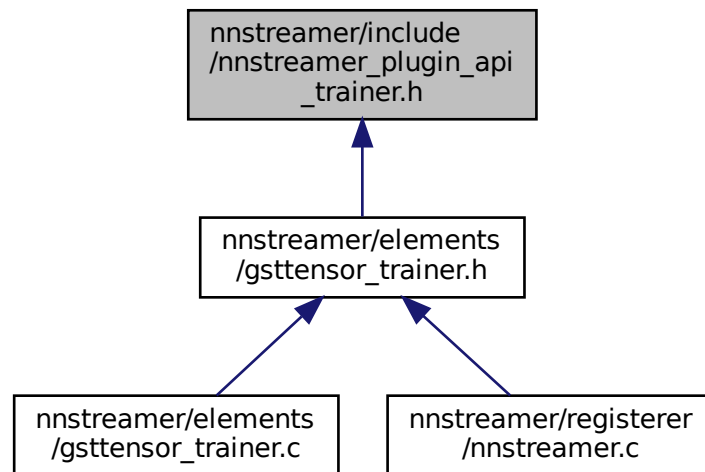
Mandatory APIs for NNStreamer Trainer sub-plugins (No External Dependencies)

```
#include "tensor_typedef.h"
```

Include dependency graph for nnstreamer_plugin_api_trainer.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorTrainerProperties](#)
GstTensorTrainer's properties for neural network framework (internal data structure)
- struct [_GstTensorTrainerFrameworkInfo](#)
GstTensorTrainer's subplugin framework related information.
- struct [_GstTensorTrainerEventNotifier](#)
GstTensorTrainer's event notifier.
- struct [_GstTensorTrainerFramework](#)
tensor_trainer subplugin definition

Macros

- #define [GST_TENSOR_TRAINER_FRAMEWORK_BASE](#) (0xDEAFDEAD00000000ULL)
- #define [GST_TENSOR_TRAINER_FRAMEWORK_V1](#) (GST_TENSOR_TRAINER_FRAMEWORK_BASE | 0x10000ULL)

Typedefs

- typedef struct [_GstTensorTrainerProperties](#) [GstTensorTrainerProperties](#)
GstTensorTrainer's properties for neural network framework (internal data structure)
- typedef struct [_GstTensorTrainerFrameworkInfo](#) [GstTensorTrainerFrameworkInfo](#)
GstTensorTrainer's subplugin framework related information.
- typedef struct [_GstTensorTrainerFramework](#) [GstTensorTrainerFramework](#)
- typedef struct [_GstTensorTrainerEventNotifier](#) [GstTensorTrainerEventNotifier](#)
GstTensorTrainer's event notifier.

Enumerations

- enum `GstTensorTrainerEventType` { `TRAINER_EVENT_EPOCH_COMPLETION`, `TRAINER_EVENT_TRAINING_COMPLETION`, `TRAINER_EVENT_UNKNOWN` }

GstTensorTrainer's event type list.

Functions

- int `nnstreamer_trainer_probe` (`GstTensorTrainerFramework` *ttsp)
Trainer's sub-plugin should call this function to register itself.
- int `nnstreamer_trainer_exit` (`GstTensorTrainerFramework` *ttsp)
Trainer's sub-plugin may call this to unregister itself.
- void `nnstreamer_trainer_notify_event` (`GstTensorTrainerEventNotifier` *notifier, `GstTensorTrainerEventType` type, void *data)
Trainer's sub-plugin call this to notify event.

9.77.1 Detailed Description

Mandatory APIs for NNStreamer Trainer sub-plugins (No External Dependencies)

NNStreamer API for Tensor_Trainer Sub-Plugins Copyright (C) 2022 Hyunil Park hyunil46.park@samsung.com

Date

1 Dec 2022

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Hyunil Park hyunil46.park@samsung.com

Bug No known bugs except for NYI items

9.77.2 Macro Definition Documentation

9.77.2.1 GST_TENSOR_TRAINER_FRAMEWORK_BASE

```
#define GST_TENSOR_TRAINER_FRAMEWORK_BASE (0xDEAFDEAD00000000ULL)
```

Definition at line 19 of file `nnstreamer_plugin_api_trainer.h`.

9.77.2.2 GST_TENSOR_TRAINER_FRAMEWORK_V1

```
#define GST_TENSOR_TRAINER_FRAMEWORK_V1 (GST_TENSOR_TRAINER_FRAMEWORK_BASE | 0x10000ULL)
```

Definition at line 20 of file nnstreamer_plugin_api_trainer.h.

9.77.3 Typedef Documentation

9.77.3.1 GstTensorTrainerEventNotifier

```
typedef struct _GstTensorTrainerEventNotifier GstTensorTrainerEventNotifier
```

GstTensorTrainer's event notifier.

Subplugins must send events to tensor_trainer with a notifier.

9.77.3.2 GstTensorTrainerFramework

```
typedef struct _GstTensorTrainerFramework GstTensorTrainerFramework
```

Definition at line 60 of file nnstreamer_plugin_api_trainer.h.

9.77.3.3 GstTensorTrainerFrameworkInfo

```
typedef struct _GstTensorTrainerFrameworkInfo GstTensorTrainerFrameworkInfo
```

GstTensorTrainer's subplugin framework related information.

All the information is provided statically.

9.77.3.4 GstTensorTrainerProperties

```
typedef struct _GstTensorTrainerProperties GstTensorTrainerProperties
```

GstTensorTrainer's properties for neural network framework (internal data structure)

Internal data of GstTensorTrainer required by tensor_trainer's custom subplugin.

9.77.4 Enumeration Type Documentation

9.77.4.1 GstTensorTrainerEventType

```
enum GstTensorTrainerEventType
```

GstTensorTrainer's event type list.

Event types that subplugins must send to tensor_trainer

Enumerator

TRAINER_EVENT_EPOCH_COMPLETION	one epoch is complete in subplugin
TRAINER_EVENT_TRAINING_COMPLETION	training is complete in subplugin
TRAINER_EVENT_UNKNOWN	unknown event

Definition at line 67 of file `nnstreamer_plugin_api_trainer.h`.

9.77.5 Function Documentation

9.77.5.1 `nnstreamer_trainer_exit()`

```
int nnstreamer_trainer_exit (
    GstTensorTrainerFramework * ttsp )
```

Trainer's sub-plugin may call this to unregister itself.

Parameters

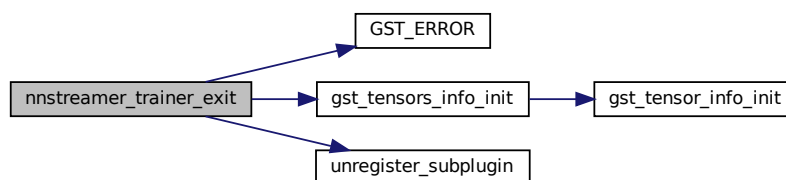
in	<i>ttsp</i>	tensor_trainer sub-plugin to be unregistered.
----	-------------	---

Returns

TRUE if unregistered. FALSE is failed.

Definition at line 1398 of file `gsttensor_trainer.c`.

Here is the call graph for this function:



9.77.5.2 nnstreamer_trainer_notify_event()

```
void nnstreamer_trainer_notify_event (
    GstTensorTrainerEventNotifier * notifier,
    GstTensorTrainerEventType type,
    void * data )
```

Trainer's sub-plugin call this to notify event.

Parameters

<i>notifier</i>	sub-plugin must send events to tensor_trainer with a notifier.
<i>type</i>	Event types that subplugins must send to tensor_trainer
<i>data</i>	Optional data for the event

Trainer's sub-plugin call this to notify event.

Parameters

in	<i>notifier</i>	event notifier, sub-plugin must send events with this.
in	<i>type</i>	event type

Definition at line 1425 of file gsttensor_trainer.c.

9.77.5.3 nnstreamer_trainer_probe()

```
int nnstreamer_trainer_probe (
    GstTensorTrainerFramework * ttsp )
```

Trainer's sub-plugin should call this function to register itself.

Parameters

in	<i>ttsp</i>	tensor_trainer sub-plugin to be registered.
----	-------------	---

Returns

TRUE if registered. FALSE is failed.

Note

Do not change the subplugins callbacks after probing the filter.

Parameters

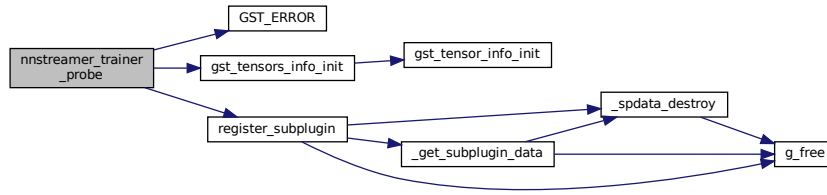
in	<i>ttsp</i>	tensor_trainer sub-plugin to be registered.
----	-------------	---

Returns

TRUE if registered. FALSE is failed or duplicated.

Definition at line 1371 of file gsttensor_trainer.c.

Here is the call graph for this function:

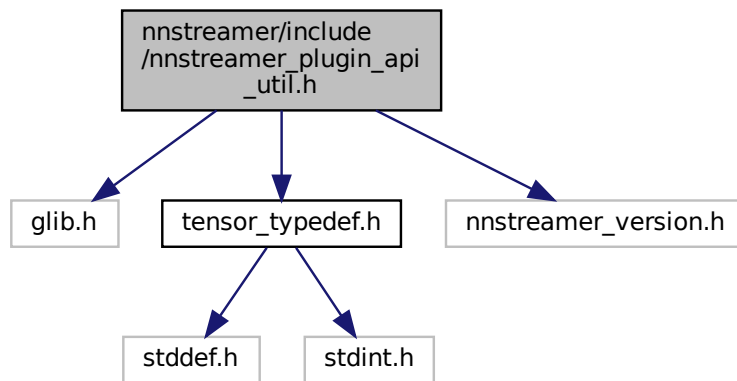


9.78 nnstreamer/include/nnstreamer_plugin_api_util.h File Reference

Optional/Additional NNStreamer APIs for sub-plugin writers. (No GStreamer dependency)

```

#include <glib.h>
#include <tensor_typedef.h>
#include <nnstreamer_version.h>
Include dependency graph for nnstreamer_plugin_api_util.h:
  
```



This graph shows which files directly or indirectly include this file:



Macros

- #define `_STR_NULL(str)` `((str) ? (str) : "(NULL)")`
If the given string is NULL, print "(NULL)". Copied from GST_STR_NULL

- #define `gst_tensors_config_is_static(c)` `((c)->info.format == _NNS_TENSOR_FORMAT_STATIC)`
Macro to check stream format (static tensors for caps negotiation)
- #define `gst_tensors_config_is_flexible(c)` `((c)->info.format == _NNS_TENSOR_FORMAT_FLEXIBLE)`
Macro to check stream format (flexible tensors for caps negotiation)
- #define `gst_tensors_config_is_sparse(c)` `((c)->info.format == _NNS_TENSOR_FORMAT_SPARSE)`
Macro to check stream format (sparse tensors for caps negotiation)

Functions

- void `gst_tensor_info_init` (`GstTensorInfo *info`)
Initialize the tensor info structure.
- void `gst_tensor_info_free` (`GstTensorInfo *info`)
Free allocated data in tensor info structure.
- gsize `gst_tensor_info_get_size` (const `GstTensorInfo *info`)
Get data size of single tensor.
- gboolean `gst_tensor_info_validate` (const `GstTensorInfo *info`)
Check the tensor info is valid.
- gboolean `gst_tensor_info_is_equal` (const `GstTensorInfo *i1`, const `GstTensorInfo *i2`)
Compare tensor info.
- void `gst_tensor_info_copy_n` (`GstTensorInfo *dest`, const `GstTensorInfo *src`, const guint n)
Copy tensor info up to n elements.
- void `gst_tensor_info_copy` (`GstTensorInfo *dest`, const `GstTensorInfo *src`)
Copy tensor info.
- gboolean `gst_tensor_info_convert_to_meta` (`GstTensorInfo *info`, `GstTensorMetaInfo *meta`)
Convert `GstTensorInfo` structure to `GstTensorMetaInfo`.
- guint `gst_tensor_info_get_rank` (const `GstTensorInfo *info`)
Get tensor rank.
- `GstTensorInfo *` `gst_tensors_info_get_nth_info` (`GstTensorsInfo *info`, guint index)
Get the pointer of nth tensor information.
- void `gst_tensors_info_init` (`GstTensorsInfo *info`)
Initialize the tensors info structure.
- void `gst_tensors_info_free` (`GstTensorsInfo *info`)
Free allocated data in tensors info structure.
- gsize `gst_tensors_info_get_size` (const `GstTensorsInfo *info`, gint index)
Get data size of single tensor.
- guint `gst_tensors_info_parse_dimensions_string` (`GstTensorsInfo *info`, const gchar *dim_string)
Parse the string of dimensions.
- guint `gst_tensors_info_parse_types_string` (`GstTensorsInfo *info`, const gchar *type_string)
Parse the string of types.
- guint `gst_tensors_info_parse_names_string` (`GstTensorsInfo *info`, const gchar *name_string)
Parse the string of names.
- gchar * `gst_tensors_info_get_dimensions_string` (const `GstTensorsInfo *info`)
Get the string of dimensions in tensors info.
- gchar * `gst_tensors_info_get_rank_dimensions_string` (const `GstTensorsInfo *info`, const unsigned int rank)
Get the string of dimensions in tensors info and rank count.
- gchar * `gst_tensors_info_get_types_string` (const `GstTensorsInfo *info`)
Get the string of types in tensors info.
- gchar * `gst_tensors_info_get_names_string` (const `GstTensorsInfo *info`)
Get the string of tensor names in tensors info.
- gboolean `gst_tensors_info_validate` (const `GstTensorsInfo *info`)

- Check the tensors info is valid.*

 - gboolean `gst_tensors_info_is_equal` (const `GstTensorsInfo` *i1, const `GstTensorsInfo` *i2)

Compare tensors info.
- void `gst_tensors_info_copy` (`GstTensorsInfo` *dest, const `GstTensorsInfo` *src)

Copy tensor info.
- gchar * `gst_tensors_info_to_string` (const `GstTensorsInfo` *info)

GstTensorsInfo represented as a string. Caller should free it.
- void `gst_tensors_config_init` (`GstTensorsConfig` *config)

Initialize the tensors config info structure (for other/tensors)
- void `gst_tensors_config_free` (`GstTensorsConfig` *config)

Free allocated data in tensors config structure.
- gboolean `gst_tensors_config_validate` (const `GstTensorsConfig` *config)

Check the tensors are all configured (for other/tensors)
- gboolean `gst_tensors_config_is_equal` (const `GstTensorsConfig` *c1, const `GstTensorsConfig` *c2)

Compare tensor config info (for other/tensors)
- void `gst_tensors_config_copy` (`GstTensorsConfig` *dest, const `GstTensorsConfig` *src)

Copy tensors config.
- gchar * `gst_tensors_config_to_string` (const `GstTensorsConfig` *config)

Tensor config represented as a string. Caller should free it.
- gboolean `gst_tensor_dimension_is_valid` (const `tensor_dim` dim)

Check the tensor dimension is valid.
- gboolean `gst_tensor_dimension_is_equal` (const `tensor_dim` dim1, const `tensor_dim` dim2)

Compare the tensor dimension.
- guint `gst_tensor_dimension_get_rank` (const `tensor_dim` dim)

Get the rank of tensor dimension.
- guint `gst_tensor_dimension_get_min_rank` (const `tensor_dim` dim)

Get the minimum rank of tensor dimension.
- guint `gst_tensor_parse_dimension` (const gchar *dimstr, `tensor_dim` dim)

Parse tensor dimension parameter string.
- gchar * `gst_tensor_get_dimension_string` (const `tensor_dim` dim)

Get dimension string from given tensor dimension.
- gchar * `gst_tensor_get_rank_dimension_string` (const `tensor_dim` dim, const unsigned int rank)

Get dimension string from given tensor dimension and rank count.
- gboolean `gst_tensor_dimension_string_is_equal` (const gchar *dimstr1, const gchar *dimstr2)

Compare dimension strings.
- gulong `gst_tensor_get_element_count` (const `tensor_dim` dim)

Count the number of elements of a tensor.
- gsize `gst_tensor_get_element_size` (`tensor_type` type)

Get element size of tensor type (byte per element)
- `tensor_type` `gst_tensor_get_type` (const gchar *typestr)

Get tensor type from string input.
- const gchar * `gst_tensor_get_type_string` (`tensor_type` type)

Get type string of tensor type.
- `tensor_format` `gst_tensor_get_format` (const gchar *format_str)

Get tensor format from string input.
- const gchar * `gst_tensor_get_format_string` (`tensor_format` format)

Get tensor format string.
- gint `find_key_strv` (const gchar **strv, const gchar *key)

Find the index value of the given key string array.
- void `gst_tensor_meta_info_init` (`GstTensorMetaInfo` *meta)

Initialize the tensor meta info structure.

- void `gst_tensor_meta_info_get_version` (`GstTensorMetaInfo *meta`, `guint *major`, `guint *minor`)
Get the version of tensor meta.
- gboolean `gst_tensor_meta_info_validate` (`GstTensorMetaInfo *meta`)
Check the meta info is valid.
- gsize `gst_tensor_meta_info_get_header_size` (`GstTensorMetaInfo *meta`)
Get the header size to handle a tensor meta.
- gsize `gst_tensor_meta_info_get_data_size` (`GstTensorMetaInfo *meta`)
Get the data size calculated from tensor meta.
- gboolean `gst_tensor_meta_info_update_header` (`GstTensorMetaInfo *meta`, `gpointer header`)
Update header from tensor meta.
- gboolean `gst_tensor_meta_info_parse_header` (`GstTensorMetaInfo *meta`, `gpointer header`)
Parse header and fill the tensor meta.
- gboolean `gst_tensor_meta_info_convert` (`GstTensorMetaInfo *meta`, `GstTensorInfo *info`)
Convert `GstTensorMetaInfo` structure to `GstTensorInfo`.
- `gchar *` `nnstreamer_version_string` (`void`)
Get the version of NNStreamer.
- void `nnstreamer_version_fetch` (`guint *major`, `guint *minor`, `guint *micro`)
Get the version of NNStreamer (int, divided).

9.78.1 Detailed Description

Optional/Additional NNStreamer APIs for sub-plugin writers. (No GStreamer dependency)

NNStreamer Common API Header for sub-plugin writers Copyright (C) 2022 Gichan Jang gichan2.jang@samsung.com

Date

28 Jan 2022

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.78.2 Macro Definition Documentation

9.78.2.1 `_STR_NULL`

```
#define _STR_NULL(  
    str ) ((str) ? (str) : "(NULL)")
```

If the given string is NULL, print "(NULL)". Copied from GST_STR_NULL

Definition at line 26 of file nnstreamer_plugin_api_util.h.

9.78.2.2 `gst_tensors_config_is_flexible`

```
#define gst_tensors_config_is_flexible(  
    c ) ((c)->info.format == \_NNS\_TENSOR\_FORMAT\_FLEXIBLE)
```

Macro to check stream format (flexible tensors for caps negotiation)

Definition at line 279 of file nnstreamer_plugin_api_util.h.

9.78.2.3 `gst_tensors_config_is_sparse`

```
#define gst_tensors_config_is_sparse(  
    c ) ((c)->info.format == \_NNS\_TENSOR\_FORMAT\_SPARSE)
```

Macro to check stream format (sparse tensors for caps negotiation)

Definition at line 284 of file nnstreamer_plugin_api_util.h.

9.78.2.4 `gst_tensors_config_is_static`

```
#define gst_tensors_config_is_static(  
    c ) ((c)->info.format == \_NNS\_TENSOR\_FORMAT\_STATIC)
```

Macro to check stream format (static tensors for caps negotiation)

Definition at line 274 of file nnstreamer_plugin_api_util.h.

9.78.3 Function Documentation

9.78.3.1 `find_key_strv()`

```
gint find_key_strv (  
    const gchar ** strv,  
    const gchar * key )
```

Find the index value of the given key string array.

Returns

Corresponding index

Parameters

<i>strv</i>	Null terminated array of gchar *
<i>key</i>	The key string value

Returns

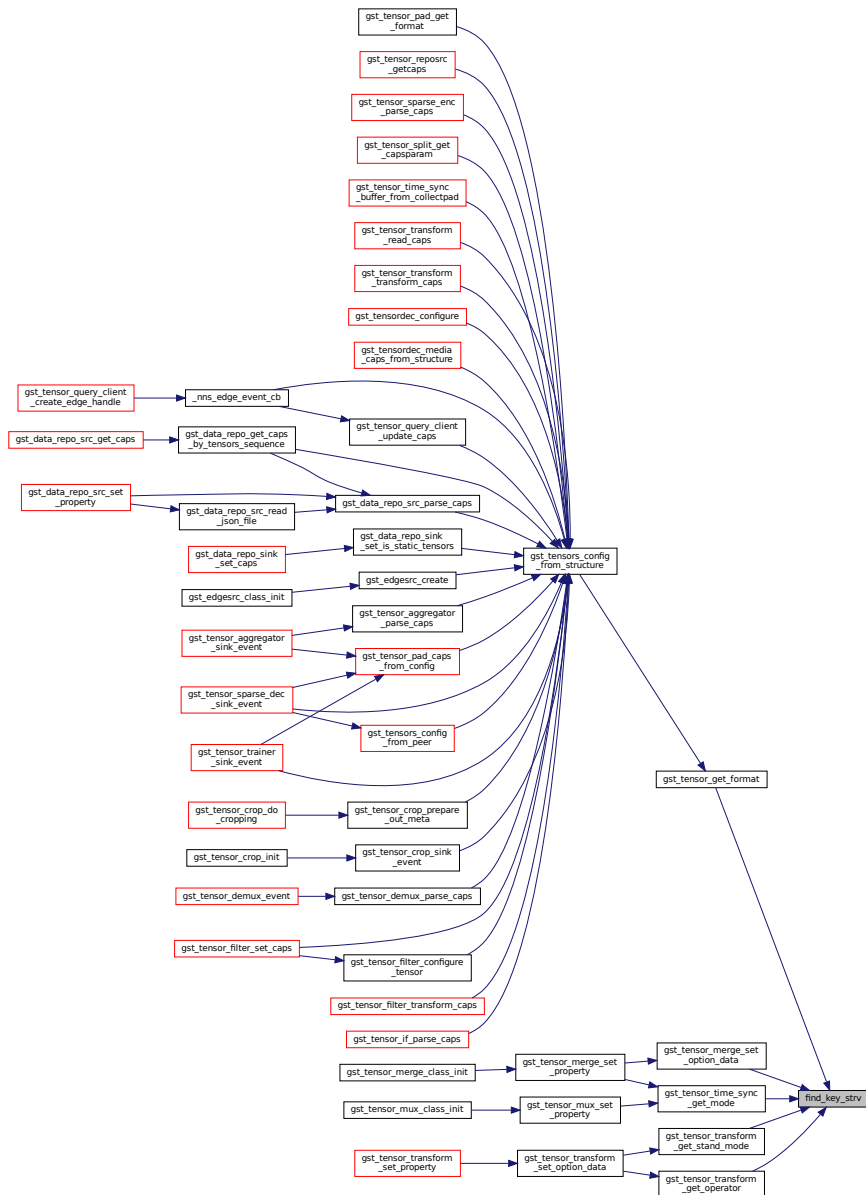
Corresponding index. Returns -1 if not found.

Parameters

<i>strv</i>	Null terminated array of gchar *
<i>key</i>	The key string value

Definition at line 1586 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



9.78.3.2 gst_tensor_dimension_get_min_rank()

```
guint gst_tensor_dimension_get_min_rank (
    const tensor_dim dim )
```

Get the minimum rank of tensor dimension.

The C-arrays with dim 4:4:4 and 4:4:4:1 have same data. In this case, this function returns min rank 3.

Parameters

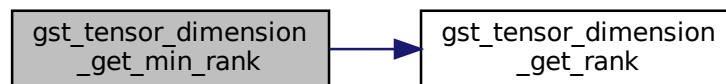
<i>dim</i>	tensor dimension.
------------	-------------------

Returns

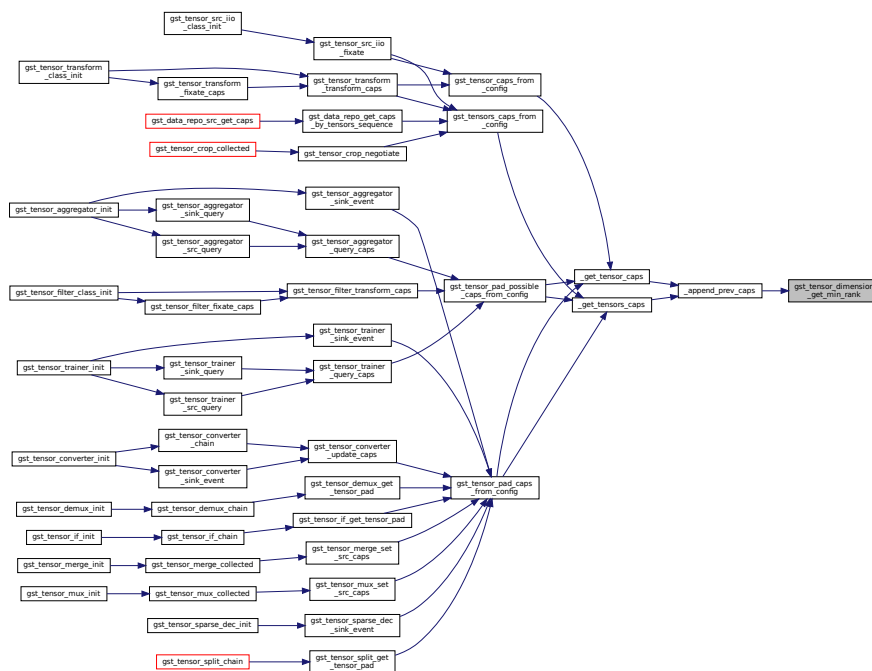
tensor rank (Minimum rank is 1 if given dimension is valid)

Definition at line 1017 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.3 `gst_tensor_dimension_get_rank()`

```

guint gst_tensor_dimension_get_rank (
    const tensor_dim dim )
  
```

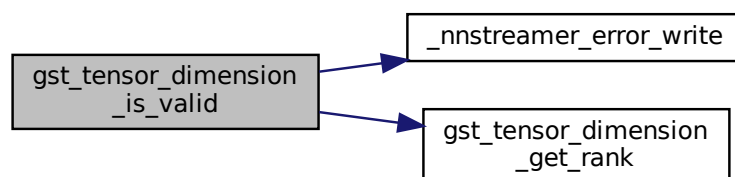
Get the rank of tensor dimension.

Returns

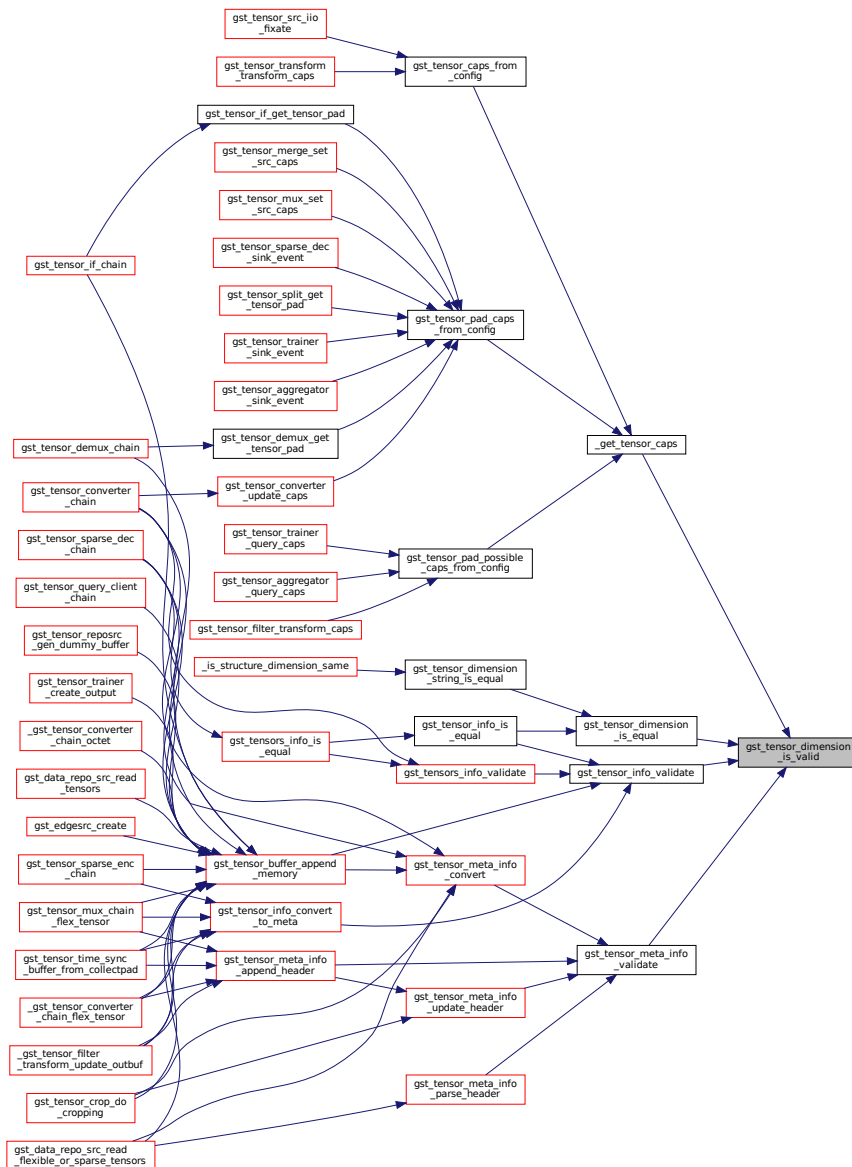
TRUE if dimension is valid

Definition at line 940 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.6 `gst_tensor_dimension_string_is_equal()`

```
gboolean gst_tensor_dimension_string_is_equal (
    const gchar * dimstr1,
    const gchar * dimstr2 )
```

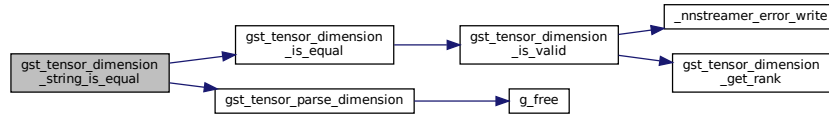
Compare dimension strings.

Returns

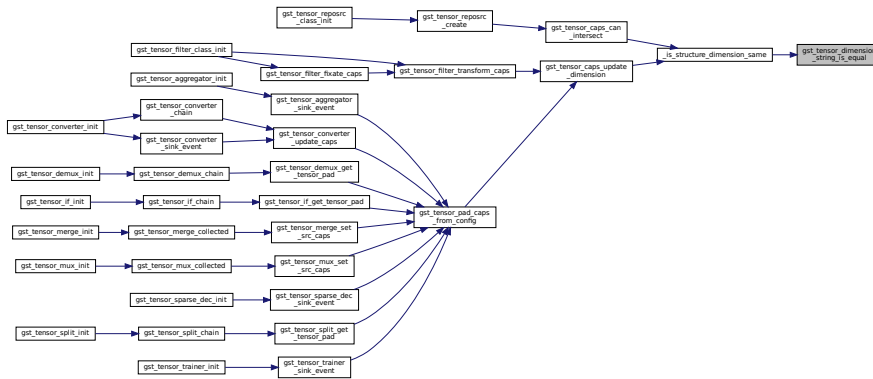
TRUE if equal, FALSE if given dimension strings are invalid or not equal.

Definition at line 1140 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.7 gst_tensor_get_dimension_string()

```

gchar* gst_tensor_get_dimension_string (
    const tensor_dim dim )
    
```

Get dimension string from given tensor dimension.

Parameters

<i>dim</i>	tensor dimension
------------	------------------

Returns

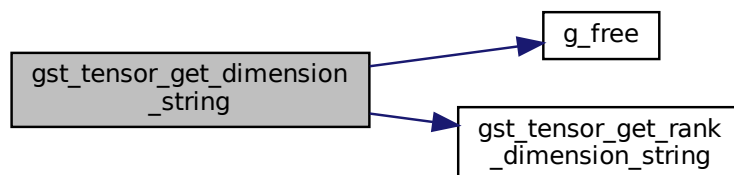
Formatted string of given dimension (d1:d2:d3:....d15:d16).

Note

The returned value should be freed with [g_free\(\)](#)

Definition at line 1083 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Returns

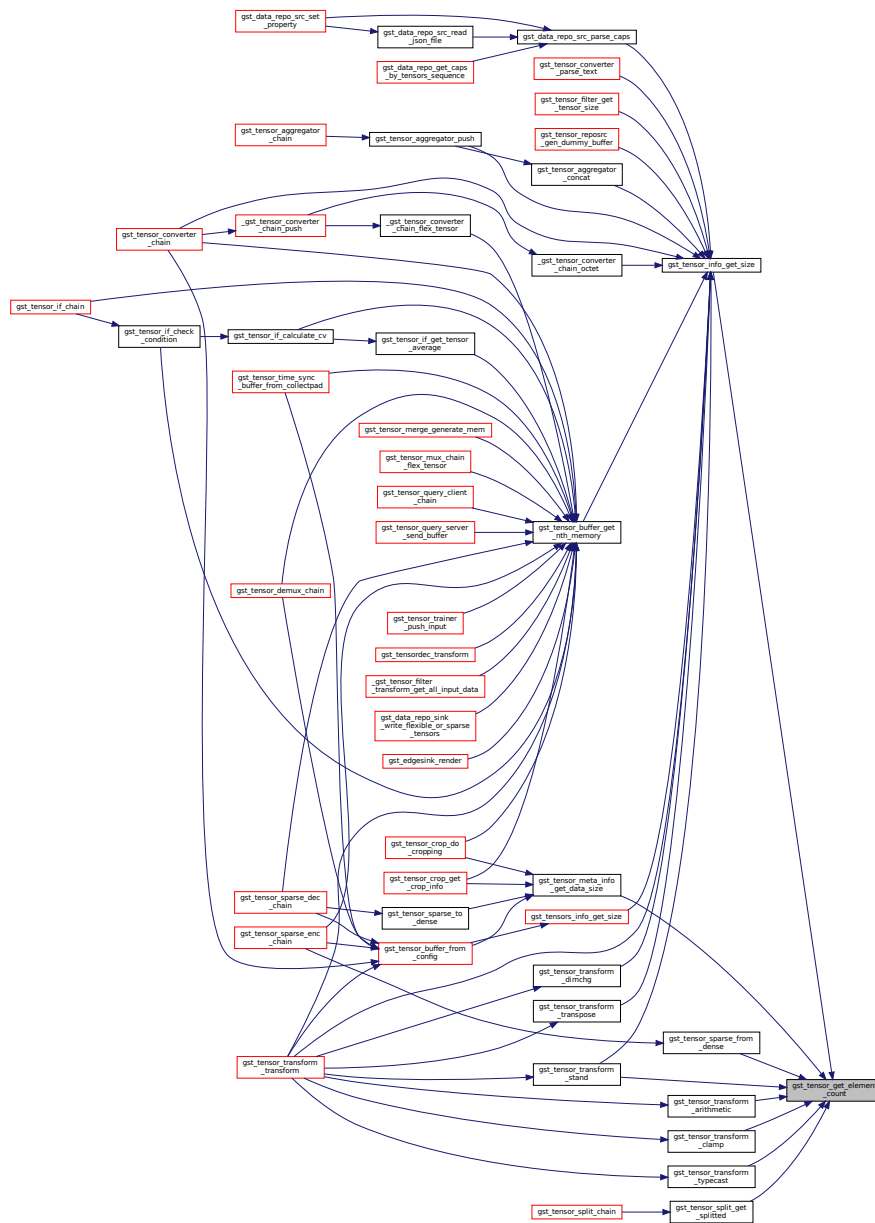
The number of elements. 0 if error.

Parameters

<i>dim</i>	The tensor dimension
------------	----------------------

Definition at line 1186 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



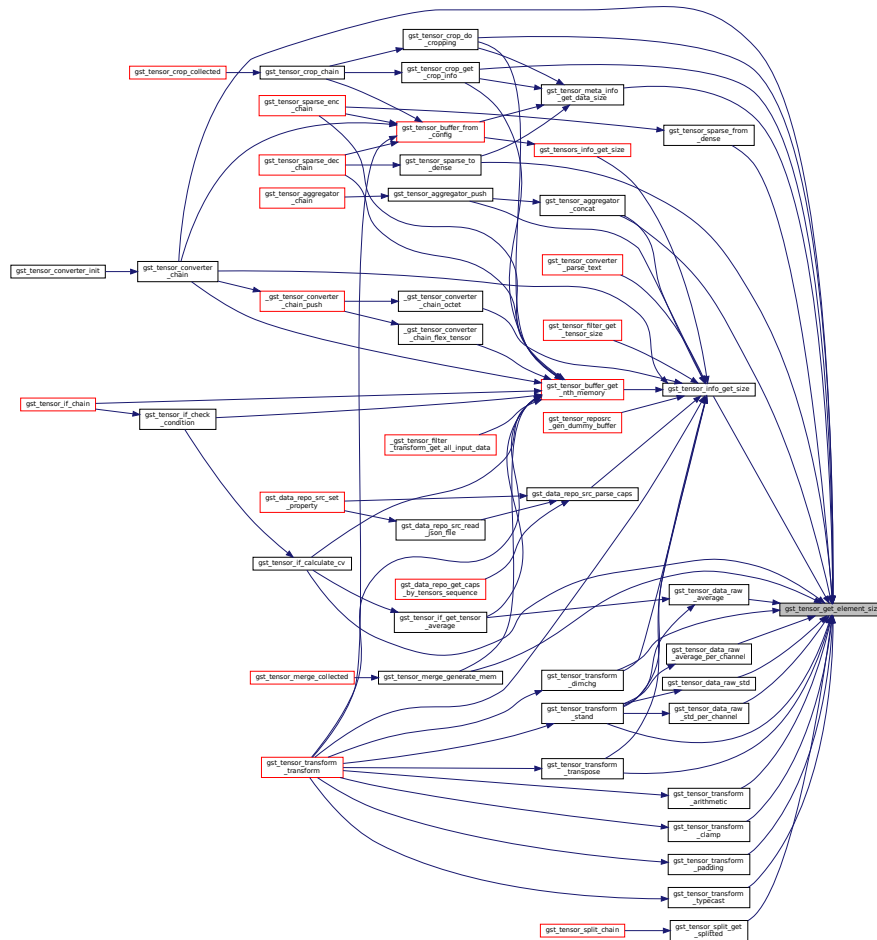
9.78.3.9 `gst_tensor_get_element_size()`

```
gsize gst_tensor_get_element_size (
    tensor_type type )
```

Get element size of tensor type (byte per element)

Definition at line 1205 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the caller graph for this function:



9.78.3.10 `gst_tensor_get_format()`

```
tensor_format gst_tensor_get_format (
    const gchar * format_str )
```

Get tensor format from string input.

Parameters

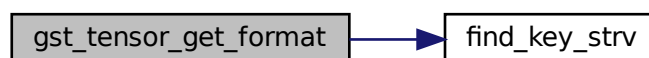
<i>format_str</i>	The string format name, supposed to be one of <code>tensor_format_name[]</code> .
-------------------	---

Returns

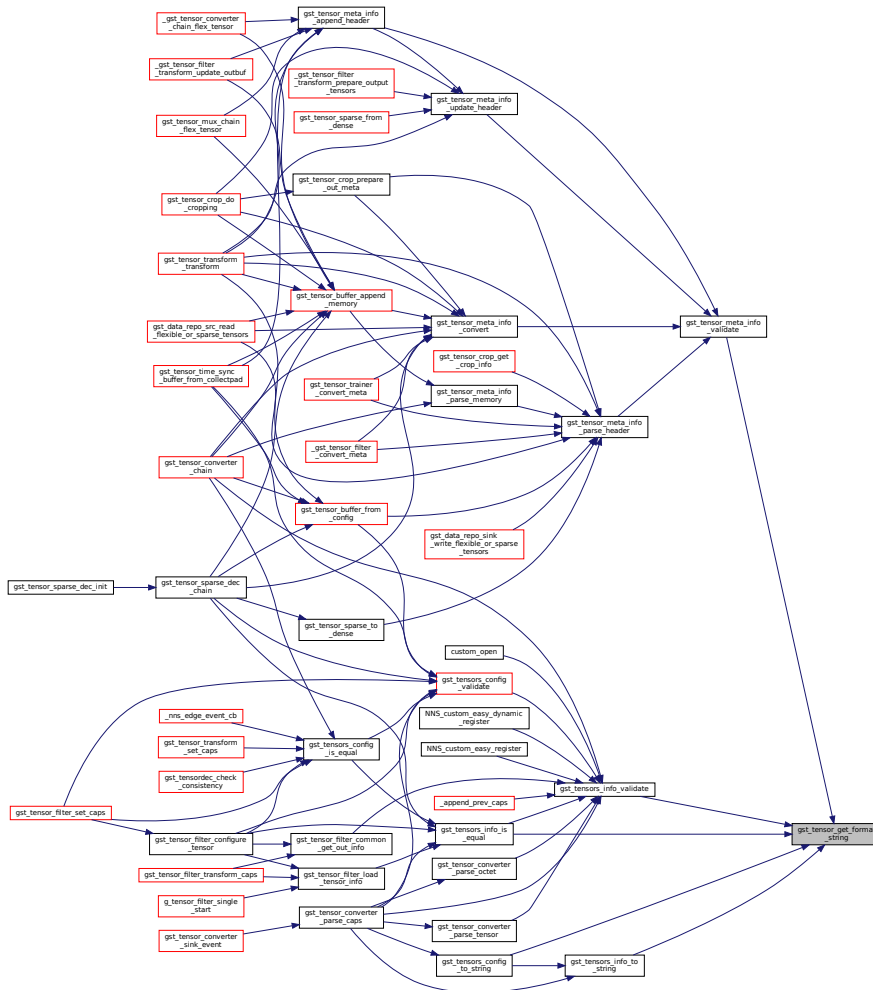
Corresponding `tensor_format`. `_NNS_TENSOR_FORMAT_END` if unrecognized value is there.

Definition at line 1309 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.12 gst_tensor_get_rank_dimension_string()

```
gchar* gst_tensor_get_rank_dimension_string (
    const tensor_dim dim,
    const unsigned int rank )
```

Get dimension string from given tensor dimension and rank count.

Parameters

<i>dim</i>	tensor dimension
<i>rank</i>	rank count of given tensor dimension

Returns

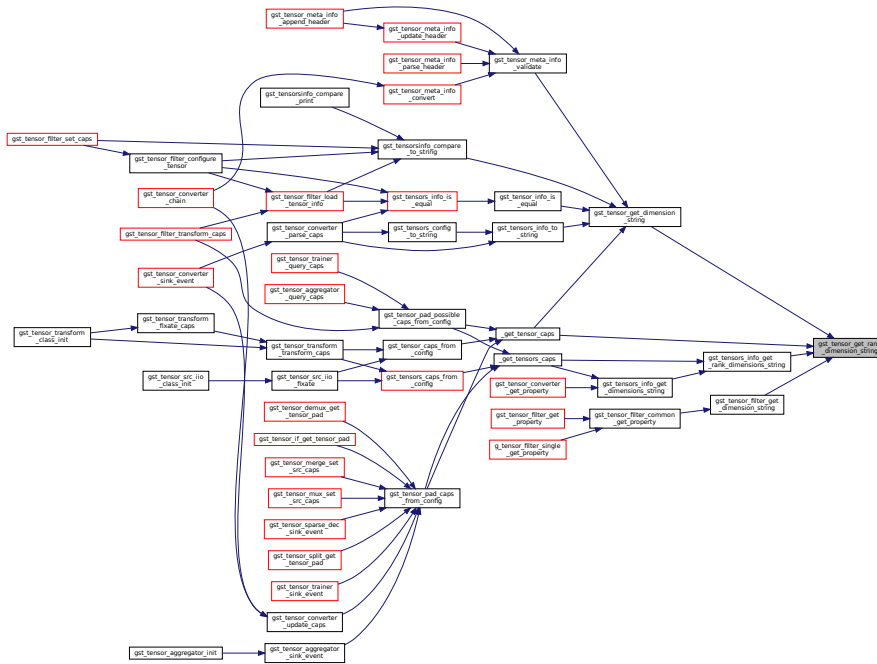
Formatted string of given dimension

Note

If rank count is 3, then returned string is 'd1:d2:d3'. The returned value should be freed with `g_free()`.

Definition at line 1107 of file `nntstreamer_plugin_api_util_impl.c`.

Here is the caller graph for this function:



9.78.3.13 `gst_tensor_get_type()`

```
tensor_type gst_tensor_get_type (
    const gchar * typestr )
```

Get tensor type from string input.

Returns

Corresponding `tensor_type`. `_NNS_END` if unrecognized value is there.

Parameters

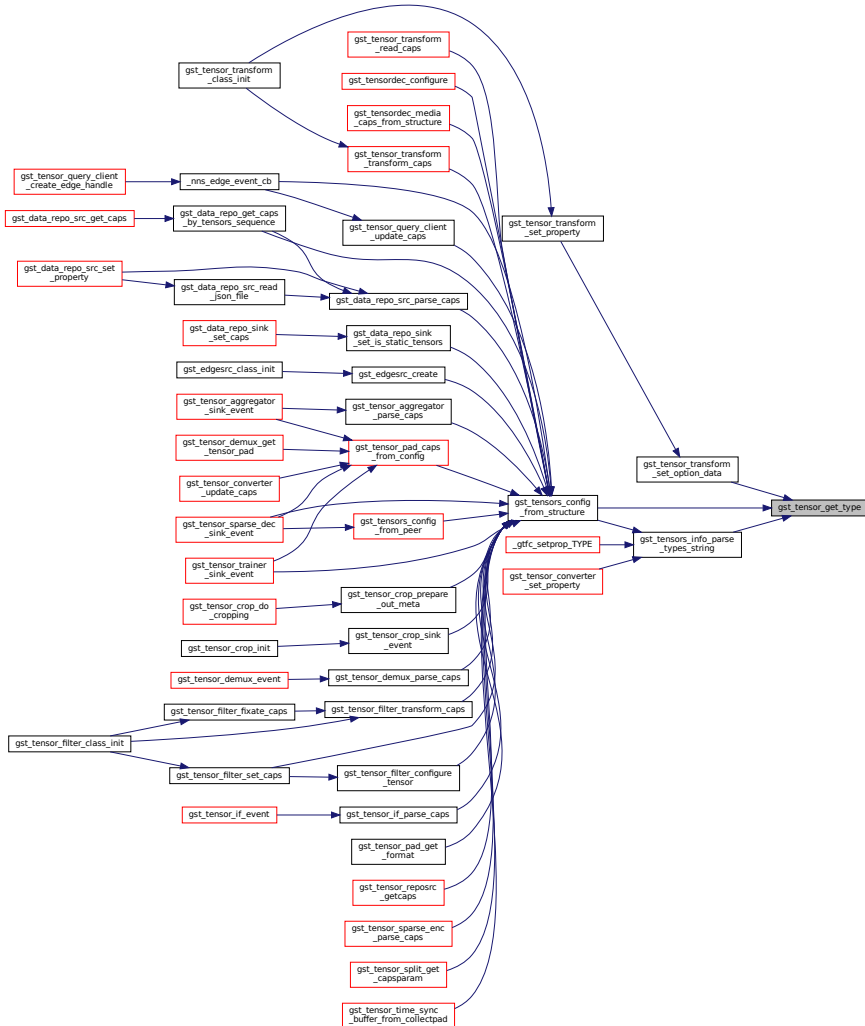
<code>typestr</code>	The string type name, supposed to be one of <code>tensor_element_typename[]</code>
----------------------	--

Definition at line 1218 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



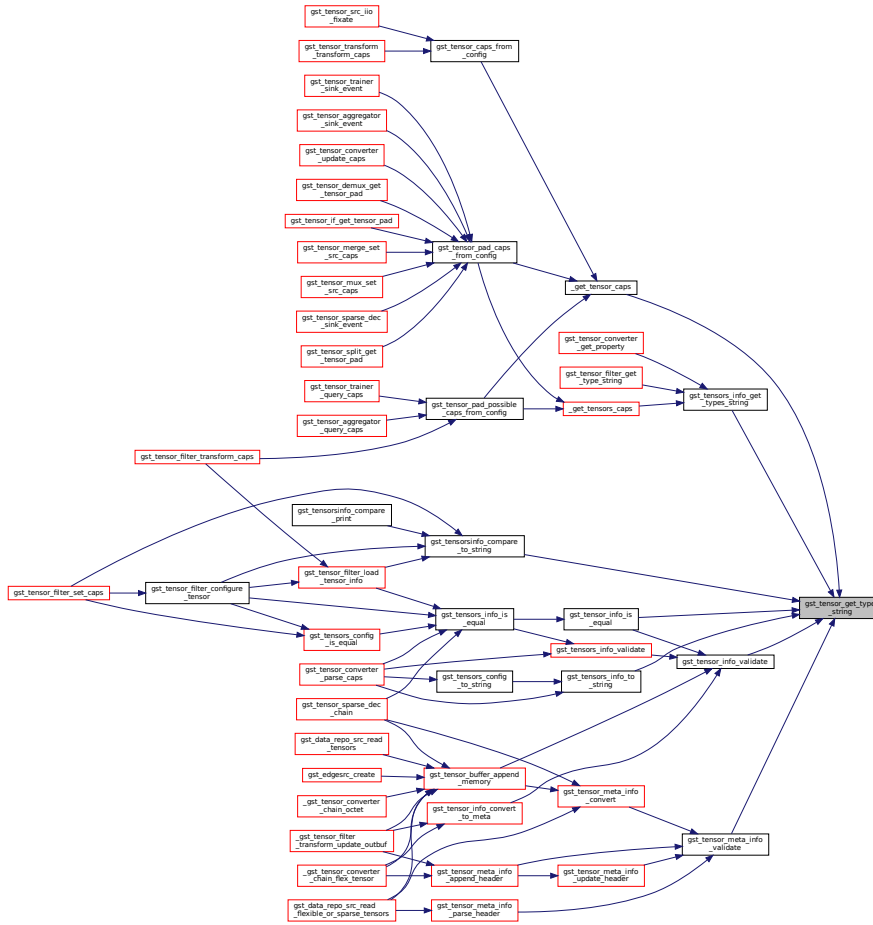
9.78.3.14 `gst_tensor_get_type_string()`

```
const gchar* gst_tensor_get_type_string (
    tensor_type type )
```

Get type string of tensor type.

Definition at line 1296 of file `nstreamer_plugin_api_util_impl.c`.

Here is the caller graph for this function:



9.78.3.15 `gst_tensor_info_convert_to_meta()`

```
gboolean gst_tensor_info_convert_to_meta (
    GstTensorInfo * info,
    GstTensorMetaInfo * meta )
```

Convert `GstTensorInfo` structure to `GstTensorMetaInfo`.

Parameters

in	<i>info</i>	GstTensorInfo to be converted
out	<i>meta</i>	tensor meta structure to be filled

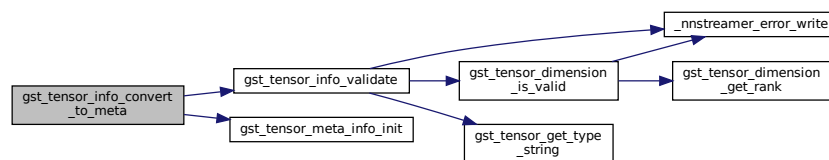
Returns

TRUE if successfully set the meta

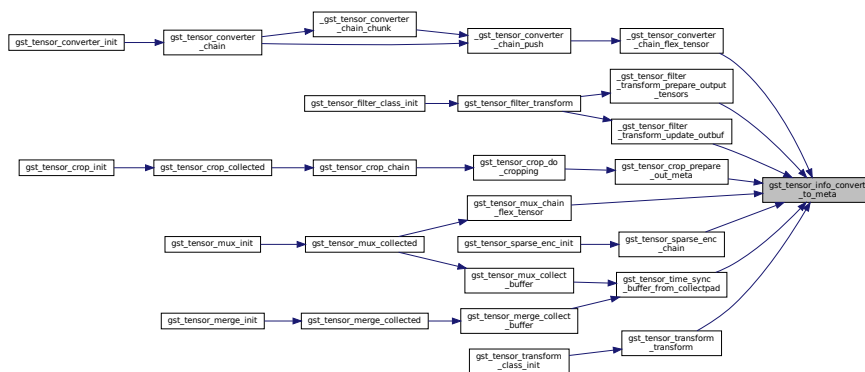
Todo handle rank from info.dimension

Definition at line 260 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.16 `gst_tensor_info_copy()`

```

void gst_tensor_info_copy (
    GstTensorInfo * dest,
    const GstTensorInfo * src )
  
```

Copy tensor info.

Note

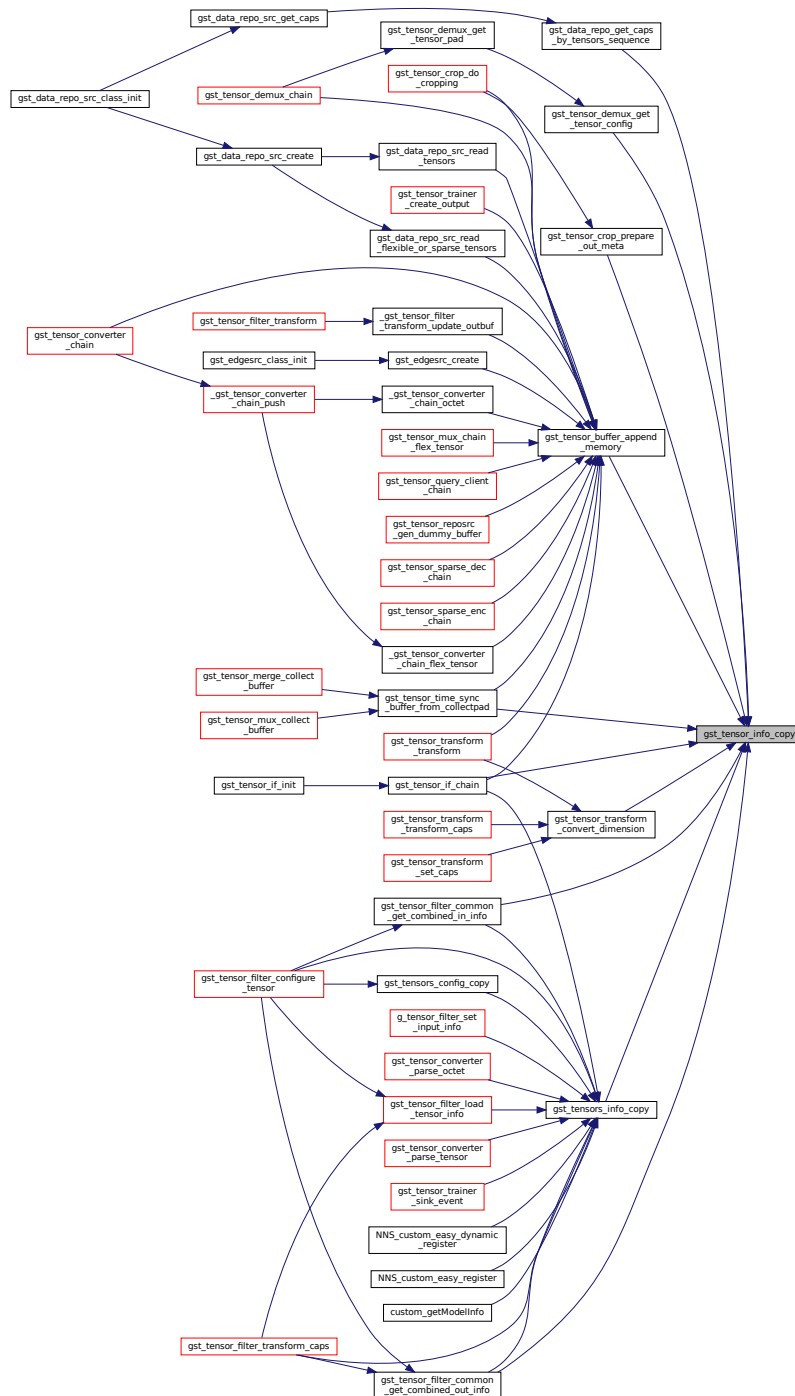
Copied info should be freed with [gst_tensor_info_free\(\)](#)

Definition at line 248 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.17 gst_tensor_info_copy_n()

```
void gst_tensor_info_copy_n (
    GstTensorInfo * dest,
```


9.78.3.18 `gst_tensor_info_free()`

```
void gst_tensor_info_free (  
    GstTensorInfo * info )
```

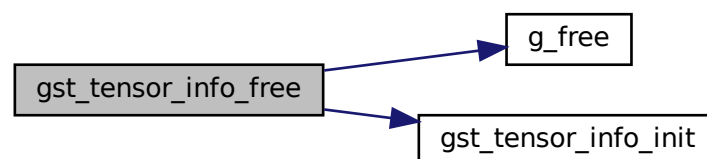
Free allocated data in tensor info structure.

Parameters

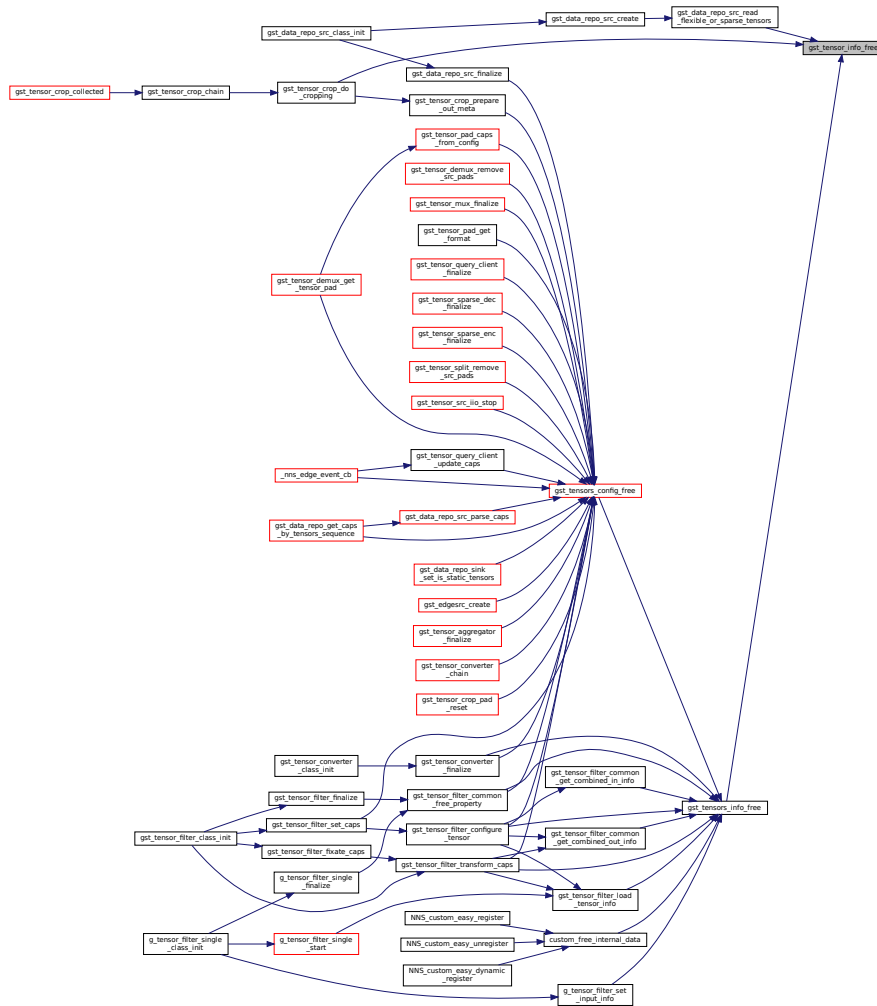
<i>info</i>	tensor info structure
-------------	-----------------------

Definition at line 140 of file `nstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.19 `gst_tensor_info_get_rank()`

```
guint gst_tensor_info_get_rank (
    const GstTensorInfo * info )
```

Get tensor rank.

Parameters

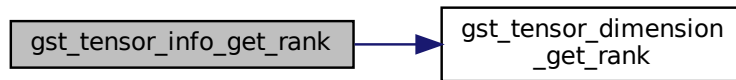
<i>info</i>	tensor info structure
-------------	-----------------------

Returns

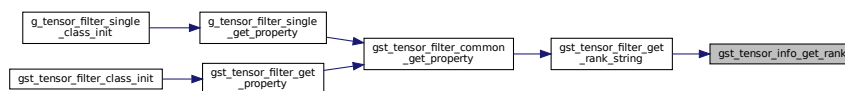
tensor rank (Minimum rank is 1 if given info is valid)

Definition at line 285 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.20 `gst_tensor_info_get_size()`

```

gsize gst_tensor_info_get_size (
    const GstTensorInfo * info )
  
```

Get data size of single tensor.

Parameters

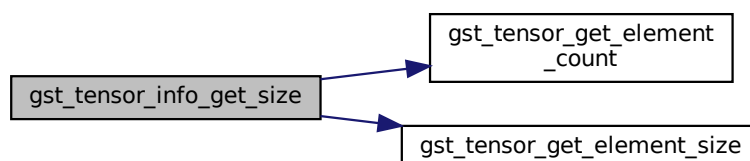
<i>info</i>	tensor info structure
-------------	-----------------------

Returns

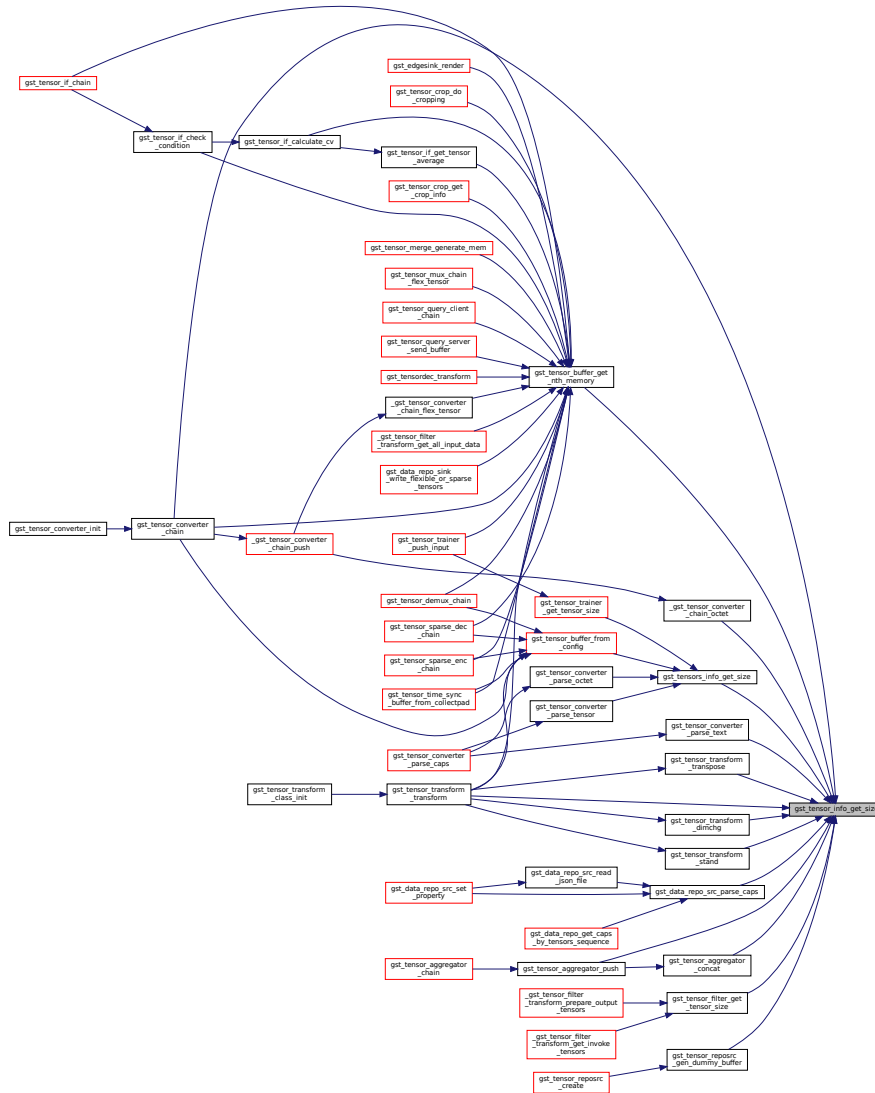
data size

Definition at line 156 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.21 gst_tensor_info_init()

```
void gst_tensor_info_init (
    GstTensorInfo * info )
```

Initialize the tensor info structure.

Parameters

<i>info</i>	tensor info structure to be initialized
-------------	---

Definition at line 121 of file nntstreamer_plugin_api_util_impl.c.

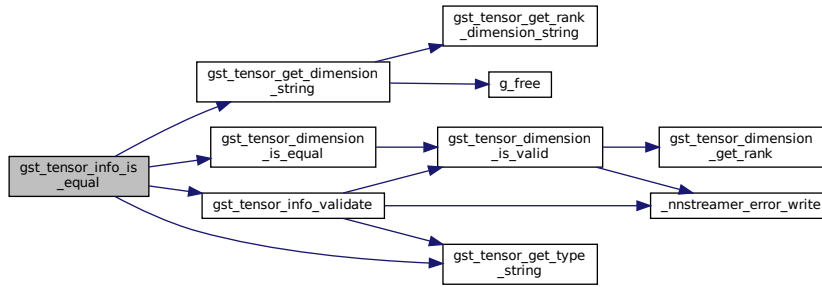
Compare tensor info.

Returns

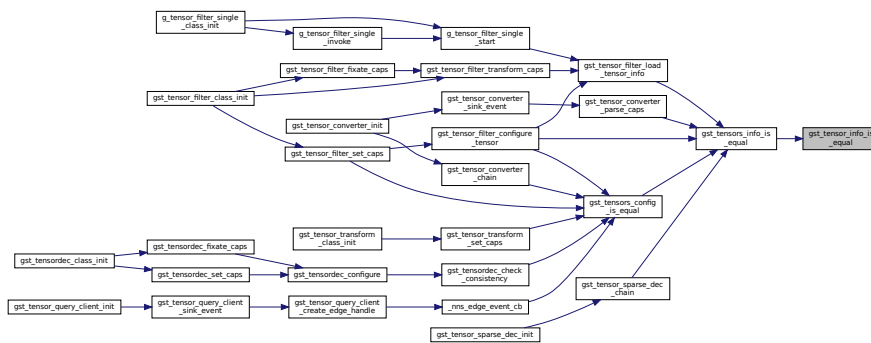
TRUE if equal, FALSE if given tensor infos are invalid or not equal.

Definition at line 197 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.23 gst_tensor_info_validate()

```

gboolean gst_tensor_info_validate (
    const GstTensorInfo * info )
    
```

Check the tensor info is valid.

Parameters

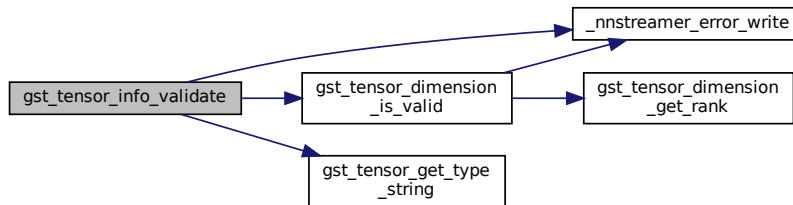
<i>info</i>	tensor info structure
-------------	-----------------------

Returns

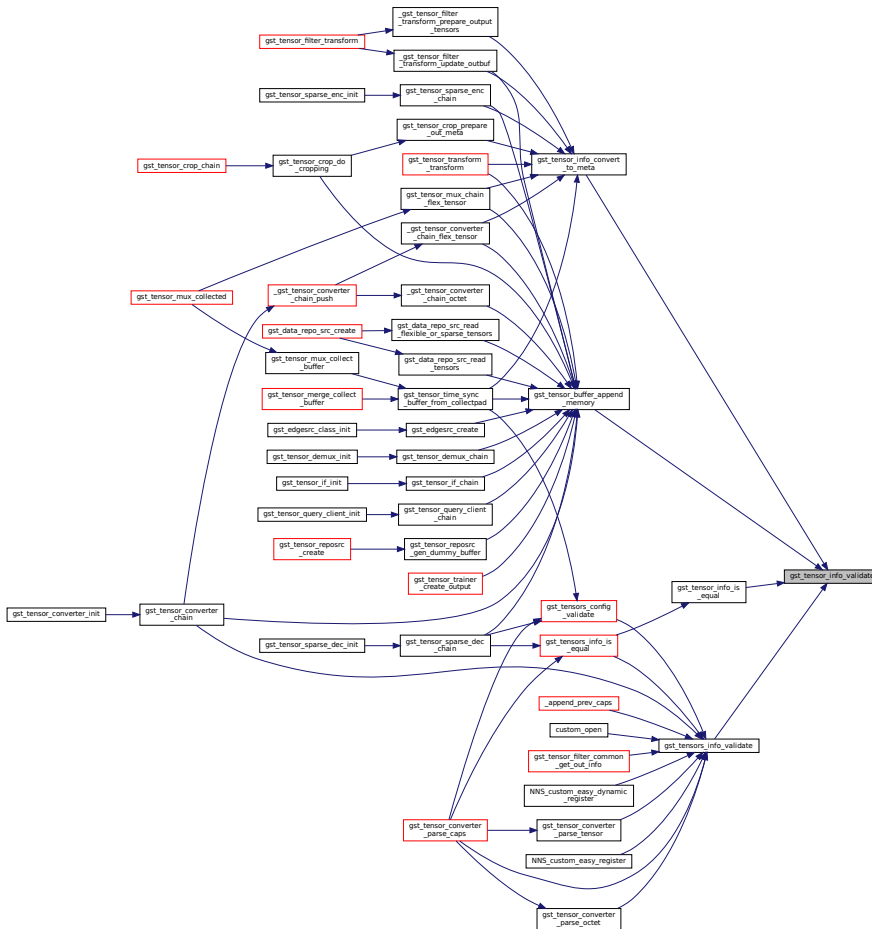
TRUE if info is valid

Definition at line 174 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.24 `gst_tensor_meta_info_convert()`

```
gboolean gst_tensor_meta_info_convert (
    GstTensorMetaInfo * meta,
    GstTensorInfo * info )
```

Convert `GstTensorMetaInfo` structure to `GstTensorInfo`.

Parameters

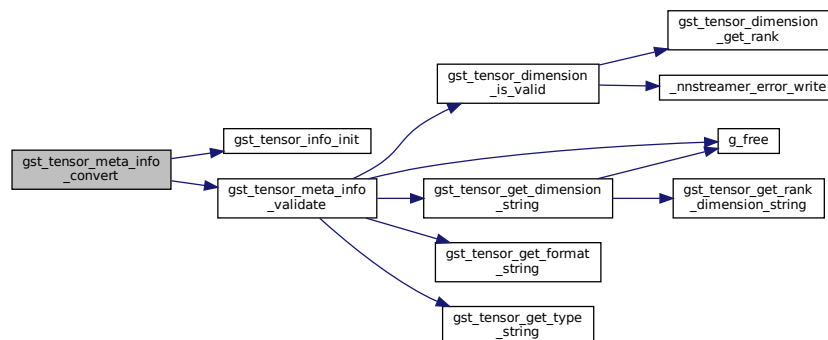
in	<i>meta</i>	tensor meta structure to be converted
out	<i>info</i>	<code>GstTensorInfo</code> to be filled

Returns

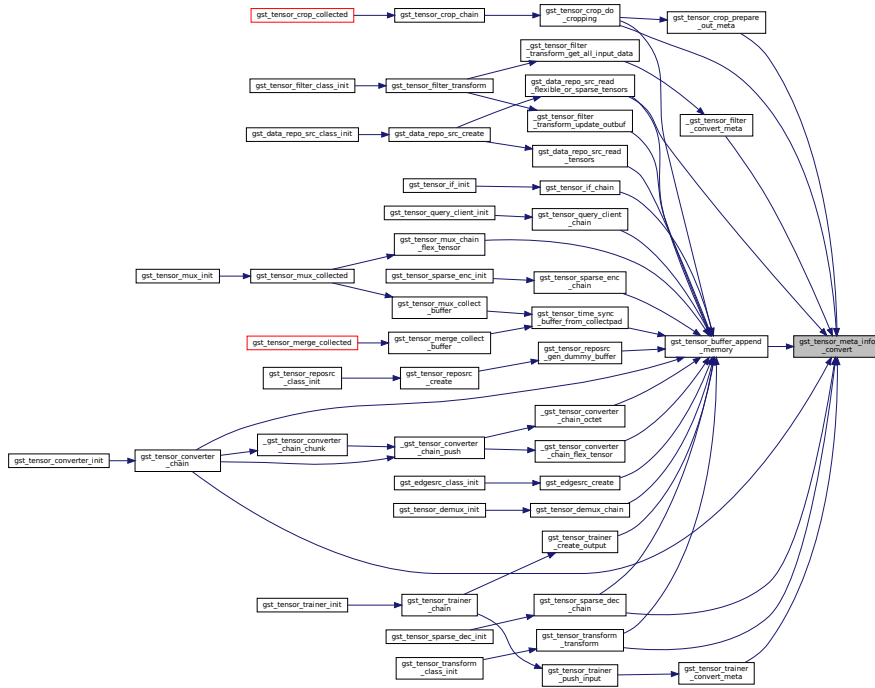
TRUE if successfully set the info

Definition at line 1562 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.25 gst_tensor_meta_info_get_data_size()

```
gsize gst_tensor_meta_info_get_data_size (
    GstTensorMetaInfo * meta )
```

Get the data size calculated from tensor meta.

Parameters

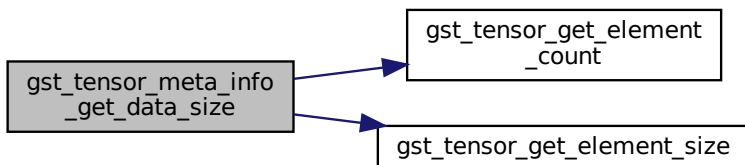
in	<i>meta</i>	tensor meta structure
----	-------------	-----------------------

Returns

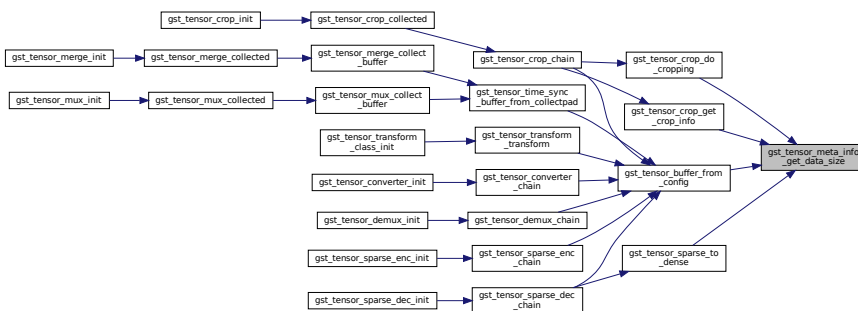
The data size for meta info (0 if meta is invalid)

Definition at line 1477 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.26 gst_tensor_meta_info_get_header_size()

```

gsize gst_tensor_meta_info_get_header_size (
    GstTensorMetaInfo * meta )
  
```

Get the header size to handle a tensor meta.

Parameters

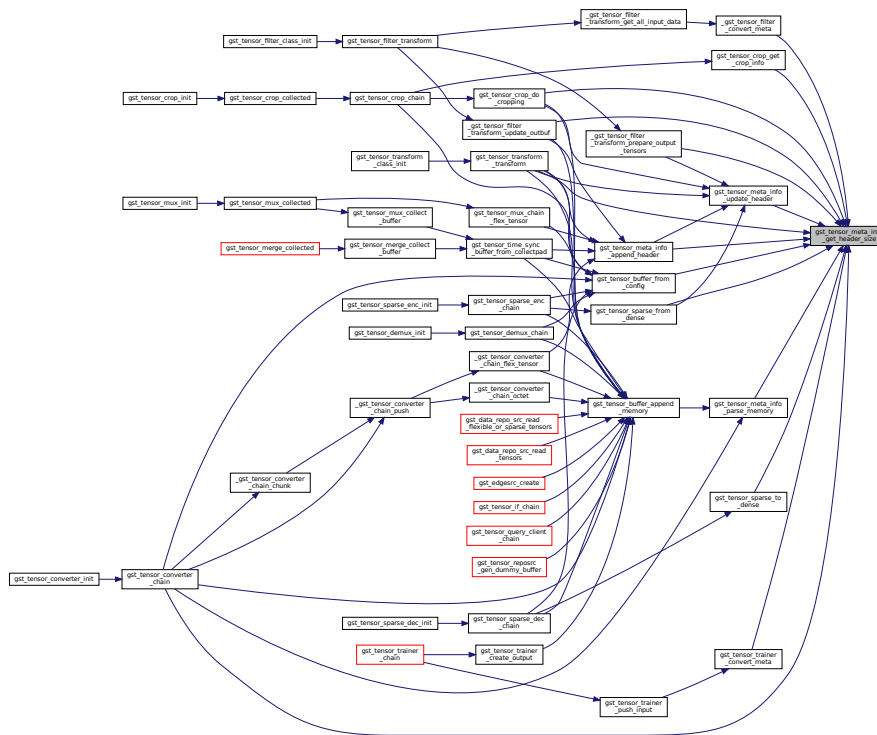
in	<i>meta</i>	tensor meta structure
----	-------------	-----------------------

Returns

Header size for meta info (0 if meta is invalid)

Definition at line 1456 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



9.78.3.27 gst_tensor_meta_info_get_version()

```
void gst_tensor_meta_info_get_version (
    GstTensorMetaInfo * meta,
    guint * major,
    guint * minor )
```

Get the version of tensor meta.

Parameters

in	<i>meta</i>	tensor meta structure
out	<i>major</i>	pointer to get the major version number
out	<i>minor</i>	pointer to get the minor version number

Definition at line 1393 of file nnstreamer_plugin_api_util_impl.c.

9.78.3.28 gst_tensor_meta_info_init()

```
void gst_tensor_meta_info_init (
```

```
GstTensorMetaInfo * meta )
```

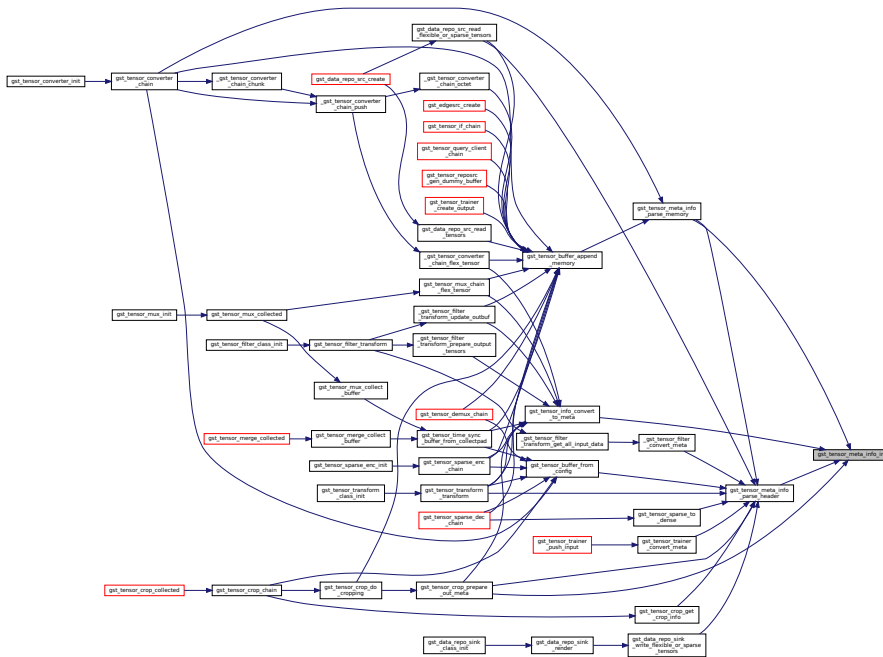
Initialize the tensor meta info structure.

Parameters

in, out	<i>meta</i>	tensor meta structure to be initialized
---------	-------------	---

Definition at line 1372 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



9.78.3.29 `gst_tensor_meta_info_parse_header()`

```
gboolean gst_tensor_meta_info_parse_header (
    GstTensorMetaInfo * meta,
    gpointer header )
```

Parse header and fill the tensor meta.

Parameters

out	<i>meta</i>	tensor meta structure to be filled
in	<i>header</i>	pointer to header to be parsed

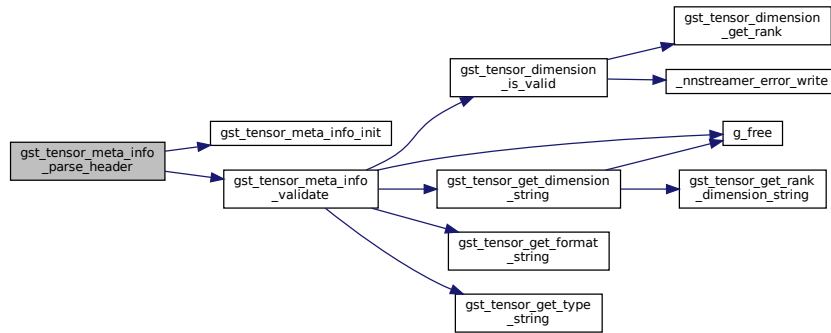
Returns

TRUE if successfully set the meta

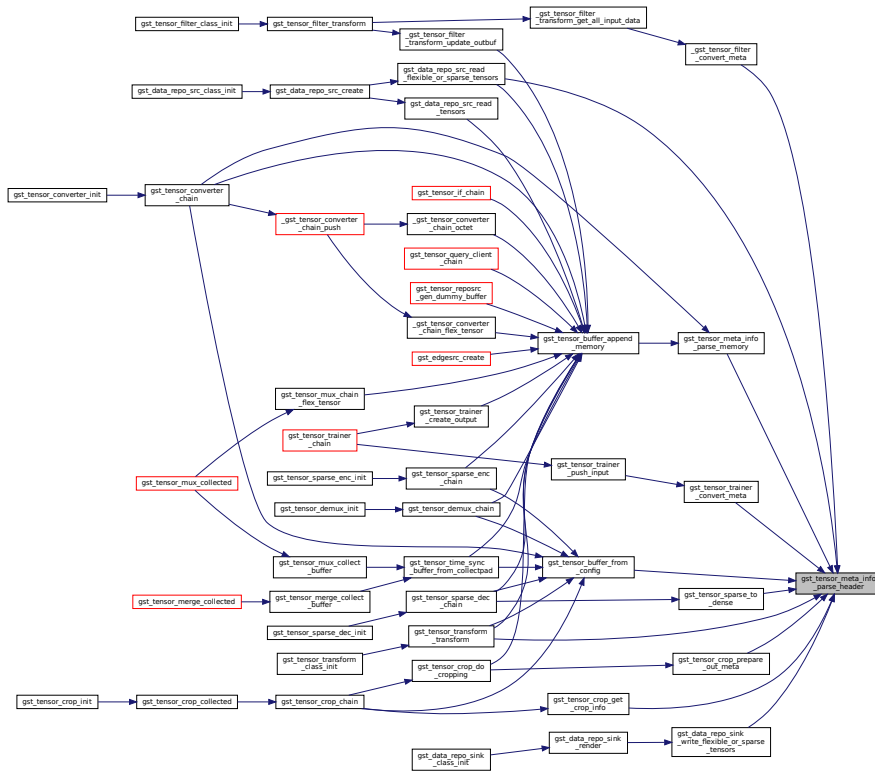
Todo update meta info for each version

Definition at line 1527 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.31 `gst_tensor_meta_info_validate()`

```
gboolean gst_tensor_meta_info_validate (
    GstTensorMetaInfo * meta )
```

Check the meta info is valid.

Parameters

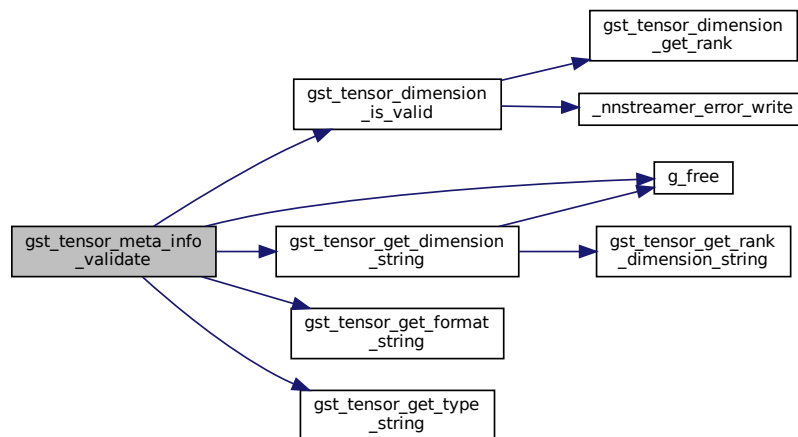
in	<i>meta</i>	tensor meta structure
----	-------------	-----------------------

Returns

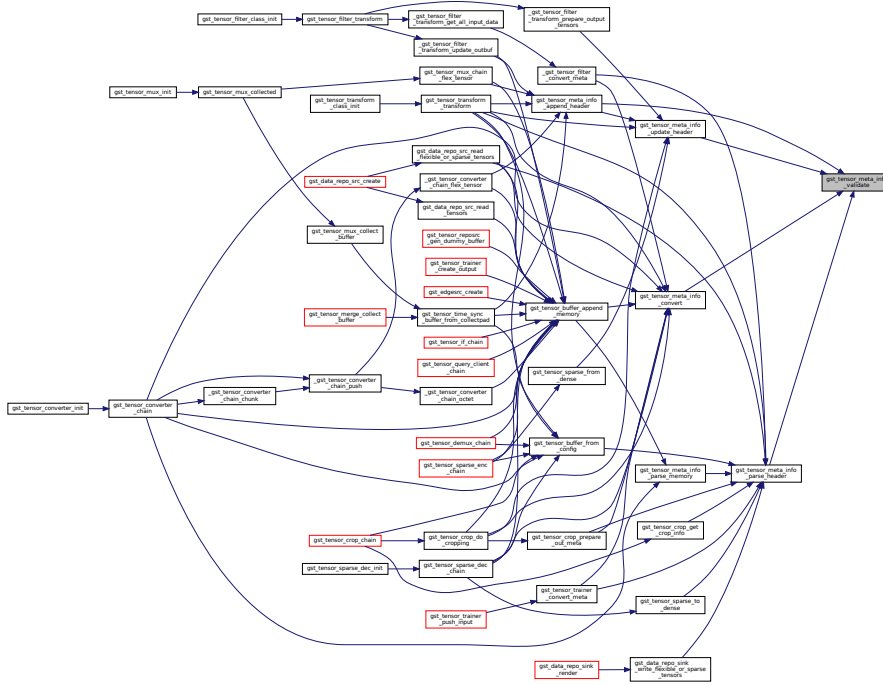
TRUE if given meta is valid

Definition at line 1414 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.32 gst_tensor_parse_dimension()

```
guint gst_tensor_parse_dimension (
    const gchar * dimstr,
    tensor_dim dim )
```

Parse tensor dimension parameter string.

Returns

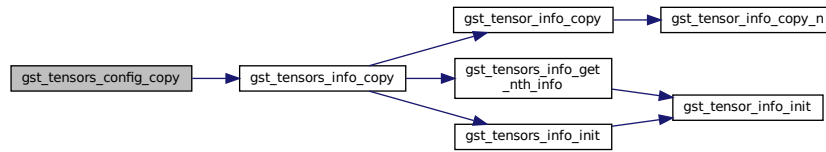
The Rank. 0 if error.

Parameters

<i>dimstr</i>	The dimension string in the format of d1:...:d16, d1:d2:d3, d1:d2, or d1, where dN is a positive integer and d1 is the innermost dimension; i.e., dim[d16]...[d1];
<i>dim</i>	dimension to be filled.

Definition at line 1040 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.34 gst_tensors_config_free()

```
void gst_tensors_config_free (
    GstTensorsConfig * config )
```

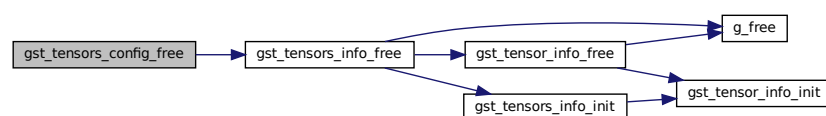
Free allocated data in tensors config structure.

Parameters

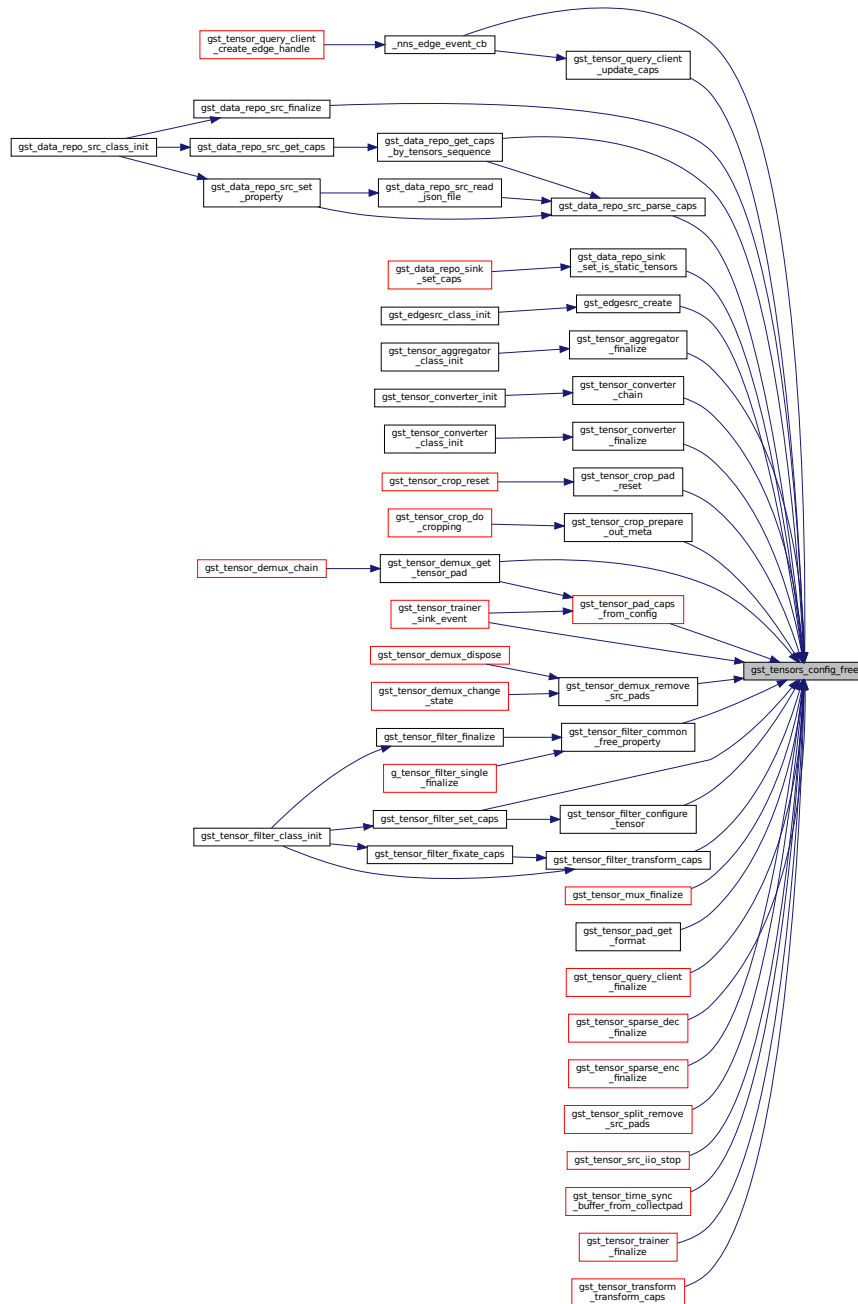
<i>config</i>	tensors config structure
---------------	--------------------------

Definition at line 845 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.35 `gst_tensors_config_init()`

```
void gst_tensors_config_init (
    GstTensorsConfig * config )
```

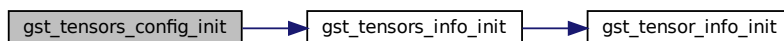
Initialize the tensors config info structure (for other/tensors)

Parameters

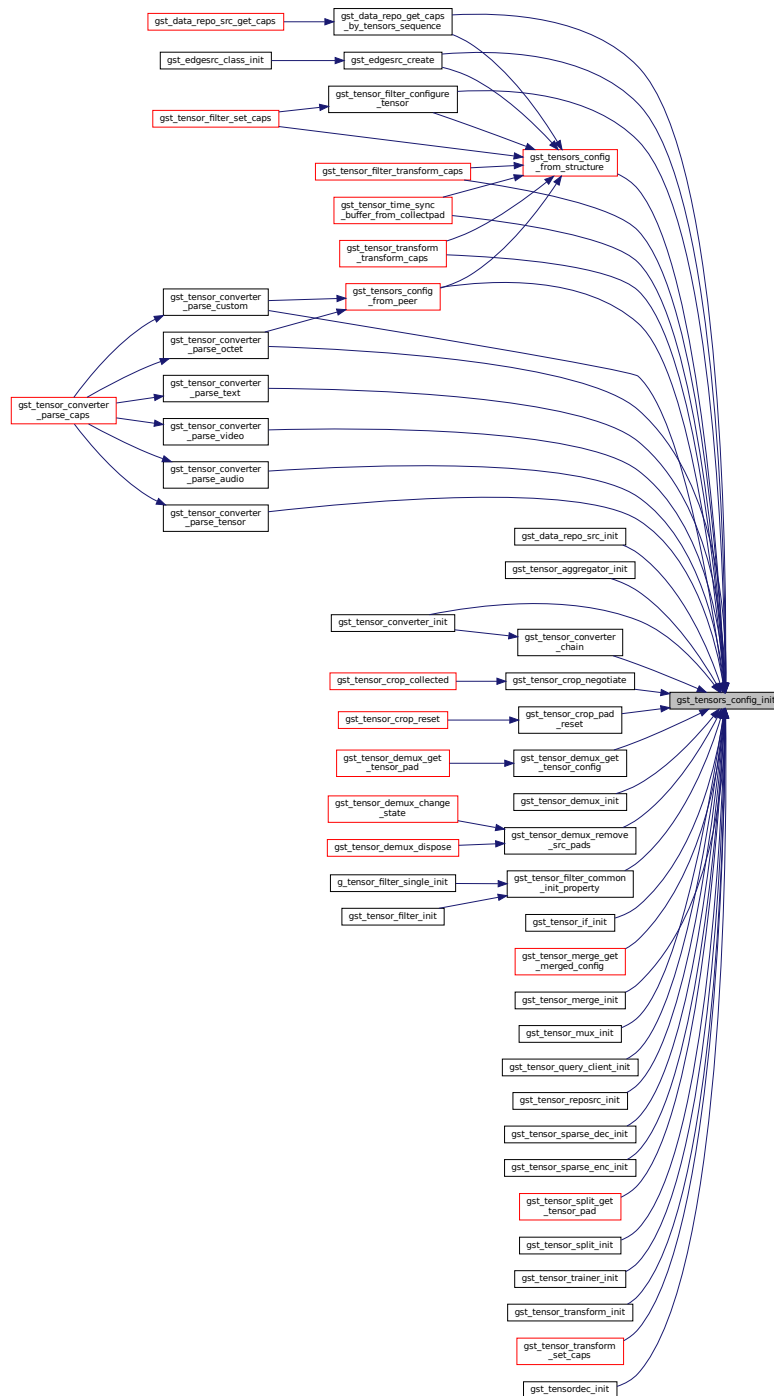
<i>config</i>	tensors config structure to be initialized
---------------	--

Definition at line 830 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.36 `gst_tensors_config_is_equal()`

```
gboolean gst_tensors_config_is_equal (
    const GstTensorsConfig * c1,
    const GstTensorsConfig * c2 )
```


Compare tensor config info (for other/tensors)

Parameters

<i>TRUE</i>	if equal
-------------	----------

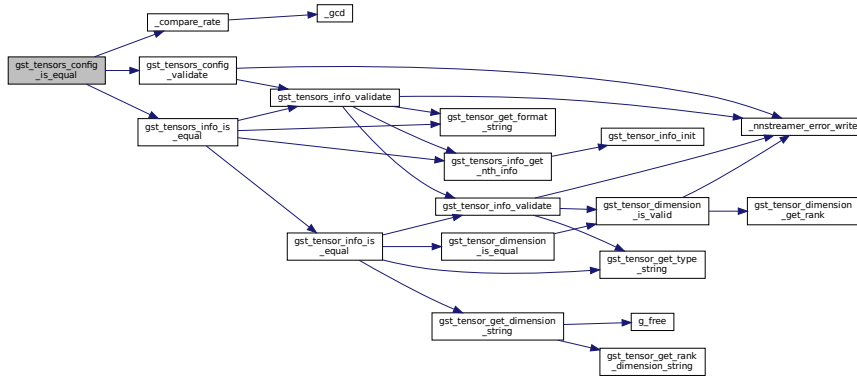
Compare tensor config info (for other/tensors)

Parameters

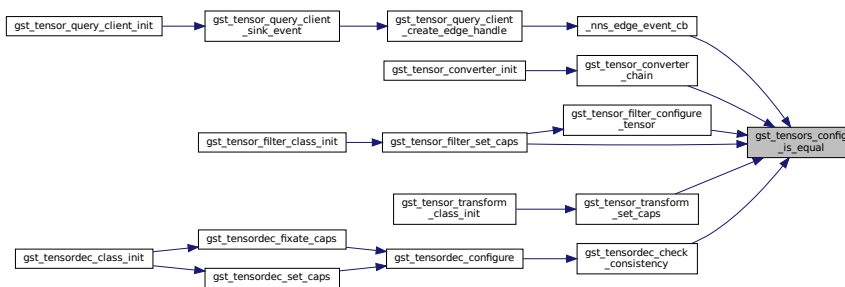
<i>TRUE</i>	if equal
-------------	----------

Definition at line 881 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.37 `gst_tensors_config_to_string()`

```

gchar* gst_tensors_config_to_string (
    const GstTensorsConfig * config )
    
```

Tensor config represented as a string. Caller should free it.

Parameters

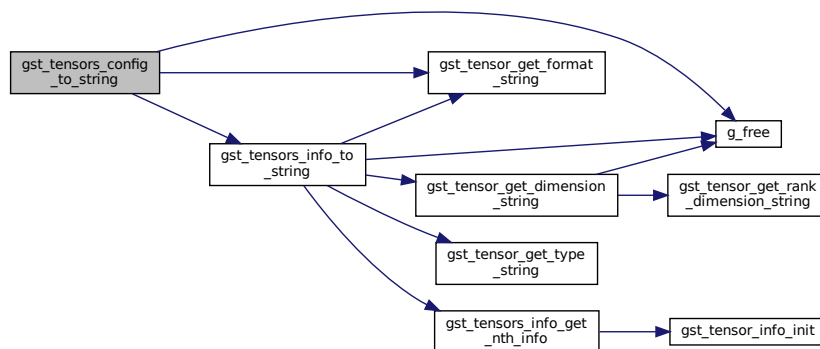
<i>config</i>	tensor config structure
---------------	-------------------------

Returns

The newly allocated string representing the config. Free after use.

Definition at line 920 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.38 gst_tensors_config_validate()

```

gboolean gst_tensors_config_validate (
    const GstTensorsConfig * config )
  
```

Check the tensors are all configured (for other/tensors)

Parameters

<i>config</i>	tensor config structure
---------------	-------------------------

Returns

TRUE if configured

Check the tensors are all configured (for other/tensors)

Parameters

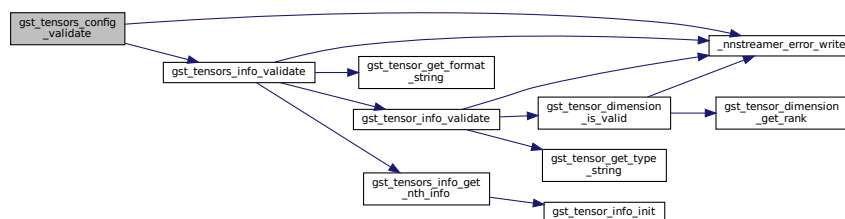
<i>config</i>	tensor config structure
---------------	-------------------------

Returns

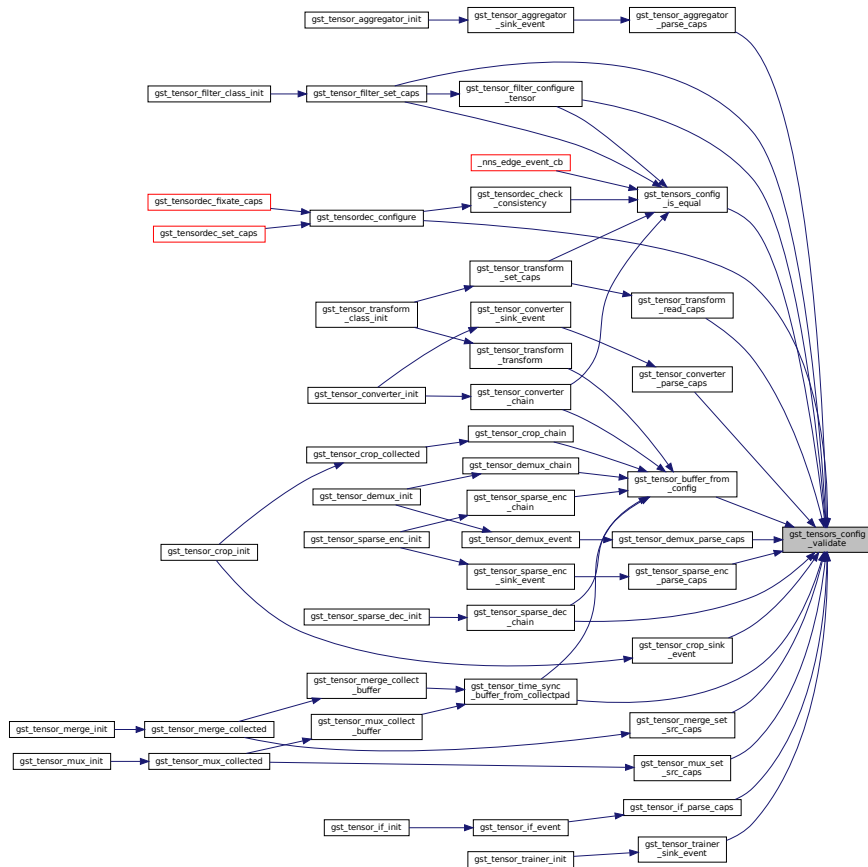
TRUE if configured

Definition at line 858 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.39 `gst_tensors_info_copy()`

```
void gst_tensors_info_copy (
    GstTensorsInfo * dest,
    const GstTensorsInfo * src )
```

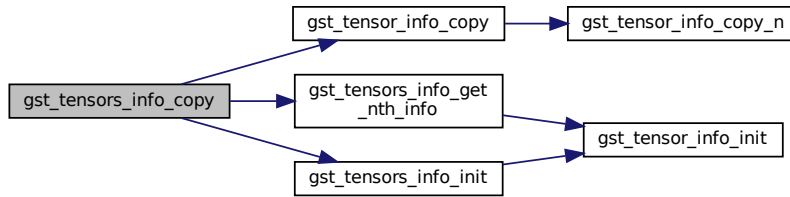
Copy tensor info.

Note

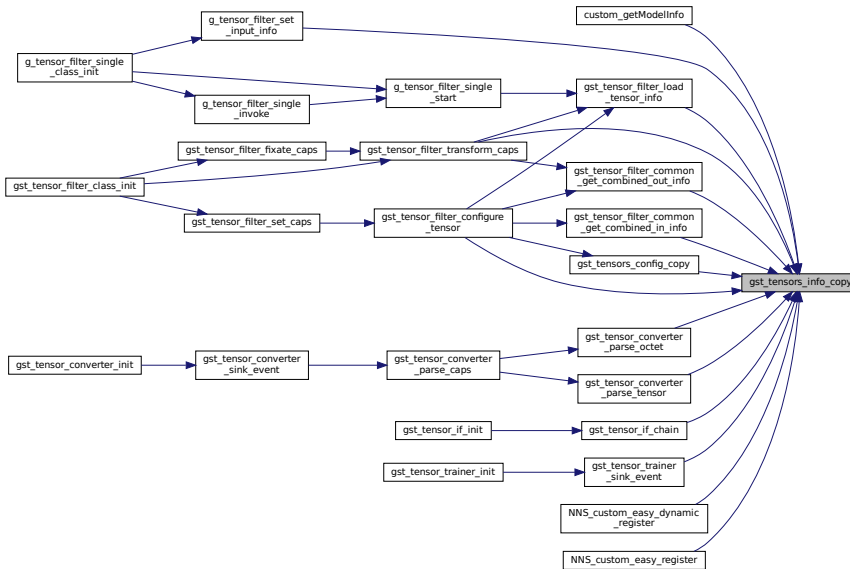
Copied info should be freed with [gst_tensors_info_free\(\)](#)

Definition at line 502 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.40 gst_tensors_info_free()

```

void gst_tensors_info_free (
    GstTensorsInfo * info )
    
```

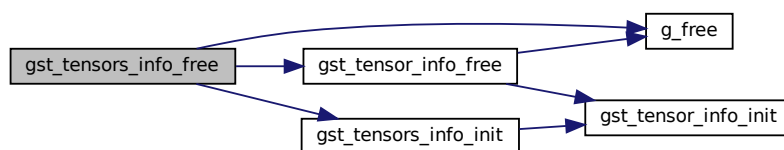
Free allocated data in tensors info structure.

Parameters

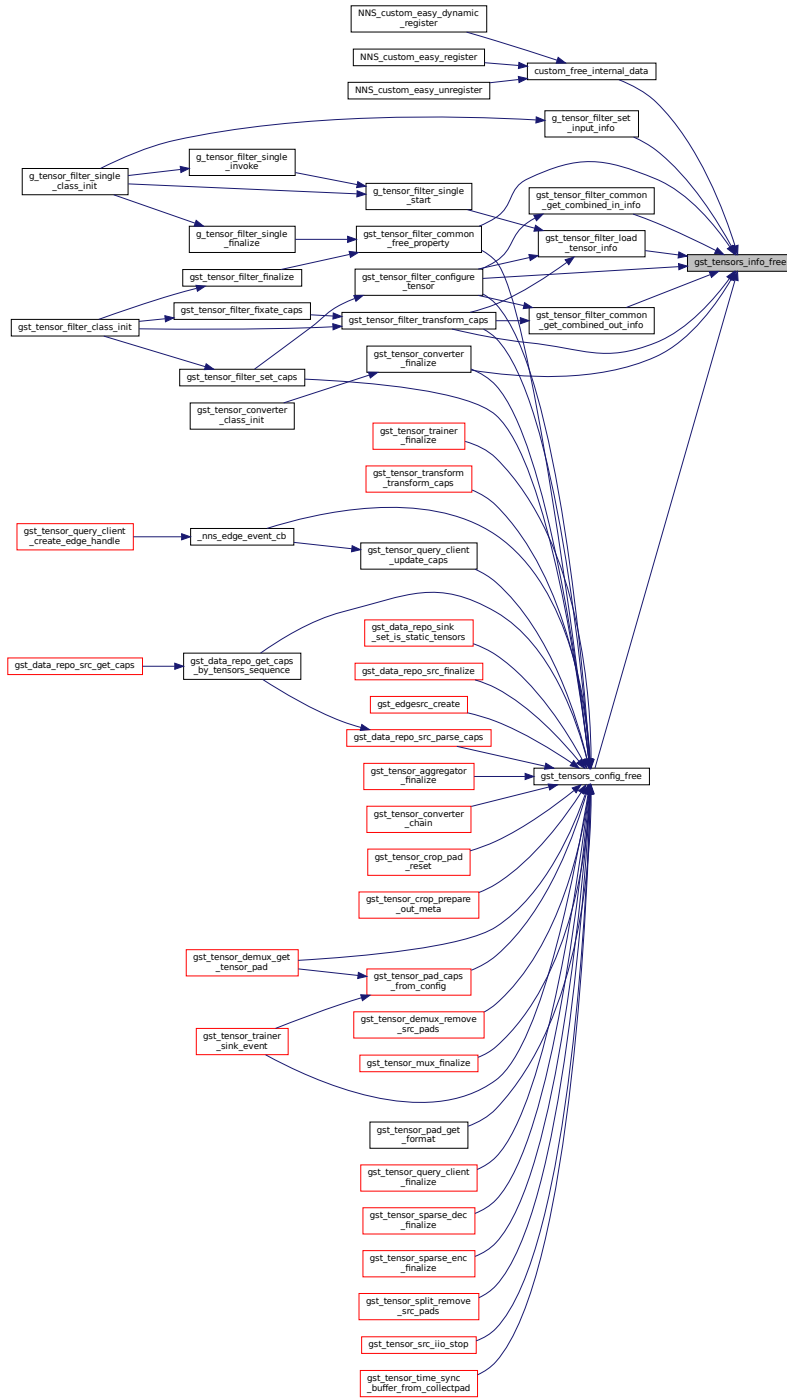
<i>info</i>	tensors info structure
-------------	------------------------

Definition at line 347 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.41 gst_tensors_info_get_dimensions_string()

```
gchar* gst_tensors_info_get_dimensions_string (
    const GstTensorsInfo * info )
```

Get the string of dimensions in tensors info.

Parameters

<i>info</i>	tensors info structure
-------------	------------------------

Returns

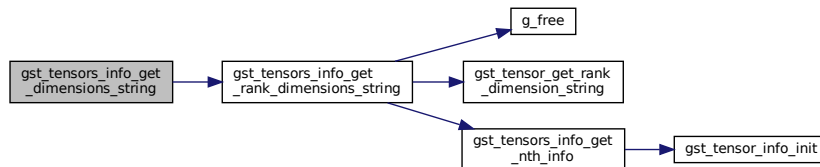
string of dimensions in tensors info (NULL if the number of tensors is 0)

Note

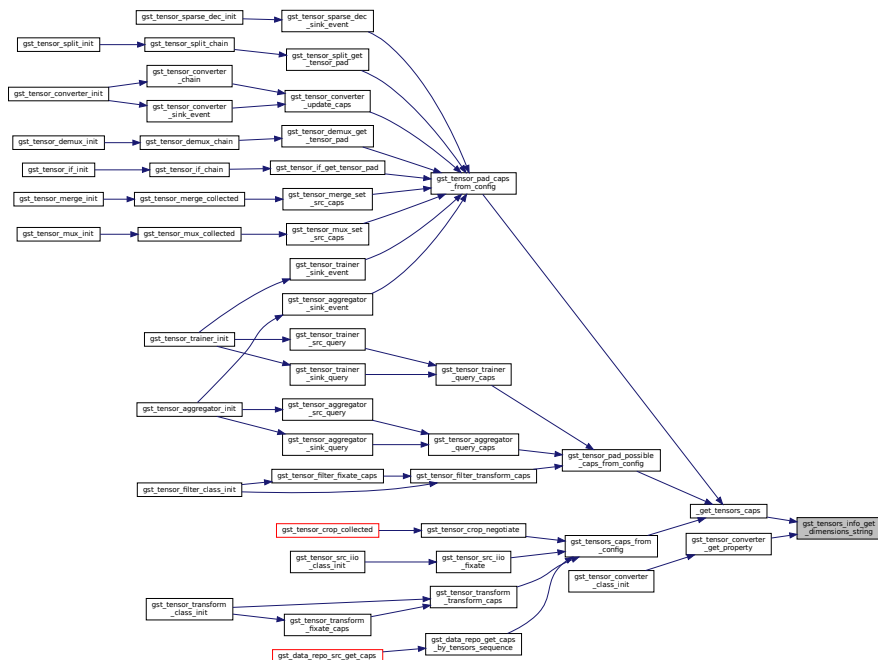
The returned value should be freed with [g_free\(\)](#)

Definition at line 661 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.42 `gst_tensors_info_get_names_string()`

```
gchar* gst_tensors_info_get_names_string (
    const GstTensorsInfo * info )
```

Get the string of tensor names in tensors info.

Parameters

<i>info</i>	tensors info structure
-------------	------------------------

Returns

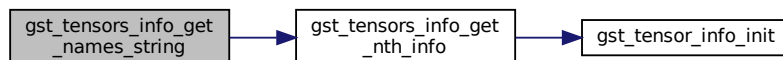
string of names in tensors info (NULL if the number of tensors is 0)

Note

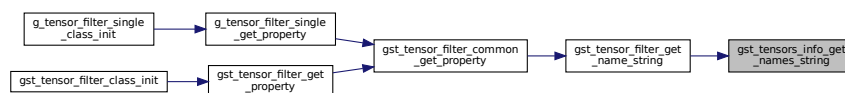
The returned value should be freed with `g_free()`

Definition at line 749 of file `nntstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.43 `gst_tensors_info_get_nth_info()`

```
GstTensorInfo* gst_tensors_info_get_nth_info (
    GstTensorsInfo * info,
    guint index )
```

Get the pointer of nth tensor information.

Parameters

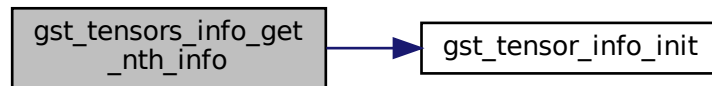
<i>info</i>	tensors info structure
<i>index</i>	the index of tensor to be fetched

Returns

The pointer to tensor info structure

Definition at line 296 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:

**9.78.3.44 gst_tensors_info_get_rank_dimensions_string()**

```

gchar* gst_tensors_info_get_rank_dimensions_string (
    const GstTensorsInfo * info,
    const unsigned int rank )
  
```

Get the string of dimensions in tensors info and rank count.

Parameters

<i>info</i>	tensors info structure
<i>rank</i>	rank count of given tensor dimension

Returns

Formatted string of given dimension

Note

If rank count is 3, then returned string is 'd1:d2:d3'. The returned value should be freed with [g_free\(\)](#)

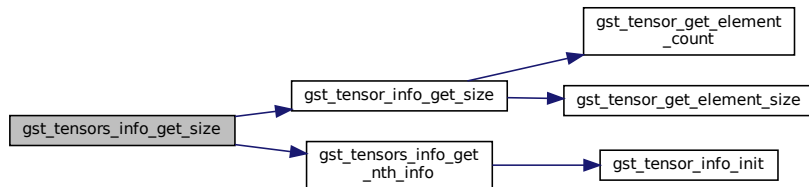
Definition at line 676 of file nnstreamer_plugin_api_util_impl.c.

Returns

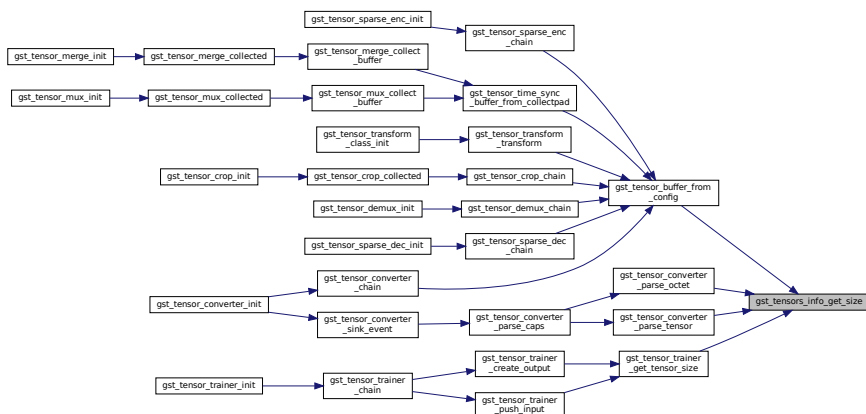
data size

Definition at line 376 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.46 `gst_tensors_info_get_types_string()`

```
gchar* gst_tensors_info_get_types_string (
    const GstTensorsInfo * info )
```

Get the string of types in tensors info.

Parameters

<i>info</i>	tensors info structure
-------------	------------------------

Returns

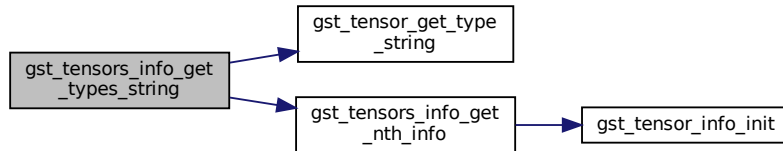
string of types in tensors info (NULL if the number of tensors is 0)

Note

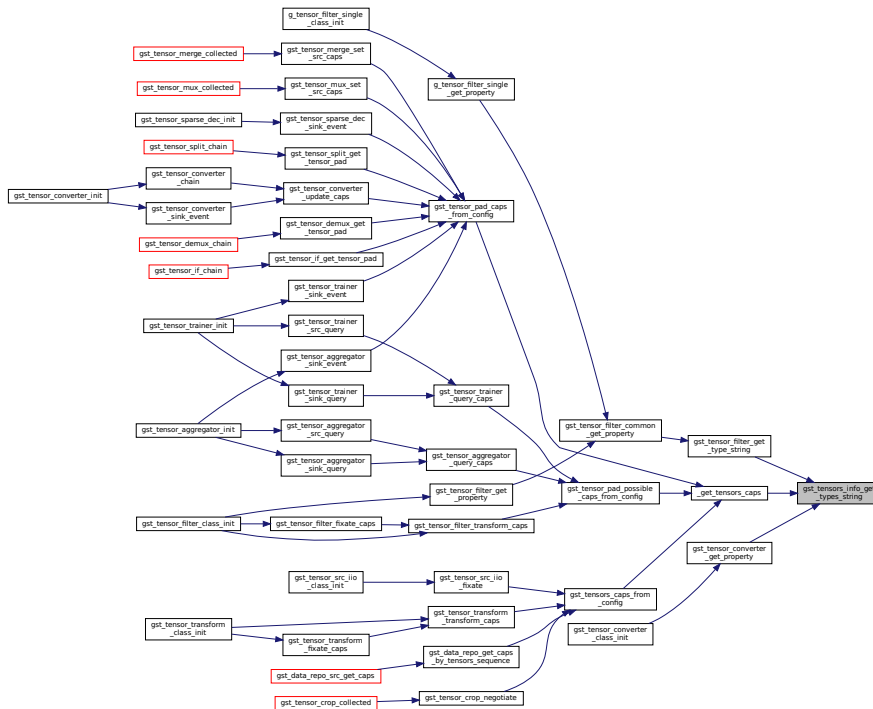
The returned value should be freed with [g_free\(\)](#)

Definition at line 714 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.47 `gst_tensors_info_init()`

```

void gst_tensors_info_init (
    GstTensorsInfo * info )
    
```

Initialize the tensors info structure.

Parameters

<i>info</i>	tensors info structure to be initialized
-------------	--

Note

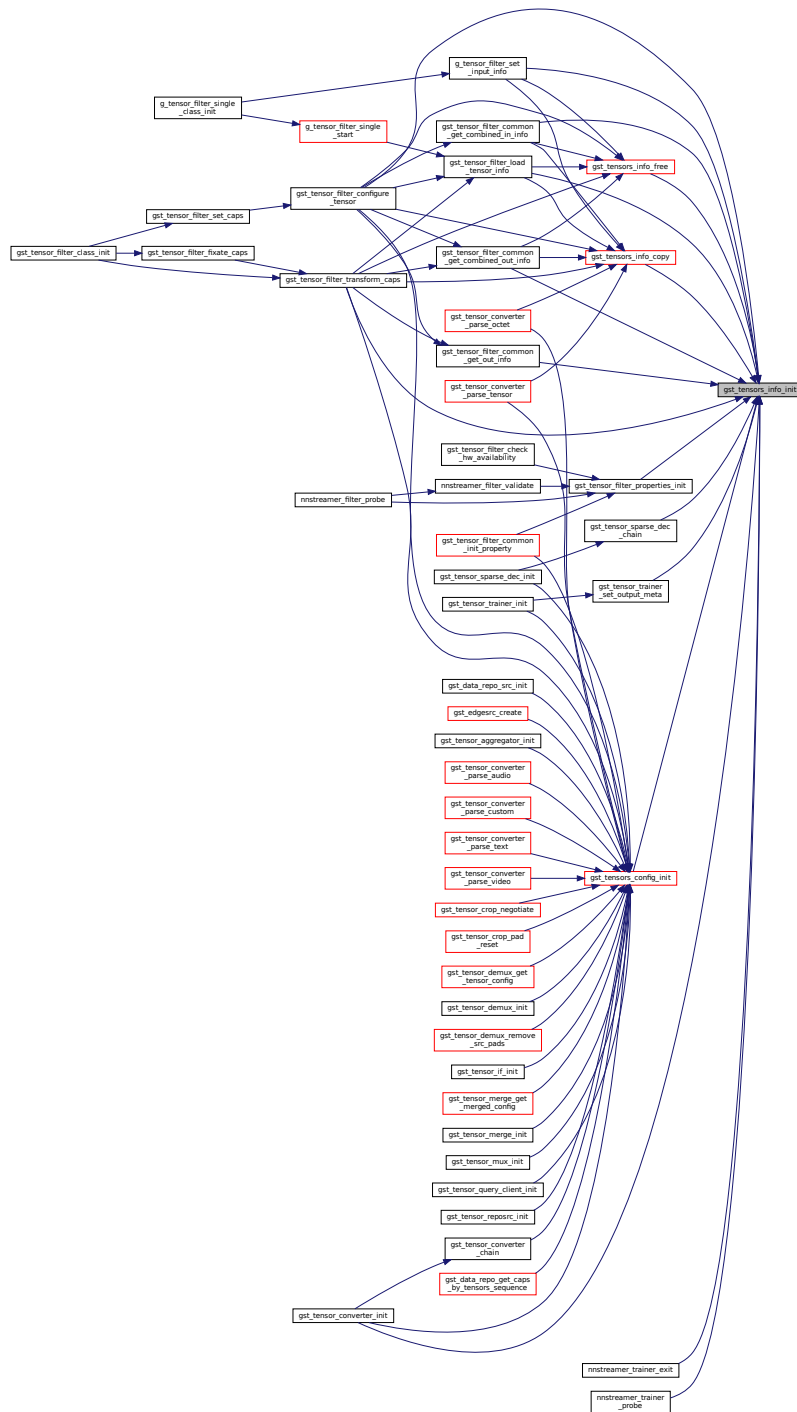
default format is static

Definition at line 325 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.48 gstreamers_info_is_equal()

```
gboolean gstreamers_info_is_equal (
    const GstreamersInfo * i1,
    const GstreamersInfo * i2 )
```


Parameters

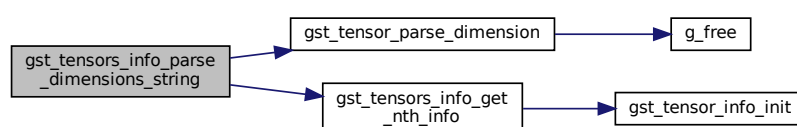
<i>info</i>	tensors info structure
<i>dim_string</i>	string of dimensions

Returns

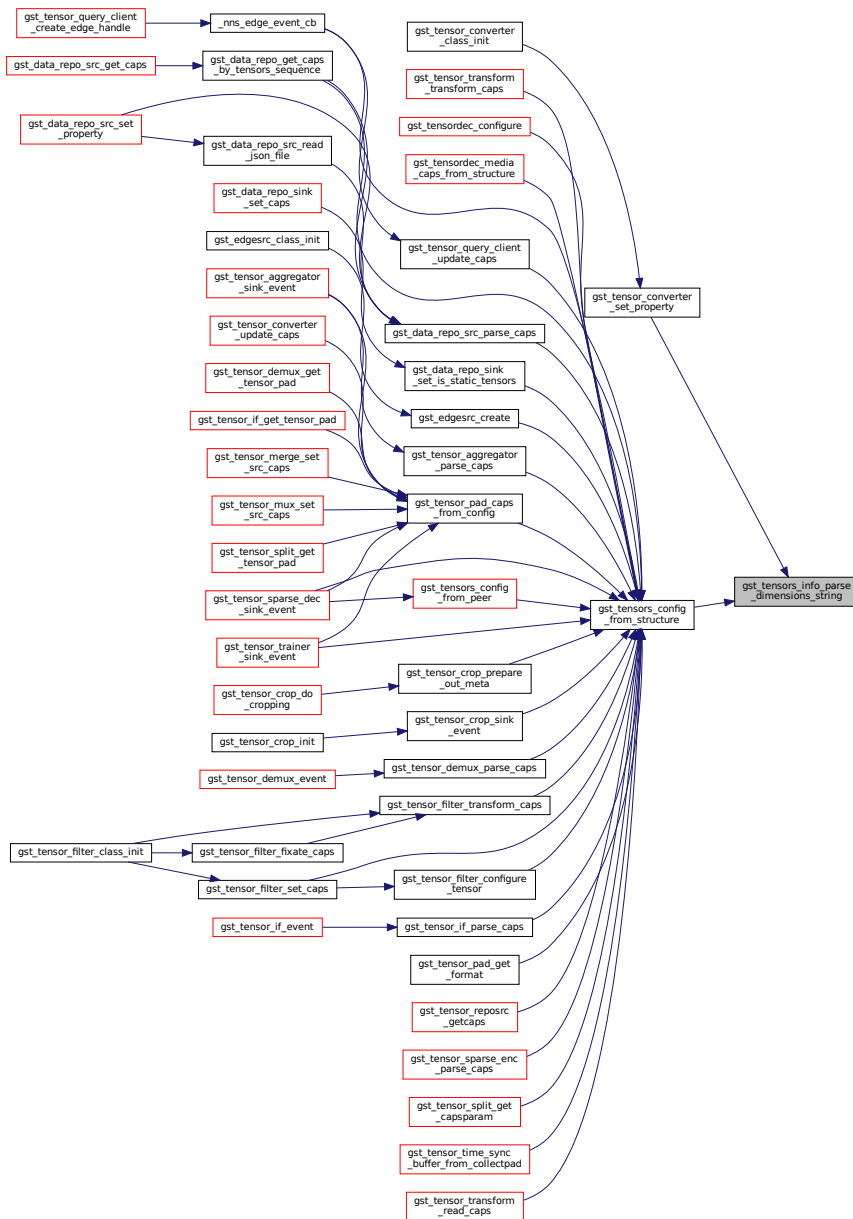
number of parsed dimensions

Definition at line 532 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.50 gst_tensors_info_parse_names_string()

```
guint gst_tensors_info_parse_names_string (
    GstTensorsInfo * info,
    const gchar * name_string )
```

Parse the string of names.

Parameters

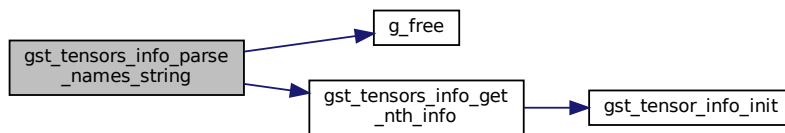
<i>info</i>	tensors info structure
<i>name_string</i>	string of names

Returns

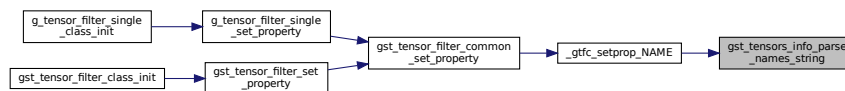
number of parsed names

Definition at line 612 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.51 gst_tensors_info_parse_types_string()

```

guint gst_tensors_info_parse_types_string (
    GstTensorsInfo * info,
    const gchar * type_string )
  
```

Parse the string of types.

Parameters

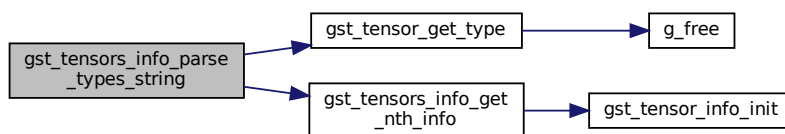
<i>info</i>	tensors info structure
<i>type_string</i>	string of types

Returns

number of parsed types

Definition at line 572 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Parameters

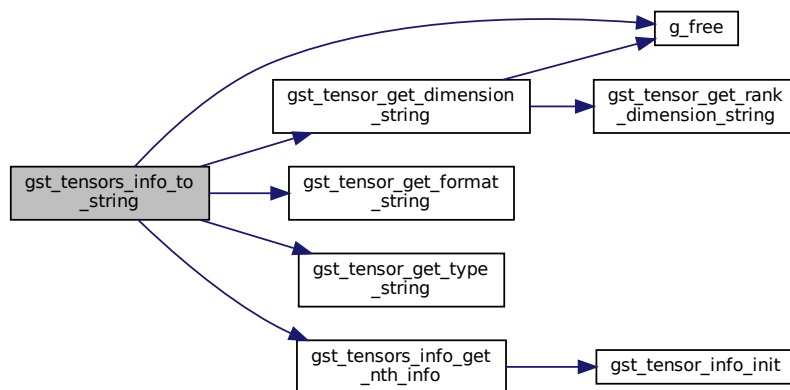
<i>info</i>	GstTensorsInfo structure
-------------	--

Returns

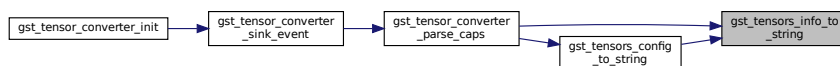
The newly allocated string representing the tensors info. Free after use.

Definition at line 783 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.53 `gst_tensors_info_validate()`

```

gboolean gst_tensors_info_validate (
    const GstTensorsInfo * info )
  
```

Check the tensors info is valid.

Parameters

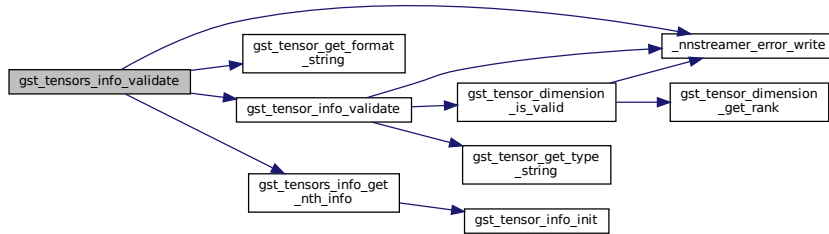
<i>info</i>	tensors info structure
-------------	------------------------

Returns

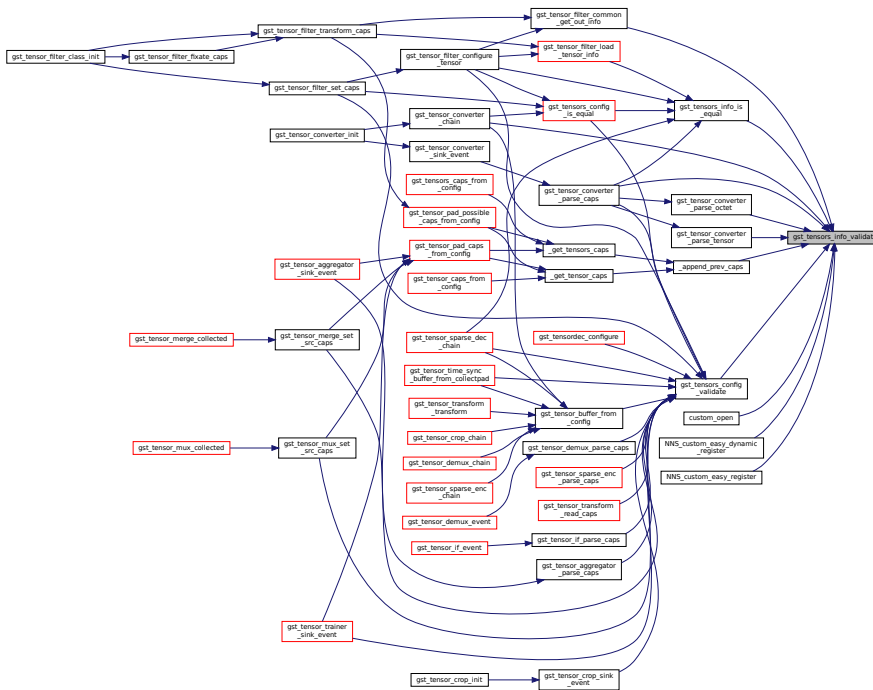
TRUE if info is valid

Definition at line 404 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.78.3.54 nnstreamer_version_fetch()

```

void nnstreamer_version_fetch (
    guint * major,
    guint * minor,
    guint * micro )
    
```

Get the version of NNStreamer (int, divided).

Parameters

out	<i>major</i>	MAJOR.minor.micro, won't set if it's null.
out	<i>minor</i>	major.MINOR.micro, won't set if it's null.
out	<i>micro</i>	major.minor.MICRO, won't set if it's null.

Definition at line 1624 of file nnstreamer_plugin_api_util_impl.c.

9.78.3.55 nnstreamer_version_string()

```
gchar* nnstreamer_version_string (
    void )
```

Get the version of NNStreamer.

Returns

Newly allocated string. The returned string should be freed with [g_free\(\)](#).

Get the version of NNStreamer.

Returns

Newly allocated string. The returned string should be freed with [g_free\(\)](#).

Definition at line 1609 of file nnstreamer_plugin_api_util_impl.c.

9.79 nnstreamer/include/nnstreamer_util.h File Reference

Optional NNStreamer utility functions for sub-plugin writers and users.

This graph shows which files directly or indirectly include this file:

**Macros**

- #define [UNUSED](#)(expr) do { (void)(expr); } while (0)
Utility to silence unused parameter warning for intentionally unused parameters (e.g., callback functions of a framework)
- #define [_g_memdup](#)(data, size) g_memdup (data, size)
g_memdup() function replaced by g_memdup2() in glib version >= 2.68

9.79.1 Detailed Description

Optional NNStreamer utility functions for sub-plugin writers and users.

NNStreamer Common Utility Header for Plug-Ins and Users Copyright (C) 2021 MyungJoo Ham myungjoo.ham@samsung.com

Date

28 Jul 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.79.2 Macro Definition Documentation

9.79.2.1 `_g_memdup`

```
#define _g_memdup(  
    data,  
    size ) g_memdup (data, size)
```

`g_memdup()` function replaced by `g_memdup2()` in glib version ≥ 2.68

Definition at line 31 of file `nnstreamer_util.h`.

9.79.2.2 `UNUSED`

```
#define UNUSED(  
    expr ) do { (void)(expr); } while (0)
```

Utility to silence unused parameter warning for intentionally unused parameters (e.g., callback functions of a framework)

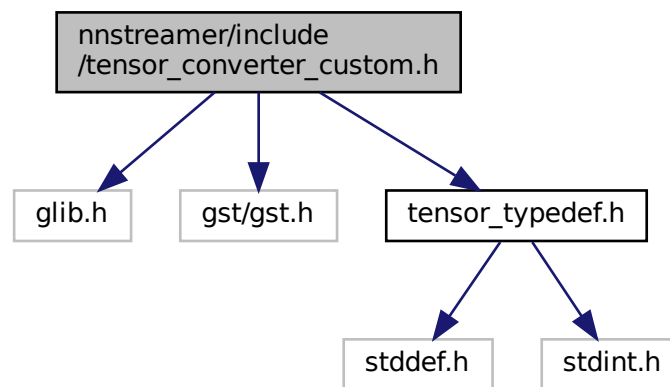
Definition at line 22 of file `nnstreamer_util.h`.

9.80 nnstreamer/include/tensor_converter_custom.h File Reference

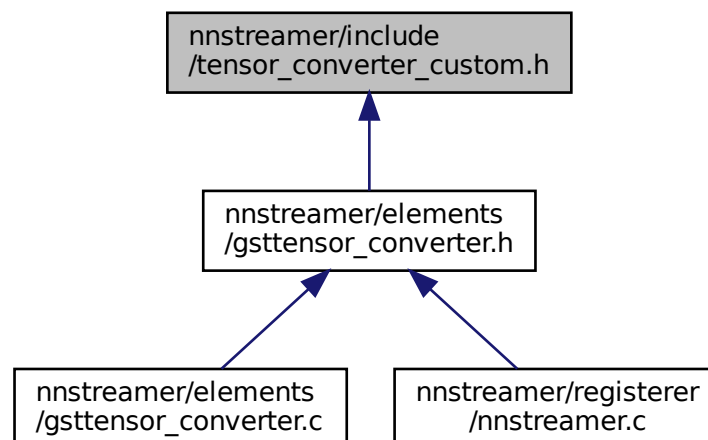
NNStreamer APIs for tensor_converter custom condition.

```
#include <glib.h>
#include <gst/gst.h>
#include "tensor_typedef.h"
```

Include dependency graph for tensor_converter_custom.h:



This graph shows which files directly or indirectly include this file:



Functions

- int `nnstreamer_converter_custom_register` (const gchar *name, `tensor_converter_custom` func, void *data)
Register the custom callback function.
- int `nnstreamer_converter_custom_unregister` (const gchar *name)
Unregister the custom callback function.

Variables

- `G_BEGIN_DECLS` typedef GstBuffer *(* `tensor_converter_custom`)(GstBuffer *in_buf, void *data, `GstTensorsConfig` *config)
Convert to tensors as customized operation.

9.80.1 Detailed Description

NNStreamer APIs for `tensor_converter` custom condition.

GStreamer/NNStreamer Tensor-Converter Copyright (C) 2021 Gichan Jang gichan2.jang@samsung.com

Date

18 Mar 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.80.2 Function Documentation

9.80.2.1 `nnstreamer_converter_custom_register()`

```
int nnstreamer_converter_custom_register (  
    const gchar * name,  
    tensor_converter_custom func,  
    void * data )
```

Register the custom callback function.

Parameters

in	<i>name</i>	The name of tensor_converter custom callback function.
in	<i>func</i>	The custom condition function body
	[in/out]	data The internal data for the function

Returns

0 if success. -ERRNO if error.

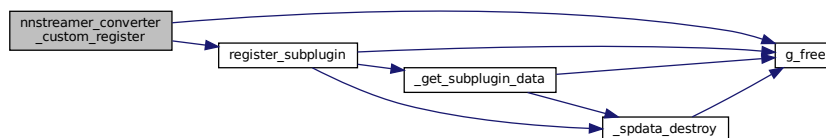
Register the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 2473 of file gsttensor_converter.c.

Here is the call graph for this function:

**9.80.2.2 nnstreamer_converter_custom_unregister()**

```
int nnstreamer_converter_custom_unregister (
    const gchar * name )
```

Unregister the custom callback function.

Parameters

in	<i>name</i>	The registered name of tensor_converter custom callback function.
----	-------------	---

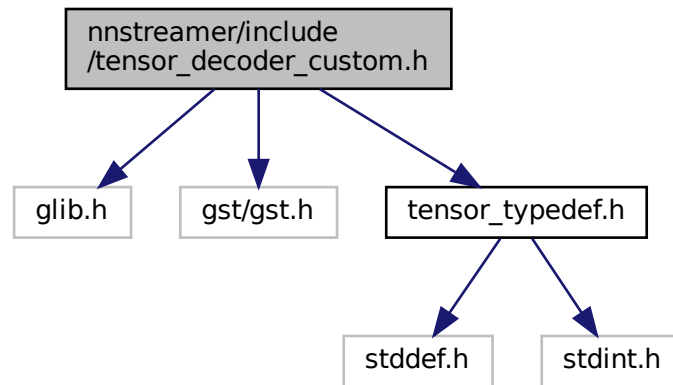
Returns

0 if success. -ERRNO if error.

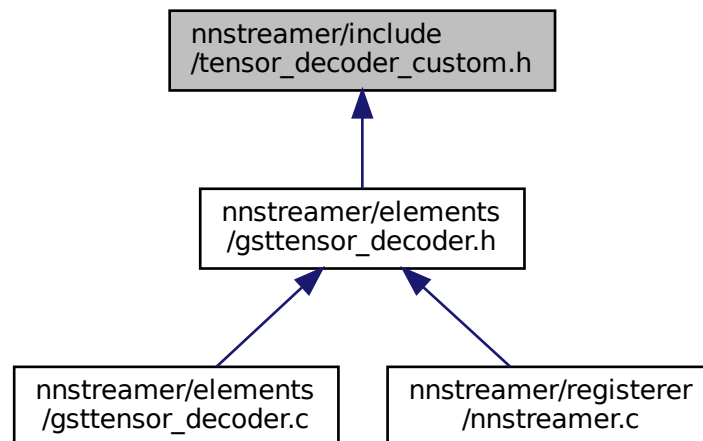
Unregister the custom callback function.


```
#include "tensor_typedef.h"
```

Include dependency graph for `tensor_decoder_custom.h`:



This graph shows which files directly or indirectly include this file:



Functions

- `int nnstreamer_decoder_custom_register` (const gchar *name, `tensor_decoder_custom` func, void *data)
Register the custom callback function.
- `int nnstreamer_decoder_custom_unregister` (const gchar *name)
Unregister the custom callback function.

Variables

- `G_BEGIN_DECLS` typedef int(* `tensor_decoder_custom`)(const `GstTensorMemory` *input, const `GstTensorsConfig` *config, void *data, `GstBuffer` *out_buf)

Decode from tensors to media as customized operation.

9.81.1 Detailed Description

NNStreamer APIs for tensor_decoder custom condition.

GStreamer/NNStreamer Tensor-Decoder Copyright (C) 2021 Gichan Jang gichan2.jang@samsung.com

Date

22 Mar 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.81.2 Function Documentation

9.81.2.1 nnstreamer_decoder_custom_register()

```
int nnstreamer_decoder_custom_register (
    const gchar * name,
    tensor_decoder_custom func,
    void * data )
```

Register the custom callback function.

Parameters

in	<i>name</i>	The name of tensor_decoder custom callback function.
in	<i>func</i>	The custom condition function body
	<i>[in/out]</i>	data The internal data for the function

Returns

0 if success. -ERRNO if error.

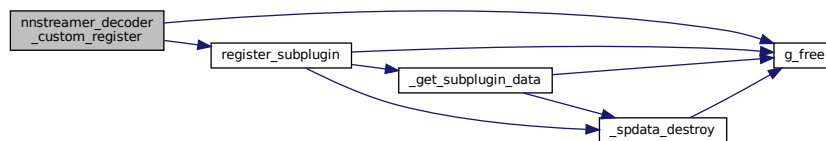
Register the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 972 of file gstreamer_decoder.c.

Here is the call graph for this function:

**9.81.2.2 nntstreamer_decoder_custom_unregister()**

```
int nntstreamer_decoder_custom_unregister (
    const gchar * name )
```

Unregister the custom callback function.

Parameters

<code>in</code>	<i>name</i>	The registered name of tensor_decoder custom callback function.
-----------------	-------------	---

Returns

0 if success. -ERRNO if error.

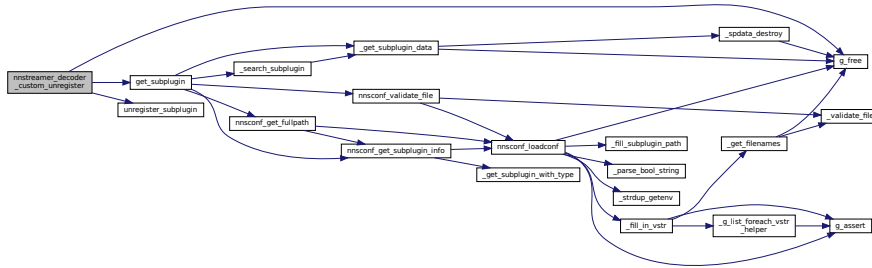
Unregister the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 998 of file gstreamer_decoder.c.

Here is the call graph for this function:



9.81.3 Variable Documentation

9.81.3.1 tensor_decoder_custom

G_BEGIN_DECLS typedef int(* tensor_decoder_custom) (const GstTensorMemory *input, const GstTensorsConfig *config, void *data, GstBuffer *out_buf)

Decode from tensors to media as customized operation.

Parameters

in	<i>input</i>	the input memory containing tensors
in	<i>config</i>	input tensors config
in	<i>data</i>	private data for the callback
out	<i>output</i>	buffer filled by user

Returns

0 if success. -ERRNO if error.

Definition at line 32 of file tensor_decoder_custom.h.

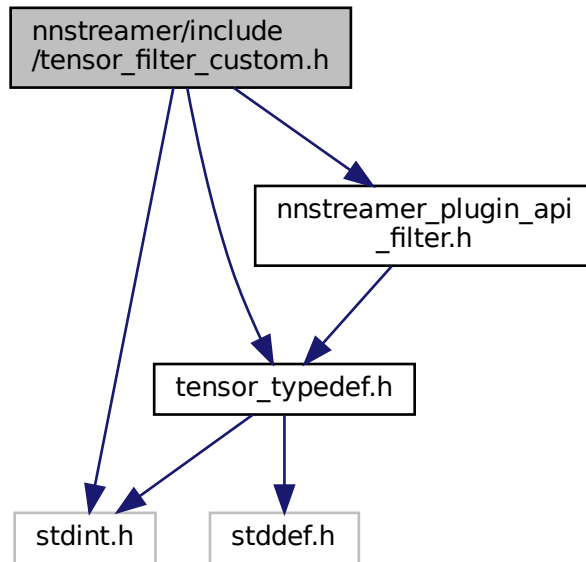
9.82 nnstreamer/include/tensor_filter_custom.h File Reference

Custom tensor post-processing interface for NNStreamer suite for post-processing code developers.

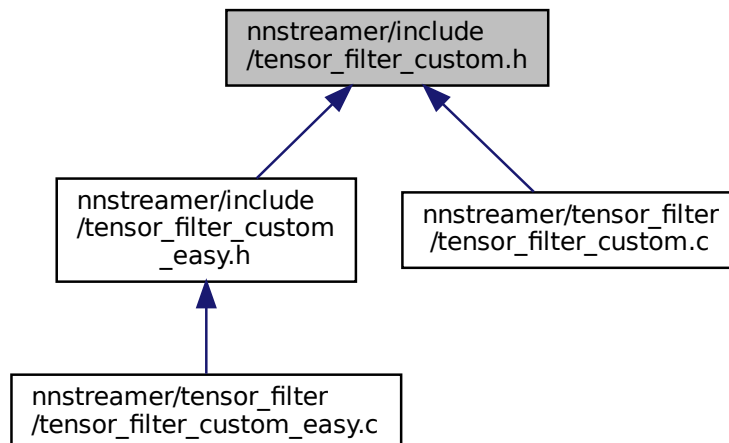
```
#include <stdint.h>
#include "tensor_typedef.h"
```

```
#include "nnstreamer_plugin_api_filter.h"
```

Include dependency graph for `tensor_filter_custom.h`:



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_NNStreamer_custom_class`](#)
Custom Filter Class.

Typedefs

- typedef void [*\(* NNS_custom_init_func\)](#) (const [GstTensorFilterProperties](#) *prop)

A function that is called before calling other functions.
- typedef void [\(* NNS_custom_exit_func\)](#) (void *private_data, const [GstTensorFilterProperties](#) *prop)

A function that is called after calling other functions, when it's ready to close.
- typedef int [\(* NNS_custom_get_input_dimension\)](#) (void *private_data, const [GstTensorFilterProperties](#) *prop, [GstTensorInfo](#) *info)

Get input tensor type.
- typedef int [\(* NNS_custom_get_output_dimension\)](#) (void *private_data, const [GstTensorFilterProperties](#) *prop, [GstTensorInfo](#) *info)

Get output tensor type.
- typedef int [\(* NNS_custom_set_input_dimension\)](#) (void *private_data, const [GstTensorFilterProperties](#) *prop, const [GstTensorInfo](#) *in_info, [GstTensorInfo](#) *out_info)

Set input dim by framework. Let custom plugin set output dim accordingly.
- typedef int [\(* NNS_custom_invoke\)](#) (void *private_data, const [GstTensorFilterProperties](#) *prop, const [GstTensorMemory](#) *input, [GstTensorMemory](#) *output)

Invoke the "main function". Without allocating output buffer. (fill in the given output buffer)
- typedef int [\(* NNS_custom_allocate_invoke\)](#) (void *private_data, const [GstTensorFilterProperties](#) *prop, const [GstTensorMemory](#) *input, [GstTensorMemory](#) *output)

Invoke the "main function". Without allocating output buffer. (fill in the given output buffer)
- typedef void [\(* NNS_custom_destroy_notify\)](#) (void *data)

It's a post-processing method about the used data pointer if it has been allocated at custom filter.
- typedef struct [_NNStreamer_custom_class](#) [NNStreamer_custom_class](#)

Variables

- [NNStreamer_custom_class](#) * [NNStreamer_custom](#)

A custom filter MUST define `NNStreamer_custom`. This object represents the custom filter itself.

9.82.1 Detailed Description

Custom tensor post-processing interface for NNStreamer suite for post-processing code developers.

GStreamer Tensor_Filter, Customized Module Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

01 Jun 2018

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

How To for NNdevelopers:

1. Define struct, "NNStreamer_custom", with the functions defined.
2. Compile as a shared object. (.so in Linux)
3. Use NNStreamer (tensor_filter framework=custom, model=FILEPATH_OF_YOUR_SO.so, ...)

To Packagers:

This file is to be packaged as "devel" package for NN developers.

9.82.2 Typedef Documentation

9.82.2.1 NNS_custom_allocate_invoke

```
typedef int (* NNS_custom_allocate_invoke) (void *private_data, const GstTensorFilterProperties
*prop, const GstTensorMemory *input, GstTensorMemory *output)
```

Invoke the "main function". Without allocating output buffer. (fill in the given output buffer)

Parameters

in	<i>private_data</i>	The pointer returned by NNStreamer_custom_init.
in	<i>prop</i>	GstTensorFilter's property values. Do not change its values.
in	<i>input</i>	The array of input tensors, each tensor size = dim_1 x dim_2 x .. x dim_n x typesize, allocated by caller
out	<i>output</i>	The array of output tensors, each tensor size = dim_1 x dim_2 x .. x dim_n x typesize, the memory block for output tensor should be allocated. (data in GstTensorMemory)

Note

rank limit (NNS_TENSOR_RANK_LIMIT) and typesize (tensor_element_size) defined in [tensor_typedef.h](#)

Returns

0 if success

Definition at line 111 of file tensor_filter_custom.h.

9.82.2.2 NNS_custom_destroy_notify

```
typedef void(* NNS_custom_destroy_notify) (void *data)
```

It's a post-processing method about the used data pointer if it has been allocated at custom filter.

Parameters

in	<i>data</i>	the data element.
----	-------------	-------------------

Definition at line 118 of file tensor_filter_custom.h.

9.82.2.3 NNS_custom_exit_func

```
typedef void(* NNS_custom_exit_func) (void *private_data, const GstTensorFilterProperties *prop)
```

A function that is called after calling other functions, when it's ready to close.

Parameters

in	<i>private_data</i>	If you have allocated *private_data at init, free it here.
in	<i>prop</i>	GstTensorFilter's property values. Do not change its values.

Definition at line 53 of file tensor_filter_custom.h.

9.82.2.4 NNS_custom_get_input_dimension

```
typedef int(* NNS_custom_get_input_dimension) (void *private_data, const GstTensorFilterProperties *prop, GstTensorsInfo *info)
```

Get input tensor type.

Parameters

in	<i>private_data</i>	The pointer returned by NNSStreamer_custom_init.
in	<i>prop</i>	GstTensorFilter's property values. Do not change its values.
out	<i>info</i>	Structure for tensor info.

Returns

0 if success

Definition at line 63 of file tensor_filter_custom.h.

9.82.2.5 NNS_custom_get_output_dimension

```
typedef int (* NNS_custom_get_output_dimension) (void *private_data, const GstTensorFilterProperties
*prop, GstTensorsInfo *info)
```

Get output tensor type.

Parameters

in	<i>private_data</i>	The pointer returned by NNStreamer_custom_init.
in	<i>prop</i>	GstTensorFilter's property values. Do not change its values.
out	<i>info</i>	Structure for tensor info.

Returns

0 if success

Definition at line 73 of file tensor_filter_custom.h.

9.82.2.6 NNS_custom_init_func

```
typedef void* (* NNS_custom_init_func) (const GstTensorFilterProperties *prop)
```

A function that is called before calling other functions.

Parameters

in	<i>prop</i>	GstTensorFilter's property values. Do not change its values.
----	-------------	--

Returns

The returned pointer will be passed to other functions as "private_data".

Definition at line 46 of file tensor_filter_custom.h.

9.82.2.7 NNS_custom_invoke

```
typedef int (* NNS_custom_invoke) (void *private_data, const GstTensorFilterProperties *prop,
const GstTensorMemory *input, GstTensorMemory *output)
```

Invoke the "main function". Without allocating output buffer. (fill in the given output buffer)

Parameters

in	<i>private_data</i>	The pointer returned by NNStreamer_custom_init.
in	<i>prop</i>	GstTensorFilter's property values. Do not change its values.
in	<i>input</i>	The array of input tensors, each tensor size = dim_1 x dim_2 x .. x dim_n x typesize, allocated by caller
out	<i>output</i>	The array of output tensors, each tensor size = dim_1 x dim_2 x .. x dim_n x typesize, allocated by caller

Note

rank limit (NNS_TENSOR_RANK_LIMIT) and typesize (tensor_element_size) defined in [tensor_typedef.h](#)

Returns

0 if success

Definition at line 99 of file tensor_filter_custom.h.

9.82.2.8 NNS_custom_set_input_dimension

```
typedef int (* NNS_custom_set_input_dimension) (void *private_data, const GstTensorFilterProperties
*prop, const GstTensorsInfo *in_info, GstTensorsInfo *out_info)
```

Set input dim by framework. Let custom plugin set output dim accordingly.

Parameters

in	<i>private_data</i>	The pointer returned by NNStreamer_custom_init.
in	<i>prop</i>	GstTensorFilter's property values. Do not change its values.
in	<i>in_info</i>	Input tensor info designated by the GStreamer framework. Note that this is not a fixed value and GStreamer may try different values during pad-cap negotiations.
out	<i>out_info</i>	Output tensor info according to the input tensor info.

@caution Do not fix internal values based on this call. GStreamer may call this function repeatedly with different values during pad-cap negotiations. Fix values when invoke is finally called.

Definition at line 87 of file tensor_filter_custom.h.

9.82.2.9 NNStreamer_custom_class

```
typedef struct _NNStreamer_custom_class NNStreamer_custom_class
```

Definition at line 136 of file tensor_filter_custom.h.

9.82.3 Variable Documentation**9.82.3.1 NNStreamer_custom**

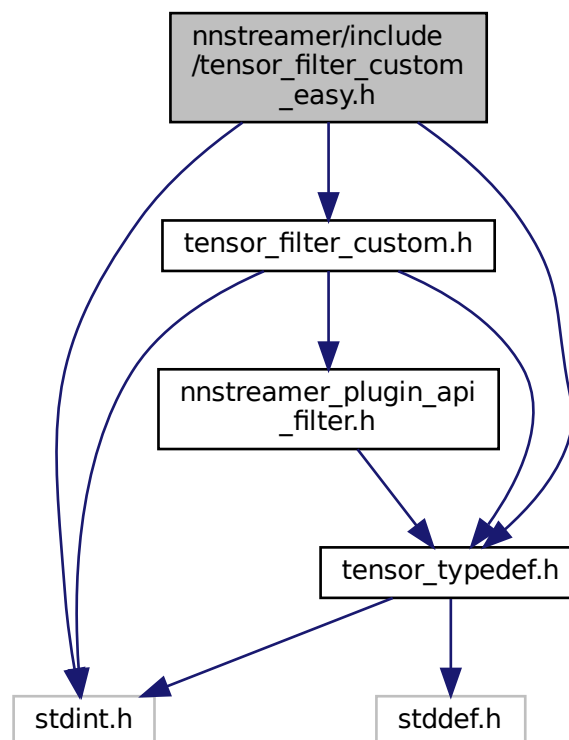
```
NNStreamer_custom_class* NNStreamer_custom
```

A custom filter MUST define NNStreamer_custom. This object represents the custom filter itself.

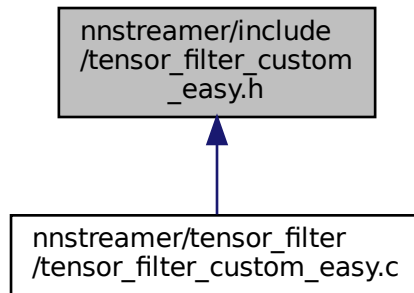
9.83 nnstreamer/include/tensor_filter_custom_easy.h File Reference

Custom tensor processing interface for simple functions.

```
#include <stdint.h>
#include "tensor_typedef.h"
#include "tensor_filter_custom.h"
Include dependency graph for tensor_filter_custom_easy.h:
```



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef int(* [NNS_custom_invoke_dynamic](#)) (void *private_data, const [GstTensorsInfo](#) *in_info, [GstTensorsInfo](#) *out_info, const [GstTensorMemory](#) *input, [GstTensorMemory](#) *output)
Invoke the "main function" with flexible input and output. Output tensor memory should be allocated.

Functions

- G_BEGIN_DECLS int [NNS_custom_easy_register](#) (const char *modelname, [NNS_custom_invoke](#) func, void *data, const [GstTensorsInfo](#) *in_info, const [GstTensorsInfo](#) *out_info)
Register the custom-easy tensor function.
- int [NNS_custom_easy_dynamic_register](#) (const char *modelname, [NNS_custom_invoke_dynamic](#) func, void *data, const [GstTensorsInfo](#) *in_info)
Register the custom-easy tensor function for dynamic invoke.
- int [NNS_custom_easy_unregister](#) (const char *modelname)
Unregister the custom-easy tensor function.

9.83.1 Detailed Description

Custom tensor processing interface for simple functions.

GStreamer Tensor_Filter, Customized Module, Easy Mode Copyright (C) 2019 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

24 Oct 2019

See also<http://github.com/nnstreamer/nnstreamer>**Author**MyungJoo Ham myungjoo.ham@samsung.com**Bug** No known bugs except for NYI items

How To for NNdevelopers:

Case 1. Provide the function as a shared object for other apps.

1. Define struct, "NNStreamer_custom_easy", with the functions defined.
2. Call [NNS_custom_easy_register\(\)](#) at a global init/constructor function.
3. Compile as a shared object. (.so in Linux) And install at the custom-filter path.
4. Use NNStreamer (tensor_filter framework=custom_easy, model=\${modelName}, custom=\${_FILEPATH_O↵
F_YOUR_SO} ...)
5. Note that you may register multiple models (functions) with a single .so.

Case 2. Define the function in the app.

1. Define struct, "NNStreamer_custom_easy", with the functions defined.
2. Register the struct with "NNS_custom_easy_register" API.
3. Construct the nnstreamer pipeline and execute it in the app.

Note that this does not support flexible dimensions.

To Packagers:

This file is to be packaged as "devel" package for NN developers.

9.83.2 Typedef Documentation

9.83.2.1 NNS_custom_invoke_dynamic

```
typedef int(* NNS_custom_invoke_dynamic) (void *private_data, const GstTensorsInfo *in_info,
GstTensorsInfo *out_info, const GstTensorMemory *input, GstTensorMemory *output)
```

Invoke the "main function" with flexible input and output. Output tensor memory should be allocated.

Parameters

	<i>[in/out]</i>	private_data A subplugin may save its internal private data here. The subplugin is responsible for alloc/free of this pointer.
in	<i>info</i>	structure of input tensors info
out	<i>info</i>	structure of output tensors info. The subplugin should fill this info.
in	<i>input</i>	The array of input tensors. Allocated and filled by tensor_filter/main
out	<i>output</i>	The array of output tensors. The subplugin should allocate the memory block for output tensor. (data in GstTensorMemory)

Note

rank limit (NNS_TENSOR_RANK_LIMIT) and typesize (tensor_element_size) defined in [tensor_typedef.h](#)

Returns

0 if success

Definition at line 76 of file tensor_filter_custom_easy.h.

9.83.3 Function Documentation

9.83.3.1 NNS_custom_easy_dynamic_register()

```
int NNS_custom_easy_dynamic_register (
    const char * modelname,
    NNS_custom_invoke_dynamic func,
    void * data,
    const GstTensorsInfo * in_info )
```

Register the custom-easy tensor function for dynamic invoke.

Parameters

in	<i>modelname</i>	The name of custom-easy tensor function.
in	<i>func</i>	The tensor function body
	<i>[in/out]</i>	private_data The internal data for the function
in	<i>in_info</i>	Input tensor metadata.

Note

NNS_custom_invoke_dynamic defined in [tensor_filter_custom.h](#) Output buffers should be allocated in the invoke function.

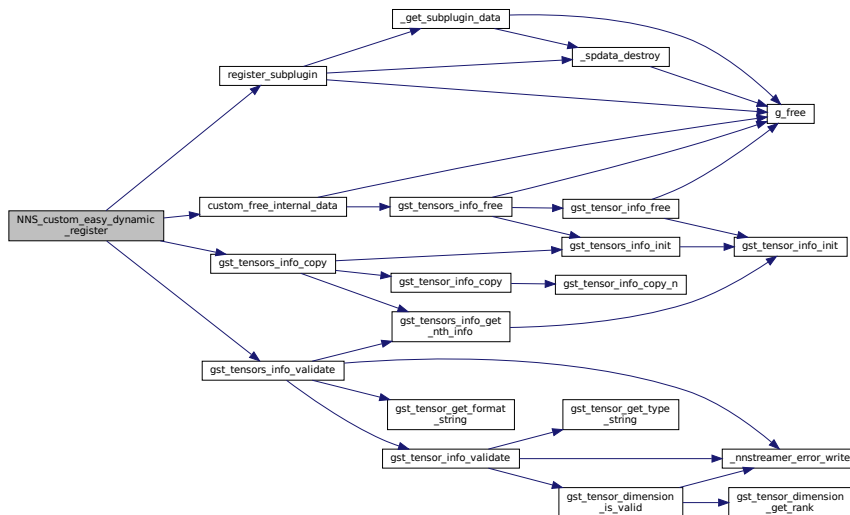
Register the custom-easy tensor function for dynamic invoke.

Returns

0 if success. -ERRNO if error.

Definition at line 112 of file `tensor_filter_custom_easy.c`.

Here is the call graph for this function:

**9.83.3.2 NNS_custom_easy_register()**

```
G_BEGIN_DECLS int NNS_custom_easy_register (
    const char * modelname,
    NNS_custom_invoke func,
    void * data,
    const GstTensorsInfo * in_info,
    const GstTensorsInfo * out_info )
```

Register the custom-easy tensor function.

Parameters

in	<i>modelname</i>	The name of custom-easy tensor function.
in	<i>func</i>	The tensor function body
	<i>[in/out]</i>	<i>private_data</i> The internal data for the function
in	<i>in_info</i>	Input tensor metadata.
in	<i>out_info</i>	Output tensor metadata

Note

`NNS_custom_invoke` defined in [tensor_filter_custom.h](#) Output buffers for `func` are preallocated.

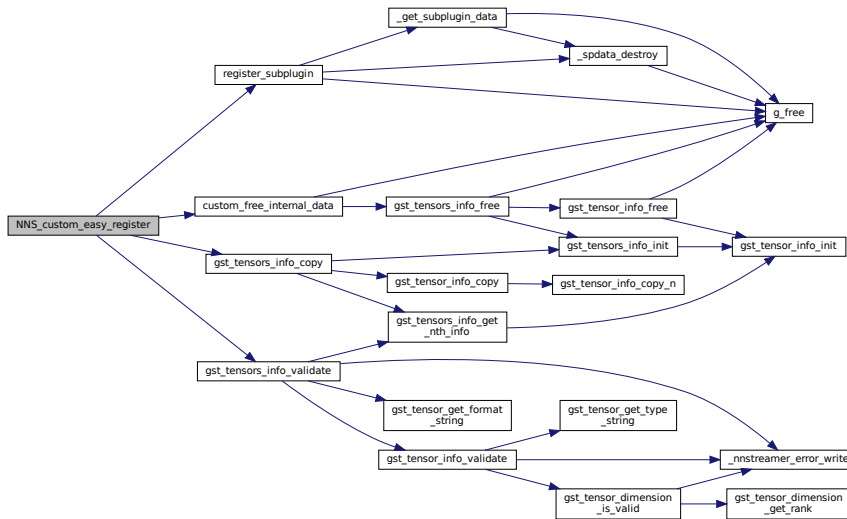
Register the custom-easy tensor function.

Returns

0 if success. -ERRNO if error.

Definition at line 76 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



9.83.3.3 NNS_custom_easy_unregister()

```
int NNS_custom_easy_unregister (
    const char * modelname )
```

Unregister the custom-easy tensor function.

Parameters

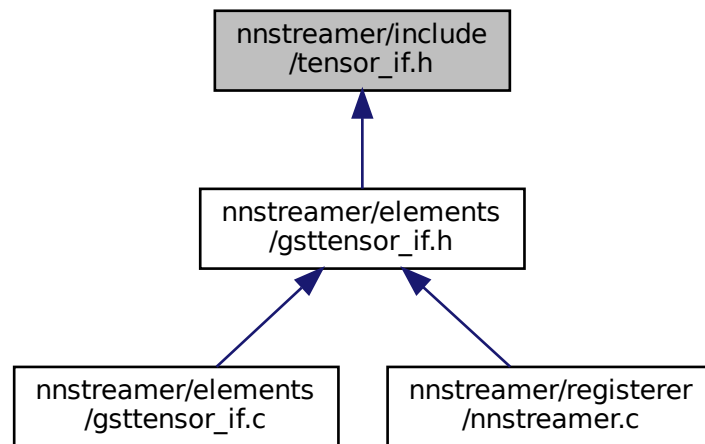
in	<i>modelname</i>	The registered name of custom-easy tensor function.
----	------------------	---

Returns

0 if success, non-zero if error
 0 if success. -EINVAL if invalid model name.

Definition at line 144 of file tensor_filter_custom_easy.c.

This graph shows which files directly or indirectly include this file:



Functions

- int `nnstreamer_if_custom_register` (const gchar *name, `tensor_if_custom` func, void *data)
Register the custom callback function.
- int `nnstreamer_if_custom_unregister` (const gchar *name)
Unregister the custom callback function.

Variables

- G_BEGIN_DECLS typedef gboolean(* `tensor_if_custom`)(const `GstTensorsInfo` *info, const `GstTensorMemory` *input, void *user_data, gboolean *result)
Calls the user defined conditions function.

9.84.1 Detailed Description

NNStreamer APIs for `tensor_if` custom condition.

GStreamer/NNStreamer Tensor-IF Copyright (C) 2020 Gichan Jang gichan2.jang@samsung.com

Date

28 Oct 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.84.2 Function Documentation

9.84.2.1 nnstreamer_if_custom_register()

```
int nnstreamer_if_custom_register (
    const gchar * name,
    tensor_if_custom func,
    void * data )
```

Register the custom callback function.

Parameters

in	<i>name</i>	The name of tensor_if custom callback function.
in	<i>func</i>	The custom condition function body
	<i>[in/out]</i>	data The internal data for the function

Returns

0 if success. -ERRNO if error.

Note

GstElement *tensor_if_h* can be get with `gst_bin_get_by_name` from the pipeline. e.g., pipeline description: ... ! tensor_if name=tif (... properties ...) ! ... GstElement **tensor_if_h* = `gst_bin_get_by_name` (GST_BIN (pipeline), "tif");

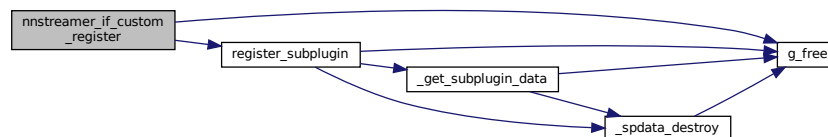
Register the custom callback function.

Returns

0 if success. -ERRNO if error.

Definition at line 1023 of file `gsttensor_if.c`.

Here is the call graph for this function:



9.84.2.2 nnstreamer_if_custom_unregister()

```
int nnstreamer_if_custom_unregister (
    const gchar * name )
```

Unregister the custom callback function.

Usage example of the tensor_if custom condition

```
// Define custom callback function describing the condition
gboolean tensor_if_custom_cb (const GstTensorsInfo *info,
    const GstTensorMemory * input, gboolean * result) {
    // Describe the conditions and pass the results
}
...
// Register custom callback function describing the condition
nnstreamer_if_custom_register ("tifx", tensor_if_custom_cb, NULL);
...
// Use the condition in a pipeline.
// E.g., Pipeline of " ... ! tensor_if compared-value=CUSTOM compared-value-option=tifx then=TENSORPICK
    then-option=0 else=TENSORPICK else-option=1 ! ... "
...
// After everything is done.
nnstreamer_if_custom_unregister ("tifx");
```

Parameters

in	<i>info</i>	input tensors info
in	<i>input</i>	memory containing input tensor data
in	<i>user_data</i>	private data for the callback
out	<i>result</i>	result of the user defined condition

Returns

TRUE if there is no error.

Definition at line 63 of file tensor_if.h.

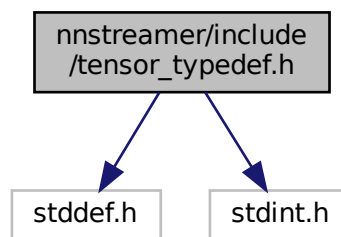
9.85 nnstreamer/include/tensor_typedef.h File Reference

Common header file for NNStreamer, the GStreamer plugin for neural networks.

```
#include <stddef.h>
```

```
#include <stdint.h>
```

Include dependency graph for tensor_typedef.h:



This graph shows which files directly or indirectly include this file:



Classes

- union [tensor_element](#)
To make the code simple with all the types. "C++ Template"-like.
- struct [GstTensorMemory](#)
The unit of each data tensors. It will be used as an input/output tensor of other/tensors.
- struct [GstTensorInfo](#)
Internal data structure for tensor info.
- struct [GstTensorsInfo](#)
Internal meta data exchange format for a other/tensors instance.
- struct [GstTensorsConfig](#)
Internal data structure for configured tensors info (for other/tensors).
- struct [GstSparseTensorInfo](#)
Internal data structure for sparse tensor info.
- struct [GstTensorMetaInfo](#)
Data structure to describe a tensor data. This represents the basic information of a memory block for tensor stream.

Macros

- #define [NNS_TENSOR_RANK_LIMIT](#) (16)
- #define [NNS_TENSOR_SIZE_LIMIT](#) (256)
The number of tensors NNStreamer supports is 256. The max memories of gst-buffer is 16 (See [NNS_TENSOR_MEMORY_MAX](#)). Internally NNStreamer handles the memories as tensors. If the number of tensors is larger than 16, we modify the last memory and combine tensors into the memory.
- #define [NNS_TENSOR_SIZE_LIMIT_STR](#) "256"
- #define [NNS_TENSOR_MEMORY_MAX](#) (16)
This value, 16, can be checked with `gst_buffer_get_max_memory()`, which is `GST_BUFFER_MEM_MAX` in `gststreamer/gstbuffer.c`. We redefined the value because `GST_BUFFER_MEM_MAX` is not exported and we need static value. To modify (increase) this value, you need to update `gststreamer/gstbuffer.c` as well.
- #define [NNS_TENSOR_SIZE_EXTRA_LIMIT](#) ([NNS_TENSOR_SIZE_LIMIT](#) - [NNS_TENSOR_MEMORY_MAX](#))
Max number of extra tensors.
- #define [NNS_MIMETYPE_TENSOR](#) "other/tensor"
- #define [NNS_MIMETYPE_TENSORS](#) "other/tensors"
- #define [GST_TENSOR_NUM_TENSORS_RANGE](#) "(int) [1, " [NNS_TENSOR_SIZE_LIMIT_STR](#) "]"
- #define [GST_TENSOR_RATE_RANGE](#) "(fraction) [0, max]"
- #define [GST_TENSOR_TYPE_ALL](#) "{ float16, float32, float64, int64, uint64, int32, uint32, int16, uint16, int8, uint8 }"
Possible tensor element types.
- #define [GST_TENSOR_FORMAT_ALL](#) "{ static, flexible, sparse }"
Possible tensor formats.
- #define [GST_TENSOR_CAP_DEFAULT](#)
Default static capability for other/tensor.
- #define [GST_TENSORS_CAP_MAKE](#)(fmt)
Caps string for the caps template of tensor stream. format should be a string that describes the data format, or possible formats of incoming tensor.
- #define [GST_TENSORS_CAP_WITH_NUM](#)(num)
Caps string for the caps template (other/tensors, static tensor stream with fixed number of tensors). num should be a string format that describes the number of tensors, or the range of incoming tensors. The types and dimensions of tensors should be described for caps negotiation.
- #define [GST_TENSORS_CAP_DEFAULT](#) [GST_TENSORS_CAP_WITH_NUM](#) ([GST_TENSOR_NUM_TENSORS_RANGE](#))
Caps string for the caps template of static tensor stream.
- #define [GST_TENSORS_FLEX_CAP_DEFAULT](#) [GST_TENSORS_CAP_MAKE](#) ("flexible")

Caps string for the caps template of flexible tensors. This mimetype handles non-static, flexible tensor stream without specifying the data type and shape of the tensor. The maximum number of tensors in a buffer is 16 (NNS_TENSOR_SIZE_LIMIT).

- #define `GST_TENSORS_SPARSE_CAP_DEFAULT` `GST_TENSORS_CAP_MAKE` ("sparse")

Caps string for the caps template of sparse tensors. This mimetype handles non-static, sparse tensor stream without specifying the data type and shape of the tensor. The maximum number of tensors in a buffer is 16 (NNS_TENSOR_SIZE_LIMIT).

Typedefs

- typedef enum `_nns_tensor_type` `tensor_type`
Possible data element types of other/tensor.
- typedef enum `_nns_media_type` `media_type`
Float16 compiler extension support.
- typedef enum `_tensor_format` `tensor_format`
Data format of tensor stream in the pipeline.
- typedef enum `_nns_tensor_layout` `tensor_layout`
Tensor layout format for other/tensor.
- typedef uint32_t `tensor_dim`[`NNS_TENSOR_RANK_LIMIT`]

Enumerations

- enum `_nns_tensor_type` {
`_NNS_INT32 = 0, _NNS_UINT32, _NNS_INT16, _NNS_UINT16,`
`_NNS_INT8, _NNS_UINT8, _NNS_FLOAT64, _NNS_FLOAT32,`
`_NNS_INT64, _NNS_UINT64, _NNS_FLOAT16, _NNS_END` }
Possible data element types of other/tensor.
- enum `_nns_media_type` {
`_NNS_MEDIA_INVALID = -1, _NNS_VIDEO = 0, _NNS_AUDIO = 1, _NNS_TEXT = 2,`
`_NNS_OCTET = 3, _NNS_TENSOR = 4, _NNS_MEDIA_ANY = 0x1000` }
Float16 compiler extension support.
- enum `_tensor_format` { `_NNS_TENSOR_FORMAT_STATIC = 0, _NNS_TENSOR_FORMAT_FLEXIBLE,`
`_NNS_TENSOR_FORMAT_SPARSE, _NNS_TENSOR_FORMAT_END` }
Data format of tensor stream in the pipeline.
- enum `_nns_tensor_layout` { `_NNS_LAYOUT_ANY = 0, _NNS_LAYOUT_NHWC, _NNS_LAYOUT_NCHW,`
`_NNS_LAYOUT_NONE` }
Tensor layout format for other/tensor.

9.85.1 Detailed Description

Common header file for NNStreamer, the GStreamer plugin for neural networks.

NNStreamer Common Header, Typedef part, for export as devel package. Copyright (C) 2018 MyungJoo Ham
myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

01 Jun 2018

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

To Packagers:

This file is to be packaged as "devel" package for NN developers.

9.85.2 Macro Definition Documentation

9.85.2.1 GST_TENSOR_CAP_DEFAULT

```
#define GST_TENSOR_CAP_DEFAULT
```

Value:

```
NNS_MIMETYPE_TENSOR ", " \  
"framerate = " GST_TENSOR_RATE_RANGE
```

Default static capability for other/tensor.

Definition at line 78 of file tensor_typedef.h.

9.85.2.2 GST_TENSOR_FORMAT_ALL

```
#define GST_TENSOR_FORMAT_ALL "{ static, flexible, sparse }"
```

Possible tensor formats.

Definition at line 73 of file tensor_typedef.h.

9.85.2.3 GST_TENSOR_NUM_TENSORS_RANGE

```
#define GST_TENSOR_NUM_TENSORS_RANGE "(int) [ 1, " NNS_TENSOR_SIZE_LIMIT_STR " ]"
```

Definition at line 62 of file tensor_typedef.h.

9.85.2.4 GST_TENSOR_RATE_RANGE

```
#define GST_TENSOR_RATE_RANGE "(fraction) [ 0, max ]"
```

Definition at line 63 of file tensor_typedef.h.

9.85.2.5 GST_TENSOR_TYPE_ALL

```
#define GST_TENSOR_TYPE_ALL "{ float16, float32, float64, int64, uint64, int32, uint32, int16, uint16, int8, uint8 }"
```

Possible tensor element types.

Definition at line 68 of file tensor_typedef.h.

9.85.2.6 GST_TENSORS_CAP_DEFAULT

```
#define GST_TENSORS_CAP_DEFAULT GST_TENSORS_CAP_WITH_NUM (GST_TENSOR_NUM_TENSORS_RANGE)
```

Caps string for the caps template of static tensor stream.

Definition at line 115 of file tensor_typedef.h.

9.85.2.7 GST_TENSORS_CAP_MAKE

```
#define GST_TENSORS_CAP_MAKE(  
    fmt )
```

Value:

```
NNS_MIMETYPE_TENSORS " , " \  
"format = (string) " fmt " , " \  
"framerate = " GST_TENSOR_RATE_RANGE
```

Caps string for the caps template of tensor stream. `format` should be a string that describes the data format, or possible formats of incoming tensor.

`type` should be one of types in `GST_TENSOR_TYPE_ALL` "`type = (string) uint8`" `dimension` should be a formatted string with rank `NNS_TENSOR_RANK_LIMIT` "`dimension = (string) dim1:dim2:dim3:dim4`" If the data format is static, another tensor information should be described in caps.

- `num_tensors`: The number of tensors in a `GstBuffer`.
- `types`: The list of data type in the tensor. `type` should be one of types in `GST_TENSOR_TYPE_ALL`. (`types=(string)"type1,type2,type3"`)
- `dimensions`: The list of dimension in the tensor. `dimension` should be a formatted string with rank `NNS_TENSOR_RANK_LIMIT`. (`dimensions=(string)"dim1:dim2:dim3:dim4,dim1:dim2:dim3:dim4"`)

Definition at line 97 of file tensor_typedef.h.

9.85.2.8 GST_TENSORS_CAP_WITH_NUM

```
#define GST_TENSORS_CAP_WITH_NUM(  
    num )
```

Value:

```
NNS_MIMETYPE_TENSORS ", " \  
"format = (string) static, num_tensors = " num ", " \  
"framerate = " GST_TENSOR_RATE_RANGE
```

Caps string for the caps template (other/tensors, static tensor stream with fixed number of tensors). num should be a string format that describes the number of tensors, or the range of incoming tensors. The types and dimensions of tensors should be described for caps negotiation.

Definition at line 107 of file tensor_typedef.h.

9.85.2.9 GST_TENSORS_FLEX_CAP_DEFAULT

```
#define GST_TENSORS_FLEX_CAP_DEFAULT GST_TENSORS_CAP_MAKE ("flexible")
```

Caps string for the caps template of flexible tensors. This mimetype handles non-static, flexible tensor stream without specifying the data type and shape of the tensor. The maximum number of tensors in a buffer is 16 (NNS↔_TENSOR_SIZE_LIMIT).

Definition at line 123 of file tensor_typedef.h.

9.85.2.10 GST_TENSORS_SPARSE_CAP_DEFAULT

```
#define GST_TENSORS_SPARSE_CAP_DEFAULT GST_TENSORS_CAP_MAKE ("sparse")
```

Caps string for the caps template of sparse tensors. This mimetype handles non-static, sparse tensor stream without specifying the data type and shape of the tensor. The maximum number of tensors in a buffer is 16 (NNS↔_TENSOR_SIZE_LIMIT).

Definition at line 131 of file tensor_typedef.h.

9.85.2.11 NNS_MIMETYPE_TENSOR

```
#define NNS_MIMETYPE_TENSOR "other/tensor"
```

Definition at line 59 of file tensor_typedef.h.

9.85.2.12 NNS_MIMETYPE_TENSORS

```
#define NNS_MIMETYPE_TENSORS "other/tensors"
```

Definition at line 60 of file tensor_typedef.h.

9.85.2.13 NNS_TENSOR_MEMORY_MAX

```
#define NNS_TENSOR_MEMORY_MAX (16)
```

This value, 16, can be checked with `gst_buffer_get_max_memory()`, which is `GST_BUFFER_MEM_MAX` in `gststreamer/gstbuffer.c`. We redefined the value because `GST_BUFFER_MEM_MAX` is not exported and we need static value. To modify (increase) this value, you need to update `gststreamer/gstbuffer.c` as well.

Definition at line 52 of file tensor_typedef.h.

9.85.2.14 NNS_TENSOR_RANK_LIMIT

```
#define NNS_TENSOR_RANK_LIMIT (16)
```

Definition at line 34 of file tensor_typedef.h.

9.85.2.15 NNS_TENSOR_SIZE_EXTRA_LIMIT

```
#define NNS_TENSOR_SIZE_EXTRA_LIMIT (NNS_TENSOR_SIZE_LIMIT - NNS_TENSOR_MEMORY_MAX)
```

Max number of extra tensors.

Definition at line 57 of file tensor_typedef.h.

9.85.2.16 NNS_TENSOR_SIZE_LIMIT

```
#define NNS_TENSOR_SIZE_LIMIT (256)
```

The number of tensors NNStreamer supports is 256. The max memories of gst-buffer is 16 (See `NNS_TENSOR_MEMORY_MAX`). Internally NNStreamer handles the memories as tensors. If the number of tensors is larger than 16, we modify the last memory and combine tensors into the memory.

Definition at line 42 of file tensor_typedef.h.

9.85.2.17 NNS_TENSOR_SIZE_LIMIT_STR

```
#define NNS_TENSOR_SIZE_LIMIT_STR "256"
```

Definition at line 43 of file tensor_typedef.h.

9.85.3 Typedef Documentation

9.85.3.1 media_type

```
typedef enum _nns_media_type media_type
```

Float16 compiler extension support.

Possible input stream types for other/tensor.

This is related with media input stream to other/tensor. There is no restrictions for the outputs.

In order to prevent enum-mix issues between device profiles, we explicitly define numbers for each enum type.

9.85.3.2 tensor_dim

```
typedef uint32_t tensor_dim[NNS_TENSOR_RANK_LIMIT]
```

Definition at line 247 of file tensor_typedef.h.

9.85.3.3 tensor_format

```
typedef enum _tensor_format tensor_format
```

Data format of tensor stream in the pipeline.

9.85.3.4 tensor_layout

```
typedef enum _nns_tensor_layout tensor_layout
```

Tensor layout format for other/tensor.

The layout is needed by some of the element to appropriately process the data based on the axis of the channel in the data. Layout information will be currently utilized by only some of the elements (SNAP, NNFW in tensor_filter, PADDING mode in tensor_transform)

Tensor layout is not part of the capabilities of the element, and does not take part in the caps negotiation.

NONE layout implies that the layout of the data is neither NHWC nor NCHW. ' However, ANY layout implies that the layout of the provided data is not relevant.

Note

Providing tensor layout can also decide acceleration to be supported as not all the accelerators might support all the layouts (NYI).

9.85.3.5 tensor_type

```
typedef enum _nns_tensor_type tensor_type
```

Possible data element types of other/tensor.

Note

When changing tensor type, you should update related type in ML-API and protobuf/flatbuf schema also.

9.85.4 Enumeration Type Documentation

9.85.4.1 _nns_media_type

```
enum _nns_media_type
```

Float16 compiler extension support.

Possible input stream types for other/tensor.

This is related with media input stream to other/tensor. There is no restrictions for the outputs.

In order to prevent enum-mix issues between device profiles, we explicitly define numbers for each enum type.

Enumerator

_NNS_MEDIA_INVALID	Uninitialized
_NNS_VIDEO	supposedly video/x-raw
_NNS_AUDIO	supposedly audio/x-raw
_NNS_TEXT	supposedly text/x-raw
_NNS_OCTET	supposedly application/octet-stream
_NNS_TENSOR	supposedly other/tensor(s) or flexible tensor
_NNS_MEDIA_ANY	any media type (find proper external converter in tensor-converter element)

Definition at line 179 of file tensor_typedef.h.

9.85.4.2 _nns_tensor_layout

```
enum _nns_tensor_layout
```

Tensor layout format for other/tensor.

The layout is needed by some of the element to appropriately process the data based on the axis of the channel in the data. Layout information will be currently utilized by only some of the elements (SNAP, NNFWD in tensor_filter, PADDING mode in tensor_transform)

Tensor layout is not part of the capabilities of the element, and does not take part in the caps negotiation.

NONE layout implies that the layout of the data is neither NHWC nor NCHW. ' However, ANY layout implies that the layout of the provided data is not relevant.

Note

Providing tensor layout can also decide acceleration to be supported as not all the accelerators might support all the layouts (NYI).

Enumerator

<code>_NNS_LAYOUT_ANY</code>	does not care about the data layout
<code>_NNS_LAYOUT_NHWC</code>	NHWC: channel last layout
<code>_NNS_LAYOUT_NCHW</code>	NCHW: channel first layout
<code>_NNS_LAYOUT_NONE</code>	NONE: none of the above defined layouts

Definition at line 220 of file `tensor_typedef.h`.

9.85.4.3 `_nns_tensor_type`

```
enum _nns_tensor_type
```

Possible data element types of other/tensor.

Note

When changing tensor type, you should update related type in ML-API and protobuf/flatbuf schema also.

Enumerator

<code>_NNS_INT32</code>	
<code>_NNS_UINT32</code>	
<code>_NNS_INT16</code>	
<code>_NNS_UINT16</code>	
<code>_NNS_INT8</code>	
<code>_NNS_UINT8</code>	
<code>_NNS_FLOAT64</code>	
<code>_NNS_FLOAT32</code>	
<code>_NNS_INT64</code>	
<code>_NNS_UINT64</code>	
<code>_NNS_FLOAT16</code>	added with nntstreamer 2.1.1-devel. If you add any operators (e.g., <code>tensor_transform</code>) to float16, it will either be not supported or be too inefficient.
<code>_NNS_END</code>	

Definition at line 138 of file `tensor_typedef.h`.

9.85.4.4 `_tensor_format`

```
enum _tensor_format
```

Data format of tensor stream in the pipeline.

Enumerator

_NNS_TENSOR_FORMAT_STATIC	
_NNS_TENSOR_FORMAT_FLEXIBLE	
_NNS_TENSOR_FORMAT_SPARSE	
_NNS_TENSOR_FORMAT_END	

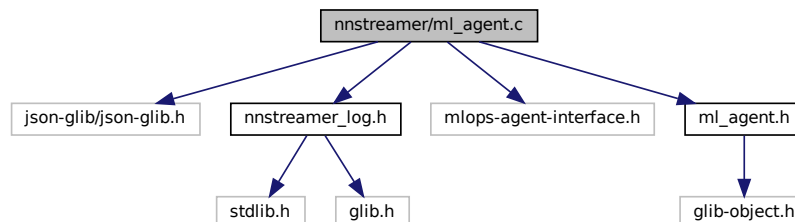
Definition at line 193 of file tensor_typedef.h.

9.86 nnstreamer/ml_agent.c File Reference

Internal helpers to make a bridge between NNS filters and the ML Agent service.

```
#include <json-glib/json-glib.h>
#include <nnstreamer_log.h>
#include <mlops-agent-interface.h>
#include "ml_agent.h"
```

Include dependency graph for ml_agent.c:



Functions

- `gchar * mlagent_get_model_path_from (const GValue *val)`
Get a path of the model file from a given GValue.

Variables

- `const gchar URI_SCHEME [] = "mlagent"`
- `const gchar URI_KEYWORD_MODEL [] = "model"`
- `const gchar JSON_KEY_MODEL_PATH [] = "path"`

9.86.1 Detailed Description

Internal helpers to make a bridge between NNS filters and the ML Agent service.

Copyright (C) 2023 Wook Song wook16.song@samsung.com Copyright (c) 2023 Samsung Electronics Co., Ltd. All Rights Reserved.

Date

23 Jun 2023

See also

<http://github.com/nntstreamer/nntstreamer>

Author

wook16.song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.86.2 Function Documentation

9.86.2.1 mlagent_get_model_path_from()

```
gchar* mlagent_get_model_path_from (  
    const GValue * val )
```

Get a path of the model file from a given GValue.

Parameters

in	<i>val</i>	A pointer to a GValue holding a G_TYPE_STRING value
----	------------	---

Returns

A newly allocated c-string representing the model file path, if the given GValue contains a valid URI Otherwise, it simply returns a duplicated (strdup'ed) c-string that the val contains.

Note

The caller should free the return c-string after using it.

Common checker for the given URI

Note

Only for the following URI formats to get the file path of the matching models are currently supported.
mlagent://model/name or mlagent://model/name/version

It is required to be revised to support more scenarios that exploit the ML Agent.

Convert the given URI for a model to the file path

Todo The specification of the data layout filled in the third argument (i.e., `stringified_json`) by the callee is not fully decided.

Todo Parse `stringified_json` to get the model's path

Definition at line 33 of file `ml_agent.c`.

Here is the caller graph for this function:



9.86.3 Variable Documentation

9.86.3.1 JSON_KEY_MODEL_PATH

```
const gchar JSON_KEY_MODEL_PATH[] = "path"
```

Definition at line 23 of file `ml_agent.c`.

9.86.3.2 URI_KEYWORD_MODEL

```
const gchar URI_KEYWORD_MODEL[] = "model"
```

Definition at line 22 of file `ml_agent.c`.

9.86.3.3 URI_SCHEME

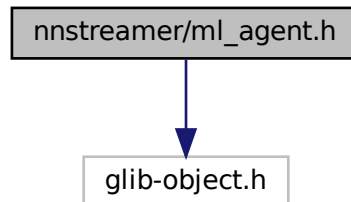
```
const gchar URI_SCHEME[] = "mlagent"
```

Definition at line 21 of file `ml_agent.c`.

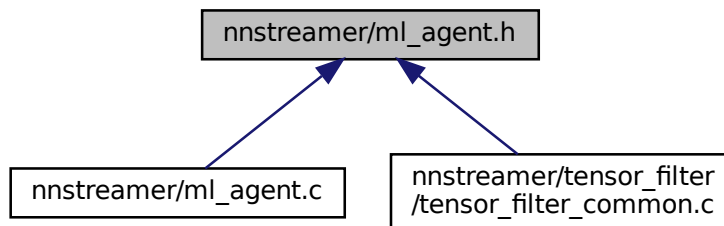
9.87 nnstreamer/ml_agent.h File Reference

Internal header to make a bridge between NNS filters and the ML Agent service.

```
#include <glib-object.h>  
Include dependency graph for ml_agent.h:
```



This graph shows which files directly or indirectly include this file:



Macros

- #define [mlagent_get_model_path_from\(v\)](#) g_value_dup_string (v)

9.87.1 Detailed Description

Internal header to make a bridge between NNS filters and the ML Agent service.

Copyright (C) 2023 Wook Song wook16.song@samsung.com Copyright (c) 2023 Samsung Electronics Co., Ltd. All Rights Reserved.

Date

23 Jun 2023

See also

<http://github.com/nnstreamer/nnstreamer>

Author

wook16.song wook16.song@samsung.com

Bug No known bugs except for NYI items

9.87.2 Macro Definition Documentation

9.87.2.1 mlagent_get_model_path_from

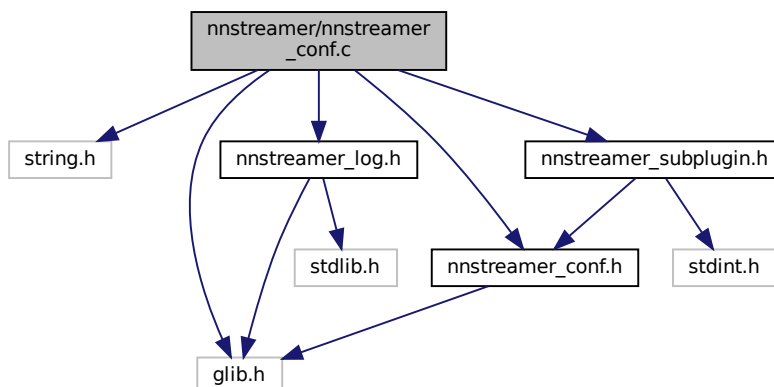
```
#define mlagent_get_model_path_from(  
    v ) g_value_dup_string (v)
```

Definition at line 33 of file ml_agent.h.

9.88 nnstreamer/nnstreamer_conf.c File Reference

NNStreamer Configuration (conf file, env-var) Management.

```
#include <string.h>  
#include <glib.h>  
#include "nnstreamer_log.h"  
#include "nnstreamer_conf.h"  
#include "nnstreamer_subplugin.h"  
Include dependency graph for nnstreamer_conf.c:
```



Classes

- struct [subplugin_conf](#)
- struct [confdata](#)
- struct [vstr_helper](#)
 - *Data structure for `_g_list_foreach_vstr_helper`.*
- struct [dump_buf](#)

Macros

- `#define NNSTREAMER_PREFIX_DECODER "libnnsreamer_decoder_"`
- `#define NNSTREAMER_PREFIX_FILTER "libnnsreamer_filter_"`
- `#define NNSTREAMER_PREFIX_CUSTOMFILTERS ""`
- `#define NNSTREAMER_PREFIX_CONVERTER "libnnsreamer_converter_"`
- `#define NNSTREAMER_PREFIX_TRAINER "libnnsreamer_trainer_"`
- `#define STR_BOOL(x) ((x) ? "TRUE" : "FALSE")`

Enumerations

- enum [conf_sources](#) {
 - `CONF_SOURCE_ENVVAR = 0, CONF_SOURCE_INI = 1, CONF_SOURCE_HARDCODE = 2,`
 - `CONF_SOURCE_EXTRA_CONF = 3,`
 - `CONF_SOURCE_END }`

Functions

- static gboolean [_parse_bool_string](#) (const gchar *strval, gboolean def)
 - *Parse string to get boolean value.*
- static gchar * [_strdup_getenv](#) (const gchar *name)
 - *Private function to get strdup-ed env-var if it's valid. Otherwise, NULL.*
- static gboolean [_validate_file](#) ([nnsconf_type_path type](#), const gchar *fullpath)
 - *Private function to validate .so file can be added to the list.*
- static gboolean [_get_filenames](#) ([nnsconf_type_path type](#), const gchar *dir, GSList **listF, GSList **listN, guint *counter)
 - *Private function to fill in ".so/.dylib list" with fullpath-filenames in a directory.*
- static gboolean [_get_subplugin_with_type](#) ([nnsconf_type_path type](#), gchar ***name, gchar ***filepath)
 - *Private function to get sub-plugins list with type.*
- static void [_g_list_foreach_vstr_helper](#) (gpointer data, gpointer user_data)
 - *Private function to help convert linked-list to vstr with foreach @data The element data of linked-list @user_data The struct to fill in vstr.*
- static void [_fill_in_vstr](#) (gchar ***fullpath_vstr, gchar ***name_vstr, gchar *searchpath[[CONF_SOURCE_END](#)], [nnsconf_type_path type](#))
 - *Private function to fill in vstr.*
- static void [_fill_subplugin_path](#) ([confdata](#) *cdata, GKeyFile *key_file, [conf_sources](#) src)
 - *Private function to fill subplugin path.*
- gboolean [nnsconf_loadconf](#) (gboolean force_reload)
 - *Public function defined in the header.*
- const gchar * [nnsconf_get_fullpath](#) (const gchar *subpluginname, [nnsconf_type_path type](#))
 - *Public function defined in the header.*
- gboolean [nnsconf_validate_file](#) ([nnsconf_type_path type](#), const gchar *fullpath)

- Public function to validate sub-plugin library is available.*

 - const gchar * [nnsconf_get_subplugin_name_prefix](#) ([nnsconf_type_path](#) type)

Get sub-plugin's name prefix.
- guint [nnsconf_get_subplugin_info](#) ([nnsconf_type_path](#) type, [subplugin_info_s](#) *info)
- Public function to get the list of sub-plugins name and path.*

 - gchar * [nnsconf_get_custom_value_string](#) (const gchar *group, const gchar *key)

Public function defined in the header.
- gboolean [nnsconf_get_custom_value_bool](#) (const gchar *group, const gchar *key, gboolean def)
- Public function defined in the header.*

 - void [nnsconf_dump](#) (gchar *str, gulong size)

Print out configurations.
- static void [_foreach_custom_property](#) (GQuark key_id, gpointer data, gpointer user_data)
- foreach callback for custom property*

 - void [nnsconf_subplugin_dump](#) (gchar *str, gulong size)

Print out the information of registered sub-plugins.

Variables

- static const gchar * [NNSTREAMER_ENVVAR](#) [[NNSCONF_PATH_END](#)]
 - static const gchar * [NNSTREAMER_PATH](#) [[NNSCONF_PATH_END](#)]
 - static const gchar * [subplugin_prefixes](#) []
 - static confdata conf = { 0 }
 - static GHashTable * [custom_table](#) = NULL
- Internal cache for the custom key-values.*

9.88.1 Detailed Description

NNStreamer Configuration (conf file, env-var) Management.

NNStreamer Configurations / Environmental Variable Manager. Copyright (C) 2018 MyungJoo Ham
myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

26 Nov 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.88.2 Macro Definition Documentation

9.88.2.1 NNSTREAMER_PREFIX_CONVERTER

```
#define NNSTREAMER_PREFIX_CONVERTER "libnnstreamer_converter_"
```

Definition at line 42 of file nnstreamer_conf.c.

9.88.2.2 NNSTREAMER_PREFIX_CUSTOMFILTERS

```
#define NNSTREAMER_PREFIX_CUSTOMFILTERS ""
```

Definition at line 41 of file nnstreamer_conf.c.

9.88.2.3 NNSTREAMER_PREFIX_DECODER

```
#define NNSTREAMER_PREFIX_DECODER "libnnstreamer_decoder_"
```

Note that users still can place their custom filters anywhere if they designate them with the full path.

Definition at line 39 of file nnstreamer_conf.c.

9.88.2.4 NNSTREAMER_PREFIX_FILTER

```
#define NNSTREAMER_PREFIX_FILTER "libnnstreamer_filter_"
```

Definition at line 40 of file nnstreamer_conf.c.

9.88.2.5 NNSTREAMER_PREFIX_TRAINER

```
#define NNSTREAMER_PREFIX_TRAINER "libnnstreamer_trainer_"
```

Definition at line 43 of file nnstreamer_conf.c.

9.88.2.6 STR_BOOL

```
#define STR_BOOL(  
    x ) ((x) ? "TRUE" : "FALSE")
```

Definition at line 622 of file nnstreamer_conf.c.

9.88.3 Enumeration Type Documentation

9.88.3.1 conf_sources

```
enum conf_sources
```

Enumerator

CONF_SOURCE_ENVVAR	
CONF_SOURCE_INI	
CONF_SOURCE_HARDCODE	
CONF_SOURCE_EXTRA_CONF	path from extra config file
CONF_SOURCE_END	

Definition at line 73 of file nnstreamer_conf.c.

9.88.4 Function Documentation

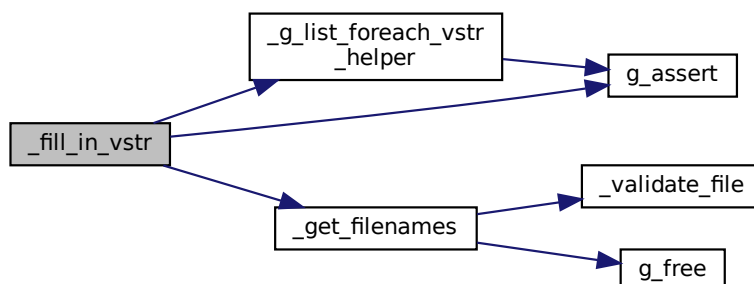
9.88.4.1 _fill_in_vstr()

```
static void _fill_in_vstr (
    gchar *** fullpath_vstr,
    gchar *** name_vstr,
    gchar * searchpath[CONF_SOURCE_END],
    nnsconf_type_path type ) [static]
```

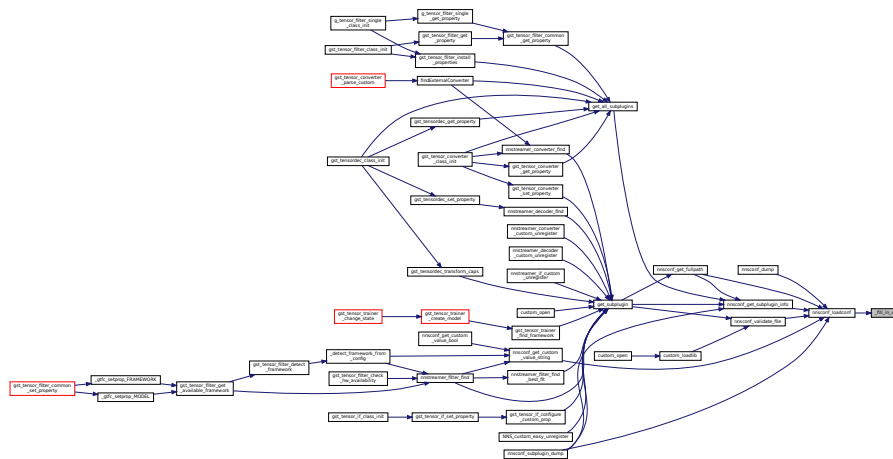
Private function to fill in vstr.

Definition at line 280 of file nnstreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:



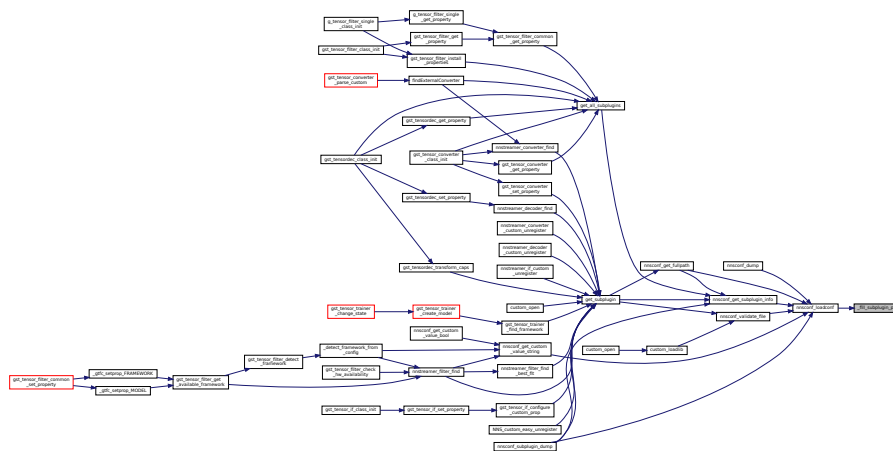
9.88.4.2 _fill_subplugin_path()

```
static void _fill_subplugin_path (
    confdata * cdata,
    GKeyFile * key_file,
    conf_sources src ) [static]
```

Private function to fill subplugin path.

Definition at line 326 of file nntstreamer_conf.c.

Here is the caller graph for this function:



9.88.4.3 `_foreach_custom_property()`

```
static void _foreach_custom_property (
    GQuark key_id,
    gpointer data,
    gpointer user_data ) [static]
```

foreach callback for custom property

Definition at line 681 of file `nnstreamer_conf.c`.

Here is the caller graph for this function:



9.88.4.4 `_g_list_foreach_vstr_helper()`

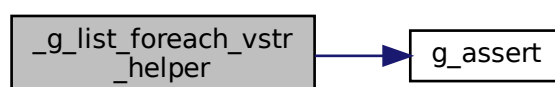
```
static void _g_list_foreach_vstr_helper (
    gpointer data,
    gpointer user_data ) [static]
```

Private function to help convert linked-list to vstr with foreach @data The element data of linked-list @user_data The struct to fill in vstr.

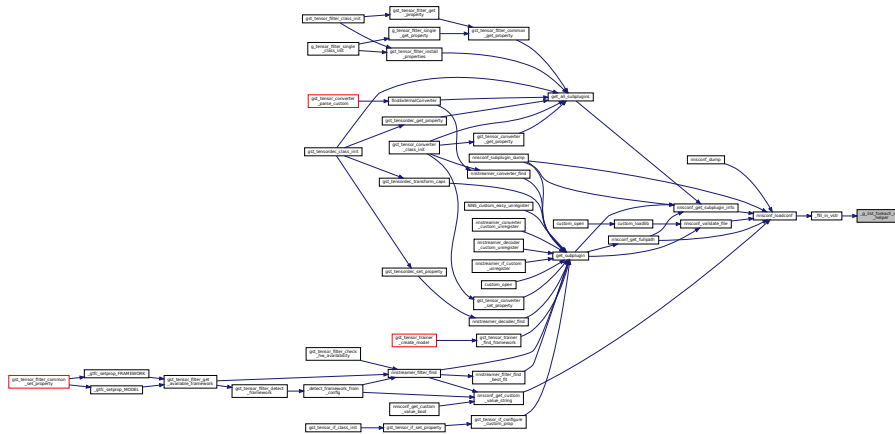
library error? internal logic error?

Definition at line 268 of file `nnstreamer_conf.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.88.4.5 _get_filenames()

```
static gboolean _get_filenames (
    nnsconf_type_path type,
    const gchar * dir,
    GSList ** listF,
    GSList ** listN,
    quint * counter ) [static]
```

Private function to fill in ".so/.dylib list" with fullpath-filenames in a directory.

Parameters

in	<i>type</i>	conf type to scan.
in	<i>dir</i>	Directory to be searched.
	<i>[in/out]</i>	listF The fullpath list to be updated.
	<i>[in/out]</i>	listN The name list to be updated.
	<i>[in/out]</i>	counter increased by the number of appended elements.

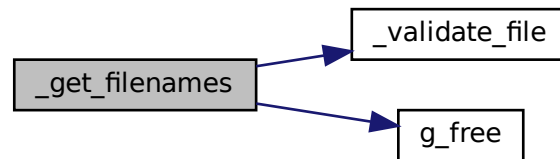
Returns

True if successfully updated.

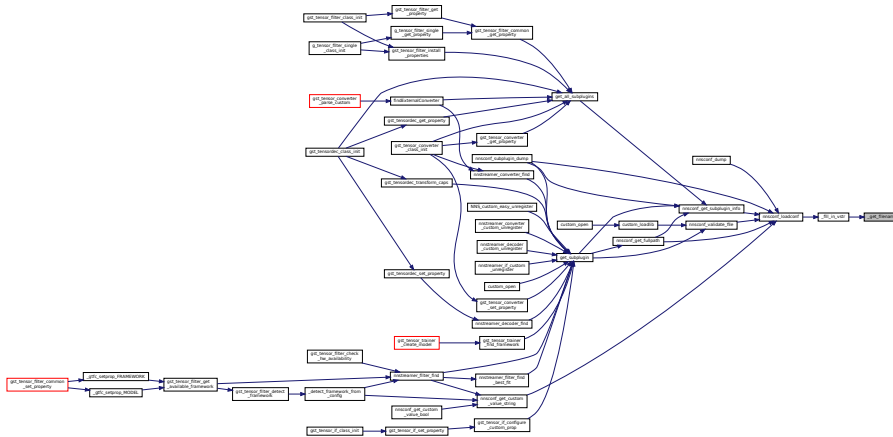
Todo This assumes .so/.dylib for all sub plugins. Support Windows!

Definition at line 183 of file nnsreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.88.4.6 `_get_subplugin_with_type()`

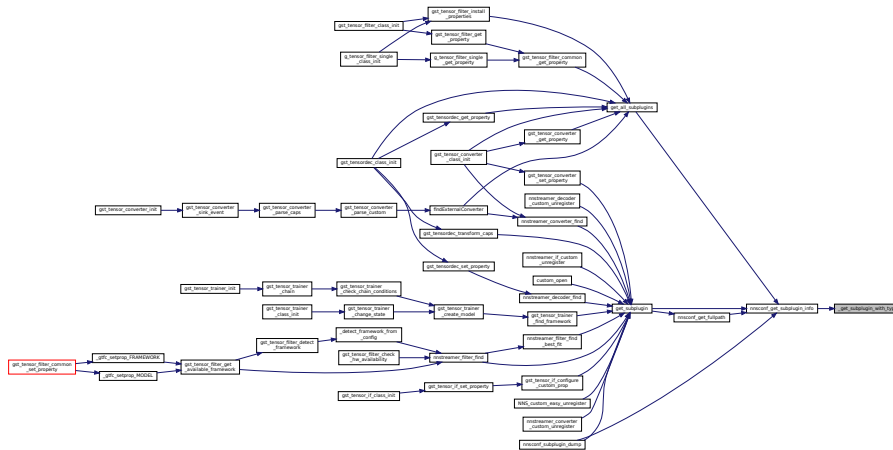
```

static gboolean _get_subplugin_with_type (
    nnsconf_type_path type,
    gchar *** name,
    gchar *** filepath ) [static]
  
```

Private function to get sub-plugins list with type.

Definition at line 229 of file `nnsreamer_conf.c`.

Here is the caller graph for this function:



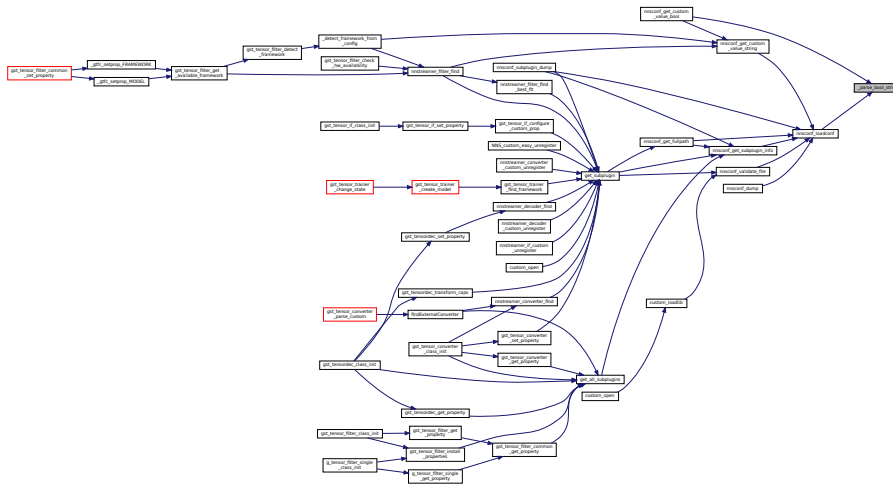
9.88.4.7 _parse_bool_string()

```
static gboolean _parse_bool_string (
    const gchar * strval,
    gboolean def ) [static]
```

Parse string to get boolean value.

Definition at line 115 of file nntstreamer_conf.c.

Here is the caller graph for this function:



9.88.4.8 _strdup_getenv()

```
static gchar* _strdup_getenv (
    const gchar * name ) [static]
```

Private function to get strdup-ed env-var if it's valid. Otherwise, NULL.

Return values

<code>strdup-ed</code>	<code>env-var value</code>
------------------------	----------------------------

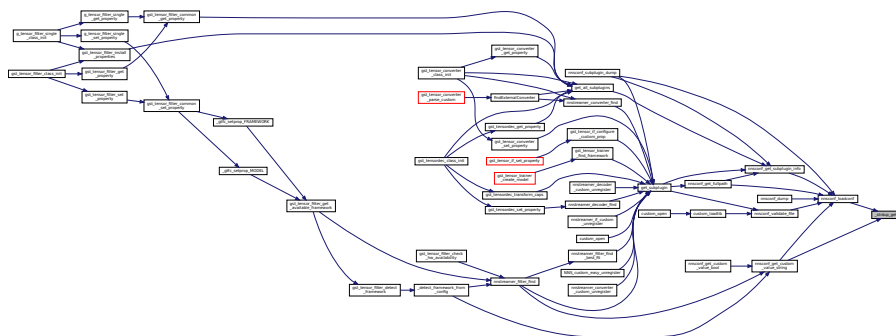
Parameters

<code>in</code>	<code>name</code>	Environmental variable name
-----------------	-------------------	-----------------------------

Todo Evaluate if we need to use `secure_getenv()` here (and compatible with other OS)

Definition at line 143 of file `nstreamer_conf.c`.

Here is the caller graph for this function:



9.88.4.9 `_validate_file()`

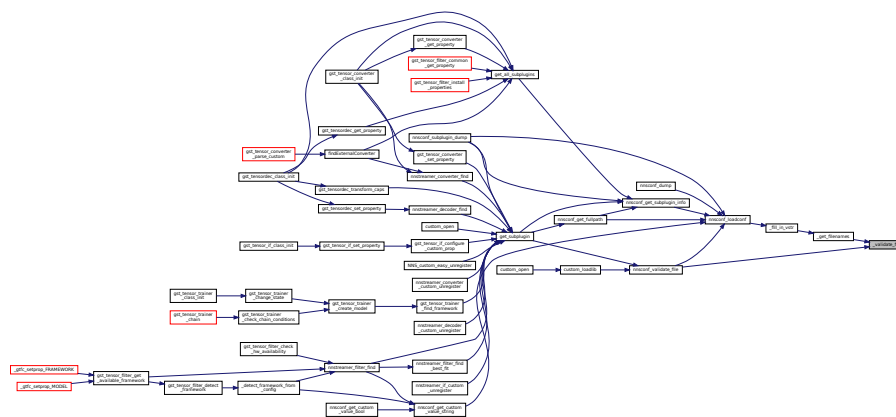
```
static gboolean _validate_file (
    nnsconf_type_path type,
    const gchar * fullpath ) [static]
```

Private function to validate .so file can be added to the list.

Todo how to validate with nnsconf type.

Definition at line 158 of file `nstreamer_conf.c`.

Here is the caller graph for this function:



9.88.4.10 nnsconf_dump()

```
void nnsconf_dump (
    gchar * str,
    gulong size )
```

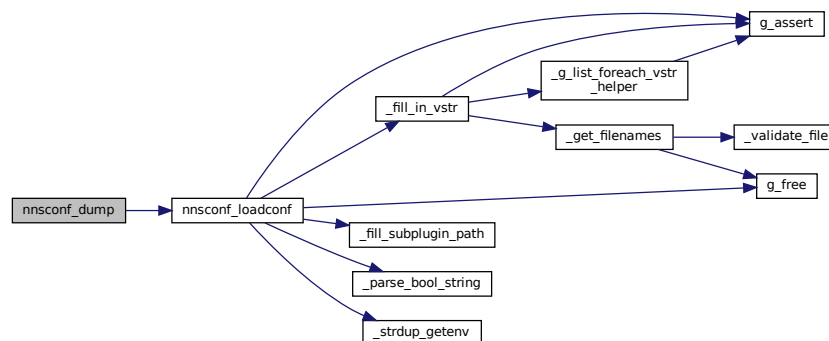
Print out configurations.

NNStreamer configuration dump as string.

Todo Add more configuration values to dump.

Definition at line 628 of file nnsreamer_conf.c.

Here is the call graph for this function:



9.88.4.11 nnsconf_get_custom_value_bool()

```
gboolean nnsconf_get_custom_value_bool (
    const gchar * group,
    const gchar * key,
    gboolean def )
```

Public function defined in the header.

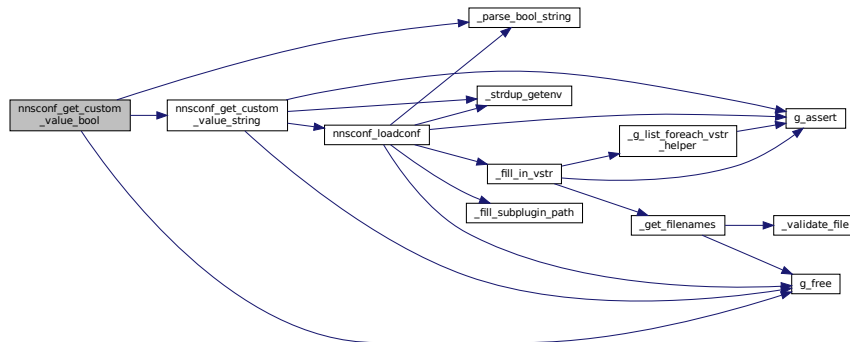
Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be `NNSTREAMER_${group}_${key}`, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Note

This function is included in nntstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nntstreamer_internal.h](#)).

Definition at line 609 of file nntstreamer_conf.c.

Here is the call graph for this function:

**9.88.4.12 nntstreamer_get_custom_value_string()**

```

gchar* nntstreamer_get_custom_value_string (
    const gchar * group,
    const gchar * key )

```

Public function defined in the header.

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

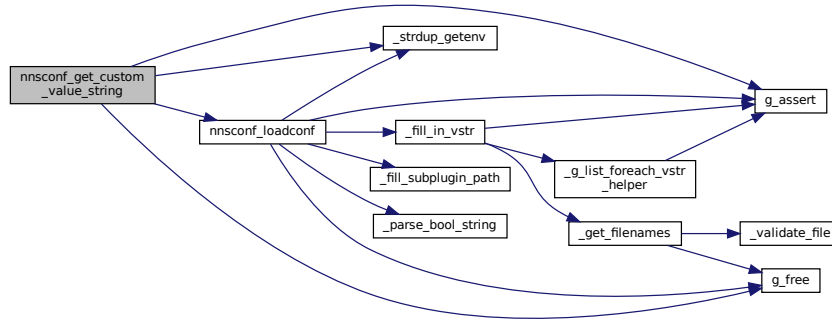
Note

This function is included in nntstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nntstreamer_internal.h](#)).

Internal lib error? out-of-memory?

Definition at line 557 of file nntstreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.88.4.13 nnsconf_get_fullpath()

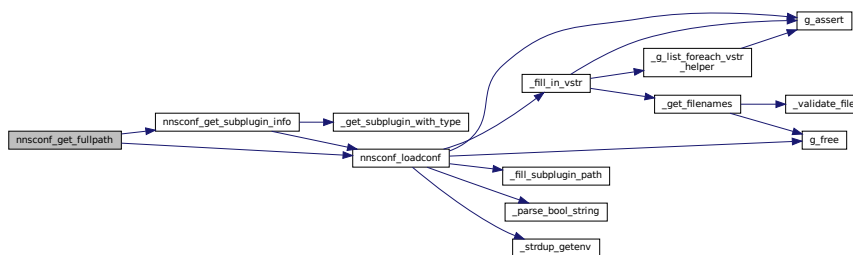
```
const gchar* nnsconf_get_fullpath (
    const gchar * subpluginname,
    nnsconf_type_path type )
```

Public function defined in the header.

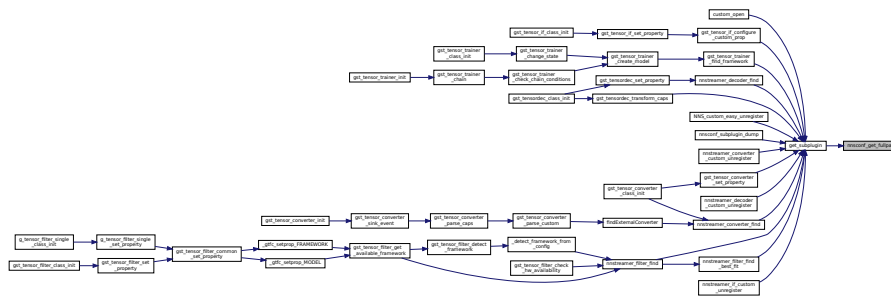
Get the configured paths for the type with sub-plugin name.

Definition at line 483 of file `nnsreamer_conf.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.88.4.14 nnsconf_get_subplugin_info()

```
guint nnsconf_get_subplugin_info (
    nnsconf_type_path type,
    subplugin_info_s * info )
```

Public function to get the list of sub-plugins name and path.

Returns

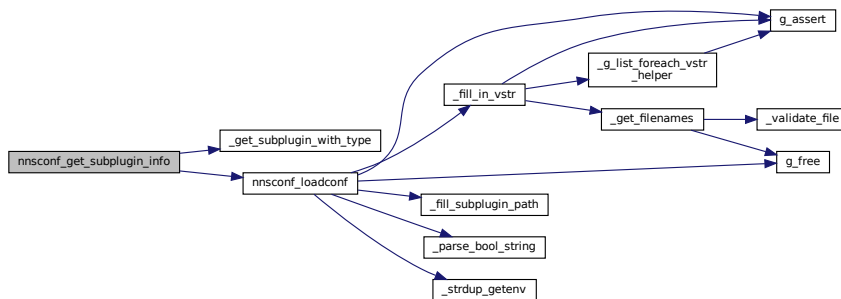
total number of sub-plugins for given type

Note

DO NOT free sub-plugins info

Definition at line 528 of file nnsstreamer_conf.c.

Here is the call graph for this function:

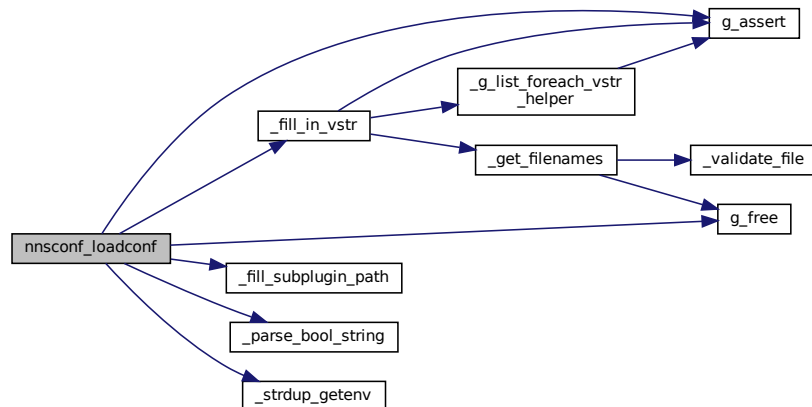


Internal lib error? out-of-memory?

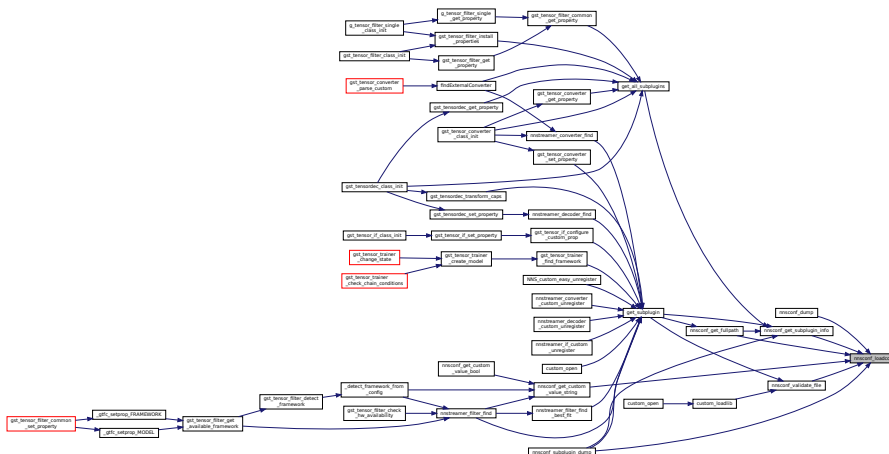
Failed to get the configuration. Note that Android API does not use the configuration.

Definition at line 342 of file nnsreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.88.4.17 nnsconf_subplugin_dump()

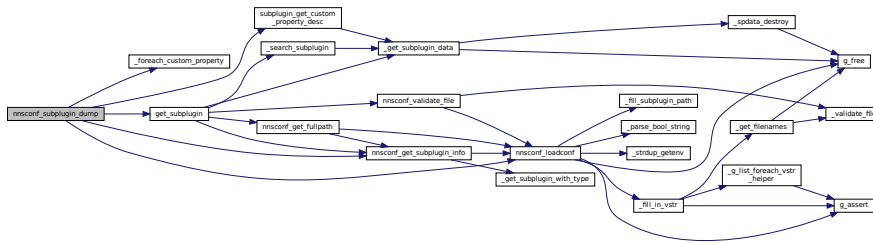
```

void nnsconf_subplugin_dump (
    gchar * str,
    gulong size )
  
```

Print out the information of registered sub-plugins.

Definition at line 695 of file nnsreamer_conf.c.

Here is the call graph for this function:



9.88.4.18 nnsreamer_validate_file()

```
gboolean nnsreamer_validate_file (
    nnsreamer_type_path type,
    const gchar * fullpath )
```

Public function to validate sub-plugin library is available.

Parameters

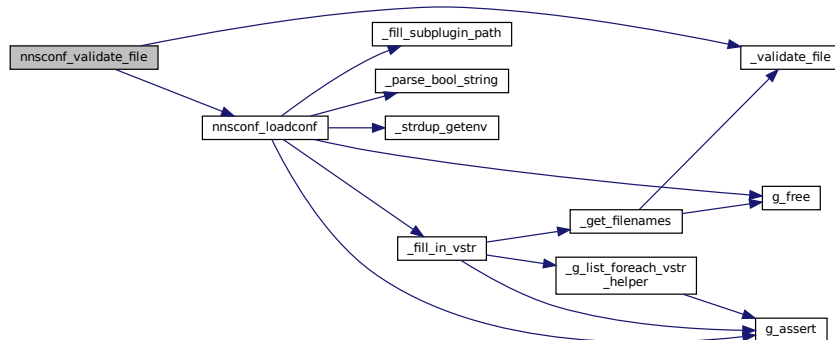
in	<i>type</i>	The type (FILTERS/DECODERS/CUSTOM_FILTERS)
in	<i>fullpath</i>	The full path to the file.

Returns

True if the file is regular and can be added to the list.

Definition at line 503 of file nnsreamer_conf.c.

Here is the call graph for this function:



9.88.5.4 NNSTREAMER_PATH

```
const gchar* NNSTREAMER_PATH[NNSCONF_PATH_END] [static]
```

Initial value:

```
= {  
  [NNSCONF_PATH_FILTERS] = "/usr/lib/nnstreamer/filters/",  
  [NNSCONF_PATH_DECODERS] = "/usr/lib/nnstreamer/decoders/",  
  [NNSCONF_PATH_CUSTOM_FILTERS] = "/usr/lib/nnstreamer/customfilters/",  
  [NNSCONF_PATH_CONVERTERS] = "/usr/lib/nnstreamer/converters/",  
  [NNSCONF_PATH_TRAINERS] = "/usr/lib/nnstreamer/trainers/"  
}
```

Definition at line 55 of file nnstreamer_conf.c.

9.88.5.5 subplugin_prefixes

```
const gchar* subplugin_prefixes[] [static]
```

Initial value:

```
= {  
  [NNSCONF_PATH_FILTERS] = NNSTREAMER_PREFIX_FILTER,  
  [NNSCONF_PATH_DECODERS] = NNSTREAMER_PREFIX_DECODER,  
  [NNSCONF_PATH_CUSTOM_FILTERS] = NNSTREAMER_PREFIX_CUSTOMFILTERS,  
  [NNSCONF_PATH_EASY_CUSTOM_FILTERS] = NNSTREAMER_PREFIX_CUSTOMFILTERS,  
  [NNSCONF_PATH_CONVERTERS] = NNSTREAMER_PREFIX_CONVERTER,  
  [NNSCONF_PATH_TRAINERS] = NNSTREAMER_PREFIX_TRAINER,  
  [NNSCONF_PATH_END] = NULL  
}
```

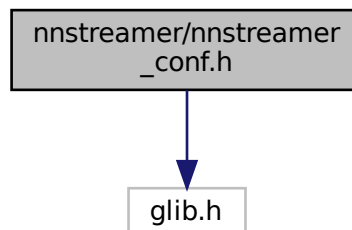
Definition at line 63 of file nnstreamer_conf.c.

9.89 nnstreamer/nnstreamer_conf.h File Reference

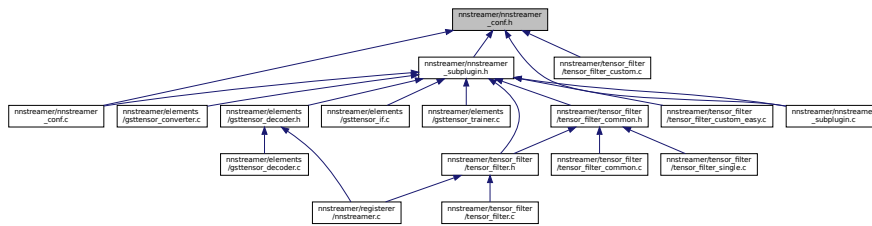
Internal header for conf/env-var management.

```
#include <glib.h>
```

Include dependency graph for nnstreamer_conf.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [subplugin_info_s](#)

Macros

- `#define NNSTREAMER_SYS_ROOT_PATH_PREFIX ""`
- `#define NNSTREAMER_SO_FILE_EXTENSION ".so"`
- `#define NNSTREAMER_DEFAULT_CONF_FILE "/etc/nnstreamer.ini"`
- `#define NNSTREAMER_CONF_FILE NNSTREAMER_DEFAULT_CONF_FILE`
- `#define NNSTREAMER_ENVVAR_CONF_FILE "NNSTREAMER_CONF"`

Enumerations

- enum `nnsconf_type_path` {
`NNSCONF_PATH_FILTERS = 0, NNSCONF_PATH_DECODERS, NNSCONF_PATH_CUSTOM_FILTERS,`
`NNSCONF_PATH_EASY_CUSTOM_FILTERS,`
`NNSCONF_PATH_CONVERTERS, NNSCONF_PATH_TRAINERS, NNSCONF_PATH_END }`

Functions

- gboolean `nnsconf_loadconf` (gboolean force_reload)
Load the .ini file.
- const gchar * `nnsconf_get_fullpath` (const gchar *subpluginname, `nnsconf_type_path` type)
Get the configured paths for the type with sub-plugin name.
- gboolean `nnsconf_validate_file` (`nnsconf_type_path` type, const gchar *fullpath)
Public function to validate sub-plugin library is available.
- const gchar * `nnsconf_get_subplugin_name_prefix` (`nnsconf_type_path` type)
Get sub-plugin's name prefix.
- guint `nnsconf_get_subplugin_info` (`nnsconf_type_path` type, `subplugin_info_s` *info)
Public function to get the list of sub-plugins name and path.
- gchar * `nnsconf_get_custom_value_string` (const gchar *group, const gchar *key)
Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_{group}_{key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).
- gboolean `nnsconf_get_custom_value_bool` (const gchar *group, const gchar *key, gboolean def)

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_{group}_{key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

- void `nnsconf_dump` (gchar *str, gulong size)
NNStreamer configuration dump as string.
- void `nnsconf_subplugin_dump` (gchar *str, gulong size)
Print out the information of registered sub-plugins.

9.89.1 Detailed Description

Internal header for conf/env-var management.

NNStreamer Configurations / Environmental Variable Manager. Copyright (C) 2018 MyungJoo Ham
myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

26 Nov 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

If there are duplicated configurations for the same element, the one with higher priority may override (if it cannot be stacked up),

- (Highest) Environmental variables
- The configuration file (default: /etc/nnstreamer.ini)
- (Lowest) Internal hardcoded values.

Do not export this to devel package. This is an internal header.

9.89.2 Macro Definition Documentation

9.89.2.1 NNSTREAMER_CONF_FILE

```
#define NNSTREAMER_CONF_FILE NNSTREAMER_DEFAULT_CONF_FILE
```

Definition at line 59 of file nnstreamer_conf.h.

9.89.2.2 NNSTREAMER_DEFAULT_CONF_FILE

```
#define NNSTREAMER_DEFAULT_CONF_FILE "/etc/nnstreamer.ini"
```

Definition at line 57 of file nnstreamer_conf.h.

9.89.2.3 NNSTREAMER_ENVVAR_CONF_FILE

```
#define NNSTREAMER_ENVVAR_CONF_FILE "NNSTREAMER_CONF"
```

Definition at line 61 of file nnstreamer_conf.h.

9.89.2.4 NNSTREAMER_SO_FILE_EXTENSION

```
#define NNSTREAMER_SO_FILE_EXTENSION ".so"
```

Hard-coded system-dependent file extension string of shared (dynamic loadable) object

Definition at line 53 of file nnstreamer_conf.h.

9.89.2.5 NNSTREAMER_SYS_ROOT_PATH_PREFIX

```
#define NNSTREAMER_SYS_ROOT_PATH_PREFIX "/"
```

Definition at line 43 of file nnstreamer_conf.h.

9.89.3 Enumeration Type Documentation

9.89.3.1 nnsconf_type_path

```
enum nnsconf_type_path
```

Enumerator

NNSCONF_PATH_FILTERS	
NNSCONF_PATH_DECODERS	
NNSCONF_PATH_CUSTOM_FILTERS	
NNSCONF_PATH_EASY_CUSTOM_FILTERS	
NNSCONF_PATH_CONVERTERS	
NNSCONF_PATH_TRAINERS	
NNSCONF_PATH_END	

Definition at line 63 of file nnsreamer_conf.h.

9.89.4 Function Documentation

9.89.4.1 nnsconf_dump()

```
void nnsconf_dump (
    gchar * str,
    gulong size )
```

NNStreamer configuration dump as string.

Parameters

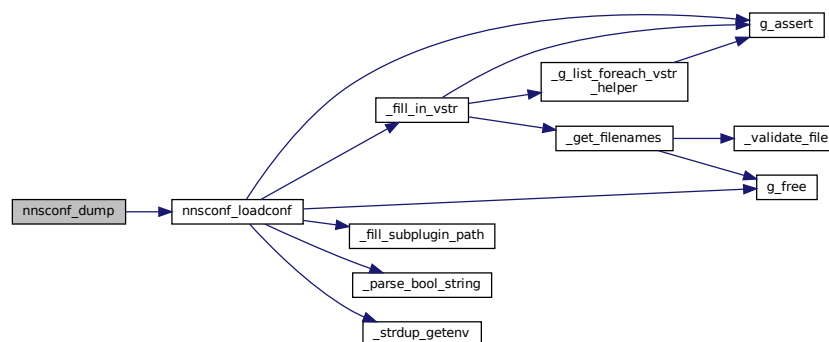
out	<i>str</i>	Preallocated string for the output (dump).
in	<i>size</i>	The size of given str.

NNStreamer configuration dump as string.

Todo Add more configuration values to dump.

Definition at line 628 of file nnsreamer_conf.c.

Here is the call graph for this function:



9.89.4.2 nnsconf_get_custom_value_bool()

```
gboolean nnsconf_get_custom_value_bool (
    const gchar * group,
    const gchar * key,
    gboolean def )
```

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Parameters

in	<i>group</i>	The group name, [group], in .ini file.
in	<i>key</i>	The key name, key = value, in .ini file.
in	<i>def</i>	The default return value in case there is no value available.

Returns

The value interpreted as TRUE/FALSE.

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Note

This function is included in nnstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nnstreamer_internal.h](#)).

Definition at line 609 of file nnstreamer_conf.c.

9.89.4.3 nnsconf_get_custom_value_string()

```
gchar* nnsconf_get_custom_value_string (
    const gchar * group,
    const gchar * key )
```

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Parameters

in	<i>group</i>	The group name, [group], in .ini file.
in	<i>key</i>	The key name, key = value, in .ini file.

Returns

The newly allocated string. A caller must free it. NULL if it's not available.

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Note

This function is included in nntstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nntstreamer_internal.h](#)).

Internal lib error? out-of-memory?

Definition at line 557 of file nntstreamer_conf.c.

9.89.4.4 nnsconf_get_fullpath()

```
const gchar* nnsconf_get_fullpath (
    const gchar * subpluginname,
    nnsconf_type_path type )
```

Get the configured paths for the type with sub-plugin name.

Parameters

in	<i>The</i>	subplugin name except for the prefix and postfix (.so) to find
in	<i>type</i>	The type (FILTERS/DECODERS/CUSTOM_FILTERS)

Returns

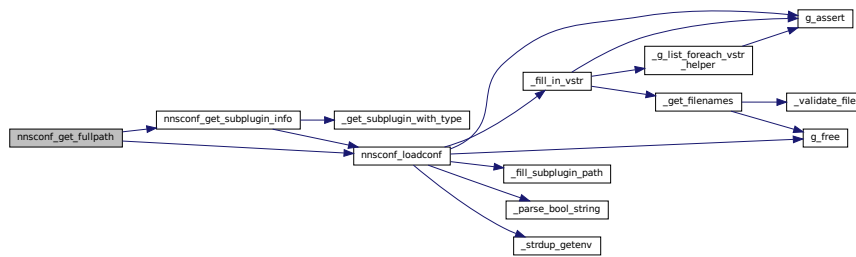
The full path to the file. Caller MUST NOT modify this. Returns NULL if we cannot find the file.

This is mainly supposed to be used by CUSTOM_FILTERS

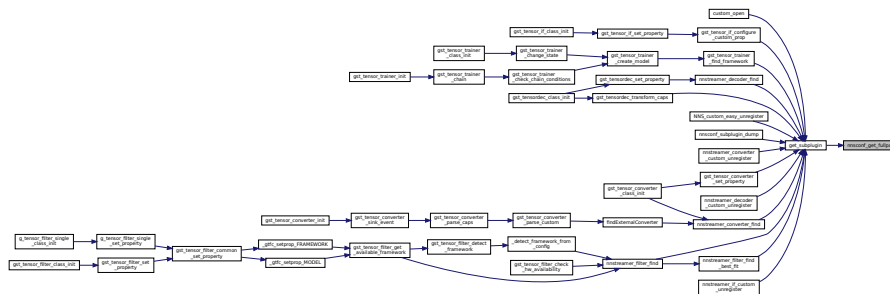
Get the configured paths for the type with sub-plugin name.

Definition at line 483 of file nntstreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.89.4.5 nnsconf_get_subplugin_info()

```
guint nnsconf_get_subplugin_info (
    nnsconf_type_path type,
    subplugin_info_s * info )
```

Public function to get the list of sub-plugins name and path.

Parameters

in	<i>type</i>	The type (FILTERS/DECODERS/CUSTOM_FILTERS)
out	<i>info</i>	The data structure which contains the name and full path of sub-plugins

Returns

total number of sub-plugins for given type

Note

DO NOT free sub-plugins info

Returns

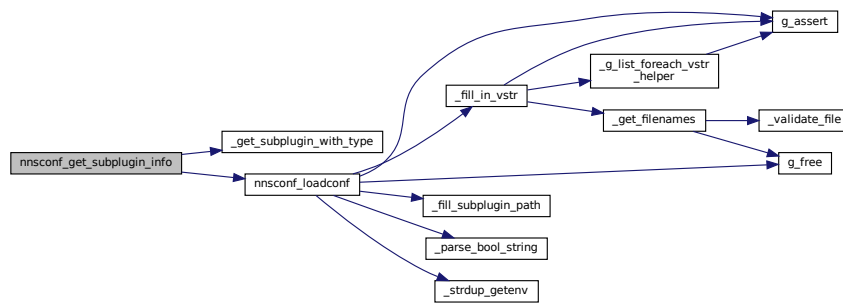
total number of sub-plugins for given type

Note

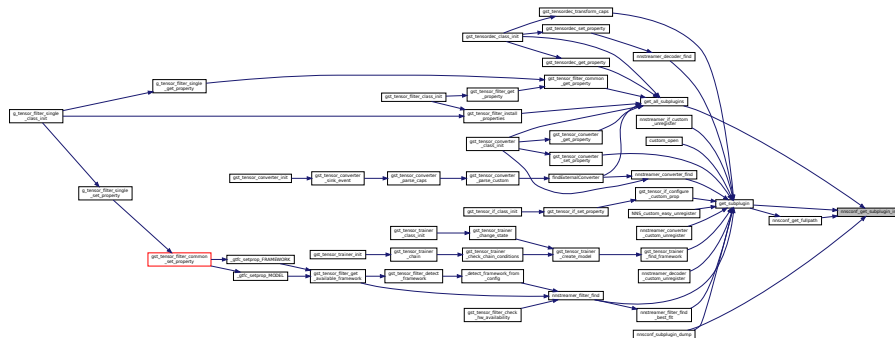
DO NOT free sub-plugins info

Definition at line 528 of file nnsreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.89.4.6 nnsreamer_get_subplugin_name_prefix()

```
const gchar* nnsreamer_get_subplugin_name_prefix (
    nnsreamer_type_path type )
```

Get sub-plugin's name prefix.

Parameters

in	type	The type (FILTERS/DECODERS/CUSTOM_FILTERS)
----	------	--

Returns

Predefined prefix string for given type.

Definition at line 516 of file nntstreamer_conf.c.

9.89.4.7 nnsconf_loadconf()

```
gboolean nnsconf_loadconf (
    gboolean force_reload )
```

Load the .ini file.

Parameters

in	<i>force_reload</i>	TRUE if you want to clean up and load conf again.
----	---------------------	---

Returns

TRUE if successful or skipped. FALSE if error reading something.

Load the .ini file. if it's not Tizen, configuration from env-var has a higher priority

Priority of reading a conf file 1) read from NNSTREAMER_CONF_FILE 2) read from NNSTREAMER_DEFAULT_CONF_FILE 3) read from env-var

default value of 'sysconfdir' in meson is 'etc'

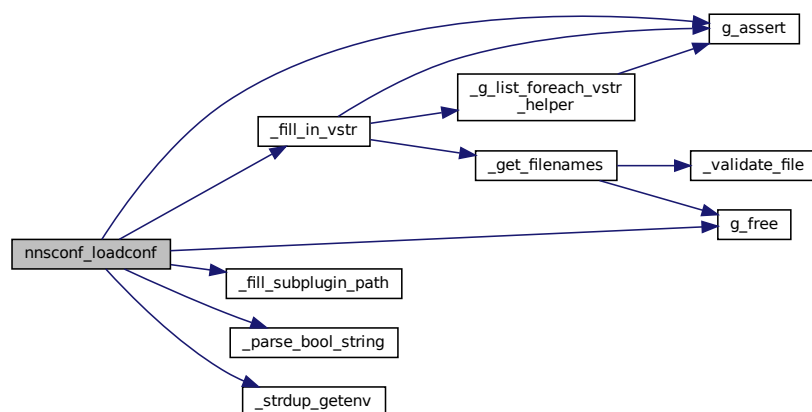
Internal lib error? out-of-memory?

Internal lib error? out-of-memory?

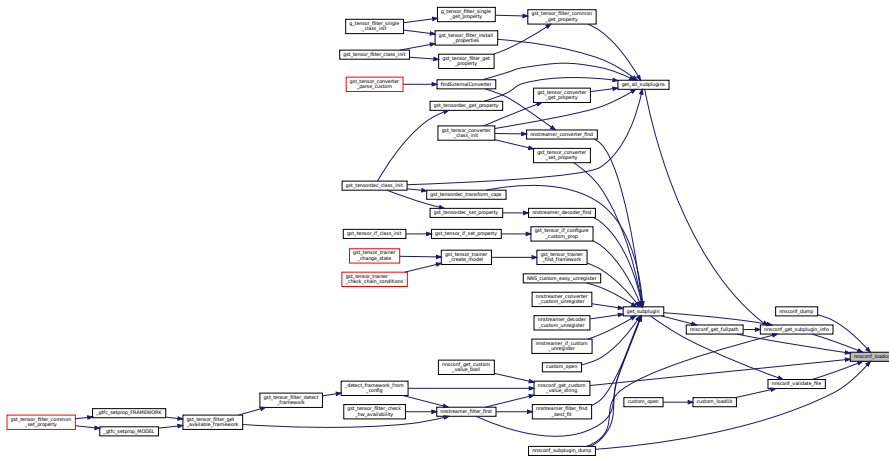
Failed to get the configuration. Note that Android API does not use the configuration.

Definition at line 342 of file nntstreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:



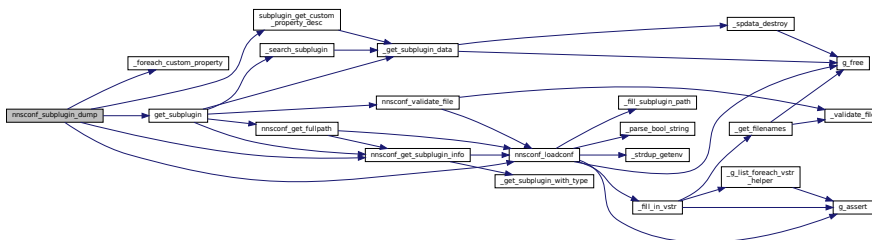
9.89.4.8 nnsconf_subplugin_dump()

```
void nnsconf_subplugin_dump (
    gchar * str,
    gulong size )
```

Print out the information of registered sub-plugins.

Definition at line 695 of file nnsreamer_conf.c.

Here is the call graph for this function:



9.89.4.9 nnsconf_validate_file()

```
gboolean nnsconf_validate_file (
    nnsconf_type_path type,
    const gchar * fullpath )
```

Public function to validate sub-plugin library is available.

Parameters

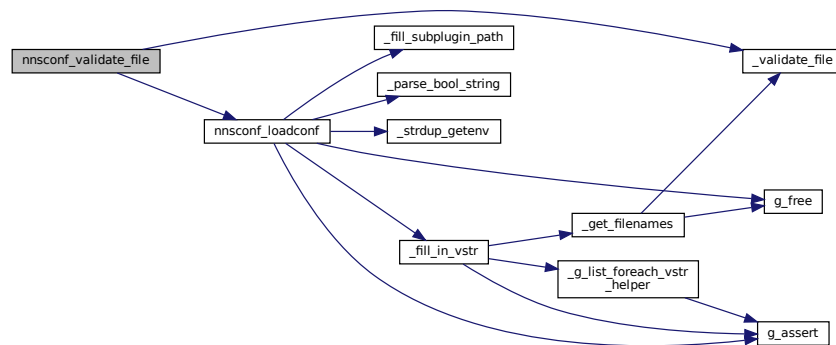
in	<i>type</i>	The type (FILTERS/DECODERS/CUSTOM_FILTERS)
in	<i>fullpath</i>	The full path to the file.

Returns

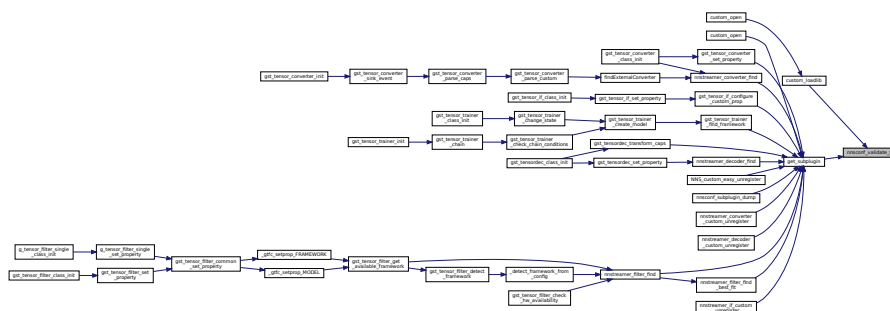
True if the file is regular and can be added to the list.

Definition at line 503 of file `nstreamer_conf.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

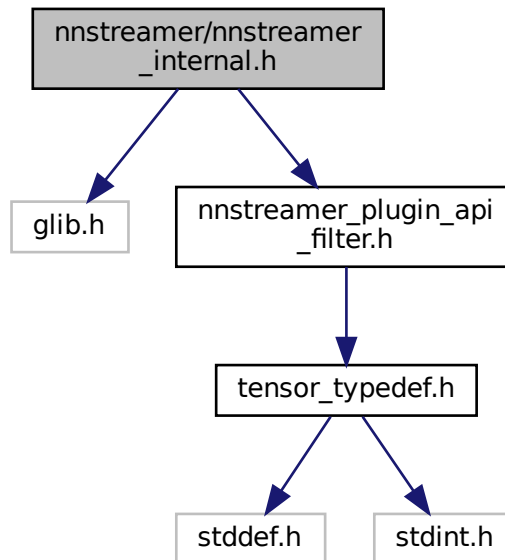


9.90 nstreamer/nstreamer_internal.h File Reference

Internal header for NNStreamer plugins and native single-shot APIs.

```
#include <glib.h>
#include <nnstreamer_plugin_api_filter.h>
```

Include dependency graph for nnstreamer_internal.h:



Functions

- `G_BEGIN_DECLS` `gchar * nnsconf_get_custom_value_string` (const `gchar *group`, const `gchar *key`)

Get the custom configuration value from `.ini` and `envvar`. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The `ENVVAR` is supposed to be `NNSTREAMER_${group}_${key}`, which has higher priority than the `.ini` configuration. Be careful not to use special characters in group name (`[,], _`).

- `gboolean nnsconf_get_custom_value_bool` (const `gchar *group`, const `gchar *key`, `gboolean def`)

Get the custom configuration value from `.ini` and `envvar`. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The `ENVVAR` is supposed to be `NNSTREAMER_${group}_${key}`, which has higher priority than the `.ini` configuration. Be careful not to use special characters in group name (`[,], _`).

- `gchar * gst_tensor_filter_detect_framework` (const `gchar *const *model_files`, const `guint num_models`, const `gboolean load_conf`)

Get neural network framework name from given model file. This does not guarantee the framework is available on the target device.

- `gboolean gst_tensor_filter_check_hw_availability` (const `gchar *name`, const `accl_hw hw`, const `char *custom`)

Check if the given `hw` is supported by the framework.

9.90.1 Detailed Description

Internal header for NNStreamer plugins and native single-shot APIs.

Copyright (c) 2021 Samsung Electronics Co., Ltd. All Rights Reserved.

Date

28 Jan 2021

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung jy1210.jung@samsung.com

Bug No known bugs except for NYI items

9.90.2 Function Documentation

9.90.2.1 `gst_tensor_filter_check_hw_availability()`

```
gboolean gst_tensor_filter_check_hw_availability (
    const gchar * name,
    const accl_hw hw,
    const char * custom )
```

Check if the given hw is supported by the framework.

Parameters

in	<i>name</i>	The name of filter sub-plugin.
in	<i>hw</i>	Backend accelerator hardware.
in	<i>custom</i>	User-defined string to handle detailed hardware option.

Returns

TRUE if given hw is available.

Note

This function is included in nnstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nnstreamer_internal.h](#)).

Only check for specific HW, DEFAULT/AUTO are always supported

Definition at line 2923 of file tensor_filter_common.c.

9.90.2.2 gst_tensor_filter_detect_framework()

```
gchar* gst_tensor_filter_detect_framework (
    const gchar *const * model_files,
    const guint num_models,
    const gboolean load_conf )
```

Get neural network framework name from given model file. This does not guarantee the framework is available on the target device.

Parameters

in	<i>model_files</i>	the prediction model paths
in	<i>num_models</i>	the number of model files
in	<i>load_conf</i>	flag to load configuration for the priority of framework

Returns

Possible framework name (NULL if it fails to detect automatically). Caller should free returned value using [g_free\(\)](#).

Parameters

in	<i>model_files</i>	the prediction model paths
in	<i>num_models</i>	the number of model files
in	<i>load_conf</i>	flag to load configuration for the priority of framework

Returns

Possible framework name (NULL if it fails to detect automatically). Caller should free returned value using [g_free\(\)](#).

Note

This function is included in nnstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nnstreamer_internal.h](#)).

Note

When adding new file extension for auto fw detection, you should check the condition to validate model file - `ml_validate_model_file()` in ML-API.

Definition at line 1232 of file tensor_filter_common.c.

9.90.2.3 nnsconf_get_custom_value_bool()

```
gboolean nnsconf_get_custom_value_bool (
    const gchar * group,
    const gchar * key,
    gboolean def )
```

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Parameters

in	<i>group</i>	The group name, [group], in .ini file.
in	<i>key</i>	The key name, key = value, in .ini file.
in	<i>def</i>	The default return value in case there is no value available.

Returns

The value interpreted as TRUE/FALSE.

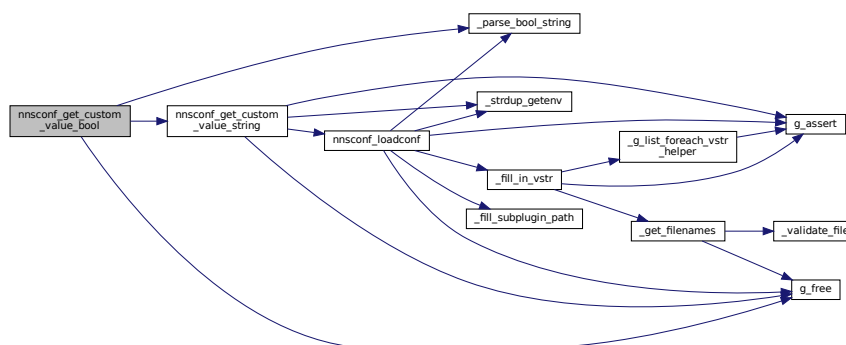
Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Note

This function is included in nnstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nnstreamer_internal.h](#)).

Definition at line 609 of file nnstreamer_conf.c.

Here is the call graph for this function:



9.90.2.4 nnsconf_get_custom_value_string()

```
G_BEGIN_DECLS
gchar* nnsconf_get_custom_value_string (
    const gchar * group,
    const gchar * key )
```

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

Parameters

in	group	The group name, [group], in .ini file.
in	key	The key name, key = value, in .ini file.

Returns

The newly allocated string. A caller must free it. NULL if it's not available.

Get the custom configuration value from .ini and envvar. @detail For predefined configurations defined in this header, use the given enum for faster configuration processing. For custom configurations not defined in this header, you may use this API to access your own custom configurations. Configuration values may be loaded only once during runtime, thus, if the values are changed in run-time, the changes are not guaranteed to be reflected. The ENVVAR is supposed to be NNSTREAMER_\${group}_\${key}, which has higher priority than the .ini configuration. Be careful not to use special characters in group name ([,], _).

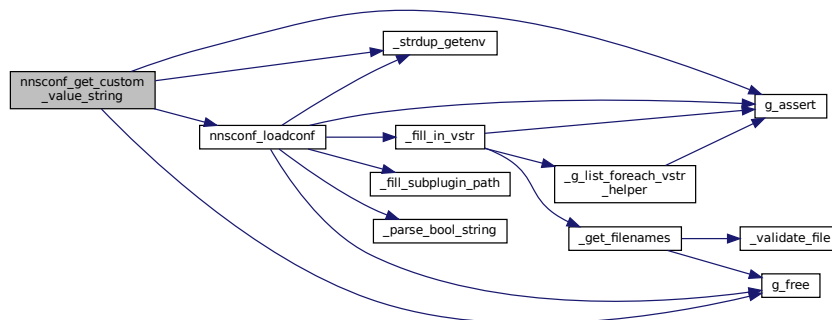
Note

This function is included in nnsreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nnsreamer_internal.h](#)).

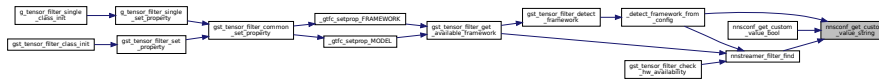
Internal lib error? out-of-memory?

Definition at line 557 of file nnsreamer_conf.c.

Here is the call graph for this function:



Here is the caller graph for this function:

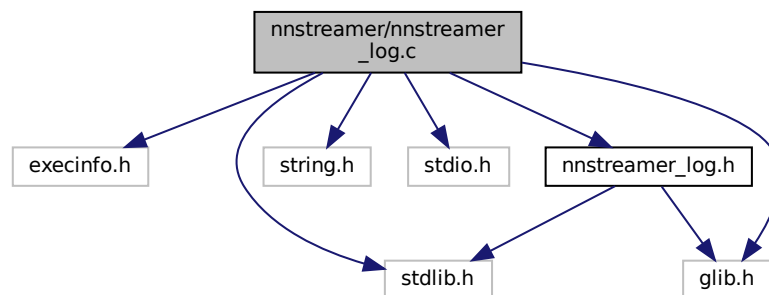


9.91 nnstreamer/nnstreamer_log.c File Reference

Internal log and error handling for NNStreamer plugins and core codes.

```
#include <execinfo.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <glib.h>
#include "nnstreamer_log.h"
```

Include dependency graph for nnstreamer_log.c:



Macros

- `#define _NNSTREAMER_ERROR_LENGTH (4096U)`

Functions

- `char * _backtrace_to_string (void)`
stack trace as a string for error messages
- `G_LOCK_DEFINE_STATIC (errlock)`
- `const char * _nnstreamer_error (void)`
return the last internal error string and clean it.
- `__attribute__ ((__format__ (__printf__, 1, 2)))`
overwrites the error message buffer with the new message.
- `void _nnstreamer_error_clean (void)`
cleans up the error message buffer.

Variables

- static char `errmsg` [`_NNSTREAMER_ERROR_LENGTH`] = { 0 }
- static int `errmsg_reported` = 0

9.91.1 Detailed Description

Internal log and error handling for NNStreamer plugins and core codes.

Date

31 Mar 2022

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.91.2 Macro Definition Documentation

9.91.2.1 `_NNSTREAMER_ERROR_LENGTH`

```
#define _NNSTREAMER_ERROR_LENGTH (4096U)
```

Definition at line 70 of file `nnstreamer_log.c`.

9.91.3 Function Documentation

9.91.3.1 `__attribute__()`

```
__attribute__ (  
    (__format__ (__printf__, 1, 2)) )
```

overwrites the error message buffer with the new message.

The attribute is for clang workaround in macos: <https://stackoverflow.com/questions/20167124/vsprintf-a>

Definition at line 97 of file `nnstreamer_log.c`.

9.91.3.2 `_backtrace_to_string()`

```
char* _backtrace_to_string (  
    void )
```

stack trace as a string for error messages

Returns

a string of stacktrace result. caller should free it.

Todo The .c file location of this function might be not appropriate.

Definition at line 35 of file `nnstreamer_log.c`.

9.91.3.3 `_nnstreamer_error()`

```
const char* _nnstreamer_error (  
    void )
```

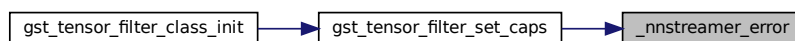
return the last internal error string and clean it.

Returns

a string of error. Do not free the returned string.

Definition at line 81 of file `nnstreamer_log.c`.

Here is the caller graph for this function:



9.91.3.4 `_nnstreamer_error_clean()`

```
void _nnstreamer_error_clean (  
    void )
```

cleans up the error message buffer.

Definition at line 120 of file `nnstreamer_log.c`.

9.91.3.5 G_LOCK_DEFINE_STATIC()

```
G_LOCK_DEFINE_STATIC (  
    errlock )
```

9.91.4 Variable Documentation

9.91.4.1 errmsg

```
char errmsg[_NNSTREAMER_ERROR_LENGTH] = { 0 } [static]
```

Definition at line 71 of file nnstreamer_log.c.

9.91.4.2 errmsg_reported

```
int errmsg_reported = 0 [static]
```

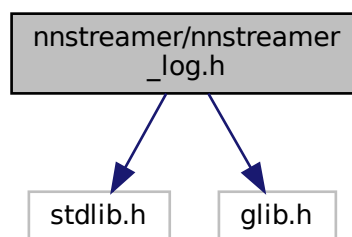
Definition at line 73 of file nnstreamer_log.c.

9.92 nnstreamer/nnstreamer_log.h File Reference

Internal log util for NNStreamer plugins and native APIs.

```
#include <stdlib.h>  
#include <glib.h>
```

Include dependency graph for nnstreamer_log.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define `TAG_NAME` "nnstreamer"
- #define `ml_logi` `g_info`
- #define `ml_logw` `g_warning`
- #define `ml_loge` `g_critical`
- #define `ml_logd` `g_debug`
- #define `ml_logf` `g_error`
- #define `GST_ELEMENT_ERROR_BTRACE`(s, errtype, errcode, mesg)
- #define `ml_logf_stacktrace`(...)
- #define `ml_log_stacktrace`(logfunc, ...)
- #define `ml_loge_stacktrace`(...) `ml_log_stacktrace`(`ml_loge`, `__VA_ARGS__`)
- #define `nns_logi` `ml_logi`
- #define `nns_logw` `ml_logw`
- #define `nns_loge` `ml_loge`
- #define `nns_logd` `ml_logd`
- #define `nns_logf` `ml_logf`

Functions

- `char * _backtrace_to_string` (void)
stack trace as a string for error messages
- `const char * _nnstreamer_error` (void)
return the last internal error string and clean it.
- `void _nnstreamer_error_write` (const char *fmt,...)
overwrites the error message buffer with the new message.
- `void _nnstreamer_error_clean` (void)
cleans up the error message buffer.

9.92.1 Detailed Description

Internal log util for NNStreamer plugins and native APIs.

Copyright (c) 2020 Samsung Electronics Co., Ltd. All Rights Reserved.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

12 Mar 2020

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung [jy1210.jung@samsung.com](mailto: jy1210.jung@samsung.com)

Bug No known bugs except for NYI items

9.92.2 Macro Definition Documentation

9.92.2.1 GST_ELEMENT_ERROR_BTRACE

```
#define GST_ELEMENT_ERROR_BTRACE(  
    s,  
    errtype,  
    errcode,  
    msg )
```

Value:

```
do { \  
char *btrace = _backtrace_to_string (); \  
if (btrace) { \  
    GST_ELEMENT_ERROR (s, errtype, errcode, msg, ("%s", btrace)); \  
    free (btrace); \  
} \  
} while (0)
```

Definition at line 94 of file nnstreamer_log.h.

9.92.2.2 ml_log_stacktrace

```
#define ml_log_stacktrace(  
    logfunc,  
    ... )
```

Value:

```
do { \  
char *btrace = _backtrace_to_string (); \  
if (btrace) { \  
    logfunc ("%s\n", btrace); \  
    free (btrace); \  
} \  
logfunc (__VA_ARGS__); \  
} while (0)
```

Definition at line 111 of file nnstreamer_log.h.

9.92.2.3 ml_logd

```
#define ml_logd g_debug
```

Definition at line 79 of file nnstreamer_log.h.

9.92.2.4 ml_loge

```
#define ml_loge g_critical
```

Definition at line 78 of file nnstreamer_log.h.

9.92.2.5 ml_loge_stacktrace

```
#define ml_loge_stacktrace(  
    ... ) ml_log_stacktrace(ml_loge, __VA_ARGS__)
```

Definition at line 119 of file nnstreamer_log.h.

9.92.2.6 ml_logf

```
#define ml_logf g_error
```

Definition at line 80 of file nnstreamer_log.h.

9.92.2.7 ml_logf_stacktrace

```
#define ml_logf_stacktrace(  
    ... )
```

Value:

```
do { \  
    char *btrace = _backtrace_to_string (); \  
    if (btrace) { \  
        ml_loge ("%s\n", btrace); \  
        free (btrace); \  
    } \  
    ml_logf (__VA_ARGS__); \  
} while (0)
```

Definition at line 102 of file nnstreamer_log.h.

9.92.2.8 ml_logi

```
#define ml_logi g_info
```

Definition at line 76 of file nnstreamer_log.h.

9.92.2.9 ml_logw

```
#define ml_logw g_warning
```

Definition at line 77 of file nnstreamer_log.h.

9.92.2.10 nns_logd

```
#define nns_logd ml_logd
```

Definition at line 143 of file nnstreamer_log.h.

9.92.2.11 nns_loge

```
#define nns_loge ml_loge
```

Definition at line 142 of file nnstreamer_log.h.

9.92.2.12 nns_logf

```
#define nns_logf ml_logf
```

Definition at line 144 of file nnstreamer_log.h.

9.92.2.13 nns_logi

```
#define nns_logi ml_logi
```

Definition at line 140 of file nnstreamer_log.h.

9.92.2.14 nns_logw

```
#define nns_logw ml_logw
```

Definition at line 141 of file nnstreamer_log.h.

9.92.2.15 TAG_NAME

```
#define TAG_NAME "nnstreamer"
```

Definition at line 28 of file nnstreamer_log.h.

9.92.3 Function Documentation

9.92.3.1 _backtrace_to_string()

```
char* _backtrace_to_string (  
    void )
```

stack trace as a string for error messages

Returns

- a string of stacktrace result. caller should free it.
- a string of stacktrace result. caller should free it.

Todo The .c file location of this function might be not appropriate.

Definition at line 35 of file nnstreamer_log.c.

9.92.3.2 _nnstreamer_error()

```
const char* _nnstreamer_error (  
    void )
```

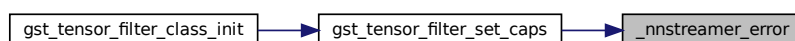
return the last internal error string and clean it.

Returns

- a string of error. Do not free the returned string.

Definition at line 81 of file nnstreamer_log.c.

Here is the caller graph for this function:



9.92.3.3 _nnstreamer_error_clean()

```
void _nnstreamer_error_clean (
    void )
```

cleans up the error message buffer.

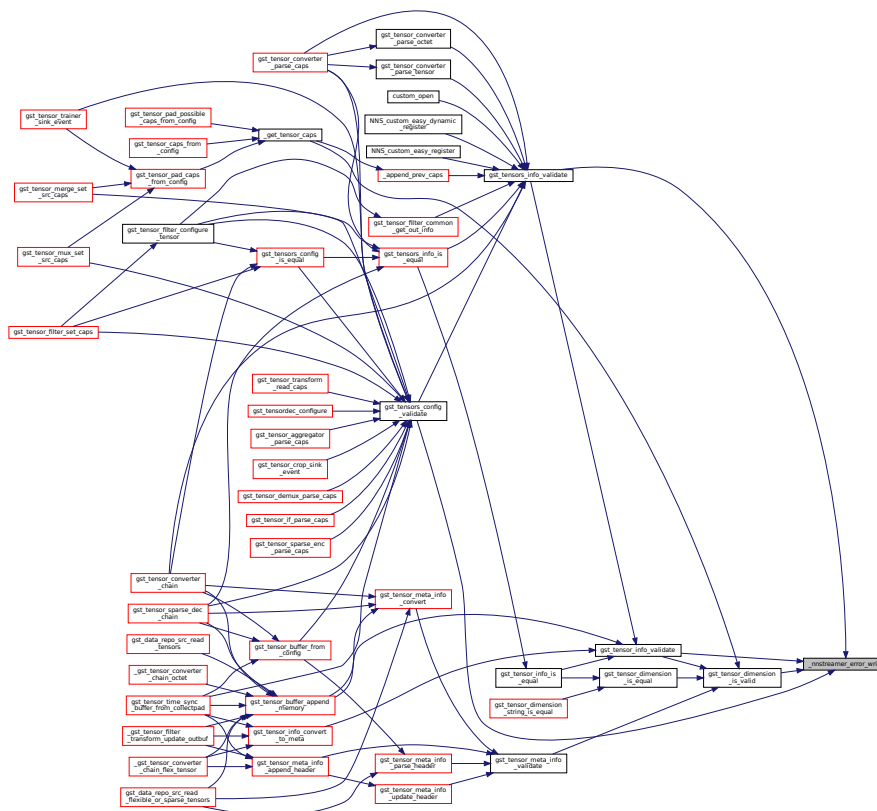
Definition at line 120 of file nnstreamer_log.c.

9.92.3.4 _nnstreamer_error_write()

```
void _nnstreamer_error_write (
    const char * fmt,
    ... )
```

overwrites the error message buffer with the new message.

Here is the caller graph for this function:

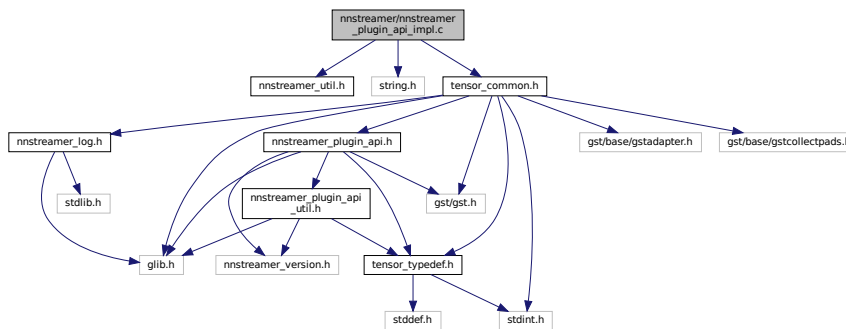


9.93 nnstreamer/nnstreamer_plugin_api_impl.c File Reference

Common data for NNStreamer, the GStreamer plugin for neural networks.

```
#include <nnstreamer_util.h>
#include <string.h>
#include <tensor_common.h>
```

Include dependency graph for nnstreamer_plugin_api_impl.c:



Classes

- struct [GstTensorExtraInfo](#)
Data structure to describe a "extra" tensor data. This represents the information of the NNS_TENSOR_SIZE_LIMIT-th memory block for tensor stream.
- struct [gst_tensor_aggregation_data_s](#)
Internal struct to handle aggregation data in hash table.

Macros

- #define [NNS_TENSOR_RANK_LIMIT_PREV](#) (4)
The old rank of tensor.
- #define [NNS_TENSOR_EXTRA_MAGIC](#) 0xf00dc0de
- #define [AGGREGATION_DEFAULT_KEY](#) 0xC0FFEEU

Functions

- static gboolean [gst_memory_map_is_extra_tensor](#) (GstMapInfo *map)
Check if given memory has extra tensors.
- static void [gst_tensor_extra_info_init](#) (GstTensorExtraInfo *extra, gsize reserved_size)
Initialize [GstTensorExtraInfo](#) structure with given memory.
- [tensor_time_sync_mode](#) [gst_tensor_time_sync_get_mode](#) (const gchar *str)
Get the corresponding mode from the string value.
- const gchar * [gst_tensor_time_sync_get_mode_string](#) (tensor_time_sync_mode mode)
Get the time-sync mode string.
- gboolean [gst_tensor_time_sync_set_option_data](#) (tensor_time_sync_data *sync)
Setup time sync option.

- static gboolean [_gst_tensor_time_sync_is_eos](#) (GstCollectPads *collect, [tensor_time_sync_data](#) *sync, guint empty)
Internal function to detect EOS using the number of empty pads.
- gboolean [gst_tensor_time_sync_get_current_time](#) (GstCollectPads *collect, [tensor_time_sync_data](#) *sync, GstClockTime *current_time, GstBuffer *tensors_buf)
A function call to decide current timestamp among collected pads based on PTS. It will decide current timestamp according to sync option. GstMeta is also copied with same sync mode.
- void [gst_tensor_time_sync_flush](#) (GstCollectPads *collect)
A function to be called while processing a flushing event. It should clear old buffer and reset pad data.
- static gboolean [_gst_tensor_time_sync_buffer_update](#) (GstCollectPads *collect, GstCollectData *data, GstClockTime current, GstClockTime base, [tensor_time_sync_data](#) *sync)
Internal function to update buffer in pad data based on the sync mode.
- gboolean [gst_tensor_time_sync_buffer_from_collectpad](#) (GstCollectPads *collect, [tensor_time_sync_data](#) *sync, GstClockTime current_time, GstBuffer *tensors_buf, [GstTensorsConfig](#) *configs, gboolean *is_eos)
A function call to make tensors from collected pads. It decide which buffer is going to be used according to sync option.
- GstBuffer * [gst_tensor_buffer_from_config](#) (GstBuffer *in, [GstTensorsConfig](#) *config)
Configure gst-buffer with tensors information. NNStreamer handles single memory chunk as single tensor. If incoming buffer has invalid memories, separate it and generate new gst-buffer using tensors information. Note that this function always takes the ownership of input buffer.
- static void [gst_tensor_aggregation_free_data](#) (gpointer data)
Internal function to free aggregation data.
- static [gst_tensor_aggregation_data_s](#) * [gst_tensor_aggregation_add_data](#) (GHashTable *table, const guint32 key)
Internal function to add new aggregation data.
- static [gst_tensor_aggregation_data_s](#) * [gst_tensor_aggregation_get_data](#) (GHashTable *table, const guint32 key)
Internal function to get aggregation data.
- static void [gst_tensor_aggregation_clear_internal](#) (gpointer key, gpointer value, gpointer user_data)
Internal function to remove all buffers from aggregation data.
- GHashTable * [gst_tensor_aggregation_init](#) (void)
Gets new hash table for tensor aggregation.
- void [gst_tensor_aggregation_clear](#) (GHashTable *table, const guint32 key)
Clears buffers from adapter.
- void [gst_tensor_aggregation_clear_all](#) (GHashTable *table)
Clears buffers from all adapters in hash table.
- GstAdapter * [gst_tensor_aggregation_get_adapter](#) (GHashTable *table, const guint32 key)
Gets adapter from hash table.
- static gboolean [_append_prev_caps](#) (const [GstTensorsConfig](#) *config)
Internal function to check tensor dimensions to append old caps for backward compatibility (rank 4).
- static GstCaps * [_get_tensor_caps](#) (const [GstTensorsConfig](#) *config)
Internal function to get caps for single tensor from config.
- static GstCaps * [_get_tensors_caps](#) (const [GstTensorsConfig](#) *config)
Internal function to get caps for multi tensors from config.
- static GstCaps * [_get_flexible_caps](#) (const [GstTensorsConfig](#) *config)
Internal function to get caps for flexible tensor from config.
- gboolean [gst_structure_is_tensor_stream](#) (const GstStructure *structure)
Check given mimetype is tensor stream.
- [media_type](#) [gst_structure_get_media_type](#) (const GstStructure *structure)
Get media type from structure.
- gboolean [gst_tensors_config_from_peer](#) (GstPad *pad, [GstTensorsConfig](#) *config, gboolean *is_fixed)
Parse caps from peer pad and set tensors config.

- static gboolean `_is_structure_dimension_same` (GstStructure *st1, GstStructure *st2, const gchar *fieldname)
Check whether two structures have the same dimension.
- void `gst_tensor_caps_update_dimension` (GstCaps *caps, GstCaps *filter)
Update caps dimensions for negotiation.
- gboolean `gst_tensor_caps_can_intersect` (GstCaps *caps1, GstCaps *caps2)
Try intersecting @caps1 and @caps2 for tensor stream.
- GstCaps * `gst_tensor_pad_caps_from_config` (GstPad *pad, const GstTensorsConfig *config)
Get pad caps from tensors config and caps of the peer connected to the pad.
- GstCaps * `gst_tensor_pad_possible_caps_from_config` (GstPad *pad, const GstTensorsConfig *config)
Get all possible caps from tensors config. Unlike `gst_tensor_pad_caps_from_config()`, this function does not check peer caps.
- tensor_format `gst_tensor_pad_get_format` (GstPad *pad)
Get tensor format of current pad caps.
- GstCaps * `gst_tensors_caps_from_config` (const GstTensorsConfig *config)
Get caps from tensors config (for other/tensors)
- GstCaps * `gst_tensor_caps_from_config` (const GstTensorsConfig *config)
Get tensor caps from tensors config.
- gboolean `gst_tensors_config_from_structure` (GstTensorsConfig *config, const GstStructure *structure)
Parse structure and set tensors config (for other/tensors)
- gboolean `gst_tensor_meta_info_parse_memory` (GstTensorMetaInfo *meta, GstMemory *mem)
Parse memory and fill the tensor meta.
- GstMemory * `gst_tensor_meta_info_append_header` (GstTensorMetaInfo *meta, GstMemory *mem)
Append header to memory.
- GstMemory * `gst_tensor_buffer_get_nth_memory` (GstBuffer *buffer, const guint index)
Get the nth GstMemory from given buffer.
- gboolean `gst_tensor_buffer_append_memory` (GstBuffer *buffer, GstMemory *memory, const GstTensorInfo *info)
Append memory to given buffer.
- guint `gst_tensor_buffer_get_count` (GstBuffer *buffer)
Get the number of tensors in the buffer.
- static void `set_property_value` (GValue *prop_value, const GParamSpec *param_spec, const gchar *property_value)
Sets the value of a property based on the specified property value and GParamSpec.
- void `gst_tensor_parse_config_file` (const gchar *config_path, const GObject *object)
Parses a configuration file and sets the corresponding properties on a GObject.

Variables

- static const gchar * `gst_tensor_time_sync_mode_string` []

9.93.1 Detailed Description

Common data for NNStreamer, the GStreamer plugin for neural networks.

NNStreamer Common Header's Contents (pipeline extension) Copyright (C) 2020 MyungJoo Ham myungjoo.ham@samsung.com

Date

14 Apr 2020

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.93.2 Macro Definition Documentation

9.93.2.1 AGGREGATION_DEFAULT_KEY

```
#define AGGREGATION_DEFAULT_KEY 0xC0FFEEU
```

Definition at line 662 of file nnstreamer_plugin_api_impl.c.

9.93.2.2 NNS_TENSOR_EXTRA_MAGIC

```
#define NNS_TENSOR_EXTRA_MAGIC 0xf00dc0de
```

Definition at line 33 of file nnstreamer_plugin_api_impl.c.

9.93.2.3 NNS_TENSOR_RANK_LIMIT_PREV

```
#define NNS_TENSOR_RANK_LIMIT_PREV (4)
```

The old rank of tensor.

Definition at line 31 of file nnstreamer_plugin_api_impl.c.

9.93.3 Function Documentation

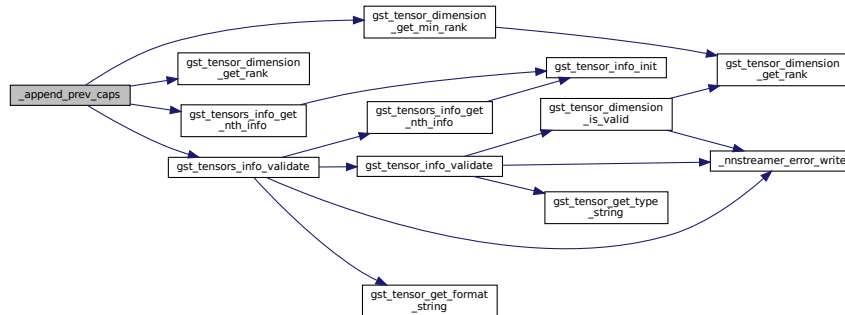
9.93.3.1 `_append_prev_caps()`

```
static gboolean _append_prev_caps (
    const GstTensorsConfig * config ) [static]
```

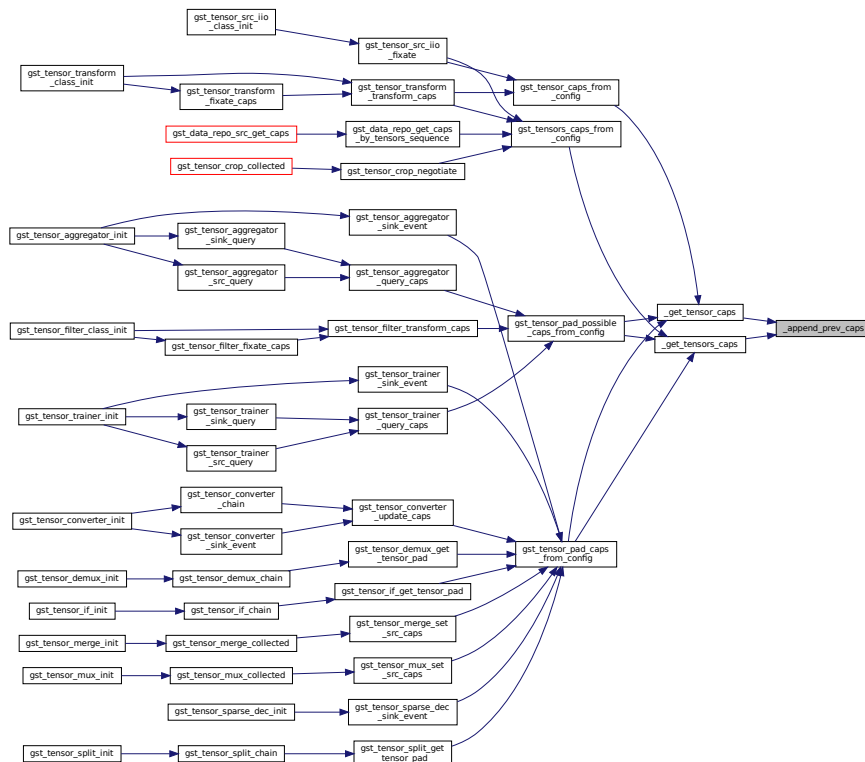
Internal function to check tensor dimensions to append old caps for backward compatibility (rank 4).

Definition at line 809 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



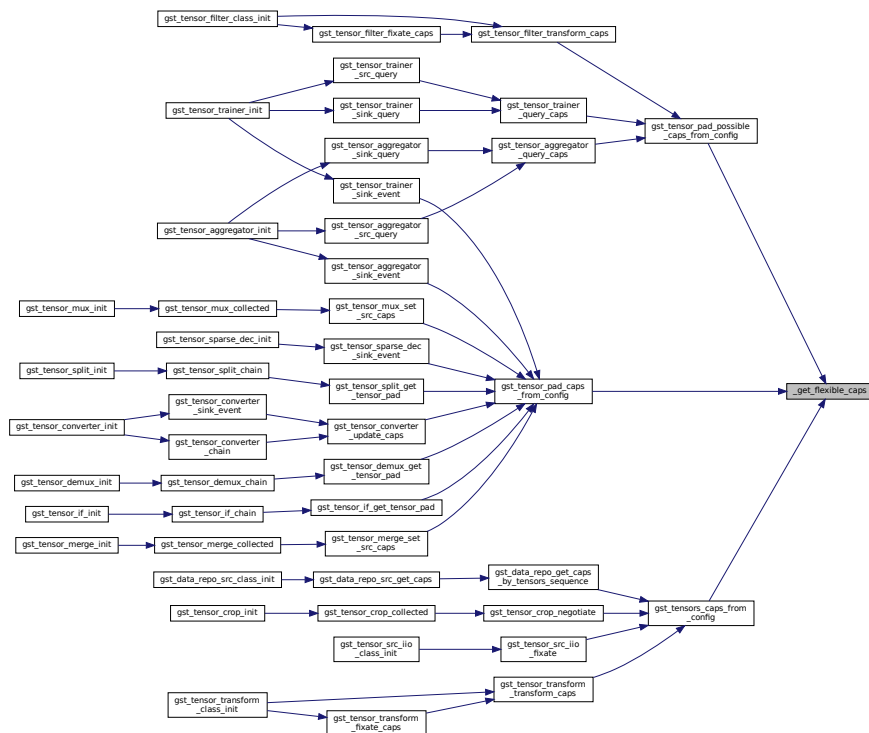
9.93.3.2 `_get_flexible_caps()`

```
static GstCaps* _get_flexible_caps (
    const GstTensorsConfig * config ) [static]
```

Internal function to get caps for flexible tensor from config.

Definition at line 964 of file nntstreamer_plugin_api_impl.c.

Here is the caller graph for this function:



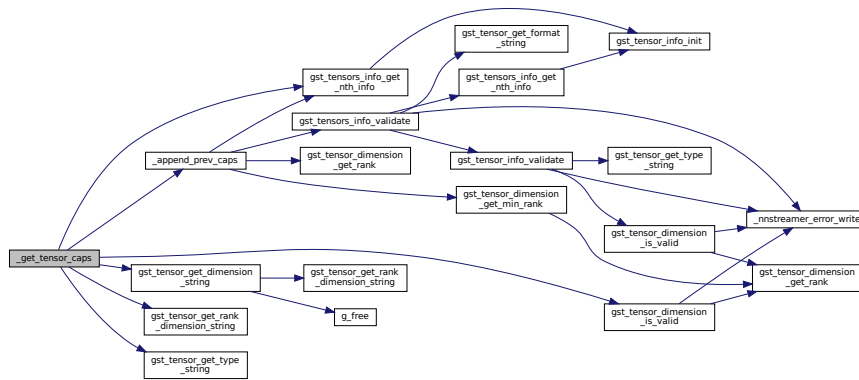
9.93.3.3 `_get_tensor_caps()`

```
static GstCaps* _get_tensor_caps (
    const GstTensorsConfig * config ) [static]
```

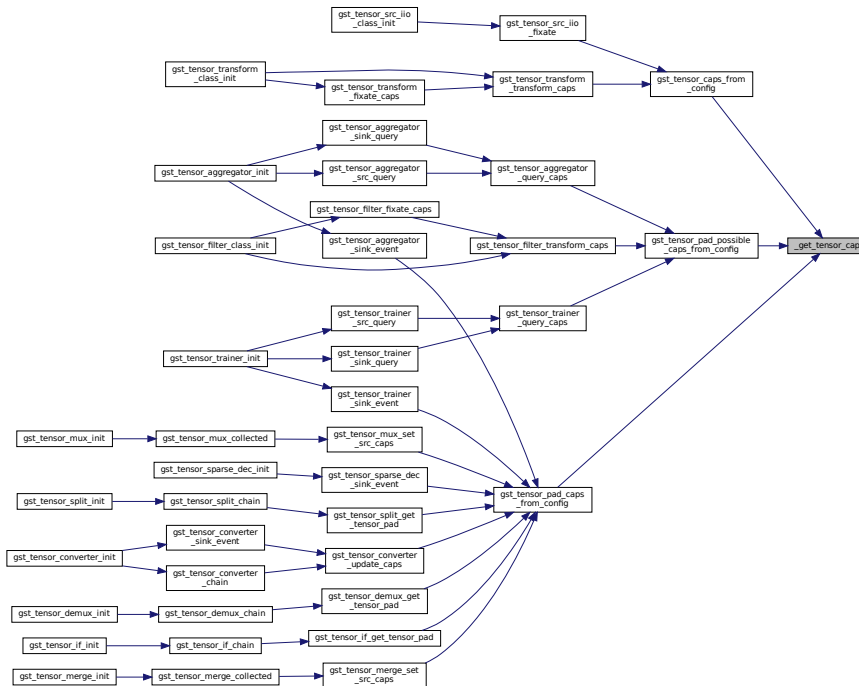
Internal function to get caps for single tensor from config.

Definition at line 839 of file nntstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.4 `_get_tensors_caps()`

```
static GstCaps* _get_tensors_caps (
    const GstTensorsConfig * config ) [static]
```

Internal function to get caps for multi tensors from config.

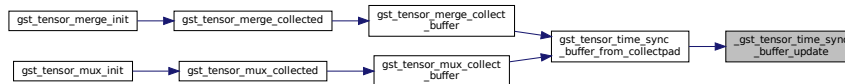
Definition at line 904 of file `nnstreamer_plugin_api_impl.c`.


```
GstClockTime current,
  GstClockTime base,
  tensor_time_sync_data * sync ) [static]
```

Internal function to update buffer in pad data based on the sync mode.

Definition at line 287 of file nnstreamer_plugin_api_impl.c.

Here is the caller graph for this function:



9.93.3.6 _gst_tensor_time_sync_is_eos()

```
static gboolean _gst_tensor_time_sync_is_eos (
  GstCollectPads * collect,
  tensor_time_sync_data * sync,
  guint empty ) [static]
```

Internal function to detect EOS using the number of empty pads.

Parameters

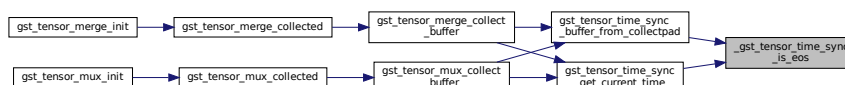
in	<i>collect</i>	Collect pad.
in	<i>sync</i>	Synchronization option.
in	<i>empty</i>	The number of empty pads (pad has no buffer).

Returns

True if EOS.

Definition at line 175 of file nnstreamer_plugin_api_impl.c.

Here is the caller graph for this function:

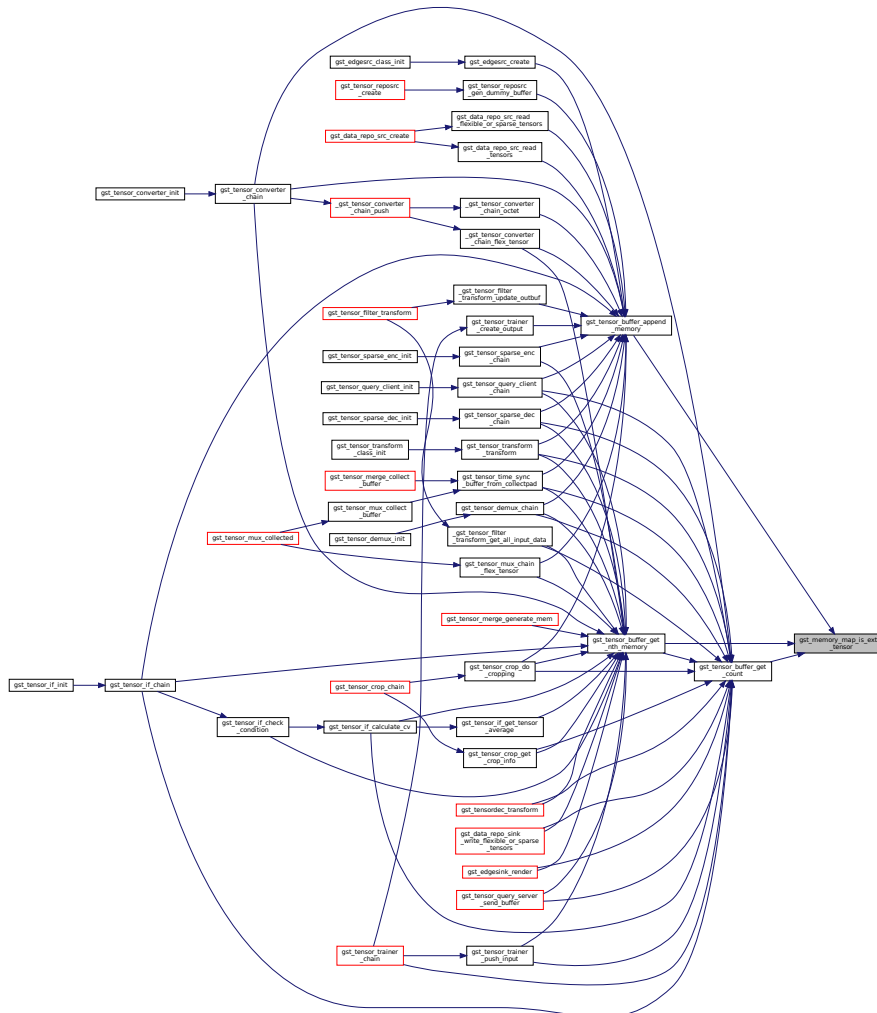


Returns

TRUE if @map has extra tensors, otherwise FALSE.

Definition at line 54 of file nnstreamer_plugin_api_impl.c.

Here is the caller graph for this function:



9.93.3.9 gst_structure_get_media_type()

```
media_type gst_structure_get_media_type (
    const GstStructure * structure )
```

Get media type from structure.

Parameters

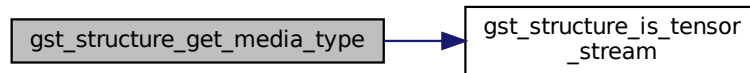
<i>structure</i>	structure to be interpreted
------------------	-----------------------------

Returns

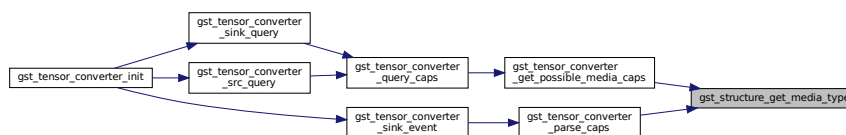
corresponding media type (returns `_NNS_MEDIA_INVALID` for unsupported type)

Definition at line 1001 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.93.3.10 gst_structure_is_tensor_stream()**

```
gboolean gst_structure_is_tensor_stream (
    const GstStructure * structure )
```

Check given mimetype is tensor stream.

Parameters

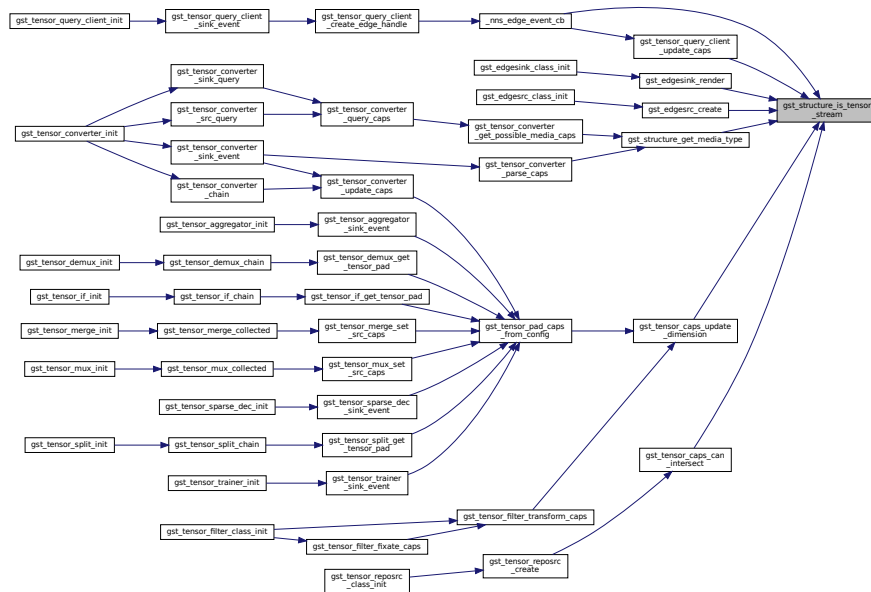
<i>structure</i>	structure to be interpreted
------------------	-----------------------------

Returns

TRUE if mimetype is tensor stream

Definition at line 984 of file `nnstreamer_plugin_api_impl.c`.

Here is the caller graph for this function:



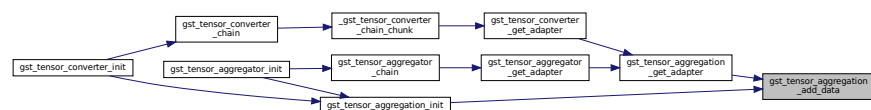
9.93.3.11 `gst_tensor_aggregation_add_data()`

```
static gst_tensor_aggregation_data_s* gst_tensor_aggregation_add_data (
    GHashTable * table,
    const guint32 key ) [static]
```

Internal function to add new aggregation data.

Definition at line 685 of file `nnstreamer_plugin_api_impl.c`.

Here is the caller graph for this function:



9.93.3.12 `gst_tensor_aggregation_clear()`

```
void gst_tensor_aggregation_clear (
    GHashTable * table,
    const guint32 key )
```

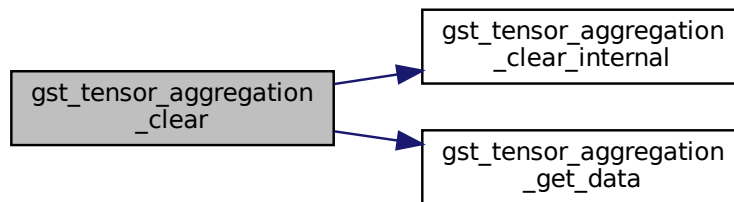
Clears buffers from adapter.

Parameters

<i>table</i>	a hash table instance initialized with gst_tensor_aggregation_init()
<i>key</i>	the key to look up (set null to get default adapter)

Definition at line 763 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



9.93.3.13 [gst_tensor_aggregation_clear_all\(\)](#)

```
void gst_tensor_aggregation_clear_all (
    GHashTable * table )
```

Clears buffers from all adapters in hash table.

Parameters

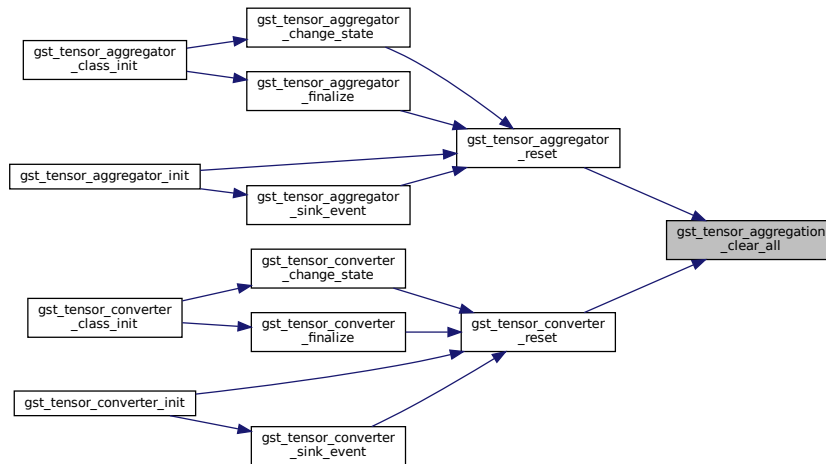
<i>table</i>	a hash table instance initialized with gst_tensor_aggregation_init()
--------------	--

Definition at line 778 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.14 `gst_tensor_aggregation_clear_internal()`

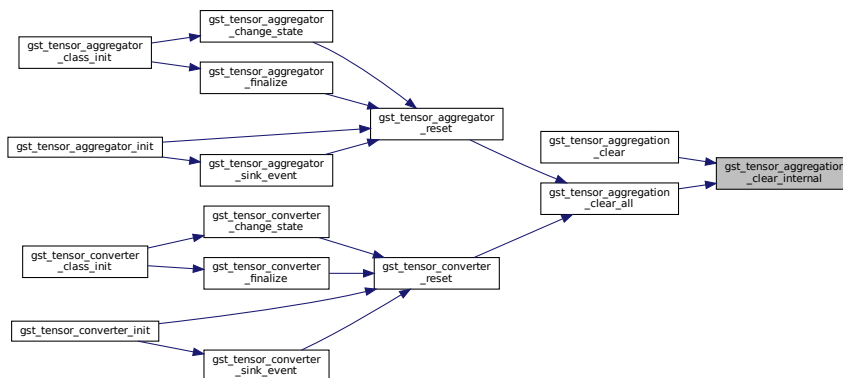
```

static void gst_tensor_aggregation_clear_internal (
    gpointer key,
    gpointer value,
    gpointer user_data ) [static]
  
```

Internal function to remove all buffers from aggregation data.

Definition at line 718 of file `nnstreamer_plugin_api_impl.c`.

Here is the caller graph for this function:



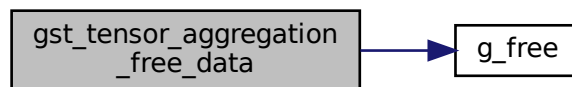
9.93.3.15 `gst_tensor_aggregation_free_data()`

```
static void gst_tensor_aggregation_free_data (
    gpointer data ) [static]
```

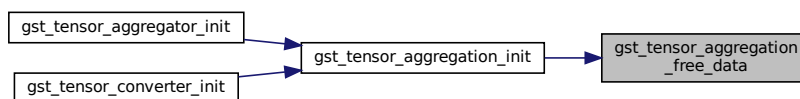
Internal function to free aggregation data.

Definition at line 668 of file `nntstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.16 `gst_tensor_aggregation_get_adapter()`

```
GstAdapter* gst_tensor_aggregation_get_adapter (
    GHashTable * table,
    const guint32 key )
```

Gets adapter from hash table.

Parameters

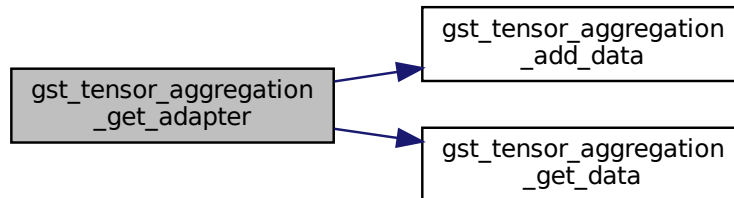
<i>table</i>	a hash table instance initialized with gst_tensor_aggregation_init()
<i>key</i>	the key to look up (set null to get default adapter)

Returns

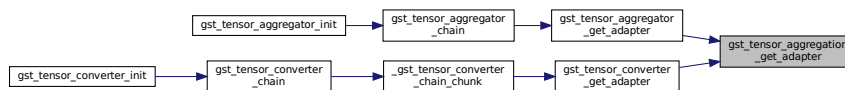
gst-adapter instance. DO NOT release this instance.

Definition at line 790 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.17 gst_tensor_aggregation_get_data()

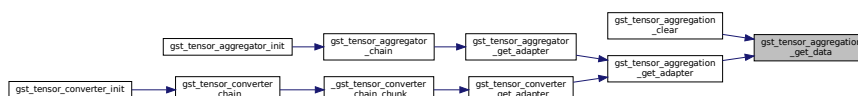
```

static gst_tensor_aggregation_data_s* gst_tensor_aggregation_get_data (
    GHashTable * table,
    const guint32 key ) [static]
  
```

Internal function to get aggregation data.

Definition at line 706 of file nnstreamer_plugin_api_impl.c.

Here is the caller graph for this function:



9.93.3.18 `gst_tensor_aggregation_init()`

```
GHashTable* gst_tensor_aggregation_init (
    void )
```

Gets new hash table for tensor aggregation.

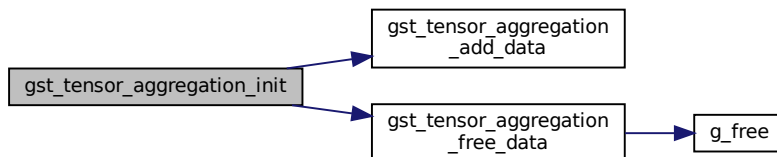
Returns

Newly allocated hash table, caller should release this using `g_hash_table_destroy()`.

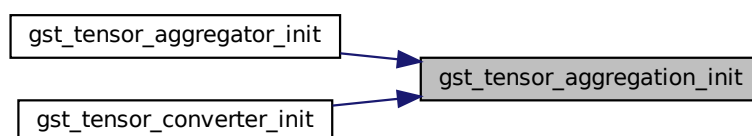
Add default adapter (for the case if buffer has no specific id). If `gst-buffer` has `tensor-meta` which includes client-id, e.g., aggregation frames from multiple clients on query-server pipeline, `nntstreamer` element should parse meta and request adapter with this id. However, on normal pipeline, `gst-buffer` does not contain `tensor-meta`, then the element may request adapter with null key string.

Definition at line 737 of file `nntstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

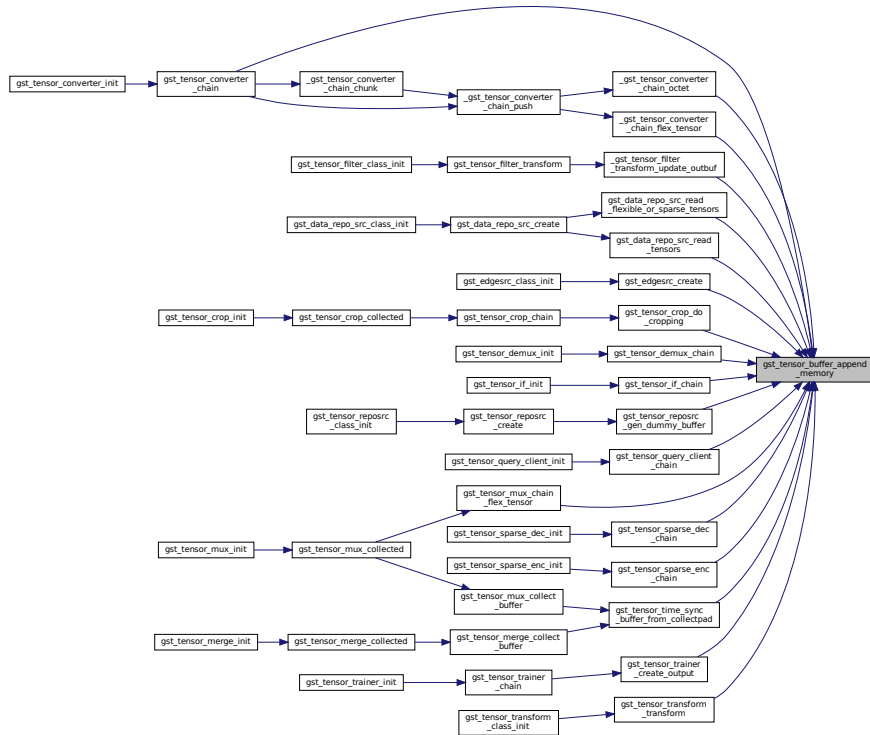


9.93.3.19 `gst_tensor_buffer_append_memory()`

```
gboolean gst_tensor_buffer_append_memory (
    GstBuffer * buffer,
    GstMemory * memory,
    const GstTensorInfo * info )
```

Append *memory* to given *buffer*.

Here is the caller graph for this function:



9.93.3.20 `gst_tensor_buffer_from_config()`

```
GstBuffer* gst_tensor_buffer_from_config (
    GstBuffer * in,
    GstTensorsConfig * config )
```

Configure `gst-buffer` with tensors information. `NNStreamer` handles single memory chunk as single tensor. If incoming buffer has invalid memories, separate it and generate new `gst-buffer` using tensors information. Note that this function always takes the ownership of input buffer.

Parameters

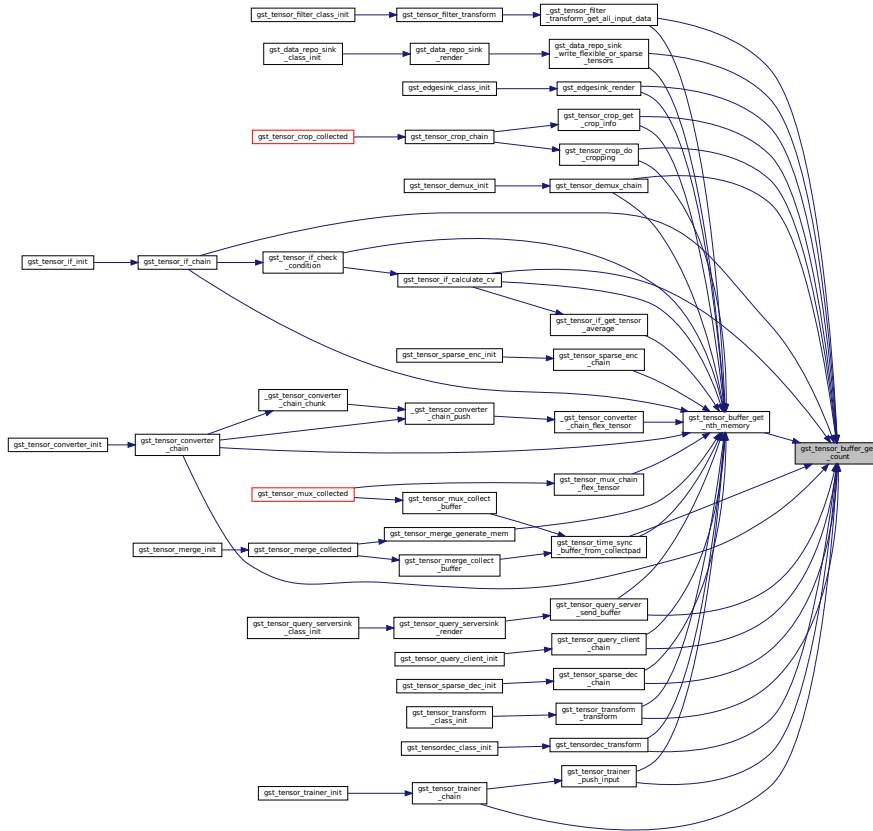
<i>in</i>	input buffer
<i>config</i>	tensors config structure

Returns

Newly allocated buffer. Null if failed. Caller should `unref` the buffer using `gst_buffer_unref()`.

Definition at line 535 of file `nnstreamer_plugin_api_impl.c`.

Here is the caller graph for this function:



9.93.3.22 `gst_tensor_buffer_get_nth_memory()`

```
GstMemory* gst_tensor_buffer_get_nth_memory (
    GstBuffer * buffer,
    const guint index )
```

Get the *nth* GstMemory from given *buffer*.

Parameters

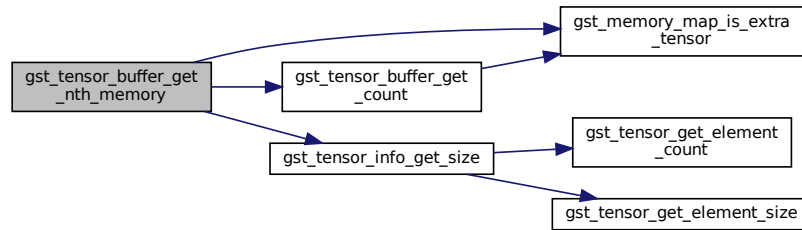
in	<i>buffer</i>	GstBuffer to be parsed.
in	<i>index</i>	Index of GstMemory to be returned.

Returns

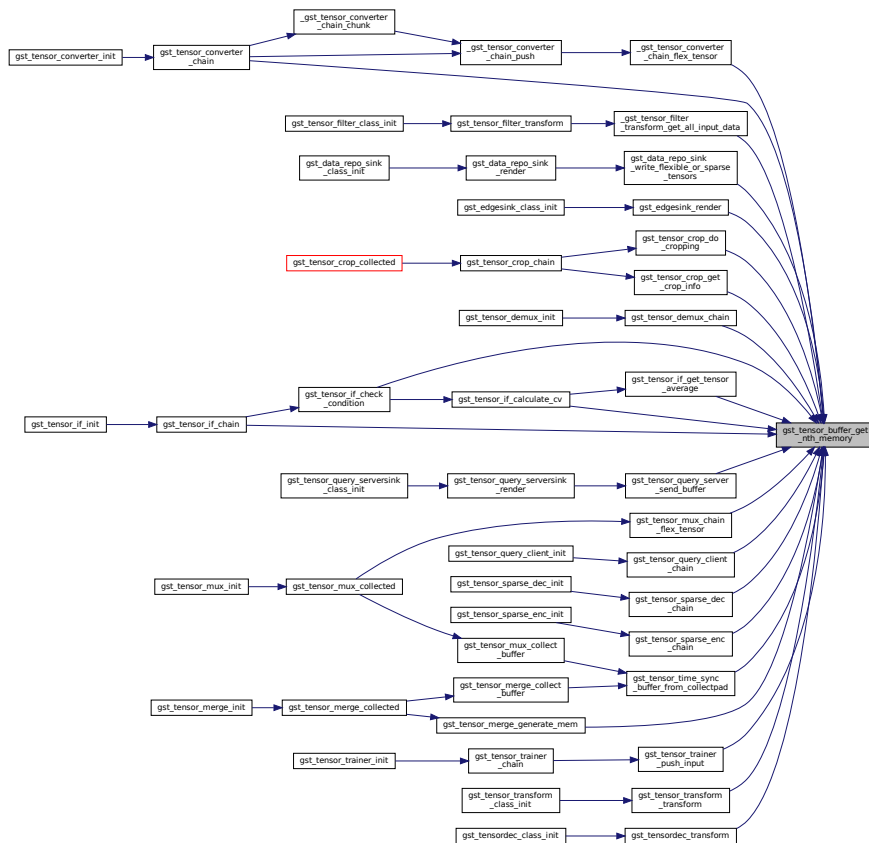
GstMemory if found, otherwise NULL (Caller should free returned memory using `gst_memory_unref()`).

Definition at line 1586 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.23 gst_tensor_caps_can_intersect()

```

gboolean gst_tensor_caps_can_intersect (
    GstCaps * caps1,
    GstCaps * caps2 )
  
```

Try intersecting @caps1 and @caps2 for tensor stream.

Parameters

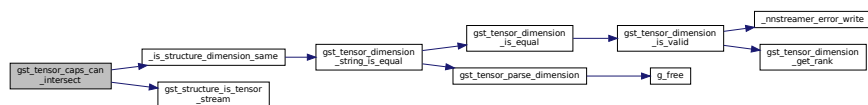
<i>caps1</i>	a GstCaps to intersect
<i>caps2</i>	a GstCaps to intersect

Returns

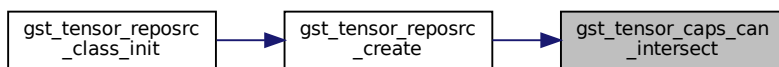
TRUE if intersection would be not empty.

Definition at line 1142 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.24 gst_tensor_caps_from_config()

```
GstCaps* gst_tensor_caps_from_config (
    const GstTensorsConfig * config )
```

Get tensor caps from tensors config.

Get tensor caps from tensors config (for other/tensor)

Parameters

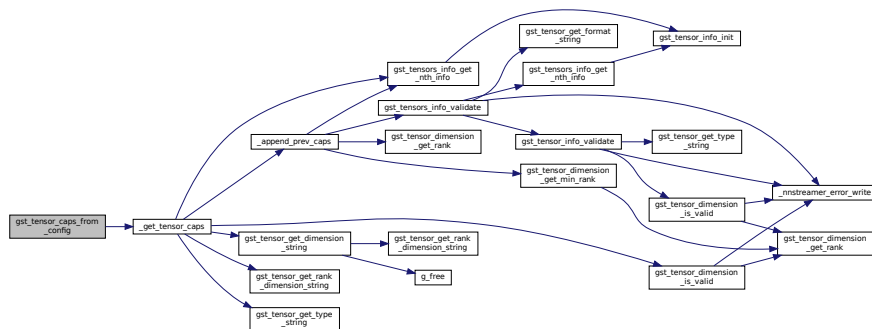
<i>config</i>	tensors config info
---------------	---------------------

Returns

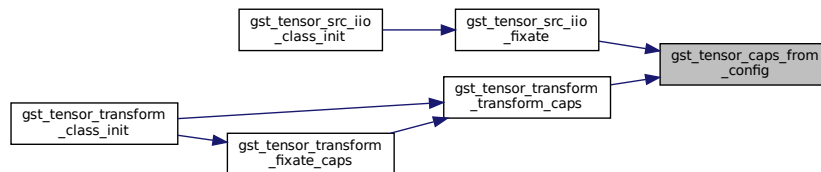
caps for given config

Definition at line 1395 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.25 gst_tensor_caps_update_dimension()

```
void gst_tensor_caps_update_dimension (
    GstCaps * caps,
    GstCaps * filter )
```

Update caps dimensions for negotiation.

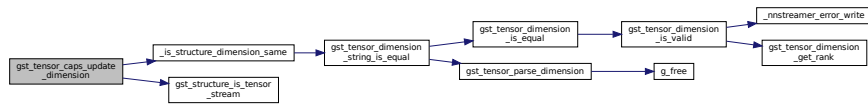
Update caps dimension for negotiation.

Parameters

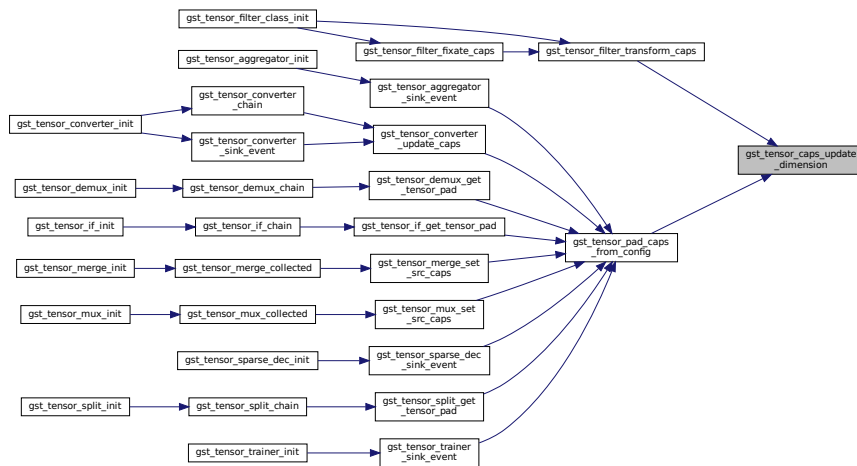
<i>caps</i>	caps to compare and update
<i>filter</i>	caps to compare

Definition at line 1093 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.26 gst_tensor_extra_info_init()

```

static void gst_tensor_extra_info_init (
    GstTensorExtraInfo * extra,
    gsize reserved_size ) [static]
  
```

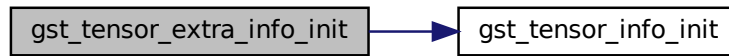
Initialize [GstTensorExtraInfo](#) structure with given *memory*.

Parameters

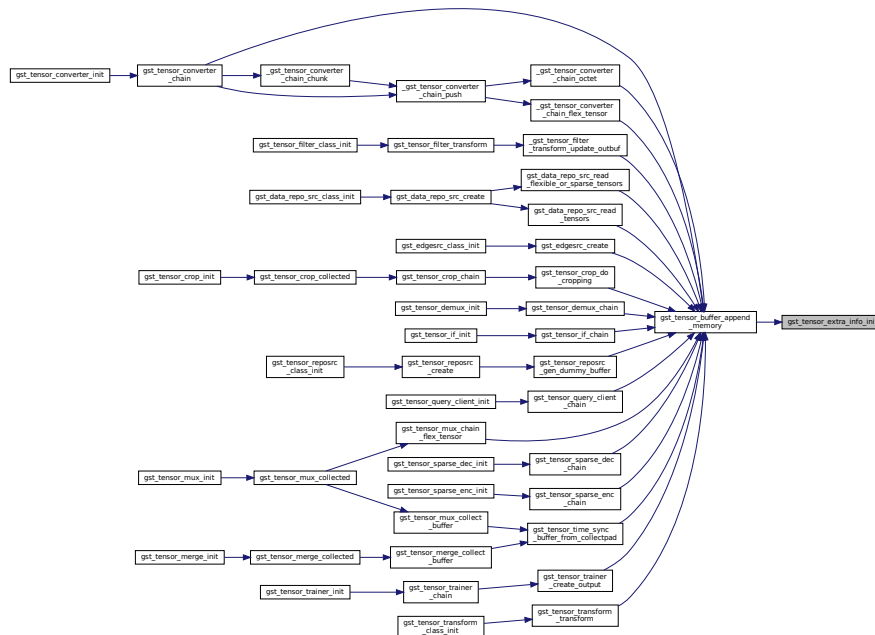
	<i>[in/out]</i>	extra GstTensorExtraInfo to be initialized.
in	<i>reserved_size</i>	The memory size of extra memory block.

Definition at line 78 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.27 gst_tensor_meta_info_append_header()

```

GstMemory* gst_tensor_meta_info_append_header (
    GstTensorMetaInfo * meta,
    GstMemory * mem )
  
```

Append header to memory.

Parameters

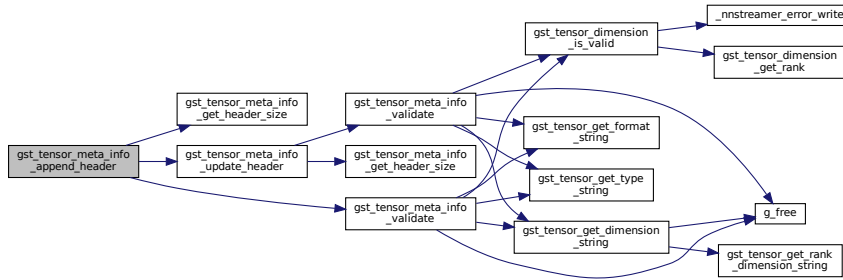
in	<i>meta</i>	tensor meta structure
in	<i>mem</i>	pointer to GstMemory

Returns

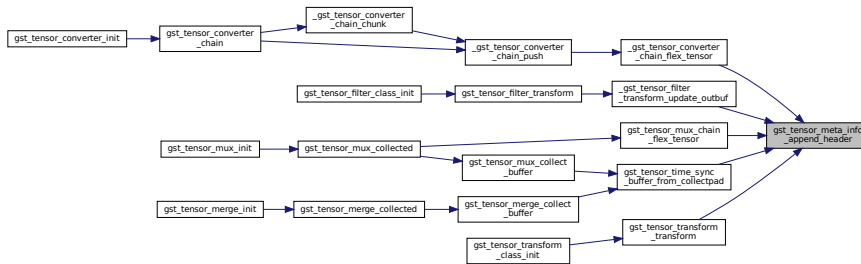
Newly allocated GstMemory (Caller should free returned memory using `gst_memory_unref()`)

Definition at line 1544 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.28 `gst_tensor_meta_info_parse_memory()`

```
gboolean gst_tensor_meta_info_parse_memory (
    GstTensorMetaInfo * meta,
    GstMemory * mem )
```

Parse memory and fill the tensor meta.

Parameters

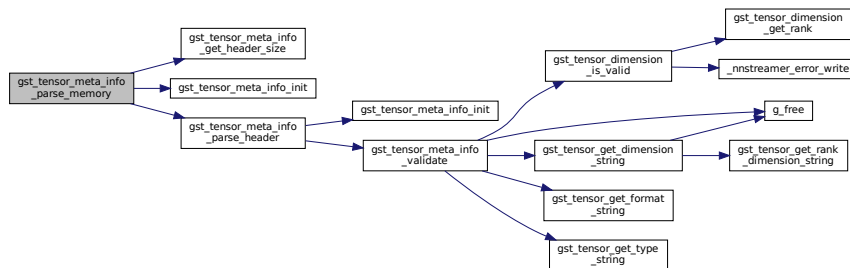
out	<i>meta</i>	tensor meta structure to be filled
in	<i>mem</i>	pointer to GstMemory to be parsed

Returns

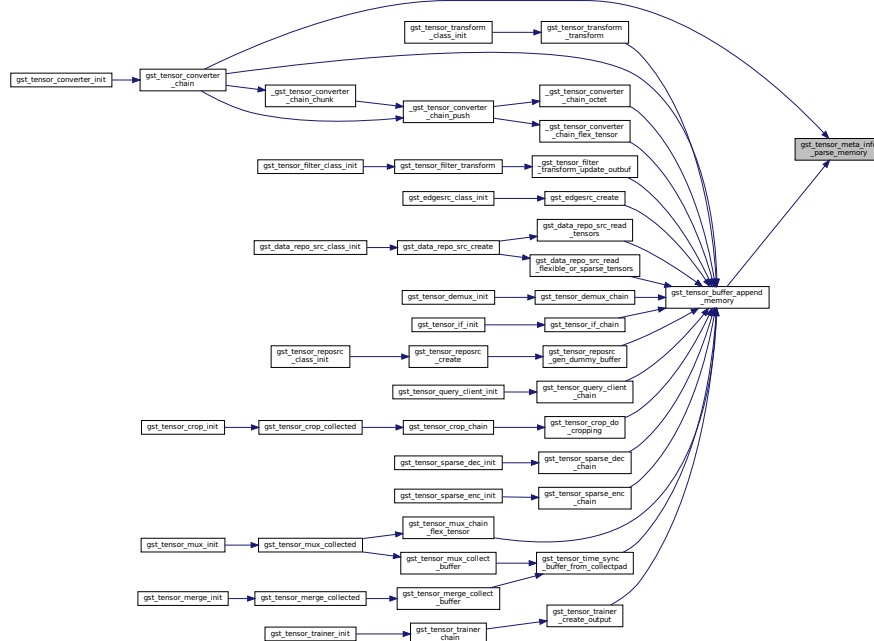
TRUE if successfully set the meta

Definition at line 1509 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:

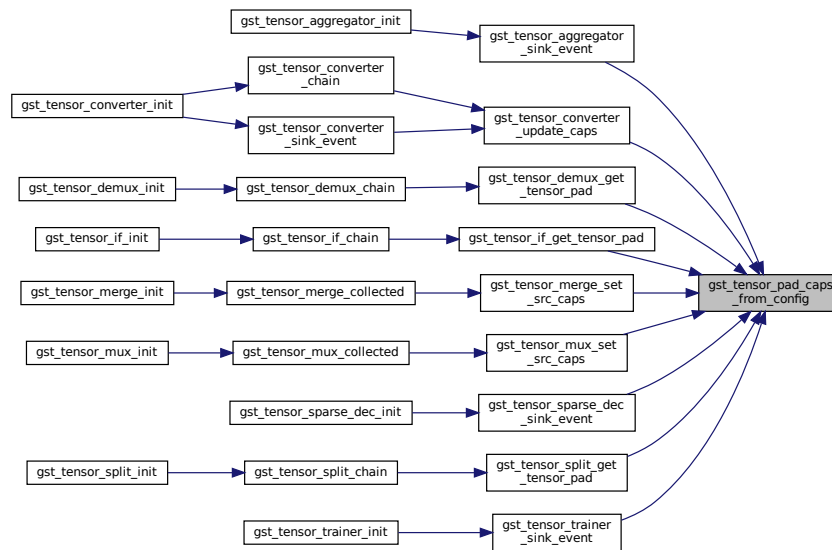


9.93.3.29 gst_tensor_pad_caps_from_config()

```
GstCaps* gst_tensor_pad_caps_from_config (
    GstPad * pad,
    const GstTensorsConfig * config )
```

Get pad caps from tensors config and caps of the peer connected to the pad.

Here is the caller graph for this function:



9.93.3.30 `gst_tensor_pad_get_format()`

```

tensor_format gst_tensor_pad_get_format (
    GstPad * pad )
  
```

Get tensor format of current pad caps.

Parameters

<i>pad</i>	GstPad to check current caps.
------------	-------------------------------

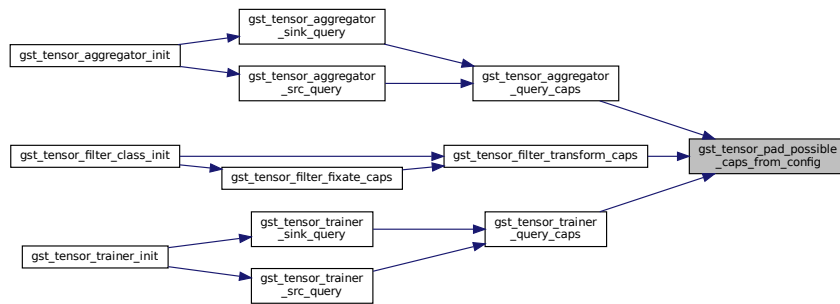
Returns

The `tensor_format` of current pad caps.

If pad does not have tensor caps return `_NNS_TENSOR_FORMAT_END`

Definition at line 1343 of file `nnstreamer_plugin_api_impl.c`.

Here is the caller graph for this function:



9.93.3.32 `gst_tensor_parse_config_file()`

```
void gst_tensor_parse_config_file (
    const gchar * config_path,
    const GObject * object )
```

Parses a configuration file and sets the corresponding properties on a GObject.

This function reads the contents of the configuration file located at the given path and sets the properties of the specified GObject based on the configuration data.

Parameters

<i>config_path</i>	The path to the configuration file.
<i>object</i>	The GObject on which to set the properties.

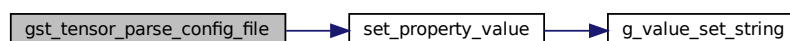
Note

The responsibility of managing the memory of the GObject passed as a parameter lies outside this function.

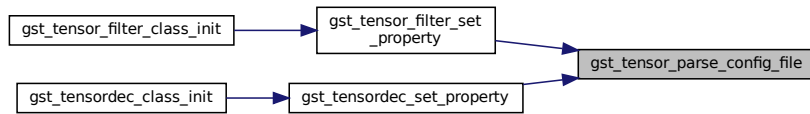
Iterate over each line

Definition at line 1902 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.33 `gst_tensor_time_sync_buffer_from_collectpad()`

```

gboolean gst_tensor_time_sync_buffer_from_collectpad (
    GstCollectPads * collect,
    tensor_time_sync_data * sync,
    GstClockTime current_time,
    GstBuffer * tensors_buf,
    GstTensorsConfig * configs,
    gboolean * is_eos )

```

A function call to make tensors from collected pads. It decide which buffer is going to be used according to sync option.

A function call to make tensors from collected pads It decide which buffer is going to be used according to sync option.

Returns

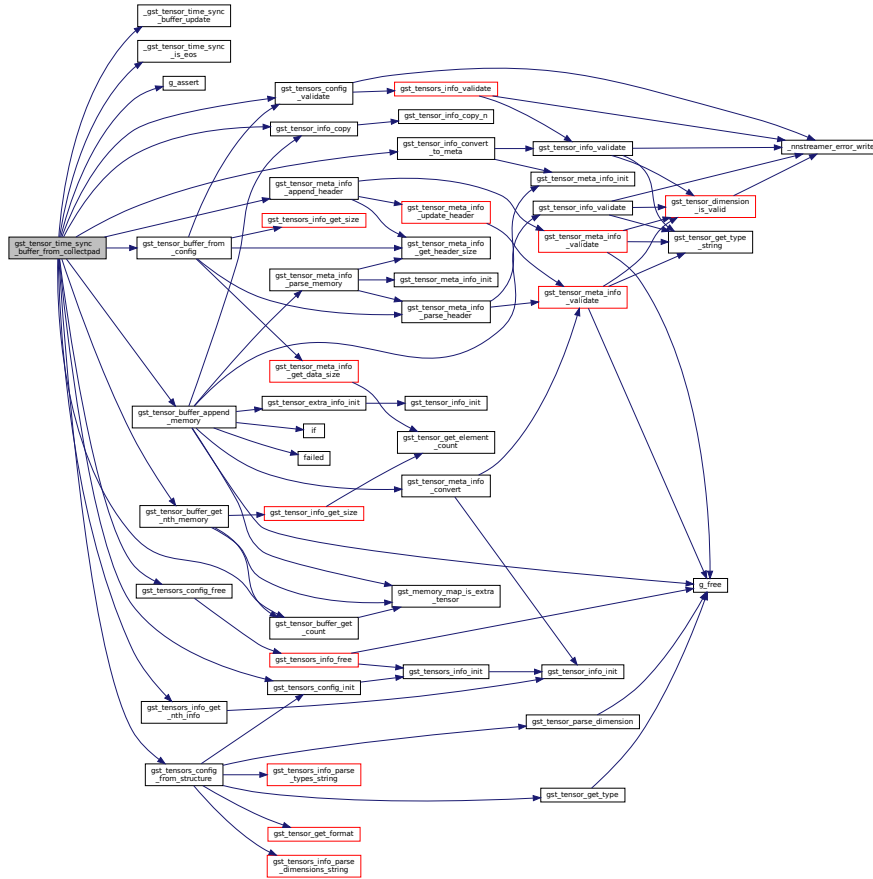
True to push buffer.

This would be an internal logic error. `in_configs` should be already confirmed valid at the negotiation phase and this function should be called in a running pipeline. If new sync mode is enabled (e.g., handle output when a pad gets new buffer), this may cause unexpected exception.

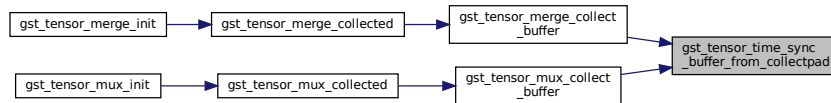
These are internal logic error. If given inputs are incorrect, the negotiation should have been failed before this stage.

Definition at line 332 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.34 gst_tensor_time_sync_flush()

```
void gst_tensor_time_sync_flush (
    GstCollectPads * collect )
```

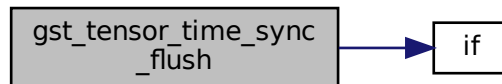
A function to be called while processing a flushing event. It should clear old buffer and reset pad data.

Parameters

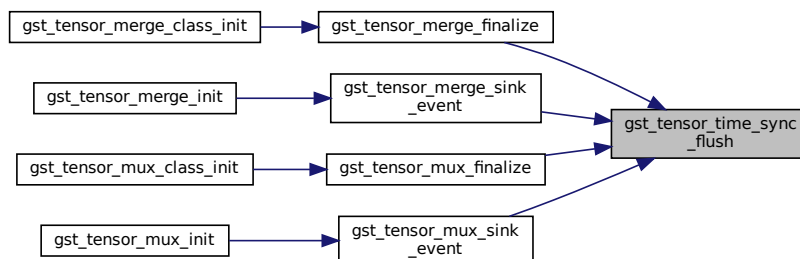
<i>collect</i>	Collect pad.
----------------	--------------

Definition at line 263 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.35 gst_tensor_time_sync_get_current_time()

```

gboolean gst_tensor_time_sync_get_current_time (
    GstCollectPads * collect,
    tensor_time_sync_data * sync,
    GstClockTime * current_time,
    GstBuffer * tensors_buf )
  
```

A function call to decide current timestamp among collected pads based on PTS. It will decide current timestamp according to sync option. GstMeta is also copied with same sync mode.

Returns

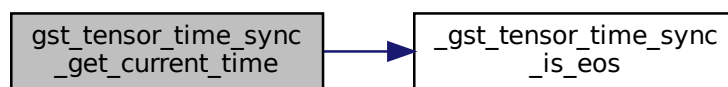
True / False. If EOS, it return TRUE.

Parameters

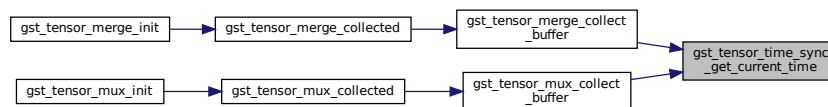
<i>collect</i>	Collect pad.
<i>sync</i>	Synchronization Option (NOSYNC, SLOWEST, BASEPAD, END)
<i>current_time</i>	Current time
<i>tensors_buf</i>	Generated GstBuffer for Collected Buffer

Definition at line 203 of file nntstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.36 gst_tensor_time_sync_get_mode()

```

tensor_time_sync_mode gst_tensor_time_sync_get_mode (
    const gchar * str )
  
```

Get the corresponding mode from the string value.

Parameters

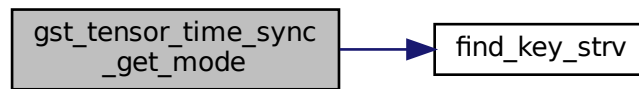
in	<i>str</i>	The string value for the mode.
----	------------	--------------------------------

Returns

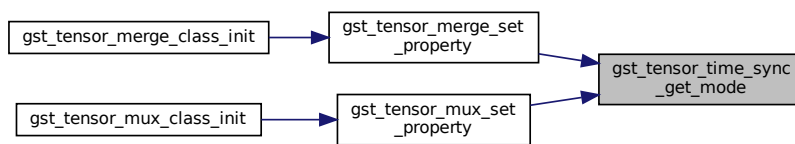
Corresponding mode for the string. SYNC_END for errors.

Definition at line 101 of file nntstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.37 gst_tensor_time_sync_get_mode_string()

```
const gchar* gst_tensor_time_sync_get_mode_string (
    tensor_time_sync_mode mode )
```

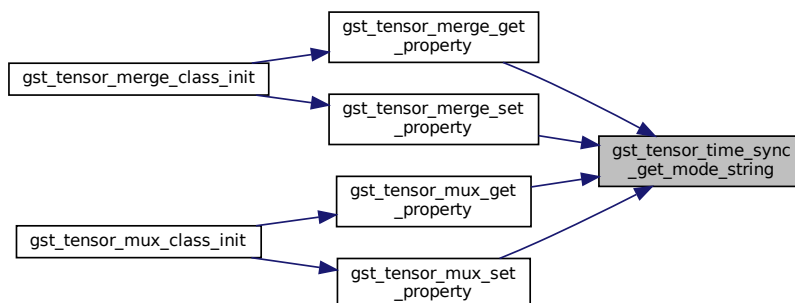
Get the time-sync mode string.

Returns

Corresponding mode string.

Definition at line 115 of file `nnstreamer_plugin_api_impl.c`.

Here is the caller graph for this function:



9.93.3.38 `gst_tensor_time_sync_set_option_data()`

```
gboolean gst_tensor_time_sync_set_option_data (
    tensor_time_sync_data * sync )
```

Setup time sync option.

Parameters

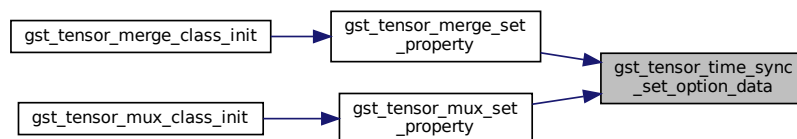
<i>[in/out]</i>	filter "this" pointer. Sync mode & option MUST BE set already.
-----------------	--

Returns

True if successfully set the option.

Definition at line 126 of file `nnstreamer_plugin_api_impl.c`.

Here is the caller graph for this function:



9.93.3.39 `gst_tensors_caps_from_config()`

```
GstCaps* gst_tensors_caps_from_config (
    const GstTensorsConfig * config )
```

Get caps from tensors config (for other/tensors)

Parameters

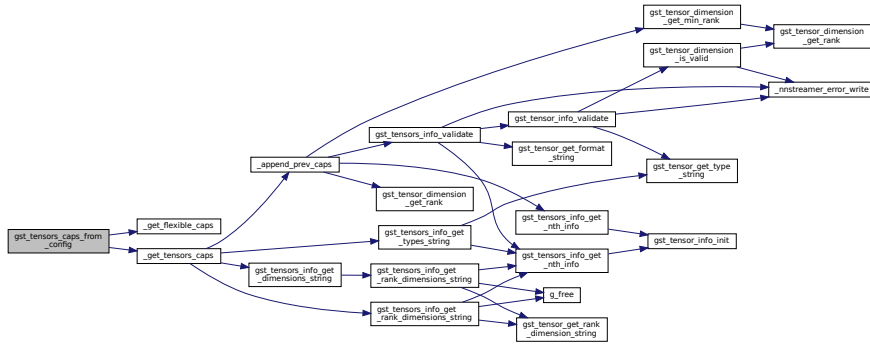
<i>config</i>	tensors config info
---------------	---------------------

Returns

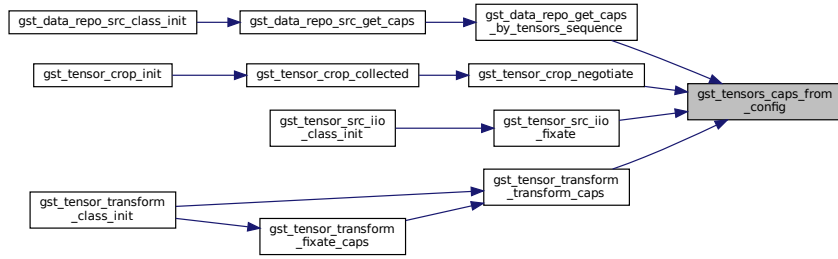
caps for given config

Definition at line 1372 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.40 gst_tensors_config_from_peer()

```
gboolean gst_tensors_config_from_peer (
    GstPad * pad,
    GstTensorsConfig * config,
    gboolean * is_fixed )
```

Parse caps from peer pad and set tensors config.

Parameters

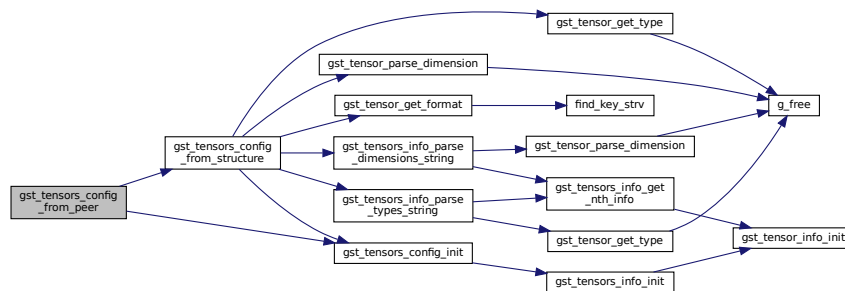
<i>pad</i>	GstPad to get the capabilities
<i>config</i>	tensors config structure to be filled
<i>is_fixed</i>	flag to be updated when peer caps is fixed (not mandatory, do nothing when the param is null)

Returns

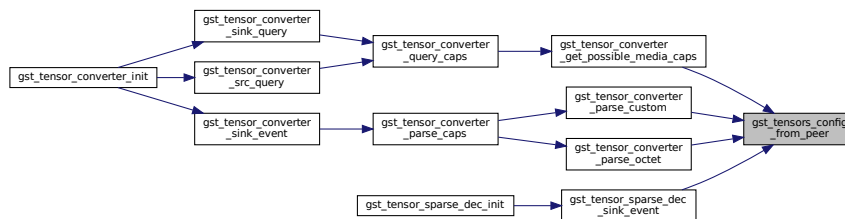
TRUE if successfully configured from peer

Definition at line 1041 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.3.41 gst_tensors_config_from_structure()

```

gboolean gst_tensors_config_from_structure (
    GstTensorsConfig * config,
    const GstStructure * structure )

```

Parse structure and set tensors config (for other/tensors)

Parameters

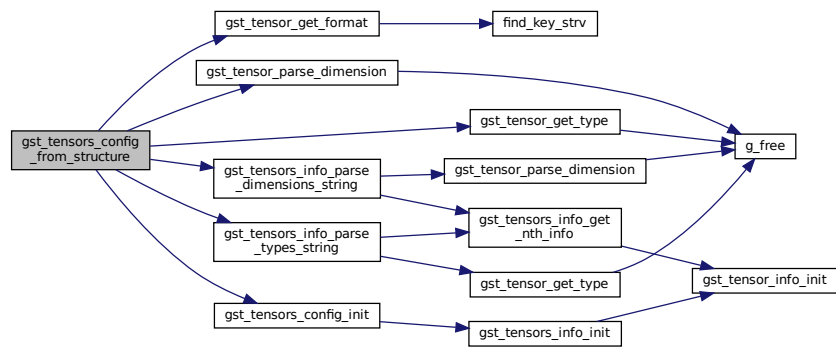
<i>config</i>	tensors config structure to be filled
<i>structure</i>	structure to be interpreted

Returns

TRUE if no error

Definition at line 1413 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:




```
const GParamSpec * param_spec,
const gchar * property_value ) [static]
```

Sets the value of a property based on the specified property value and GParamSpec.

Parameters

<i>prop_value</i>	A pointer to the GValue where the property value will be set.
<i>param_spec</i>	A pointer to the GParamSpec that describes the property.
<i>property_value</i>	A string representing the value to be set for the property.

Note

This API is intended to be used by `gst_tensor_parse_config_file ()`

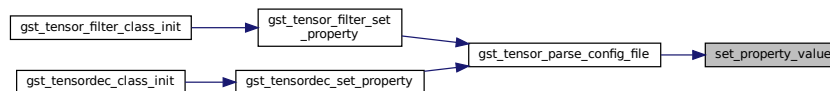
default is string

Definition at line 1862 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.93.4 Variable Documentation

9.93.4.1 gst_tensor_time_sync_mode_string

```
const gchar* gst_tensor_time_sync_mode_string[] [static]
```

Initial value:

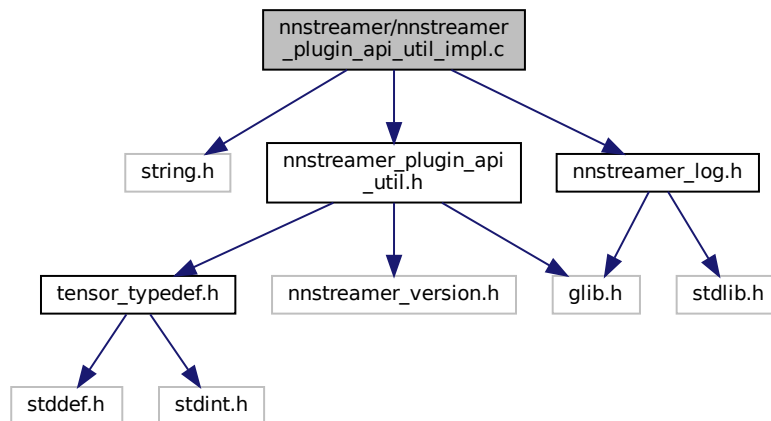
```
= {
  [SYNC_NOSYNC] = "nosync",
  [SYNC_SLOWEST] = "slowest",
  [SYNC_BASEPAD] = "basepad",
  [SYNC_REFRESH] = "refresh",
  [SYNC_END] = NULL
}
```

Definition at line 20 of file `nnstreamer_plugin_api_impl.c`.

9.94 nnstreamer/nnstreamer_plugin_api_util_impl.c File Reference

Tensor common util functions for NNStreamer. (No gst dependency)

```
#include <string.h>
#include "nnstreamer_plugin_api_util.h"
#include "nnstreamer_log.h"
Include dependency graph for nnstreamer_plugin_api_util_impl.c:
```



Macros

- #define `GST_TENSOR_META_MAGIC` (0xfeedccd)
- Magic number of tensor meta.*
- #define `GST_TENSOR_META_MAGIC_VALID(m)` ((m) == `GST_TENSOR_META_MAGIC`)
- Macro to check the tensor meta.*
- #define `GST_TENSOR_META_VERSION_VALID(v)` (((v) & 0xDE000000) == 0xDE000000)
- Macro to check the meta version.*
- #define `GST_TENSOR_META_MAKE_VERSION(major, minor)` ((major) << 12 | (minor) | 0xDE000000)
- Macro to get the version of tensor meta.*
- #define `GST_TENSOR_META_VERSION` `GST_TENSOR_META_MAKE_VERSION(1,0)`
- The version of tensor meta.*
- #define `GST_TENSOR_META_IS_V1(v)` (`GST_TENSOR_META_VERSION_VALID(v)` && (((v) & 0x00FF ← F000) & `GST_TENSOR_META_MAKE_VERSION(1,0)`))
- Macro to check the version of tensor meta.*
- #define `GST_TENSOR_META_IS_VALID(m)` ((m) && `GST_TENSOR_META_MAGIC_VALID` ((m)->magic) && `GST_TENSOR_META_VERSION_VALID` ((m)->version))
- Macro to check the meta is valid.*

Functions

- static gint [_gcd](#) (gint a, gint b)
Internal function, copied from `gst_util_greatest_common_divisor()` to remove dependency of `gststreamer`.
- static gint [_compare_rate](#) (gint a_n, gint a_d, gint b_n, gint b_d)
Internal function, copied from `gst_util_fraction_compare()` to remove dependency of `gststreamer`.
- void [gst_tensor_info_init](#) ([GstTensorInfo](#) *info)
Initialize the tensor info structure.
- void [gst_tensor_info_free](#) ([GstTensorInfo](#) *info)
Free allocated data in tensor info structure.
- gsize [gst_tensor_info_get_size](#) (const [GstTensorInfo](#) *info)
Get data size of single tensor.
- gboolean [gst_tensor_info_validate](#) (const [GstTensorInfo](#) *info)
Check the tensor info is valid.
- gboolean [gst_tensor_info_is_equal](#) (const [GstTensorInfo](#) *i1, const [GstTensorInfo](#) *i2)
Compare tensor info.
- void [gst_tensor_info_copy_n](#) ([GstTensorInfo](#) *dest, const [GstTensorInfo](#) *src, const guint n)
Copy tensor info up to n elements.
- void [gst_tensor_info_copy](#) ([GstTensorInfo](#) *dest, const [GstTensorInfo](#) *src)
Copy tensor info.
- gboolean [gst_tensor_info_convert_to_meta](#) ([GstTensorInfo](#) *info, [GstTensorMetaInfo](#) *meta)
Convert `GstTensorInfo` structure to `GstTensorMetaInfo`.
- guint [gst_tensor_info_get_rank](#) (const [GstTensorInfo](#) *info)
Get tensor rank.
- [GstTensorInfo](#) * [gst_tensors_info_get_nth_info](#) ([GstTensorsInfo](#) *info, guint index)
Get the pointer of nth tensor information.
- void [gst_tensors_info_init](#) ([GstTensorsInfo](#) *info)
Initialize the tensors info structure.
- void [gst_tensors_info_free](#) ([GstTensorsInfo](#) *info)
Free allocated data in tensors info structure.
- gsize [gst_tensors_info_get_size](#) (const [GstTensorsInfo](#) *info, gint index)
Get data size of single tensor.
- gboolean [gst_tensors_info_validate](#) (const [GstTensorsInfo](#) *info)
Check the tensors info is valid.
- gboolean [gst_tensors_info_is_equal](#) (const [GstTensorsInfo](#) *i1, const [GstTensorsInfo](#) *i2)
Compare tensors info.
- void [gst_tensors_info_copy](#) ([GstTensorsInfo](#) *dest, const [GstTensorsInfo](#) *src)
Copy tensor info.
- guint [gst_tensors_info_parse_dimensions_string](#) ([GstTensorsInfo](#) *info, const gchar *dim_string)
Parse the string of dimensions.
- guint [gst_tensors_info_parse_types_string](#) ([GstTensorsInfo](#) *info, const gchar *type_string)
Parse the string of types.
- guint [gst_tensors_info_parse_names_string](#) ([GstTensorsInfo](#) *info, const gchar *name_string)
Parse the string of names.
- gchar * [gst_tensors_info_get_dimensions_string](#) (const [GstTensorsInfo](#) *info)
Get the string of dimensions in tensors info.
- gchar * [gst_tensors_info_get_rank_dimensions_string](#) (const [GstTensorsInfo](#) *info, const unsigned int rank)
Get the string of dimensions in tensors info and rank count.
- gchar * [gst_tensors_info_get_types_string](#) (const [GstTensorsInfo](#) *info)
Get the string of types in tensors info.
- gchar * [gst_tensors_info_get_names_string](#) (const [GstTensorsInfo](#) *info)

- Get the string of tensor names in tensors info.*

 - gchar * [gst_tensors_info_to_string](#) (const [GstTensorsInfo](#) *info)

GstTensorsInfo represented as a string. Caller should free it.
- void [gst_tensors_config_init](#) ([GstTensorsConfig](#) *config)

Initialize the tensors config info structure (for other/tensors)
- void [gst_tensors_config_free](#) ([GstTensorsConfig](#) *config)

Free allocated data in tensors config structure.
- gboolean [gst_tensors_config_validate](#) (const [GstTensorsConfig](#) *config)

Check the tensors are all configured.
- gboolean [gst_tensors_config_is_equal](#) (const [GstTensorsConfig](#) *c1, const [GstTensorsConfig](#) *c2)

Compare tensor config info.
- void [gst_tensors_config_copy](#) ([GstTensorsConfig](#) *dest, const [GstTensorsConfig](#) *src)

Copy tensors config.
- gchar * [gst_tensors_config_to_string](#) (const [GstTensorsConfig](#) *config)

Tensor config represented as a string. Caller should free it.
- gboolean [gst_tensor_dimension_is_valid](#) (const [tensor_dim](#) dim)

Check the tensor dimension is valid.
- gboolean [gst_tensor_dimension_is_equal](#) (const [tensor_dim](#) dim1, const [tensor_dim](#) dim2)

Compare the tensor dimension.
- guint [gst_tensor_dimension_get_rank](#) (const [tensor_dim](#) dim)

Get the rank of tensor dimension.
- guint [gst_tensor_dimension_get_min_rank](#) (const [tensor_dim](#) dim)

Get the minimum rank of tensor dimension.
- guint [gst_tensor_parse_dimension](#) (const gchar *dimstr, [tensor_dim](#) dim)

Parse tensor dimension parameter string.
- gchar * [gst_tensor_get_dimension_string](#) (const [tensor_dim](#) dim)

Get dimension string from given tensor dimension.
- gchar * [gst_tensor_get_rank_dimension_string](#) (const [tensor_dim](#) dim, const unsigned int rank)

Get dimension string from given tensor dimension and rank count.
- gboolean [gst_tensor_dimension_string_is_equal](#) (const gchar *dimstr1, const gchar *dimstr2)

Compare dimension strings.
- gulong [gst_tensor_get_element_count](#) (const [tensor_dim](#) dim)

Count the number of elements of a tensor.
- gsize [gst_tensor_get_element_size](#) ([tensor_type](#) type)

Get element size of tensor type (byte per element)
- [tensor_type](#) [gst_tensor_get_type](#) (const gchar *typestr)

Get tensor type from string input.
- const gchar * [gst_tensor_get_type_string](#) ([tensor_type](#) type)

Get type string of tensor type.
- [tensor_format](#) [gst_tensor_get_format](#) (const gchar *format_str)

Get tensor format from string input.
- const gchar * [gst_tensor_get_format_string](#) ([tensor_format](#) format)

Get tensor format string.
- void [gst_tensor_meta_info_init](#) ([GstTensorMetaInfo](#) *meta)

Initialize the tensor meta info structure.
- void [gst_tensor_meta_info_get_version](#) ([GstTensorMetaInfo](#) *meta, guint *major, guint *minor)

Get the version of tensor meta.
- gboolean [gst_tensor_meta_info_validate](#) ([GstTensorMetaInfo](#) *meta)

Check the meta info is valid.
- gsize [gst_tensor_meta_info_get_header_size](#) ([GstTensorMetaInfo](#) *meta)

Get the header size to handle a tensor meta.

- gsize [gst_tensor_meta_info_get_data_size](#) ([GstTensorMetaInfo](#) *meta)
Get the data size calculated from tensor meta.
- gboolean [gst_tensor_meta_info_update_header](#) ([GstTensorMetaInfo](#) *meta, gpointer header)
Update header from tensor meta.
- gboolean [gst_tensor_meta_info_parse_header](#) ([GstTensorMetaInfo](#) *meta, gpointer header)
Parse header and fill the tensor meta.
- gboolean [gst_tensor_meta_info_convert](#) ([GstTensorMetaInfo](#) *meta, [GstTensorInfo](#) *info)
Convert [GstTensorMetaInfo](#) structure to [GstTensorInfo](#).
- gint [find_key_strv](#) (const gchar **strv, const gchar *key)
Find the index value of the given key string array.
- gchar * [nnstreamer_version_string](#) (void)
Get the version of NNStreamer (string).
- void [nnstreamer_version_fetch](#) (guint *major, guint *minor, guint *micro)
Get the version of NNStreamer (int, divided).

Variables

- static const gchar * [tensor_element_typename](#) []
String representations for each tensor element type.
- static const guint [tensor_element_size](#) []
Byte-per-element of each tensor element type.
- static const gchar * [tensor_format_name](#) []
String representations for tensor format.

9.94.1 Detailed Description

Tensor common util functions for NNStreamer. (No gst dependency)

Copyright (c) 2022 Samsung Electronics Co., Ltd. All Rights Reserved.

Date

28 Jan 2022

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.94.2 Macro Definition Documentation

9.94.2.1 GST_TENSOR_META_IS_V1

```
#define GST_TENSOR_META_IS_V1(  
    v ) (GST_TENSOR_META_VERSION_VALID(v) && ((v) & 0x00FFF000) & GST_TENSOR_META_MAKE_VERSION(1,0))
```

Macro to check the version of tensor meta.

Definition at line 1360 of file nnstreamer_plugin_api_util_impl.c.

9.94.2.2 GST_TENSOR_META_IS_VALID

```
#define GST_TENSOR_META_IS_VALID(  
    m ) ((m) && GST_TENSOR_META_MAGIC_VALID ((m)->magic) && GST_TENSOR_META_VERSION_VALID  
    ((m)->version))
```

Macro to check the meta is valid.

Definition at line 1365 of file nnstreamer_plugin_api_util_impl.c.

9.94.2.3 GST_TENSOR_META_MAGIC

```
#define GST_TENSOR_META_MAGIC (0xfeedcccd)
```

Magic number of tensor meta.

Definition at line 1335 of file nnstreamer_plugin_api_util_impl.c.

9.94.2.4 GST_TENSOR_META_MAGIC_VALID

```
#define GST_TENSOR_META_MAGIC_VALID(  
    m ) ((m) == GST_TENSOR_META_MAGIC)
```

Macro to check the tensor meta.

Definition at line 1340 of file nnstreamer_plugin_api_util_impl.c.

9.94.2.5 GST_TENSOR_META_MAKE_VERSION

```
#define GST_TENSOR_META_MAKE_VERSION(  
    major,  
    minor ) ((major) << 12 | (minor) | 0xDE000000)
```

Macro to get the version of tensor meta.

Definition at line 1350 of file nnstreamer_plugin_api_util_impl.c.

9.94.2.6 GST_TENSOR_META_VERSION

```
#define GST_TENSOR_META_VERSION GST_TENSOR_META_MAKE_VERSION(1,0)
```

The version of tensor meta.

Definition at line 1355 of file nnsreamer_plugin_api_util_impl.c.

9.94.2.7 GST_TENSOR_META_VERSION_VALID

```
#define GST_TENSOR_META_VERSION_VALID(  
    v ) (((v) & 0xDE000000) == 0xDE000000)
```

Macro to check the meta version.

Definition at line 1345 of file nnsreamer_plugin_api_util_impl.c.

9.94.3 Function Documentation

9.94.3.1 _compare_rate()

```
static gint _compare_rate (  
    gint a_n,  
    gint a_d,  
    gint b_n,  
    gint b_d ) [static]
```

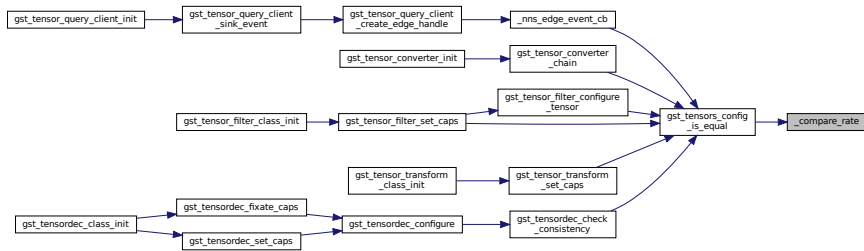
Internal function, copied from gst_util_fraction_compare() to remove dependency of gstreamer.

Definition at line 83 of file nnsreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.2 _gcd()

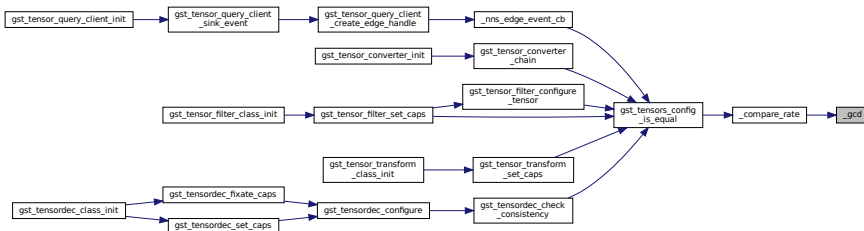
```

static gint _gcd (
    gint a,
    gint b ) [static]
  
```

Internal function, copied from `gst_util_greatest_common_divisor()` to remove dependency of `gststreamer`.

Definition at line 67 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the caller graph for this function:



9.94.3.3 find_key_strv()

```

gint find_key_strv (
    const gchar ** strv,
    const gchar * key )
  
```

Find the index value of the given key string array.

Returns

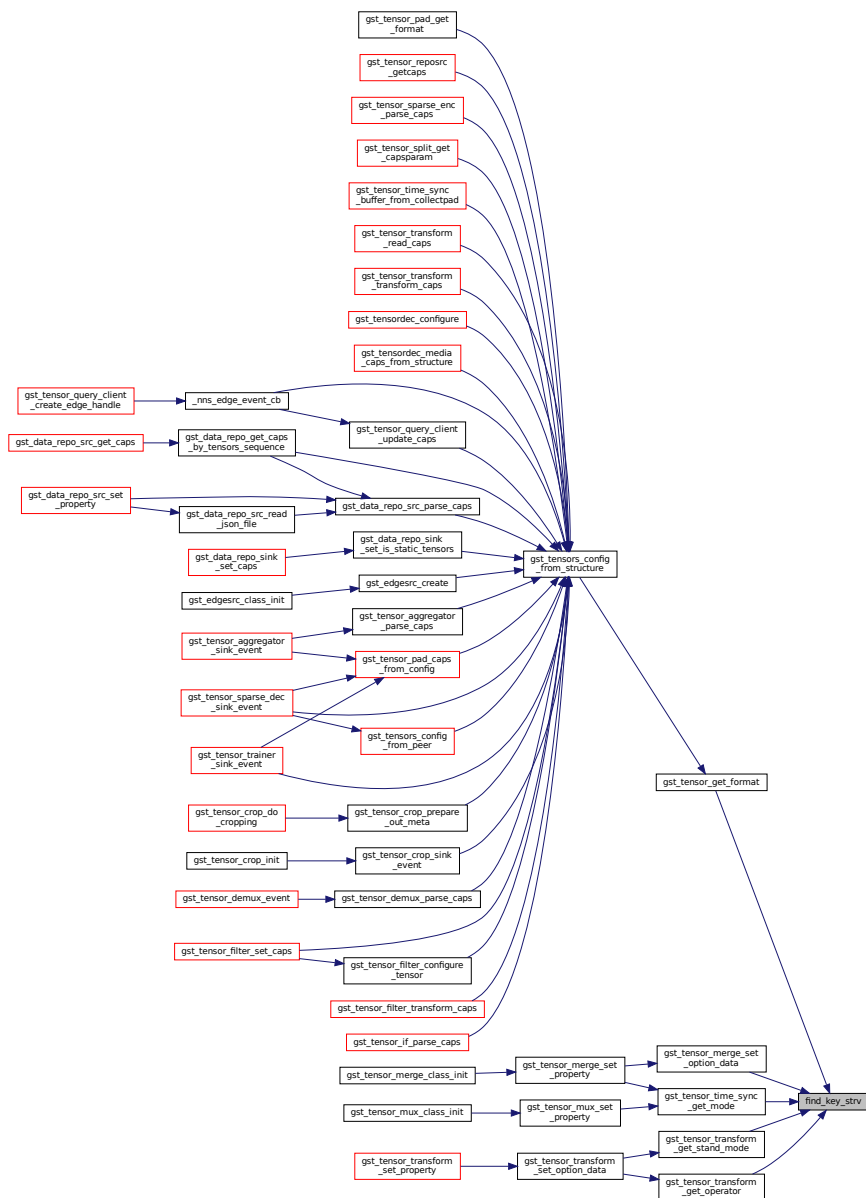
Corresponding index. Returns -1 if not found.

Parameters

<i>strv</i>	Null terminated array of gchar *
<i>key</i>	The key string value

Definition at line 1586 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:

9.94.3.4 `gst_tensor_dimension_get_min_rank()`

```

guint gst_tensor_dimension_get_min_rank (
    const tensor\_dim dim )
  
```

Get the minimum rank of tensor dimension.

The C-arrays with dim 4:4:4 and 4:4:4:1 have same data. In this case, this function returns min rank 3.

Parameters

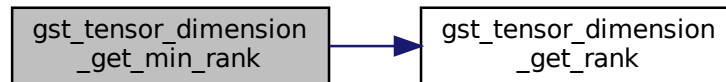
<i>dim</i>	tensor dimension.
------------	-------------------

Returns

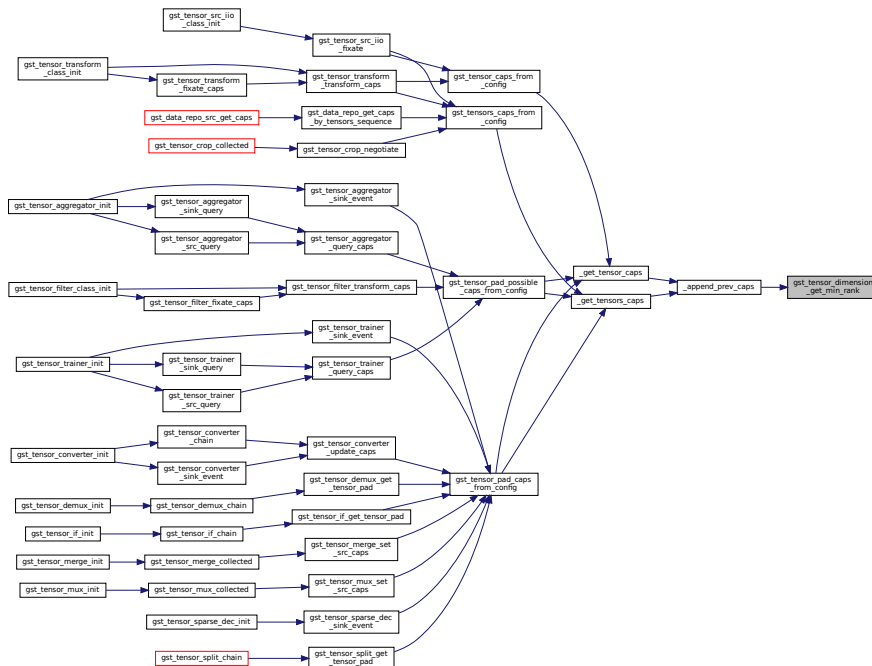
tensor rank (Minimum rank is 1 if given dimension is valid)

Definition at line 1017 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.5 gst_tensor_dimension_get_rank()

```
guint gst_tensor_dimension_get_rank (
    const tensor_dim dim )
```

Get the rank of tensor dimension.

Parameters

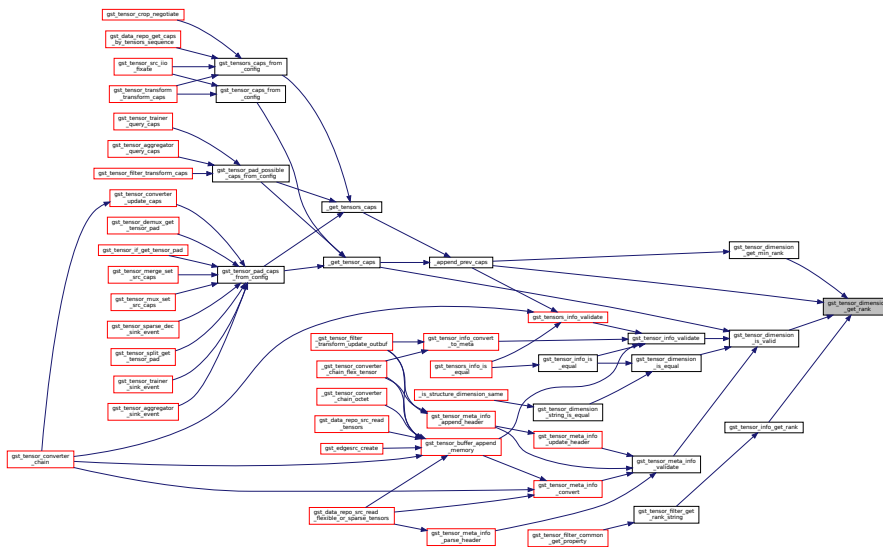
<i>dim</i>	tensor dimension.
------------	-------------------

Returns

tensor rank (Minimum rank is 1 if given dimension is valid)

Definition at line 998 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



9.94.3.6 gst_tensor_dimension_is_equal()

```
gboolean gst_tensor_dimension_is_equal (
    const tensor_dim dim1,
    const tensor_dim dim2 )
```

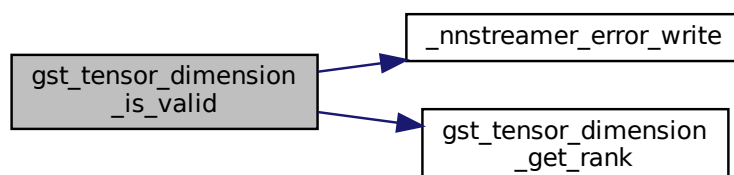
Compare the tensor dimension.

Returns

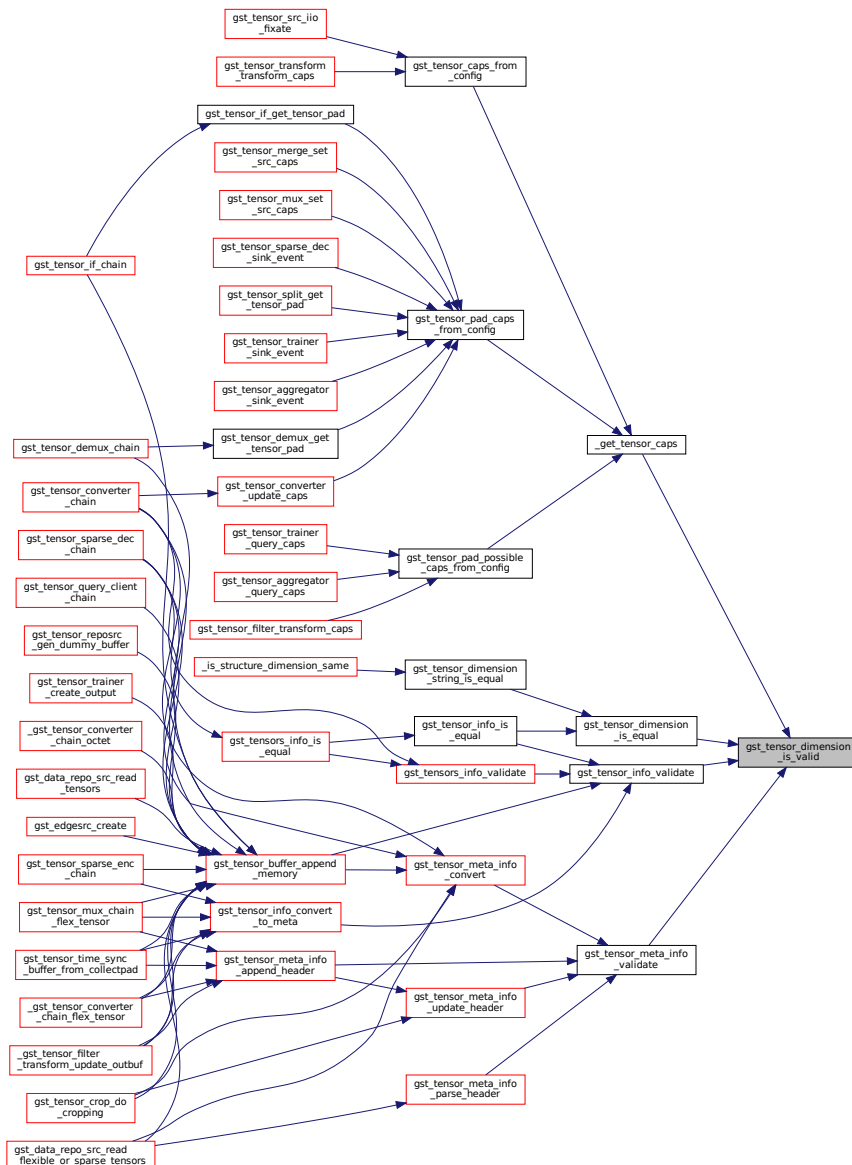
TRUE if dimension is valid

Definition at line 940 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.8 `gst_tensor_dimension_string_is_equal()`

```
gboolean gst_tensor_dimension_string_is_equal (
    const gchar * dimstr1,
    const gchar * dimstr2 )
```

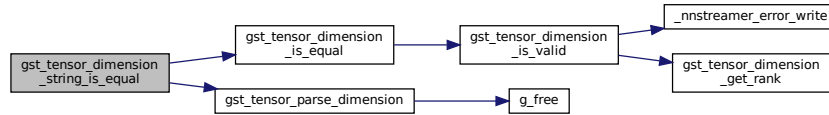
Compare dimension strings.

Returns

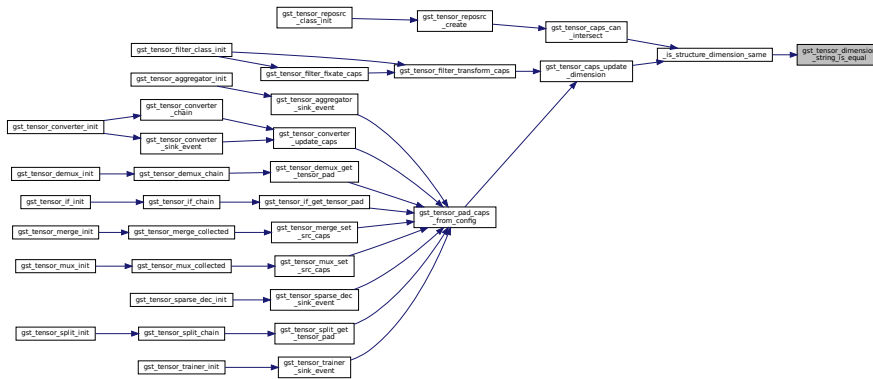
TRUE if equal, FALSE if given dimension strings are invalid or not equal.

Definition at line 1140 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.9 gst_tensor_get_dimension_string()

```
gchar* gst_tensor_get_dimension_string (
    const tensor_dim dim )
```

Get dimension string from given tensor dimension.

Parameters

<i>dim</i>	tensor dimension
------------	------------------

Returns

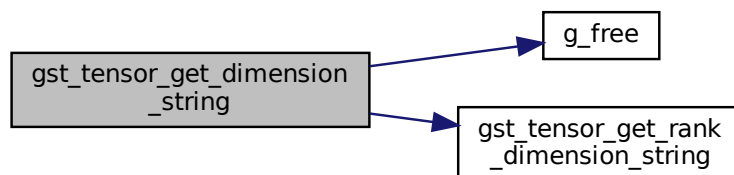
Formatted string of given dimension (d1:d2:d3:...:d15:d16).

Note

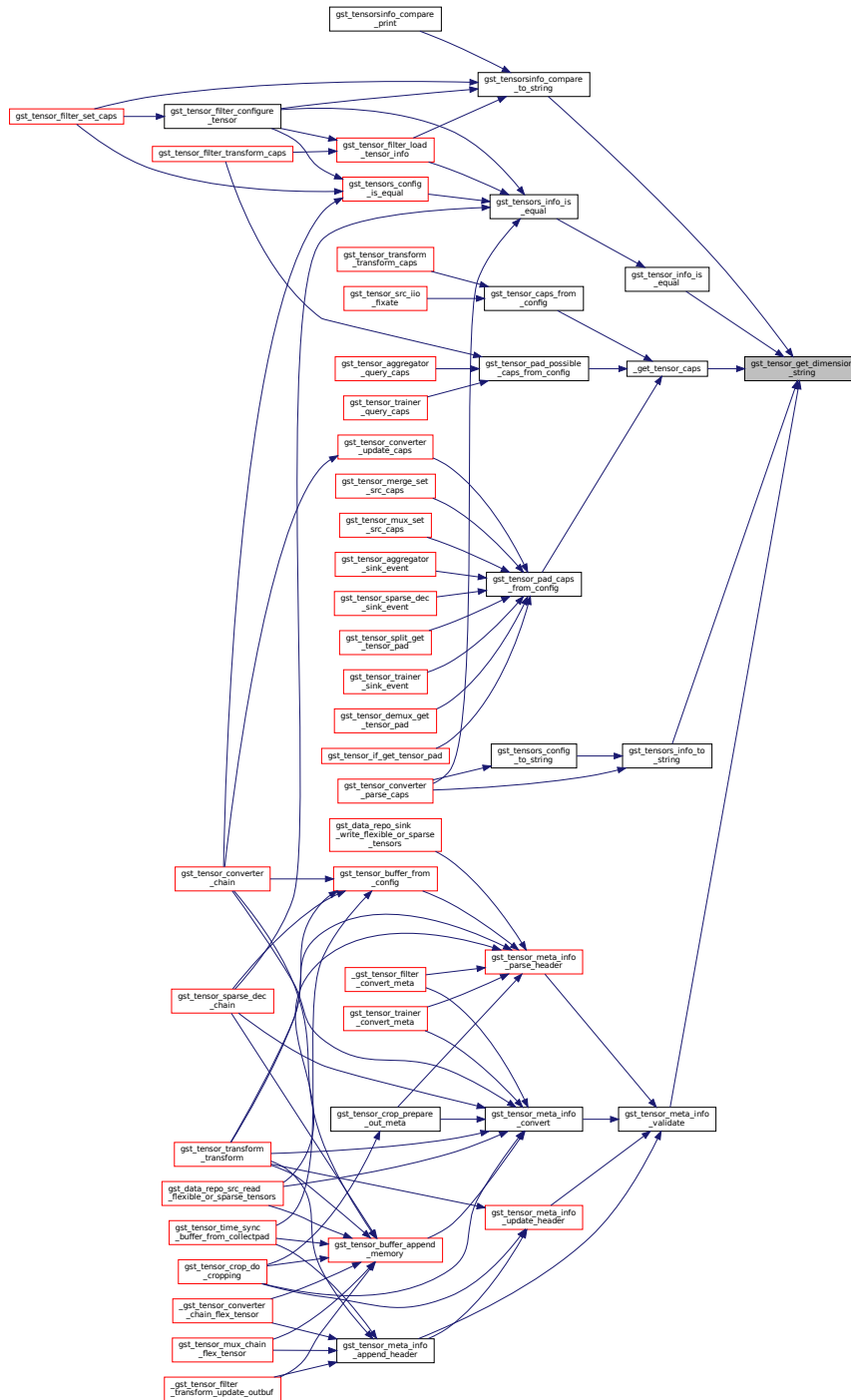
The returned value should be freed with [g_free\(\)](#)

Definition at line 1083 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.10 gst_tensor_get_element_count()

```

gulong gst_tensor_get_element_count (
    const tensor_dim dim )
    
```

Count the number of elements of a tensor.

Returns

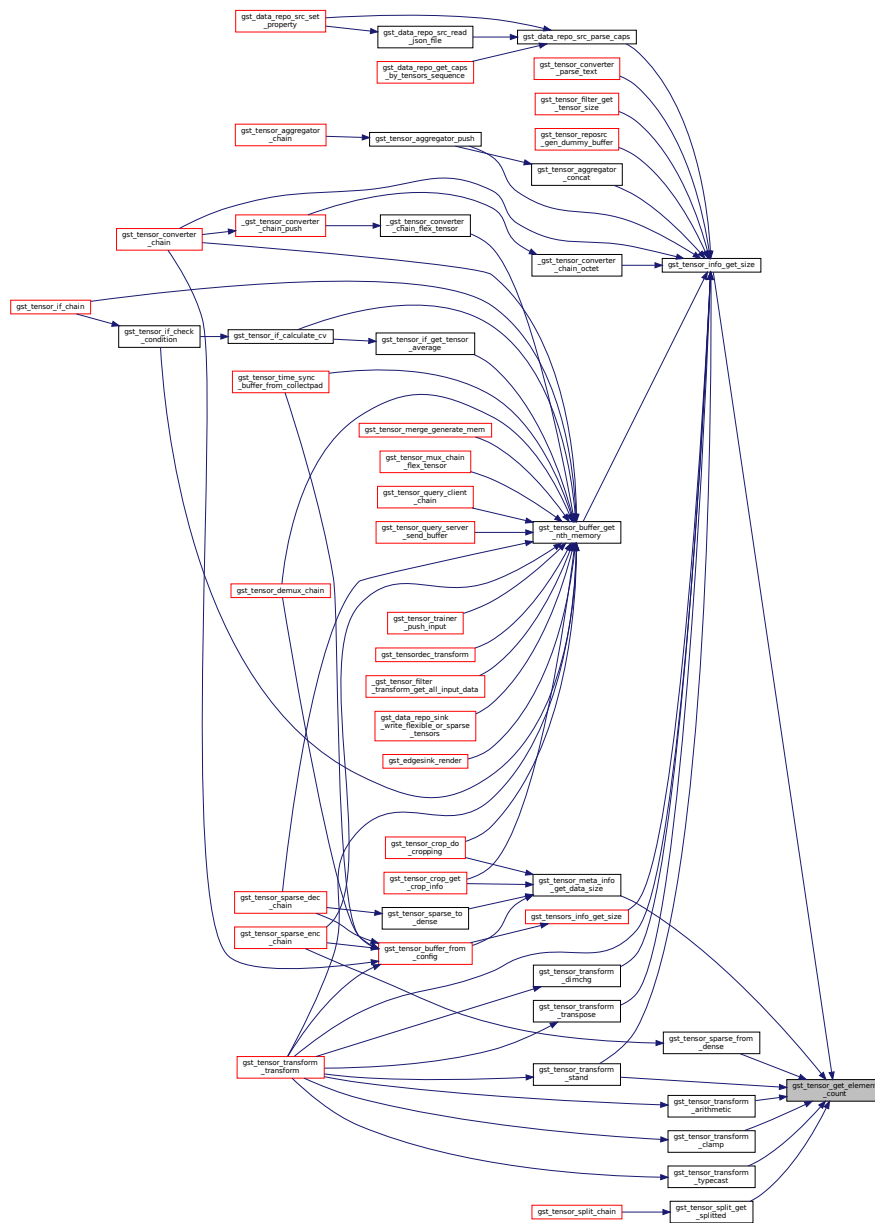
The number of elements. 0 if error.

Parameters

<i>dim</i>	The tensor dimension
------------	----------------------

Definition at line 1186 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



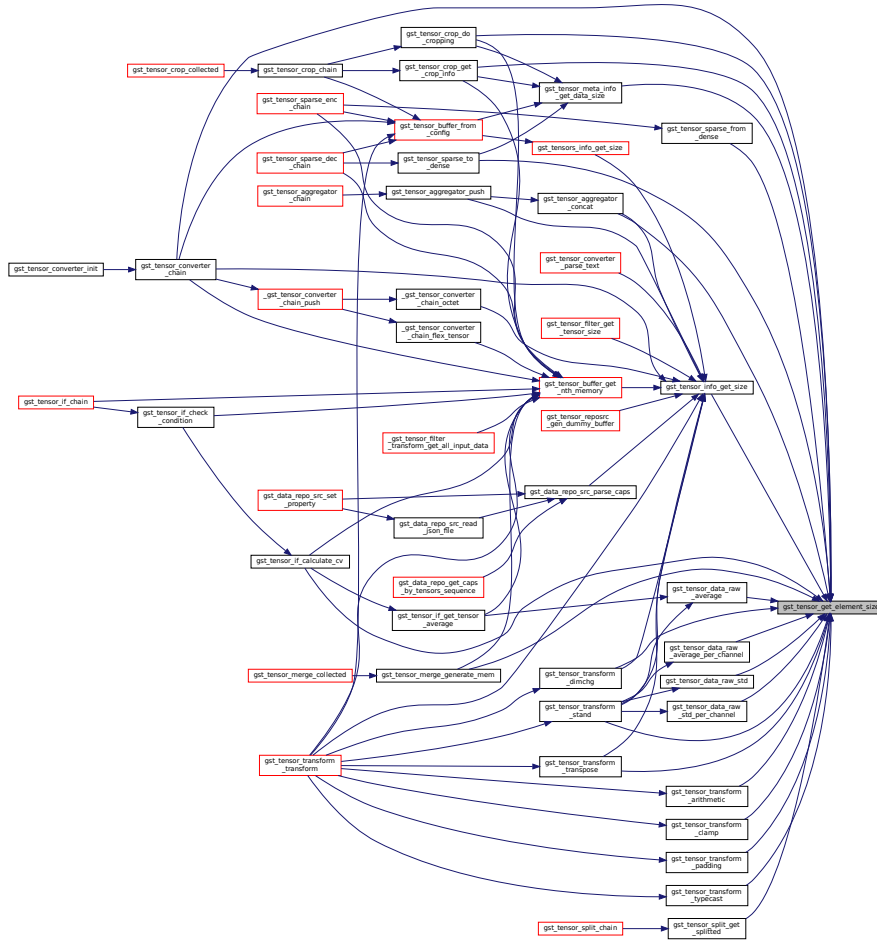
9.94.3.11 gst_tensor_get_element_size()

```
gsize gst_tensor_get_element_size (
    tensor_type type )
```

Get element size of tensor type (byte per element)

Definition at line 1205 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



9.94.3.12 gst_tensor_get_format()

```
tensor_format gst_tensor_get_format (
    const gchar * format_str )
```

Get tensor format from string input.

Parameters

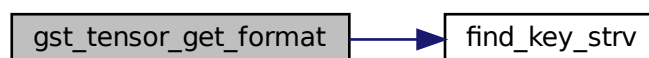
<i>format_str</i>	The string format name, supposed to be one of <code>tensor_format_name[]</code> .
-------------------	---

Returns

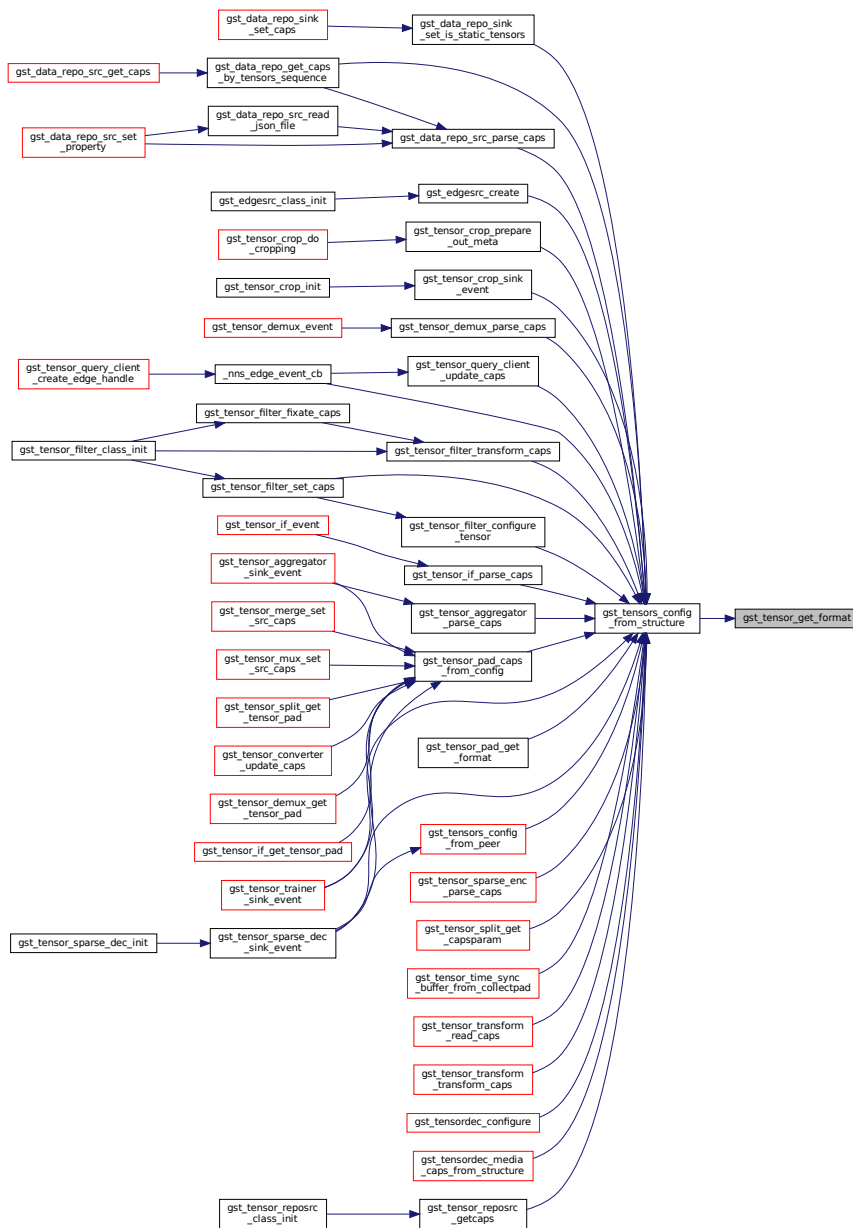
Corresponding `tensor_format`. `_NNS_TENSOR_FORMAT_END` if unrecognized value is there.

Definition at line 1309 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



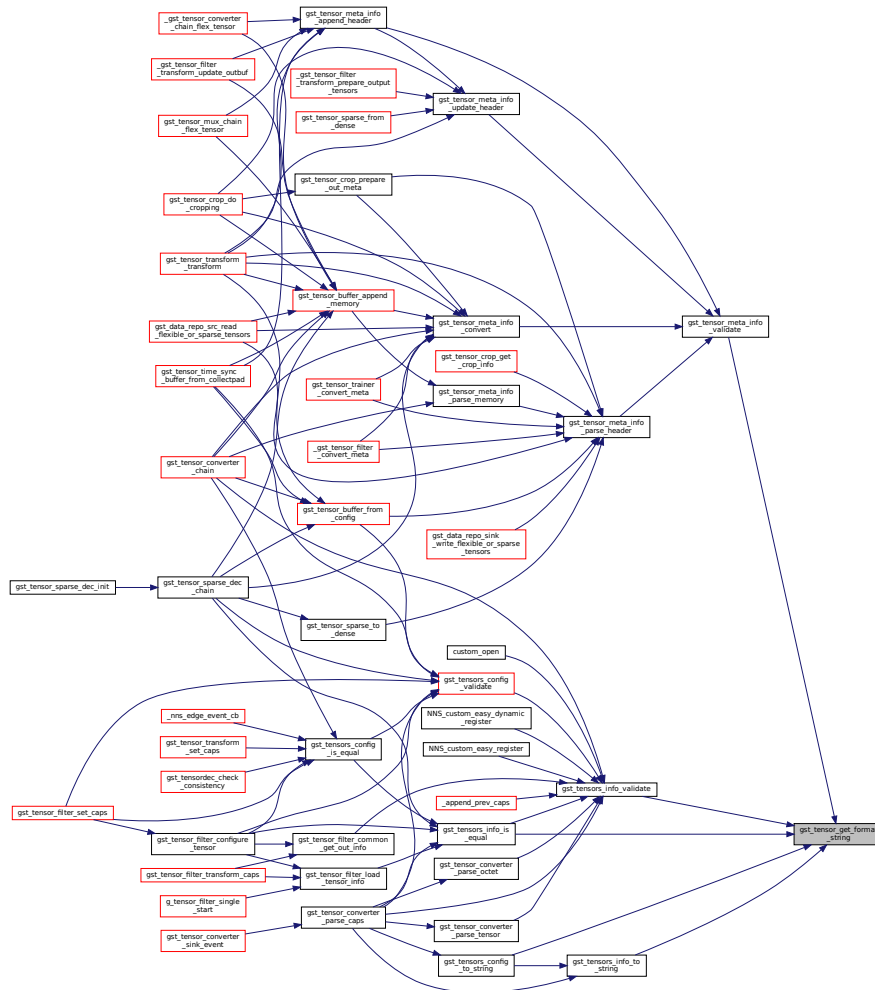
9.94.3.13 `gst_tensor_get_format_string()`

```
const gchar* gst_tensor_get_format_string (
    tensor_format format )
```

Get tensor format string.

Definition at line 1325 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the caller graph for this function:



9.94.3.14 `gst_tensor_get_rank_dimension_string()`

```
gchar* gst_tensor_get_rank_dimension_string (
    const tensor_dim dim,
    const unsigned int rank )
```

Get dimension string from given tensor dimension and rank count.

Parameters

<i>dim</i>	tensor dimension
<i>rank</i>	rank count of given tensor dimension

Parameters

in	<i>info</i>	GstTensorInfo to be converted
out	<i>meta</i>	tensor meta structure to be filled

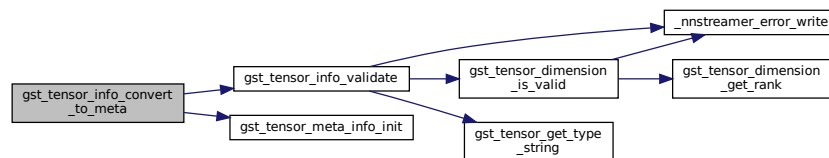
Returns

TRUE if successfully set the meta

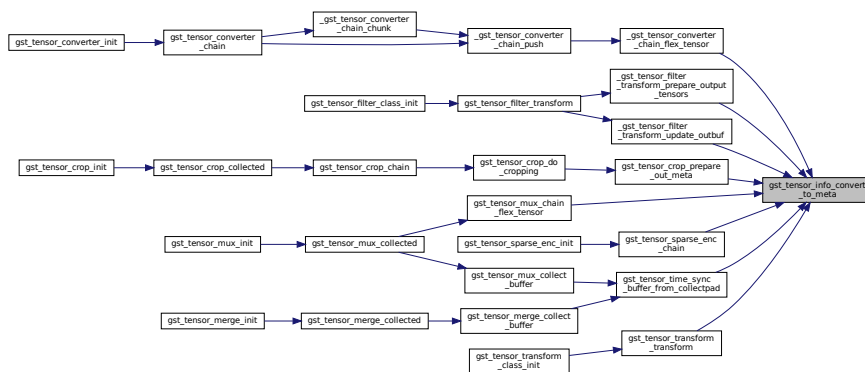
Todo handle rank from info.dimension

Definition at line 260 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.18 `gst_tensor_info_copy()`

```

void gst_tensor_info_copy (
    GstTensorInfo * dest,
    const GstTensorInfo * src )
  
```

Copy tensor info.

Note

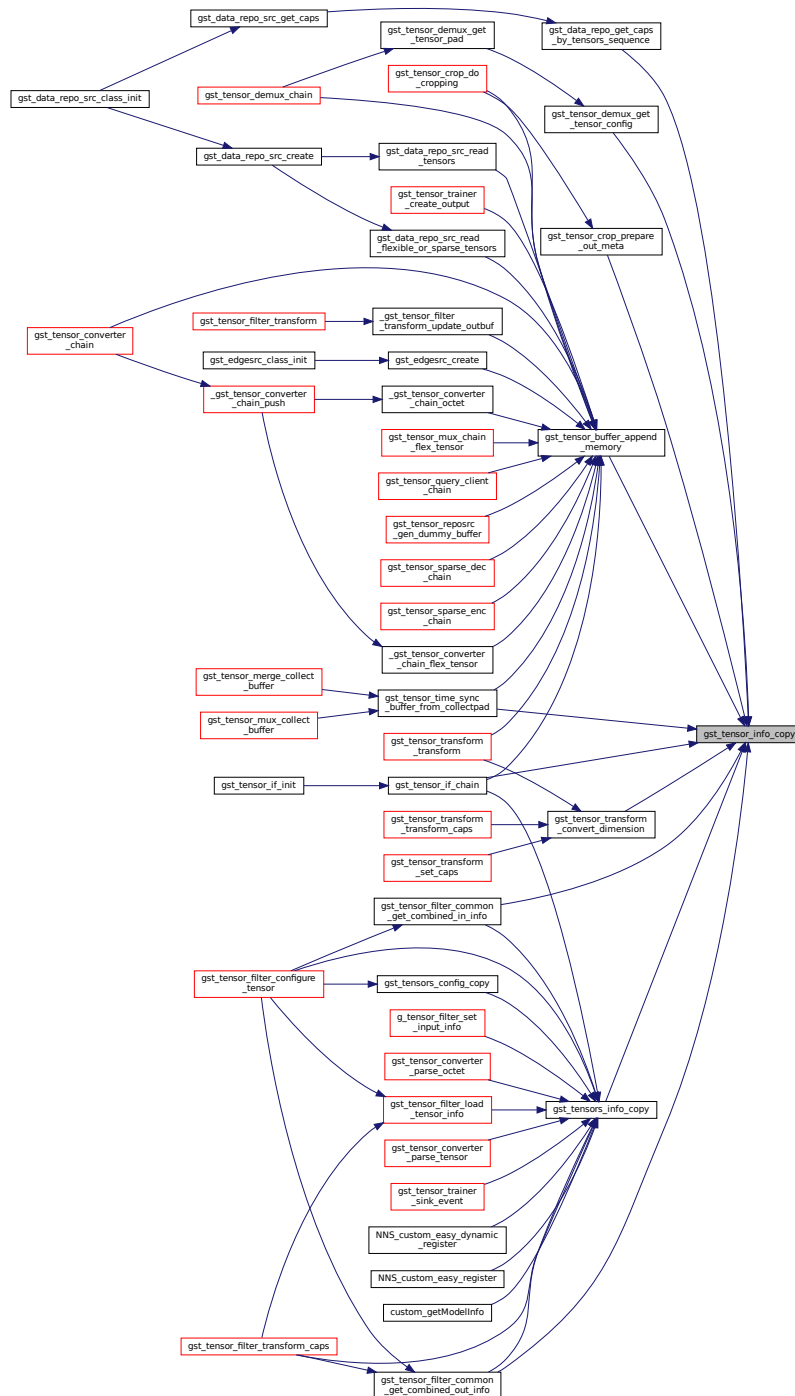
Copied info should be freed with [gst_tensor_info_free\(\)](#)

Definition at line 248 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.19 gst_tensor_info_copy_n()

```
void gst_tensor_info_copy_n (
    GstTensorInfo * dest,
```

```
const GstTensorInfo * src,
const guint n )
```

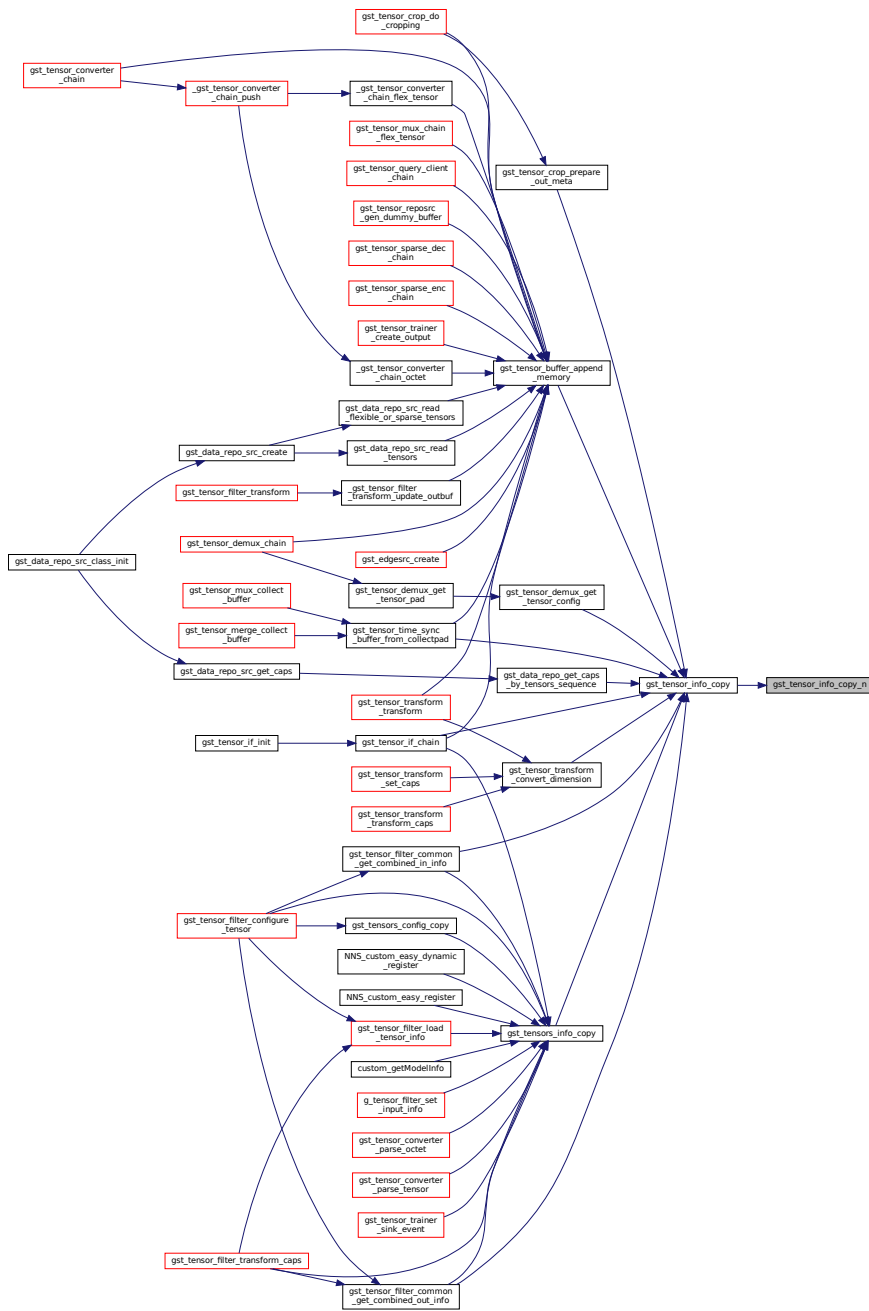
Copy tensor info up to n elements.

Note

Copied info should be freed with `gst_tensor_info_free()`

Definition at line 227 of file `nntstreamer_plugin_api_util_impl.c`.

Here is the caller graph for this function:



9.94.3.20 `gst_tensor_info_free()`

```
void gst_tensor_info_free (  
    GstTensorInfo * info )
```

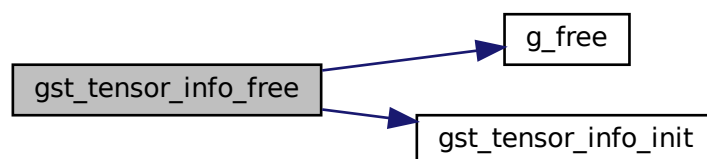
Free allocated data in tensor info structure.

Parameters

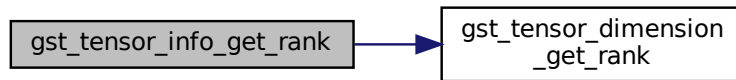
<i>info</i>	tensor info structure
-------------	-----------------------

Definition at line 140 of file `nntstreamer_plugin_api_util_impl.c`.

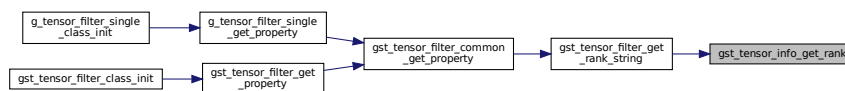
Here is the call graph for this function:



Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.22 gst_tensor_info_get_size()

```

gsize gst_tensor_info_get_size (
    const GstTensorInfo * info )
  
```

Get data size of single tensor.

Parameters

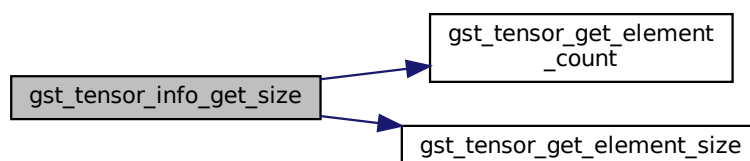
<i>info</i>	tensor info structure
-------------	-----------------------

Returns

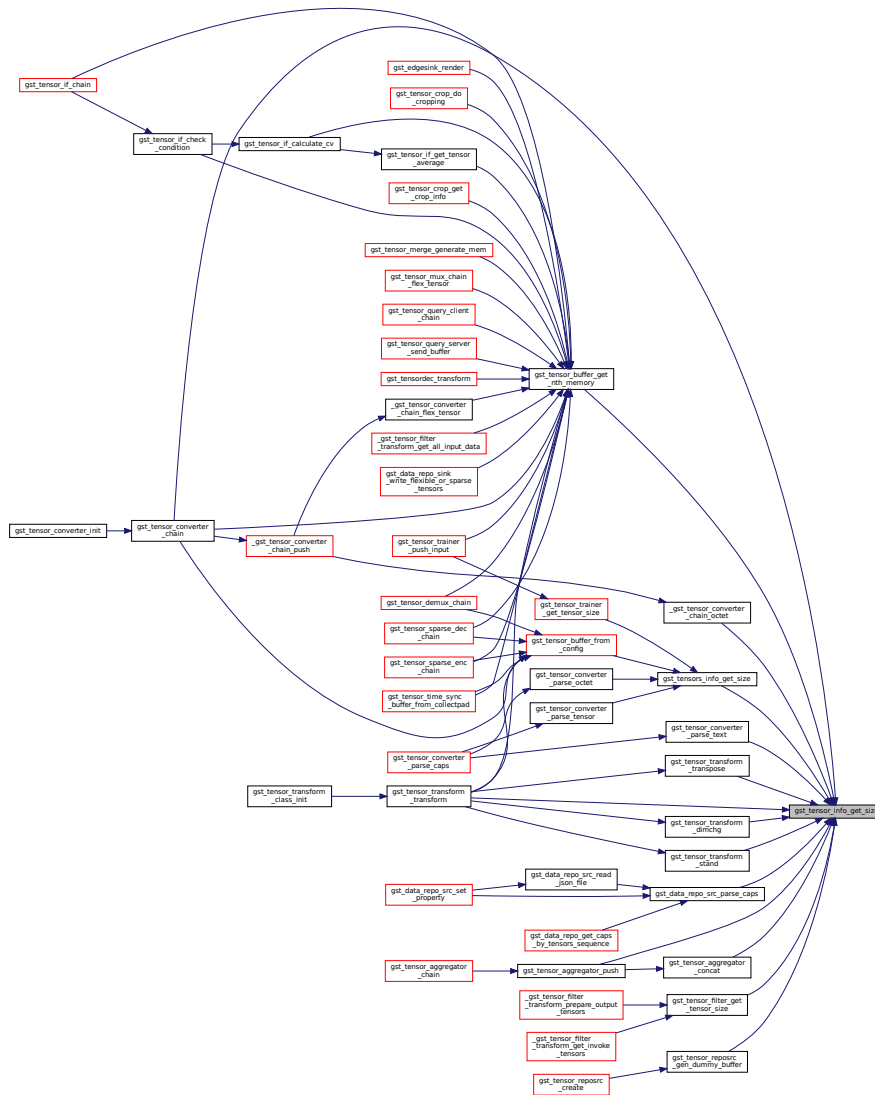
data size

Definition at line 156 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.23 gst_tensor_info_init()

```
void gst_tensor_info_init (
    GstTensorInfo * info )
```

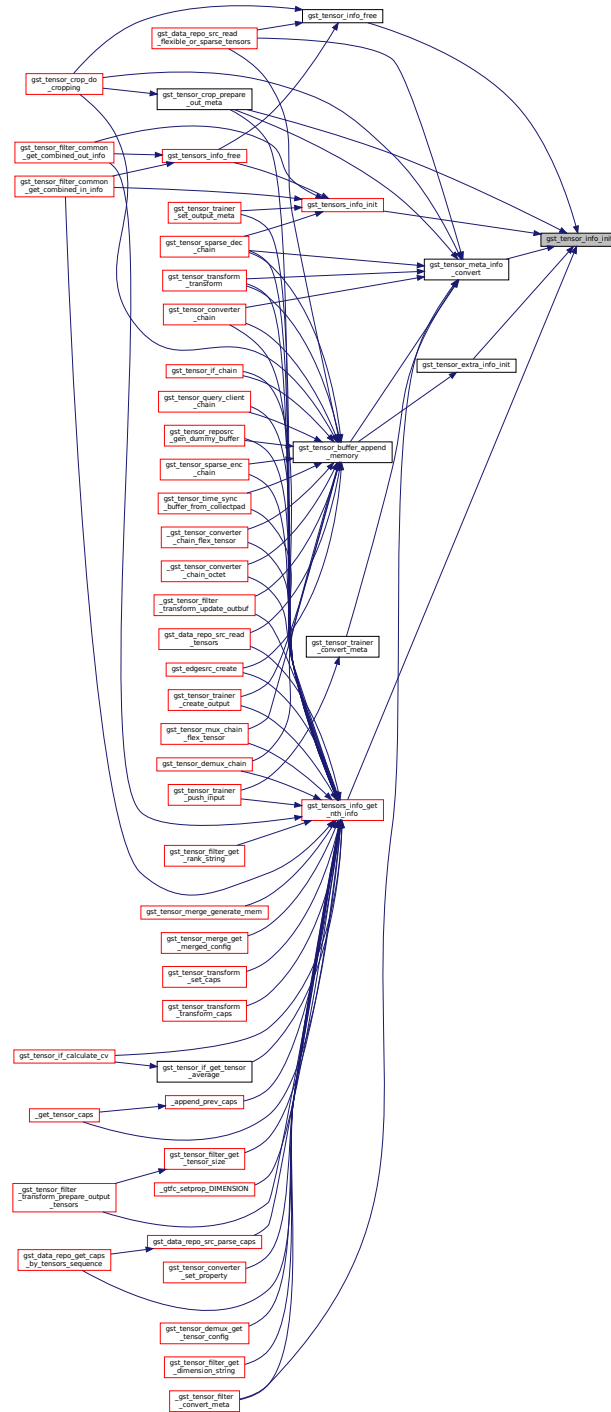
Initialize the tensor info structure.

Parameters

<i>info</i>	tensor info structure to be initialized
-------------	---

Definition at line 121 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



9.94.3.24 `gst_tensor_info_is_equal()`

```
gboolean gst_tensor_info_is_equal (
    const GstTensorInfo * i1,
    const GstTensorInfo * i2 )
```

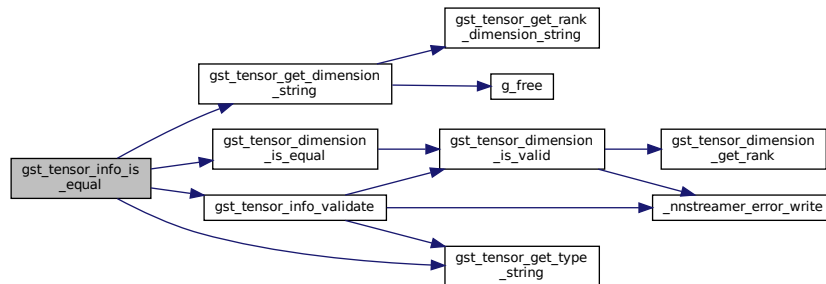
Compare tensor info.

Returns

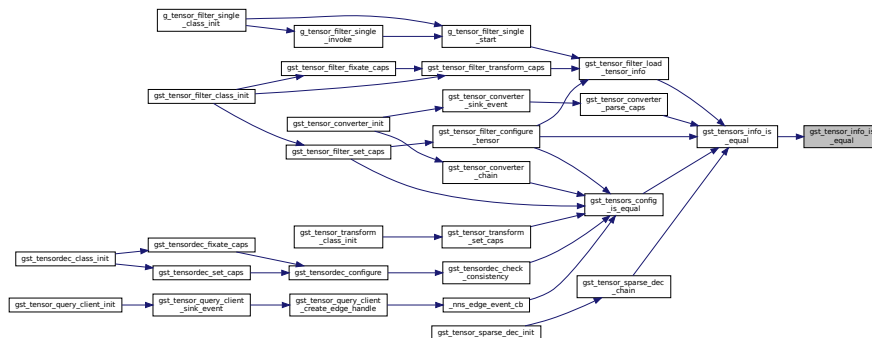
TRUE if equal, FALSE if given tensor infos are invalid or not equal.

Definition at line 197 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.25 gst_tensor_info_validate()

```
gboolean gst_tensor_info_validate (
    const GstTensorInfo * info )
```

Check the tensor info is valid.

Parameters

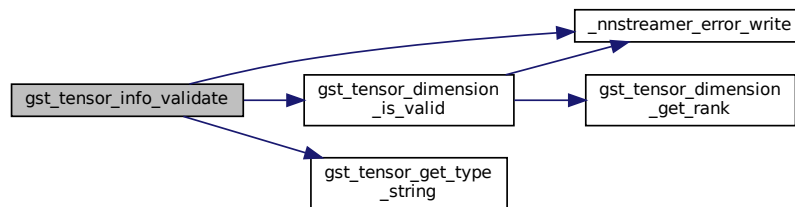
<i>info</i>	tensor info structure
-------------	-----------------------

Returns

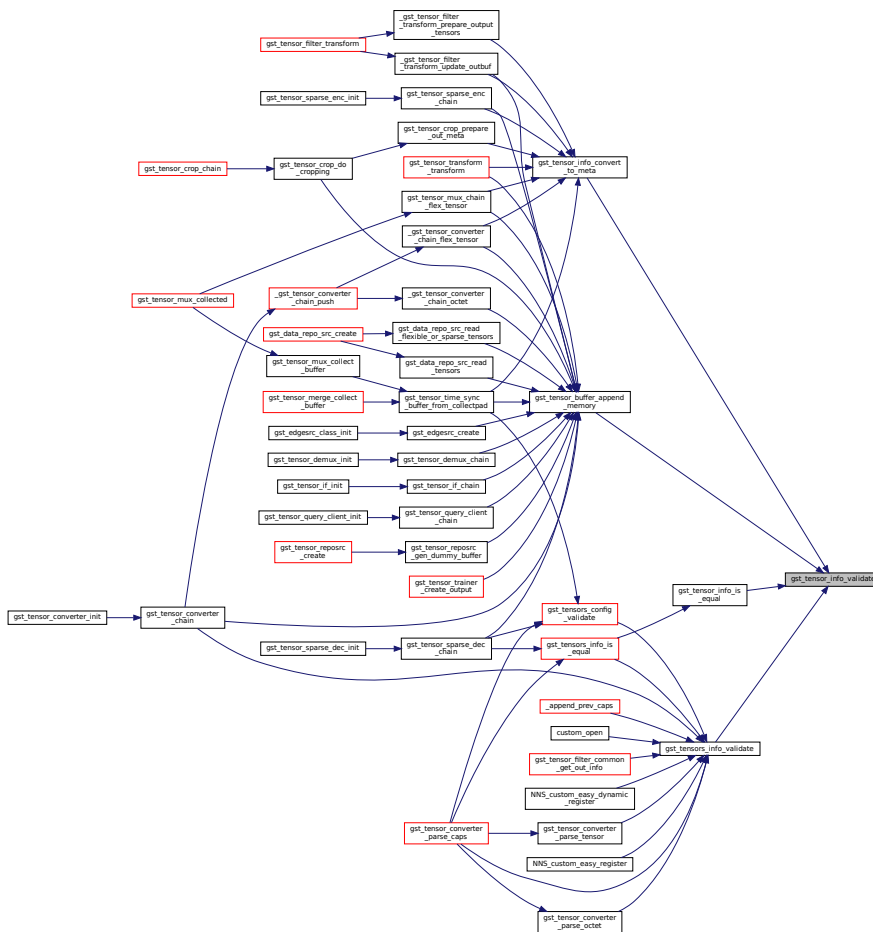
TRUE if info is valid

Definition at line 174 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.26 `gst_tensor_meta_info_convert()`

```
gboolean gst_tensor_meta_info_convert (
    GstTensorMetaInfo * meta,
    GstTensorInfo * info )
```

Convert `GstTensorMetaInfo` structure to `GstTensorInfo`.

Parameters

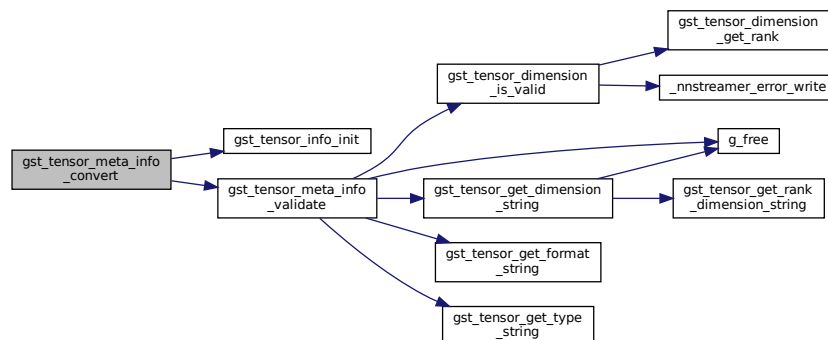
in	<i>meta</i>	tensor meta structure to be converted
out	<i>info</i>	<code>GstTensorInfo</code> to be filled

Returns

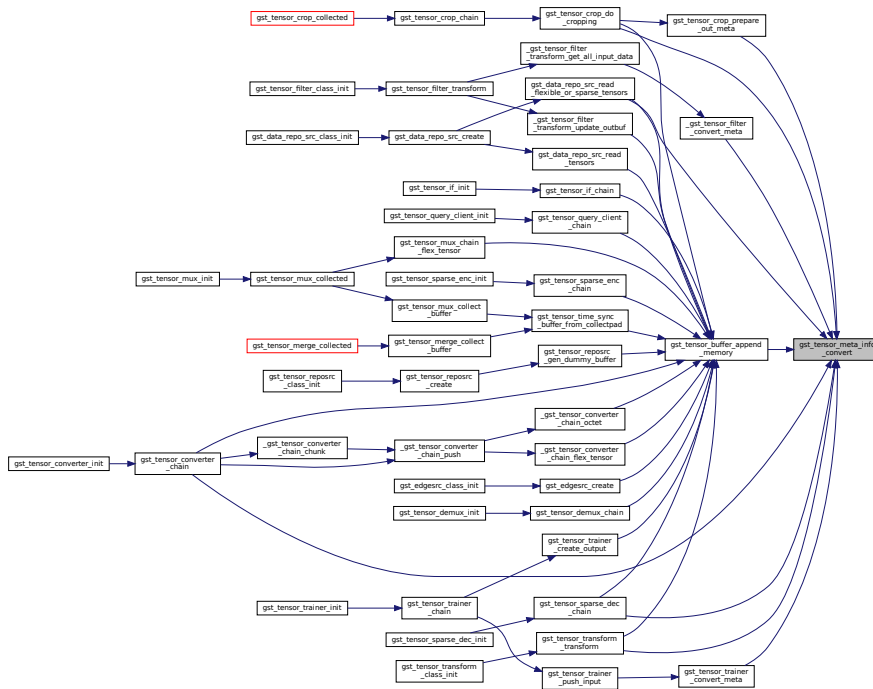
TRUE if successfully set the info

Definition at line 1562 of file `nntstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.27 gst_tensor_meta_info_get_data_size()

```
gsize gst_tensor_meta_info_get_data_size (
    GstTensorMetaInfo * meta )
```

Get the data size calculated from tensor meta.

Parameters

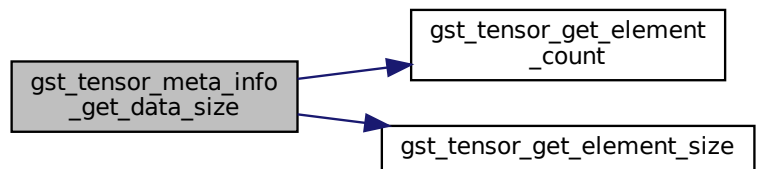
in	<i>meta</i>	tensor meta structure
----	-------------	-----------------------

Returns

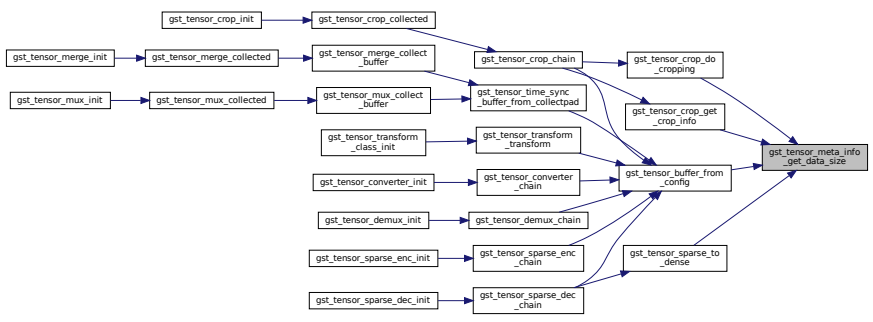
The data size for meta info (0 if meta is invalid)

Definition at line 1477 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.28 `gst_tensor_meta_info_get_header_size()`

```
gsize gst_tensor_meta_info_get_header_size (
    GstTensorMetaInfo * meta )
```

Get the header size to handle a tensor meta.

Parameters

in	<i>meta</i>	tensor meta structure
----	-------------	-----------------------

Returns

Header size for meta info (0 if meta is invalid)


```
GstTensorMetaInfo * meta )
```

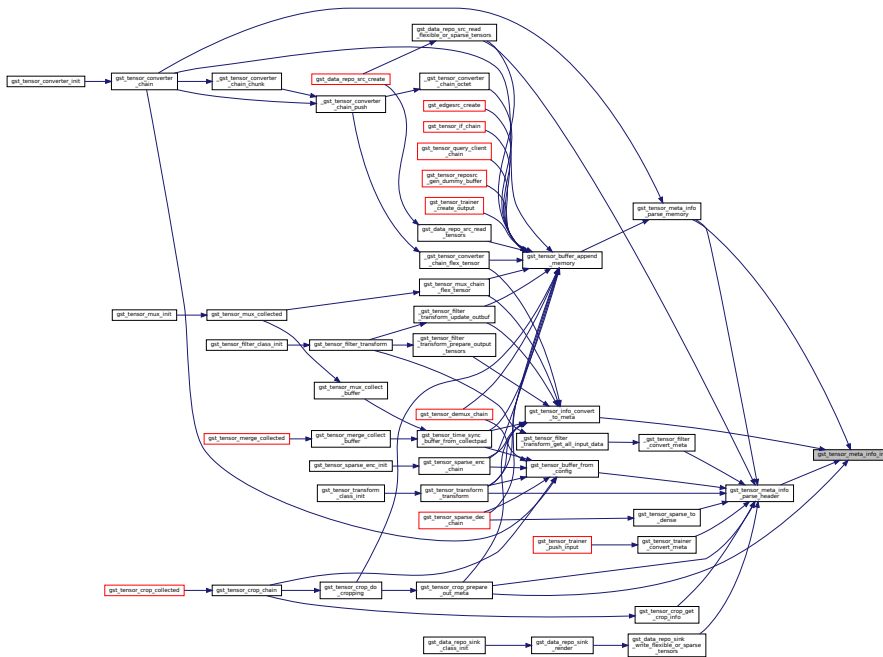
Initialize the tensor meta info structure.

Parameters

in, out	<i>meta</i>	tensor meta structure to be initialized
---------	-------------	---

Definition at line 1372 of file nnstreamer_plugin_api_util_impl.c.

Here is the caller graph for this function:



9.94.3.31 `gst_tensor_meta_info_parse_header()`

```
gboolean gst_tensor_meta_info_parse_header (
    GstTensorMetaInfo * meta,
    gpointer header )
```

Parse header and fill the tensor meta.

Parameters

out	<i>meta</i>	tensor meta structure to be filled
in	<i>header</i>	pointer to header to be parsed

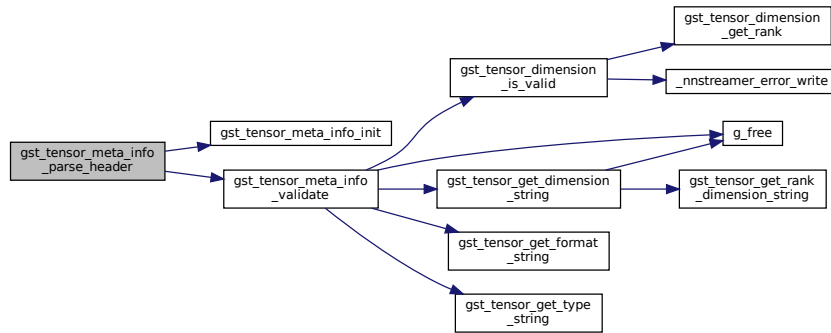
Returns

TRUE if successfully set the meta

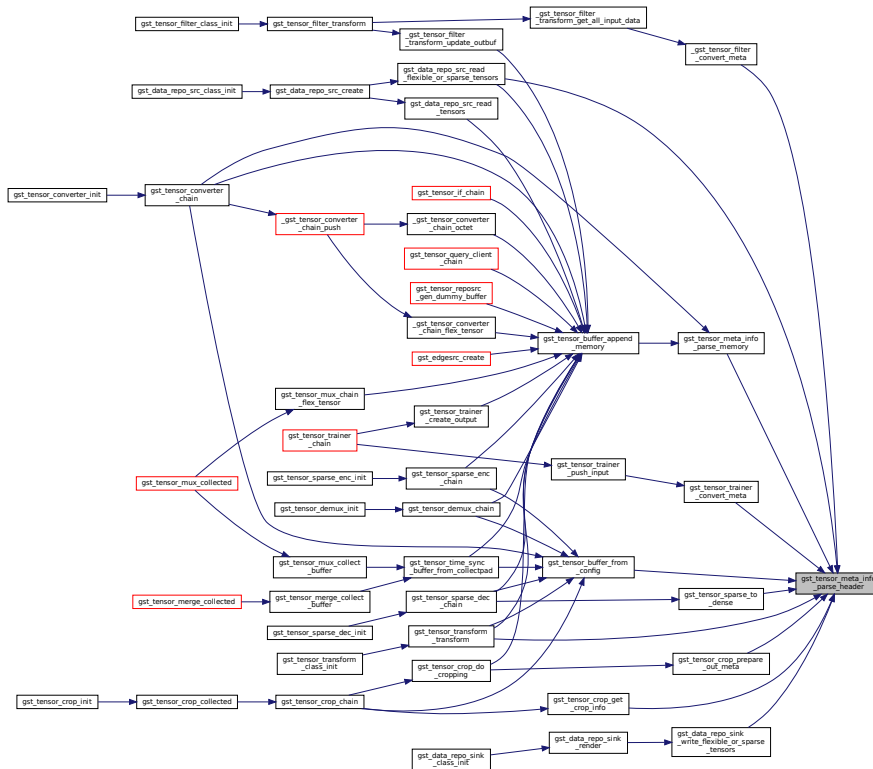
Todo update meta info for each version

Definition at line 1527 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.32 `gst_tensor_meta_info_update_header()`

```
gboolean gst_tensor_meta_info_update_header (
    GstTensorMetaInfo * meta,
    gpointer header )
```

Update header from tensor meta.

Parameters

in	<i>meta</i>	tensor meta structure
out	<i>header</i>	pointer to header to be updated

Returns

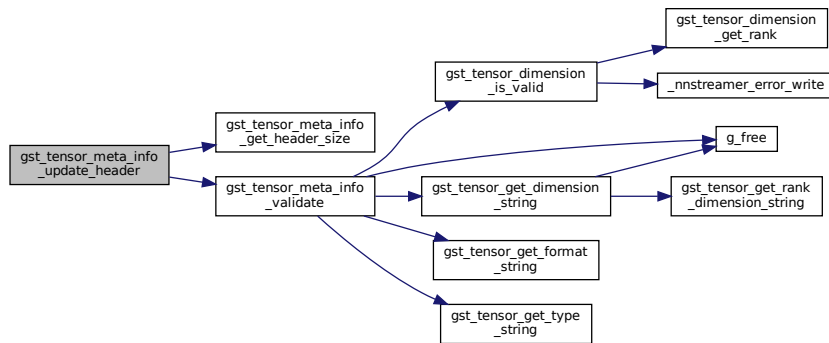
TRUE if successfully set the header

Note

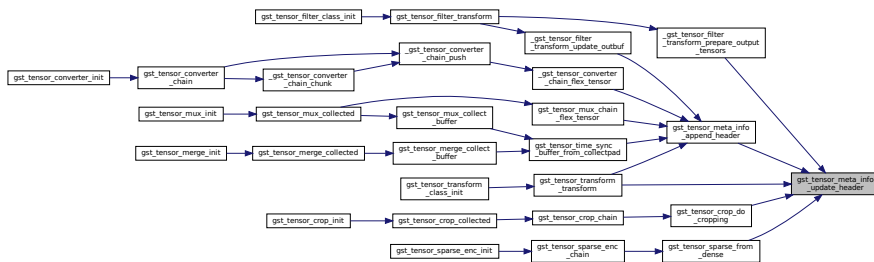
User should allocate enough memory for header (see [gst_tensor_meta_info_get_header_size\(\)](#)).

Definition at line 1505 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.33 `gst_tensor_meta_info_validate()`

```
gboolean gst_tensor_meta_info_validate (
    GstTensorMetaInfo * meta )
```

Check the meta info is valid.

Parameters

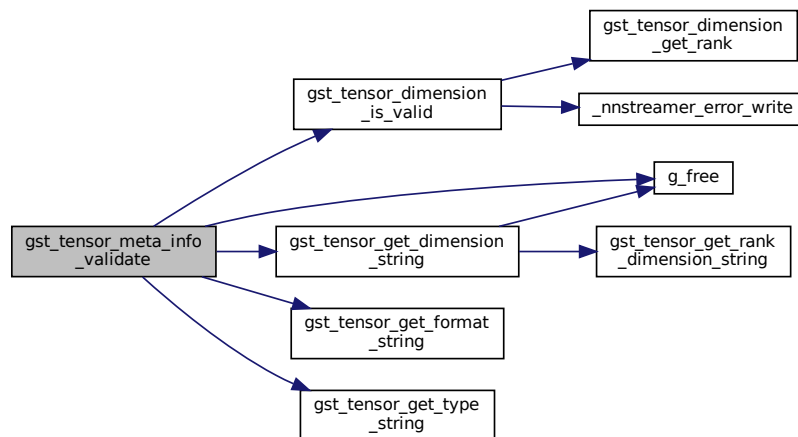
in	<i>meta</i>	tensor meta structure
----	-------------	-----------------------

Returns

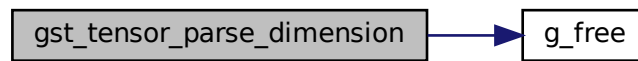
TRUE if given meta is valid

Definition at line 1414 of file `nnstreamer_plugin_api_util_impl.c`.

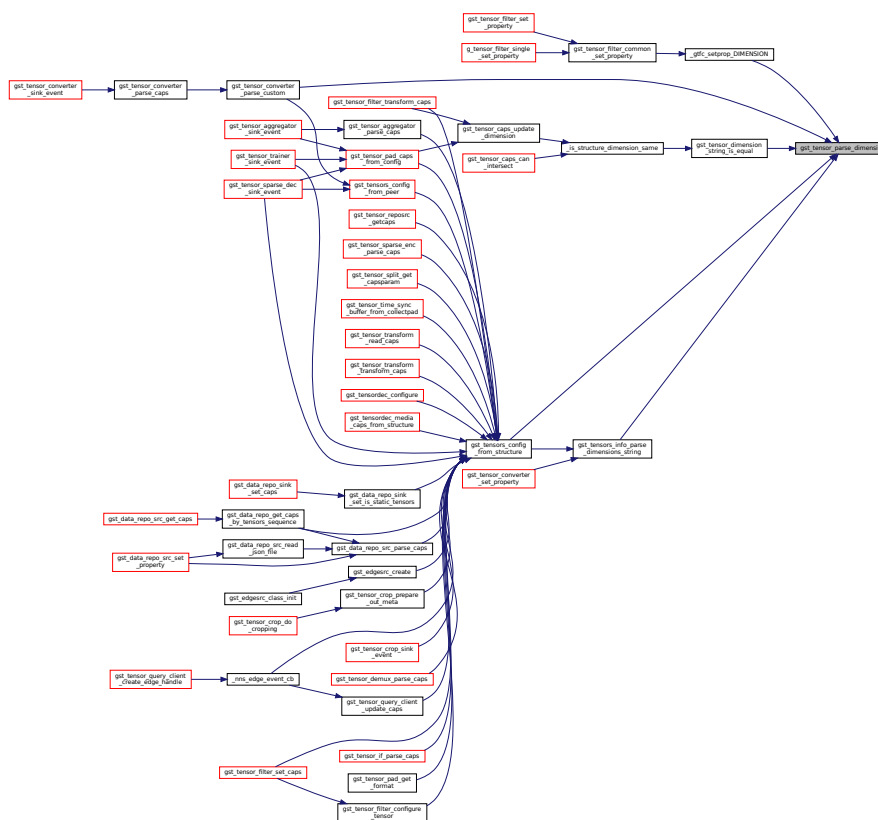
Here is the call graph for this function:



Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.35 gst_tensors_config_copy()

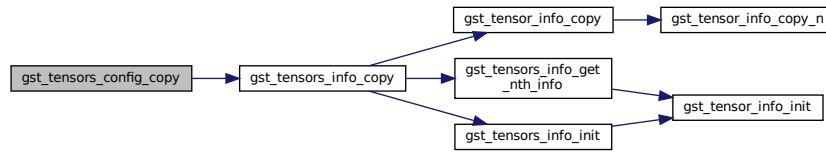
```

void gst_tensors_config_copy (
    GstTensorsConfig * dest,
    const GstTensorsConfig * src )
  
```

Copy tensors config.

Definition at line 904 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.36 gst_tensors_config_free()

```
void gst_tensors_config_free (
    GstTensorsConfig * config )
```

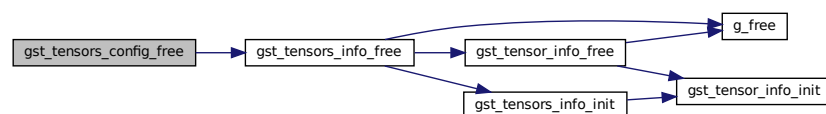
Free allocated data in tensors config structure.

Parameters

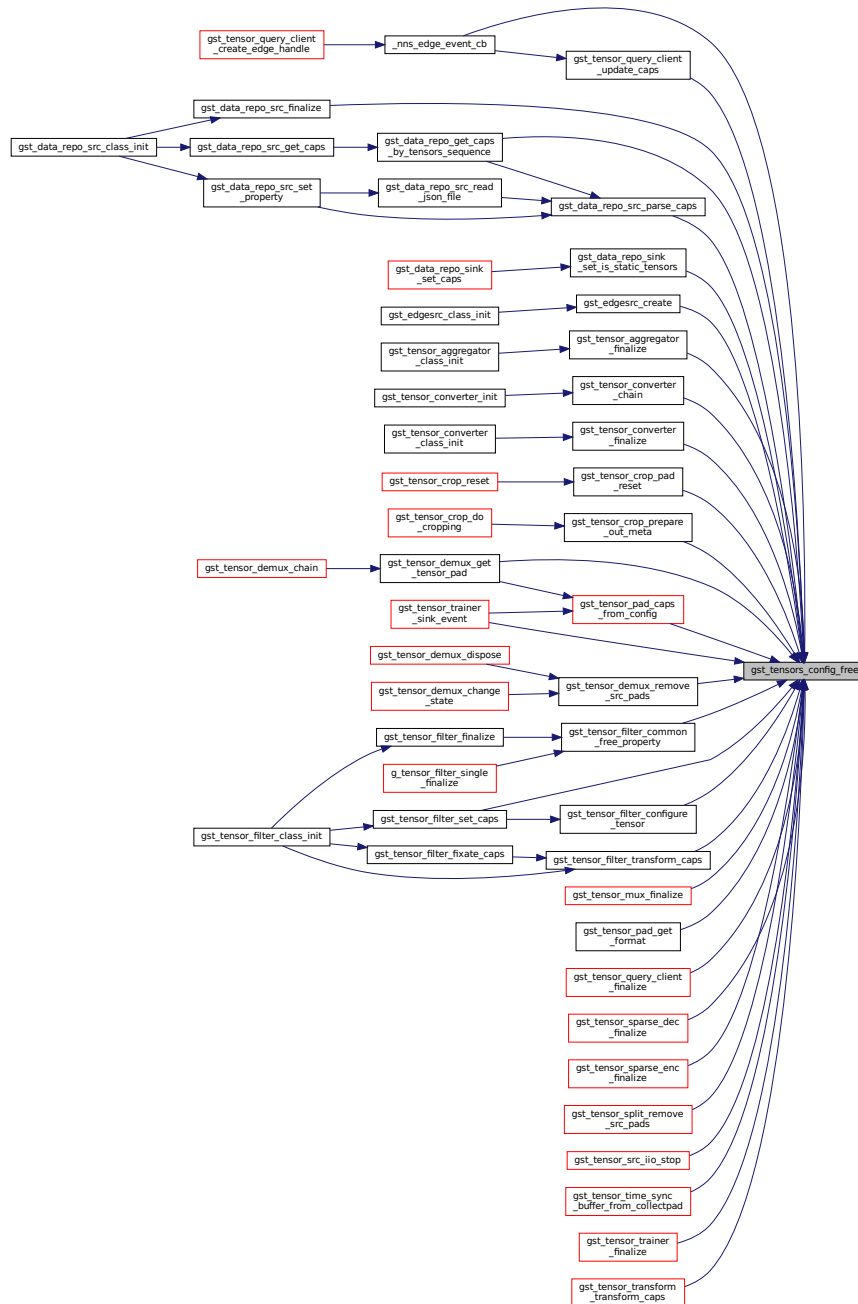
<i>config</i>	tensors config structure
---------------	--------------------------

Definition at line 845 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.37 gst_tensors_config_init()

```
void gst_tensors_config_init (
    GstTensorsConfig * config )
```

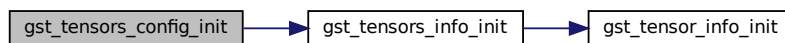
Initialize the tensors config info structure (for other/tensors)

Parameters

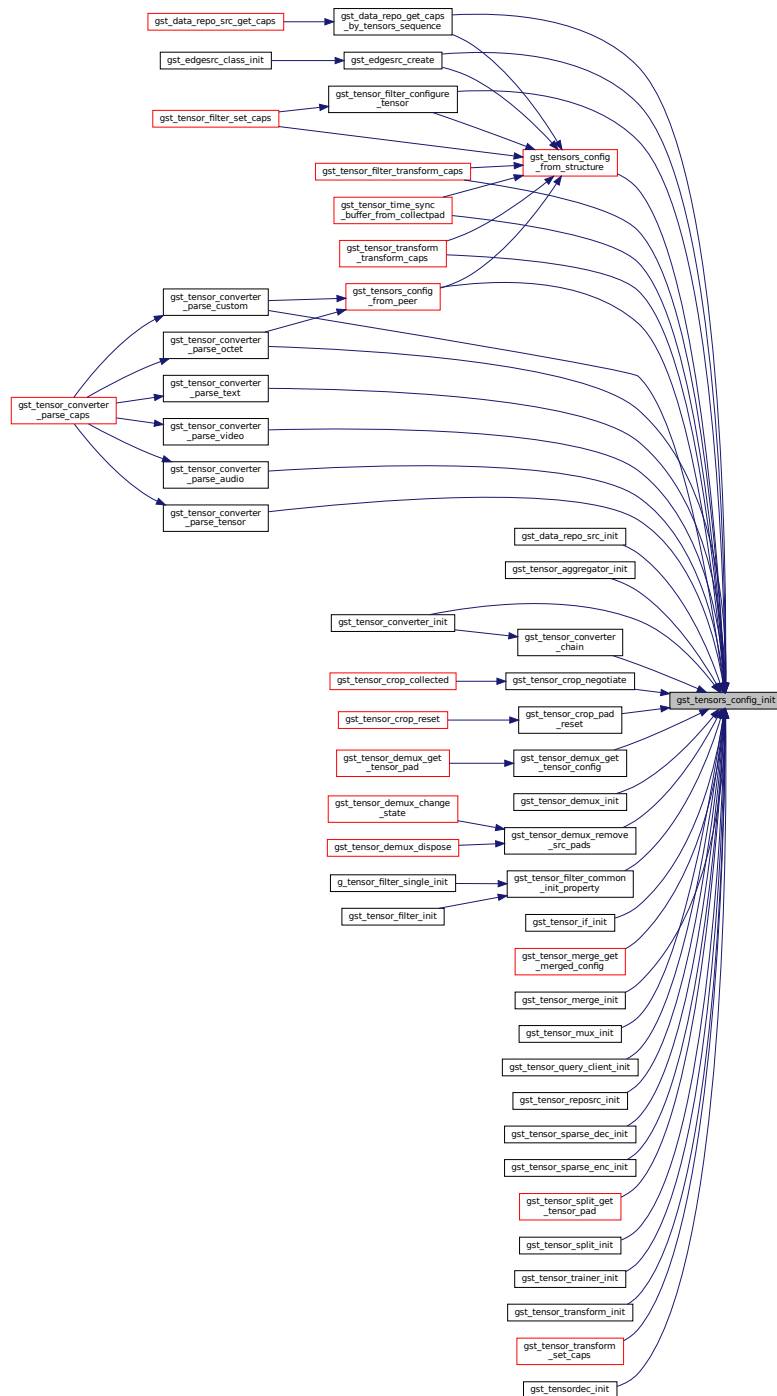
<i>config</i>	tensors config structure to be initialized
---------------	--

Definition at line 830 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.38 `gst_tensors_config_is_equal()`

```
gboolean gst_tensors_config_is_equal (
    const GstTensorsConfig * c1,
    const GstTensorsConfig * c2 )
```

Compare tensor config info.

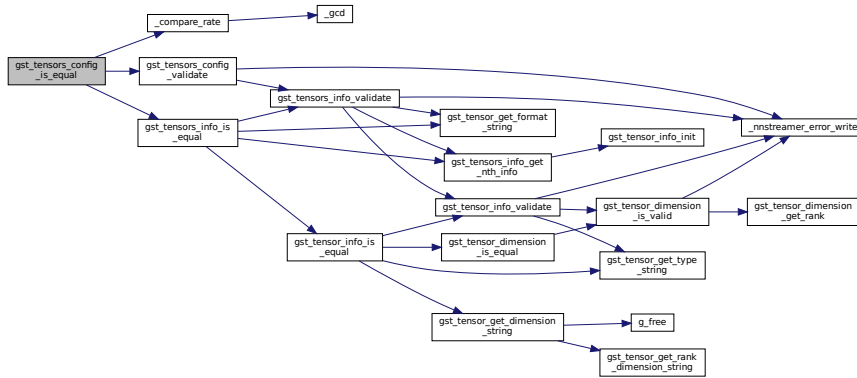
Compare tensor config info (for other/tensors)

Parameters

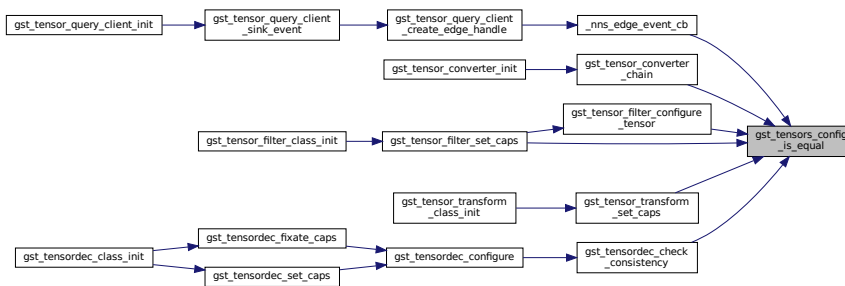
<i>TRUE</i>	if equal
-------------	----------

Definition at line 881 of file nnnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.39 gst_tensors_config_to_string()

```

gchar* gst_tensors_config_to_string (
    const GstTensorsConfig * config )
    
```

Tensor config represented as a string. Caller should free it.

Parameters

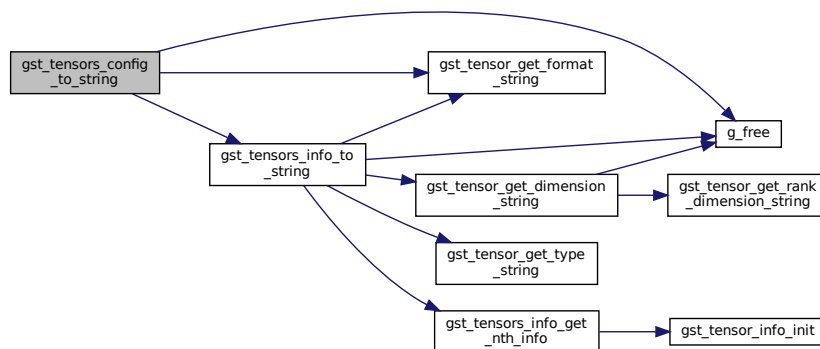
<i>config</i>	tensor config structure
---------------	-------------------------

Returns

The newly allocated string representing the config. Free after use.

Definition at line 920 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.40 gst_tensors_config_validate()

```

gboolean gst_tensors_config_validate (
    const GstTensorsConfig * config )
  
```

Check the tensors are all configured.

Check the tensors are all configured (for other/tensors)

Parameters

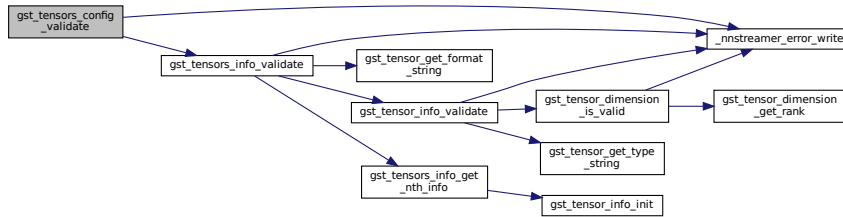
<i>config</i>	tensor config structure
---------------	-------------------------

Returns

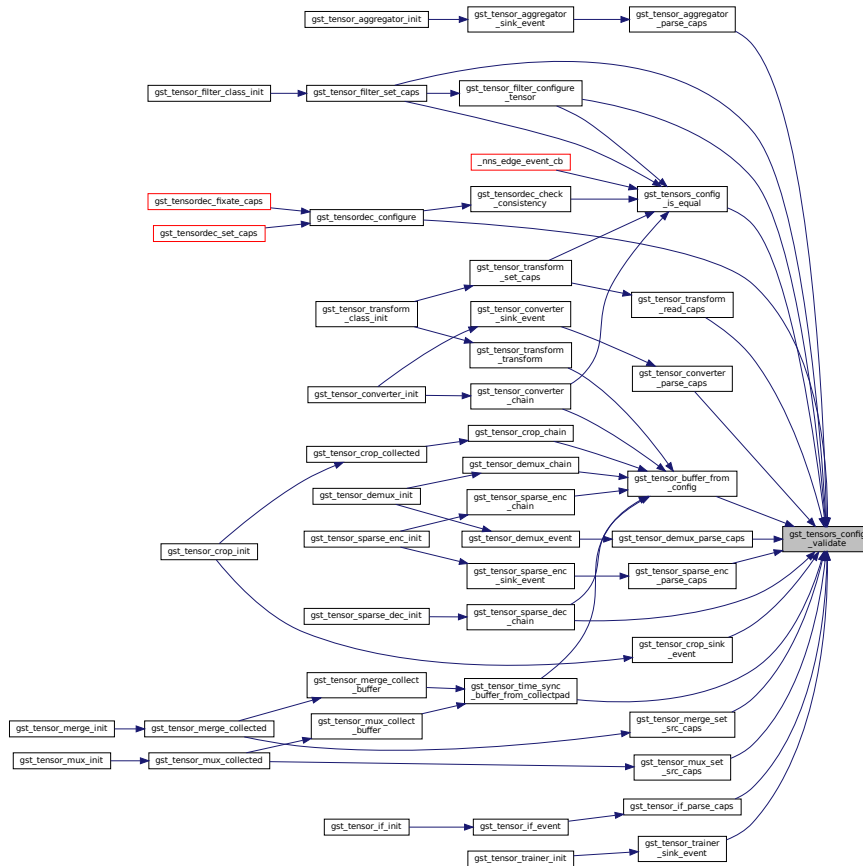
TRUE if configured

Definition at line 858 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.41 `gst_tensors_info_copy()`

```
void gst_tensors_info_copy (
    GstTensorsInfo * dest,
    const GstTensorsInfo * src )
```

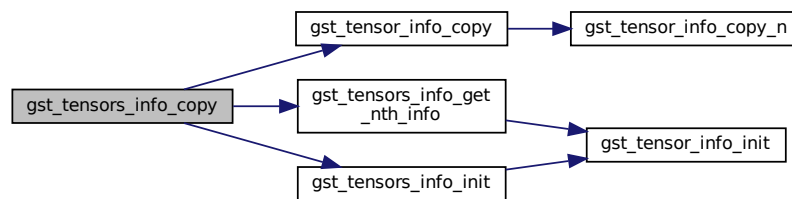
Copy tensor info.

Note

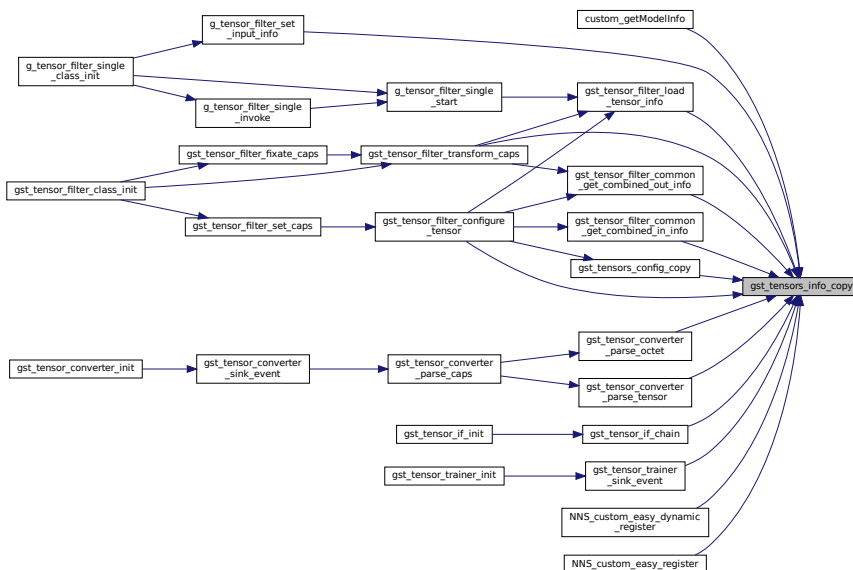
Copied info should be freed with `gst_tensors_info_free()`

Definition at line 502 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.42 `gst_tensors_info_free()`

```
void gst_tensors_info_free (
    GstTensorsInfo * info )
```

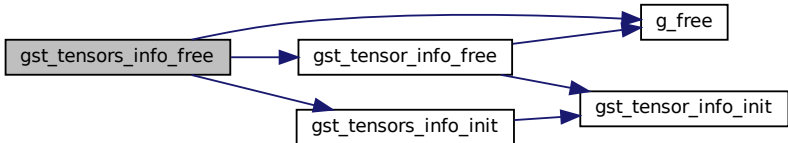
Free allocated data in tensors info structure.

Parameters

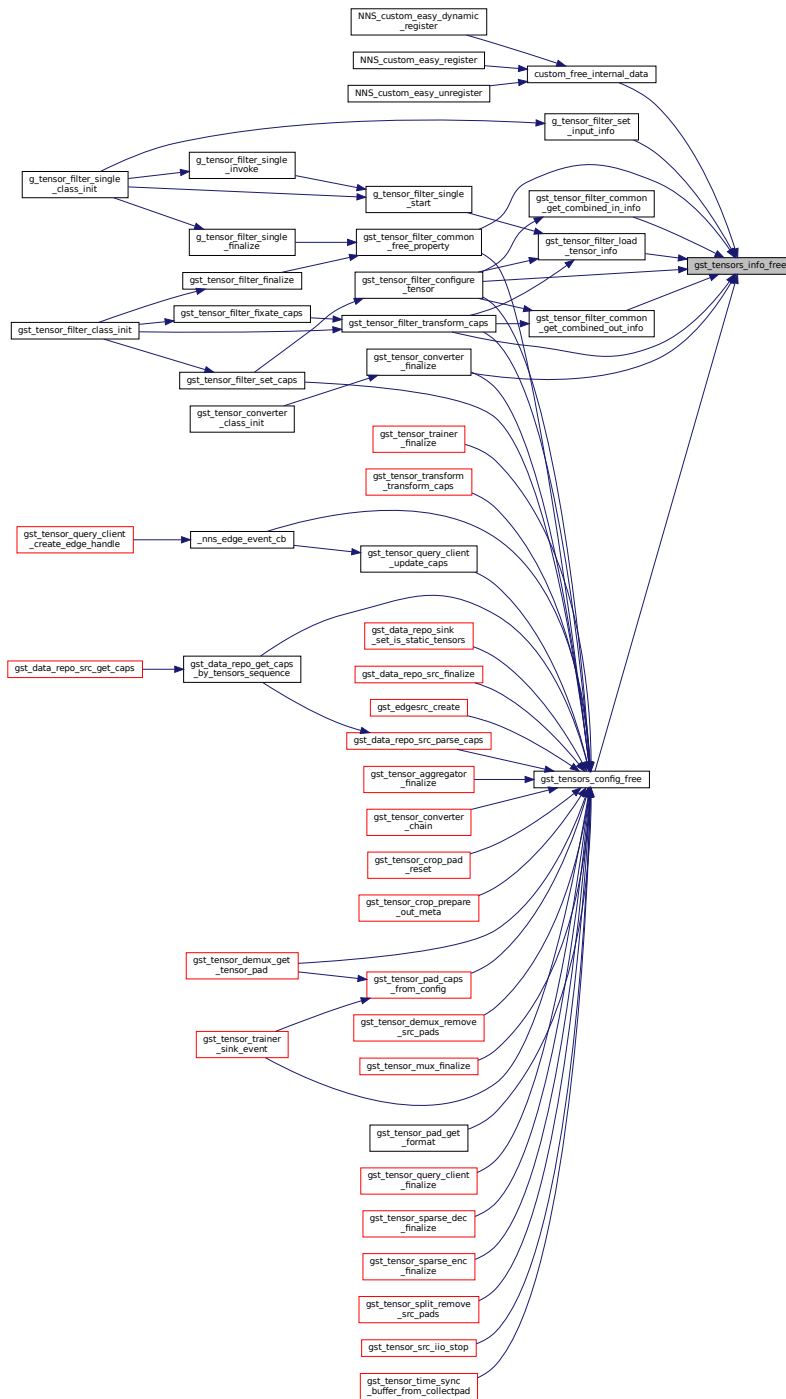
<i>info</i>	tensors info structure
-------------	------------------------

Definition at line 347 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.43 `gst_tensors_info_get_dimensions_string()`

```
gchar* gst_tensors_info_get_dimensions_string (
    const GstTensorsInfo * info )
```

Get the string of dimensions in tensors info.

Parameters

<i>info</i>	tensors info structure
-------------	------------------------

Returns

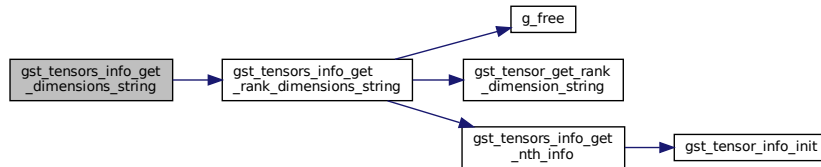
string of dimensions in tensors info (NULL if the number of tensors is 0)

Note

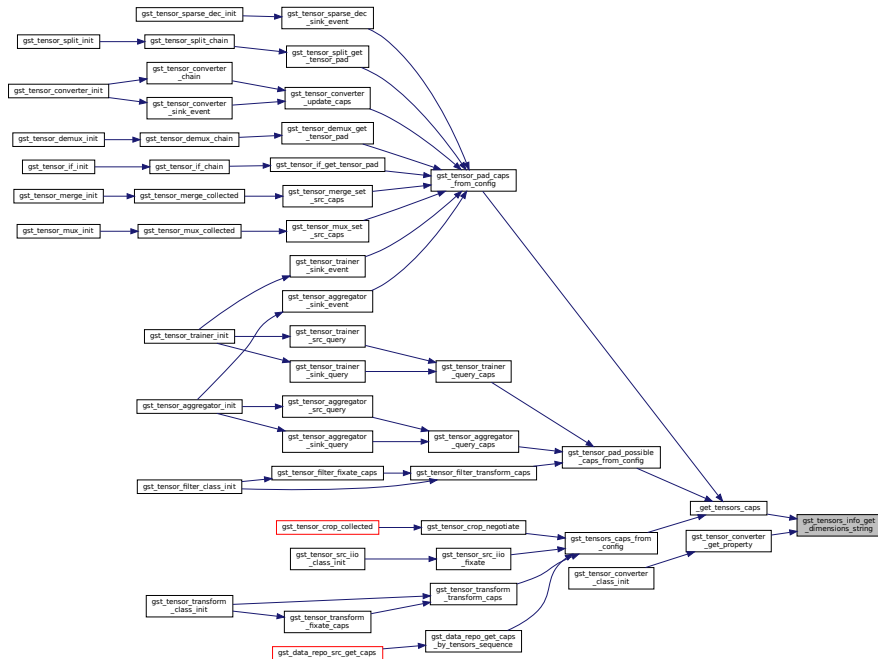
The returned value should be freed with [g_free\(\)](#)

Definition at line 661 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.44 `gst_tensors_info_get_names_string()`

```
gchar* gst_tensors_info_get_names_string (
    const GstTensorsInfo * info )
```

Get the string of tensor names in tensors info.

Parameters

<i>info</i>	tensors info structure
-------------	------------------------

Returns

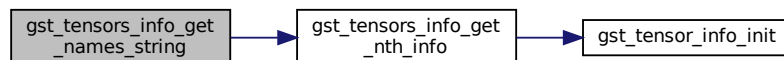
string of names in tensors info (NULL if the number of tensors is 0)

Note

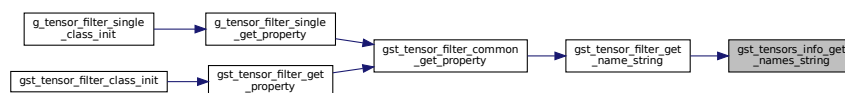
The returned value should be freed with `g_free()`

Definition at line 749 of file `nstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.45 `gst_tensors_info_get_nth_info()`

```
GstTensorInfo* gst_tensors_info_get_nth_info (
    GstTensorsInfo * info,
    guint index )
```

Get the pointer of nth tensor information.

Parameters

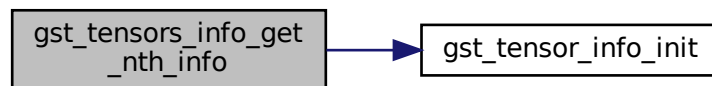
<i>info</i>	tensors info structure
<i>index</i>	the index of tensor to be fetched

Returns

The pointer to tensor info structure

Definition at line 296 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



9.94.3.46 `gst_tensors_info_get_rank_dimensions_string()`

```

gchar* gst_tensors_info_get_rank_dimensions_string (
    const GstTensorsInfo * info,
    const unsigned int rank )
  
```

Get the string of dimensions in tensors info and rank count.

Parameters

<i>info</i>	tensors info structure
<i>rank</i>	rank count of given tensor dimension

Returns

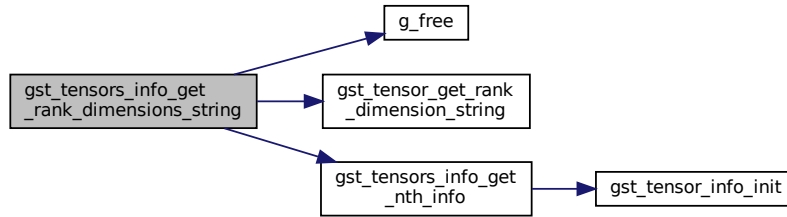
Formatted string of given dimension

Note

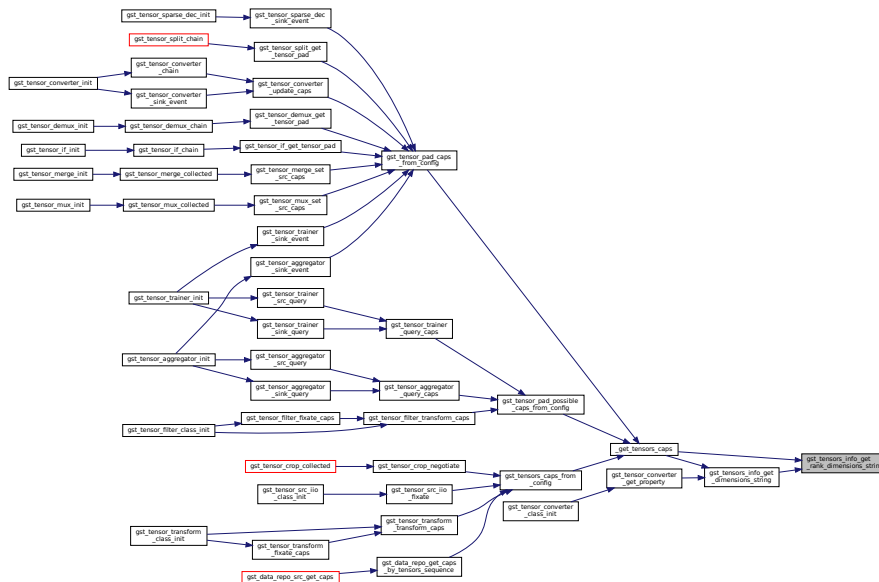
If rank count is 3, then returned string is 'd1:d2:d3'. The returned value should be freed with [g_free\(\)](#)

Definition at line 676 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.47 gst_tensors_info_get_size()

```

gsize gst_tensors_info_get_size (
    const GstTensorsInfo * info,
    gint index )
    
```

Get data size of single tensor.

Parameters

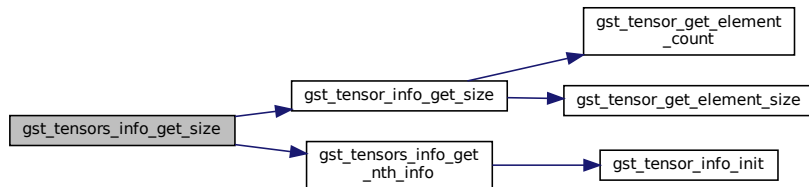
<i>info</i>	tensors info structure
<i>index</i>	the index of tensor (-1 to get total size of tensors)

Returns

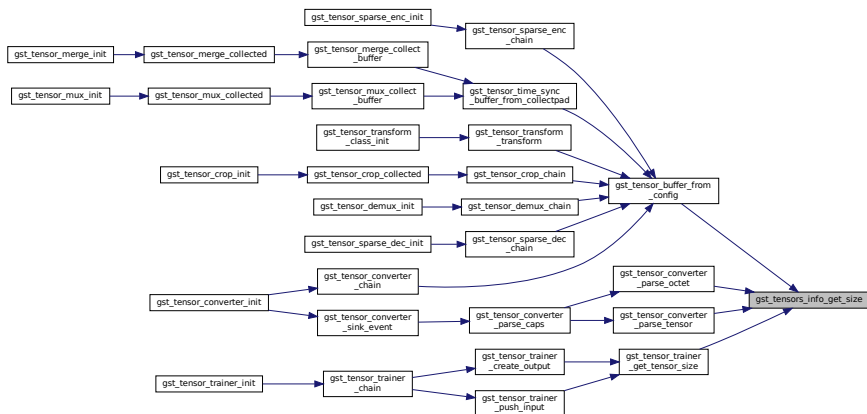
data size

Definition at line 376 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.48 `gst_tensors_info_get_types_string()`

```

gchar* gst_tensors_info_get_types_string (
    const GstTensorsInfo * info )
    
```

Get the string of types in tensors info.

Parameters

<i>info</i>	tensors info structure
-------------	------------------------

Returns

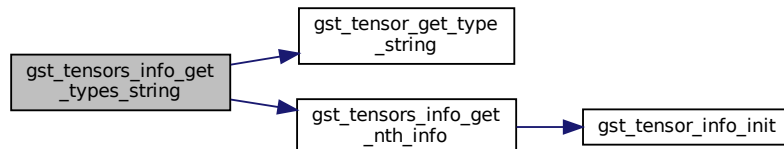
string of types in tensors info (NULL if the number of tensors is 0)

Note

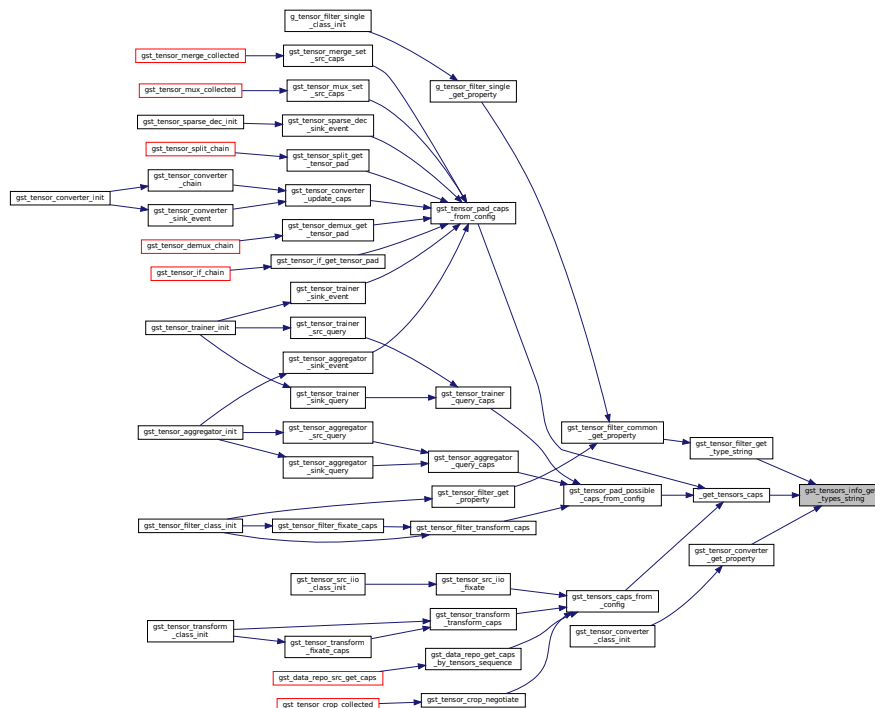
The returned value should be freed with [g_free\(\)](#)

Definition at line 714 of file `nnstreamer_plugin_api_util_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.94.3.49 gst_tensors_info_init()**

```
void gst_tensors_info_init (
    GstTensorsInfo * info )
```

Initialize the tensors info structure.

Parameters

<i>info</i>	tensors info structure to be initialized
-------------	--

Note

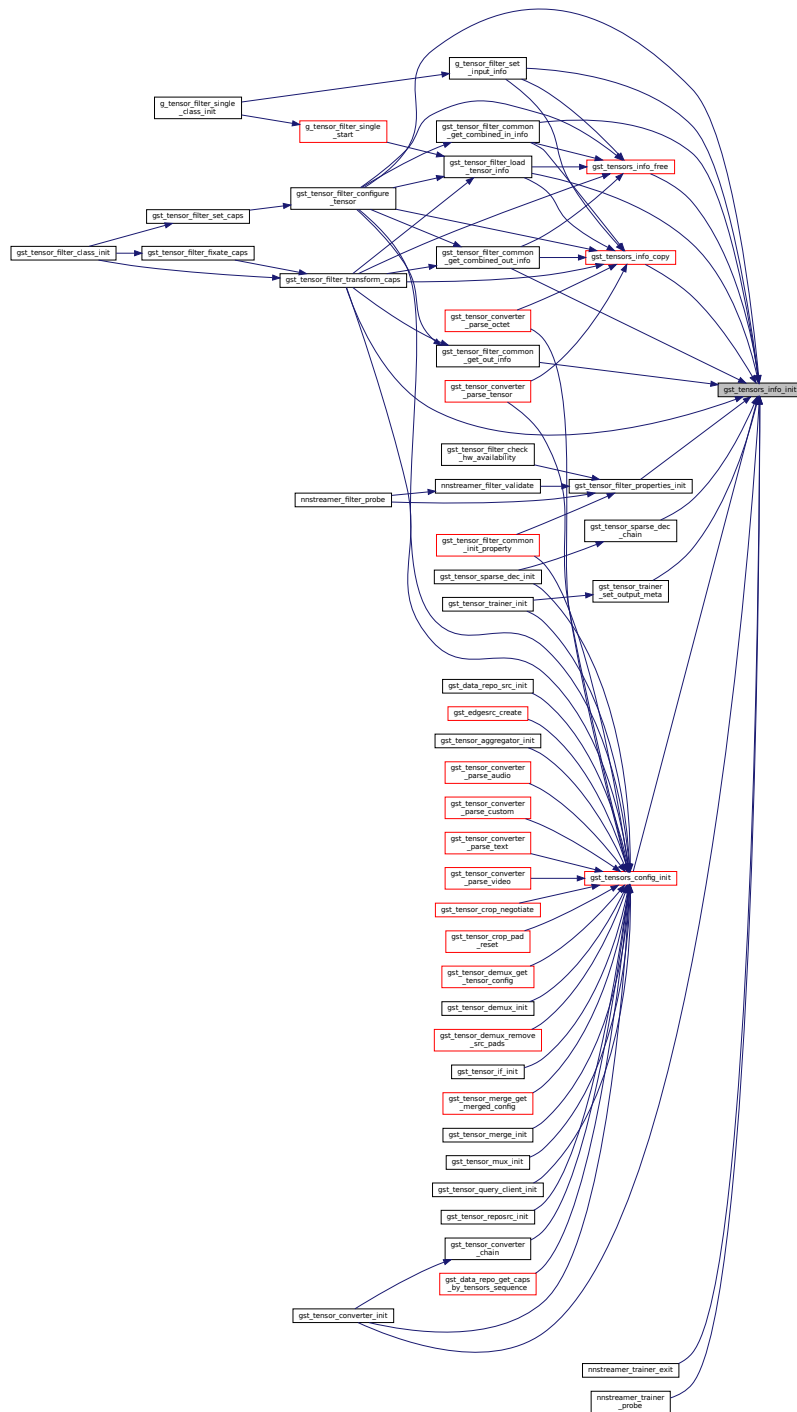
default format is static

Definition at line 325 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.50 gst_tensors_info_is_equal()

```
gboolean gst_tensors_info_is_equal (
    const GstTensorsInfo * i1,
    const GstTensorsInfo * i2 )
```


Parameters

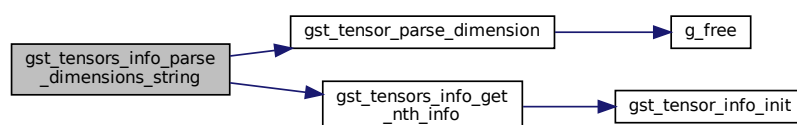
<i>info</i>	tensors info structure
<i>dim_string</i>	string of dimensions

Returns

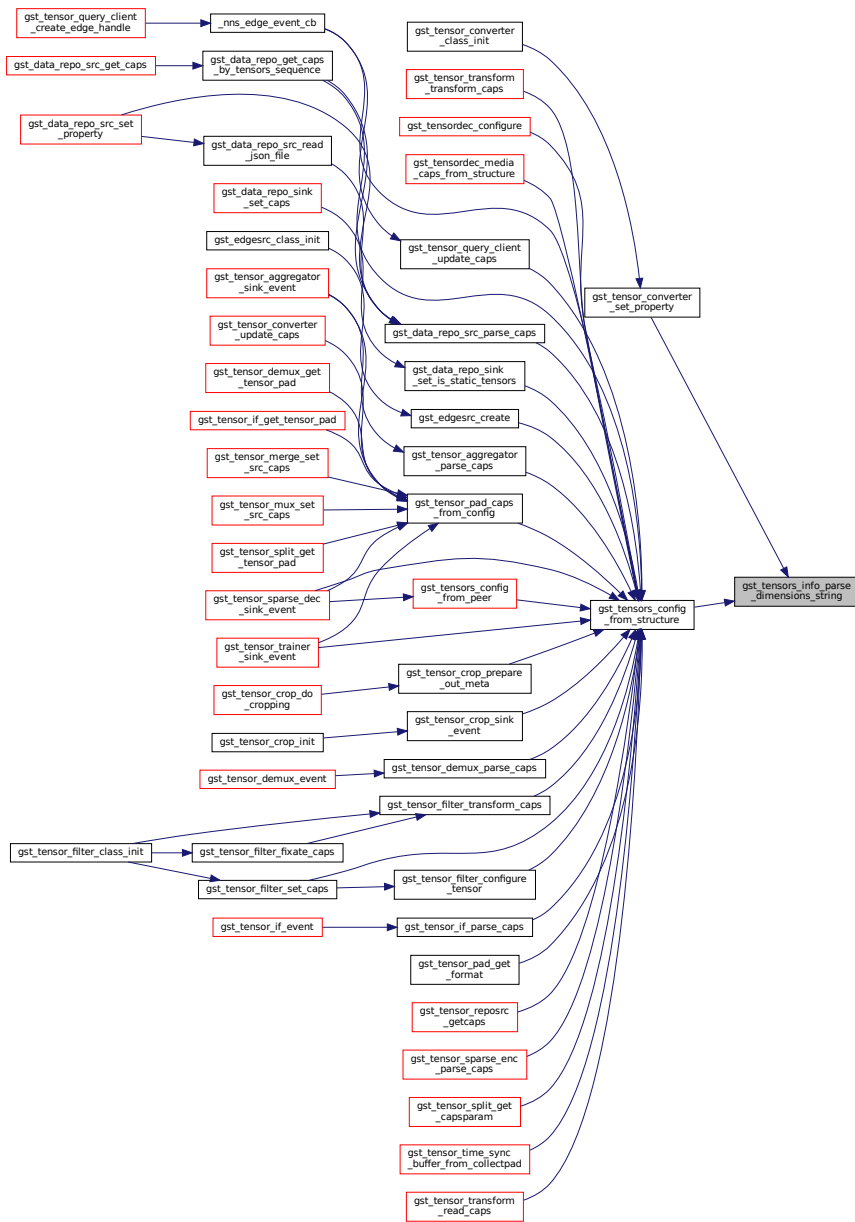
number of parsed dimensions

Definition at line 532 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.52 `gst_tensors_info_parse_names_string()`

```
guint gst_tensors_info_parse_names_string (
    GstTensorsInfo * info,
    const gchar * name_string )
```

Parse the string of names.

Parameters

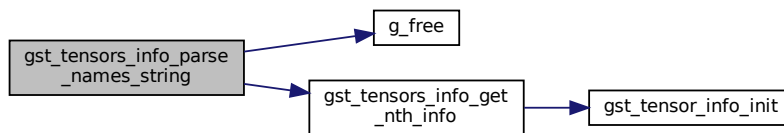
<i>info</i>	tensors info structure
<i>name_string</i>	string of names

Returns

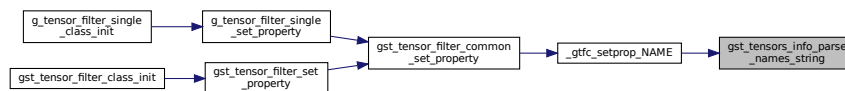
number of parsed names

Definition at line 612 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.53 gst_tensors_info_parse_types_string()

```

guint gst_tensors_info_parse_types_string (
    GstTensorsInfo * info,
    const gchar * type_string )
  
```

Parse the string of types.

Parameters

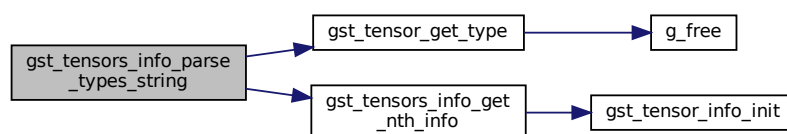
<i>info</i>	tensors info structure
<i>type_string</i>	string of types

Returns

number of parsed types

Definition at line 572 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Parameters

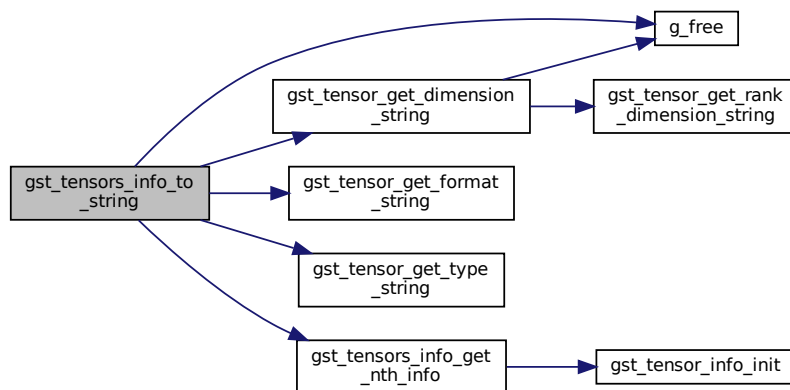
<i>info</i>	GstTensorsInfo structure
-------------	--

Returns

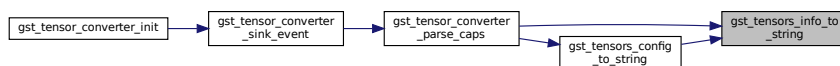
The newly allocated string representing the tensors info. Free after use.

Definition at line 783 of file nntstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.55 gst_tensors_info_validate()

```

gboolean gst_tensors_info_validate (
    const GstTensorsInfo * info )
  
```

Check the tensors info is valid.

Parameters

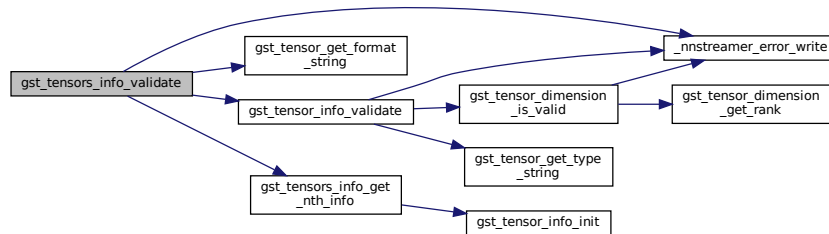
<i>info</i>	tensors info structure
-------------	------------------------

Returns

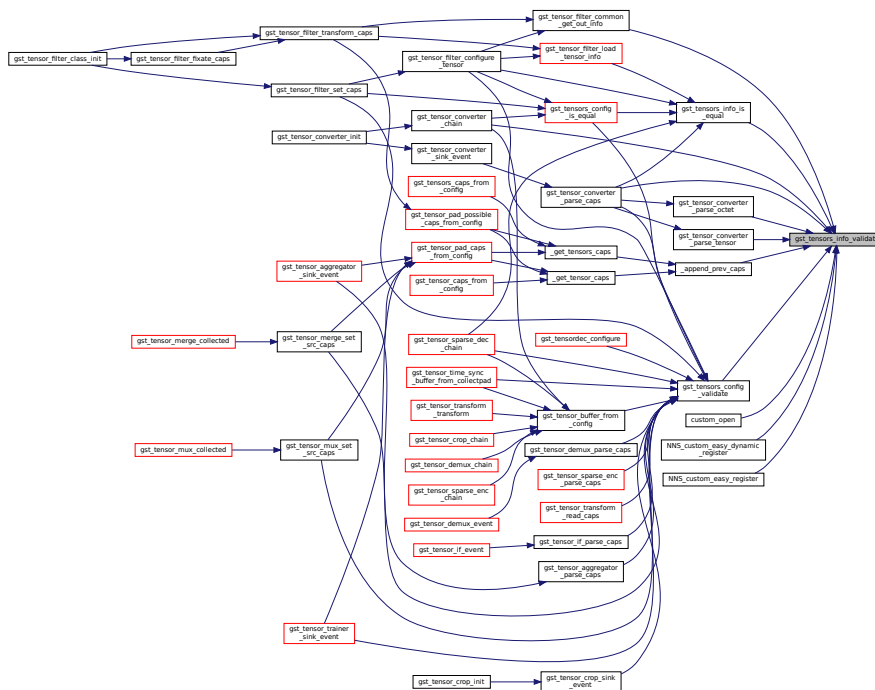
TRUE if info is valid

Definition at line 404 of file nnstreamer_plugin_api_util_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.94.3.56 nnstreamer_version_fetch()

```

void nnstreamer_version_fetch (
    guint * major,
    guint * minor,
    guint * micro )
  
```

Get the version of NNStreamer (int, divided).

Parameters

out	<i>major</i>	MAJOR.minor.micro, won't set if it's null.
out	<i>minor</i>	major.MINOR.micro, won't set if it's null.
out	<i>micro</i>	major.minor.MICRO, won't set if it's null.

Definition at line 1624 of file nnstreamer_plugin_api_util_impl.c.

9.94.3.57 nnstreamer_version_string()

```
gchar* nnstreamer_version_string (
    void )
```

Get the version of NNStreamer (string).

Get the version of NNStreamer.

Returns

Newly allocated string. The returned string should be freed with [g_free\(\)](#).

Definition at line 1609 of file nnstreamer_plugin_api_util_impl.c.

9.94.4 Variable Documentation**9.94.4.1 tensor_element_size**

```
const quint tensor_element_size[] [static]
```

Initial value:

```
= {
    [_NNS_INT32] = 4,
    [_NNS_UINT32] = 4,
    [_NNS_INT16] = 2,
    [_NNS_UINT16] = 2,
    [_NNS_INT8] = 1,
    [_NNS_UINT8] = 1,
    [_NNS_FLOAT64] = 8,
    [_NNS_FLOAT32] = 4,
    [_NNS_INT64] = 8,
    [_NNS_UINT64] = 8,
    [_NNS_FLOAT16] = 2,
    [_NNS_END] = 0,
}
```

Byte-per-element of each tensor element type.

Definition at line 38 of file nnstreamer_plugin_api_util_impl.c.

9.94.4.2 tensor_element_typename

```
const gchar* tensor_element_typename[] [static]
```

Initial value:

```
= {
  [_NNS_INT32] = "int32",
  [_NNS_UINT32] = "uint32",
  [_NNS_INT16] = "int16",
  [_NNS_UINT16] = "uint16",
  [_NNS_INT8] = "int8",
  [_NNS_UINT8] = "uint8",
  [_NNS_FLOAT64] = "float64",
  [_NNS_FLOAT32] = "float32",
  [_NNS_INT64] = "int64",
  [_NNS_UINT64] = "uint64",
  [_NNS_FLOAT16] = "float16",
  [_NNS_END] = NULL,
}
```

String representations for each tensor element type.

Definition at line 20 of file nnstreamer_plugin_api_util_impl.c.

9.94.4.3 tensor_format_name

```
const gchar* tensor_format_name[] [static]
```

Initial value:

```
= {
  [_NNS_TENSOR_FORMAT_STATIC] = "static",
  [_NNS_TENSOR_FORMAT_FLEXIBLE] = "flexible",
  [_NNS_TENSOR_FORMAT_SPARSE] = "sparse",
  [_NNS_TENSOR_FORMAT_END] = NULL
}
```

String representations for tensor format.

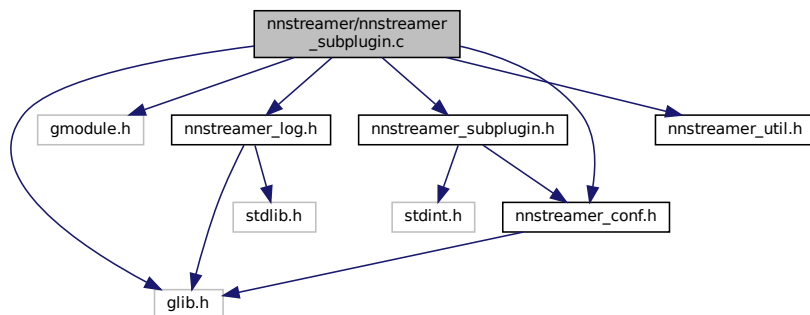
Definition at line 56 of file nnstreamer_plugin_api_util_impl.c.

9.95 nnstreamer/nnstreamer_subplugin.c File Reference

Subplugin Manager for NNStreamer.

```
#include <glib.h>
#include <gmodule.h>
#include "nnstreamer_log.h"
#include "nnstreamer_subplugin.h"
#include "nnstreamer_conf.h"
#include <nnstreamer_util.h>
```

Include dependency graph for nnstreamer_subplugin.c:



Enumerations

- enum `subpluginSearchLogic` { `NNS_SEARCH_FILENAME`, `NNS_SEARCH_GETALL`, `NNS_SEARCH_NO_OP` }

Functions

- static void `init_subplugin` (void)
Create handles at the start of library.
- `G_LOCK_DEFINE_STATIC` (splock)
Protects handles and subplugins.
- static void `_spdata_destroy` (gpointer _data)
Private function for g_hash_table data destructor, GDestroyNotify.
- static `subpluginData` * `_get_subplugin_data` (subpluginType type, const gchar *name)
Internal function to get sub-plugin data.
- static `subpluginData` * `_search_subplugin` (subpluginType type, const gchar *name, const gchar *path)
Internal function to scan sub-plugin.
- const void * `get_subplugin` (subpluginType type, const char *name)
Public function defined in the header.
- gchar ** `get_all_subplugins` (subpluginType type)
Public function defined in the header.
- gboolean `register_subplugin` (subpluginType type, const char *name, const void *data)
Public function defined in the header.
- gboolean `unregister_subplugin` (subpluginType type, const char *name)
Public function defined in the header.
- static void `_close_handle` (gpointer data)
dealloc function for handles
- void `subplugin_set_custom_property_desc` (subpluginType type, const char *name, const gchar *prop, va↵_list varargs)
common interface to set custom property description of a sub-plugin.
- GData * `subplugin_get_custom_property_desc` (subpluginType type, const char *name)
common interface to get custom property description of a sub-plugin.
- static void `fini_subplugin` (void)
Free handles at the start of library.

Variables

- static GPtrArray * `handles` = NULL
Array of dynamic loaded handles.
- `subpluginData`
- static GHashTable * `subplugins` [`NNS_SUBPLUGIN_END`] = { 0 }
- static `subpluginSearchLogic` `searchAlgorithm` []

9.95.1 Detailed Description

Subplugin Manager for NNStreamer.

NNStreamer Subplugin Manager Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

27 Nov 2018

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.95.2 Enumeration Type Documentation

9.95.2.1 subpluginSearchLogic

enum `subpluginSearchLogic`

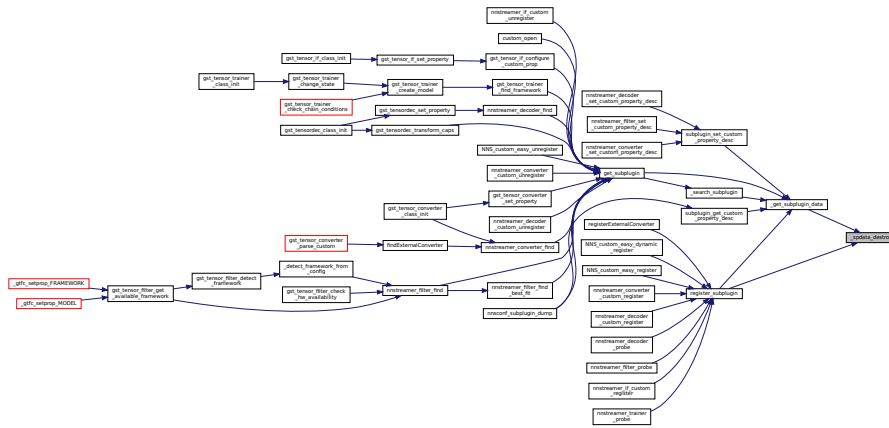
Enumerator

NNS_SEARCH_FILENAME	
NNS_SEARCH_GETALL	
NNS_SEARCH_NO_OP	

Definition at line 64 of file `nnstreamer_subplugin.c`.

9.95.3 Function Documentation

Here is the caller graph for this function:



9.95.3.5 fini_subplugin()

```
static void fini_subplugin (
    void ) [static]
```

Free handles at the start of library.

Internal error (init not called?)

Definition at line 385 of file nnstreamer_subplugin.c.

Here is the call graph for this function:



9.95.3.6 G_LOCK_DEFINE_STATIC()

```
G_LOCK_DEFINE_STATIC (
    splock )
```

Protects handles and subplugins.

9.95.3.7 get_all_subplugins()

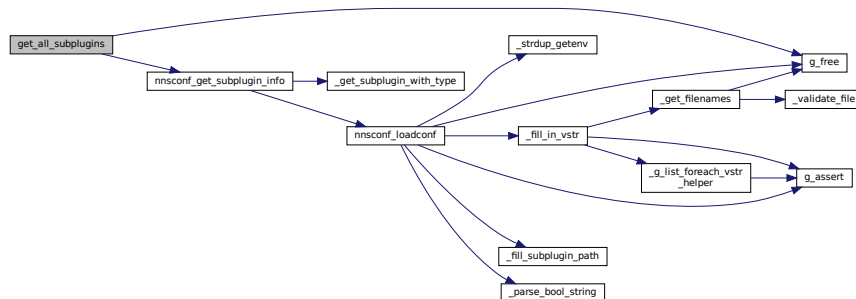
```
gchar** get_all_subplugins (
    subpluginType type )
```

Public function defined in the header.

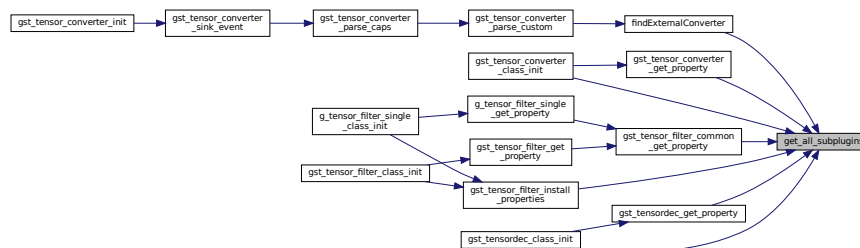
Get the list of registered subplugins.

Definition at line 176 of file nnstreamer_subplugin.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.95.3.8 get_subplugin()

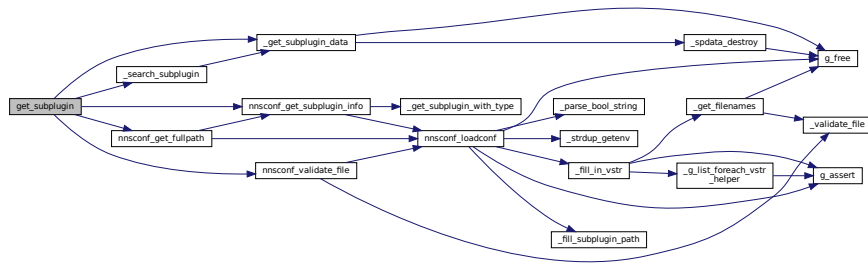
```
const void* get_subplugin (
    subpluginType type,
    const char * name )
```

Public function defined in the header.

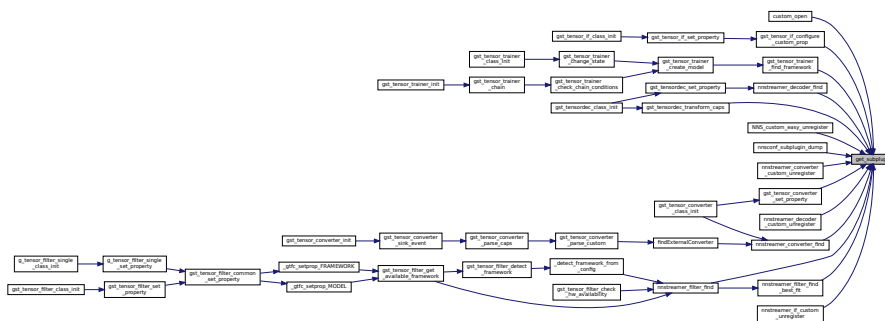
Retrieve the registered data with the subplugin name. Search and register if found with the conf

Definition at line 141 of file nnstreamer_subplugin.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.95.3.9 init_subplugin()

```
static void init_subplugin (
    void ) [static]
```

Create handles at the start of library.

- < The name of subplugin
- < subplugin specific data forwarded from the subplugin
- < [OPTIONAL] subplugin specific custom property desc list

Internal error (duplicated init call?)

Definition at line 37 of file nnstreamer_subplugin.c.

9.95.3.10 register_subplugin()

```
gboolean register_subplugin (  
    subpluginType type,  
    const char * name,  
    const void * data )
```

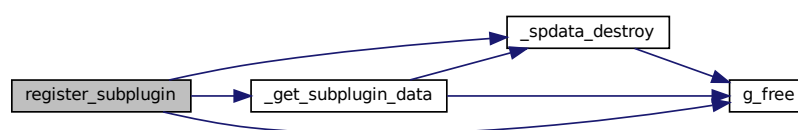
Public function defined in the header.

Register the subplugin. If duplicated name exists, it is rejected.

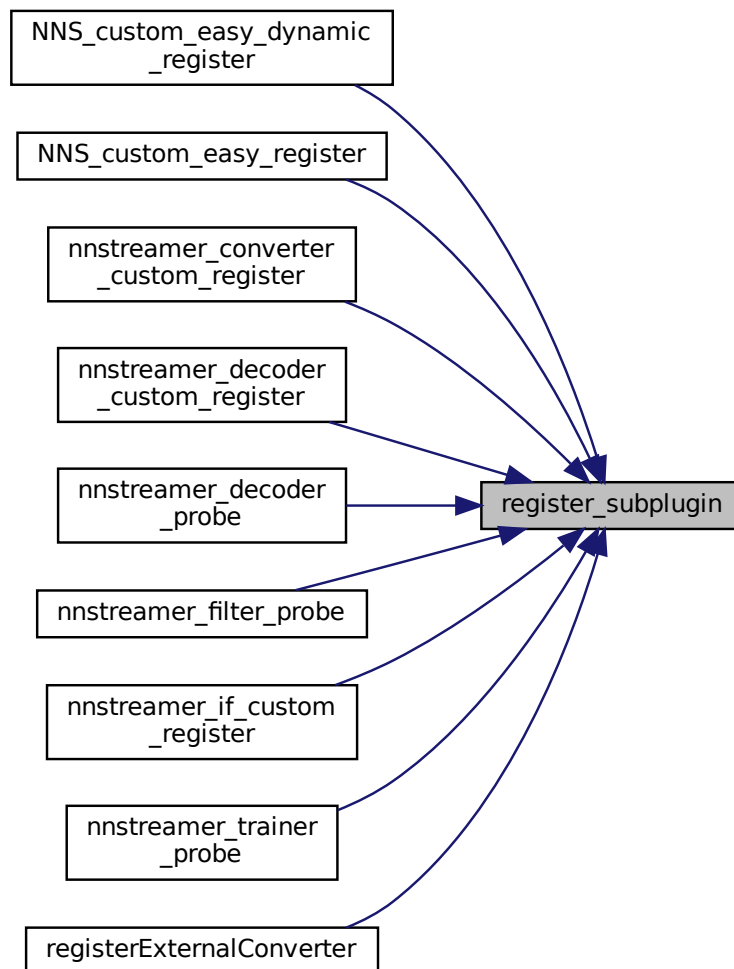
Todo data out of scope at add

Definition at line 225 of file nntstreamer_subplugin.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.95.3.11 subplugin_get_custom_property_desc()

```

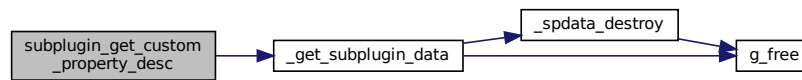
GData* subplugin_get_custom_property_desc (
    subpluginType type,
    const char * name )

```

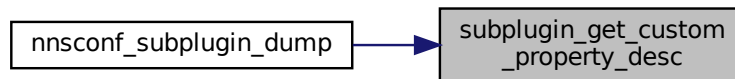
common interface to get custom property description of a sub-plugin.

Definition at line 359 of file `nnstreamer_subplugin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.95.3.12 subplugin_set_custom_property_desc()

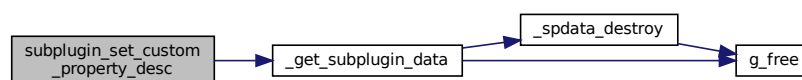
```

void subplugin_set_custom_property_desc (
    subpluginType type,
    const char * name,
    const gchar * prop,
    va_list varargs )
  
```

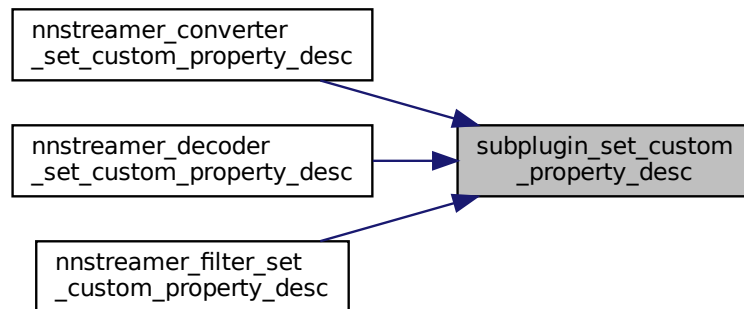
common interface to set custom property description of a sub-plugin.

Definition at line 329 of file `nnsreamer_subplugin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.95.3.13 unregister_subplugin()

```

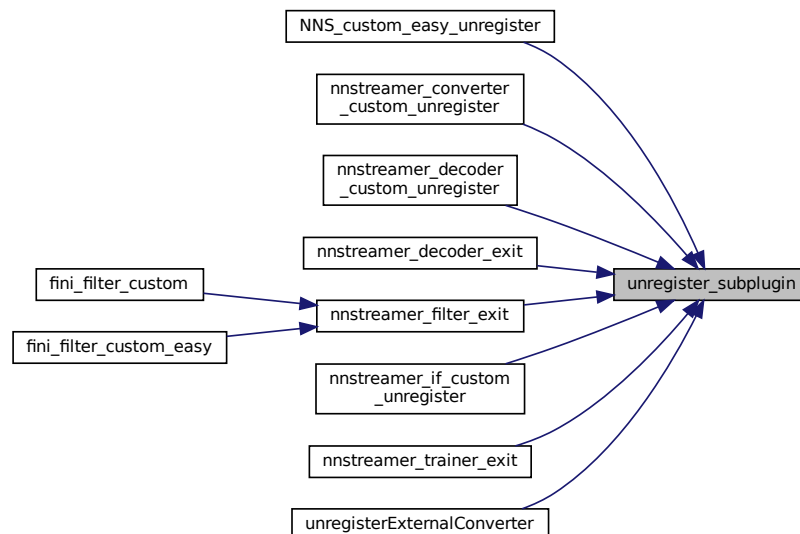
gboolean unregister_subplugin (
    subpluginType type,
    const char * name )
  
```

Public function defined in the header.

Unregister the subplugin.

Definition at line 289 of file `nnstreamer_subplugin.c`.

Here is the caller graph for this function:



9.95.4 Variable Documentation

9.95.4.1 handles

```
GPtArray* handles = NULL [static]
```

Array of dynamic loaded handles.

Definition at line 35 of file nnstreamer_subplugin.c.

9.95.4.2 searchAlgorithm

```
subpluginSearchLogic searchAlgorithm[] [static]
```

Initial value:

```
= {  
  [NNS_SUBPLUGIN_FILTER] = NNS_SEARCH_FILENAME,  
  [NNS_SUBPLUGIN_DECODER] = NNS_SEARCH_FILENAME,  
  [NNS_EASY_CUSTOM_FILTER] = NNS_SEARCH_FILENAME,  
  [NNS_SUBPLUGIN_CONVERTER] = NNS_SEARCH_GETALL,  
  [NNS_SUBPLUGIN_TRAINER] = NNS_SEARCH_FILENAME,  
  [NNS_CUSTOM_CONVERTER] = NNS_SEARCH_NO_OP,  
  [NNS_CUSTOM_DECODER] = NNS_SEARCH_NO_OP,  
  [NNS_IF_CUSTOM] = NNS_SEARCH_NO_OP,  
  [NNS_SUBPLUGIN_END] = NNS_SEARCH_NO_OP,  
}
```

Definition at line 71 of file nnstreamer_subplugin.c.

9.95.4.3 subpluginData

```
subpluginData
```

Definition at line 45 of file nnstreamer_subplugin.c.

9.95.4.4 subplugins

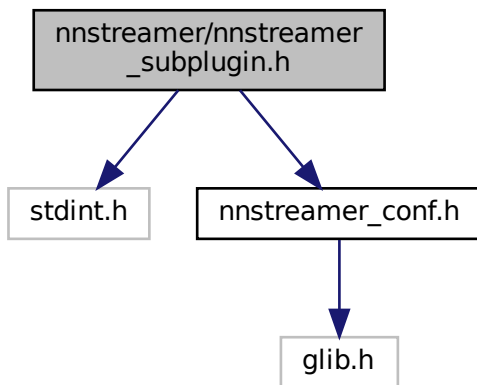
```
GHashTable* subplugins[NNS_SUBPLUGIN_END] = { 0 } [static]
```

Definition at line 47 of file nnstreamer_subplugin.c.

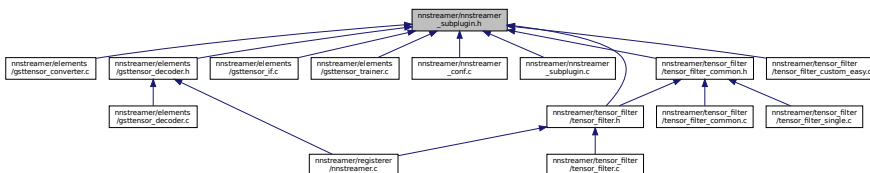
9.96 nnstreamer/nnstreamer_subplugin.h File Reference

Subplugin Manager for NNStreamer.

```
#include <stdint.h>
#include "nnstreamer_conf.h"
Include dependency graph for nnstreamer_subplugin.h:
```



This graph shows which files directly or indirectly include this file:



Macros

- #define `NNS_SUBPLUGIN_CHECKER` (0xdeadbeef)

Enumerations

- enum `subpluginType` {
 - `NNS_SUBPLUGIN_FILTER` = `NNSCONF_PATH_FILTERS`, `NNS_SUBPLUGIN_DECODER` = `NNSCONF_PATH_DECODERS`, `NNS_EASY_CUSTOM_FILTER` = `NNSCONF_PATH_EASY_CUSTOM_FILTERS`, `NNS_SUBPLUGIN_CONVERTER` = `NNSCONF_PATH_CONVERTERS`,
 - `NNS_SUBPLUGIN_TRAINER` = `NNSCONF_PATH_TRAINERS`, `NNS_CUSTOM_CONVERTER`, `NNS_CUSTOM_DECODER`,
 - `NNS_IF_CUSTOM`,
 - `NNS_SUBPLUGIN_END` }

Functions

- const void * [get_subplugin](#) (subpluginType type, const char *name)
Retrieve the registered data with the subplugin name.
- gchar ** [get_all_subplugins](#) (subpluginType type)
Get the list of registered subplugins.
- gboolean [register_subplugin](#) (subpluginType type, const char *name, const void *data)
Register the subplugin. If duplicated name exists, it is rejected.
- gboolean [unregister_subplugin](#) (subpluginType type, const char *name)
Unregister the subplugin.
- void [subplugin_set_custom_property_desc](#) (subpluginType type, const char *name, const gchar *prop, va↔
_list varargs)
common interface to set custom property description of a sub-plugin.
- GData * [subplugin_get_custom_property_desc](#) (subpluginType type, const char *name)
common interface to get custom property description of a sub-plugin.

9.96.1 Detailed Description

Subplugin Manager for NNStreamer.

NNStreamer Subplugin Manager Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

27 Nov 2018

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

To Packagers:

This file is to be packaged as "devel" package for NN developers. (subplugin writers)

Note

Any independent subplugin (existing as an independent .so) should call register_subplugin () (or its wrapper) with the subplugin's constructor function.

9.96.2 Macro Definition Documentation

9.96.2.1 NNS_SUBPLUGIN_CHECKER

```
#define NNS_SUBPLUGIN_CHECKER (0xdeadbeef)
```

Definition at line 53 of file nntstreamer_subplugin.h.

9.96.3 Enumeration Type Documentation

9.96.3.1 subpluginType

```
enum subpluginType
```

Enumerator

NNS_SUBPLUGIN_FILTER	
NNS_SUBPLUGIN_DECODER	
NNS_EASY_CUSTOM_FILTER	
NNS_SUBPLUGIN_CONVERTER	
NNS_SUBPLUGIN_TRAINER	
NNS_CUSTOM_CONVERTER	
NNS_CUSTOM_DECODER	
NNS_IF_CUSTOM	
NNS_SUBPLUGIN_END	

Definition at line 40 of file nntstreamer_subplugin.h.

9.96.4 Function Documentation

9.96.4.1 get_all_subplugins()

```
gchar** get_all_subplugins (  
    subpluginType type )
```

Get the list of registered subplugins.

Parameters

in	<i>type</i>	Subplugin Type
----	-------------	----------------

Returns

The list of subplugin name

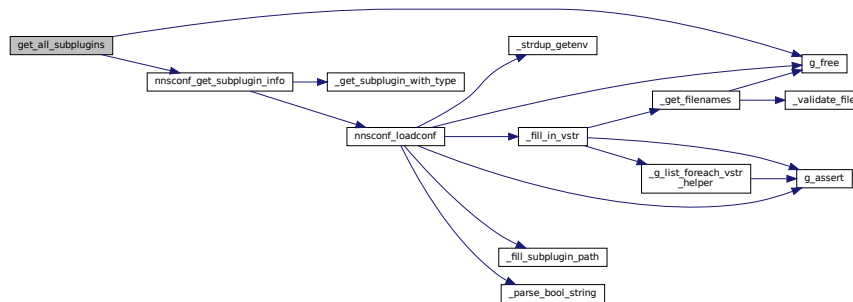
Note

Caller should free the returned value using `g_strfreev()`

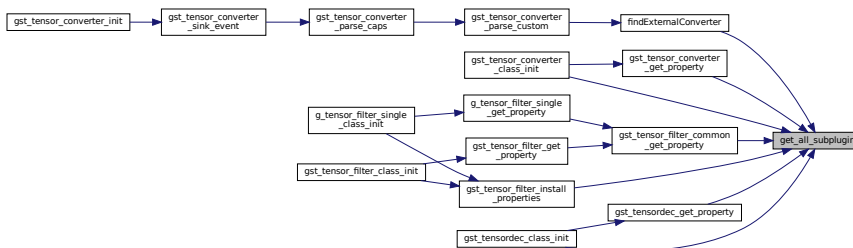
Get the list of registered subplugins.

Definition at line 176 of file `nnstreamer_subplugin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.96.4.2 get_subplugin()

```

const void* get_subplugin (
    subpluginType type,
    const char * name )
  
```

Retrieve the registered data with the subplugin name.

Parameters

in	<i>type</i>	Subplugin Type
in	<i>name</i>	Subplugin Name. The filename should be libnntstreamer_\${type}_\${name}.so

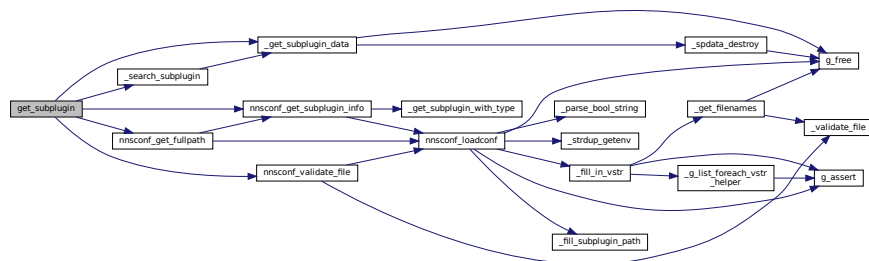
Returns

The registered data

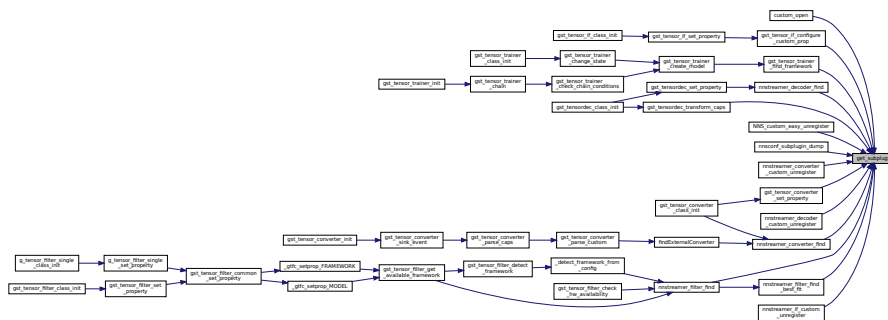
Retrieve the registered data with the subplugin name. Search and register if found with the conf

Definition at line 141 of file nntstreamer_subplugin.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.96.4.3 register_subplugin()

```
gboolean register_subplugin (
    subpluginType type,
    const char * name,
    const void * data )
```

Register the subplugin. If duplicated name exists, it is rejected.

Parameters

in	<i>type</i>	Subplugin Type
in	<i>name</i>	Subplugin Name. The filename should be subplugin_prefixes[<i>type</i>]{ <i>name</i> }.so
in	<i>data</i>	The registered data

Returns

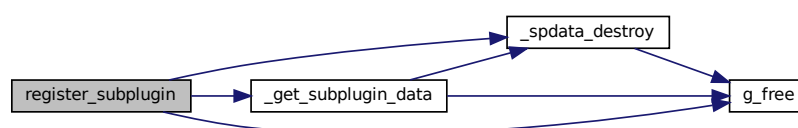
TRUE if registered as new. FALSE if duplicated (overwritten/updated).

Register the subplugin. If duplicated name exists, it is rejected.

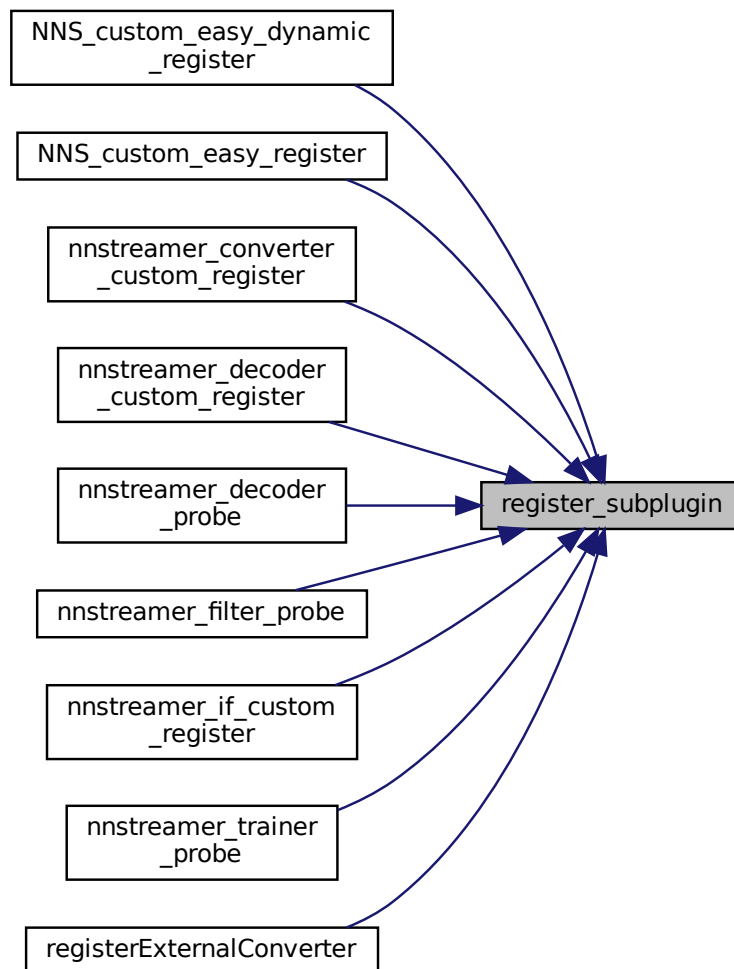
Todo data out of scope at add

Definition at line 225 of file nstreamer_subplugin.c.

Here is the call graph for this function:



Here is the caller graph for this function:



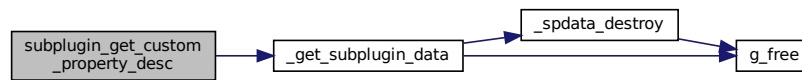
9.96.4.4 subplugin_get_custom_property_desc()

```
GData* subplugin_get_custom_property_desc (  
    subpluginType type,  
    const char * name )
```

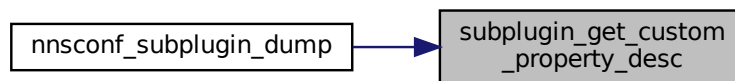
common interface to get custom property description of a sub-plugin.

Definition at line 359 of file `nnstreamer_subplugin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.96.4.5 subplugin_set_custom_property_desc()

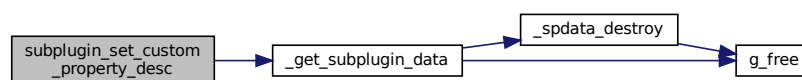
```

void subplugin_set_custom_property_desc (
    subpluginType type,
    const char * name,
    const gchar * prop,
    va_list varargs )
  
```

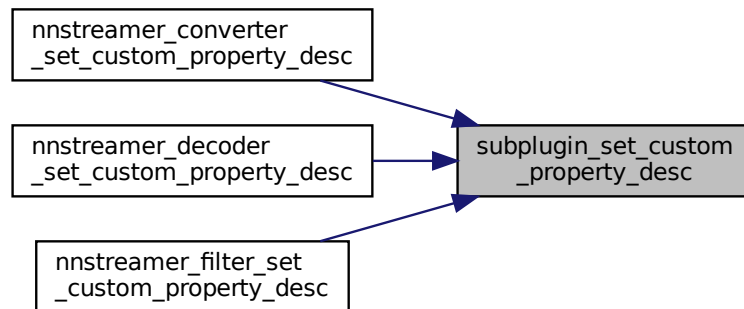
common interface to set custom property description of a sub-plugin.

Definition at line 329 of file `nnsreamer_subplugin.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.96.4.6 unregister_subplugin()

```

gboolean unregister_subplugin (
    subpluginType type,
    const char * name )
  
```

Unregister the subplugin.

Parameters

in	<i>type</i>	Subplugin type
in	<i>name</i>	Subplugin Name. The filename should be <code>subplugin_prefixes[type]\${name}.so</code>

Returns

TRUE if unregistered. FALSE if rejected or error.

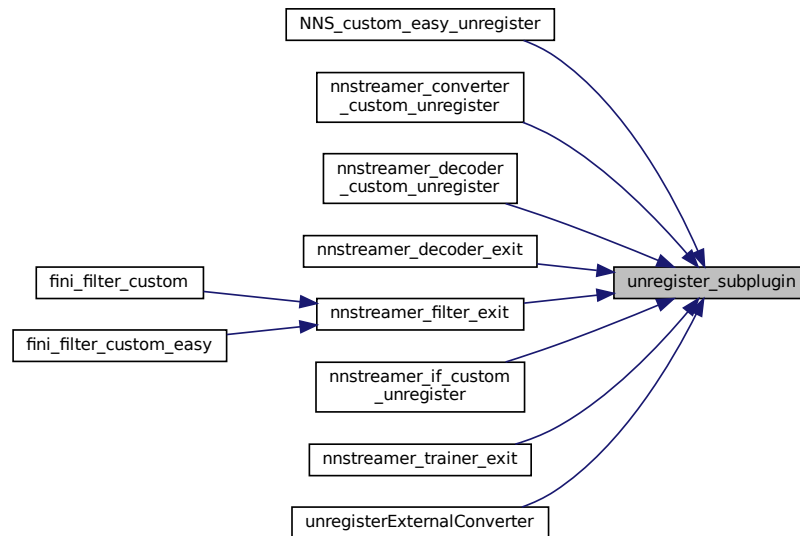
Warning

Subplugins checked out with `get_subplugins` can still be used after `unregister`.

Unregister the subplugin.

Definition at line 289 of file `nnstreamer_subplugin.c`.

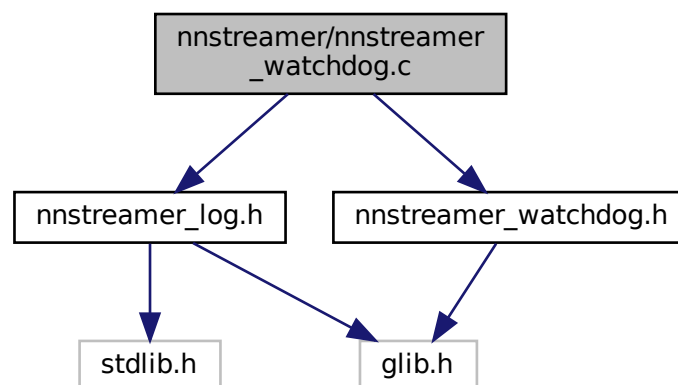
Here is the caller graph for this function:



9.97 nnstreamer/nnstreamer_watchdog.c File Reference

NNStreamer watchdog to manage the schedule in the element.

```
#include <nnstreamer_log.h>
#include "nnstreamer_watchdog.h"
Include dependency graph for nnstreamer_watchdog.c:
```



Classes

- [struct _NnstWatchdog](#)

Structure for NNStreamer watchdog.

Typedefs

- typedef struct [_NnstWatchdog](#) [NnstWatchdog](#)
Structure for NNStreamer watchdog.

Functions

- static gboolean [_loop_running_cb](#) ([NnstWatchdog](#) *watchdog)
Called when loop is running.
- static gpointer [_nntstreamer_watchdog_thread](#) (gpointer ptr)
Watchdog thread.
- gboolean [nntstreamer_watchdog_create](#) ([nns_watchdog_h](#) *watchdog_h)
Create nntstreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())
- void [nntstreamer_watchdog_destroy](#) ([nns_watchdog_h](#) watchdog_h)
Destroy watchdog source. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())
- void [nntstreamer_watchdog_release](#) ([nns_watchdog_h](#) watchdog_h)
Release watchdog source. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())
- gboolean [nntstreamer_watchdog_feed](#) ([nns_watchdog_h](#) watchdog_h, GSourceFunc func, guint interval, void *user_data)
Set watchdog timeout. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

9.97.1 Detailed Description

NNStreamer watchdog to manage the schedule in the element.

NNStreamer watchdog Copyright (C) 2024 Gichan Jang gichan2.jang@samsung.com

Date

30 Oct 2024

See also

<https://github.com/nntstreamer/nntstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.97.2 Typedef Documentation

9.97.2.1 NnstWatchdog

```
typedef struct _NnstWatchdog NnstWatchdog
```

Structure for NNStreamer watchdog.

9.97.3 Function Documentation

9.97.3.1 _loop_running_cb()

```
static gboolean _loop_running_cb (
    NnstWatchdog * watchdog ) [static]
```

Called when loop is running.

Definition at line 35 of file nnstreamer_watchdog.c.

Here is the caller graph for this function:



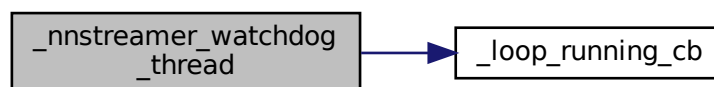
9.97.3.2 _nnstreamer_watchdog_thread()

```
static gpointer _nnstreamer_watchdog_thread (
    gpointer ptr ) [static]
```

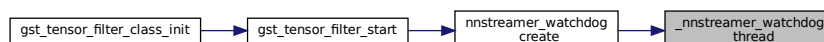
Watchdog thread.

Definition at line 48 of file nnstreamer_watchdog.c.

Here is the call graph for this function:



Here is the caller graph for this function:



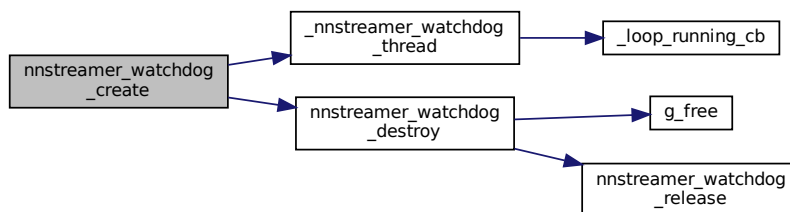
9.97.3.3 nnstreamer_watchdog_create()

```
gboolean nnstreamer_watchdog_create (
    nns_watchdog_h * watchdog_h )
```

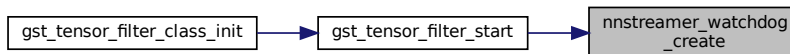
Create nnstreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

Definition at line 72 of file nnstreamer_watchdog.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.97.3.4 nnstreamer_watchdog_destroy()

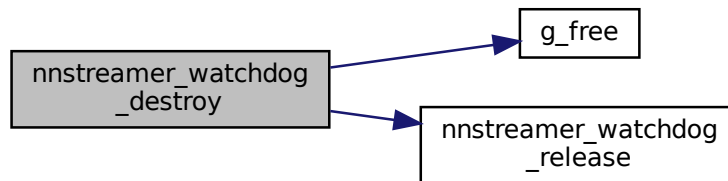
```
void nnstreamer_watchdog_destroy (
    nns_watchdog_h watchdog_h )
```

Destroy watchdog source. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

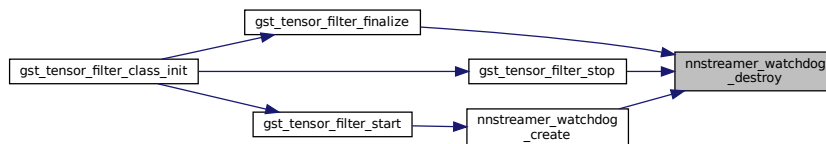
Destroy nnstreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

Definition at line 133 of file nnstreamer_watchdog.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.97.3.5 nnstreamer_watchdog_feed()

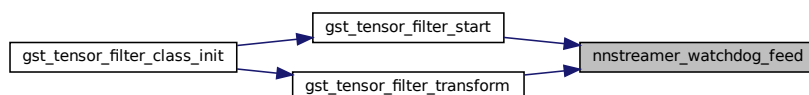
```

gboolean nnstreamer_watchdog_feed (
    nns_watchdog_h watchdog_h,
    GSourceFunc func,
    guint interval,
    void * user_data )
  
```

Set watchdog timeout. Recommended using watchdog handle with proper lock (e.g., `GST_OBJECT_LOCK()`)

Definition at line 171 of file `nnstreamer_watchdog.c`.

Here is the caller graph for this function:



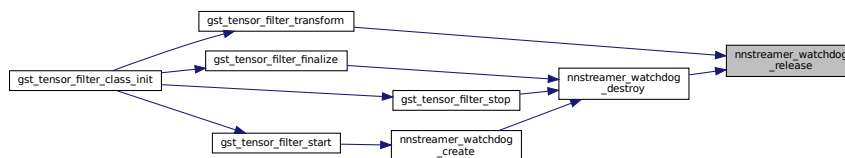
9.97.3.6 nnstreamer_watchdog_release()

```
void nnstreamer_watchdog_release (
    nns_watchdog_h watchdog_h )
```

Release watchdog source. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

Definition at line 157 of file nnstreamer_watchdog.c.

Here is the caller graph for this function:

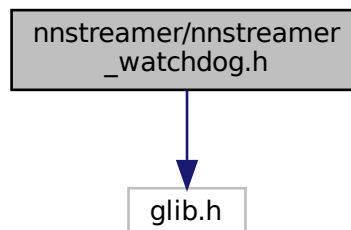


9.98 nnstreamer/nnstreamer_watchdog.h File Reference

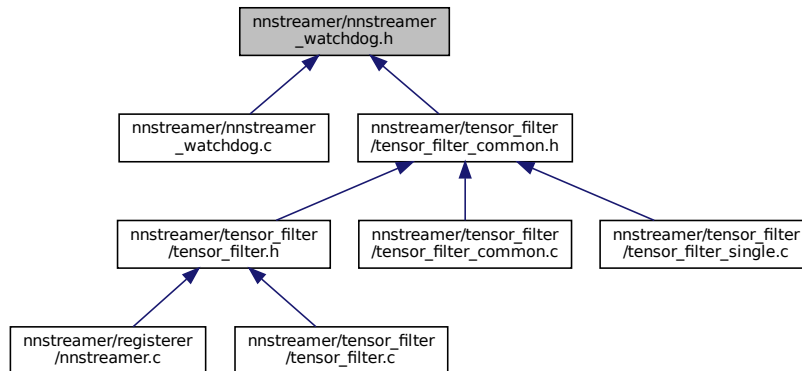
NNStreamer watchdog header to manage the schedule in the element.

```
#include <glib.h>
```

Include dependency graph for nnstreamer_watchdog.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef void * [nns_watchdog_h](#)

Functions

- gboolean [nnstreamer_watchdog_create](#) ([nns_watchdog_h](#) *watchdog_h)
Create nnstreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())
- void [nnstreamer_watchdog_destroy](#) ([nns_watchdog_h](#) watchdog_h)
Destroy nnstreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())
- void [nnstreamer_watchdog_release](#) ([nns_watchdog_h](#) watchdog_h)
Release watchdog source. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())
- gboolean [nnstreamer_watchdog_feed](#) ([nns_watchdog_h](#) watchdog_h, GSourceFunc func, guint interval, void *user_data)
Set watchdog timeout. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

9.98.1 Detailed Description

NNStreamer watchdog header to manage the schedule in the element.

NNStreamer watchdog header Copyright (C) 2024 Gichan Jang gichan2.jang@samsung.com

Date

30 Oct 2024

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.98.2 Typedef Documentation

9.98.2.1 nns_watchdog_h

```
typedef void* nns_watchdog_h
```

Definition at line 25 of file nnsreamer_watchdog.h.

9.98.3 Function Documentation

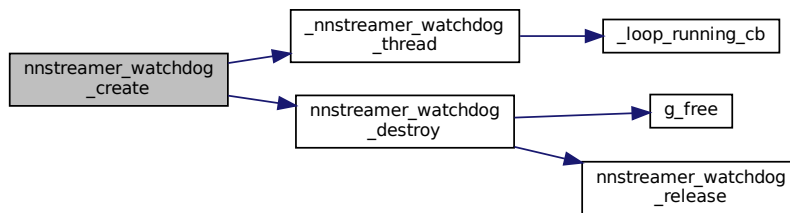
9.98.3.1 nnsreamer_watchdog_create()

```
gboolean nnsreamer_watchdog_create (
    nns_watchdog_h * watchdog_h )
```

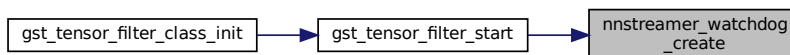
Create nnsreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

Definition at line 72 of file nnsreamer_watchdog.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.98.3.2 nnstreamer_watchdog_destroy()

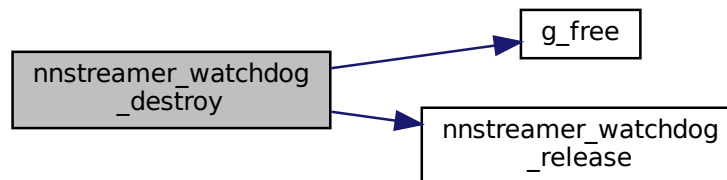
```
void nnstreamer_watchdog_destroy (
    nns_watchdog_h watchdog_h )
```

Destroy nnstreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

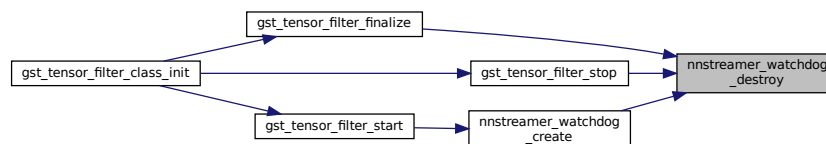
Destroy nnstreamer watchdog. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

Definition at line 133 of file nnstreamer_watchdog.c.

Here is the call graph for this function:



Here is the caller graph for this function:



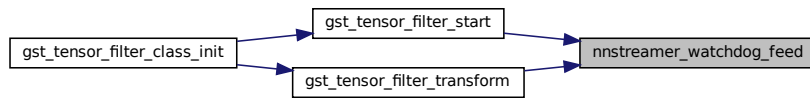
9.98.3.3 nnstreamer_watchdog_feed()

```
gboolean nnstreamer_watchdog_feed (
    nns_watchdog_h watchdog_h,
    GSourceFunc func,
    guint interval,
    void * user_data )
```

Set watchdog timeout. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

Definition at line 171 of file nnstreamer_watchdog.c.

Here is the caller graph for this function:



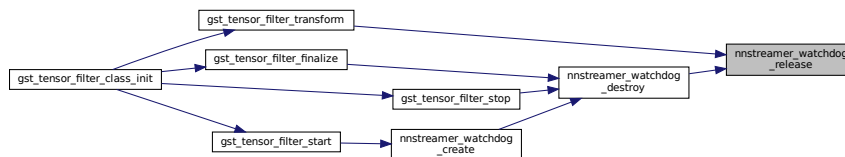
9.98.3.4 nnstreamer_watchdog_release()

```
void nnstreamer_watchdog_release (
    nns_watchdog_h watchdog_h )
```

Release watchdog source. Recommended using watchdog handle with proper lock (e.g., GST_OBJECT_LOCK())

Definition at line 157 of file nnstreamer_watchdog.c.

Here is the caller graph for this function:

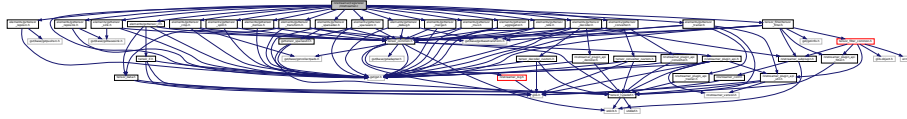


9.99 nnstreamer/registerer/nnstreamer.c File Reference

Registers all nnstreamer plugins for gstreamer so that we can have a single big binary.

```
#include <gst/gst.h>
#include <elements/gsttensor_aggregator.h>
#include <elements/gsttensor_converter.h>
#include <elements/gsttensor_crop.h>
#include <elements/gsttensor_debug.h>
#include <elements/gsttensor_decoder.h>
#include <elements/gsttensor_demux.h>
#include <elements/gsttensor_if.h>
#include <elements/gsttensor_merge.h>
#include <elements/gsttensor_mux.h>
#include <elements/gsttensor_rate.h>
#include <elements/gsttensor_reposink.h>
#include <elements/gsttensor_reposrc.h>
#include <elements/gsttensor_sink.h>
#include <elements/gsttensor_sparsedec.h>
#include <elements/gsttensor_sparseenc.h>
```

```
#include <elements/gsttensor_split.h>
#include <elements/gsttensor_transform.h>
#include <elements/gsttensor_trainer.h>
#include <tensor_filter/tensor_filter.h>
Include dependency graph for nnstreamer.c:
```



Macros

- #define `NNSTREAMER_INIT`(plugin, name, type)
- #define `PACKAGE` "nnstreamer"

Functions

- `GST_ERROR` ("Failed to register nnstreamer plugin : tensor_" # name)
 - type)) { \
- `while` (0)
 - Function to initialize all nnstreamer elements.
- `GST_PLUGIN_DEFINE` (GST_VERSION_MAJOR, GST_VERSION_MINOR, nnstreamer, "NNStreamer plugin library allows neural networks in GStreamer pipelines. Use nnstreamer-check utility for more information of the current NNStreamer installation.", gst_nnstreamer_init, VERSION, "LGPL", "nnstreamer", "https://github.com/nnstreamer/nnstreamer")

Variables

- return `FALSE`

9.99.1 Detailed Description

Registers all nnstreamer plugins for gstreamer so that we can have a single big binary.

Date

11 Oct 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.99.2 Macro Definition Documentation

9.99.2.1 NNSTREAMER_INIT

```
#define NNSTREAMER_INIT(
    plugin,
    name,
    type )
```

Value:

```
do { \
    if (!gst_element_register (plugin, "tensor_" # name, GST_RANK_NONE, GST_TYPE_TENSOR_
```

Definition at line 80 of file nnstreamer.c.

9.99.2.2 PACKAGE

```
#define PACKAGE "nnstreamer"
```

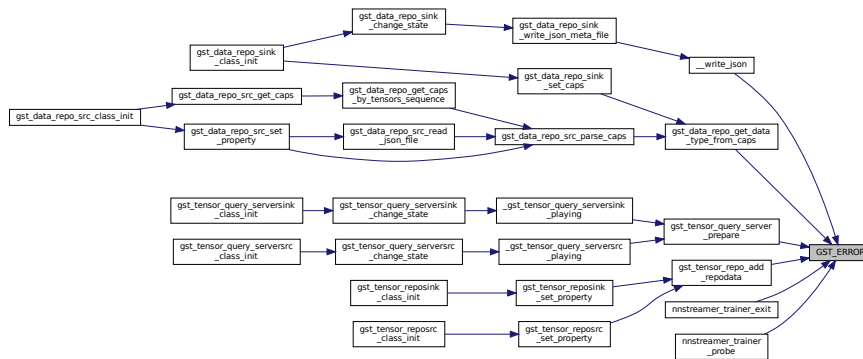
Definition at line 125 of file nnstreamer.c.

9.99.3 Function Documentation

9.99.3.1 GST_ERROR()

```
GST_ERROR (
    "Failed to register nnstreamer plugin : tensor_" # name )
type)) { \
```

Here is the caller graph for this function:



9.99.3.2 GST_PLUGIN_DEFINE()

```
GST_PLUGIN_DEFINE (
    GST_VERSION_MAJOR ,
    GST_VERSION_MINOR ,
    nnstreamer ,
    "NNStreamer plugin library allows neural networks in GStreamer pipelines. Use
nnstreamer-check utility for more information of the current NNStreamer installation." ,
    gst_nnstreamer_init ,
    VERSION ,
    "LGPL" ,
    "nnstreamer" ,
    "https://github.com/nnstreamer/nnstreamer" )
```

9.99.3.3 while()

```
while (
    0 )
```

Function to initialize all nnstreamer elements.

Definition at line 86 of file nnstreamer.c.

9.99.4 Variable Documentation

9.99.4.1 FALSE

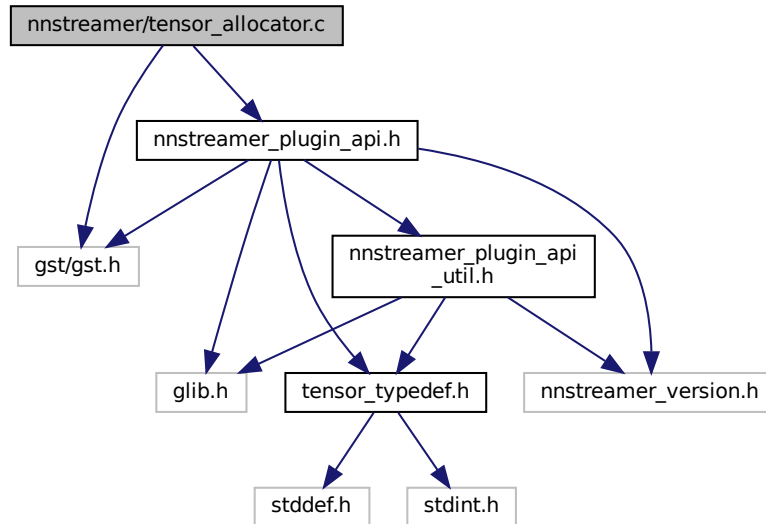
```
return FALSE
```

Definition at line 84 of file nnstreamer.c.

9.100 nnstreamer/tensor_allocator.c File Reference

Allocator for memory alignment.

```
#include <gst/gst.h>
#include "nnstreamer_plugin_api.h"
Include dependency graph for tensor_allocator.c:
```



Classes

- struct [GstTensorAllocator](#)
struct for type [GstTensorAllocator](#)
- struct [GstTensorAllocatorClass](#)
struct for class [GstTensorAllocatorClass](#)

Macros

- `#define` [GST_TENSOR_ALLOCATOR](#) "GstTensorAllocator"

Functions

- static GType [gst_tensor_allocator_get_type](#) (void)
- [G_DEFINE_TYPE](#) ([GstTensorAllocator](#), [gst_tensor_allocator](#), [GST_TYPE_ALLOCATOR](#))
- static GstMemory * [_alloc](#) (GstAllocator *allocator, gsize size, GstAllocationParams *params)
allocation wrapper that binds alignment parameter
- static void [gst_tensor_allocator_class_init](#) (GstTensorAllocatorClass *class)
class initialization for [GstTensorAllocatorClass](#)
- static void [gst_tensor_allocator_init](#) (GstTensorAllocator *allocator)
initialization for [GstTensorAllocator](#)
- void [gst_tensor_alloc_init](#) (gsize alignment)
set alignment that default allocator would align to

Variables

- static gsize [gst_tensor_allocator_alignment](#) = 0

9.100.1 Detailed Description

Allocator for memory alignment.

Copyright (C) 2021 Junhwan Kim jejudo.kim@samsung.com

Date

12 May 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.100.2 Macro Definition Documentation

9.100.2.1 GST_TENSOR_ALLOCATOR

```
#define GST_TENSOR_ALLOCATOR "GstTensorAllocator"
```

Definition at line 17 of file tensor_allocator.c.

9.100.3 Function Documentation

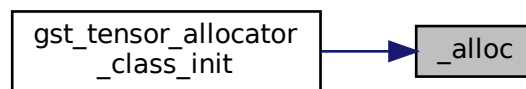
9.100.3.1 `_alloc()`

```
static GstMemory* _alloc (
    GstAllocator * allocator,
    gsize size,
    GstAllocationParams * params ) [static]
```

allocation wrapper that binds alignment parameter

Definition at line 45 of file tensor_allocator.c.

Here is the caller graph for this function:



9.100.3.2 `G_DEFINE_TYPE()`

```
G_DEFINE_TYPE (
    GstTensorAllocator ,
    gst_tensor_allocator ,
    GST_TYPE_ALLOCATOR )
```

9.100.3.3 `gst_tensor_alloc_init()`

```
void gst_tensor_alloc_init (
    gsize alignment )
```

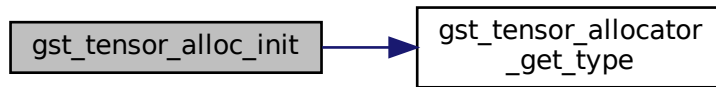
set alignment that default allocator would align to

Parameters

<i>alignment</i>	bytes of alignment
------------------	--------------------

Definition at line 109 of file tensor_allocator.c.

Here is the call graph for this function:



9.100.3.4 `gst_tensor_allocator_class_init()`

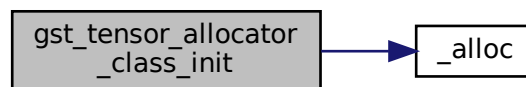
```

static void gst_tensor_allocator_class_init (
    GstTensorAllocatorClass * klass ) [static]
  
```

class initialization for [GstTensorAllocatorClass](#)

Definition at line 68 of file `tensor_allocator.c`.

Here is the call graph for this function:

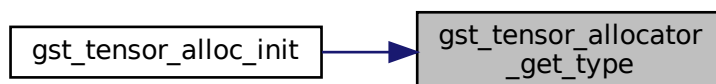


9.100.3.5 `gst_tensor_allocator_get_type()`

```

static GType gst_tensor_allocator_get_type (
    void ) [static]
  
```

Here is the caller graph for this function:



9.100.3.6 `gst_tensor_allocator_init()`

```
static void gst_tensor_allocator_init (
    GstTensorAllocator * allocator ) [static]
```

initialization for [GstTensorAllocator](#)

Definition at line 87 of file `tensor_allocator.c`.

9.100.4 Variable Documentation

9.100.4.1 `gst_tensor_allocator_alignment`

```
gsize gst_tensor_allocator_alignment = 0 [static]
```

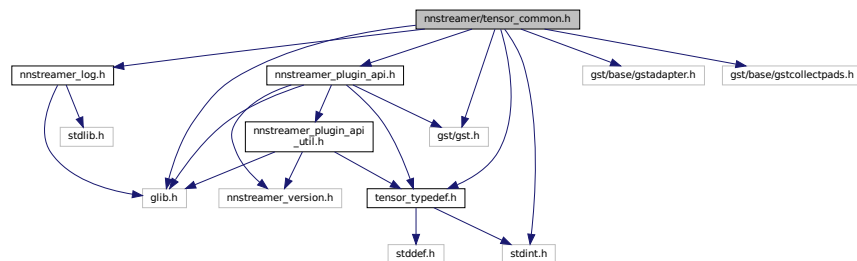
Definition at line 19 of file `tensor_allocator.c`.

9.101 nnstreamer/tensor_common.h File Reference

Common header file for NNStreamer, the GStreamer plugin for neural networks.

```
#include <glib.h>
#include <stdint.h>
#include <gst/gst.h>
#include <gst/base/gstadapter.h>
#include <gst/base/gstcollectpads.h>
#include "tensor_typedef.h"
#include "nnstreamer_log.h"
#include "nnstreamer_plugin_api.h"
```

Include dependency graph for `tensor_common.h`:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_tensor_sync_basepad_data](#)
Tensor Merge/Mux sync data for baspad mode.
- struct [_tensor_time_sync_data](#)
Tensor Merge/Mux time sync data.
- struct [GstTensorCollectPadData](#)
Internal data structure for Collect Pad in mux / merge.
- struct [GstTensorPad](#)
Internal data structure for pad in demux / split.

Macros

- #define [nns_memcpy](#) memcpy
- #define [nns_memset](#) memset
- #define [gst_tensor_pad_caps_is_static\(p\)](#) ([gst_tensor_pad_get_format](#) (p) == [_NNS_TENSOR_FORMAT_STATIC](#))
Macro to check current pad caps is static tensor.
- #define [gst_tensor_pad_caps_is_flexible\(p\)](#) ([gst_tensor_pad_get_format](#) (p) == [_NNS_TENSOR_FORMAT_FLEXIBLE](#))
Macro to check current pad caps is flexible tensor.
- #define [gst_tensor_pad_caps_is_sparse\(p\)](#) ([gst_tensor_pad_get_format](#) (p) == [_NNS_TENSOR_FORMAT_SPARSE](#))
Macro to check current pad caps is sparse tensor.
- #define [silent_debug](#)(self, ...)
Macro for debug message.
- #define [silent_debug_caps](#)(self, caps, msg)
Macro for capability debug message.

Typedefs

- typedef struct [_tensor_sync_basepad_data](#) [tensor_sync_basepad_data](#)
Tensor Merge/Mux sync data for baspad mode.
- typedef struct [_tensor_time_sync_data](#) [tensor_time_sync_data](#)
Tensor Merge/Mux time sync data.

Enumerations

- enum [tensor_time_sync_mode](#) {
[SYNC_NOSYNC](#) = 0, [SYNC_SLOWEST](#) = 1, [SYNC_BASEPAD](#) = 2, [SYNC_REFRESH](#) = 3,
[SYNC_END](#) }
- time synchronization options*

Functions

- void [gst_tensor_parse_config_file](#) (const gchar *config_path, const GObject *object)
Parses a configuration file and sets the corresponding properties on a GObject.
- [tensor_time_sync_mode](#) [gst_tensor_time_sync_get_mode](#) (const gchar *str)
Get the corresponding mode from the string value.
- const gchar * [gst_tensor_time_sync_get_mode_string](#) ([tensor_time_sync_mode](#) mode)
Get the time-sync mode string.
- gboolean [gst_tensor_time_sync_set_option_data](#) ([tensor_time_sync_data](#) *sync)
Setup time sync option.
- gboolean [gst_tensor_time_sync_get_current_time](#) (GstCollectPads *collect, [tensor_time_sync_data](#) *sync, GstClockTime *current_time, GstBuffer *tensors_buf)
A function call to decide current timestamp among collected pads based on PTS. It will decide current timestamp according to sync option. GstMeta is also copied with same sync mode.
- void [gst_tensor_time_sync_flush](#) (GstCollectPads *collect)
A function to be called while processing a flushing event. It should clear old buffer and reset pad data.
- gboolean [gst_tensor_time_sync_buffer_from_collectpad](#) (GstCollectPads *collect, [tensor_time_sync_data](#) *sync, GstClockTime current_time, GstBuffer *tensors_buf, [GstTensorsConfig](#) *configs, gboolean *is_eos)
A function call to make tensors from collected pads It decide which buffer is going to be used according to sync option.
- GstBuffer * [gst_tensor_buffer_from_config](#) (GstBuffer *in, [GstTensorsConfig](#) *config)
Configure gst-buffer with tensors information. NNStreamer handles single memory chunk as single tensor. If incoming buffer has invalid memories, separate it and generate new gst-buffer using tensors information. Note that this function always takes the ownership of input buffer.
- GstCaps * [gst_tensor_pad_caps_from_config](#) (GstPad *pad, const [GstTensorsConfig](#) *config)
Get pad caps from tensors config and caps of the peer connected to the pad.
- GstCaps * [gst_tensor_pad_possible_caps_from_config](#) (GstPad *pad, const [GstTensorsConfig](#) *config)
Get all possible caps from tensors config. Unlike [gst_tensor_pad_caps_from_config\(\)](#), this function does not check peer caps.
- [tensor_format](#) [gst_tensor_pad_get_format](#) (GstPad *pad)
Get tensor format of current pad caps.
- GHashTable * [gst_tensor_aggregation_init](#) (void)
Gets new hash table for tensor aggregation.
- void [gst_tensor_aggregation_clear](#) (GHashTable *table, const guint32 key)
Clears buffers from adapter.
- void [gst_tensor_aggregation_clear_all](#) (GHashTable *table)
Clears buffers from all adapters in hash table.
- GstAdapter * [gst_tensor_aggregation_get_adapter](#) (GHashTable *table, const guint32 key)
Gets adapter from hash table.

9.101.1 Detailed Description

Common header file for NNStreamer, the GStreamer plugin for neural networks.

NNStreamer Common Header Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

23 May 2018

See also

<https://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.101.2 Macro Definition Documentation

9.101.2.1 `gst_tensor_pad_caps_is_flexible`

```
#define gst_tensor_pad_caps_is_flexible(  
    p ) (gst_tensor_pad_get_format (p) == _NNS_TENSOR_FORMAT_FLEXIBLE)
```

Macro to check current pad caps is flexible tensor.

Definition at line 231 of file tensor_common.h.

9.101.2.2 `gst_tensor_pad_caps_is_sparse`

```
#define gst_tensor_pad_caps_is_sparse(  
    p ) (gst_tensor_pad_get_format (p) == _NNS_TENSOR_FORMAT_SPARSE)
```

Macro to check current pad caps is sparse tensor.

Definition at line 236 of file tensor_common.h.

9.101.2.3 `gst_tensor_pad_caps_is_static`

```
#define gst_tensor_pad_caps_is_static(  
    p ) (gst_tensor_pad_get_format (p) == _NNS_TENSOR_FORMAT_STATIC)
```

Macro to check current pad caps is static tensor.

Definition at line 226 of file tensor_common.h.

9.101.2.4 nns_memcpy

```
#define nns_memcpy memcpy
```

Definition at line 52 of file tensor_common.h.

9.101.2.5 nns_memset

```
#define nns_memset memset
```

Definition at line 53 of file tensor_common.h.

9.101.2.6 silent_debug

```
#define silent_debug(  
    self,  
    ... )
```

Value:

```
do { \  
    if (DBG) { \  
        GST_DEBUG_OBJECT (self, __VA_ARGS__); \  
    } \  
} while (0)
```

Macro for debug message.

Definition at line 276 of file tensor_common.h.

9.101.2.7 silent_debug_caps

```
#define silent_debug_caps(  
    self,  
    caps,  
    msg )
```

Value:

```
do { \  
    if (DBG) { \  
        if (caps) { \  
            gchar *caps_s_string = gst_caps_to_string (caps); \  
            GST_DEBUG_OBJECT (self, msg " = %s\n", caps_s_string); \  
            g_free (caps_s_string); \  
        } \  
    } \  
} while (0)
```

Macro for capability debug message.

Definition at line 285 of file tensor_common.h.

9.101.3 Typedef Documentation

9.101.3.1 `tensor_sync_basepad_data`

```
typedef struct _tensor_sync_basepad_data tensor_sync_basepad_data
```

Tensor Merge/Mux sync data for baspad mode.

9.101.3.2 `tensor_time_sync_data`

```
typedef struct _tensor_time_sync_data tensor_time_sync_data
```

Tensor Merge/Mux time sync data.

9.101.4 Enumeration Type Documentation

9.101.4.1 `tensor_time_sync_mode`

```
enum tensor_time_sync_mode
```

time synchronization options

See also

<https://github.com/nnstreamer/nnstreamer/wiki/Synchronization-Policies-at-Mux-and-Merge>

Enumerator

SYNC_NOSYNC	
SYNC_SLOWEST	
SYNC_BASEPAD	
SYNC_REFRESH	
SYNC_END	

Definition at line 62 of file `tensor_common.h`.

9.101.5 Function Documentation

9.101.5.1 `gst_tensor_aggregation_clear()`

```
void gst_tensor_aggregation_clear (
    GHashTable * table,
    const guint32 key )
```

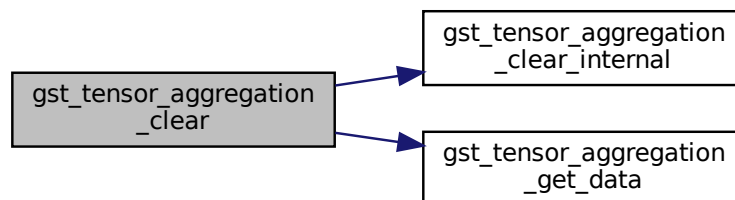
Clears buffers from adapter.

Parameters

<i>table</i>	a hash table instance initialized with gst_tensor_aggregation_init()
<i>key</i>	the key to look up (set null to get default adapter)

Definition at line 763 of file nntstreamer_plugin_api_impl.c.

Here is the call graph for this function:



9.101.5.2 `gst_tensor_aggregation_clear_all()`

```
void gst_tensor_aggregation_clear_all (
    GHashTable * table )
```

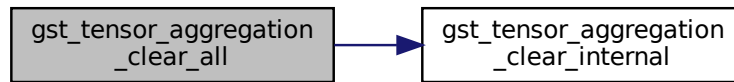
Clears buffers from all adapters in hash table.

Parameters

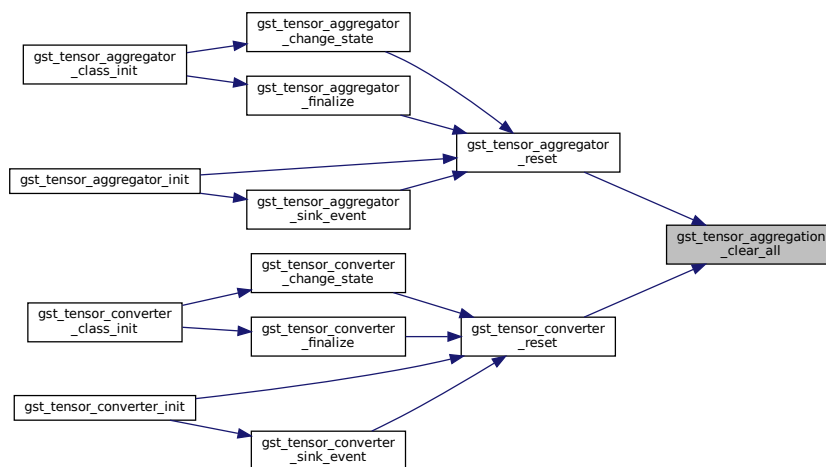
<i>table</i>	a hash table instance initialized with gst_tensor_aggregation_init()
--------------	--

Definition at line 778 of file nntstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.101.5.3 `gst_tensor_aggregation_get_adapter()`

```
GstAdapter* gst_tensor_aggregation_get_adapter (
    GHashTable * table,
    const guint32 key )
```

Gets adapter from hash table.

Parameters

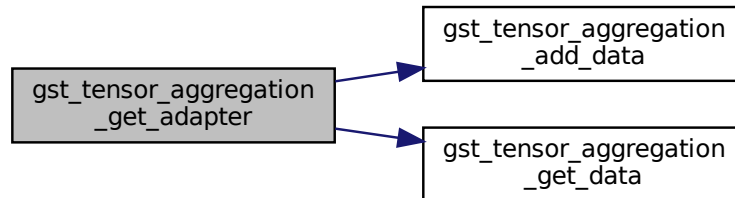
<i>table</i>	a hash table instance initialized with gst_tensor_aggregation_init()
<i>key</i>	the key to look up (set null to get default adapter)

Returns

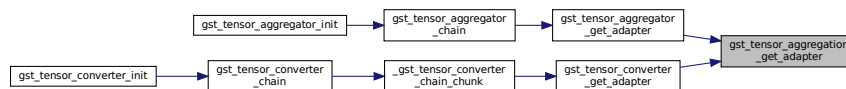
gst-adapter instance. DO NOT release this instance.

Definition at line 790 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.101.5.4 `gst_tensor_aggregation_init()`

```

GHashTable* gst_tensor_aggregation_init (
    void )
  
```

Gets new hash table for tensor aggregation.

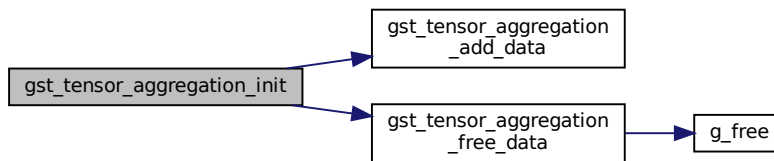
Returns

Newly allocated hash table, caller should release this using `g_hash_table_destroy()`.

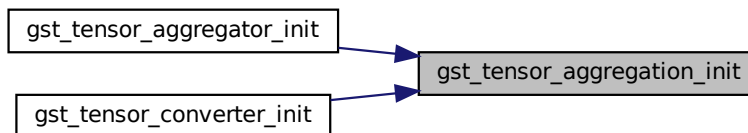
Add default adapter (for the case if buffer has no specific id). If `gst-buffer` has `tensor-meta` which includes `client-id`, e.g., aggregation frames from multiple clients on query-server pipeline, `nnstreamer` element should parse meta and request adapter with this id. However, on normal pipeline, `gst-buffer` does not contain `tensor-meta`, then the element may request adapter with null key string.

Definition at line 737 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.101.5.5 `gst_tensor_buffer_from_config()`

```

GstBuffer* gst_tensor_buffer_from_config (
    GstBuffer * in,
    GstTensorsConfig * config )
  
```

Configure `gst-buffer` with tensors information. `NNStreamer` handles single memory chunk as single tensor. If incoming buffer has invalid memories, separate it and generate new `gst-buffer` using tensors information. Note that this function always takes the ownership of input buffer.

Parameters

<i>in</i>	input buffer
<i>config</i>	tensors config structure

Returns

Newly allocated buffer. Null if failed. Caller should unref the buffer using `gst_buffer_unref()`.

Definition at line 535 of file `nnstreamer_plugin_api_impl.c`.

9.101.5.7 `gst_tensor_pad_get_format()`

```
tensor_format gst_tensor_pad_get_format (
    GstPad * pad )
```

Get tensor format of current pad caps.

Parameters

<i>pad</i>	GstPad to check current caps.
------------	-------------------------------

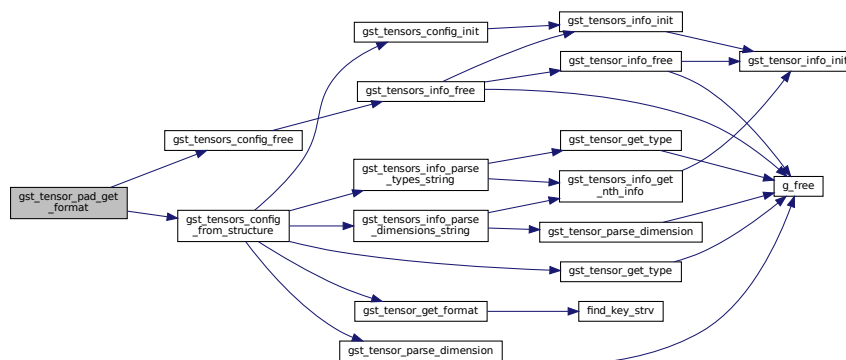
Returns

The `tensor_format` of current pad caps.

If pad does not have tensor caps return `_NNS_TENSOR_FORMAT_END`

Definition at line 1343 of file `nnstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



9.101.5.8 `gst_tensor_pad_possible_caps_from_config()`

```
GstCaps* gst_tensor_pad_possible_caps_from_config (
    GstPad * pad,
    const GstTensorsConfig * config )
```

Get all possible caps from tensors config. Unlike `gst_tensor_pad_caps_from_config()`, this function does not check peer caps.

Parameters

<i>pad</i>	GstPad to get possible caps
<i>config</i>	tensors config structure

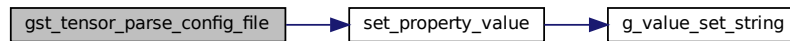
Note

The responsibility of managing the memory of the GObject passed as a parameter lies outside this function.

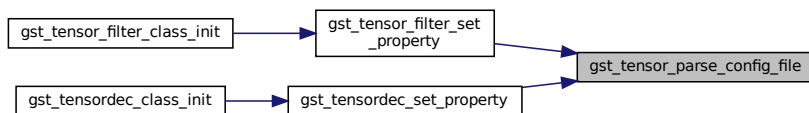
Iterate over each line

Definition at line 1902 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.101.5.10 gst_tensor_time_sync_buffer_from_collectpad()**

```

gboolean gst_tensor_time_sync_buffer_from_collectpad (
    GstCollectPads * collect,
    tensor_time_sync_data * sync,
    GstClockTime current_time,
    GstBuffer * tensors_buf,
    GstTensorsConfig * configs,
    gboolean * is_eos )
  
```

A function call to make tensors from collected pads It decide which buffer is going to be used according to sync option.

Returns

True to push buffer.

Parameters

<i>collect</i>	Collect pad.
<i>sync</i>	Synchronization Option (NOSYNC, SLOWEST, BASEPAD, END)
<i>current_time</i>	Current Timestamp
<i>tensors_buf</i>	Generated GstBuffer for Collected Buffer
<i>configs</i>	Configuration Info for Collected Buffer
<i>is_eos</i>	True when EOS (end-of-stream)

9.101.5.11 `gst_tensor_time_sync_flush()`

```
void gst_tensor_time_sync_flush (
    GstCollectPads * collect )
```

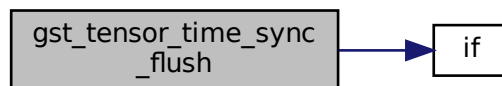
A function to be called while processing a flushing event. It should clear old buffer and reset pad data.

Parameters

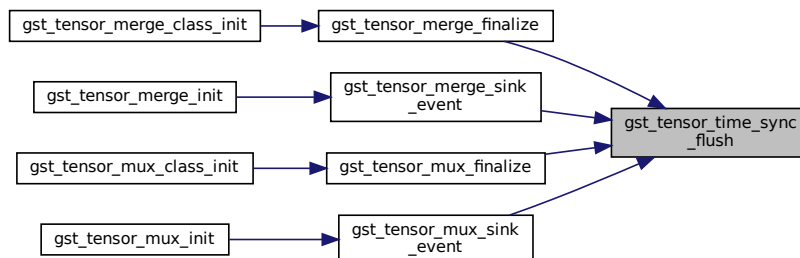
<i>collect</i>	Collect pad.
----------------	--------------

Definition at line 263 of file `nntstreamer_plugin_api_impl.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.101.5.12 `gst_tensor_time_sync_get_current_time()`

```
gboolean gst_tensor_time_sync_get_current_time (
    GstCollectPads * collect,
    tensor_time_sync_data * sync,
    GstClockTime * current_time,
    GstBuffer * tensors_buf )
```

A function call to decide current timestamp among collected pads based on PTS. It will decide current timestamp according to sync option. `GstMeta` is also copied with same sync mode.

Returns

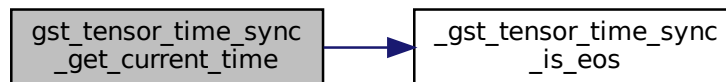
True / False. If EOS, it return TRUE.

Parameters

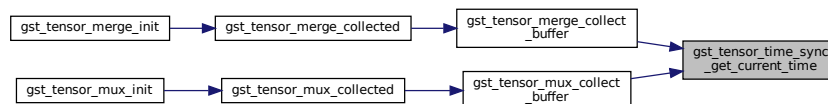
<i>collect</i>	Collect pad.
<i>sync</i>	Synchronization Option (NOSYNC, SLOWEST, BASEPAD, END)
<i>current_time</i>	Current time
<i>tensors_buf</i>	Generated GstBuffer for Collected Buffer

Definition at line 203 of file nnstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.101.5.13 gst_tensor_time_sync_get_mode()**

```

tensor_time_sync_mode gst_tensor_time_sync_get_mode (
    const gchar * str )
  
```

Get the corresponding mode from the string value.

Parameters

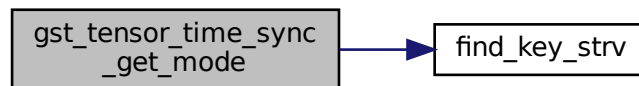
in	<i>str</i>	The string value for the mode.
----	------------	--------------------------------

Returns

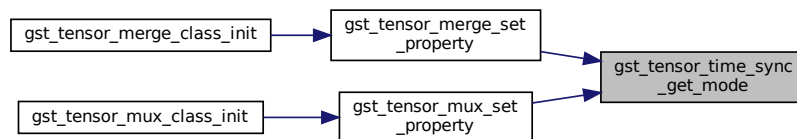
Corresponding mode for the string. SYNC_END for errors.

Definition at line 101 of file nntstreamer_plugin_api_impl.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.101.5.14 gst_tensor_time_sync_get_mode_string()**

```

const gchar* gst_tensor_time_sync_get_mode_string (
    tensor_time_sync_mode mode )
  
```

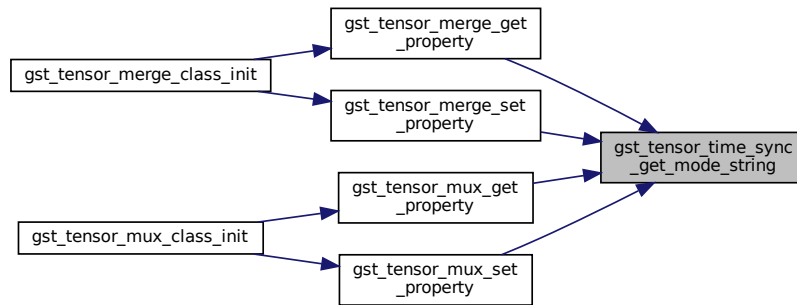
Get the time-sync mode string.

Returns

Corresponding mode string.

Definition at line 115 of file nntstreamer_plugin_api_impl.c.

Here is the caller graph for this function:



9.101.5.15 `gst_tensor_time_sync_set_option_data()`

```
gboolean gst_tensor_time_sync_set_option_data (
    tensor_time_sync_data * sync )
```

Setup time sync option.

Parameters

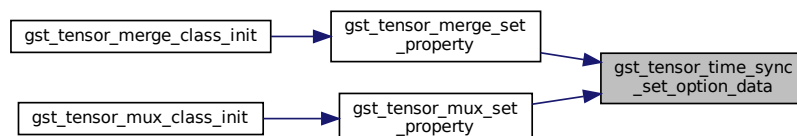
<i>[in/out]</i>	filter "this" pointer. Sync mode & option MUST BE set already.
-----------------	--

Returns

True if successfully set the option.

Definition at line 126 of file `nstreamer_plugin_api_impl.c`.

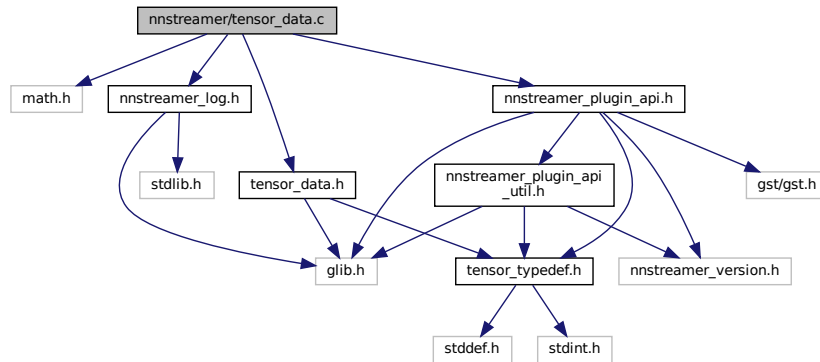
Here is the caller graph for this function:



9.102 `nstreamer/tensor_data.c` File Reference

Internal functions to handle various tensor type and value.

```
#include <math.h>
#include "tensor_data.h"
#include "nnstreamer_log.h"
#include "nnstreamer_plugin_api.h"
Include dependency graph for tensor_data.c:
```



Macros

- `#define td_set_data(td, v, dtype)`
Macro to set data in struct.
- `#define td_get_data(td, v, dtype)`
- `#define td_typecast_to(td, itype, otype)`
- `#define td_typecast_to_fromf16(td, itype)`
- `#define td_typecast(td, otype)`

Functions

- `while (0)`
`dtype = *((dtype *) v); |`
- gboolean `gst_tensor_data_get (tensor_data_s *td, gpointer value)`
Get tensor element value.
- gboolean `gst_tensor_data_typecast (tensor_data_s *td, tensor_type type)`
Typecast tensor element data.
- gboolean `gst_tensor_data_raw_typecast (gpointer input, tensor_type in_type, gpointer output, tensor_type out_type)`
Typecast tensor element value.
- gboolean `gst_tensor_data_raw_average (gpointer raw, gsize length, tensor_type type, gdouble **result)`
Calculate average value of the tensor.
- gboolean `gst_tensor_data_raw_average_per_channel (gpointer raw, gsize length, tensor_type type, tensor_dim dim, gdouble **results)`
Calculate average value of the tensor per channel (the first dim).
- gboolean `gst_tensor_data_raw_std (gpointer raw, gsize length, tensor_type type, gdouble *average, gdouble **result)`
Calculate standard deviation of the tensor.
- gboolean `gst_tensor_data_raw_std_per_channel (gpointer raw, gsize length, tensor_type type, tensor_dim dim, gdouble *averages, gdouble **results)`
Calculate standard deviation of the tensor per channel (the first dim).

9.102.1 Detailed Description

Internal functions to handle various tensor type and value.

Copyright (c) 2021 Samsung Electronics Co., Ltd. All Rights Reserved.

Date

10 Mar 2021

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung [jy1210.jung@samsung.com](mailto: jy1210.jung@samsung.com)

Bug No known bugs except for NYI items

9.102.2 Macro Definition Documentation

9.102.2.1 td_get_data

```
#define td_get_data(  
    td,  
    v,  
    dtype )
```

Value:

```
do { \  
    *((dtype *) v) = (td)->data._
```

9.102.2.2 td_set_data

```
#define td_set_data(  
    td,  
    v,  
    dtype )
```

Value:

```
do { \  
    (td)->data._
```

Macro to set data in struct.

Definition at line 21 of file tensor_data.c.

9.102.2.3 td_typecast

```
#define td_typecast(
    td,
    otype )
```

Value:

```
do { \
    switch ((td)->type) { \
        case _NNS_INT32: td_typecast_to (td, int32_t, otype); break; \
        case _NNS_UINT32: td_typecast_to (td, uint32_t, otype); break; \
        case _NNS_INT16: td_typecast_to (td, int16_t, otype); break; \
        case _NNS_UINT16: td_typecast_to (td, uint16_t, otype); break; \
        case _NNS_INT8: td_typecast_to (td, int8_t, otype); break; \
        case _NNS_UINT8: td_typecast_to (td, uint8_t, otype); break; \
        case _NNS_FLOAT64: td_typecast_to (td, double, otype); break; \
        case _NNS_FLOAT32: td_typecast_to (td, float, otype); break; \
        case _NNS_FLOAT16: td_typecast_to_fromf16 (td, otype); break; \
        case _NNS_INT64: td_typecast_to (td, int64_t, otype); break; \
        case _NNS_UINT64: td_typecast_to (td, uint64_t, otype); break; \
        default: g_assert (0); break; \
    } \
} while (0)
```

9.102.2.4 td_typecast_to

```
#define td_typecast_to(
    td,
    itype,
    otype )
```

Value:

```
do { \
    itype in_val = (td)->data._
```

9.102.2.5 td_typecast_to_fromf16

```
#define td_typecast_to_fromf16(
    td,
    itype )
```

Value:

```
do { \
    nns_loge ("Your nstreamer binary is built without -DFLOAT16_SUPPORT option; thus float16 is not supported.\n"); \
    g_assert (0); \
} while (0)
```

9.102.3 Function Documentation

9.102.3.1 gst_tensor_data_get()

```
gboolean gst_tensor_data_get (
    tensor_data_s * td,
    gpointer value )
```

Get tensor element value.

Parameters

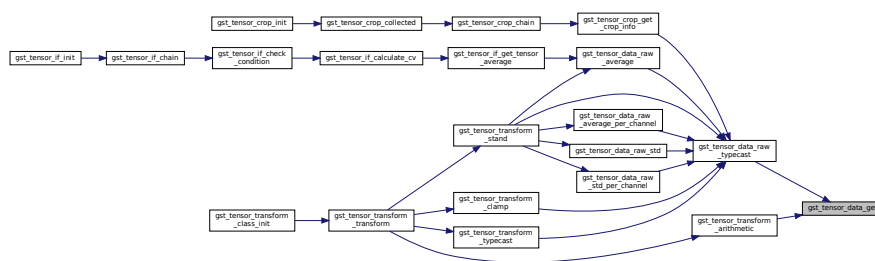
<i>td</i>	struct for tensor data
<i>value</i>	pointer of tensor element value

Returns

TRUE if no error

Definition at line 143 of file tensor_data.c.

Here is the caller graph for this function:



9.102.3.2 `gst_tensor_data_raw_average()`

```

gboolean gst_tensor_data_raw_average (
    gpointer raw,
    gsize length,
    tensor_type type,
    gdouble ** result )
  
```

Calculate average value of the tensor.

Parameters

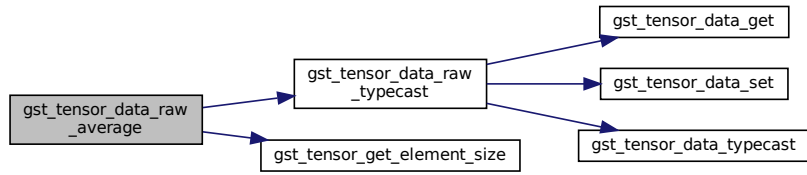
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>result</i>	double pointer for average value of given tensor. Caller should release allocated memory.

Returns

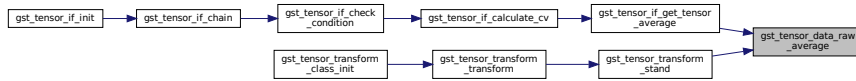
TRUE if no error

Definition at line 315 of file tensor_data.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.102.3.3 gst_tensor_data_raw_average_per_channel()

```

gboolean gst_tensor_data_raw_average_per_channel (
    gpointer raw,
    gsize length,
    tensor_type type,
    tensor_dim dim,
    gdouble ** results )
  
```

Calculate average value of the tensor per channel (the first dim).

Parameters

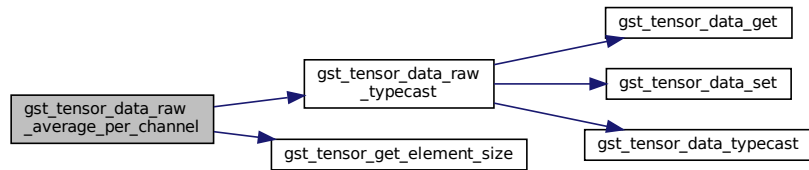
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>tensor_dim</i>	tensor dimension
<i>results</i>	double array contains average values of each channel. Caller should release allocated array.

Returns

TRUE if no error

Definition at line 360 of file tensor_data.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.102.3.4 `gst_tensor_data_raw_std()`

```

gboolean gst_tensor_data_raw_std (
    gpointer raw,
    gsize length,
    tensor_type type,
    gdouble * average,
    gdouble ** result )
  
```

Calculate standard deviation of the tensor.

Parameters

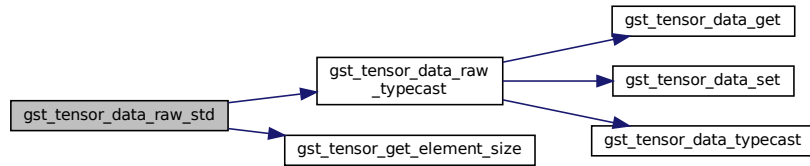
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>average</i>	average value of given tensor
<i>result</i>	double pointer for standard deviation of given tensor. Caller should release allocated memory.

Returns

TRUE if no error

Definition at line 409 of file `tensor_data.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.102.3.5 gst_tensor_data_raw_std_per_channel()

```

gboolean gst_tensor_data_raw_std_per_channel (
    gpointer raw,
    gsize length,
    tensor_type type,
    tensor_dim dim,
    gdouble * averages,
    gdouble ** results )
  
```

Calculate standard deviation of the tensor per channel (the first dim).

Parameters

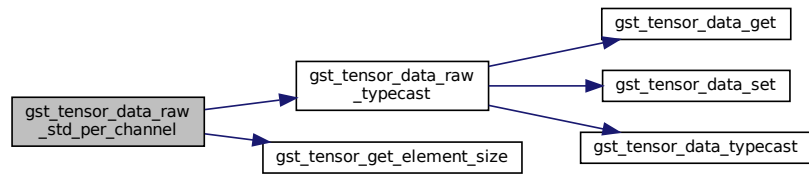
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>tensor_dim</i>	tensor dimension
<i>averages</i>	average values of given tensor per-channel
<i>results</i>	double array contains standard deviation of each channel. Caller should release allocated array.

Returns

TRUE if no error

Definition at line 455 of file tensor_data.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.102.3.6 `gst_tensor_data_raw_typecast()`

```

gboolean gst_tensor_data_raw_typecast (
    gpointer input,
    tensor_type in_type,
    gpointer output,
    tensor_type out_type )
  
```

Typecast tensor element value.

Parameters

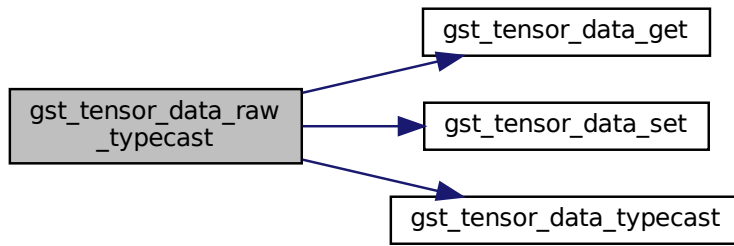
<i>input</i>	pointer of input tensor data
<i>in_type</i>	input tensor type
<i>output</i>	pointer of output tensor data
<i>out_type</i>	output tensor type

Returns

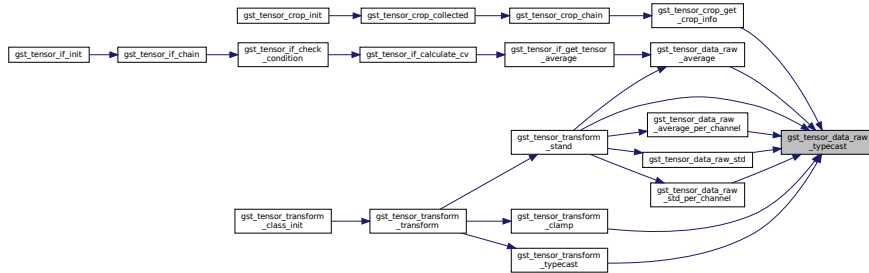
TRUE if no error

Definition at line 290 of file `tensor_data.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.102.3.7 gst_tensor_data_typecast()

```

gboolean gst_tensor_data_typecast (
    tensor_data_s * td,
    tensor_type type )
  
```

Typecast tensor element data.

Parameters

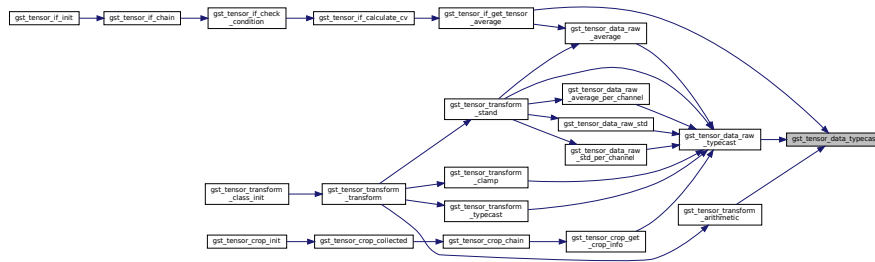
<i>td</i>	struct for tensor data
<i>type</i>	tensor type to be transformed

Returns

TRUE if no error

Definition at line 203 of file tensor_data.c.

Here is the caller graph for this function:



9.102.3.8 while()

```
while (
    0 )

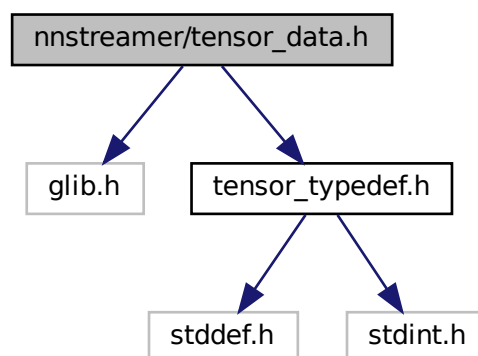
dtype = *((dtype *) v); \
```

Definition at line 23 of file tensor_data.c.

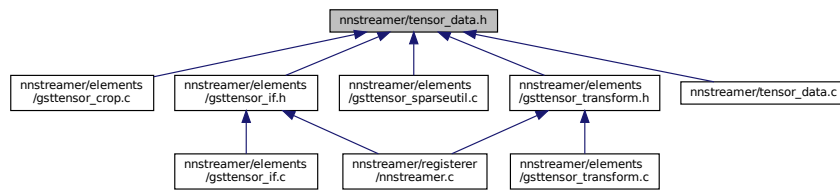
9.103 nnstreamer/tensor_data.h File Reference

Internal functions to handle various tensor type and value.

```
#include <glib.h>
#include <tensor_typedef.h>
Include dependency graph for tensor_data.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [tensor_data_s](#)
Structure for tensor data.

Functions

- gboolean [gst_tensor_data_set](#) ([tensor_data_s](#) *td, [tensor_type](#) type, gpointer value)
Set tensor element data with given type.
- gboolean [gst_tensor_data_get](#) ([tensor_data_s](#) *td, gpointer value)
Get tensor element value.
- gboolean [gst_tensor_data_typecast](#) ([tensor_data_s](#) *td, [tensor_type](#) type)
Typecast tensor element data.
- gboolean [gst_tensor_data_raw_typecast](#) (gpointer input, [tensor_type](#) in_type, gpointer output, [tensor_type](#) out_type)
Typecast tensor element value.
- gboolean [gst_tensor_data_raw_average](#) (gpointer raw, gsize length, [tensor_type](#) type, gdouble **result)
Calculate average value of the tensor.
- gboolean [gst_tensor_data_raw_average_per_channel](#) (gpointer raw, gsize length, [tensor_type](#) type, [tensor_dim](#) dim, gdouble **results)
Calculate average value of the tensor per channel (the first dim).
- gboolean [gst_tensor_data_raw_std](#) (gpointer raw, gsize length, [tensor_type](#) type, gdouble *average, gdouble **result)
Calculate standard deviation of the tensor.
- gboolean [gst_tensor_data_raw_std_per_channel](#) (gpointer raw, gsize length, [tensor_type](#) type, [tensor_dim](#) dim, gdouble *averages, gdouble **results)
Calculate standard deviation of the tensor per channel (the first dim).

9.103.1 Detailed Description

Internal functions to handle various tensor type and value.

Copyright (c) 2021 Samsung Electronics Co., Ltd. All Rights Reserved.

Date

10 Mar 2021

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Jaeyun Jung jy1210.jung@samsung.com

Bug No known bugs except for NYI items

9.103.2 Function Documentation

9.103.2.1 `gst_tensor_data_get()`

```
gboolean gst_tensor_data_get (
    tensor_data_s * td,
    gpointer value )
```

Get tensor element value.

Parameters

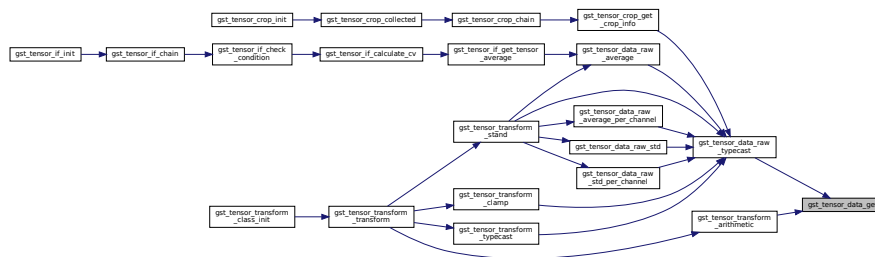
<i>td</i>	struct for tensor data
<i>value</i>	pointer of tensor element value

Returns

TRUE if no error

Definition at line 143 of file `tensor_data.c`.

Here is the caller graph for this function:



9.103.2.2 `gst_tensor_data_raw_average()`

```
gboolean gst_tensor_data_raw_average (
    gpointer raw,
    gsize length,
    tensor_type type,
    gdouble ** result )
```

Calculate average value of the tensor.

Parameters

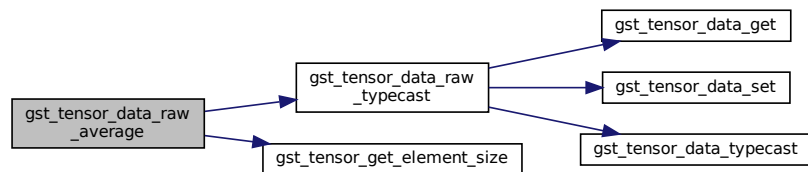
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>result</i>	double pointer for average value of given tensor. Caller should release allocated memory.

Returns

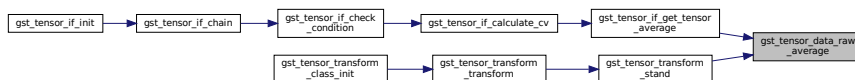
TRUE if no error

Definition at line 315 of file `tensor_data.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.103.2.3 `gst_tensor_data_raw_average_per_channel()`

```
gboolean gst_tensor_data_raw_average_per_channel (
    gpointer raw,
    gsize length,
    tensor_type type,
    tensor_dim dim,
    gdouble ** results )
```

Calculate average value of the tensor per channel (the first dim).

Parameters

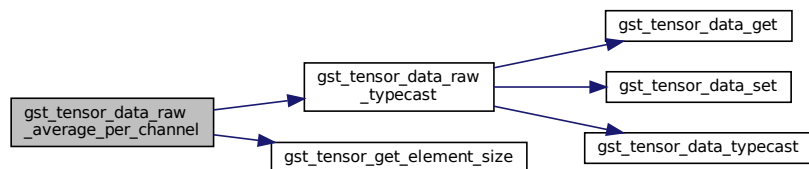
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>tensor_dim</i>	tensor dimension
<i>results</i>	double array contains average values of each channel. Caller should release allocated array.

Returns

TRUE if no error

Definition at line 360 of file tensor_data.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.103.2.4 `gst_tensor_data_raw_std()`

```

gboolean gst_tensor_data_raw_std (
    gpointer raw,
    gsize length,
    tensor_type type,
    gdouble * average,
    gdouble ** result )
  
```

Calculate standard deviation of the tensor.

Parameters

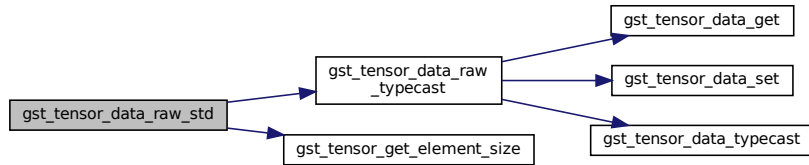
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>average</i>	average value of given tensor
<i>result</i>	double pointer for standard deviation of given tensor. Caller should release allocated memory.

Returns

TRUE if no error

Definition at line 409 of file tensor_data.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.103.2.5 `gst_tensor_data_raw_std_per_channel()`

```

gboolean gst_tensor_data_raw_std_per_channel (
    gpointer raw,
    gsize length,
    tensor_type type,
    tensor_dim dim,
    gdouble * averages,
    gdouble ** results )
  
```

Calculate standard deviation of the tensor per channel (the first dim).

Parameters

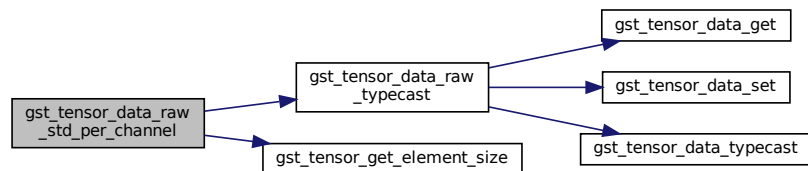
<i>raw</i>	pointer of raw tensor data
<i>length</i>	byte size of raw tensor data
<i>type</i>	tensor type
<i>tensor_dim</i>	tensor dimension
<i>averages</i>	average values of given tensor per-channel
<i>results</i>	double array contains standard deviation of each channel. Caller should release allocated array.

Returns

TRUE if no error

Definition at line 455 of file tensor_data.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.103.2.6 gst_tensor_data_raw_typecast()**

```

gboolean gst_tensor_data_raw_typecast (
    gpointer input,
    tensor_type in_type,
    gpointer output,
    tensor_type out_type )
  
```

Typecast tensor element value.

Parameters

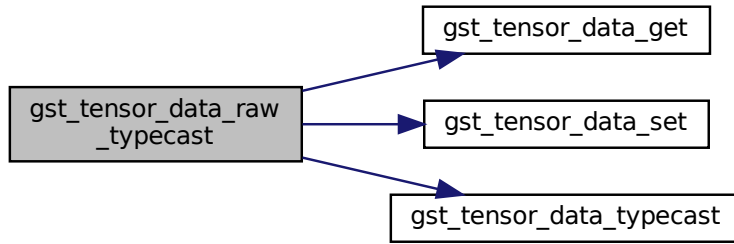
<i>input</i>	pointer of input tensor data
<i>in_type</i>	input tensor type
<i>output</i>	pointer of output tensor data
<i>out_type</i>	output tensor type

Returns

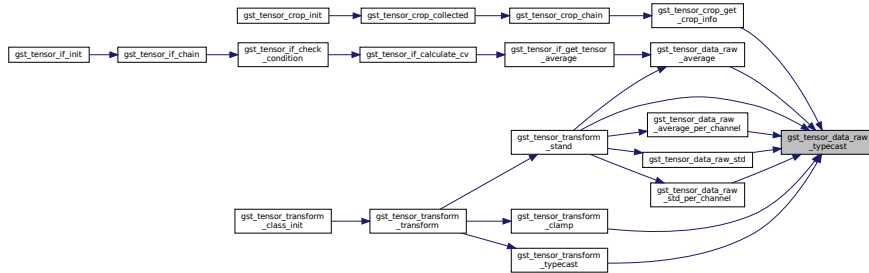
TRUE if no error

Definition at line 290 of file tensor_data.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.103.2.7 gst_tensor_data_set()

```

gboolean gst_tensor_data_set (
    tensor_data_s * td,
    tensor_type type,
    gpointer value )
  
```

Set tensor element data with given type.

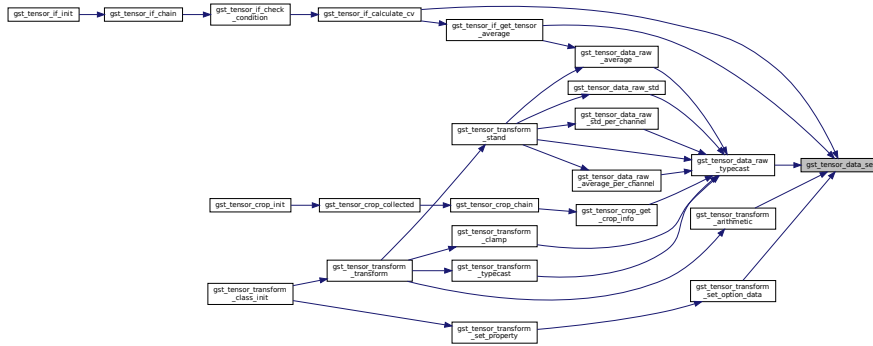
Parameters

<i>td</i>	struct for tensor data
<i>type</i>	tensor type
<i>value</i>	pointer of tensor element value

Returns

TRUE if no error

Here is the caller graph for this function:



9.103.2.8 gst_tensor_data_typecast()

```
gboolean gst_tensor_data_typecast (
    tensor_data_s * td,
    tensor_type type )
```

Typecast tensor element data.

Parameters

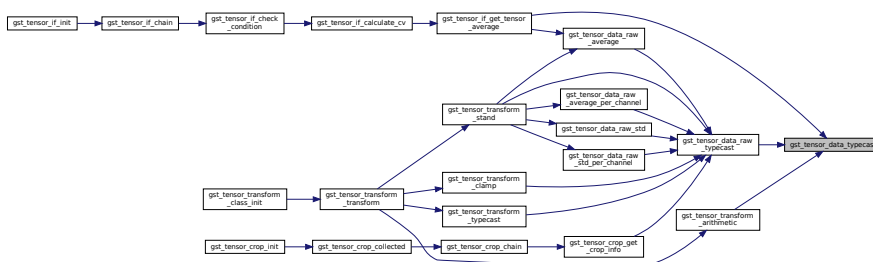
<i>td</i>	struct for tensor data
<i>type</i>	tensor type to be transformed

Returns

TRUE if no error

Definition at line 203 of file tensor_data.c.

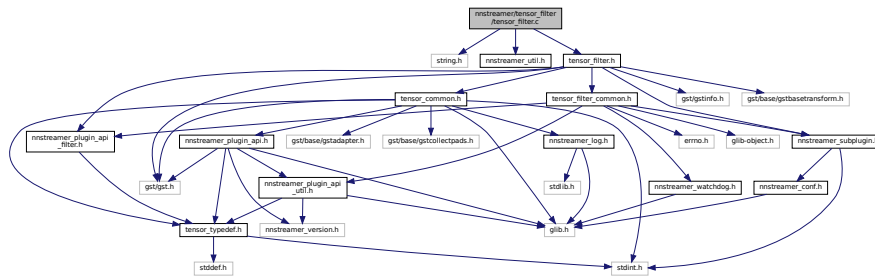
Here is the caller graph for this function:



9.104 nnstreamer/tensor_filter/tensor_filter.c File Reference

GStreamer plugin to use general neural network frameworks as filters.

```
#include <string.h>
#include <nnstreamer_util.h>
#include "tensor_filter.h"
Include dependency graph for tensor_filter.c:
```



Classes

- struct [_FilterTransformData](#)

Internal data structure for tensor_filter transform data.

Macros

- #define [EVENT_NAME_UPDATE_MODEL](#) "evt_update_model"
- #define [GST_CAT_DEFAULT](#) gst_tensor_filter_debug
- #define [TF_MODELNAME\(prop\)](#) ((prop)->model_files ? ((prop)->model_files[0]) : "[No Model File]")
- #define [CAPS_STRING](#) [GST_TENSOR_CAP_DEFAULT](#) ";" [GST_TENSORS_CAP_MAKE](#) ("{" static, flexible }")
Default caps string for both sink and source pad.
- #define [gst_tensor_filter_parent_class](#) parent_class
- #define [LATENCY_REPORT_HEADROOM](#) 0.05
Headroom (extra duration) added to actual latency estimate reported to LATENCY query, to limit number of updates when tracking the maximum value - arbitrarily set to 5%.
- #define [LATENCY_REPORT_THRESHOLD](#) 0.25
Threshold deciding when tracking latency estimate that current value is sufficiently lower than reported value so that a notification update is necessary - arbitrarily set to 25%.
- #define [THRESHOLD_DROP_OLD](#) (2000)
- #define [THRESHOLD_CACHE_OLD](#) (1000)

Typedefs

- typedef struct [_FilterTransformData](#) [FilterTransformData](#)

Internal data structure for tensor_filter transform data.

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_filter_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorFilter](#), [gst_tensor_filter](#), [GST_TYPE_BASE_TRANSFORM](#))
- static void [gst_tensor_filter_set_property](#) ([GObject](#) *object, [guint](#) prop_id, [const GValue](#) *value, [GParamSpec](#) *pspec)
Setter for tensor_filter properties.
- static void [gst_tensor_filter_get_property](#) ([GObject](#) *object, [guint](#) prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
Getter for tensor_filter properties.
- static void [gst_tensor_filter_finalize](#) ([GObject](#) *object)
Function to finalize instance.
- static [GstFlowReturn](#) [gst_tensor_filter_transform](#) ([GstBaseTransform](#) *trans, [GstBuffer](#) *inbuf, [GstBuffer](#) *outbuf)
non-ip transform. required vmethod of GstBaseTransform.
- static [GstCaps](#) * [gst_tensor_filter_transform_caps](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [GstCaps](#) *filter)
configure tensor-srcpad cap from "proposed" cap.
- static [GstCaps](#) * [gst_tensor_filter_fixate_caps](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [GstCaps](#) *othercaps)
fixate caps. required vmethod of GstBaseTransform.
- static [gboolean](#) [gst_tensor_filter_set_caps](#) ([GstBaseTransform](#) *trans, [GstCaps](#) *incaps, [GstCaps](#) *outcaps)
set caps. required vmethod of GstBaseTransform.
- static [gboolean](#) [gst_tensor_filter_query](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstQuery](#) *query)
query handling, optional vmethod of GstBaseTransform.
- static [gboolean](#) [gst_tensor_filter_transform_size](#) ([GstBaseTransform](#) *trans, [GstPadDirection](#) direction, [GstCaps](#) *caps, [gsize](#) size, [GstCaps](#) *othercaps, [gsize](#) *othersize)
Tell the framework the required size of buffer based on the info of the other side pad. optional vmethod of BaseTransform.
- static [gboolean](#) [gst_tensor_filter_start](#) ([GstBaseTransform](#) *trans)
Called when the element starts processing. optional vmethod of BaseTransform.
- static [gboolean](#) [gst_tensor_filter_stop](#) ([GstBaseTransform](#) *trans)
Called when the element stops processing. optional vmethod of BaseTransform.
- static [gboolean](#) [gst_tensor_filter_sink_event](#) ([GstBaseTransform](#) *trans, [GstEvent](#) *event)
Event handler for sink pad of tensor filter.
- static [gboolean](#) [gst_tensor_filter_src_event](#) ([GstBaseTransform](#) *trans, [GstEvent](#) *event)
Event handler for src pad of tensor filter.
- static void [gst_tensor_filter_class_init](#) ([GstTensorFilterClass](#) *klass)
initialize the tensor_filter's class
- static void [gst_tensor_filter_init](#) ([GstTensorFilter](#) *self)
initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure
- static [gsize](#) [gst_tensor_filter_get_tensor_size](#) ([GstTensorFilter](#) *self, [guint](#) index, [gboolean](#) is_input)
Calculate tensor buffer size.
- static void [gst_tensor_filter_destroy_notify](#) ([void](#) *data)
Free the data allocated for tensor transform.
- static [GstMemory](#) * [gst_tensor_filter_get_wrapped_mem](#) ([GstTensorFilter](#) *self, [gpointer](#) data, [gsize](#) size)
Allocate new memory block from given data.
- static void [prepare_statistics](#) ([GstTensorFilterPrivate](#) *priv)
Prepare statistics for performance profiling (e.g, latency, throughput)
- static void [accumulate_latency](#) ([void](#) *data, [void](#) *user_data)

- Helper function to accumulate latencies.*
- static void [record_statistics](#) ([GstTensorFilterPrivate](#) *priv)

Record statistics for performance profiling (e.g, latency, throughput)
 - static void [track_latency](#) ([GstTensorFilter](#) *self)

Track estimated latency and notify pipeline when it changes. Latency estimates may be a bit jittery. On the principle we want to inform pipeline with the latency from longest inference. However, first inference may take much longer, or model filter configuration may change. Therefore any change of more than 10% (arbitrary value) to a lower latency is also reported to pipeline. Notification is done sending LATENCY message to bus. Upon receipt, application will initiate a pipeline latency probe via LATENCY query.
 - static gboolean [gst_tensor_filter_check_throttling_delay](#) ([GstBaseTransform](#) *trans, [GstBuffer](#) *inbuf)

Check throttling delay and send qos overflow event to upstream elements.
 - static [GstFlowReturn](#) [_gst_tensor_filter_transform_validate](#) ([GstBaseTransform](#) *trans, [GstBuffer](#) *inbuf, [GstBuffer](#) *outbuf)

Check input parameters for [gst_tensor_filter_transform](#) ();
 - static void [_gst_tensor_filter_release_mem_until_idx](#) ([FilterTransformData](#) *trans_data, guint end_index)

Internal function to release mem and unmap info with index.
 - static gsize [_gst_tensor_filter_convert_meta](#) ([FilterTransformData](#) *trans_data, [GstTensorsInfo](#) *info, guint idx)

Internal function to convert tensor meta and get header size of flexible tensor.
 - static [FilterTransformData](#) * [_gst_tensor_filter_transform_get_all_input_data](#) ([GstBaseTransform](#) *trans, [GstBuffer](#) *buf)

Internal function to get input tensors.
 - static [GstTensorMemory](#) * [_gst_tensor_filter_transform_get_invoke_tensors](#) ([GstBaseTransform](#) *trans, [FilterTransformData](#) *trans_data)

Internal function to get invoke tensors.
 - static [FilterTransformData](#) * [_gst_tensor_filter_transform_get_output_data](#) ([GstBaseTransform](#) *trans)

Internal function to get output tensors.
 - static [GstFlowReturn](#) [_gst_tensor_filter_transform_prepare_output_tensors](#) ([GstBaseTransform](#) *trans, [FilterTransformData](#) *trans_data)

Internal function to get output tensors.
 - static [GstFlowReturn](#) [_gst_tensor_filter_transform_check_invoke_result](#) ([GstBaseTransform](#) *trans, [FilterTransformData](#) *in_trans_data, [FilterTransformData](#) *out_trans_data, gint invoke_res)

Internal function to check the invoke result.
 - static void [_gst_tensor_filter_transform_update_outbuf](#) ([GstBaseTransform](#) *trans, [FilterTransformData](#) *in_trans_data, [FilterTransformData](#) *out_trans_data, [GstBuffer](#) *outbuf)

Internal function to make output buffer.
 - static gboolean [gst_tensor_filter_watchdog_trigger](#) (gpointer ptr)

Called when there is no input within suspend time specified by the user.
 - static gboolean [gst_tensor_filter_configure_tensor](#) ([GstTensorFilter](#) *self, const [GstCaps](#) *incaps)

Configure input and output tensor info from incaps.

Variables

- static [GstStaticPadTemplate](#) [sink_factory](#)

The capabilities of the inputs.
- static [GstStaticPadTemplate](#) [src_factory](#)

The capabilities of the outputs.

9.104.1 Detailed Description

GStreamer plugin to use general neural network frameworks as filters.

GStreamer Tensor_Filter Copyright (C) 2005 Thomas Vander Stichele thomas@pestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

24 May 2018

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

Todo set priority among properties

logic for dynamic properties(like model change)

This is the main plugin for per-NN-framework plugins. Specific implementations for each NN framework must be written in each framework specific files; e.g., `tensor_filter_tensorflow_lite.c`

9.104.2 Macro Definition Documentation

9.104.2.1 CAPS_STRING

```
#define CAPS_STRING GST_TENSOR_CAP_DEFAULT ";" GST_TENSORS_CAP_MAKE ("{ static, flexible }")
```

Default caps string for both sink and source pad.

Definition at line 87 of file `tensor_filter.c`.

9.104.2.2 EVENT_NAME_UPDATE_MODEL

```
#define EVENT_NAME_UPDATE_MODEL "evt_update_model"
```

SECTION:element-tensor_filter

A plugin that invokes neural network models and their framework or an independent shared object implementing [tensor_filter_custom.h](#). The input and output are always in the format of other/tensor or other/tensors.

```
<refsect2> <title>Example launch line</title> |[ gst-launch -v -m fakesrc ! tensor_filter framework=tensorflow-lite, model=./inception_v3.pb, input=3:224:224, output=1000 ! fakesink silent=TRUE ]| </refsect2>
```

If input is other/tensor C array input[1][224][224][3] and output is other/tensor C array output[1][1][1][1000]

The current QoS policy: In a nntstreamer pipeline, the QoS is currently satisfied by adjusting a input or output framerate, initiated by 'tensor_rate' element. When 'tensor_filter' receives a throttling QoS event from the 'tensor_rate' element, it compares the average processing latency and throttling delay, and takes the maximum value as the threshold to drop incoming frames by checking a buffer timestamp. In this way, 'tensor filter' can avoid unnecessary calculation and adjust a framerate, effectively reducing resource utilizations. Even in the case of receiving QoS events from multiple downstream pipelines (e.g., tee), 'tensor_filter' takes the minimum value as the throttling delay for downstream pipeline with more tight QoS requirement. Lastly, 'tensor_filter' also sends QoS events to upstream elements (e.g., tensor_converter, tensor_src) to possibly reduce incoming framerates, which is a better solution than dropping framerates.

Todo rename & move this to better location

Definition at line 76 of file tensor_filter.c.

9.104.2.3 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_filter_debug
```

Definition at line 79 of file tensor_filter.c.

9.104.2.4 gst_tensor_filter_parent_class

```
#define gst_tensor_filter_parent_class parent_class
```

Definition at line 105 of file tensor_filter.c.

9.104.2.5 LATENCY_REPORT_HEADROOM

```
#define LATENCY_REPORT_HEADROOM 0.05
```

Headroom (extra duration) added to actual latency estimate reported to LATENCY query, to limit number of updates when tracking the maximum value - arbitrarily set to 5%.

Definition at line 113 of file tensor_filter.c.

9.104.2.6 LATENCY_REPORT_THRESHOLD

```
#define LATENCY_REPORT_THRESHOLD 0.25
```

Threshold deciding when tracking latency estimate that current value is sufficiently lower than reported value so that a notification update is necessary - arbitrarily set to 25%.

Definition at line 120 of file `tensor_filter.c`.

9.104.2.7 TF_MODELNAME

```
#define TF_MODELNAME(  
    prop ) ((prop)->model_files ? ((prop)->model_files[0]) : "[No Model File]")
```

Definition at line 81 of file `tensor_filter.c`.

9.104.2.8 THRESHOLD_CACHE_OLD

```
#define THRESHOLD_CACHE_OLD (1000)
```

Definition at line 406 of file `tensor_filter.c`.

9.104.2.9 THRESHOLD_DROP_OLD

```
#define THRESHOLD_DROP_OLD (2000)
```

Definition at line 405 of file `tensor_filter.c`.

9.104.3 Typedef Documentation

9.104.3.1 FilterTransformData

```
typedef struct _FilterTransformData FilterTransformData
```

Internal data structure for `tensor_filter` transform data.

9.104.4 Function Documentation

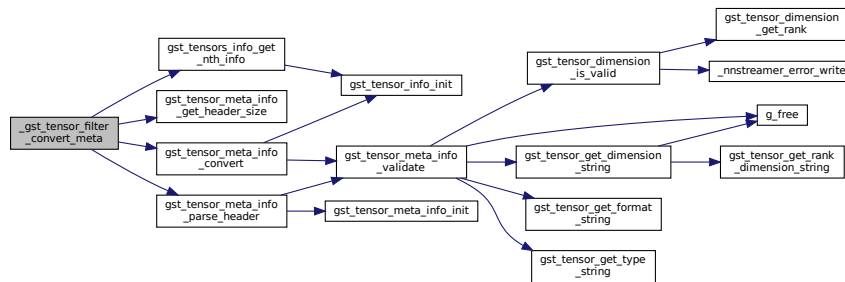
9.104.4.1 `_gst_tensor_filter_convert_meta()`

```
static gsize _gst_tensor_filter_convert_meta (
    FilterTransformData * trans_data,
    GstTensorsInfo * info,
    guint idx ) [static]
```

Internal function to convert tensor meta and get header size of flexible tensor.

Definition at line 684 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



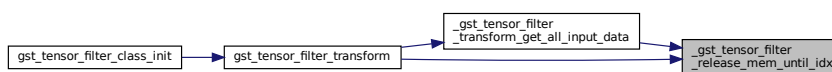
9.104.4.2 `_gst_tensor_filter_release_mem_until_idx()`

```
static void _gst_tensor_filter_release_mem_until_idx (
    FilterTransformData * trans_data,
    guint end_index ) [static]
```

Internal function to release mem and unmap info with index.

Definition at line 667 of file tensor_filter.c.

Here is the caller graph for this function:



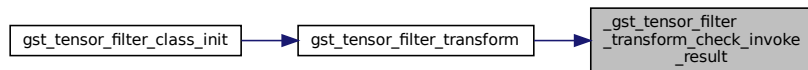
9.104.4.3 `_gst_tensor_filter_transform_check_invoke_result()`

```
static GstFlowReturn _gst_tensor_filter_transform_check_invoke_result (
    GstBaseTransform * trans,
    FilterTransformData * in_trans_data,
    FilterTransformData * out_trans_data,
    gint invoke_res ) [static]
```

Internal function to check the invoke result.

Definition at line 936 of file tensor_filter.c.

Here is the caller graph for this function:



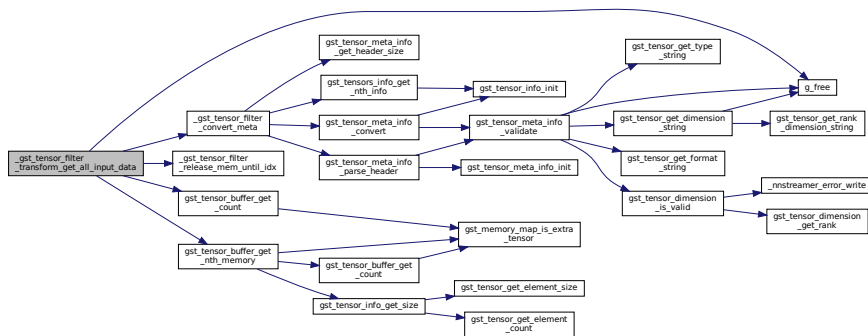
9.104.4.4 `_gst_tensor_filter_transform_get_all_input_data()`

```
static FilterTransformData* _gst_tensor_filter_transform_get_all_input_data (
    GstBaseTransform * trans,
    GstBuffer * buf ) [static]
```

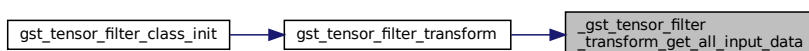
Internal function to get input tensors.

Definition at line 707 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



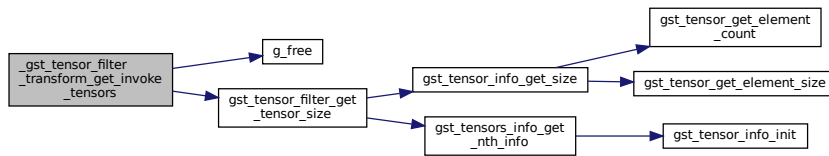
9.104.4.5 `_gst_tensor_filter_transform_get_invoke_tensors()`

```
static GstTensorMemory* _gst_tensor_filter_transform_get_invoke_tensors (
    GstBaseTransform * trans,
    FilterTransformData * trans_data ) [static]
```

Internal function to get invoke tensors.

Definition at line 754 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



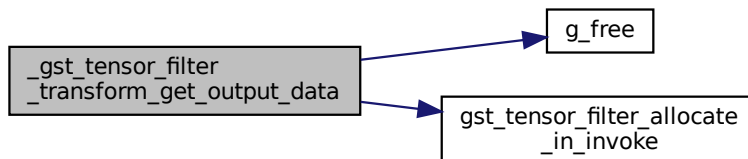
9.104.4.6 `_gst_tensor_filter_transform_get_output_data()`

```
static FilterTransformData* _gst_tensor_filter_transform_get_output_data (
    GstBaseTransform * trans ) [static]
```

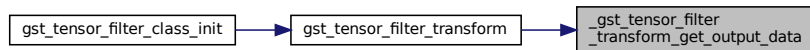
Internal function to get output tensors.

Definition at line 836 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.7 `_gst_tensor_filter_transform_prepare_output_tensors()`

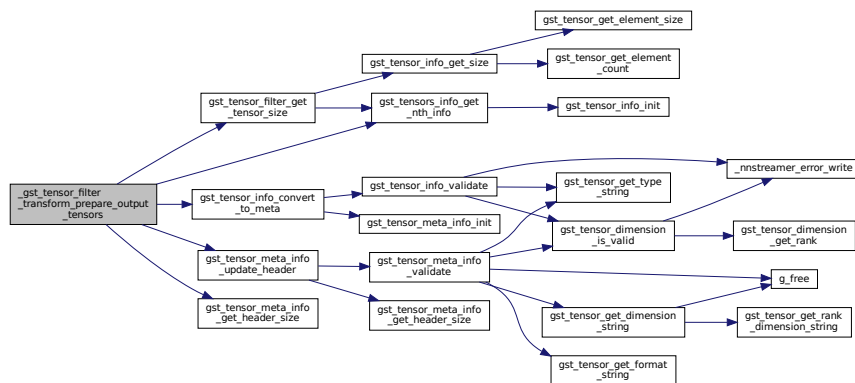
```

static GstFlowReturn _gst_tensor_filter_transform_prepare_output_tensors (
    GstBaseTransform * trans,
    FilterTransformData * trans_data ) [static]
  
```

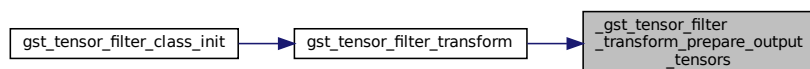
Internal function to get output tensors.

Definition at line 870 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



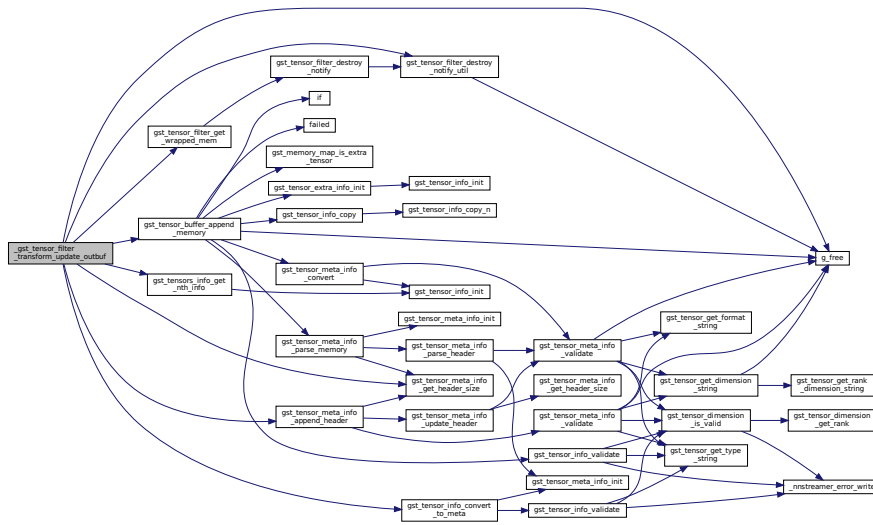
9.104.4.8 `_gst_tensor_filter_transform_update_outbuf()`

```
static void _gst_tensor_filter_transform_update_outbuf (
    GstBaseTransform * trans,
    FilterTransformData * in_trans_data,
    FilterTransformData * out_trans_data,
    GstBuffer * outbuf ) [static]
```

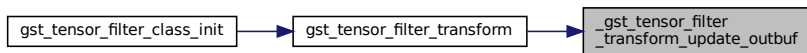
Internal function to make output buffer.

Definition at line 976 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.9 `_gst_tensor_filter_transform_validate()`

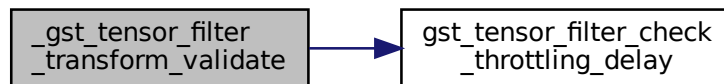
```
static GstFlowReturn _gst_tensor_filter_transform_validate (
    GstBaseTransform * trans,
    GstBuffer * inbuf,
    GstBuffer * outbuf ) [static]
```

Check input parameters for `gst_tensor_filter_transform ()`;

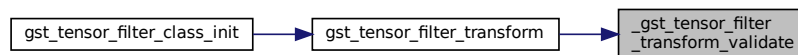
This is fatal; if framework is not configured until this stage, it means that an extension is missing or not configured. We need readable messages for non-developers

Definition at line 593 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.10 accumulate_latency()

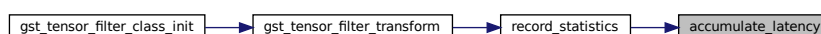
```

static void accumulate_latency (
    void * data,
    void * user_data ) [static]
  
```

Helper function to accumulate latencies.

Definition at line 397 of file tensor_filter.c.

Here is the caller graph for this function:



9.104.4.11 G_DEFINE_TYPE()

```
G_DEFINE_TYPE (
    GstTensorFilter ,
    gst_tensor_filter ,
    GST_TYPE_BASE_TRANSFORM )
```

9.104.4.12 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_filter_debug )
```

9.104.4.13 gst_tensor_filter_check_throttling_delay()

```
static gboolean gst_tensor_filter_check_throttling_delay (
    GstBaseTransform * trans,
    GstBuffer * inbuf ) [static]
```

Check throttling delay and send qos overflow event to upstream elements.

Send qos overflow event to upstream elements. Upstream elements (e.g., tensor_src, tensor_converter) may handle this.

Definition at line 536 of file tensor_filter.c.

Here is the caller graph for this function:



9.104.4.15 `gst_tensor_filter_configure_tensor()`

```
static gboolean gst_tensor_filter_configure_tensor (
    GstTensorFilter * self,
    const GstCaps * incaps ) [static]
```

Configure input and output tensor info from incaps.

Parameters

<i>self</i>	"this" pointer
<i>incaps</i>	received caps for sink pad

Returns

TRUE if fully configured

GstTensorFilter has to parse the tensor dimension and type from NN model.

1. Call functions `getInputDimension` and `getOutputDimension` to get the dimension and type.
2. If these functions are not defined, call `setInputDimension` with parsed info from caps.
3. If set-prop configured dimension, verify the dimension with fw callbacks.

Check configuration from caps. If true, fully configured tensor info from caps.

if set-property called and already has info, verify it!

If incoming tensor is flexible, we cannot validate tensor info here. Need to compare buffer size in `transform()`.

Todo We do not support this (flexible tensor for flexible input model). Cap-negotiation of the current tensor-filter requires either side of "model / set-property" or "incoming gstcaps" to be static/explicit. Ideally, this should support flexible tensor for flexible input model, leaving the negotiation to other elements, but we didn't implement it yet.

call `setInputDimension` if output tensor is not configured

if set-property called and already has info, verify it!

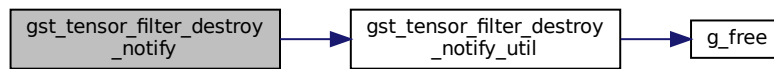
Todo framerate of output tensors How can we update the framerate? `GstTensorFilter` cannot assure the framerate. Simply set the framerate of out-tensor from incaps.

already configured, compare to old.

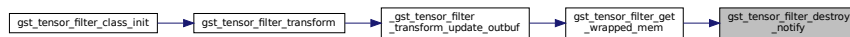
Definition at line 1207 of file `tensor_filter.c`.

Definition at line 357 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



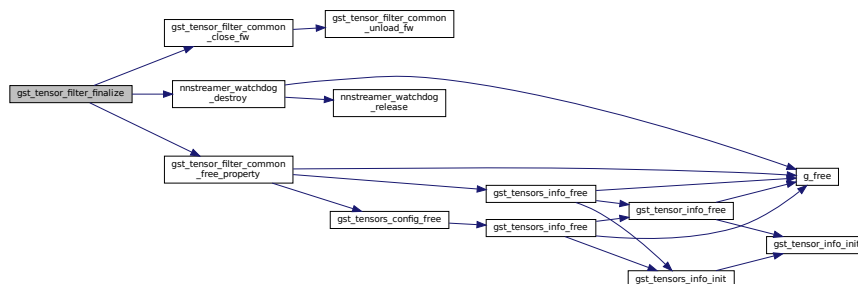
9.104.4.17 `gst_tensor_filter_finalize()`

```
static void gst_tensor_filter_finalize (
    GObject * object ) [static]
```

Function to finalize instance.

Definition at line 249 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.18 `gst_tensor_filter_fixate_caps()`

```
static GstCaps * gst_tensor_filter_fixate_caps (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    GstCaps * othercaps ) [static]
```

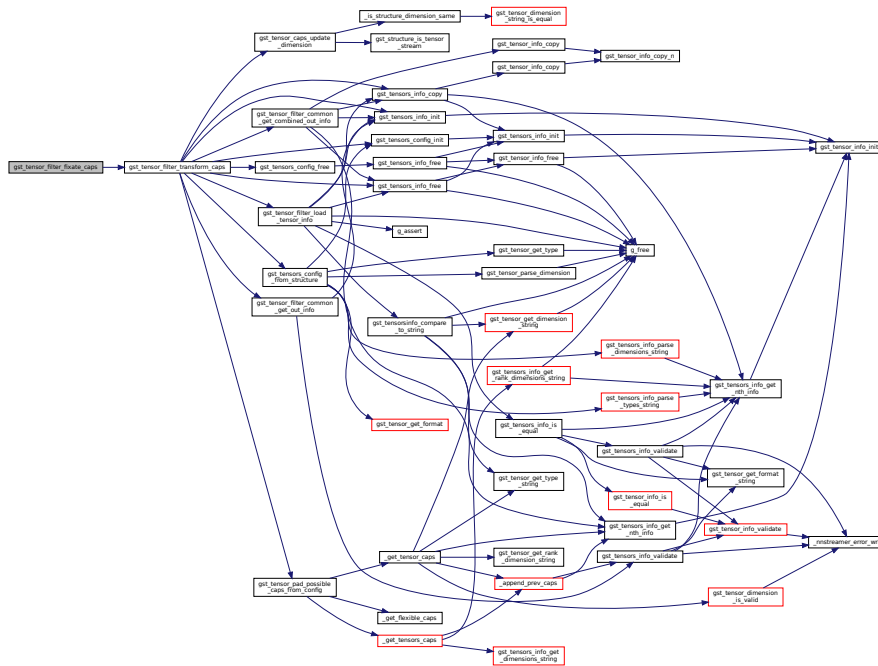
fixate caps. required vmethod of GstBaseTransform.

Removes no-used-variable warning for priv in when DBG is set

To get the out-caps, GstTensorFilter has to parse tensor info from NN model.

Definition at line 1528 of file `tensor_filter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



Parameters

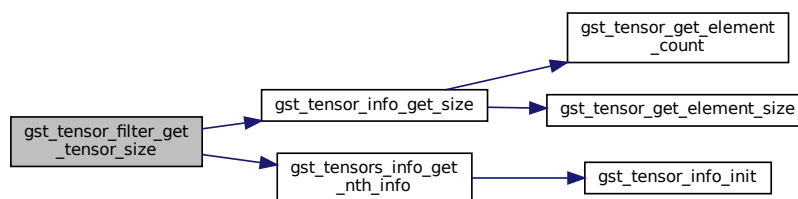
<i>self</i>	"this" pointer
<i>index</i>	index of tensors

Returns

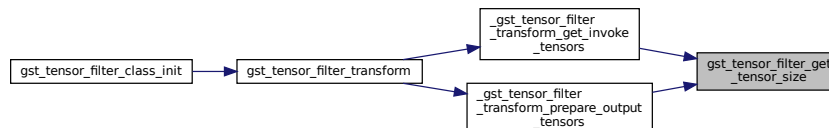
tensor buffer size

Definition at line 277 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.21 `gst_tensor_filter_get_wrapped_mem()`

```

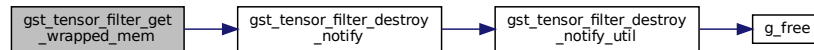
static GstMemory* gst_tensor_filter_get_wrapped_mem (
    GstTensorFilter * self,
    gpointer data,
    gsize size ) [static]
  
```

Allocate new memory block from given data.

tensor-filter should send event to sub-plugin when memory is freed.

Definition at line 372 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



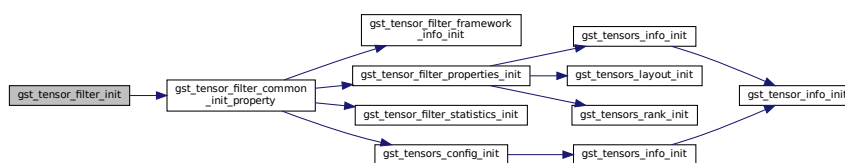
9.104.4.22 gst_tensor_filter_init()

```
static void gst_tensor_filter_init (
    GstTensorFilter * self ) [static]
```

initialize the new element instantiate pads and add them to element set pad callback functions initialize instance structure

Definition at line 234 of file tensor_filter.c.

Here is the call graph for this function:



9.104.4.23 gst_tensor_filter_query()

```
static gboolean gst_tensor_filter_query (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstQuery * query ) [static]
```

query handling, optional vmethod of GstBaseTransform.

Definition at line 1623 of file tensor_filter.c.

Here is the caller graph for this function:



9.104.4.24 `gst_tensor_filter_set_caps()`

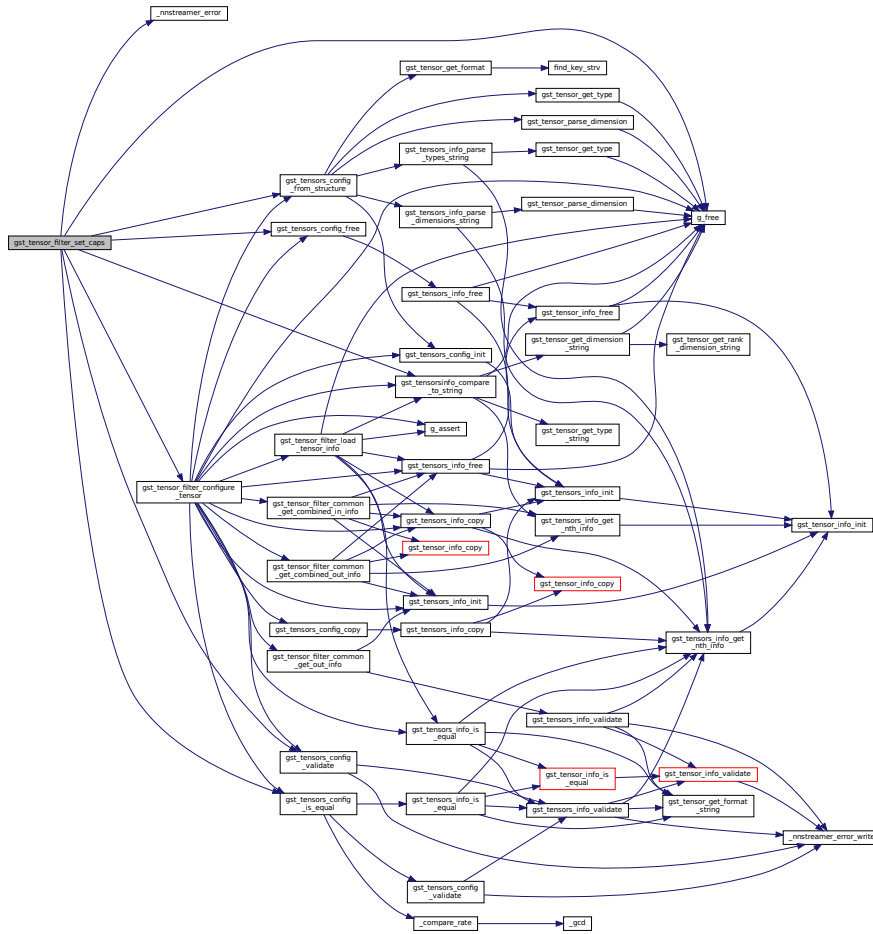
```
static gboolean gst_tensor_filter_set_caps (  
    GstBaseTransform * trans,  
    GstCaps * incaps,  
    GstCaps * outcaps ) [static]
```

set caps. required vmethod of GstBaseTransform.

compare output tensor

Definition at line 1563 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



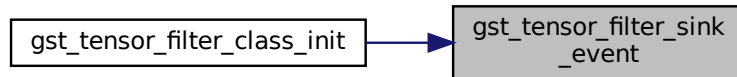
9.104.4.25 gst_tensor_filter_set_property()

```
static void gst_tensor_filter_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```


other events are handled in the default event handler

Definition at line 1723 of file tensor_filter.c.

Here is the caller graph for this function:



9.104.4.27 `gst_tensor_filter_src_event()`

```

static gboolean gst_tensor_filter_src_event (
    GstBaseTransform * trans,
    GstEvent * event ) [static]
  
```

Event handler for src pad of tensor filter.

Parameters

<i>trans</i>	"this" pointer
<i>event</i>	a passed event object

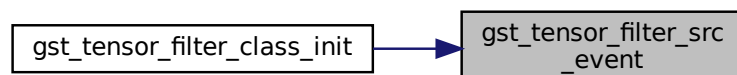
Returns

TRUE if there is no error.

other events are handled in the default event handler

Definition at line 1764 of file tensor_filter.c.

Here is the caller graph for this function:



9.104.4.28 `gst_tensor_filter_start()`

```
static gboolean gst_tensor_filter_start (
    GstBaseTransform * trans ) [static]
```

Called when the element starts processing. optional vmethod of BaseTransform.

Parameters

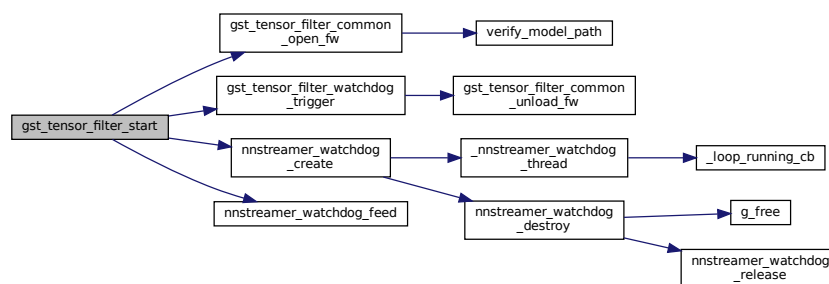
<i>trans</i>	"this" pointer
--------------	----------------

Returns

TRUE if there is no error.

Definition at line 1802 of file `tensor_filter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.29 `gst_tensor_filter_stop()`

```
static gboolean gst_tensor_filter_stop (
    GstBaseTransform * trans ) [static]
```

Called when the element stops processing. optional vmethod of BaseTransform.

Parameters

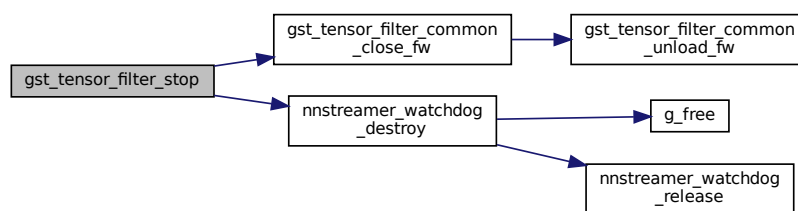
<i>trans</i>	"this" pointer
--------------	----------------

Returns

TRUE if there is no error.

Definition at line 1836 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.30 gst_tensor_filter_transform()

```

static GstFlowReturn gst_tensor_filter_transform (
    GstBaseTransform * trans,
    GstBuffer * inbuf,
    GstBuffer * outbuf ) [static]
  
```

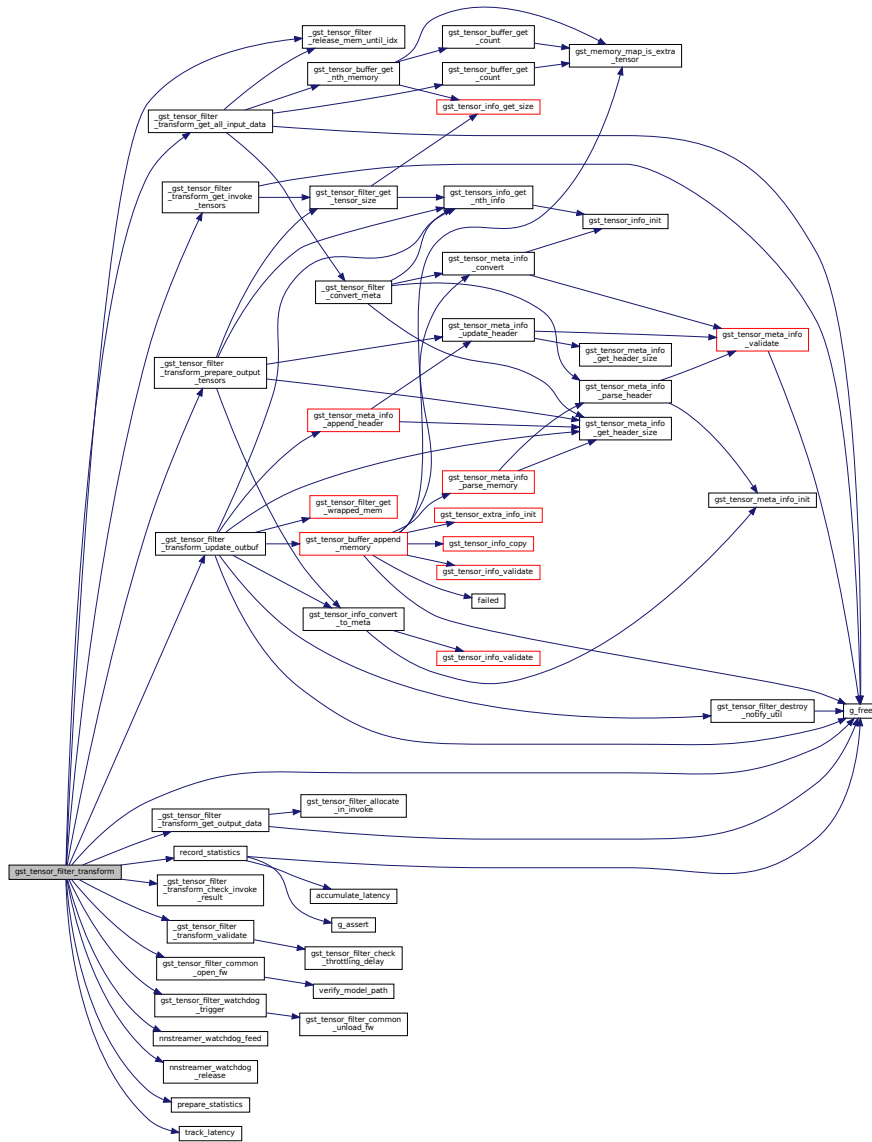
non-ip transform. required vmethod of GstBaseTransform.

Reset suspend timeout

Set suspend timeout

Definition at line 1097 of file tensor_filter.c.

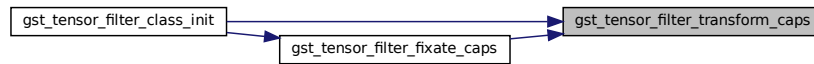
Here is the call graph for this function:



Here is the caller graph for this function:



Here is the caller graph for this function:



9.104.4.32 `gst_tensor_filter_transform_size()`

```

static gboolean gst_tensor_filter_transform_size (
    GstBaseTransform * trans,
    GstPadDirection direction,
    GstCaps * caps,
    gsize size,
    GstCaps * othercaps,
    gsize * othersize ) [static]
  
```

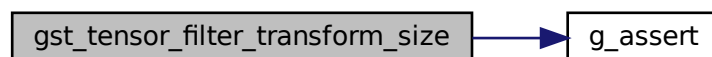
Tell the framework the required size of buffer based on the info of the other side pad. optional vmethod of Base↔ Transform.

We cannot directly get the value from size value, we need to review the pad-caps. This is called when non-ip mode is used. Internal Logic Error. Cannot proceed without configured pipeline

Consider multi-tensors. Set each memory block in transform()

Definition at line 1694 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.104.4.33 `gst_tensor_filter_watchdog_trigger()`

```
static gboolean gst_tensor_filter_watchdog_trigger (  
    gpointer ptr ) [static]
```

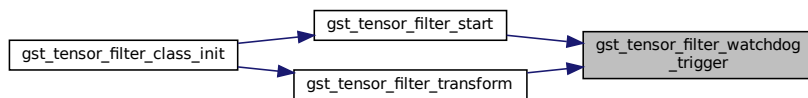
Called when there is no input within suspend time specified by the user.

Definition at line 1083 of file `tensor_filter.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



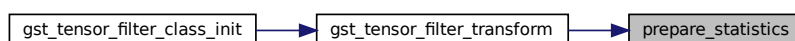
9.104.4.34 `prepare_statistics()`

```
static void prepare_statistics (  
    GstTensorFilterPrivate * priv ) [static]
```

Prepare statistics for performance profiling (e.g, latency, throughput)

Definition at line 388 of file `tensor_filter.c`.

Here is the caller graph for this function:



9.104.4.35 record_statistics()

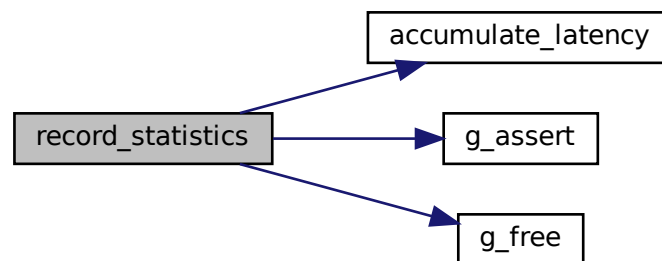
```
static void record_statistics (
    GstTensorFilterPrivate * priv ) [static]
```

Record statistics for performance profiling (e.g, latency, throughput)

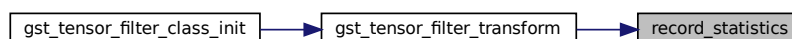
statistics values are monotonously increasing. to avoid potential overflow, let's cache old values and subtract them from the statistics if some threshold is exceeded.

Definition at line 412 of file tensor_filter.c.

Here is the call graph for this function:



Here is the caller graph for this function:



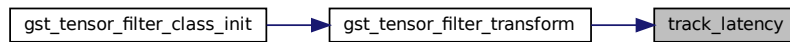
9.104.4.36 track_latency()

```
static void track_latency (
    GstTensorFilter * self ) [static]
```

Track estimated latency and notify pipeline when it changes. Latency estimates may be a bit jittery. On the principle we want to inform pipeline with the latency from longest inference. However, first inference may take much longer, or model filter configuration may change. Therefore any change of more than 10% (arbitrary value) to a lower latency is also reported to pipeline. Notification is done sending LATENCY message to bus. Upon receipt, application will initiate a pipeline latency probe via LATENCY query.

Definition at line 503 of file tensor_filter.c.

Here is the caller graph for this function:



9.104.5 Variable Documentation

9.104.5.1 sink_factory

```
GstStaticPadTemplate sink_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",  
    GST_PAD_SINK,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the inputs.

Definition at line 92 of file `tensor_filter.c`.

9.104.5.2 src_factory

```
GstStaticPadTemplate src_factory [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",  
    GST_PAD_SRC,  
    GST_PAD_ALWAYS,  
    GST_STATIC_CAPS (CAPS_STRING))
```

The capabilities of the outputs.

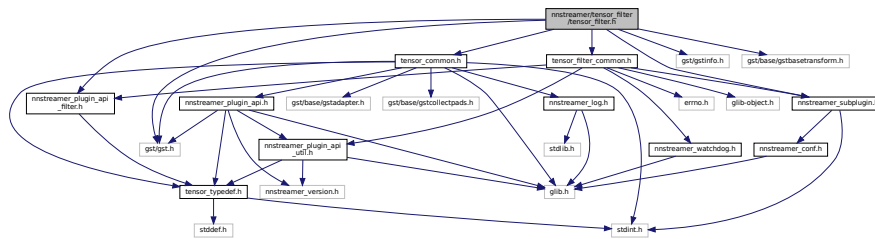
Definition at line 100 of file `tensor_filter.c`.

9.105 nnstreamer/tensor_filter/tensor_filter.h File Reference

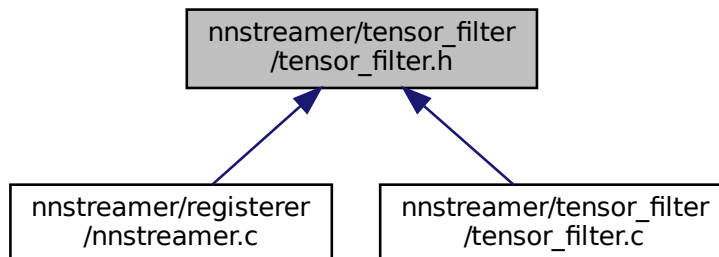
GStreamer plugin to use general neural network frameworks as filters.

```
#include <gst/gst.h>
#include <gst/gstinfo.h>
#include <gst/base/gstbasetransform.h>
#include "tensor_common.h"
#include "nnstreamer_subplugin.h"
#include "nnstreamer_plugin_api_filter.h"
#include "tensor_filter_common.h"
```

Include dependency graph for tensor_filter.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorFilter](#)
Internal data structure for tensor_filter instances.
- struct [_GstTensorFilterClass](#)
GstTensorFilterClass inherits GstBaseTransformClass.

Macros

- #define [GST_TYPE_TENSOR_FILTER](#) ([gst_tensor_filter_get_type\(\)](#))
- #define [GST_TENSOR_FILTER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_FILTER, GstTensorFilter\)](#))
- #define [GST_TENSOR_FILTER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_FILTER, GstTensorFilterClass\)](#))
- #define [GST_IS_TENSOR_FILTER\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_FILTER\)](#))
- #define [GST_IS_TENSOR_FILTER_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_FILTER\)](#))
- #define [GST_TENSOR_FILTER_CAST\(obj\)](#) ([\(\(GstTensorFilter *\) \(obj\)\)](#))

Typedefs

- typedef struct [_GstTensorFilter](#) [GstTensorFilter](#)
- typedef struct [_GstTensorFilterClass](#) [GstTensorFilterClass](#)

Functions

- GType [gst_tensor_filter_get_type](#) (void)
Get Type function required for gst elements.

9.105.1 Detailed Description

GStreamer plugin to use general neural network frameworks as filters.

GStreamer Tensor_Filter Copyright (C) 2005 Thomas Vander Stichele thomas@apestaart.org Copyright (C) 2005 Ronald S. Bultje rbultje@ronald.bitfreak.net Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

24 May 2018

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

Todo TBD: Should we disable "in-place" mode? (what if output size > input size?)

9.105.2 Macro Definition Documentation

9.105.2.1 GST_IS_TENSOR_FILTER

```
#define GST_IS_TENSOR_FILTER(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_FILTER))
```

Definition at line 49 of file tensor_filter.h.

9.105.2.2 GST_IS_TENSOR_FILTER_CLASS

```
#define GST_IS_TENSOR_FILTER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_FILTER))
```

Definition at line 51 of file tensor_filter.h.

9.105.2.3 GST_TENSOR_FILTER

```
#define GST_TENSOR_FILTER(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_FILTER, GstTensorFilter))
```

Definition at line 45 of file tensor_filter.h.

9.105.2.4 GST_TENSOR_FILTER_CAST

```
#define GST_TENSOR_FILTER_CAST(  
    obj ) ((GstTensorFilter *) (obj))
```

Definition at line 53 of file tensor_filter.h.

9.105.2.5 GST_TENSOR_FILTER_CLASS

```
#define GST_TENSOR_FILTER_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_FILTER, GstTensorFilterClass))
```

Definition at line 47 of file tensor_filter.h.

9.105.2.6 GST_TYPE_TENSOR_FILTER

```
#define GST_TYPE_TENSOR_FILTER (gst_tensor_filter_get_type())
```

Definition at line 43 of file tensor_filter.h.

9.105.3 Typedef Documentation

9.105.3.1 GstTensorFilter

```
typedef struct _GstTensorFilter GstTensorFilter
```

Definition at line 55 of file tensor_filter.h.

9.105.3.2 GstTensorFilterClass

```
typedef struct _GstTensorFilterClass GstTensorFilterClass
```

Definition at line 56 of file tensor_filter.h.

9.105.4 Function Documentation

9.105.4.1 gst_tensor_filter_get_type()

```
GType gst_tensor_filter_get_type (
    void )
```

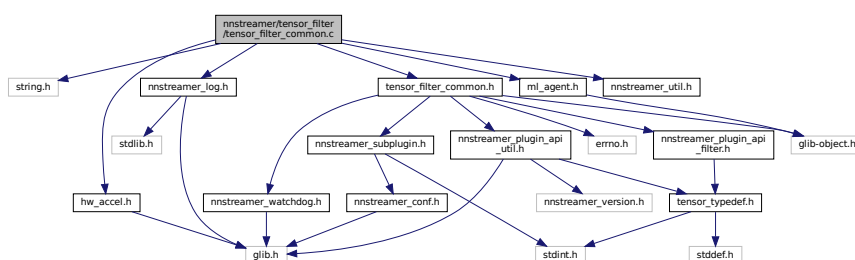
Get Type function required for gst elements.

9.106 nnstreamer/tensor_filter/tensor_filter_common.c File Reference

Common functions for various tensor_filters.

```
#include <string.h>
#include <hw_accel.h>
#include <ml_agent.h>
#include <nnstreamer_log.h>
#include <nnstreamer_util.h>
#include "tensor_filter_common.h"
```

Include dependency graph for tensor_filter_common.c:



Macros

- #define `silent_debug_info`(i, msg)
- #define `REGEX_ACCL_ELEM_START` "("
- #define `REGEX_ACCL_ELEM_PREFIX` "(?<!!)"
- #define `REGEX_ACCL_ELEM_SUFFIX` ""
- #define `REGEX_ACCL_ELEM_DELIMITER` "|"
- #define `REGEX_ACCL_ELEM_END` ")?"
- #define `REGEX_ACCL_START` "(^(true)[:]?([{}]?("
- #define `REGEX_ACCL_PREFIX` ""
- #define `REGEX_ACCL_SUFFIX` ""
- #define `REGEX_ACCL_DELIMITER` "|"
- #define `REGEX_ACCL_END` ")*[]]?))"
- #define `g_free_const`(x) `g_free`((void*)(long)(x))
Free memory.
- #define `g_strfreev_const`(x) `g_strfreev`((void*)(long)(x))

Functions

- static GType `accl_hw_get_type` (void)
to get and register hardware accelerator backend enum
- static GList * `parse_accl_hw_all` (const gchar *accelerators, const gchar **supported_accelerators)
parse user given string to extract list of accelerators based on given regex
- static gint `_gtfc_setprop_IS_UPDATABLE` (GstTensorFilterPrivate *priv, GstTensorFilterProperties *prop, const GValue *value)
Handle "PROP_IS_UPDATABLE" for set-property.
- static gint `_gtfc_setprop_ACCELERATOR` (GstTensorFilterPrivate *priv, GstTensorFilterProperties *prop, const GValue *value)
Handle "PROP_ACCELERATOR" for set-property.
- `G_LOCK_DEFINE_STATIC` (shared_model_table)
mutex for shared model table.
- static void `gst_tensors_layout_init` (tensors_layout layout)
Initialize the tensors layout.
- static void `gst_tensors_rank_init` (unsigned int ranks[])
Initialize the tensors ranks.
- static tensor_layout `gst_tensor_parse_layout_string` (const gchar *layoutstr)
Get tensor layout from string input.
- static guint `gst_tensors_parse_layouts_string` (tensors_layout layout, const gchar *layout_string)
Parse the string of tensor layouts.
- static gchar * `gst_tensor_filter_get_rank_string` (const GstTensorFilterProperties *prop, gboolean isInput)
Get the rank string of the tensors.
- static gchar * `gst_tensor_filter_get_dimension_string` (const GstTensorFilterProperties *prop, const gboolean isInput)
Get the dimension string of tensors considering rank count.
- static gchar * `gst_tensor_filter_get_type_string` (const GstTensorFilterProperties *prop, const gboolean is↵_input)
Get the type string of tensors.
- static gchar * `gst_tensor_filter_get_name_string` (const GstTensorFilterProperties *prop, const gboolean is↵_input)
Get the name string of tensors.
- static const gchar * `gst_tensor_get_layout_string` (tensor_layout layout)
Get layout string of tensor layout.

- static gchar * [gst_tensor_filter_get_layout_string](#) (const [GstTensorFilterProperties](#) *prop, const gboolean is←
_input)
Get the string of layout of tensors.
- static gchar * [strcpy2](#) (gchar *dest, const gchar *src)
copy the string from src to destination
- static gchar * [create_regex](#) (const gchar **enum_list, const gchar **regex_utils)
create regex for the given string list and regex basic elements
- static gboolean [verify_model_path](#) (const [GstTensorFilterPrivate](#) *priv)
Verify validity of path for given model file if verify_model_path is set.
- static void [gst_tensor_filter_properties_init](#) ([GstTensorFilterProperties](#) *prop)
Initialize the GstTensorFilterProperties object.
- static void [gst_tensor_filter_framework_info_init](#) ([GstTensorFilterFrameworkInfo](#) *info)
Initialize the GstTensorFilterFrameworkInfo object.
- static void [gst_tensor_filter_statistics_init](#) ([GstTensorFilterStatistics](#) *stat)
Initialize the GstTensorFilterFrameworkInfo object.
- static gboolean [nnstreamer_filter_validate](#) (const [GstTensorFilterFramework](#) *tfsp)
Validate filter sub-plugin's data.
- int [nnstreamer_filter_probe](#) ([GstTensorFilterFramework](#) *tfsp)
Filter's sub-plugin should call this function to register itself.
- void [nnstreamer_filter_exit](#) (const char *name)
Filter's sub-plugin may call this to unregister itself.
- void [nnstreamer_filter_set_custom_property_desc](#) (const char *name, const char *prop,...)
set custom property description for tensor filter sub-plugin
- static const [GstTensorFilterFramework](#) * [nnstreamer_filter_find_best_fit](#) (const char *names)
Find sub-plugin filter given the name list.
- const [GstTensorFilterFramework](#) * [nnstreamer_filter_find](#) (const char *name)
Find filter sub-plugin with the name.
- static void [gst_tensor_filter_parse_modelpaths_string](#) ([GstTensorFilterProperties](#) *prop, const gchar *model_files)
Parse the string of model.
- gboolean [gst_tensor_filter_allocate_in_invoke](#) ([GstTensorFilterPrivate](#) *priv)
check if the allocate_in_invoke is valid for the framework
- void [gst_tensor_filter_destroy_notify_util](#) ([GstTensorFilterPrivate](#) *priv, void *data)
Free the data allocated for tensor filter output.
- gchar * [gst_tensorsinfo_compare_to_string](#) (const [GstTensorsInfo](#) *info1, const [GstTensorsInfo](#) *info2)
Printout the comparison results of two tensors as a string.
- void [gst_tensorsinfo_compare_print](#) (const [GstTensorsInfo](#) *info1, const [GstTensorsInfo](#) *info2)
Printout the comparison results of two tensors.
- void [gst_tensor_filter_install_properties](#) (GObjectClass *gobject_class)
Installs all the properties for tensor_filter.
- void [gst_tensor_filter_common_init_property](#) ([GstTensorFilterPrivate](#) *priv)
Initialize the properties for tensor-filter.
- void [gst_tensor_filter_common_free_property](#) ([GstTensorFilterPrivate](#) *priv)
Free the properties for tensor-filter.
- static void [gst_tensor_filter_parse_accelerator](#) ([GstTensorFilterPrivate](#) *priv, [GstTensorFilterProperties](#) *prop, const char *accelerators)
Parse the hardware accelerators to be used for this framework.
- static gchar * [_detect_framework_from_config](#) (const gchar *extension)
Get available framework from config.
- gchar * [gst_tensor_filter_detect_framework](#) (const gchar *const *model_files, const guint num_models, const gboolean load_conf)

Get neural network framework name from given model file. This does not guarantee the framework is available on the target device.

- static void `gst_tensor_filter_get_available_framework` (`GstTensorFilterPrivate` *priv, const char *fw_name)
automatically selecting framework for tensor filter
- static gint `_gtfc_setprop_FRAMEWORK` (`GstTensorFilterPrivate` *priv, `GstTensorFilterProperties` *prop, const `GValue` *value)
Handle "PROP_FRAMEWORK" for set-property.
- static gint `_gtfc_setprop_MODEL` (`GstTensorFilterPrivate` *priv, `GstTensorFilterProperties` *prop, const `GValue` *value)
Handle "PROP_MODEL" for set-property.
- static gint `_gtfc_setprop_DIMENSION` (`GstTensorFilterPrivate` *priv, const `GValue` *value, const gboolean is_input)
Handle "PROP_INPUT" and "PROP_OUTPUT" for set-property.
- static gint `_gtfc_setprop_TYPE` (`GstTensorFilterPrivate` *priv, const `GValue` *value, const gboolean is_input)
Handle "PROP_INPUTTYPE" and "PROP_OUTPUTTYPE" for set-property.
- static gint `_gtfc_setprop_NAME` (`GstTensorFilterPrivate` *priv, const `GValue` *value, const gboolean is_input)
Handle "PROP_INPUTNAME" and "PROP_OUTPUTNAME" for set-property.
- static gint `_gtfc_setprop_CUSTOM` (`GstTensorFilterPrivate` *priv, `GstTensorFilterProperties` *prop, const `GValue` *value)
Handle "PROP_CUSTOM" for set-property.
- static gint `_gtfc_setprop_LAYOUT` (`GstTensorFilterPrivate` *priv, const `GValue` *value, const gboolean is_input)
Handle "PROP_INPUTLAYOUT" and "PROP_OUTPUTLAYOUT" for set-property.
- static gint `_gtfc_setprop_LATENCY` (`GstTensorFilterPrivate` *priv, `GstTensorFilterProperties` *prop, const `GValue` *value)
Handle "PROP_LATENCY" for set-property.
- static gint `_gtfc_setprop_THROUGHPUT` (`GstTensorFilterPrivate` *priv, `GstTensorFilterProperties` *prop, const `GValue` *value)
Handle "PROP_THROUGHPUT" for set-property.
- static gint `_gtfc_setprop_INPUTCOMBINATION` (`GstTensorFilterPrivate` *priv, `GList` **prop_list, const `GValue` *value)
Handle "PROP_INPUTCOMBINATION" for set-property.
- static gint `_gtfc_setprop_OUTPUTCOMBINATION` (`GstTensorFilterPrivate` *priv, `GList` **prop_list1, `GList` **prop_list2, const `GValue` *value)
Handle "PROP_OUTPUTCOMBINATION" for set-property.
- static gint `_gtfc_setprop_SHARED_TENSOR_FILTER_KEY` (`GstTensorFilterProperties` *prop, const `GValue` *value)
Handle "PROP_SHARED_TENSOR_FILTER_KEY" for set-property.
- static gint `_gtfc_setprop_PROP_INVOKE_DYNAMIC` (`GstTensorFilterPrivate` *priv, const `GValue` *value)
Handle "PROP_INVOKE_DYNAMIC" for set-property.
- static gint `_gtfc_setprop_SUSPEND` (`GstTensorFilterPrivate` *priv, const `GValue` *value)
Handle "PROP_SUSPEND" for set-property.
- gboolean `gst_tensor_filter_common_set_property` (`GstTensorFilterPrivate` *priv, guint prop_id, const `GValue` *value, `GParamSpec` *pspec)
Set the properties for tensor_filter.
- static void `gst_tensor_filter_property_to_string` (`GValue` *value, `GstTensorFilterPrivate` *priv, guint prop_id)
Convert `GList` to `GValue`.
- gboolean `gst_tensor_filter_common_get_property` (`GstTensorFilterPrivate` *priv, guint prop_id, `GValue` *value, `GParamSpec` *pspec)
Get the properties for tensor_filter.
- gboolean `gst_tensor_filter_common_get_combined_in_info` (`GstTensorFilterPrivate` *priv, const `GstTensorsInfo` *in, `GstTensorsInfo` *combined)
Configure input tensor info with combi option.

- gboolean [gst_tensor_filter_common_get_combined_out_info](#) (GstTensorFilterPrivate *priv, const GstTensorsInfo *in, const GstTensorsInfo *out, GstTensorsInfo *combined)
Configure output tensor info with combi option.
- gboolean [gst_tensor_filter_common_get_out_info](#) (GstTensorFilterPrivate *priv, GstTensorsInfo *in, GstTensorsInfo *out)
Get output tensor info from NN model with given input info.
- void [gst_tensor_filter_load_tensor_info](#) (GstTensorFilterPrivate *priv)
Load tensor info from NN model. (both input and output tensor)
- void [gst_tensor_filter_common_open_fw](#) (GstTensorFilterPrivate *priv)
Open NN framework.
- void [gst_tensor_filter_common_unload_fw](#) (GstTensorFilterPrivate *priv)
Unload NN framework.
- void [gst_tensor_filter_common_close_fw](#) (GstTensorFilterPrivate *priv)
Close NN framework.
- [accl_hw_get_accl_hw_type](#) (const gchar *key)
return accl_hw type from string
- const gchar * [get_accl_hw_str](#) (const accl_hw key)
return string based on accl_hw type
- static const gchar ** [add_basic_supported_accelerators](#) (const gchar **supported_accelerators)
Added basic accelerators (auto, default) to supported accelerators.
- static const gchar ** [filter_supported_accelerators](#) (const gchar **supported_accelerators)
Filter accelerators based on the runtime system.
- static [accl_hw_parse_accl_hw_util](#) (const gchar *accelerators, const gchar **supported_accelerators, const gchar *auto_accelerator, const gchar *default_accelerator)
parse user given string to extract accelerator based on given regex
- static gint [runtime_check_supported_accelerator](#) (const gchar *accl)
Check if this accelerator can be used based on the runtime system.
- [accl_hw_parse_accl_hw_fill](#) ([parse_accl_args](#) accl_args)
parse user given string to extract accelerator based on given regex filling in optional arguments
- gboolean [gst_tensor_filter_check_hw_availability](#) (const gchar *name, const [accl_hw](#) hw, const char *custom)
Check if the given hw is supported by the framework.
- void * [nnstreamer_filter_shared_model_get](#) (void *instance, const char *key)
Get the shared model representation that is already shared and has the same key.
- void * [nnstreamer_filter_shared_model_insert_and_get](#) (void *instance, char *key, void *interpreter)
Insert the new shared model representation and get the value.
- int [nnstreamer_filter_shared_model_remove](#) (void *instance, const char *key, void(*free_callback)(void *))
Remove the instance registered at the referred list of shared model table. If referred list is empty, free_callback is executed.
- void [nnstreamer_filter_shared_model_replace](#) (void *instance, const char *key, void *new_interpreter, void(*replace_callback)(void *, void *), void(*free_callback)(void *))
Helper to reload interpreter for instances that has shared key. replace_callback is called iterating instances in referred list.

Variables

- static const gchar * [regex_accl_utils](#) []
- static const gchar * [regex_accl_elem_utils](#) []
- static GHashTable * [shared_model_table](#) = NULL

9.106.1 Detailed Description

Common functions for various `tensor_filters`.

Copyright (C) 2019 Parichay Kapoor pk.kapoor@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

28 Aug 2019

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.106.2 Macro Definition Documentation

9.106.2.1 `g_free_const`

```
#define g_free_const(  
    x ) g_free((void*)(long)(x))
```

Free memory.

Definition at line 87 of file `tensor_filter_common.c`.

9.106.2.2 `g_strfreev_const`

```
#define g_strfreev_const(  
    x ) g_strfreev((void*)(long)(x))
```

Definition at line 88 of file `tensor_filter_common.c`.

9.106.2.3 REGEX_ACCL_DELIMITER

```
#define REGEX_ACCL_DELIMITER "|"
```

Definition at line 63 of file tensor_filter_common.c.

9.106.2.4 REGEX_ACCL_ELEM_DELIMITER

```
#define REGEX_ACCL_ELEM_DELIMITER "|"
```

Definition at line 57 of file tensor_filter_common.c.

9.106.2.5 REGEX_ACCL_ELEM_END

```
#define REGEX_ACCL_ELEM_END ")?"
```

Definition at line 58 of file tensor_filter_common.c.

9.106.2.6 REGEX_ACCL_ELEM_PREFIX

```
#define REGEX_ACCL_ELEM_PREFIX "(?<!!)"
```

Definition at line 55 of file tensor_filter_common.c.

9.106.2.7 REGEX_ACCL_ELEM_START

```
#define REGEX_ACCL_ELEM_START "("
```

Basic elements to form accelerator regex forming

Definition at line 54 of file tensor_filter_common.c.

9.106.2.8 REGEX_ACCL_ELEM_SUFFIX

```
#define REGEX_ACCL_ELEM_SUFFIX ""
```

Definition at line 56 of file tensor_filter_common.c.

9.106.2.9 REGEX_ACCL_END

```
#define REGEX_ACCL_END ")*[[]]?)"
```

Definition at line 64 of file `tensor_filter_common.c`.

9.106.2.10 REGEX_ACCL_PREFIX

```
#define REGEX_ACCL_PREFIX ""
```

Definition at line 61 of file `tensor_filter_common.c`.

9.106.2.11 REGEX_ACCL_START

```
#define REGEX_ACCL_START "(^(true)[:]?([])?("
```

Definition at line 60 of file `tensor_filter_common.c`.

9.106.2.12 REGEX_ACCL_SUFFIX

```
#define REGEX_ACCL_SUFFIX ""
```

Definition at line 62 of file `tensor_filter_common.c`.

9.106.2.13 silent_debug_info

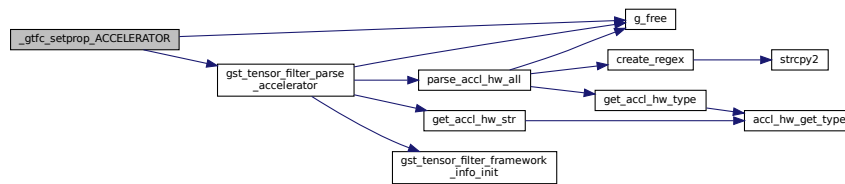
```
#define silent_debug_info(  
    i,  
    msg )
```

Value:

```
do { \
  if (DBG) { \
    quint info_idx; \
    gchar *dim_str; \
    ml_logd (msg " total %d", (i)->num_tensors); \
    for (info_idx = 0; info_idx < (i)->num_tensors; info_idx++) { \
      GstTensorInfo *nth = gst_tensors_info_get_nth_info (i, info_idx); \
      if (nth) { \
        dim_str = gst_tensor_get_dimension_string (nth->dimension); \
        ml_logd ("[%d] type=%d dim=%s", info_idx, nth->type, dim_str); \
        g_free (dim_str); \
      } \
    } \
  } \
} while (0)
```

Definition at line 35 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.3 `_gtfc_setprop_CUSTOM()`

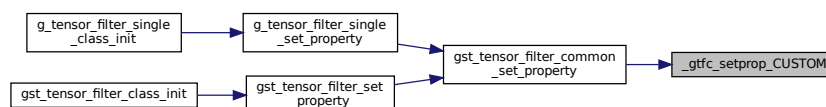
```

static gint _gtfc_setprop_CUSTOM (
    GstTensorFilterPrivate * priv,
    GstTensorFilterProperties * prop,
    const GValue * value ) [static]
  
```

Handle "PROP_CUSTOM" for set-property.

Definition at line 1627 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



9.106.3.4 `_gtfc_setprop_DIMENSION()`

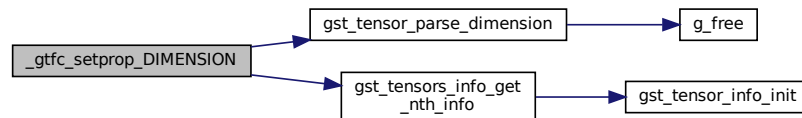
```
static gint _gtfc_setprop_DIMENSION (
    GstTensorFilterPrivate * priv,
    const GValue * value,
    const gboolean is_input ) [static]
```

Handle "PROP_INPUT" and "PROP_OUTPUT" for set-property.

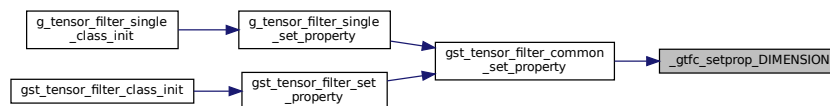
Once configured, it cannot be changed in runtime for now

Definition at line 1490 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.5 `_gtfc_setprop_FRAMEWORK()`

```
static gint _gtfc_setprop_FRAMEWORK (
    GstTensorFilterPrivate * priv,
    GstTensorFilterProperties * prop,
    const GValue * value ) [static]
```

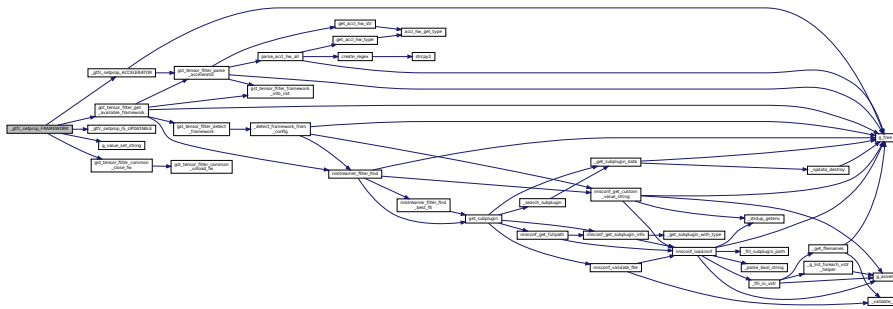
Handle "PROP_FRAMEWORK" for set-property.

set PROP_IS_UPDATABLE in case it was set before framework

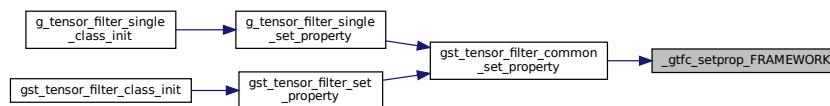
set PROP_ACCELERATOR in case it was set before framework

Definition at line 1381 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.6 _gtfc_setprop_INPUTCOMBINATION()

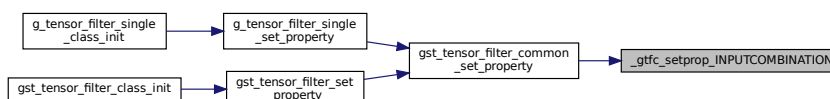
```

static gint _gtfc_setprop_INPUTCOMBINATION (
    GstTensorFilterPrivate * priv,
    GList ** prop_list,
    const GValue * value ) [static]
  
```

Handle "PROP_INPUTCOMBINATION" for set-property.

Definition at line 1843 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.7 `_gtfc_setprop_IS_UPDATABLE()`

```
static gint _gtfc_setprop_IS_UPDATABLE (
    GstTensorFilterPrivate * priv,
    GstTensorFilterProperties * prop,
    const GValue * value ) [static]
```

Handle "PROP_IS_UPDATABLE" for set-property.

Definition at line 1710 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



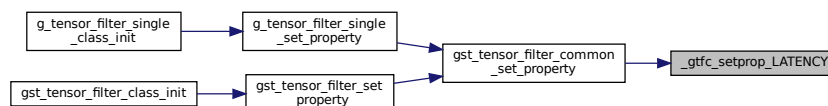
9.106.3.8 `_gtfc_setprop_LATENCY()`

```
static gint _gtfc_setprop_LATENCY (
    GstTensorFilterPrivate * priv,
    GstTensorFilterProperties * prop,
    const GValue * value ) [static]
```

Handle "PROP_LATENCY" for set-property.

Definition at line 1799 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



9.106.3.9 `_gtfc_setprop_LAYOUT()`

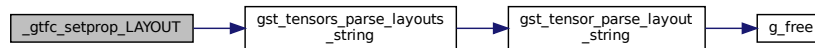
```
static gint _gtfc_setprop_LAYOUT (
    GstTensorFilterPrivate * priv,
    const GValue * value,
    const gboolean is_input ) [static]
```

Handle "PROP_INPUTLAYOUT" and "PROP_OUTPUTLAYOUT" for set-property.

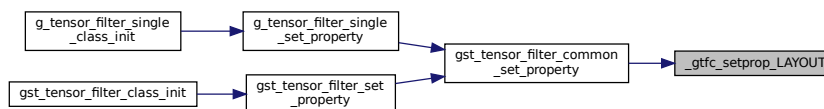
Update the properties

Definition at line 1732 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.10 `_gtfc_setprop_MODEL()`

```
static gint _gtfc_setprop_MODEL (
    GstTensorFilterPrivate * priv,
    GstTensorFilterProperties * prop,
    const GValue * value ) [static]
```

Handle "PROP_MODEL" for set-property.

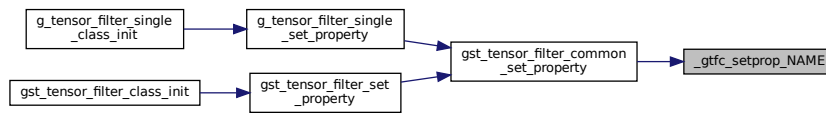
Store a copy of the original prop in case the reload fails

Reload model if FW has been already opened; In the case of reloading model files, each `priv->fw` (tensor filter for each `nnfw`) has responsibility for the verification of the path regardless of `priv->fw->verify_model_path`.

original prop is sent and not the updated prop

Definition at line 1428 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



9.106.3.12 `_gtfc_setprop_OUTPUTCOMBINATION()`

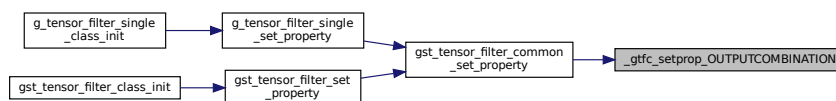
```

static gint _gtfc_setprop_OUTPUTCOMBINATION (
    GstTensorFilterPrivate * priv,
    GList ** prop_list1,
    GList ** prop_list2,
    const GValue * value ) [static]
  
```

Handle "PROP_OUTPUTCOMBINATION" for set-property.

Definition at line 1874 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.13 `_gtfc_setprop_PROP_INVOKE_DYNAMIC()`

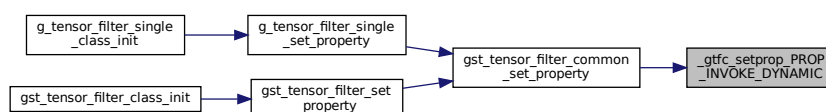
```

static gint _gtfc_setprop_PROP_INVOKE_DYNAMIC (
    GstTensorFilterPrivate * priv,
    const GValue * value ) [static]
  
```

Handle "PROP_INVOKE_DYNAMIC" for set-property.

Definition at line 1937 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.14 `_gtfc_setprop_SHARED_TENSOR_FILTER_KEY()`

```
static gint _gtfc_setprop_SHARED_TENSOR_FILTER_KEY (
    GstTensorFilterProperties * prop,
    const GValue * value ) [static]
```

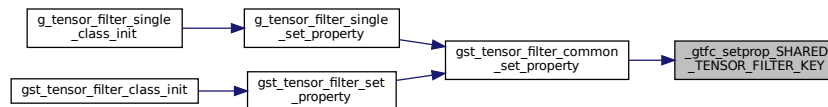
Handle "PROP_SHARED_TENSOR_FILTER_KEY" for set-property.

Definition at line 1917 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



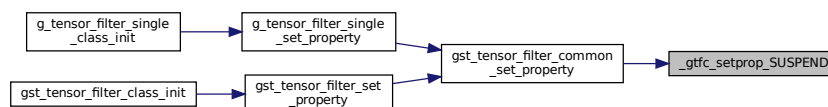
9.106.3.15 `_gtfc_setprop_SUSPEND()`

```
static gint _gtfc_setprop_SUSPEND (
    GstTensorFilterPrivate * priv,
    const GValue * value ) [static]
```

Handle "PROP_SUSPEND" for set-property.

Definition at line 1950 of file tensor_filter_common.c.

Here is the caller graph for this function:



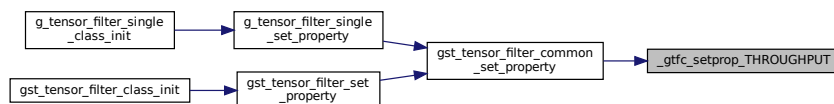
9.106.3.16 `_gtfc_setprop_THROUGHPUT()`

```
static gint _gtfc_setprop_THROUGHPUT (
    GstTensorFilterPrivate * priv,
    GstTensorFilterProperties * prop,
    const GValue * value ) [static]
```

Handle "PROP_THROUGHPUT" for set-property.

Definition at line 1821 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



9.106.3.17 `_gtfc_setprop_TYPE()`

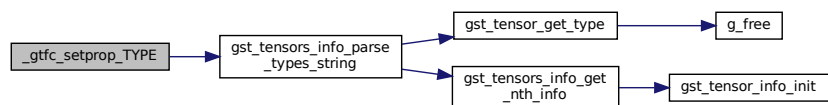
```
static gint _gtfc_setprop_TYPE (
    GstTensorFilterPrivate * priv,
    const GValue * value,
    const gboolean is_input ) [static]
```

Handle "PROP_INPUTTYPE" and "PROP_OUTPUTTYPE" for set-property.

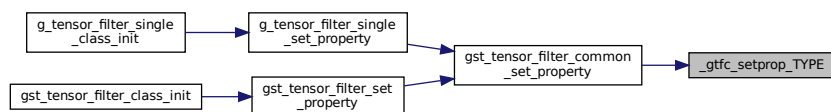
Once configured, it cannot be changed in runtime for now

Definition at line 1549 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



Parameters

in	<i>enum_list</i>	list of strings to form regex for
in	<i>regex_utils</i>	list of basic elements to form regex

Returns

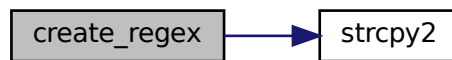
the formed regex (to be freed by the caller), NULL on error

create the regex string

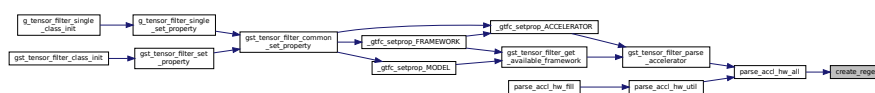
escape the special characters

Definition at line 431 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.21 filter_supported_accelerators()

```

static const gchar** filter_supported_accelerators (
    const gchar ** supported_accelerators ) [static]
  
```

Filter accelerators based on the runtime system.

Note

returned array must be freed by the caller

This filters out NEON accelerator if the system running the tensor_filter does not support NEON instructions Count number of elements for the array

Allocate the array

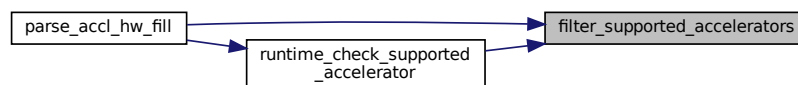
Fill the array

Definition at line 2722 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.106.3.22 G_LOCK_DEFINE_STATIC()**

```
G_LOCK_DEFINE_STATIC (
    shared_model_table )
```

mutex for shared model table.

9.106.3.23 get_accl_hw_str()

```
const gchar* get_accl_hw_str (
    const accl_hw key )
```

return string based on accl_hw type

Parameters

key	The key enum value
-----	--------------------

Returns

Corresponding string. Returns ACCL_NONE_STR if not found.

Note

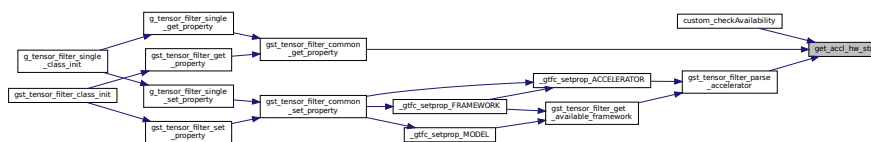
Do not free the returned char *

Definition at line 2598 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.24 get_accl_hw_type()

```
accl_hw get_accl_hw_type (
    const gchar * key )
```

return accl_hw type from string

Parameters

key	The key string value
-----	----------------------

Returns

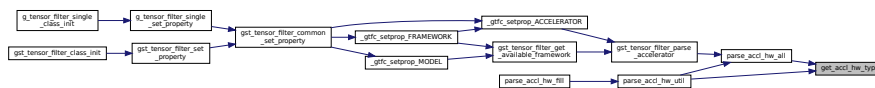
Corresponding index. Returns ACCL_NONE if not found.

Definition at line 2577 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:

**9.106.3.25 gst_tensor_filter_allocate_in_invoke()**

```

gboolean gst_tensor_filter_allocate_in_invoke (
    GstTensorFilterPrivate * priv )
  
```

check if the allocate_in_invoke is valid for the framework

Parameters

<i>in</i>	<i>priv</i>	Struct containing the properties of the object
-----------	-------------	--

Returns

TRUE if valid, FALSE on error

Definition at line 757 of file tensor_filter_common.c.

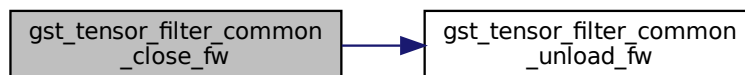
9.106.3.27 `gst_tensor_filter_common_close_fw()`

```
void gst_tensor_filter_common_close_fw (
    GstTensorFilterPrivate * priv )
```

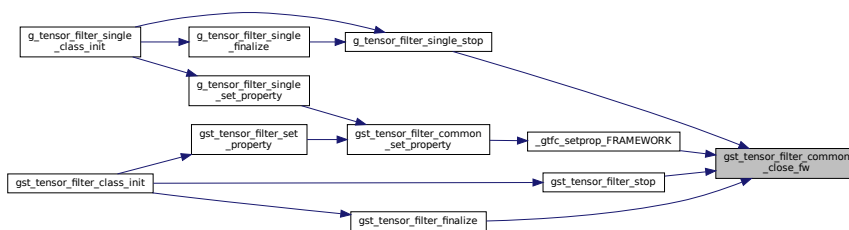
Close NN framework.

Definition at line 2560 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



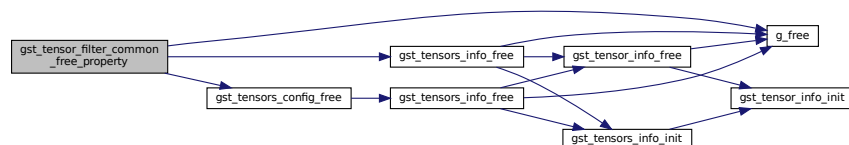
9.106.3.28 `gst_tensor_filter_common_free_property()`

```
void gst_tensor_filter_common_free_property (
    GstTensorFilterPrivate * priv )
```

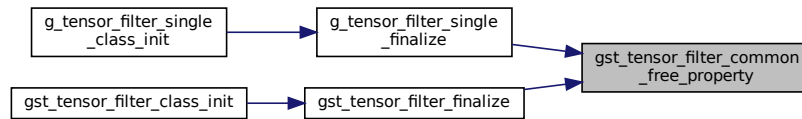
Free the properties for tensor-filter.

Definition at line 1065 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.29 gst_tensor_filter_common_get_combined_in_info()

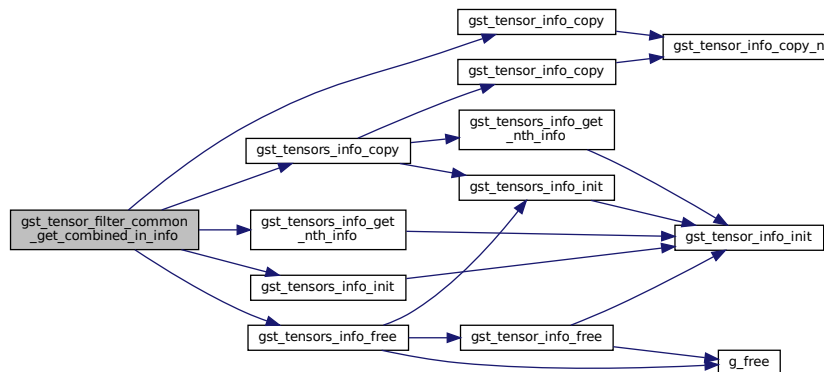
```

gboolean gst_tensor_filter_common_get_combined_in_info (
    GstTensorFilterPrivate * priv,
    const GstTensorsInfo * in,
    GstTensorsInfo * combined )
  
```

Configure input tensor info with combi option.

Definition at line 2270 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



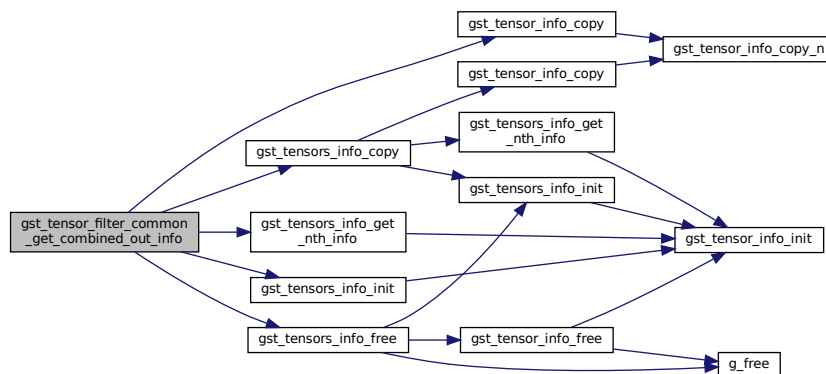
9.106.3.30 `gst_tensor_filter_common_get_combined_out_info()`

```
gboolean gst_tensor_filter_common_get_combined_out_info (
    GstTensorFilterPrivate * priv,
    const GstTensorsInfo * in,
    const GstTensorsInfo * out,
    GstTensorsInfo * combined )
```

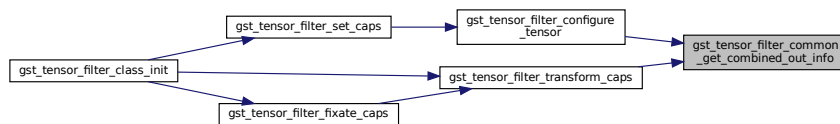
Configure output tensor info with combi option.

Definition at line 2315 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



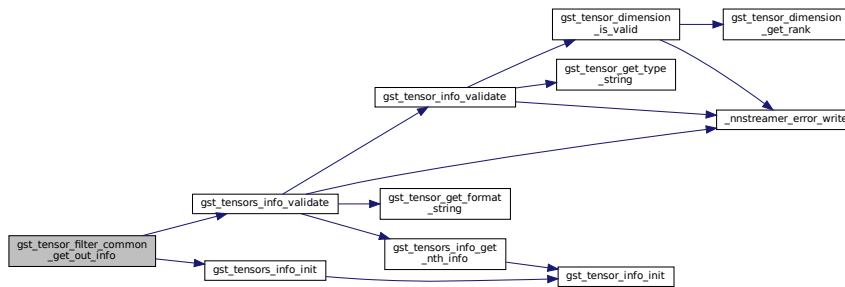
9.106.3.31 `gst_tensor_filter_common_get_out_info()`

```
gboolean gst_tensor_filter_common_get_out_info (
    GstTensorFilterPrivate * priv,
    GstTensorsInfo * in,
    GstTensorsInfo * out )
```

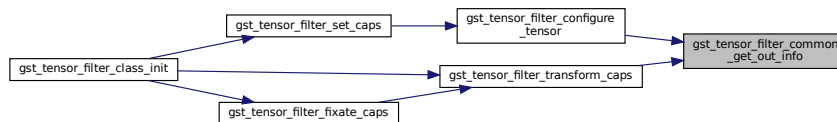
Get output tensor info from NN model with given input info.

Definition at line 2374 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.32 gst_tensor_filter_common_get_property()

```

gboolean gst_tensor_filter_common_get_property (
    GstTensorFilterPrivate * priv,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec )

```

Get the properties for tensor_filter.

Parameters

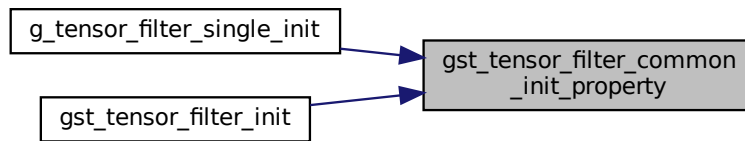
in	<i>priv</i>	Struct containing the properties of the object
in	<i>prop_id</i>	Id for the property
in	<i>value</i>	Container to return the asked property
in	<i>pspec</i>	Metadata to specify the parameter

Returns

TRUE if prop_id is value, else FALSE

Definition at line 2102 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.34 `gst_tensor_filter_common_open_fw()`

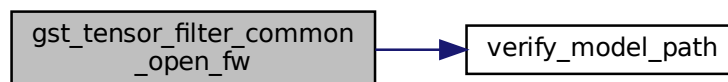
```

void gst_tensor_filter_common_open_fw (
    GstTensorFilterPrivate * priv )
  
```

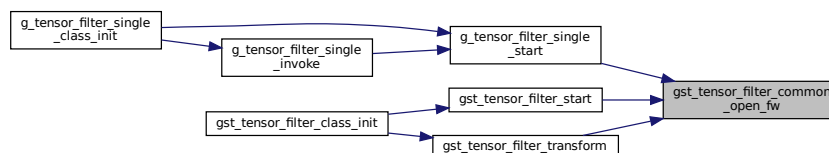
Open NN framework.

Definition at line 2491 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.35 `gst_tensor_filter_common_set_property()`

```
gboolean gst_tensor_filter_common_set_property (
    GstTensorFilterPrivate * priv,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec )
```

Set the properties for tensor_filter.

Parameters

in	<i>priv</i>	Struct containing the properties of the object
in	<i>prop_id</i>	Id for the property
in	<i>value</i>	Container to return the asked property
in	<i>pspec</i>	Metadata to specify the parameter

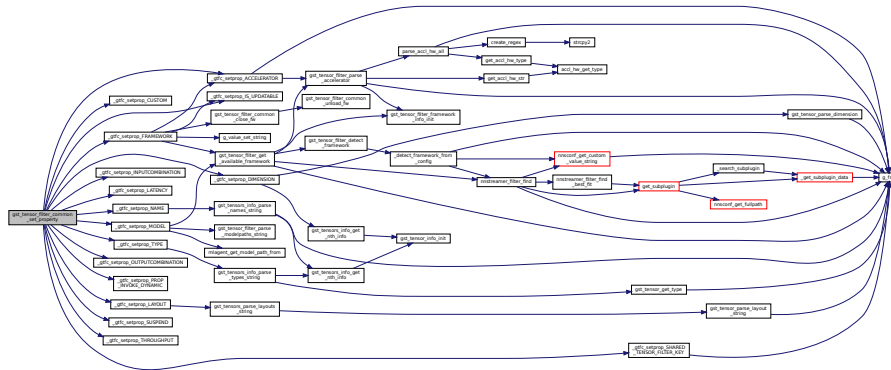
Returns

TRUE if prop_id is value, else FALSE

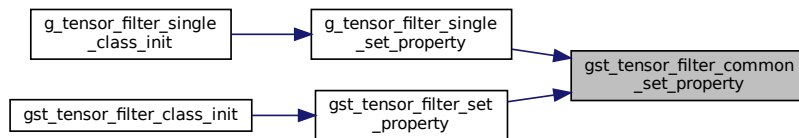
Although no one return !0 at this point, let's enable error handling.

Definition at line 1967 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



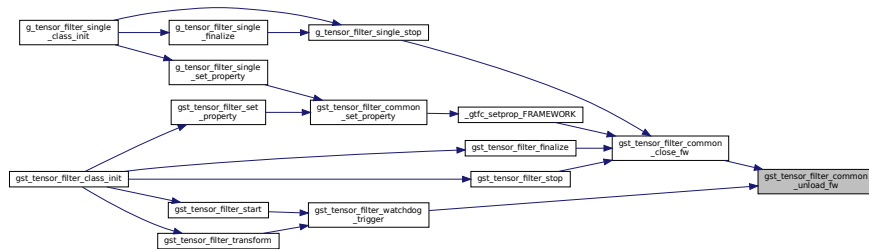
9.106.3.36 `gst_tensor_filter_common_unload_fw()`

```
void gst_tensor_filter_common_unload_fw (
    GstTensorFilterPrivate * priv )
```

Unload NN framework.

Definition at line 2545 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



9.106.3.37 `gst_tensor_filter_destroy_notify_util()`

```
void gst_tensor_filter_destroy_notify_util (
    GstTensorFilterPrivate * priv,
    void * data )
```

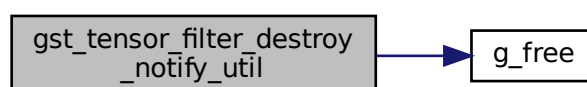
Free the data allocated for tensor filter output.

Parameters

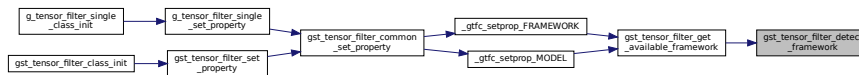
in	<i>priv</i>	Struct containing the properties of the object
in	<i>data</i>	Data to be freed

Definition at line 786 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



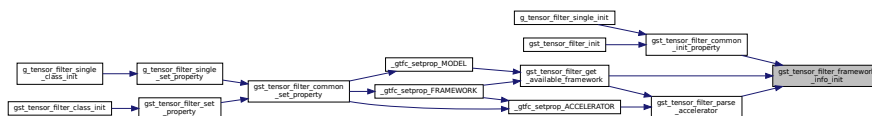
9.106.3.39 g_st_tensor_filter_framework_info_init()

```
static void g_st_tensor_filter_framework_info_init (
    GstTensorFilterFrameworkInfo * info ) [static]
```

Initialize the GstTensorFilterFrameworkInfo object.

Definition at line 528 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.40 g_st_tensor_filter_get_available_framework()

```
static void g_st_tensor_filter_get_available_framework (
    GstTensorFilterPrivate * priv,
    const char * fw_name ) [static]
```

automatically selecting framework for tensor filter

Parameters

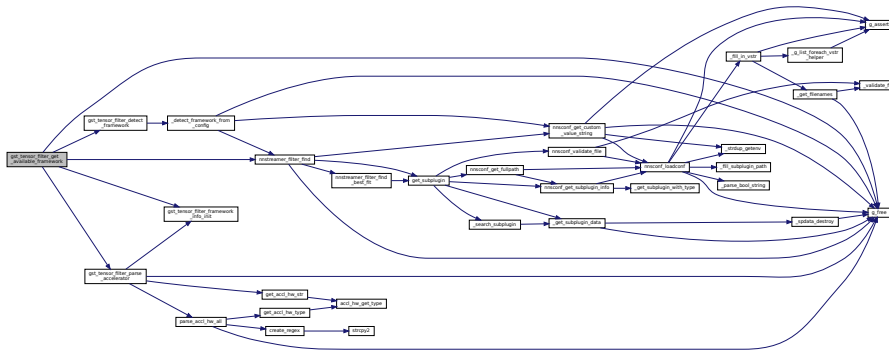
in	<i>priv</i>	Struct containing the properties of the object
in	<i>fw_name</i>	Framework name

Get framework info for v1

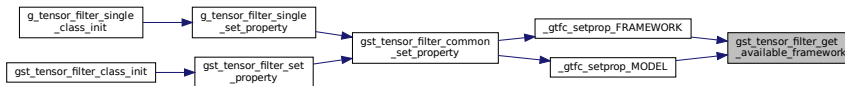
update the accelerator if already set based on v0 or v1

Definition at line 1317 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.41 gst_tensor_filter_get_dimension_string()

```
static gchar* gst_tensor_filter_get_dimension_string (
    const GstTensorFilterProperties * prop,
    const gboolean isInput ) [static]
```

Get the dimension string of tensors considering rank count.

Parameters

in	<i>prop</i>	GstTensorFilterProperties object
	<i>isInput</i>	TRUE if target is input tensors

Returns

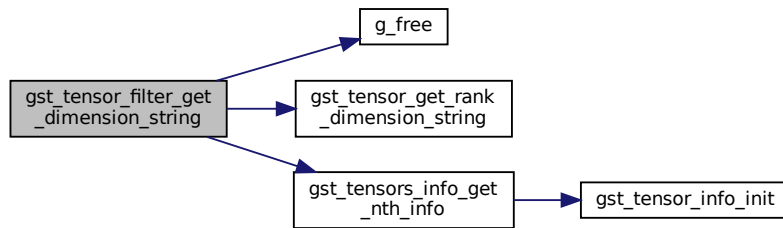
dimension string of tensors

Note

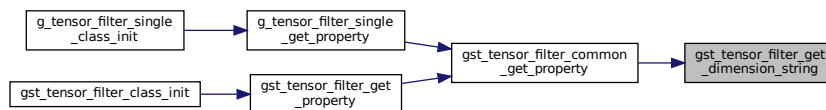
If rank count is 3, then returned string is 'd1:d2:d3'.

Definition at line 259 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.42 `gst_tensor_filter_get_layout_string()`

```

static gchar* gst_tensor_filter_get_layout_string (
    const GstTensorFilterProperties * prop,
    const gboolean is_input ) [static]
  
```

Get the string of layout of tensors.

Parameters

<i>layout</i>	layout of the tensors
---------------	-----------------------

Returns

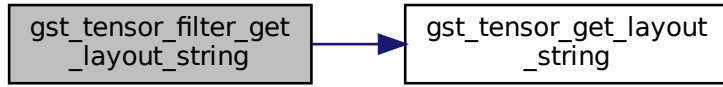
string of layouts in tensors

Note

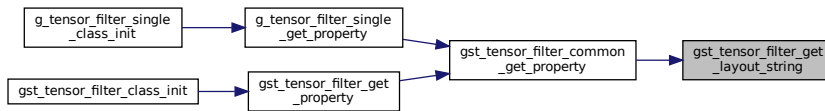
The returned value should be freed with [g_free\(\)](#)

Definition at line 372 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.43 gst_tensor_filter_get_name_string()

```

static gchar* gst_tensor_filter_get_name_string (
    const GstTensorFilterProperties * prop,
    const gboolean is_input ) [static]
  
```

Get the name string of tensors.

Parameters

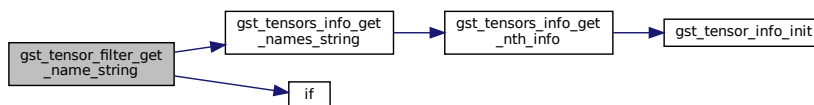
in	<i>prop</i>	GstTensorFilterProperties object
	<i>is_input</i>	TRUE if target is input tensors

Returns

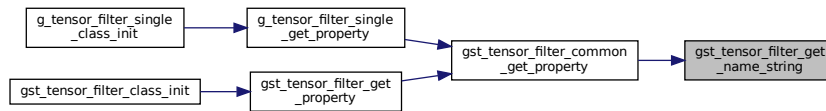
name string of tensors

Definition at line 327 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.44 gst_tensor_filter_get_rank_string()

```

static gchar* gst_tensor_filter_get_rank_string (
    const GstTensorFilterProperties * prop,
    gboolean isInput ) [static]
  
```

Get the rank string of the tensors.

Parameters

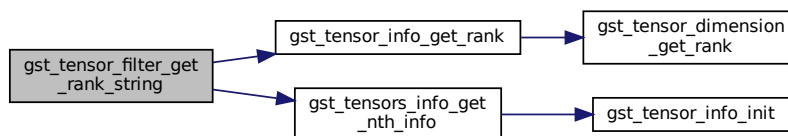
<i>prop</i>	GstTensorFilterProperties object
<i>isInput</i>	TRUE if target is input tensors

Returns

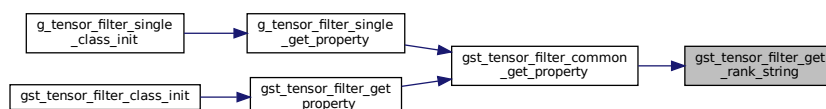
rank string of the tensors

Definition at line 211 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.45 `gst_tensor_filter_get_type_string()`

```
static gchar* gst_tensor_filter_get_type_string (
    const GstTensorFilterProperties * prop,
    const gboolean is_input ) [static]
```

Get the type string of tensors.

Parameters

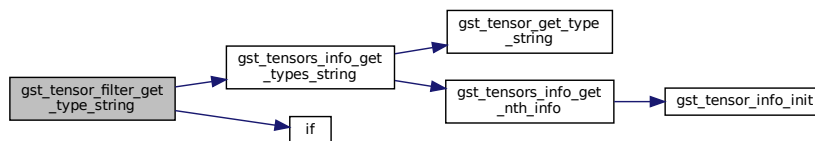
<code>in</code>	<code>prop</code>	GstTensorFilterProperties object
	<code>is_input</code>	TRUE if target is input tensors

Returns

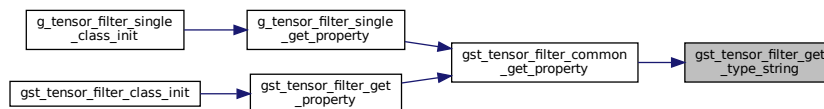
type string of tensors

Definition at line 304 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.46 `gst_tensor_filter_install_properties()`

```
void gst_tensor_filter_install_properties (
    GObjectClass * gobject_class )
```

Installs all the properties for `tensor_filter`.

Parameters

in	<i>object_class</i>	Glib object class whose properties will be set
----	---------------------	--

min

max

default: off

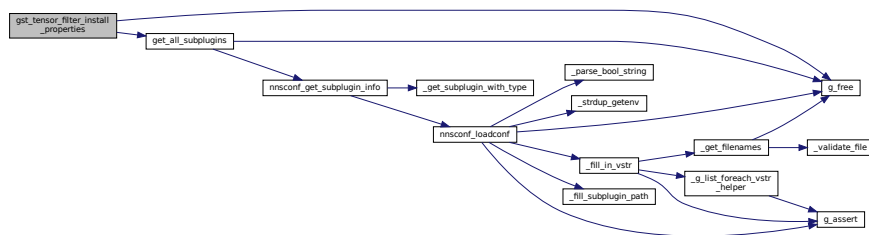
min

max

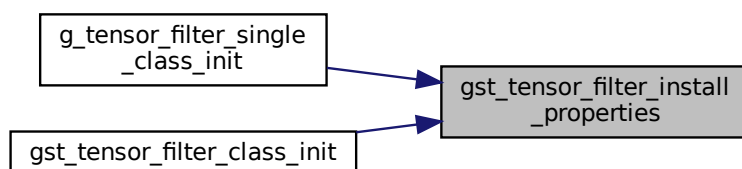
default: off

Definition at line 888 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.48 `gst_tensor_filter_parse_accelerator()`

```
static void gst_tensor_filter_parse_accelerator (
    GstTensorFilterPrivate * priv,
    GstTensorFilterProperties * prop,
    const char * accelerators ) [static]
```

Parse the hardware accelerators to be used for this framework.

Parameters

in	<i>priv</i>	Struct containing the properties of the object
in	<i>prop</i>	Struct containing the properties of the framework
in	<i>accelerators</i>	user given input for hardware accelerators

Note

The order of preference set by the user is maintained

Get h/w accelerators supported by framework (V1 only)

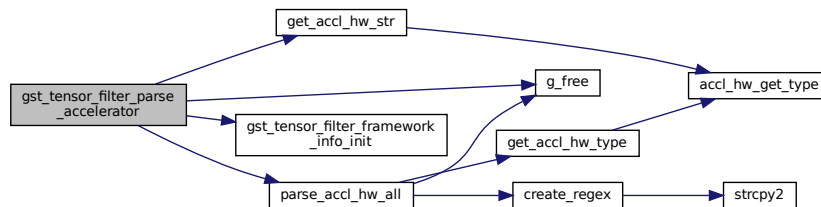
Convert the list to string format Extra 2 entries for basic accelerators : auto and default

Parse the user given h/w accelerators intersected with supported h/w

Convert the GList to regular array

Definition at line 1123 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.49 `gst_tensor_filter_parse_modelpaths_string()`

```
static void gst_tensor_filter_parse_modelpaths_string (
    GstTensorFilterProperties * prop,
    const gchar * model_files ) [static]
```

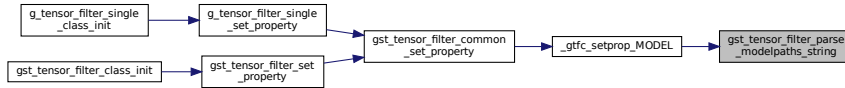
Parse the string of model.

Parameters

out	<i>prop</i>	Struct containing the properties of the object
in	<i>model_files</i>	the prediction model paths

Definition at line 728 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.50 **gst_tensor_filter_properties_init()**

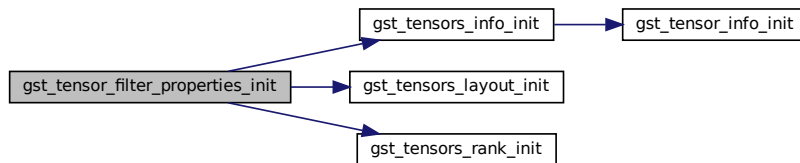
```

static void gst_tensor_filter_properties_init (
    GstTensorFilterProperties * prop ) [static]
  
```

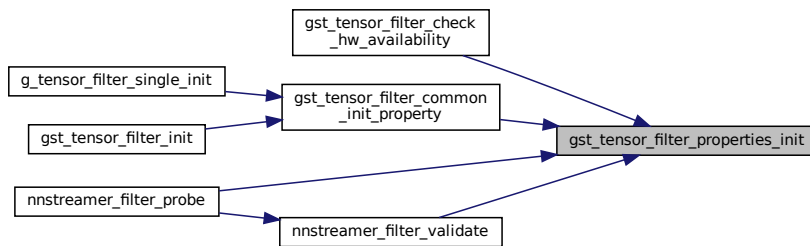
Initialize the GstTensorFilterProperties object.

Definition at line 510 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



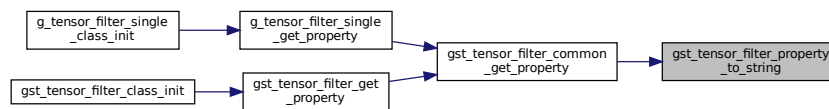
9.106.3.51 `gst_tensor_filter_property_to_string()`

```
static void gst_tensor_filter_property_to_string (
    GValue * value,
    GstTensorFilterPrivate * priv,
    guint prop_id ) [static]
```

Convert GList to GValue.

Definition at line 2063 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



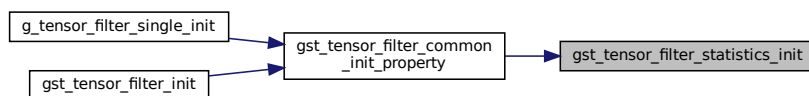
9.106.3.52 `gst_tensor_filter_statistics_init()`

```
static void gst_tensor_filter_statistics_init (
    GstTensorFilterStatistics * stat ) [static]
```

Initialize the `GstTensorFilterFrameworkInfo` object.

Definition at line 545 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



9.106.3.53 `gst_tensor_get_layout_string()`

```
static const gchar* gst_tensor_get_layout_string (
    tensor_layout layout ) [static]
```

Get layout string of tensor layout.

Parameters

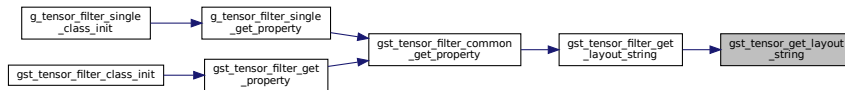
<i>layout</i>	layout of the tensor
---------------	----------------------

Returns

string of layout in tensor

Definition at line 349 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.54 **gst_tensor_parse_layout_string()**

```
static tensor_layout gst_tensor_parse_layout_string (
    const gchar * layoutstr ) [static]
```

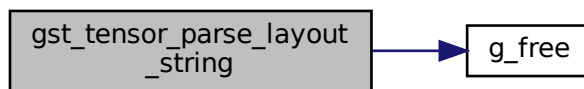
Get tensor layout from string input.

Returns

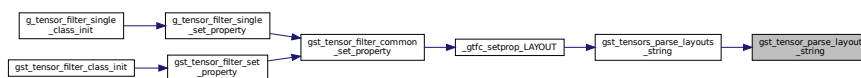
Corresponding tensor_layout.

Definition at line 134 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



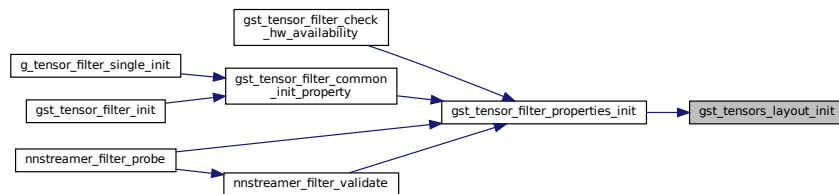
9.106.3.55 `gst_tensors_layout_init()`

```
static void gst_tensors_layout_init (
    tensors_layout layout ) [static]
```

Initialize the tensors layout.

Definition at line 108 of file `tensor_filter_common.c`.

Here is the caller graph for this function:



9.106.3.56 `gst_tensors_parse_layouts_string()`

```
static quint gst_tensors_parse_layouts_string (
    tensors_layout layout,
    const gchar * layout_string ) [static]
```

Parse the string of tensor layouts.

Parameters

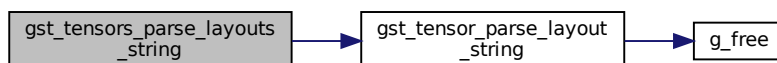
<i>layout</i>	layout of the tensors
<i>layout_string</i>	string of layout

Returns

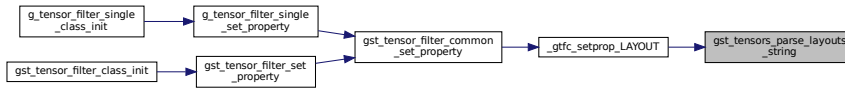
number of parsed layouts

Definition at line 173 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.57 gst_tensors_rank_init()

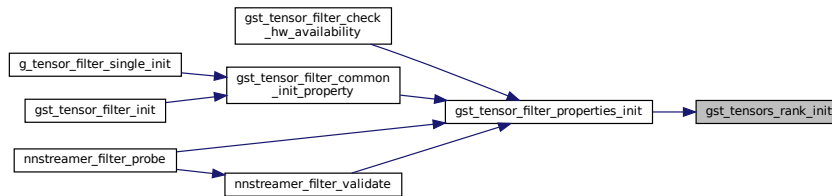
```

static void gst_tensors_rank_init (
    unsigned int ranks[] ) [static]
  
```

Initialize the tensors ranks.

Definition at line 121 of file tensor_filter_common.c.

Here is the caller graph for this function:



9.106.3.58 gst_tensorsinfo_compare_print()

```

void gst_tensorsinfo_compare_print (
    const GstTensorsInfo * info1,
    const GstTensorsInfo * info2 )
  
```

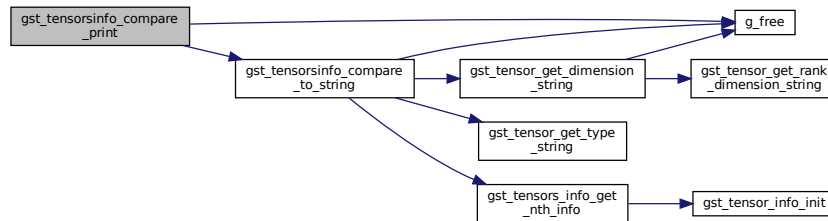
Printout the comparison results of two tensors.

Parameters

in	<i>info1</i>	The tensors to be shown on the left hand side
in	<i>info2</i>	The tensors to be shown on the right hand side

Definition at line 874 of file tensor_filter_common.c.

Here is the call graph for this function:



9.106.3.59 `gst_tensorsinfo_compare_to_string()`

```

gchar* gst_tensorsinfo_compare_to_string (
    const GstTensorsInfo * info1,
    const GstTensorsInfo * info2 )
  
```

Printout the comparison results of two tensors as a string.

Parameters

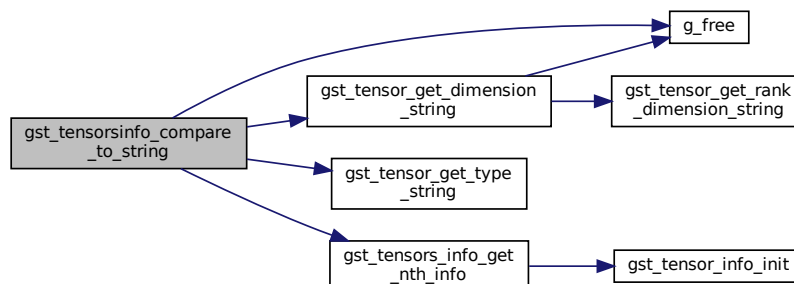
in	<i>info1</i>	The tensors to be shown on the left hand side
in	<i>info2</i>	The tensors to be shown on the right hand side

Returns

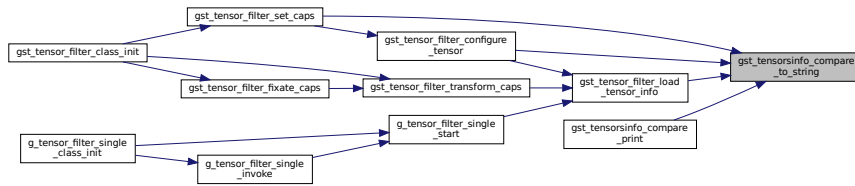
The printout string allocated. Caller should free the value.

Definition at line 811 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.60 nntstreamer_filter_exit()

```
void nntstreamer_filter_exit (
    const char * name )
```

Filter's sub-plugin may call this to unregister itself.

Parameters

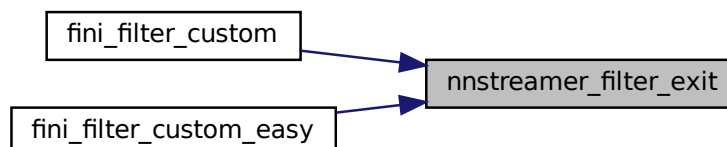
in	<i>name</i>	The name of filter sub-plugin.
----	-------------	--------------------------------

Definition at line 638 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:

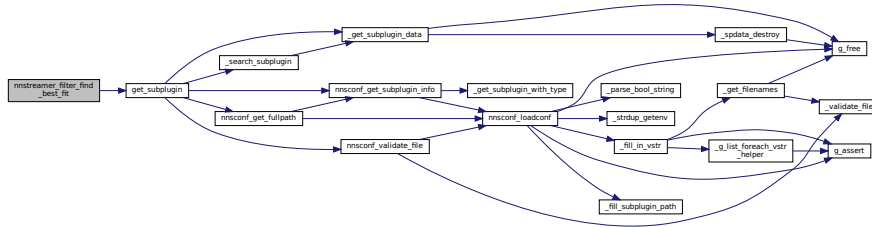


Returns

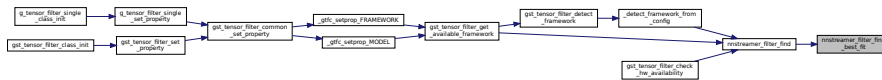
the best-fit sub-plugin object or NULL if not found.

Definition at line 664 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.63 nnstreamer_filter_probe()

```
int nnstreamer_filter_probe (
    GstTensorFilterFramework * tfsp )
```

Filter's sub-plugin should call this function to register itself.

Parameters

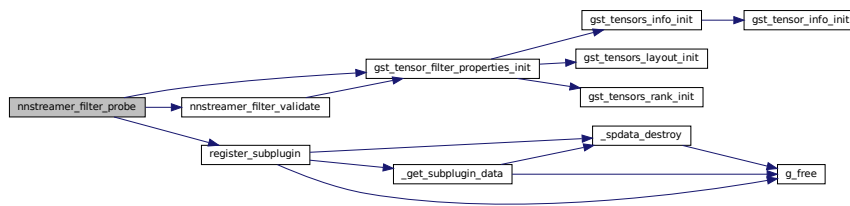
in	<i>tfsp</i>	Tensor-Filter Sub-Plugin to be registered.
----	-------------	--

Returns

TRUE if registered. FALSE is failed or duplicated.

Definition at line 611 of file tensor_filter_common.c.

Here is the call graph for this function:



9.106.3.64 nnstreamer_filter_set_custom_property_desc()

```

void nnstreamer_filter_set_custom_property_desc (
    const char * name,
    const char * prop,
    ... )
  
```

set custom property description for tensor filter sub-plugin

Definition at line 647 of file tensor_filter_common.c.

Here is the call graph for this function:



9.106.3.65 nnstreamer_filter_shared_model_get()

```

void* nnstreamer_filter_shared_model_get (
    void * instance,
    const char * key )
  
```

Get the shared model representation that is already shared and has the same key.

Parameters

in	<i>instance</i>	The instance that is sharing the model representation. It will be registered at the referred list.
in	<i>key</i>	The key to find the matched shared representation.

Returns

The model interpreter. NULL if it does not exist.

Definition at line 2993 of file tensor_filter_common.c.

9.106.3.66 nnstreamer_filter_shared_model_insert_and_get()

```
void* nnstreamer_filter_shared_model_insert_and_get (
    void * instance,
    char * key,
    void * interpreter )
```

Insert the new shared model representation and get the value.

Parameters

in	<i>instance</i>	The instance that is sharing the model representation. It will be registered at the referred list.
in	<i>key</i>	The key for shared model.
in	<i>interpreter</i>	The interpreter to be shared.

Returns

The model interpreter inserted. NULL if it is already inserted.

Internal error case. The interpreter already exists in shared table, do not insert and return null.

Definition at line 3026 of file tensor_filter_common.c.

9.106.3.67 nnstreamer_filter_shared_model_remove()

```
int nnstreamer_filter_shared_model_remove (
    void * instance,
    const char * key,
    void(*) (void *) free_callback )
```

Remove the instance registered at the referred list of shared model table. If referred list is empty, *free_callback* is executed.

Parameters

in	<i>instance</i>	The instance that should be removed from the referred list.
in	<i>key</i>	The key to find the shared model.
in	<i>free_callback</i>	The callback function to destroy the interpreter, which takes the interpreter as arg.

Returns

TRUE if the instance is removed. FALSE if failed to remove it.

Definition at line 3081 of file `tensor_filter_common.c`.

9.106.3.68 nnstreamer_filter_shared_model_replace()

```
void nnstreamer_filter_shared_model_replace (
    void * instance,
    const char * key,
    void * new_interpreter,
    void(*) (void *, void *) replace_callback,
    void(*) (void *) free_callback )
```

Helper to reload interpreter for instances that has shared key. `replace_callback` is called iterating instances in referred list.

Parameters

in	<i>instance</i>	The instance that is sharing the model representation.
in	<i>key</i>	The key to find the shared model.
in	<i>interpreter</i>	The new interpreter to replace.
in	<i>replace_callback</i>	The callback function to replace with new interpreter.
in	<i>free_callback</i>	The callback function to destroy the old interpreter.

Definition at line 3128 of file `tensor_filter_common.c`.

9.106.3.69 nnstreamer_filter_validate()

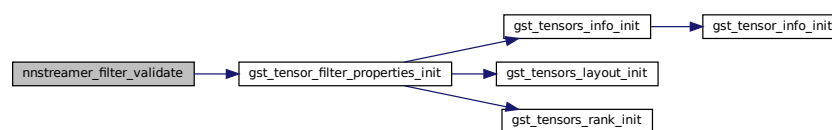
```
static gboolean nnstreamer_filter_validate (
    const GstTensorFilterFramework * tfsp ) [static]
```

Validate filter sub-plugin's data.

Mandatory callbacks are not defined

Definition at line 560 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.70 parse_accl_hw_all()

```

static GList * parse_accl_hw_all (
    const gchar * accelerators,
    const gchar ** supported_accelerators ) [static]
  
```

parse user given string to extract list of accelerators based on given regex

Parameters

in	<i>accelerators</i>	user given input
in	<i>supported_accelerators</i>	list of supported accelerators

Returns

Corresponding list of accelerators maintaining given order

Note

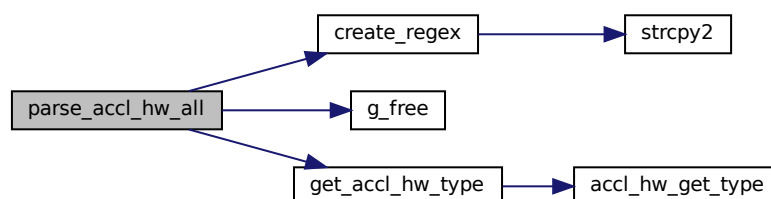
Returned list must be freed by the caller

Default to auto mode

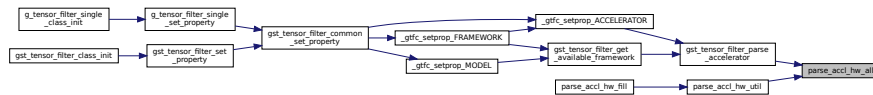
Now match each provided element and get specific accelerator

Definition at line 2620 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.71 parse_accl_hw_fill()

```
accl_hw parse_accl_hw_fill (
    parse_accl_args accl_args )
```

parse user given string to extract accelerator based on given regex filling in optional arguments

Note

The order of arguments for calling this function is:

- in_accl: user provided input accelerator string
- sup_accl: list of supported accelerator
- auto_accl: auto accelerator (optional)
- def_accl: default accelerator (optional)

remove unsupported accelerators from this list based on runtime system

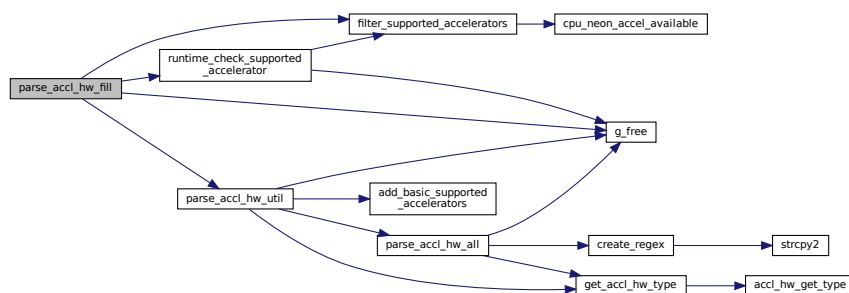
filtered supported accelerators can be empty

update default accelerator if it is not available at runtime

update auto accelerator if it is not available at runtime

Definition at line 2833 of file tensor_filter_common.c.

Here is the call graph for this function:



9.106.3.72 parse_accl_hw_util()

```
static accl_hw parse_accl_hw_util (  
    const gchar * accelerators,  
    const gchar ** supported_accelerators,  
    const gchar * auto_accelerator,  
    const gchar * default_accelerator ) [static]
```

parse user given string to extract accelerator based on given regex

Parameters

in	<i>accelerators</i>	user given input
in	<i>supported_accelerators</i>	list of supported accelerators
in	<i>auto_accelerator</i>	accelerator to use in auto case (when acceleration is enabled but specific accelerator is not provided or not matching)
in	<i>default_accelerator</i>	accelerator to use by default

Returns

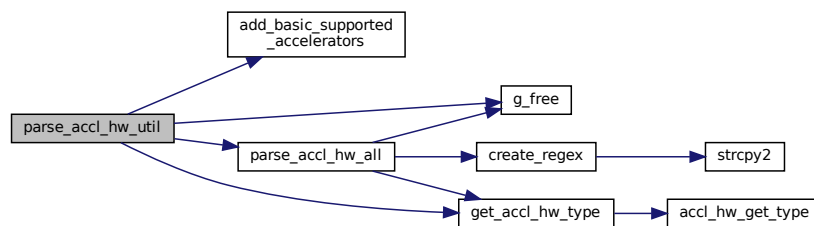
Corresponding accelerator. Returns ACCL_NONE if not found.

add auto and default accelerator into list of supported accelerators

This can be ACCL_NONE (no acceleration) or a specific accelerator

Definition at line 2763 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.73 runtime_check_supported_accelerator()

```
static gint runtime_check_supported_accelerator (
    const gchar * accl ) [static]
```

Check if this accelerator can be used based on the runtime system.

Return values

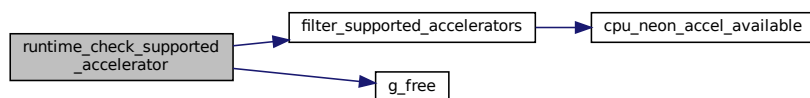
0	if filter can be used, -errno otherwise
---	---

Allocate the array

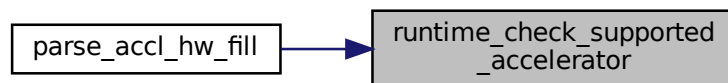
Fill the array

Definition at line 2804 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.106.3.74 strcpy2()

```

static gchar* strcpy2 (
    gchar * dest,
    const gchar * src ) [static]
  
```

copy the string from src to destination

Parameters

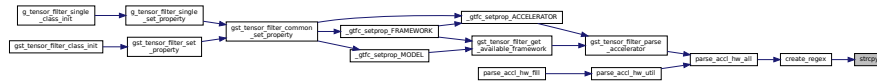
in	<i>dest</i>	destination string
in	<i>src</i>	source string

Returns

updated destination string

Definition at line 414 of file tensor_filter_common.c.

Here is the caller graph for this function:

**9.106.3.75 verify_model_path()**

```
static gboolean verify_model_path (
    const GstTensorFilterPrivate * priv ) [inline], [static]
```

Verify validity of path for given model file if verify_model_path is set.

Parameters

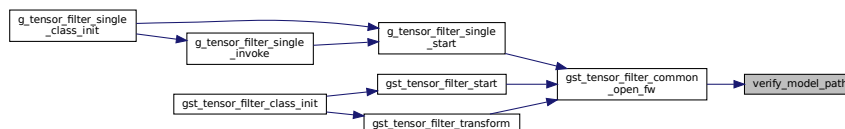
in	priv	Struct containing the common tensor-filter properties of the object
----	------	---

Returns

TRUE if there is no error

Definition at line 473 of file tensor_filter_common.c.

Here is the caller graph for this function:

**9.106.4 Variable Documentation**

9.106.4.1 regex_accl_elem_utils

```
const gchar* regex_accl_elem_utils[] [static]
```

Initial value:

```
= {  
    REGEX_ACCL_ELEM_START,  
    REGEX_ACCL_ELEM_PREFIX,  
    REGEX_ACCL_ELEM_SUFFIX,  
    REGEX_ACCL_ELEM_DELIMITER,  
    REGEX_ACCL_ELEM_END,  
    NULL  
}
```

Definition at line 75 of file tensor_filter_common.c.

9.106.4.2 regex_accl_utils

```
const gchar* regex_accl_utils[] [static]
```

Initial value:

```
= {  
    REGEX_ACCL_START,  
    REGEX_ACCL_PREFIX,  
    REGEX_ACCL_SUFFIX,  
    REGEX_ACCL_DELIMITER,  
    REGEX_ACCL_END,  
    NULL  
}
```

Definition at line 66 of file tensor_filter_common.c.

9.106.4.3 shared_model_table

```
GHashTable* shared_model_table = NULL [static]
```

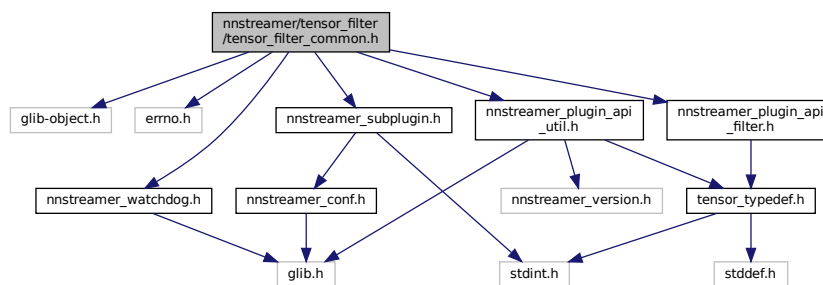
Definition at line 102 of file tensor_filter_common.c.

9.107 nnstreamer/tensor_filter/tensor_filter_common.h File Reference

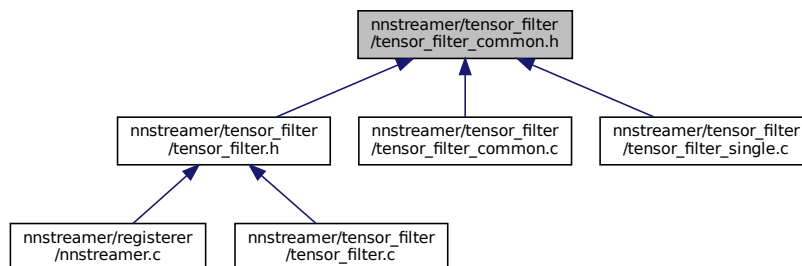
Common functions for various tensor_filters.

```
#include <glib-object.h>  
#include <errno.h>  
#include <nnstreamer_subplugin.h>  
#include <nnstreamer_plugin_api_util.h>  
#include <nnstreamer_plugin_api_filter.h>
```

```
#include "nnstreamer_watchdog.h"
Include dependency graph for tensor_filter_common.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorFilterStatistics](#)
Structure definition for tensor-filter statistics.
- struct [_GstTensorFilterCombination](#)
Structure definition for tensor-filter in/out combination.
- struct [GstTensorFilterSharedModelRepresentation](#)
Data Structure to store shared table.
- struct [_GstTensorFilterPrivate](#)
Structure definition for common tensor-filter properties.

Macros

- #define [DBG](#) (!priv->silent)
Macro for debug mode.
- #define [GST_TF_FW_VN](#)(fw, vn) (fw && [checkGstTensorFilterFrameworkVersion](#) (fw->version, vn))
- #define [GST_TF_FW_V0](#)(fw) [GST_TF_FW_VN](#) (fw, 0)
- #define [GST_TF_FW_V1](#)(fw) [GST_TF_FW_VN](#) (fw, 1)
- #define [gst_tensor_filter_v0_call](#)(priv, ret, funcname, ...)
Invoke callbacks of nn framework. Guarantees calling open for the first call.
- #define [gst_tensor_filter_v1_call](#)(priv, ret, funcname, ...)
- #define [GST_TF_FW_INVOKE_COMPAT](#)(priv, ret, in, out)
- #define [GST_TF_STAT_MAX_RECENT](#) (10)

Typedefs

- typedef struct [_GstTensorFilterStatistics](#) [GstTensorFilterStatistics](#)
Structure definition for tensor-filter statistics.
- typedef struct [_GstTensorFilterCombination](#) [GstTensorFilterCombination](#)
Structure definition for tensor-filter in/out combination.
- typedef struct [_GstTensorFilterPrivate](#) [GstTensorFilterPrivate](#)
Structure definition for common tensor-filter properties.

Enumerations

- enum {
[PROP_0](#), [PROP_SILENT](#), [PROP_FRAMEWORK](#), [PROP_MODEL](#),
[PROP_INPUT](#), [PROP_INPUTTYPE](#), [PROP_INPUTNAME](#), [PROP_INPUTLAYOUT](#),
[PROP_INPUTRANKS](#), [PROP_OUTPUT](#), [PROP_OUTPUTTYPE](#), [PROP_OUTPUTNAME](#),
[PROP_OUTPUTLAYOUT](#), [PROP_OUTPUTRANKS](#), [PROP_CUSTOM](#), [PROP_SUBPLUGINS](#),
[PROP_ACCELERATOR](#), [PROP_IS_UPDATABLE](#), [PROP_LATENCY](#), [PROP_THROUGHPUT](#),
[PROP_INPUTCOMBINATION](#), [PROP_OUTPUTCOMBINATION](#), [PROP_SHARED_TENSOR_FILTER_KEY](#),
[PROP_LATENCY_REPORT](#),
[PROP_INVOKE_DYNAMIC](#), [PROP_CONFIG](#), [PROP_SUSPEND](#) }
GstTensorFilter properties.

Functions

- gchar * [gst_tensorsinfo_compare_to_string](#) (const [GstTensorsInfo](#) *info, const [GstTensorsInfo](#) *info2)
Printout the comparison results of two tensors as a string.
- void [gst_tensorsinfo_compare_print](#) (const [GstTensorsInfo](#) *info1, const [GstTensorsInfo](#) *info2)
Printout the comparison results of two tensors.
- gboolean [gst_tensor_filter_allocate_in_invoke](#) ([GstTensorFilterPrivate](#) *priv)
check if the allocate_in_invoke is valid for the framework
- void [gst_tensor_filter_install_properties](#) (GObjectClass *gobject_class)
Installs all the properties for tensor_filter.
- void [gst_tensor_filter_common_init_property](#) ([GstTensorFilterPrivate](#) *priv)
Initialize the properties for tensor-filter.
- void [gst_tensor_filter_common_free_property](#) ([GstTensorFilterPrivate](#) *priv)
Free the properties for tensor-filter.
- gboolean [gst_tensor_filter_common_set_property](#) ([GstTensorFilterPrivate](#) *priv, guint prop_id, const GValue *value, GParamSpec *pspec)
Set the properties for tensor_filter.
- gboolean [gst_tensor_filter_common_get_property](#) ([GstTensorFilterPrivate](#) *priv, guint prop_id, GValue *value, GParamSpec *pspec)
Get the properties for tensor_filter.
- gboolean [gst_tensor_filter_common_get_combined_in_info](#) ([GstTensorFilterPrivate](#) *priv, const [GstTensorsInfo](#) *in, [GstTensorsInfo](#) *combined)
Configure input tensor info with combi option.
- gboolean [gst_tensor_filter_common_get_combined_out_info](#) ([GstTensorFilterPrivate](#) *priv, const [GstTensorsInfo](#) *in, const [GstTensorsInfo](#) *out, [GstTensorsInfo](#) *combined)
Configure output tensor info with combi option.
- gboolean [gst_tensor_filter_common_get_out_info](#) ([GstTensorFilterPrivate](#) *priv, [GstTensorsInfo](#) *in, [GstTensorsInfo](#) *out)
Get output tensor info from NN model with given input info.

- void `gst_tensor_filter_load_tensor_info` (`GstTensorFilterPrivate *priv`)
Load tensor info from NN model. (both input and output tensor)
- void `gst_tensor_filter_common_open_fw` (`GstTensorFilterPrivate *priv`)
Open NN framework.
- void `gst_tensor_filter_common_unload_fw` (`GstTensorFilterPrivate *priv`)
Unload NN framework.
- void `gst_tensor_filter_common_close_fw` (`GstTensorFilterPrivate *priv`)
Close NN framework.
- `gchar *` `gst_tensor_filter_detect_framework` (`const gchar *const *model_files`, `const guint num_models`, `const gboolean load_conf`)
Get neural network framework name from given model file. This does not guarantee the framework is available on the target device.
- `gboolean` `gst_tensor_filter_check_hw_availability` (`const gchar *name`, `const accl_hw hw`, `const char *custom`)
Check if the given hw is supported by the framework.
- void `gst_tensor_filter_destroy_notify_util` (`GstTensorFilterPrivate *priv`, `void *data`)
Free the data allocated for tensor filter output.

9.107.1 Detailed Description

Common functions for various tensor_filters.

Copyright (C) 2019 Parichay Kapoor pk.kapoor@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

28 Aug 2019

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.107.2 Macro Definition Documentation

9.107.2.1 DBG

```
#define DBG (!priv->silent)
```

Macro for debug mode.

Definition at line 41 of file tensor_filter_common.h.

9.107.2.2 gst_tensor_filter_v0_call

```
#define gst_tensor_filter_v0_call(  
    priv,  
    ret,  
    funcname,  
    ... )
```

Value:

```
do { \  
gst_tensor_filter_common_open_fw (priv); \  
ret = -1; \  
if ((priv)->prop.fw_opened && (priv)->fw && (priv)->fw->funcname) { \  
    ret = (priv)->fw->funcname (&(priv)->prop, &(priv)->privateData, __VA_ARGS__); \  
} \  
} while (0)
```

Invoke callbacks of nn framework. Guarantees calling open for the first call.

Definition at line 53 of file tensor_filter_common.h.

9.107.2.3 gst_tensor_filter_v1_call

```
#define gst_tensor_filter_v1_call(  
    priv,  
    ret,  
    funcname,  
    ... )
```

Value:

```
do { \  
gst_tensor_filter_common_open_fw (priv); \  
ret = -1; \  
if ((priv)->prop.fw_opened && (priv)->fw && (priv)->fw->funcname) { \  
    ret = (priv)->fw->funcname ((priv)->fw, &(priv)->prop, (priv)->privateData, __VA_ARGS__); \  
} \  
} while (0)
```

Definition at line 61 of file tensor_filter_common.h.

9.107.2.4 GST_TF_FW_INVOKE_COMPAT

```
#define GST_TF_FW_INVOKE_COMPAT(
    priv,
    ret,
    in,
    out )
```

Value:

```
do { \
    ret = -1; \
    if (GST_TF_FW_V0 ((priv)->fw)) { \
        ret = (priv)->fw->invoke_NN (&(priv)->prop, &(priv)->privateData, (in), (out)); \
    } else if (GST_TF_FW_V1 ((priv)->fw)) { \
        ret = (priv)->fw->invoke ((priv)->fw, &(priv)->prop, (priv)->privateData, (in), (out)); \
    } \
} while (0)
```

Definition at line 69 of file `tensor_filter_common.h`.

9.107.2.5 GST_TF_FW_V0

```
#define GST_TF_FW_V0(
    fw ) GST_TF_FW_VN (fw, 0)
```

Definition at line 47 of file `tensor_filter_common.h`.

9.107.2.6 GST_TF_FW_V1

```
#define GST_TF_FW_V1(
    fw ) GST_TF_FW_VN (fw, 1)
```

Definition at line 48 of file `tensor_filter_common.h`.

9.107.2.7 GST_TF_FW_VN

```
#define GST_TF_FW_VN(
    fw,
    vn ) (fw && checkGstTensorFilterFrameworkVersion (fw->version, vn))
```

Check `tensor_filter` framework version

Definition at line 45 of file `tensor_filter_common.h`.

9.107.2.8 GST_TF_STAT_MAX_RECENT

```
#define GST_TF_STAT_MAX_RECENT (10)
```

Definition at line 78 of file tensor_filter_common.h.

9.107.3 Typedef Documentation

9.107.3.1 GstTensorFilterCombination

```
typedef struct _GstTensorFilterCombination GstTensorFilterCombination
```

Structure definition for tensor-filter in/out combination.

9.107.3.2 GstTensorFilterPrivate

```
typedef struct _GstTensorFilterPrivate GstTensorFilterPrivate
```

Structure definition for common tensor-filter properties.

9.107.3.3 GstTensorFilterStatistics

```
typedef struct _GstTensorFilterStatistics GstTensorFilterStatistics
```

Structure definition for tensor-filter statistics.

9.107.4 Enumeration Type Documentation

Enumerator

9.107.4.1 anonymous enum

anonymous enum

GstTensorFilter properties.

Enumerator

PROP_0	
PROP_SILENT	
PROP_FRAMEWORK	
PROP_MODEL	
PROP_INPUT	
PROP_INPUTTYPE	
PROP_INPUTNAME	
PROP_INPUTLAYOUT	
PROP_INPUTRANKS	
PROP_OUTPUT	
PROP_OUTPUTTYPE	
PROP_OUTPUTNAME	
PROP_OUTPUTLAYOUT	
PROP_OUTPUTRANKS	
PROP_CUSTOM	
PROP_SUBPLUGINS	
PROP_ACCELERATOR	
PROP_IS_UPDATABLE	
PROP_LATENCY	
PROP_THROUGHPUT	
PROP_INPUTCOMBINATION	
PROP_OUTPUTCOMBINATION	
PROP_SHARED_TENSOR_FILTER_KEY	
PROP_LATENCY_REPORT	
PROP_INVOKE_DYNAMIC	
PROP_CONFIG	
PROP_SUSPEND	

Definition at line 83 of file tensor_filter_common.h.

9.107.5 Function Documentation**9.107.5.1 gst_tensor_filter_allocate_in_invoke()**

```
gboolean gst_tensor_filter_allocate_in_invoke (
    GstTensorFilterPrivate * priv )
```

check if the allocate_in_invoke is valid for the framework

Parameters

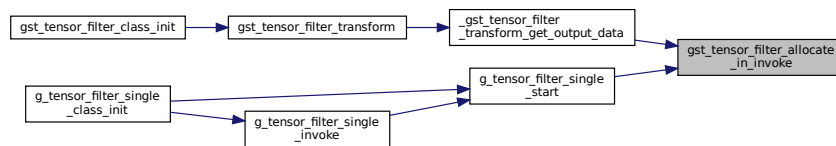
in	<i>priv</i>	Struct containing the properties of the object
----	-------------	--

Returns

TRUE if valid, FALSE on error

Definition at line 757 of file tensor_filter_common.c.

Here is the caller graph for this function:

9.107.5.2 `gst_tensor_filter_check_hw_availability()`

```

gboolean gst_tensor_filter_check_hw_availability (
    const gchar * name,
    const accl_hw hw,
    const char * custom )

```

Check if the given hw is supported by the framework.

Parameters

in	<i>name</i>	The name of filter sub-plugin.
in	<i>hw</i>	Backend accelerator hardware.
in	<i>custom</i>	User-defined string to handle detailed hardware option.

Returns

TRUE if given hw is available.

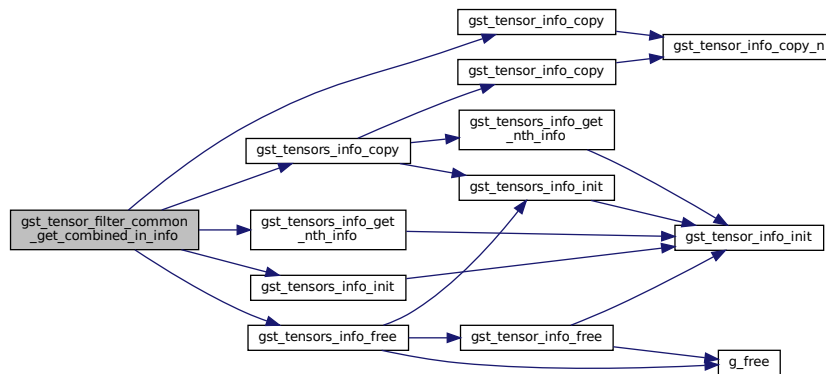
Note

This function is included in nnstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nnstreamer_internal.h](#)).

Only check for specific HW, DEFAULT/AUTO are always supported

Definition at line 2923 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.6 gst_tensor_filter_common_get_combined_out_info()

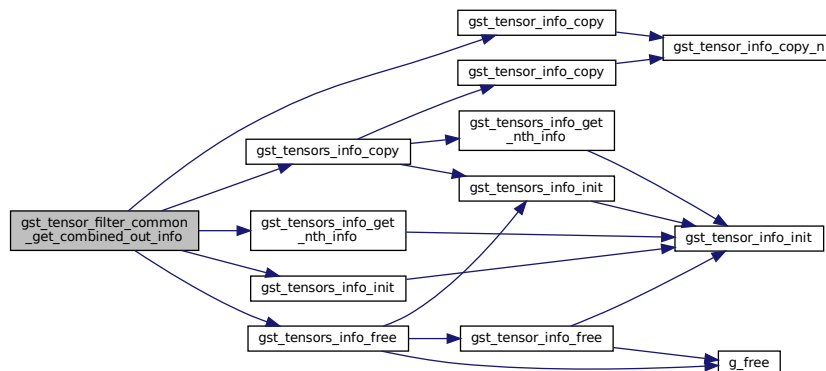
```

gboolean gst_tensor_filter_common_get_combined_out_info (
    GstTensorFilterPrivate * priv,
    const GstTensorsInfo * in,
    const GstTensorsInfo * out,
    GstTensorsInfo * combined )
  
```

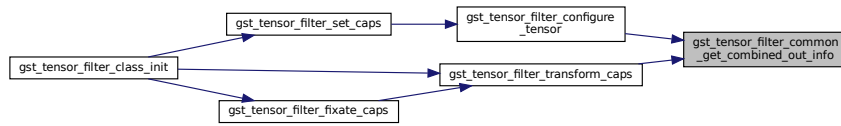
Configure output tensor info with combi option.

Definition at line 2315 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.7 gst_tensor_filter_common_get_out_info()

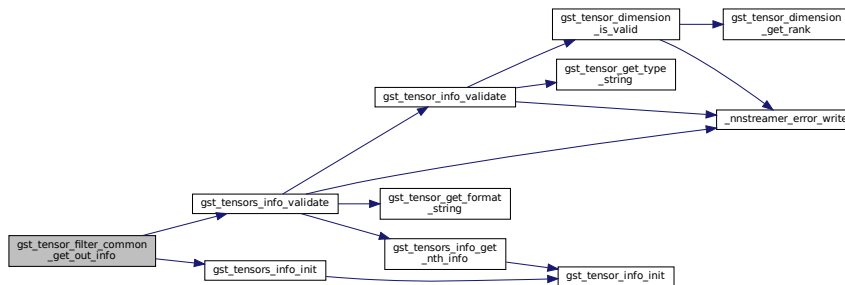
```

gboolean gst_tensor_filter_common_get_out_info (
    GstTensorFilterPrivate * priv,
    GstTensorsInfo * in,
    GstTensorsInfo * out )
    
```

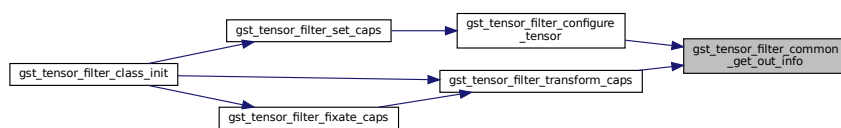
Get output tensor info from NN model with given input info.

Definition at line 2374 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.8 `gst_tensor_filter_common_get_property()`

```
gboolean gst_tensor_filter_common_get_property (
    GstTensorFilterPrivate * priv,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec )
```

Get the properties for `tensor_filter`.

Parameters

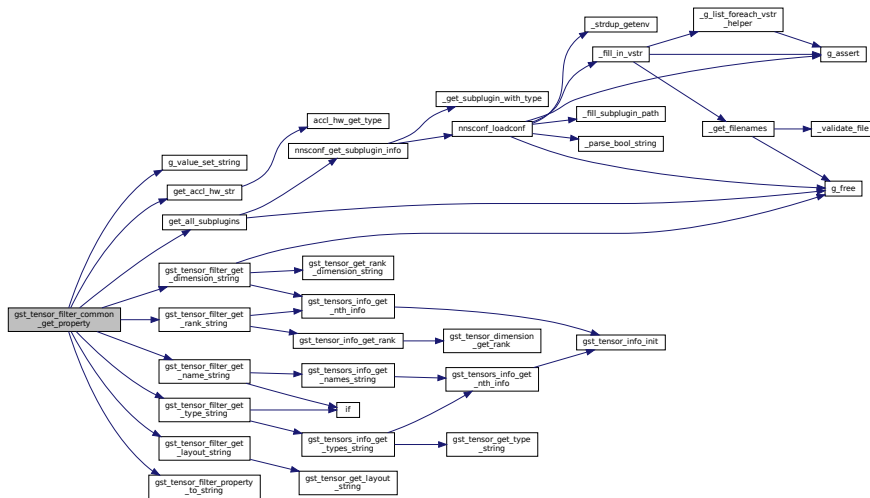
in	<i>priv</i>	Struct containing the properties of the object
in	<i>prop_id</i>	Id for the property
in	<i>value</i>	Container to return the asked property
in	<i>pspec</i>	Metadata to specify the parameter

Returns

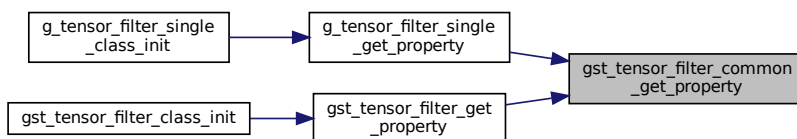
TRUE if `prop_id` is value, else FALSE

Definition at line 2102 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



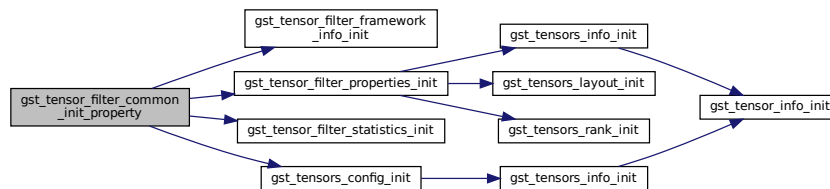
9.107.5.9 `gst_tensor_filter_common_init_property()`

```
void gst_tensor_filter_common_init_property (
    GstTensorFilterPrivate * priv )
```

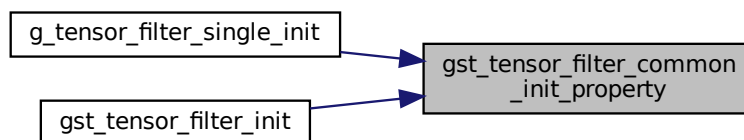
Initialize the properties for tensor-filter.

Definition at line 1041 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



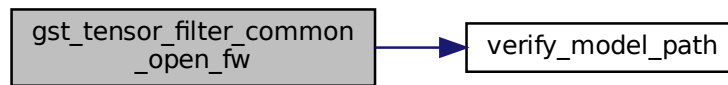
9.107.5.10 `gst_tensor_filter_common_open_fw()`

```
void gst_tensor_filter_common_open_fw (
    GstTensorFilterPrivate * priv )
```

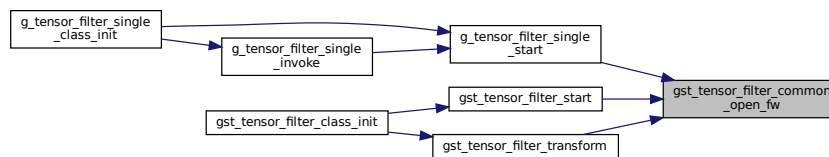
Open NN framework.

Definition at line 2491 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.11 gst_tensor_filter_common_set_property()

```

gboolean gst_tensor_filter_common_set_property (
    GstTensorFilterPrivate * priv,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec )
  
```

Set the properties for tensor_filter.

Parameters

in	<i>priv</i>	Struct containing the properties of the object
in	<i>prop_id</i>	Id for the property
in	<i>value</i>	Container to return the asked property
in	<i>pspec</i>	Metadata to specify the parameter

Returns

TRUE if *prop_id* is value, else FALSE

Although no one return !0 at this point, let's enable error handling.

Definition at line 1967 of file tensor_filter_common.c.

9.107.5.13 `gst_tensor_filter_destroy_notify_util()`

```
void gst_tensor_filter_destroy_notify_util (
    GstTensorFilterPrivate * priv,
    void * data )
```

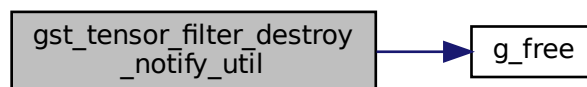
Free the data allocated for tensor filter output.

Parameters

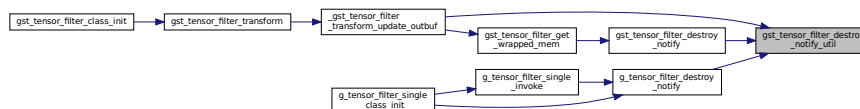
in	<i>priv</i>	Struct containing the properties of the object
in	<i>data</i>	Data to be freed

Definition at line 786 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.14 `gst_tensor_filter_detect_framework()`

```
gchar* gst_tensor_filter_detect_framework (
    const gchar *const * model_files,
    const guint num_models,
    const gboolean load_conf )
```

Get neural network framework name from given model file. This does not guarantee the framework is available on the target device.

Parameters

in	<i>model_files</i>	the prediction model paths
in	<i>num_models</i>	the number of model files
in	<i>load_conf</i>	flag to load configuration for the priority of framework

Returns

Possible framework name (NULL if it fails to detect automatically). Caller should free returned value using [g_free\(\)](#).

Parameters

in	<i>model_files</i>	the prediction model paths
in	<i>num_models</i>	the number of model files
in	<i>load_conf</i>	flag to load configuration for the priority of framework

Returns

Possible framework name (NULL if it fails to detect automatically). Caller should free returned value using [g_free\(\)](#).

Note

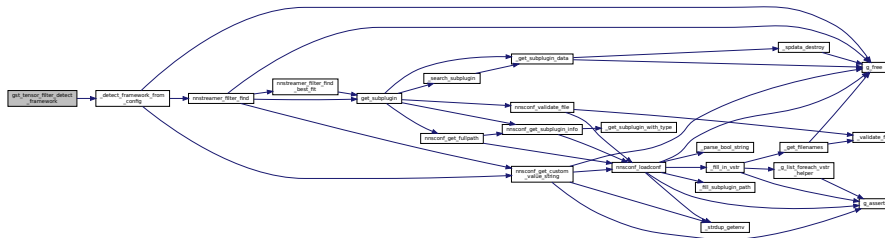
This function is included in nnstreamer internal header for native APIs. When changing the declaration, you should update the internal header ([nnstreamer_internal.h](#)).

Note

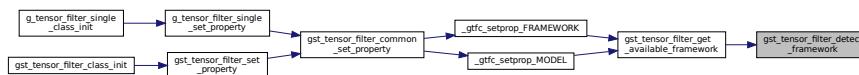
When adding new file extension for auto fw detection, you should check the condition to validate model file - `ml_validate_model_file()` in ML-API.

Definition at line 1232 of file `tensor_filter_common.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.15 `gst_tensor_filter_install_properties()`

```
void gst_tensor_filter_install_properties (
    GObjectClass * gobject_class )
```

Installs all the properties for `tensor_filter`.

Parameters

in	<i>object_class</i>	Glib object class whose properties will be set
----	---------------------	--

min

max

default: off

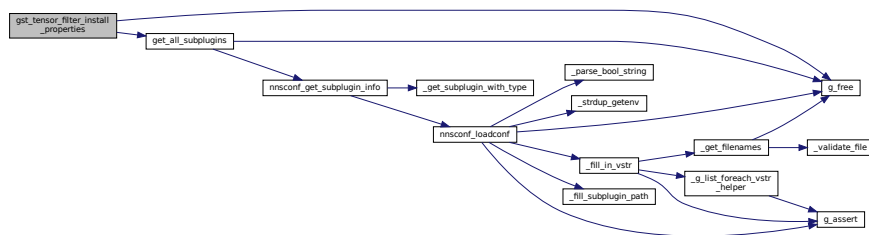
min

max

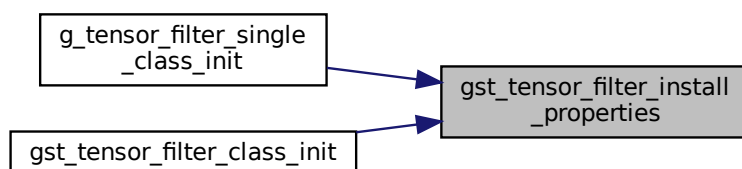
default: off

Definition at line 888 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.16 `gst_tensor_filter_load_tensor_info()`

```
void gst_tensor_filter_load_tensor_info (
    GstTensorFilterPrivate * priv )
```

Load tensor info from NN model. (both input and output tensor)

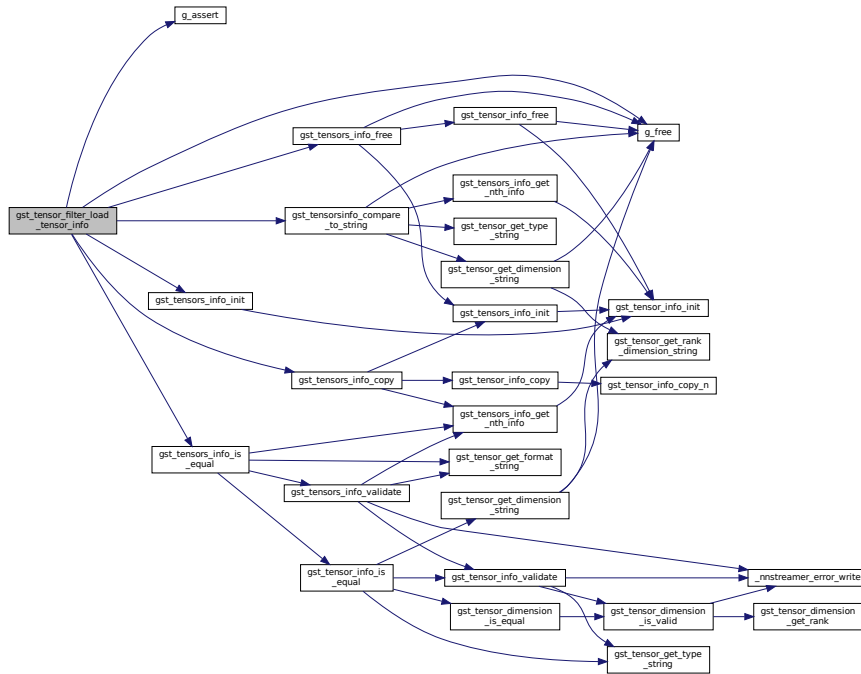
if set-property called and already has info, verify it!

In case of dynamic invoke, output tensors info is determined after invoke.

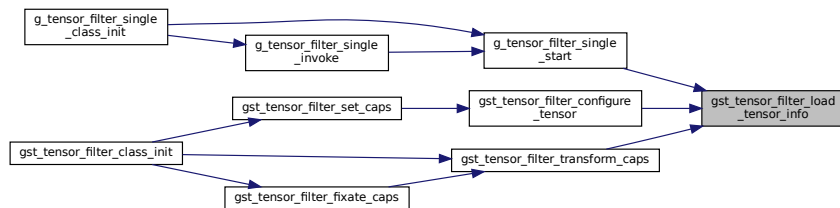
if set-property called and already has info, verify it!

Definition at line 2409 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.107.5.17 `gst_tensorsinfo_compare_print()`

```
void gst_tensorsinfo_compare_print (
    const GstTensorsInfo * info1,
    const GstTensorsInfo * info2 )
```

Printout the comparison results of two tensors.

Parameters

in	<i>info1</i>	The tensors to be shown on the left hand side
in	<i>info2</i>	The tensors to be shown on the right hand side

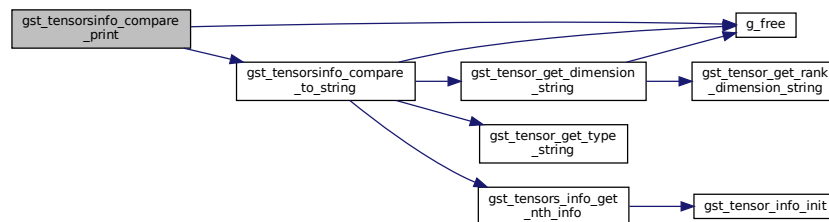
Todo If this is going to be used by other elements, move this to nnstreamer/tensor_common.

Parameters

in	<i>info1</i>	The tensors to be shown on the left hand side
in	<i>info2</i>	The tensors to be shown on the right hand side

Definition at line 874 of file tensor_filter_common.c.

Here is the call graph for this function:



9.107.5.18 `gst_tensorsinfo_compare_to_string()`

```
gchar* gst_tensorsinfo_compare_to_string (
    const GstTensorsInfo * info1,
    const GstTensorsInfo * info2 )
```

Printout the comparison results of two tensors as a string.

Parameters

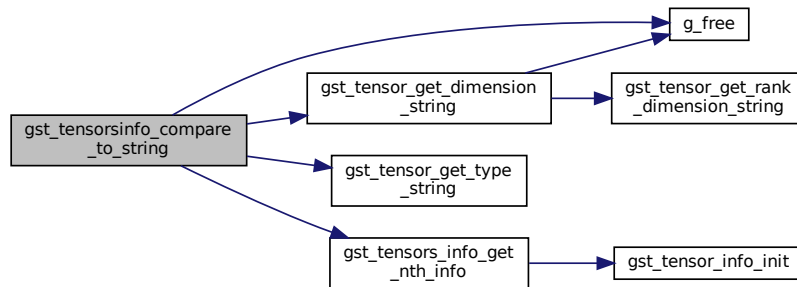
in	<i>info1</i>	The tensors to be shown on the left hand side
in	<i>info2</i>	The tensors to be shown on the right hand side

Returns

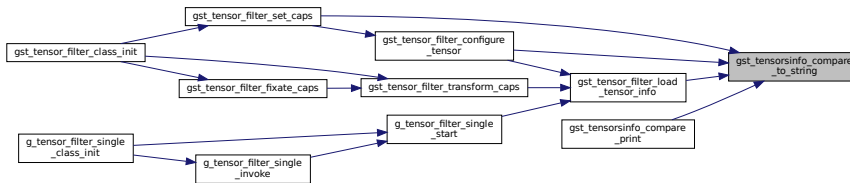
The printout string allocated. Caller should free the value.

Definition at line 811 of file tensor_filter_common.c.

Here is the call graph for this function:



Here is the caller graph for this function:



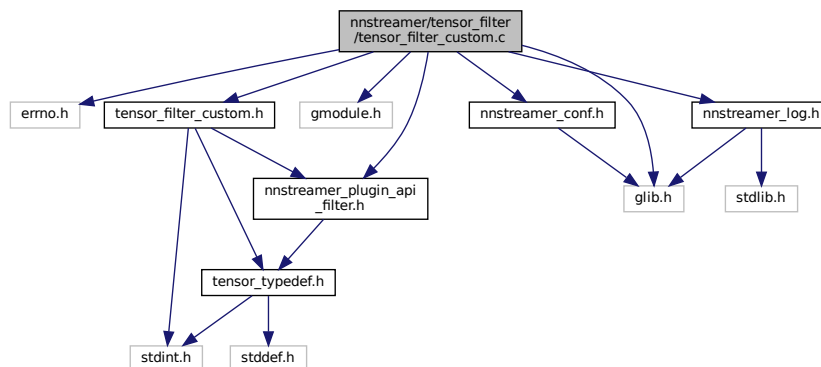
9.108 nnstreamer/tensor_filter/tensor_filter_custom.c File Reference

Custom tensor post-processing interface for NNStreamer suite between NN developer-plugins and NNStreamer.

```

#include <errno.h>
#include <glib.h>
#include <gmodule.h>
#include "tensor_filter_custom.h"
#include "nnstreamer_plugin_api_filter.h"
#include "nnstreamer_conf.h"
  
```

```
#include <nnstreamer_log.h>
Include dependency graph for tensor_filter_custom.c:
```



Classes

- struct [_internal_data](#)
internal_data

Typedefs

- typedef struct [_internal_data](#) *internal_data*

Functions

- void [init_filter_custom](#) (void)
Initialize this object for tensor_filter subplugin runtime register.
- static int [custom_loadlib](#) (const [GstTensorFilterProperties](#) *prop, void **private_data)
Load the custom library. Will skip loading if it's already loaded.
- static int [custom_open](#) (const [GstTensorFilterProperties](#) *prop, void **private_data)
The open callback for GstTensorFilterFramework. Called before anything else.
- static int [custom_invoke](#) (const [GstTensorFilterProperties](#) *prop, void **private_data, const [GstTensorMemory](#) *input, [GstTensorMemory](#) *output)
The mandatory callback for GstTensorFilterFramework.
- static int [custom_getInputDim](#) (const [GstTensorFilterProperties](#) *prop, void **private_data, [GstTensorsInfo](#) *info)
The optional callback for GstTensorFilterFramework.
- static int [custom_getOutputDim](#) (const [GstTensorFilterProperties](#) *prop, void **private_data, [GstTensorsInfo](#) *info)
The optional callback for GstTensorFilterFramework.
- static int [custom_setInputDim](#) (const [GstTensorFilterProperties](#) *prop, void **private_data, const [GstTensorsInfo](#) *in_info, [GstTensorsInfo](#) *out_info)
The set-input-dim callback for GstTensorFilterFramework.
- static void [custom_close](#) (const [GstTensorFilterProperties](#) *prop, void **private_data)
Free privateData and move on.
- static void [custom_destroyNotify](#) (void **private_data, void *data)

The optional callback for GstTensorFilterFramework.

- static int [custom_allocateInInvoke](#) (void **private_data)

The optional callback for GstTensorFilterFramework.

- static int [custom_checkAvailability](#) ([accl_hw](#) hw)

Check support of the backend.

- void [fini_filter_custom](#) (void)

Destruct the subplugin.

Variables

- static gchar [filter_subplugin_custom](#) [] = "custom"
- static [GstTensorFilterFramework](#) [NNS_support_custom](#)

9.108.1 Detailed Description

Custom tensor post-processing interface for NNStreamer suite between NN developer-plugins and NNStreamer.

GStreamer Tensor_Filter Module Copyright (C) 2018 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

01 Jun 2018

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

This is the per-NN-framework plugin (custom) for tensor_filter. Fill in "GstTensorFilterFramework" for [tensor_filter.h/c](#)

9.108.2 Typedef Documentation

9.108.2.1 `internal_data`

```
typedef struct _internal_data internal_data
```

Definition at line 57 of file `tensor_filter_custom.c`.

9.108.3 Function Documentation

9.108.3.1 `custom_allocateInInvoke()`

```
static int custom_allocateInInvoke (  
    void ** private_data ) [static]
```

The optional callback for `GstTensorFilterFramework`.

Definition at line 283 of file `tensor_filter_custom.c`.

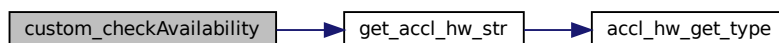
9.108.3.2 `custom_checkAvailability()`

```
static int custom_checkAvailability (  
    accl_hw hw ) [static]
```

Check support of the backend.

Definition at line 298 of file `tensor_filter_custom.c`.

Here is the call graph for this function:



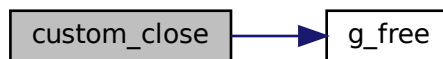
9.108.3.3 custom_close()

```
static void custom_close (  
    const GstTensorFilterProperties * prop,  
    void ** private_data ) [static]
```

Free privateData and move on.

Definition at line 252 of file tensor_filter_custom.c.

Here is the call graph for this function:



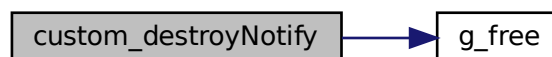
9.108.3.4 custom_destroyNotify()

```
static void custom_destroyNotify (  
    void ** private_data,  
    void * data ) [static]
```

The optional callback for GstTensorFilterFramework.

Definition at line 267 of file tensor_filter_custom.c.

Here is the call graph for this function:



9.108.3.5 custom_getInputDim()

```
static int custom_getInputDim (
    const GstTensorFilterProperties * prop,
    void ** private_data,
    GstTensorsInfo * info ) [static]
```

The optional callback for GstTensorFilterFramework.

Definition at line 193 of file tensor_filter_custom.c.

9.108.3.6 custom_getOutputDim()

```
static int custom_getOutputDim (
    const GstTensorFilterProperties * prop,
    void ** private_data,
    GstTensorsInfo * info ) [static]
```

The optional callback for GstTensorFilterFramework.

Definition at line 212 of file tensor_filter_custom.c.

9.108.3.7 custom_invoke()

```
static int custom_invoke (
    const GstTensorFilterProperties * prop,
    void ** private_data,
    const GstTensorMemory * input,
    GstTensorMemory * output ) [static]
```

The mandatory callback for GstTensorFilterFramework.

Parameters

	<i>prop</i>	The properties of parent object
in	<i>input</i>	The array of input tensors
out	<i>output</i>	The array of output tensors

Returns

0 if OK. non-zero if error.

Definition at line 167 of file tensor_filter_custom.c.

9.108.3.8 custom_loadlib()

```
static int custom_loadlib (
    const GstTensorFilterProperties * prop,
    void ** private_data ) [static]
```

Load the custom library. Will skip loading if it's already loaded.

Returns

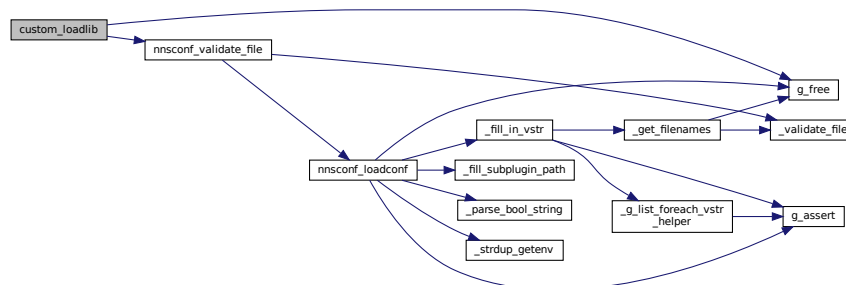
0 if successfully loaded. 1 if skipped (already loaded). -1 if error

Todo : Check the integrity of filter->data and filter->model_file, nnfw

Todo Double check if this check is really required and safe

Definition at line 64 of file tensor_filter_custom.c.

Here is the call graph for this function:



Here is the caller graph for this function:



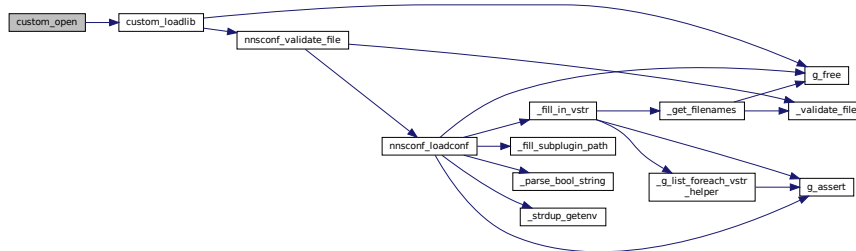
9.108.3.9 custom_open()

```
static int custom_open (
    const GstTensorFilterProperties * prop,
    void ** private_data ) [static]
```

The open callback for GstTensorFilterFramework. Called before anything else.

Definition at line 138 of file tensor_filter_custom.c.

Here is the call graph for this function:



9.108.3.10 custom_setInputDim()

```
static int custom_setInputDim (
    const GstTensorFilterProperties * prop,
    void ** private_data,
    const GstTensorsInfo * in_info,
    GstTensorsInfo * out_info ) [static]
```

The set-input-dim callback for GstTensorFilterFramework.

Definition at line 231 of file tensor_filter_custom.c.

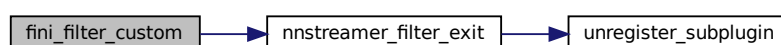
9.108.3.11 fini_filter_custom()

```
void fini_filter_custom (
    void )
```

Destruct the subplugin.

Definition at line 335 of file tensor_filter_custom.c.

Here is the call graph for this function:



9.108.3.12 init_filter_custom()

```
void init_filter_custom (
    void )
```

Initialize this object for tensor_filter subplugin runtime register.

Definition at line 38 of file tensor_filter_custom.c.

9.108.4 Variable Documentation

9.108.4.1 filter_subplugin_custom

```
gchar filter_subplugin_custom[] = "custom" [static]
```

Definition at line 306 of file tensor_filter_custom.c.

9.108.4.2 NNS_support_custom

```
GstTensorFilterFramework NNS_support_custom [static]
```

Initial value:

```
= {
    .version = GST_TENSOR_FILTER_FRAMEWORK_V0,
    .name = filter_subplugin_custom,
    .allow_in_place = FALSE,
    .allocate_in_invoke = TRUE,
    .run_without_model = FALSE,
    .invoke_NN = custom_invoke,
    .handleEvent = NULL,
    .getInputDimension = custom_getInputDim,
    .getOutputDimension = custom_getOutputDim,
    .setInputDimension = custom_setInputDim,
    .open = custom_open,
    .close = custom_close,
    .destroyNotify = custom_destroyNotify,
    .allocateInInvoke = custom_allocateInInvoke,
    .checkAvailability = custom_checkAvailability,
}
```

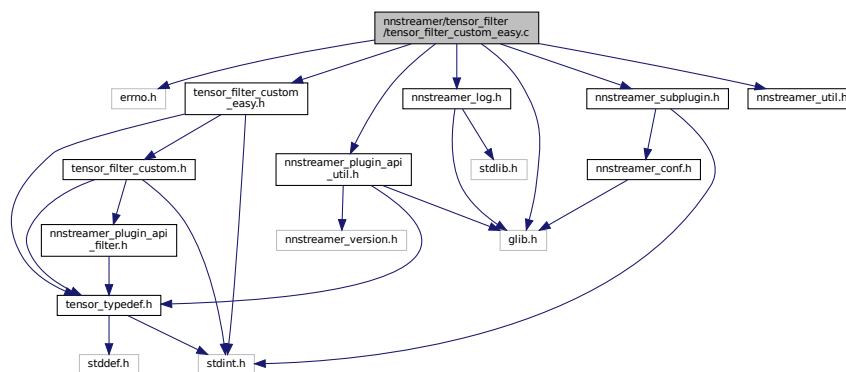
Definition at line 308 of file tensor_filter_custom.c.

9.109 nnstreamer/tensor_filter/tensor_filter_custom_easy.c File Reference

Custom tensor processing interface for simple functions.

```
#include <errno.h>
#include <glib.h>
#include <tensor_filter_custom_easy.h>
#include <nnstreamer_log.h>
#include <nnstreamer_plugin_api_util.h>
#include <nnstreamer_subplugin.h>
#include <nnstreamer_util.h>
```

Include dependency graph for tensor_filter_custom_easy.c:



Classes

- struct [runtime_data](#)
The easy-filter user's data.

Functions

- void [init_filter_custom_easy](#) (void)
internal_data
- static void [custom_free_internal_data](#) ([internal_data](#) *data)
Internal function to release internal data.
- int [NNS_custom_easy_register](#) (const char *modelname, [NNS_custom_invoke](#) func, void *data, const [GstTensorInfo](#) *in_info, const [GstTensorInfo](#) *out_info)
Register the custom-easy tensor function. More info in .h.
- int [NNS_custom_easy_dynamic_register](#) (const char *modelname, [NNS_custom_invoke_dynamic](#) func, void *data, const [GstTensorInfo](#) *in_info)
Register the custom-easy tensor function. More info in .h.
- int [NNS_custom_easy_unregister](#) (const char *modelname)
Unregister the custom-easy tensor function.
- static int [custom_open](#) (const [GstTensorFilterProperties](#) *prop, void **private_data)
Callback required by tensor_filter subplugin.
- static void [custom_close](#) (const [GstTensorFilterProperties](#) *prop, void **private_data)

Callback required by tensor_filter subplugin.

- static int [custom_invoke](#) (const [GstTensorFilterFramework](#) *self, [GstTensorFilterProperties](#) *prop, void *private_data, const [GstTensorMemory](#) *input, [GstTensorMemory](#) *output)

Callback required by tensor_filter subplugin.

- static int [custom_getFrameworkInfo](#) (const [GstTensorFilterFramework](#) *self, const [GstTensorFilterProperties](#) *prop, void *private_data, [GstTensorFilterFrameworkInfo](#) *fw_info)

V1 tensor-filter wrapper callback function, "getFrameworkInfo".

- static int [custom_getModelInfo](#) (const [GstTensorFilterFramework](#) *self, const [GstTensorFilterProperties](#) *prop, void *private_data, [model_info_ops](#) ops, [GstTensorsInfo](#) *in_info, [GstTensorsInfo](#) *out_info)

C V1 tensor-filter wrapper callback function, "getModelInfo".

- static int [custom_eventHandler](#) (const [GstTensorFilterFramework](#) *self, const [GstTensorFilterProperties](#) *prop, void *private_data, [event_ops](#) ops, [GstTensorFilterFrameworkEventData](#) *data)

C V1 tensor-filter wrapper callback function, "eventHandler".

- void [fini_filter_custom_easy](#) (void)

Destruct the subplugin.

Variables

- [internal_data](#)
- static [GstTensorFilterFramework](#) [NNS_support_custom_easy](#)

9.109.1 Detailed Description

Custom tensor processing interface for simple functions.

GStreamer Tensor_Filter, Customized Module, Easy Mode Copyright (C) 2019 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

24 Oct 2019

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.109.2 Function Documentation

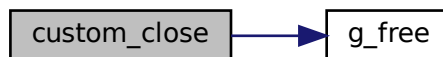
9.109.2.1 custom_close()

```
static void custom_close (  
    const GstTensorFilterProperties * prop,  
    void ** private_data ) [static]
```

Callback required by tensor_filter subplugin.

Definition at line 230 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



9.109.2.2 custom_eventHandler()

```
static int custom_eventHandler (  
    const GstTensorFilterFramework * self,  
    const GstTensorFilterProperties * prop,  
    void * private_data,  
    event_ops ops,  
    GstTensorFilterFrameworkEventData * data ) [static]
```

C V1 tensor-filter wrapper callback function, "eventHandler".

Definition at line 319 of file tensor_filter_custom_easy.c.

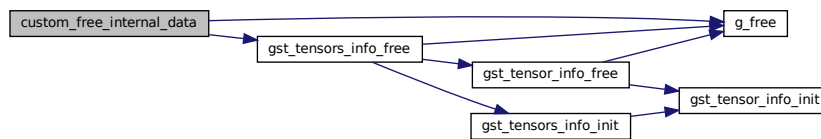
9.109.2.3 custom_free_internal_data()

```
static void custom_free_internal_data (
    internal_data * data ) [static]
```

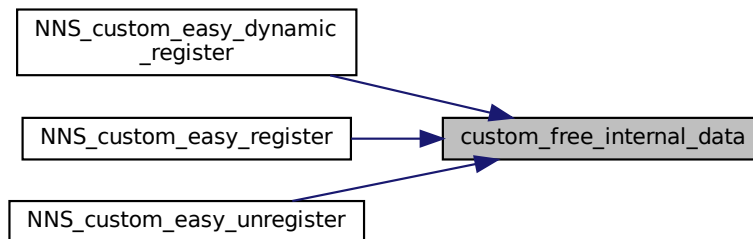
Internal function to release internal data.

Definition at line 62 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.109.2.4 custom_getFrameworkInfo()

```
static int custom_getFrameworkInfo (
    const GstTensorFilterFramework * self,
    const GstTensorFilterProperties * prop,
    void * private_data,
    GstTensorFilterFrameworkInfo * fw_info ) [static]
```

V1 tensor-filter wrapper callback function, "getFrameworkInfo".

Definition at line 278 of file tensor_filter_custom_easy.c.

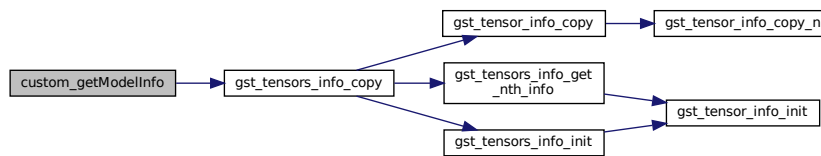
9.109.2.5 custom_getModelInfo()

```
static int custom_getModelInfo (
    const GstTensorFilterFramework * self,
    const GstTensorFilterProperties * prop,
    void * private_data,
    model_info_ops ops,
    GstTensorsInfo * in_info,
    GstTensorsInfo * out_info ) [static]
```

C V1 tensor-filter wrapper callback function, "getModelInfo".

Definition at line 300 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



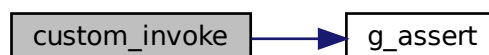
9.109.2.6 custom_invoke()

```
static int custom_invoke (
    const GstTensorFilterFramework * self,
    GstTensorFilterProperties * prop,
    void * private_data,
    const GstTensorMemory * input,
    GstTensorMemory * output ) [static]
```

Callback required by tensor_filter subplugin.

Definition at line 243 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



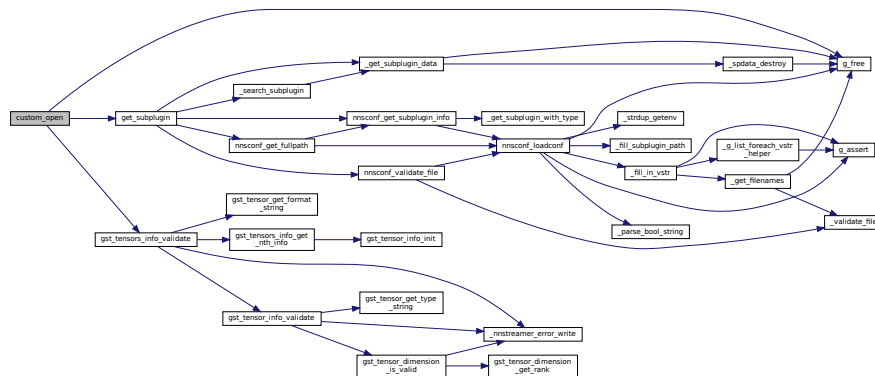
9.109.2.7 custom_open()

```
static int custom_open (
    const GstTensorFilterProperties * prop,
    void ** private_data ) [static]
```

Callback required by tensor_filter subplugin.

Definition at line 165 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



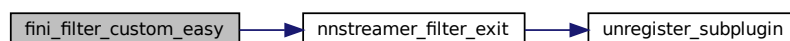
9.109.2.8 fini_filter_custom_easy()

```
void fini_filter_custom_easy (
    void )
```

Destruct the subplugin.

Definition at line 352 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



9.109.2.9 init_filter_custom_easy()

```
void init_filter_custom_easy (
    void )
```

internal_data

Initialize this object for tensor_filter subplugin runtime register. < The easy-filter writer's data

Definition at line 33 of file tensor_filter_custom_easy.c.

9.109.2.10 NNS_custom_easy_dynamic_register()

```
int NNS_custom_easy_dynamic_register (
    const char * modelname,
    NNS_custom_invoke_dynamic_func,
    void * data,
    const GstTensorsInfo * in_info )
```

Register the custom-easy tensor function. More info in .h.

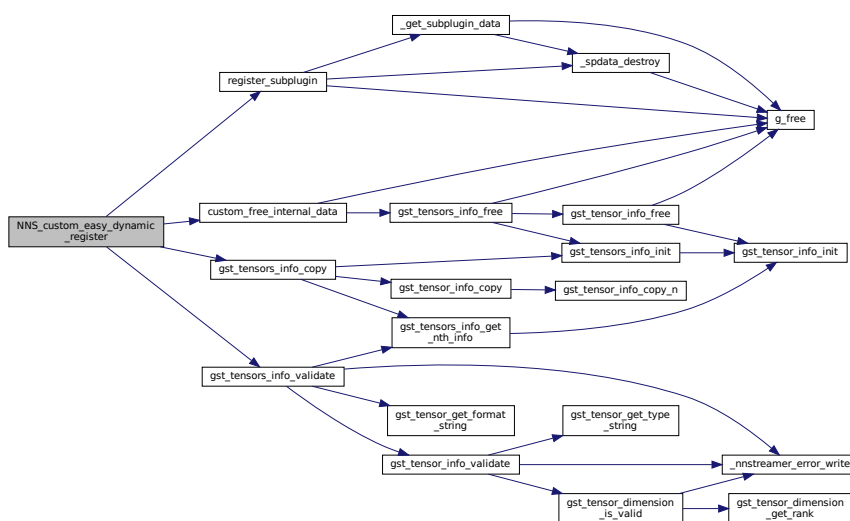
Register the custom-easy tensor function for dynamic invoke.

Returns

0 if success. -ERRNO if error.

Definition at line 112 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



9.109.2.11 NNS_custom_easy_register()

```
int NNS_custom_easy_register (
    const char * modelname,
    NNS_custom_invoke func,
    void * data,
    const GstTensorsInfo * in_info,
    const GstTensorsInfo * out_info )
```

Register the custom-easy tensor function. More info in .h.

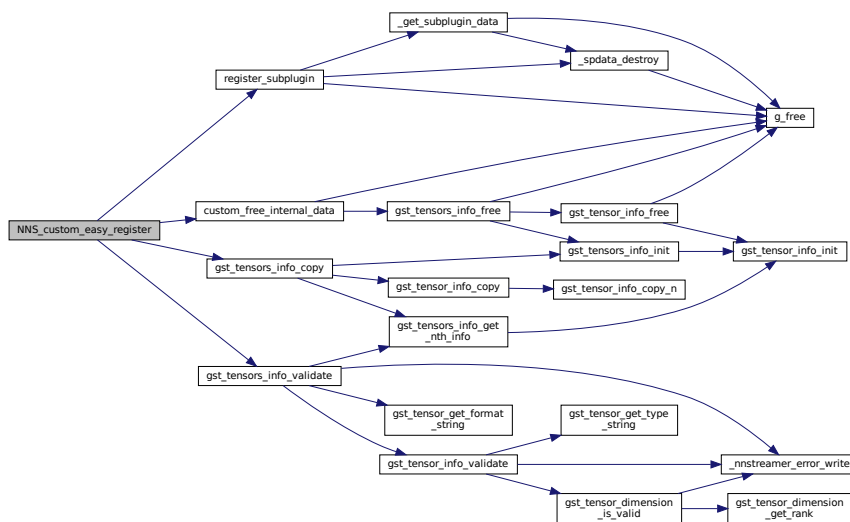
Register the custom-easy tensor function.

Returns

0 if success. -ERRNO if error.

Definition at line 76 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:



9.109.2.12 NNS_custom_easy_unregister()

```
int NNS_custom_easy_unregister (
    const char * modelname )
```

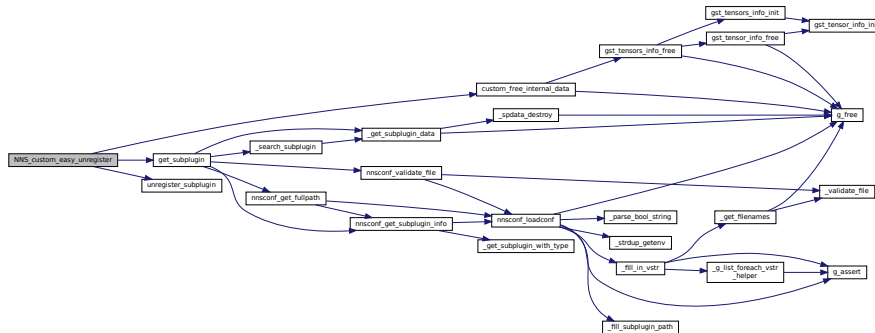
Unregister the custom-easy tensor function.

Returns

0 if success. -EINVAL if invalid model name.

Definition at line 144 of file tensor_filter_custom_easy.c.

Here is the call graph for this function:

**9.109.3 Variable Documentation****9.109.3.1 internal_data**

`internal_data`

Definition at line 48 of file tensor_filter_custom_easy.c.

9.109.3.2 NNS_support_custom_easy

`GstTensorFilterFramework NNS_support_custom_easy [static]`

Initial value:

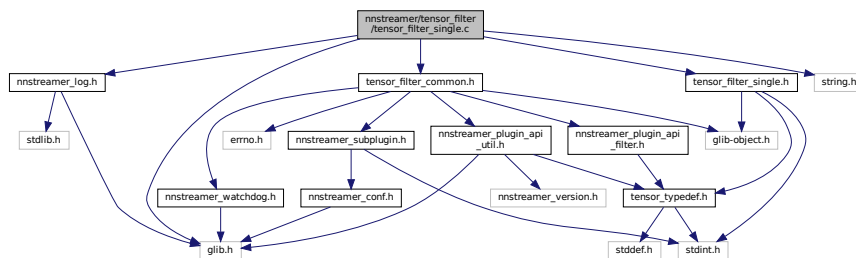
```
= {
  .version = GST_TENSOR_FILTER_FRAMEWORK_V1,
  .open = custom_open,
  .close = custom_close,
  .invoke = custom_invoke,
  .getFrameworkInfo = custom_getFrameworkInfo,
  .getModelInfo = custom_getModelInfo,
  .eventHandler = custom_eventHandler,
  .subplugin_data = NULL,
}
```

Definition at line 332 of file tensor_filter_custom_easy.c.

9.110 nnstreamer/tensor_filter/tensor_filter_single.c File Reference

Element to use general neural network framework directly without gstreamer pipeline.

```
#include <glib.h>
#include <string.h>
#include "nnstreamer_log.h"
#include "tensor_filter_common.h"
#include "tensor_filter_single.h"
Include dependency graph for tensor_filter_single.c:
```



Classes

- [struct `_GTensorFilterSinglePrivate`](#)
Private data struct for tensor-filter single class.

Macros

- `#define G_TENSOR_FILTER_SINGLE_PRIV\(obj\) ((GTensorFilterSinglePrivate *) (obj)->priv)`
- `#define g_tensor_filter_single_parent_class parent_class`

Typedefs

- `typedef struct _GTensorFilterSinglePrivate GTensorFilterSinglePrivate`
Private data struct for tensor-filter single class.

Functions

- `G_DEFINE_TYPE_WITH_PRIVATE (GTensorFilterSingle, g_tensor_filter_single, G_TYPE_OBJECT)`
- `static void g_tensor_filter_single_finalize (GObject *object)`
Function to finalize instance.
- `static void g_tensor_filter_single_set_property (GObject *object, guint prop_id, const GValue *value, GParamSpec *pspec)`
Setter for tensor_filter_single properties.
- `static void g_tensor_filter_single_get_property (GObject *object, guint prop_id, GValue *value, GParamSpec *pspec)`
Getter for tensor_filter_single properties.

- static gboolean `g_tensor_filter_single_invoke` (`GTensorFilterSingle *self`, const `GstTensorMemory *input`, `GstTensorMemory *output`, gboolean allocate)
Called when an input supposed to be invoked.
- static gboolean `g_tensor_filter_input_configured` (`GTensorFilterSingle *self`)
Determine if input is configured (both input and output tensor)
- static gboolean `g_tensor_filter_output_configured` (`GTensorFilterSingle *self`)
Determine if output is configured (both input and output tensor)
- static gint `g_tensor_filter_set_input_info` (`GTensorFilterSingle *self`, const `GstTensorsInfo *in_info`, `GstTensorsInfo *out_info`)
Set input tensor information in the framework.
- static void `g_tensor_filter_destroy_notify` (`GTensorFilterSingle *self`, `GstTensorMemory *mem`)
Called to notify the framework to destroy the allocated memory.
- static gboolean `g_tensor_filter_allocate_in_invoke` (`GTensorFilterSingle *self`)
Determine if this filter framework supports allocation in invoke.
- static gboolean `g_tensor_filter_single_start` (`GTensorFilterSingle *self`)
Called when the element starts processing, if fw not loaded.
- static gboolean `g_tensor_filter_single_stop` (`GTensorFilterSingle *self`)
Called when the element stops processing, if fw loaded.
- static void `g_tensor_filter_single_class_init` (`GTensorFilterSingleClass *klass`)
initialize the tensor_filter's class
- static void `g_tensor_filter_single_init` (`GTensorFilterSingle *self`)
initialize the new element

9.110.1 Detailed Description

Element to use general neural network framework directly without gstreamer pipeline.

Copyright (C) 2019 Parichay Kapoor pk.kapoor@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

28 Aug 2019

See also

<http://github.com/nstreamer/nstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

Bug No known bugs except for NYI items

This is the main element for per-NN-framework plugins. Specific implementations for each NN framework must be written in each framework specific files; e.g., `tensor_filter_tensorflow_lite.c`

9.110.2 Macro Definition Documentation

9.110.2.1 g_tensor_filter_single_parent_class

```
#define g_tensor_filter_single_parent_class parent_class
```

Definition at line 61 of file tensor_filter_single.c.

9.110.2.2 G_TENSOR_FILTER_SINGLE_PRIV

```
#define G_TENSOR_FILTER_SINGLE_PRIV(  
    obj ) ((GTensorFilterSinglePrivate *) (obj)->priv)
```

Definition at line 59 of file tensor_filter_single.c.

9.110.3 Typedef Documentation

9.110.3.1 GTensorFilterSinglePrivate

```
typedef struct _GTensorFilterSinglePrivate GTensorFilterSinglePrivate
```

Private data struct for tensor-filter single class.

SECTION:element-tensor_filter_single

An element that invokes neural network models and their framework or an independent shared object implementing [tensor_filter_custom.h](#). The input and output are always in the format of other/tensor or other/tensors. This element is going to be the basis of single shot api.

9.110.4 Function Documentation

9.110.4.1 G_DEFINE_TYPE_WITH_PRIVATE()

```
G_DEFINE_TYPE_WITH_PRIVATE (  
    GTensorFilterSingle ,  
    g_tensor_filter_single ,  
    G_TYPE_OBJECT )
```

9.110.4.2 `g_tensor_filter_allocate_in_invoke()`

```
static gboolean g_tensor_filter_allocate_in_invoke (
    GTensorFilterSingle * self ) [inline], [static]
```

Determine if this filter framework supports allocation in invoke.

Definition at line 231 of file `tensor_filter_single.c`.

Here is the caller graph for this function:



9.110.4.3 `g_tensor_filter_destroy_notify()`

```
static void g_tensor_filter_destroy_notify (
    GTensorFilterSingle * self,
    GstTensorMemory * mem ) [static]
```

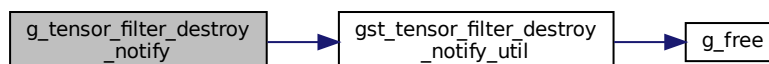
Called to notify the framework to destroy the allocated memory.

Parameters

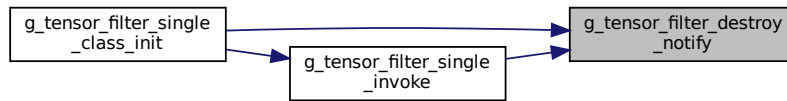
<i>self</i>	"this" pointer
<i>mem</i>	Memory wrapper for the allocated memory by the filter

Definition at line 296 of file `tensor_filter_single.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



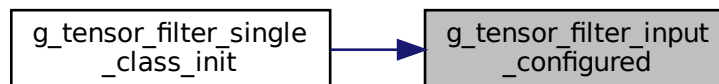
9.110.4.4 g_tensor_filter_input_configured()

```
static gboolean g_tensor_filter_input_configured (
    GTensorFilterSingle * self ) [static]
```

Determine if input is configured (both input and output tensor)

Definition at line 200 of file tensor_filter_single.c.

Here is the caller graph for this function:



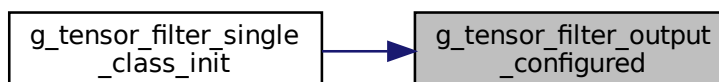
9.110.4.5 g_tensor_filter_output_configured()

```
static gboolean g_tensor_filter_output_configured (
    GTensorFilterSingle * self ) [static]
```

Determine if output is configured (both input and output tensor)

Definition at line 216 of file tensor_filter_single.c.

Here is the caller graph for this function:



9.110.4.6 g_tensor_filter_set_input_info()

```
static gint g_tensor_filter_set_input_info (
    GTensorFilterSingle * self,
    const GstTensorsInfo * in_info,
    GstTensorsInfo * out_info ) [static]
```

Set input tensor information in the framework.

Parameters

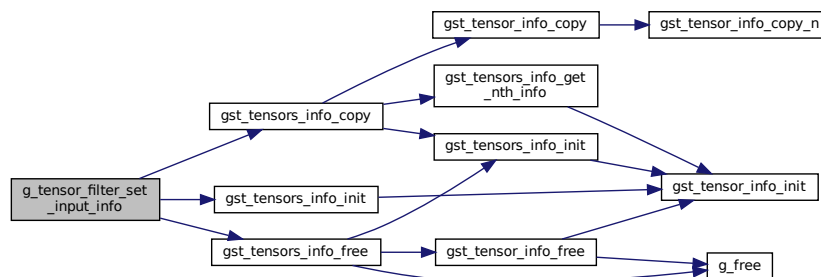
<i>self</i>	"this" pointer
<i>in_info</i>	information on the input tensor
<i>out_info</i>	updated information on the output tensor

Returns

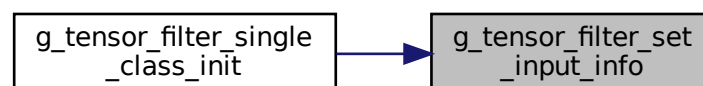
0 for success, -errno for failure.

Definition at line 400 of file tensor_filter_single.c.

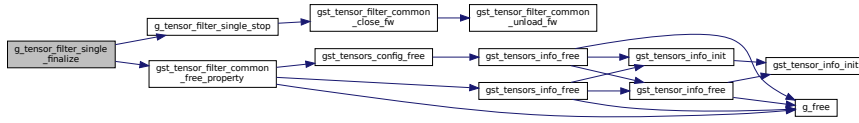
Here is the call graph for this function:



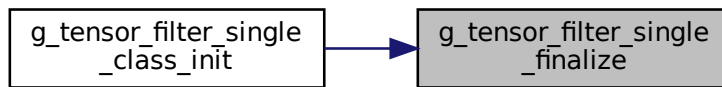
Here is the caller graph for this function:



Here is the call graph for this function:



Here is the caller graph for this function:



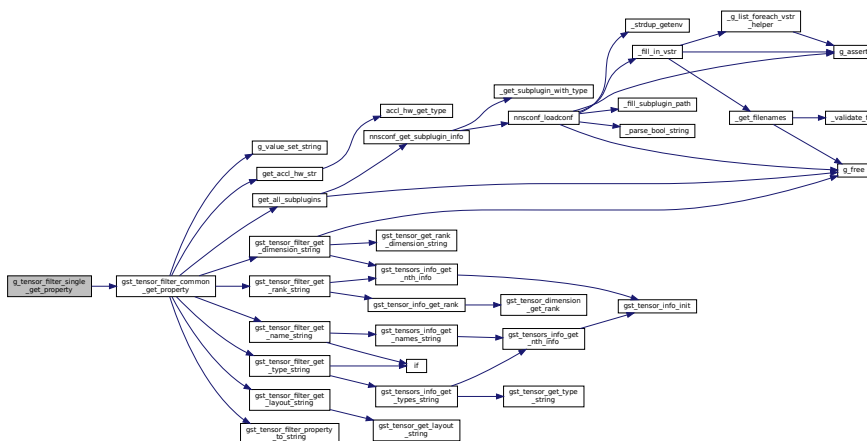
9.110.4.9 g_tensor_filter_single_get_property()

```
static void g_tensor_filter_single_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

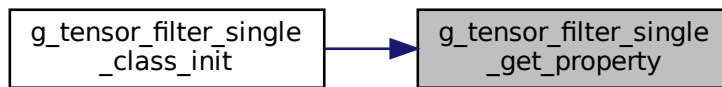
Getter for `tensor_filter_single` properties.

Definition at line 178 of file `tensor_filter_single.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.110.4.10 g_tensor_filter_single_init()

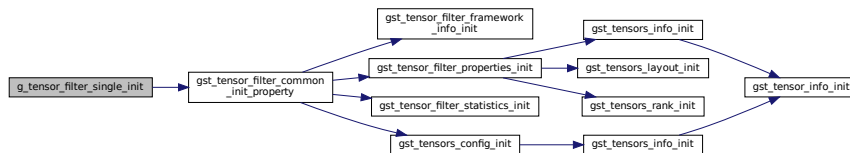
```

static void g_tensor_filter_single_init (
    GTensorFilterSingle * self ) [static]
  
```

initialize the new element

Definition at line 115 of file tensor_filter_single.c.

Here is the call graph for this function:



9.110.4.11 g_tensor_filter_single_invoke()

```

static gboolean g_tensor_filter_single_invoke (
    GTensorFilterSingle * self,
    const GstTensorMemory * input,
    GstTensorMemory * output,
    gboolean allocate ) [static]
  
```

Called when an input supposed to be invoked.

Parameters

<i>self</i>	"this" pointer
<i>input</i>	memory containing input data to run processing on
<i>output</i>	memory to put output data into after processing
<i>allocate</i>	true to allocate output data (false means tensor data is already allocated)

Returns

TRUE if there is no error.

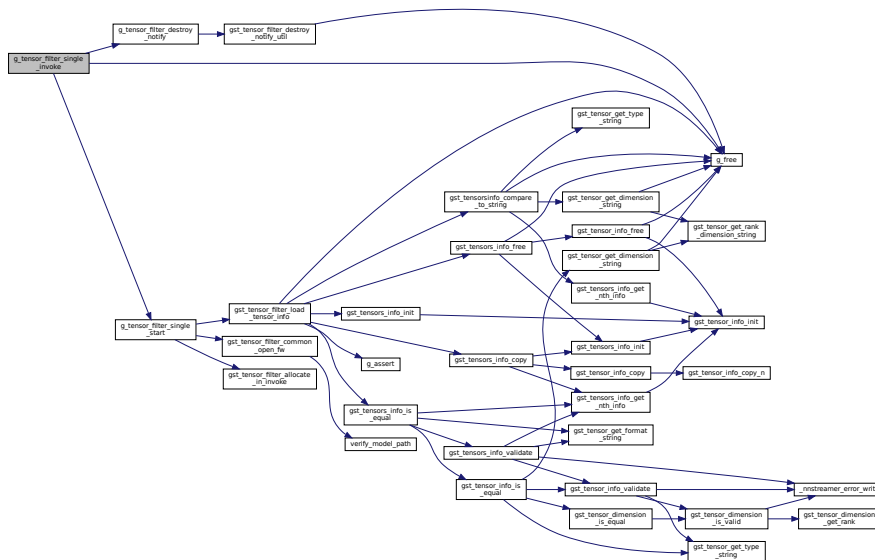
Todo refactor this local variable

start if not already started

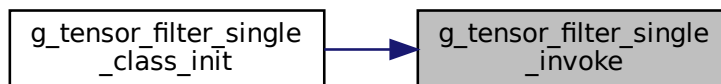
Todo how can we remove memcpy if output data is already allocated single-shot should fill the output data, but sub-plugin allocates new memory.

Definition at line 321 of file tensor_filter_single.c.

Here is the call graph for this function:



Here is the caller graph for this function:



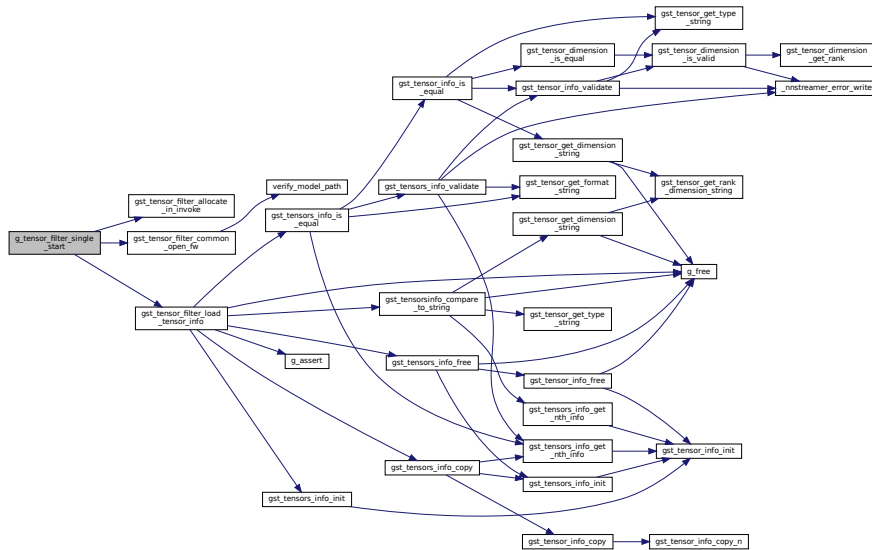
Returns

TRUE if there is no error.

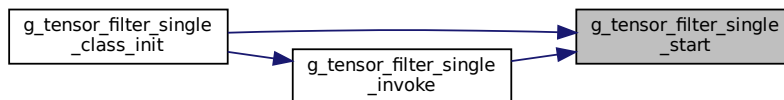
open framework, load model

Definition at line 246 of file tensor_filter_single.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.110.4.14 g_tensor_filter_single_stop()

```
static gboolean g_tensor_filter_single_stop (
    GTensorFilterSingle * self ) [static]
```

Called when the element stops processing, if fw loaded.

Parameters

self	"this" pointer
------	----------------

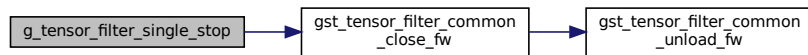
Returns

TRUE if there is no error.

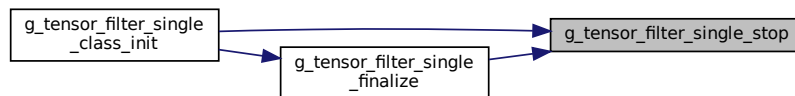
close framework, unload model

Definition at line 277 of file tensor_filter_single.c.

Here is the call graph for this function:

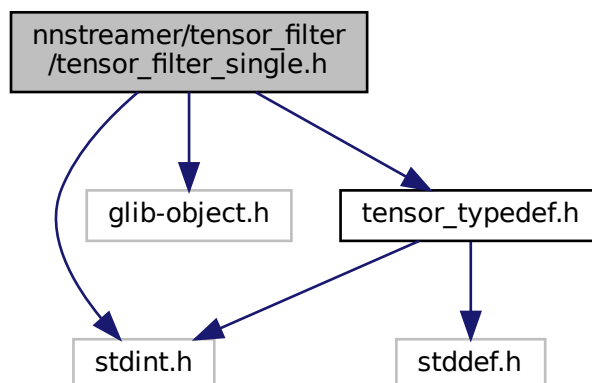


Here is the caller graph for this function:

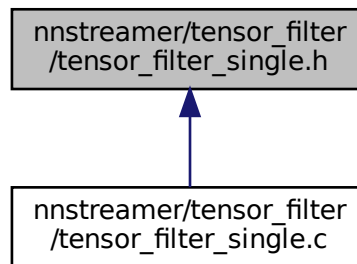
**9.111 nnstreamer/tensor_filter/tensor_filter_single.h File Reference**

Element to use general neural network framework individually without gstreamer pipeline.

```
#include <stdint.h>
#include <glib-object.h>
#include <tensor_typedef.h>
Include dependency graph for tensor_filter_single.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GTensorFilterSingle](#)
Internal data structure for tensor_filter_single instances.
- struct [_GTensorFilterSingleClass](#)
GTensorFilterSingleClass inherits GObjectClass.

Macros

- #define [G_TYPE_TENSOR_FILTER_SINGLE](#) ([g_tensor_filter_single_get_type\(\)](#))
- #define [G_TENSOR_FILTER_SINGLE](#)(obj) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj), [G_TYPE_TENSOR_FILTER_SINGLE](#)))
- #define [G_TENSOR_FILTER_SINGLE_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_CAST](#)((klass), [G_TYPE_TENSOR_FILTER_SINGLE](#)))
- #define [G_IS_TENSOR_FILTER_SINGLE](#)(obj) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj), [G_TYPE_TENSOR_FILTER_SINGLE](#)))
- #define [G_IS_TENSOR_FILTER_SINGLE_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass), [G_TYPE_TENSOR_FILTER_SINGLE](#)))
- #define [G_TENSOR_FILTER_SINGLE_CAST](#)(obj) (([GTensorFilterSingle *](#))(obj))

Typedefs

- typedef struct [_GTensorFilterSingle](#) [GTensorFilterSingle](#)
- typedef struct [_GTensorFilterSingleClass](#) [GTensorFilterSingleClass](#)

Functions

- GType [g_tensor_filter_single_get_type](#) (void)
Get Type function required for gst elements.

9.111.1 Detailed Description

Element to use general neural network framework individually without gstreamer pipeline.

Copyright (C) 2019 Parichay Kapoor pk.kapoor@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

28 Aug 2019

See also

<http://github.com/nnstreamer/nnstreamer>

Author

Parichay Kapoor pk.kapoor@samsung.com

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

9.111.2 Macro Definition Documentation

9.111.2.1 G_IS_TENSOR_FILTER_SINGLE

```
#define G_IS_TENSOR_FILTER_SINGLE(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), G_TYPE_TENSOR_FILTER_SINGLE))
```

Definition at line 39 of file tensor_filter_single.h.

9.111.2.2 G_IS_TENSOR_FILTER_SINGLE_CLASS

```
#define G_IS_TENSOR_FILTER_SINGLE_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), G_TYPE_TENSOR_FILTER_SINGLE))
```

Definition at line 41 of file tensor_filter_single.h.

9.111.2.3 G_TENSOR_FILTER_SINGLE

```
#define G_TENSOR_FILTER_SINGLE(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), G_TYPE_TENSOR_FILTER_SINGLE, GTensorFilterSingle))
```

Definition at line 35 of file tensor_filter_single.h.

9.111.2.4 G_TENSOR_FILTER_SINGLE_CAST

```
#define G_TENSOR_FILTER_SINGLE_CAST(  
    obj ) ((GTensorFilterSingle *) (obj))
```

Definition at line 43 of file tensor_filter_single.h.

9.111.2.5 G_TENSOR_FILTER_SINGLE_CLASS

```
#define G_TENSOR_FILTER_SINGLE_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), G_TYPE_TENSOR_FILTER_SINGLE, GTensorFilterSingleClass))
```

Definition at line 37 of file tensor_filter_single.h.

9.111.2.6 G_TYPE_TENSOR_FILTER_SINGLE

```
#define G_TYPE_TENSOR_FILTER_SINGLE (g_tensor_filter_single_get_type())
```

Definition at line 33 of file tensor_filter_single.h.

9.111.3 Typedef Documentation

9.111.3.1 GTensorFilterSingle

```
typedef struct _GTensorFilterSingle GTensorFilterSingle
```

Definition at line 45 of file tensor_filter_single.h.

9.111.3.2 GTensorFilterSingleClass

```
typedef struct _GTensorFilterSingleClass GTensorFilterSingleClass
```

Definition at line 46 of file tensor_filter_single.h.

9.111.4 Function Documentation

9.111.4.1 g_tensor_filter_single_get_type()

```
GType g_tensor_filter_single_get_type (
    void )
```

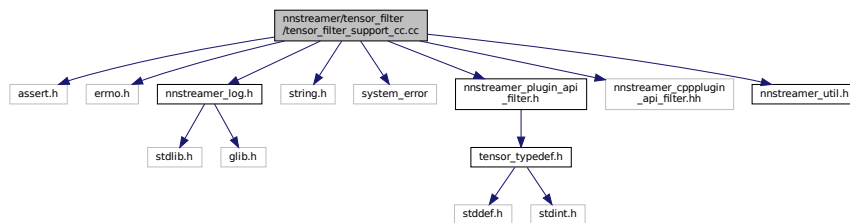
Get Type function required for gst elements.

9.112 nnstreamer/tensor_filter/tensor_filter_support_cc.cc File Reference

Base class for tensor_filter subplugins of C++ classes.

```
#include <assert.h>
#include <errno.h>
#include <nnstreamer_log.h>
#include <string.h>
#include <system_error>
#include <nnstreamer_plugin_api_filter.h>
#include <nnstreamer_cppplugin_api_filter.hh>
#include <nnstreamer_util.h>
```

Include dependency graph for tensor_filter_support_cc.cc:



Namespaces

- [nnstreamer](#)

Macros

- #define `NO_ANONYMOUS_NESTED_STRUCT`
- #define `_SANITY_CHECK` (0xFACE217714DEADE7ULL)
- #define `_RETURN_ERR_WITH_MSG`(c, m)
- #define `GET_TFSP_WITH_CHECKS`(obj, private_data)

9.112.1 Detailed Description

Base class for `tensor_filter` subplugins of C++ classes.

GStreamer `Tensor_filter`, C++ Subplugin Support. (this is not a subplugin) Copyright (C) 2020 MyungJoo Ham myungjoo.ham@samsung.com

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; version 2.1 of the License.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

Date

22 Jan 2020

See also

<http://github.com/nnstreamer/nnstreamer>

Author

MyungJoo Ham myungjoo.ham@samsung.com

Bug No known bugs except for NYI items

This is not a subplugin, but a helper for C++ subplugins. If you want to write a wrapper for neural network frameworks or a hardware adaptor in C++, this is what you want. If you want to attach a single C++ class/object as a filter (a.k.a. custom filter) to a pipeline, use `tensor_filter_cpp`

9.112.2 Macro Definition Documentation

9.112.2.1 `_RETURN_ERR_WITH_MSG`

```
#define _RETURN_ERR_WITH_MSG(
    c,
    m )
```

Value:

```
do {
    nns_loge ("%s", m);
    return c;
} while (0);
```

Definition at line 54 of file `tensor_filter_support_cc.cc`.

9.112.2.2 `_SANITY_CHECK`

```
#define _SANITY_CHECK (0xFACE217714DEADE7ULL)
```

Definition at line 53 of file `tensor_filter_support_cc.cc`.

9.112.2.3 `GET_TFSP_WITH_CHECKS`

```
#define GET_TFSP_WITH_CHECKS(  
    obj,  
    private_data )
```

Value:

```
do {  
    try {  
        obj = get_tfsp_with_checks (private_data);  
    } catch (const std::exception &e) {  
        return -EINVAL;  
    }  
} while (0);
```

```
\\  
\\  
\\  
\\
```

Definition at line 60 of file `tensor_filter_support_cc.cc`.

9.112.2.4 `NO_ANONYMOUS_NESTED_STRUCT`

```
#define NO_ANONYMOUS_NESTED_STRUCT
```

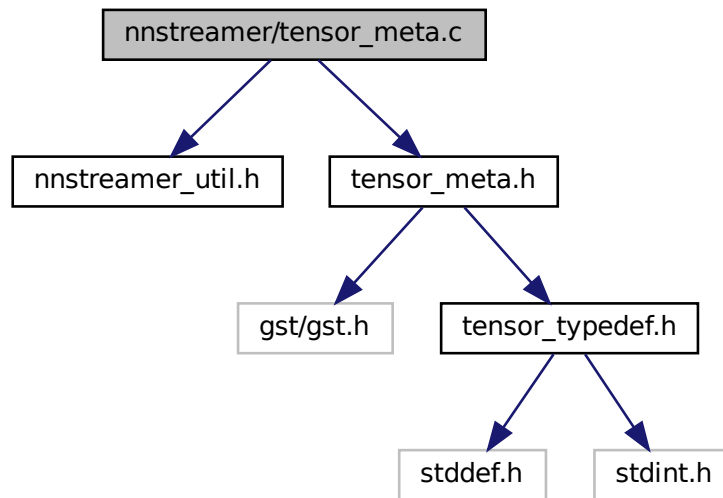
Definition at line 39 of file `tensor_filter_support_cc.cc`.

9.113 nnstreamer/tensor_meta.c File Reference

Internal tensor meta implementation for nnstreamer.

```
#include <nnstreamer_util.h>  
#include "tensor_meta.h"
```

Include dependency graph for `tensor_meta.c`:



Functions

- GType `gst_meta_query_api_get_type` (void)
Define meta_query type to register.
- static gboolean `gst_meta_query_init` (GstMeta *meta, gpointer params, GstBuffer *buffer)
meta_query init
- static void `gst_meta_query_free` (GstMeta *meta, GstBuffer *buffer)
free meta_query
- static gboolean `gst_meta_query_transform` (GstBuffer *transbuf, GstMeta *meta, GstBuffer *buffer, GQuark type, gpointer data)
tensor_query meta data transform (source to dest)
- const GstMetaInfo * `gst_meta_query_get_info` (void)
Get meta_query info.

9.113.1 Detailed Description

Internal tensor meta implementation for nnstreamer.

Copyright (C) 2021 Junhwan Kim jejudo.kim@samsung.com

Date

09 Aug 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.113.2 Function Documentation

9.113.2.1 `gst_meta_query_api_get_type()`

```
GType gst_meta_query_api_get_type (  
    void )
```

Define meta_query type to register.

Definition at line 24 of file tensor_meta.c.

9.113.2.2 `gst_meta_query_free()`

```
static void gst_meta_query_free (  
    GstMeta * meta,  
    GstBuffer * buffer ) [static]
```

free meta_query

Definition at line 61 of file tensor_meta.c.

Here is the caller graph for this function:



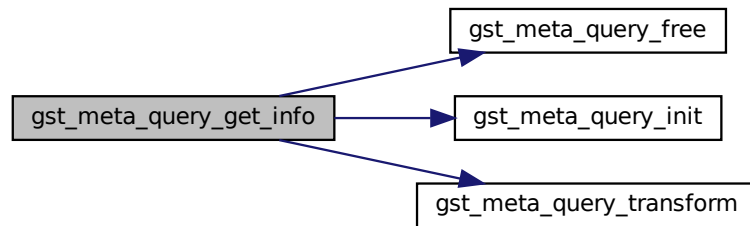
9.113.2.3 `gst_meta_query_get_info()`

```
const GstMetaInfo* gst_meta_query_get_info (  
    void )
```

Get meta_query info.

Definition at line 87 of file tensor_meta.c.

Here is the call graph for this function:



9.113.2.4 `gst_meta_query_init()`

```

static gboolean gst_meta_query_init (
    GstMeta * meta,
    gpointer params,
    GstBuffer * buffer ) [static]
  
```

`meta_query` init

Definition at line 48 of file `tensor_meta.c`.

Here is the caller graph for this function:



9.113.2.5 `gst_meta_query_transform()`

```

static gboolean gst_meta_query_transform (
    GstBuffer * transbuf,
    GstMeta * meta,
    GstBuffer * buffer,
    GQuark type,
    gpointer data ) [static]
  
```

tensor_query meta data transform (source to dest)

Definition at line 71 of file tensor_meta.c.

Here is the caller graph for this function:

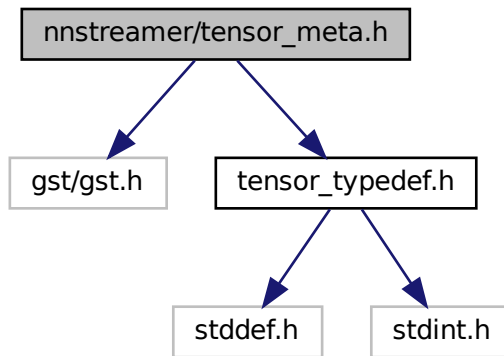


9.114 nnstreamer/tensor_meta.h File Reference

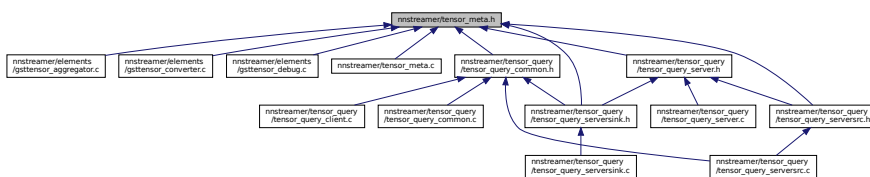
Internal tensor meta header for nnstreamer.

```

#include <gst/gst.h>
#include <tensor_typedef.h>
Include dependency graph for tensor_meta.h:
  
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [GstMetaQuery](#)
GstMetaQuery meta structure.

Macros

- #define [GST_META_QUERY_API_TYPE](#) ([gst_meta_query_api_get_type](#)())
- #define [GST_META_QUERY_INFO](#) ([gst_meta_query_get_info](#)())
- #define [gst_buffer_get_meta_query](#)(b) (([GstMetaQuery](#) *) [gst_buffer_get_meta](#) ((b), [GST_META_QUERY_API_TYPE](#)))
- #define [gst_buffer_add_meta_query](#)(b) (([GstMetaQuery](#) *) [gst_buffer_add_meta](#) ((b), [GST_META_QUERY_INFO](#), NULL))

Functions

- GType [gst_meta_query_api_get_type](#) (void)
Define meta_query type to register.
- const [GstMetaInfo](#) * [gst_meta_query_get_info](#) (void)
Get meta_query info.

Variables

- [G_BEGIN_DECLS](#) typedef int64_t [query_client_id_t](#)

9.114.1 Detailed Description

Internal tensor meta header for nnstreamer.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

09 Aug 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.114.2 Macro Definition Documentation

9.114.2.1 `gst_buffer_add_meta_query`

```
#define gst_buffer_add_meta_query(  
    b ) ((GstMetaQuery *) gst_buffer_add_meta ((b), GST_META_QUERY_INFO, NULL))
```

Definition at line 46 of file `tensor_meta.h`.

9.114.2.2 `gst_buffer_get_meta_query`

```
#define gst_buffer_get_meta_query(  
    b ) ((GstMetaQuery *) gst_buffer_get_meta ((b), GST_META_QUERY_API_TYPE))
```

Definition at line 44 of file `tensor_meta.h`.

9.114.2.3 `GST_META_QUERY_API_TYPE`

```
#define GST_META_QUERY_API_TYPE (gst_meta_query_api_get_type())
```

Definition at line 37 of file `tensor_meta.h`.

9.114.2.4 `GST_META_QUERY_INFO`

```
#define GST_META_QUERY_INFO (gst_meta_query_get_info())
```

Definition at line 43 of file `tensor_meta.h`.

9.114.3 Function Documentation

9.114.3.1 `gst_meta_query_api_get_type()`

```
GType gst_meta_query_api_get_type (  
    void )
```

Define `meta_query` type to register.

Definition at line 24 of file `tensor_meta.c`.

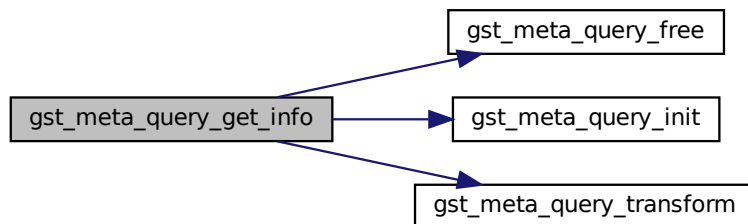
9.114.3.2 `gst_meta_query_get_info()`

```
const GstMetaInfo* gst_meta_query_get_info (  
    void )
```

Get meta_query info.

Definition at line 87 of file `tensor_meta.c`.

Here is the call graph for this function:



9.114.4 Variable Documentation

9.114.4.1 `query_client_id_t`

```
G_BEGIN_DECLS typedef int64_t query_client_id_t
```

Definition at line 21 of file `tensor_meta.h`.

9.115 `nnstreamer/tensor_query/tensor_query_client.c` File Reference

GStreamer plugin to handle tensor query client.

```
#include "nnstreamer_util.h"  
#include "tensor_query_client.h"  
#include <gio/gio.h>  
#include <glib.h>  
#include <string.h>  
#include "tensor_query_common.h"  
#include <stdio.h>  
#include <stdlib.h>
```


- This function handles sink event.*

 - static gboolean [gst_tensor_query_client_sink_query](#) (GstPad *pad, GstObject *parent, GstQuery *query)

This function handles sink pad query.

 - static GstFlowReturn [gst_tensor_query_client_chain](#) (GstPad *pad, GstObject *parent, GstBuffer *buf)

Chain function, this function does the actual processing.

 - static GstCaps * [gst_tensor_query_client_query_caps](#) (GstTensorQueryClient *self, GstPad *pad, GstCaps *filter)

Get pad caps for caps negotiation.

 - static void [gst_tensor_query_client_class_init](#) (GstTensorQueryClientClass *klass)

initialize the class

 - static void [gst_tensor_query_client_init](#) (GstTensorQueryClient *self)

initialize the new element

 - static gboolean [gst_tensor_query_client_update_caps](#) (GstTensorQueryClient *self, const gchar *caps_str)

Update src pad caps from tensors config.

 - static gchar * [_nns_edge_parse_caps](#) (gchar *caps_str, gboolean is_src)

Parse caps from received event data.

 - static int [_nns_edge_event_cb](#) (nns_edge_event_h event_h, void *user_data)

nnstreamer-edge event callback.

 - static gboolean [gst_tensor_query_client_create_edge_handle](#) (GstTensorQueryClient *self)

Internal function to create edge handle.

Variables

- static GstStaticPadTemplate [sinktemplate](#)
the capabilities of the inputs.
- static GstStaticPadTemplate [srctemplate](#)
the capabilities of the outputs.

9.115.1 Detailed Description

GStreamer plugin to handle tensor query client.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

09 Jul 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.115.2 Macro Definition Documentation

9.115.2.1 DBG

```
#define DBG (!self->silent)
```

Macro for debug mode.

Definition at line 32 of file tensor_query_client.c.

9.115.2.2 DEFAULT_CLIENT_TIMEOUT

```
#define DEFAULT_CLIENT_TIMEOUT 0
```

Definition at line 56 of file tensor_query_client.c.

9.115.2.3 DEFAULT_MAX_REQUEST

```
#define DEFAULT_MAX_REQUEST 2
```

Definition at line 58 of file tensor_query_client.c.

9.115.2.4 DEFAULT_SILENT

```
#define DEFAULT_SILENT TRUE
```

Definition at line 57 of file tensor_query_client.c.

9.115.2.5 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_query_client_debug
```

Definition at line 61 of file tensor_query_client.c.

9.115.2.6 `gst_tensor_query_client_parent_class`

```
#define gst_tensor_query_client_parent_class parent_class
```

Definition at line 79 of file `tensor_query_client.c`.

9.115.2.7 `TCP_DEFAULT_CLIENT_SRC_PORT`

```
#define TCP_DEFAULT_CLIENT_SRC_PORT 3001
```

Definition at line 55 of file `tensor_query_client.c`.

9.115.2.8 `TCP_DEFAULT_HOST`

```
#define TCP_DEFAULT_HOST "localhost"
```

Definition at line 53 of file `tensor_query_client.c`.

9.115.2.9 `TCP_DEFAULT_SRV_SRC_PORT`

```
#define TCP_DEFAULT_SRV_SRC_PORT 3000
```

Definition at line 54 of file `tensor_query_client.c`.

9.115.2.10 `TCP_HIGHEST_PORT`

```
#define TCP_HIGHEST_PORT 65535
```

Definition at line 52 of file `tensor_query_client.c`.

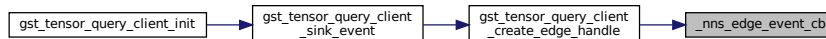
9.115.3 Enumeration Type Documentation

9.115.3.1 anonymous enum

```
anonymous enum
```

Properties.

Here is the caller graph for this function:



9.115.4.2 `_nns_edge_parse_caps()`

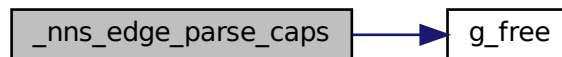
```

static gchar* _nns_edge_parse_caps (
    gchar * caps_str,
    gboolean is_src ) [static]
  
```

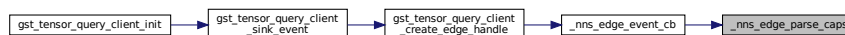
Parse caps from received event data.

Definition at line 400 of file tensor_query_client.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.115.4.3 `G_DEFINE_TYPE()`

```

G_DEFINE_TYPE (
    GstTensorQueryClient ,
    gst_tensor_query_client ,
    GST_TYPE_ELEMENT )
  
```

9.115.4.4 GST_DEBUG_CATEGORY_STATIC()

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_query_client_debug )
```

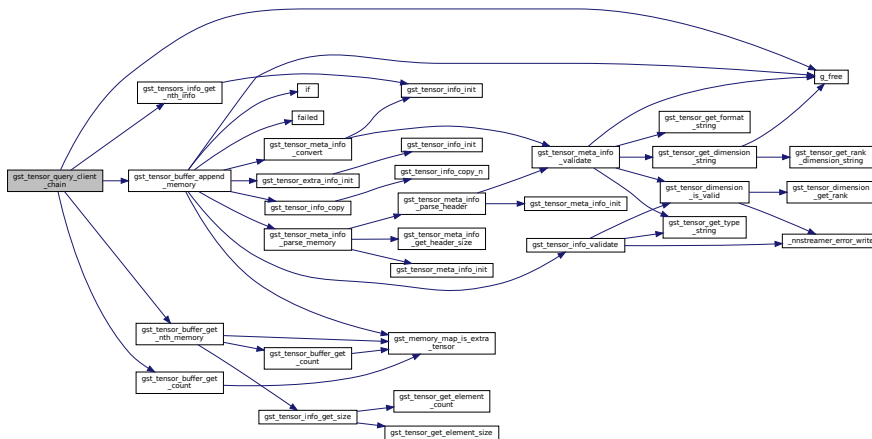
9.115.4.5 gst_tensor_query_client_chain()

```
static GstFlowReturn gst_tensor_query_client_chain (
    GstPad * pad,
    GstObject * parent,
    GstBuffer * buf ) [static]
```

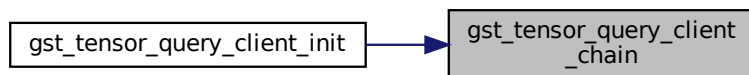
Chain function, this function does the actual processing.

Definition at line 674 of file tensor_query_client.c.

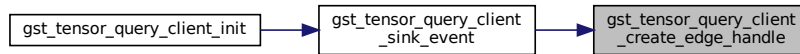
Here is the call graph for this function:



Here is the caller graph for this function:



Here is the caller graph for this function:



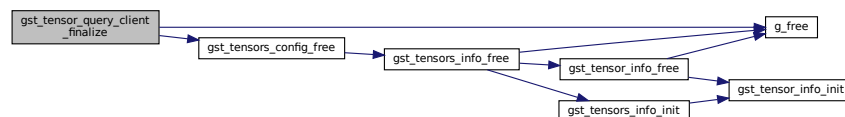
9.115.4.8 `gst_tensor_query_client_finalize()`

```
static void gst_tensor_query_client_finalize (
    GObject * object ) [static]
```

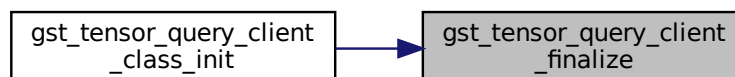
finalize the object

Definition at line 213 of file `tensor_query_client.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



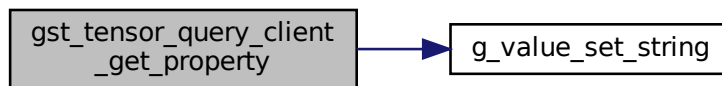
9.115.4.9 `gst_tensor_query_client_get_property()`

```
static void gst_tensor_query_client_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

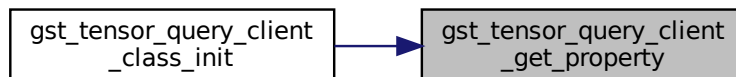
get property

Definition at line 309 of file `tensor_query_client.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.115.4.10 `gst_tensor_query_client_init()`

```
static void gst_tensor_query_client_init (
    GstTensorQueryClient * self ) [static]
```

initialize the new element

setup sink pad

setup src pad

Definition at line 175 of file `tensor_query_client.c`.

9.115.4.12 `gst_tensor_query_client_set_property()`

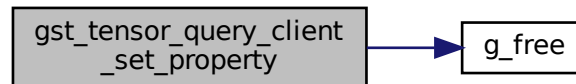
```
static void gst_tensor_query_client_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

set property

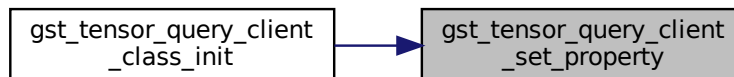
Todo DO NOT update properties (host, port, ..) while pipeline is running.

Definition at line 250 of file `tensor_query_client.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



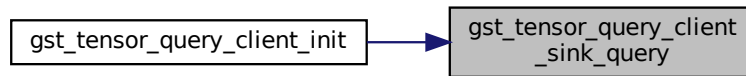
9.115.4.13 `gst_tensor_query_client_sink_event()`

```
static gboolean gst_tensor_query_client_sink_event (
    GstPad * pad,
    GstObject * parent,
    GstEvent * event ) [static]
```

This function handles sink event.

Definition at line 587 of file `tensor_query_client.c`.

Here is the caller graph for this function:



9.115.4.15 gst_tensor_query_client_update_caps()

```

static gboolean gst_tensor_query_client_update_caps (
    GstTensorQueryClient * self,
    const gchar * caps_str ) [static]
  
```

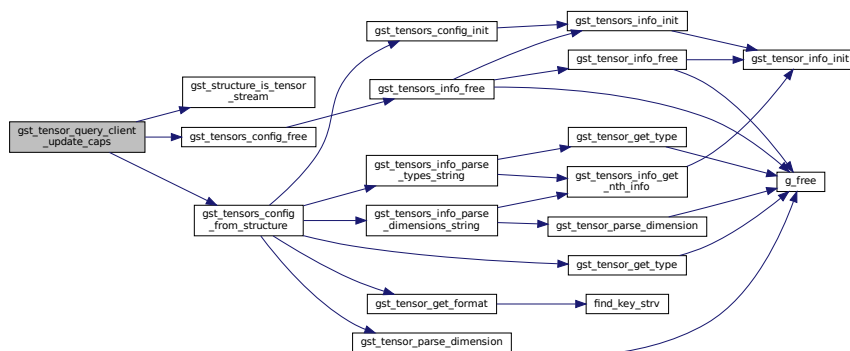
Update src pad caps from tensors config.

If pad caps is updated, prepare the tensor information here. It will be used in chain function, to push tensor buffer into src pad.

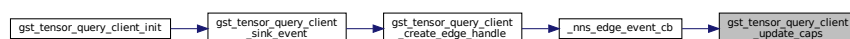
Don't need to update when the capability is the same.

Definition at line 352 of file tensor_query_client.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.115.5 Variable Documentation

9.115.5.1 sinktemplate

```
GstStaticPadTemplate sinktemplate [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS_ANY)
```

the capabilities of the inputs.

Definition at line 66 of file tensor_query_client.c.

9.115.5.2 srctemplate

```
GstStaticPadTemplate srctemplate [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS_ANY)
```

the capabilities of the outputs.

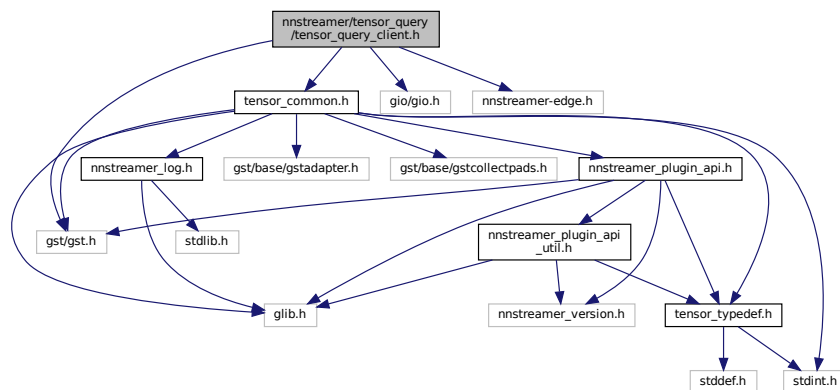
Definition at line 74 of file tensor_query_client.c.

9.116 nnstreamer/tensor_query/tensor_query_client.h File Reference

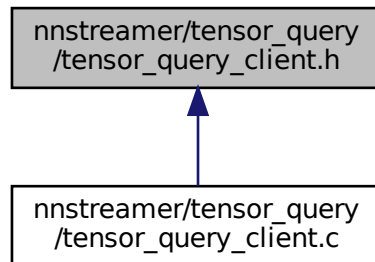
GStreamer plugin to handle tensor query client.

```
#include <gst/gst.h>
#include <gio/gio.h>
#include <tensor_common.h>
#include <nnstreamer-edge.h>
```

Include dependency graph for tensor_query_client.h:



This graph shows which files directly or indirectly include this file:



Classes

- [struct `_GstTensorQueryClient`](#)
GstTensorQueryClient data structure.
- [struct `_GstTensorQueryClientClass`](#)
GstTensorQueryClientClass data structure.

Macros

- [#define `GST_TYPE_TENSOR_QUERY_CLIENT`](#) ([gst_tensor_query_client_get_type\(\)](#))
- [#define `GST_TENSOR_QUERY_CLIENT\(obj\)`](#) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj),[GST_TYPE_TENSOR_QUERY_CLIENT](#)))
- [#define `GST_TENSOR_QUERY_CLIENT_CLASS\(klass\)`](#) ([G_TYPE_CHECK_CLASS_CAST](#)((klass),[GST_TYPE_TENSOR_QUERY_CLIENT](#)))
- [#define `GST_IS_TENSOR_QUERY_CLIENT\(obj\)`](#) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_QUERY_CLIENT](#)))
- [#define `GST_IS_TENSOR_QUERY_CLIENT_CLASS\(klass\)`](#) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_QUERY_CLIENT](#)))
- [#define `GST_TENSOR_QUERY_CLIENT_CAST\(obj\)`](#) ([\(\(GstTensorQueryClient *\) \(obj\)\)](#))

Typedefs

- [typedef struct `_GstTensorQueryClient`](#) [GstTensorQueryClient](#)
- [typedef struct `_GstTensorQueryClientClass`](#) [GstTensorQueryClientClass](#)

Functions

- GType [gst_tensor_query_client_get_type](#) (void)

9.116.1 Detailed Description

GStreamer plugin to handle tensor query client.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

09 Jul 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.116.2 Macro Definition Documentation

9.116.2.1 GST_IS_TENSOR_QUERY_CLIENT

```
#define GST_IS_TENSOR_QUERY_CLIENT(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_QUERY_CLIENT))
```

Definition at line 29 of file tensor_query_client.h.

9.116.2.2 GST_IS_TENSOR_QUERY_CLIENT_CLASS

```
#define GST_IS_TENSOR_QUERY_CLIENT_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_QUERY_CLIENT))
```

Definition at line 31 of file tensor_query_client.h.

9.116.2.3 GST_TENSOR_QUERY_CLIENT

```
#define GST_TENSOR_QUERY_CLIENT(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_QUERY_CLIENT, GstTensorQueryClient))
```

Definition at line 25 of file tensor_query_client.h.

9.116.2.4 GST_TENSOR_QUERY_CLIENT_CAST

```
#define GST_TENSOR_QUERY_CLIENT_CAST(  
    obj ) ((GstTensorQueryClient *) (obj))
```

Definition at line 33 of file tensor_query_client.h.

9.116.2.5 GST_TENSOR_QUERY_CLIENT_CLASS

```
#define GST_TENSOR_QUERY_CLIENT_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_QUERY_CLIENT, GstTensorQueryClientClass))
```

Definition at line 27 of file tensor_query_client.h.

9.116.2.6 GST_TYPE_TENSOR_QUERY_CLIENT

```
#define GST_TYPE_TENSOR_QUERY_CLIENT (gst_tensor_query_client_get_type())
```

Definition at line 23 of file tensor_query_client.h.

9.116.3 Typedef Documentation

9.116.3.1 GstTensorQueryClient

```
typedef struct _GstTensorQueryClient GstTensorQueryClient
```

Definition at line 35 of file tensor_query_client.h.

9.116.3.2 GstTensorQueryClientClass

```
typedef struct _GstTensorQueryClientClass GstTensorQueryClientClass
```

Definition at line 36 of file tensor_query_client.h.

9.116.4 Function Documentation

9.116.4.1 `gst_tensor_query_client_get_type()`

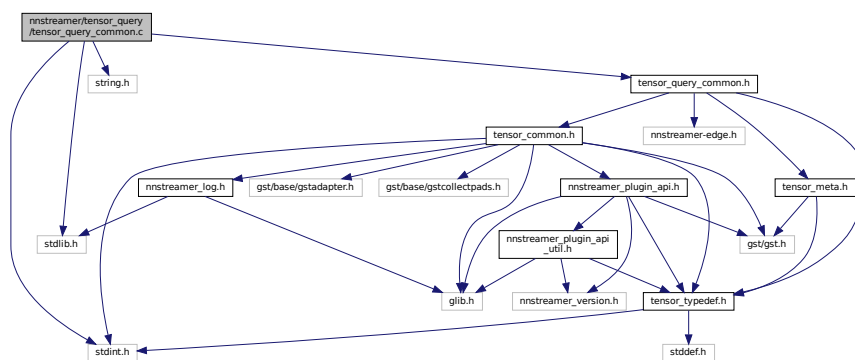
```
GType gst_tensor_query_client_get_type (
    void )
```

9.117 nnstreamer/tensor_query/tensor_query_common.c File Reference

Utility functions for tensor query.

```
#include <stdint.h>
#include <stdlib.h>
#include <string.h>
#include "tensor_query_common.h"
```

Include dependency graph for `tensor_query_common.c`:



Macros

- `#define EREMOTEIO 121 /* This is Linux-specific. Define this for non-Linux systems */`

Functions

- GType `gst_tensor_query_get_connect_type` (void)
Register GEnumValue array for query connect-type property.

9.117.1 Detailed Description

Utility functions for tensor query.

Copyright (C) 2021 Gichan Jang gichan2.jang@samsung.com

Date

09 July 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Junhwan Kim jejudo.kim@samsung.com

Bug No known bugs except for NYI items

9.117.2 Macro Definition Documentation

9.117.2.1 EREMOTEIO

```
#define EREMOTEIO 121 /* This is Linux-specific. Define this for non-Linux systems */
```

Definition at line 23 of file tensor_query_common.c.

9.117.3 Function Documentation

9.117.3.1 gst_tensor_query_get_connect_type()

```
GType gst_tensor_query_get_connect_type (
    void )
```

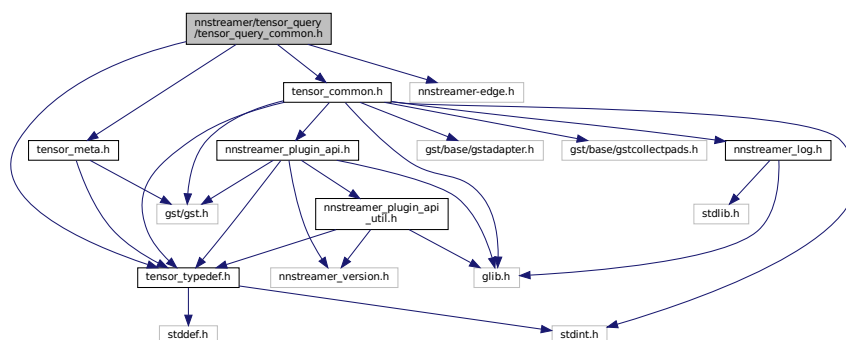
Register GEnumValue array for query connect-type property.

Definition at line 30 of file tensor_query_common.c.

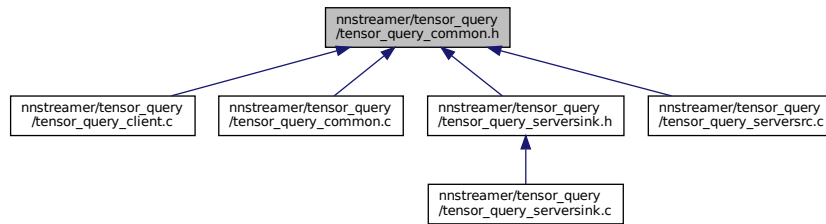
9.118 nnstreamer/tensor_query/tensor_query_common.h File Reference

Utility functions for tensor query.

```
#include "tensor_typedef.h"
#include "tensor_common.h"
#include "tensor_meta.h"
#include <nnstreamer-edge.h>
Include dependency graph for tensor_query_common.h:
```



This graph shows which files directly or indirectly include this file:



Macros

- #define `QUERY_DEFAULT_TIMEOUT_SEC` 10
Default timeout, in seconds.
- #define `DEFAULT_HOST` "localhost"
protocol options for tensor query.
- #define `DEFAULT_CONNECT_TYPE` (NNS_EDGE_CONNECT_TYPE_TCP)
- #define `GST_TYPE_QUERY_CONNECT_TYPE` (`gst_tensor_query_get_connect_type` ())

Functions

- GType `gst_tensor_query_get_connect_type` (void)
Register GEnumValue array for query connect-type property.

9.118.1 Detailed Description

Utility functions for tensor query.

Copyright (C) 2021 Gichan Jang gichan2.jang@samsung.com

Date

09 July 2021

See also

<https://github.com/nnstreamer/nnstreamer>

Author

Gichan Jang gichan2.jang@samsung.com

Bug No known bugs except for NYI items

9.118.2 Macro Definition Documentation

9.118.2.1 DEFAULT_CONNECT_TYPE

```
#define DEFAULT_CONNECT_TYPE (NNS_EDGE_CONNECT_TYPE_TCP)
```

Definition at line 34 of file tensor_query_common.h.

9.118.2.2 DEFAULT_HOST

```
#define DEFAULT_HOST "localhost"
```

protocol options for tensor query.

Definition at line 33 of file tensor_query_common.h.

9.118.2.3 GST_TYPE_QUERY_CONNECT_TYPE

```
#define GST_TYPE_QUERY_CONNECT_TYPE (gst_tensor_query_get_connect_type ())
```

Definition at line 35 of file tensor_query_common.h.

9.118.2.4 QUERY_DEFAULT_TIMEOUT_SEC

```
#define QUERY_DEFAULT_TIMEOUT_SEC 10
```

Default timeout, in seconds.

Definition at line 28 of file tensor_query_common.h.

9.118.3 Function Documentation

9.118.3.1 gst_tensor_query_get_connect_type()

```
GType gst_tensor_query_get_connect_type (  
    void )
```

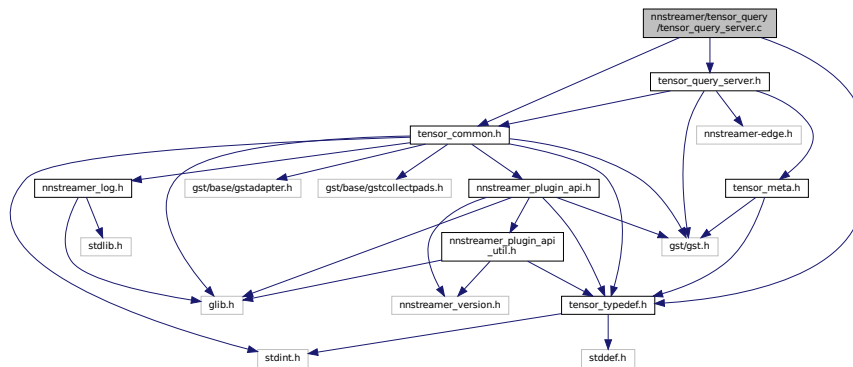
Register GEnumValue array for query connect-type property.

Definition at line 30 of file tensor_query_common.c.

9.119 nnstreamer/tensor_query/tensor_query_server.c File Reference

GStreamer plugin to handle meta_query for server elements.

```
#include "tensor_query_server.h"
#include <tensor_typedef.h>
#include <tensor_common.h>
Include dependency graph for tensor_query_server.c:
```



Functions

- [G_LOCK_DEFINE_STATIC](#) (`query_server_table`)
mutex for tensor-query server table.
- static void [init_queryserver](#) (void)
Internal function to release query server data.
- static [GstTensorQueryServer](#) * [gst_tensor_query_server_get_handle](#) (const guint id)
Get nnstreamer edge server handle.
- gboolean [gst_tensor_query_server_add_data](#) (const guint id)
Add nnstreamer edge server handle into hash table.
- gboolean [gst_tensor_query_server_prepare](#) (const guint id, `nns_edge_connect_type_e` `connect_type`, [GstTensorQueryEdgeInfo](#) *`edge_info`)
Prepare edge connection and its handle.
- gboolean [gst_tensor_query_server_send_buffer](#) (const guint id, `GstBuffer` *`buffer`)
Send buffer to connected edge device.
- void [gst_tensor_query_server_release_edge_handle](#) (const guint id)
Release nnstreamer edge handle of query server.
- void [gst_tensor_query_server_remove_data](#) (const guint id)
Remove [GstTensorQueryServer](#).
- gboolean [gst_tensor_query_server_wait_sink](#) (const guint id)
Wait until the sink is configured and get server info handle.
- void [gst_tensor_query_server_set_configured](#) (const guint id)
set query server sink configured.
- void [gst_tensor_query_server_set_caps](#) (const guint id, const `gchar` *`caps_str`)
set query server caps.
- static void [fini_queryserver](#) (void)
Destruct the query server.

Variables

- static GHashTable * `_qs_table` = NULL
Table for query server data.

9.119.1 Detailed Description

GStreamer plugin to handle `meta_query` for server elements.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

03 Aug 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.119.2 Function Documentation

9.119.2.1 `fini_queryserver()`

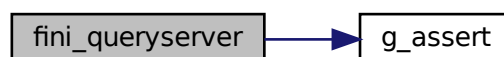
```
static void fini_queryserver (  
    void ) [static]
```

Destruct the query server.

Internal error (init not called?)

Definition at line 385 of file `tensor_query_server.c`.

Here is the call graph for this function:



9.119.2.2 G_LOCK_DEFINE_STATIC()

```
G_LOCK_DEFINE_STATIC (
    query_server_table )
```

mutex for tensor-query server table.

9.119.2.3 gst_tensor_query_server_add_data()

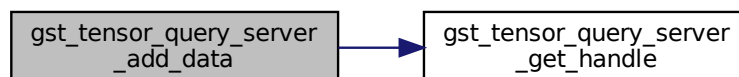
```
gboolean gst_tensor_query_server_add_data (
    const guint id )
```

Add nntstreamer edge server handle into hash table.

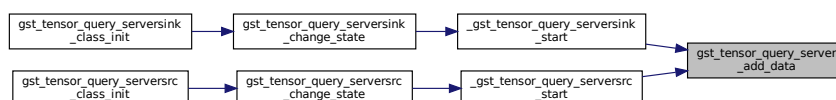
Add [GstTensorQueryServer](#).

Definition at line 77 of file tensor_query_server.c.

Here is the call graph for this function:



Here is the caller graph for this function:



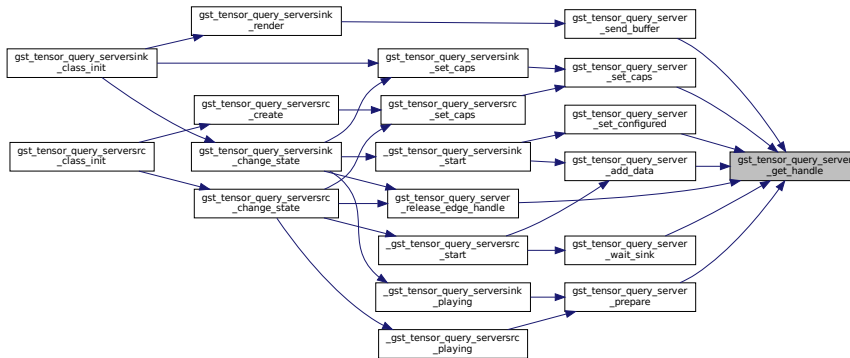
9.119.2.4 `gst_tensor_query_server_get_handle()`

```
static GstTensorQueryServer* gst_tensor_query_server_get_handle (
    const guint id ) [static]
```

Get nnstreamer edge server handle.

Definition at line 62 of file `tensor_query_server.c`.

Here is the caller graph for this function:



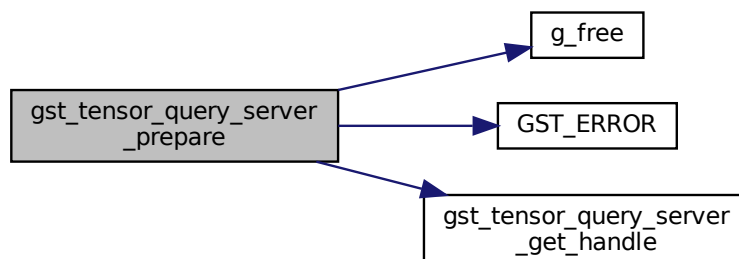
9.119.2.5 `gst_tensor_query_server_prepare()`

```
gboolean gst_tensor_query_server_prepare (
    const guint id,
    nns_edge_connect_type_e connect_type,
    GstTensorQueryEdgeInfo * edge_info )
```

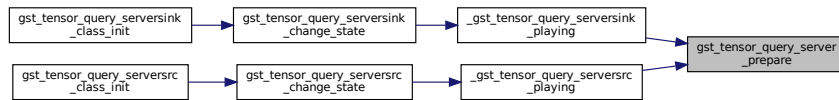
Prepare edge connection and its handle.

Definition at line 114 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



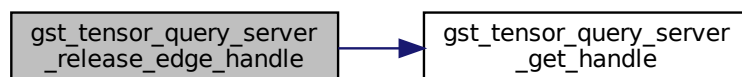
9.119.2.6 `gst_tensor_query_server_release_edge_handle()`

```
void gst_tensor_query_server_release_edge_handle (
    const guint id )
```

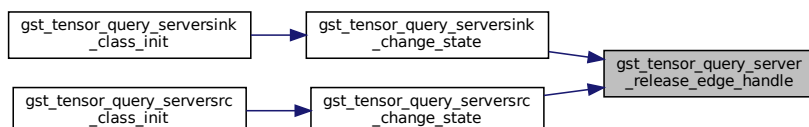
Release nntstreamer edge handle of query server.

Definition at line 258 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



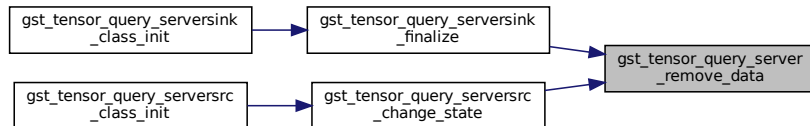
9.119.2.7 `gst_tensor_query_server_remove_data()`

```
void gst_tensor_query_server_remove_data (
    const guint id )
```

Remove [GstTensorQueryServer](#).

Definition at line 280 of file `tensor_query_server.c`.

Here is the caller graph for this function:



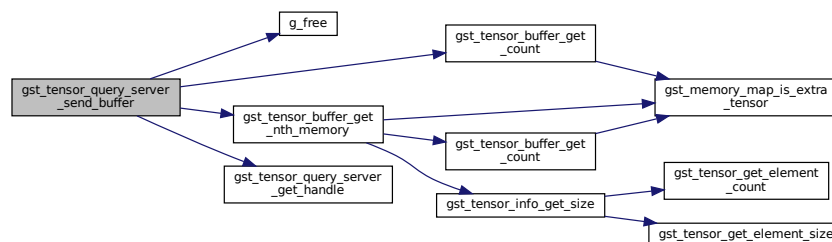
9.119.2.8 `gst_tensor_query_server_send_buffer()`

```
gboolean gst_tensor_query_server_send_buffer (
    const guint id,
    GstBuffer * buffer )
```

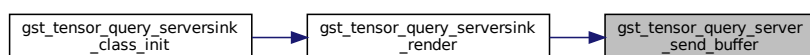
Send buffer to connected edge device.

Definition at line 183 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



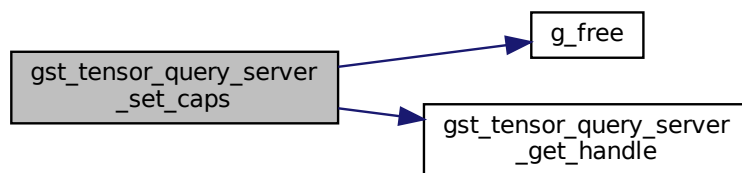
9.119.2.9 `gst_tensor_query_server_set_caps()`

```
void gst_tensor_query_server_set_caps (
    const guint id,
    const gchar * caps_str )
```

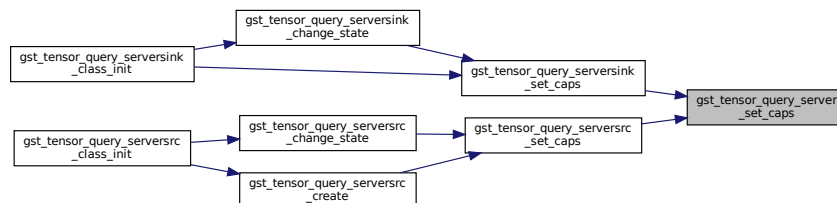
set query server caps.

Definition at line 342 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



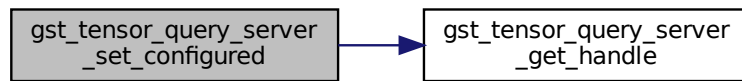
9.119.2.10 `gst_tensor_query_server_set_configured()`

```
void gst_tensor_query_server_set_configured (
    const guint id )
```

set query server sink configured.

Definition at line 322 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



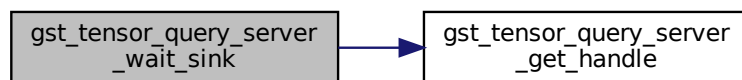
9.119.2.11 `gst_tensor_query_server_wait_sink()`

```
gboolean gst_tensor_query_server_wait_sink (
    const guint id )
```

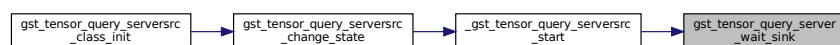
Wait until the sink is configured and get server info handle.

Definition at line 292 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.119.2.12 init_queryserver()

```
static void init_queryserver (
    void ) [static]
```

Internal function to release query server data.

Initialize the query server. Internal error (duplicated init call?)

Definition at line 31 of file tensor_query_server.c.

9.119.3 Variable Documentation

9.119.3.1 _qs_table

```
GHashTable* _qs_table = NULL [static]
```

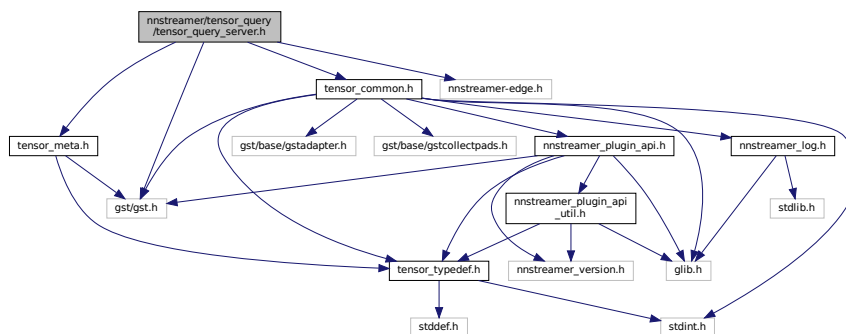
Table for query server data.

Definition at line 29 of file tensor_query_server.c.

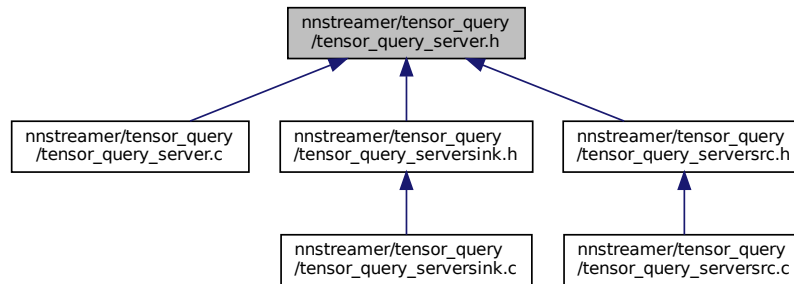
9.120 nnstreamer/tensor_query/tensor_query_server.h File Reference

GStreamer plugin to handle meta_query for server elements.

```
#include <gst/gst.h>
#include <tensor_common.h>
#include <nnstreamer-edge.h>
#include "tensor_meta.h"
Include dependency graph for tensor_query_server.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [GstTensorQueryEdgeInfo](#)
Internal data structure for nns-edge info to prepare edge connection.
- struct [GstTensorQueryServer](#)
GstTensorQueryServer internal info data structure.

Macros

- #define [DEFAULT_SERVER_ID](#) 0
- #define [DEFAULT_QUERY_INFO_TIMEOUT](#) 5

Functions

- gboolean [gst_tensor_query_server_add_data](#) (const guint id)
Add GstTensorQueryServer.
- gboolean [gst_tensor_query_server_prepare](#) (const guint id, nns_edge_connect_type_e connect_type, [GstTensorQueryEdgeInfo](#) *edge_info)
Prepare edge connection and its handle.
- void [gst_tensor_query_server_remove_data](#) (const guint id)
Remove GstTensorQueryServer.
- gboolean [gst_tensor_query_server_wait_sink](#) (const guint id)
Wait until the sink is configured and get server info handle.
- gboolean [gst_tensor_query_server_send_buffer](#) (const guint id, GstBuffer *buffer)
Send buffer to connected edge device.
- void [gst_tensor_query_server_set_configured](#) (const guint id)
set query server sink configured.
- void [gst_tensor_query_server_set_caps](#) (const guint id, const gchar *caps_str)
set query server caps.
- void [gst_tensor_query_server_release_edge_handle](#) (const guint id)
Release nnstreamer edge handle of query server.

9.120.1 Detailed Description

GStreamer plugin to handle meta_query for server elements.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

03 Aug 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.120.2 Macro Definition Documentation

9.120.2.1 DEFAULT_QUERY_INFO_TIMEOUT

```
#define DEFAULT_QUERY_INFO_TIMEOUT 5
```

Definition at line 23 of file tensor_query_server.h.

9.120.2.2 DEFAULT_SERVER_ID

```
#define DEFAULT_SERVER_ID 0
```

Definition at line 22 of file tensor_query_server.h.

9.120.3 Function Documentation

9.120.3.1 `gst_tensor_query_server_add_data()`

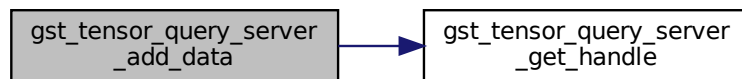
```
gboolean gst_tensor_query_server_add_data (
    const guint id )
```

Add [GstTensorQueryServer](#).

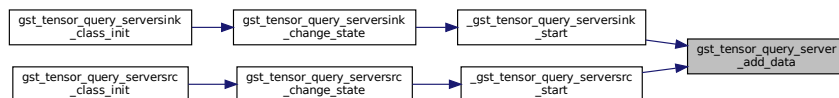
Add [GstTensorQueryServer](#).

Definition at line 77 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



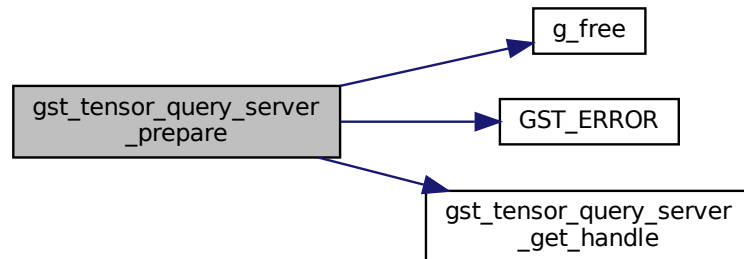
9.120.3.2 `gst_tensor_query_server_prepare()`

```
gboolean gst_tensor_query_server_prepare (
    const guint id,
    nns_edge_connect_type_e connect_type,
    GstTensorQueryEdgeInfo * edge_info )
```

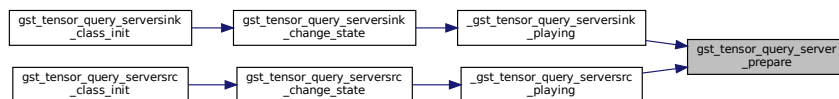
Prepare edge connection and its handle.

Definition at line 114 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



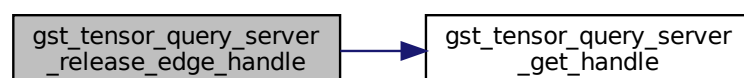
9.120.3.3 `gst_tensor_query_server_release_edge_handle()`

```
void gst_tensor_query_server_release_edge_handle (
    const quint id )
```

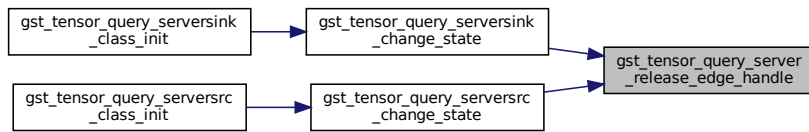
Release nnstreamer edge handle of query server.

Definition at line 258 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



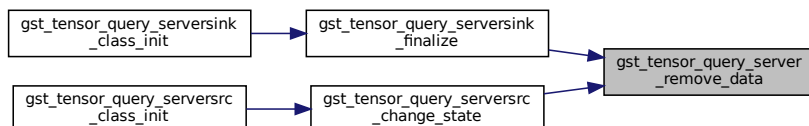
9.120.3.4 `gst_tensor_query_server_remove_data()`

```
void gst_tensor_query_server_remove_data (
    const guint id )
```

Remove [GstTensorQueryServer](#).

Definition at line 280 of file `tensor_query_server.c`.

Here is the caller graph for this function:



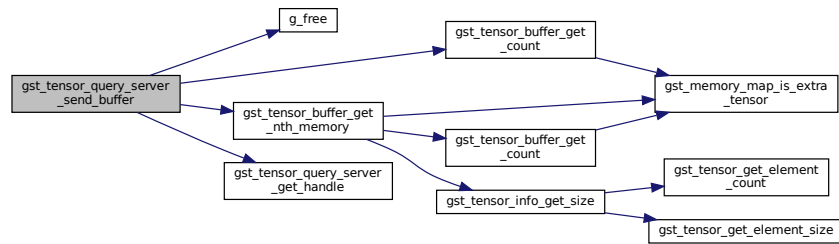
9.120.3.5 `gst_tensor_query_server_send_buffer()`

```
gboolean gst_tensor_query_server_send_buffer (
    const guint id,
    GstBuffer * buffer )
```

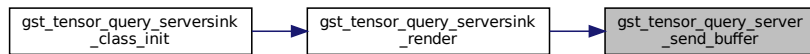
Send buffer to connected edge device.

Definition at line 183 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.120.3.6 gst_tensor_query_server_set_caps()

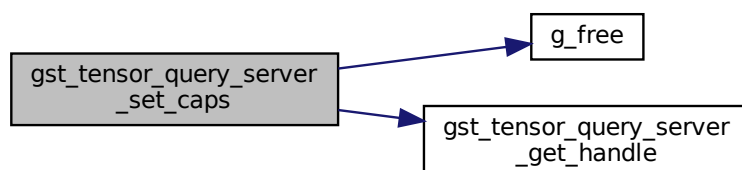
```

void gst_tensor_query_server_set_caps (
    const guint id,
    const gchar * caps_str )
  
```

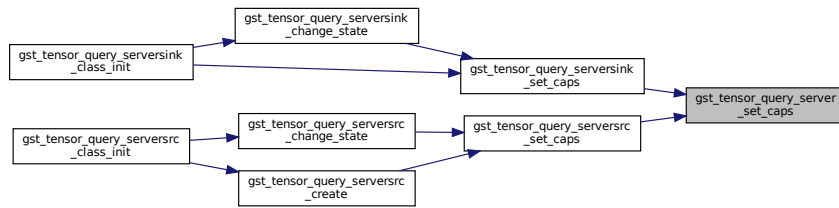
set query server caps.

Definition at line 342 of file tensor_query_server.c.

Here is the call graph for this function:



Here is the caller graph for this function:



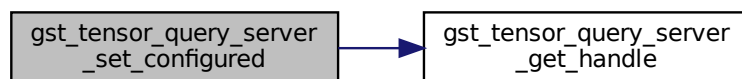
9.120.3.7 `gst_tensor_query_server_set_configured()`

```
void gst_tensor_query_server_set_configured (
    const guint id )
```

set query server sink configured.

Definition at line 322 of file `tensor_query_server.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



Macros

- #define `GST_CAT_DEFAULT` `gst_tensor_query_serversink_debug`
- #define `DEFAULT_METALESS_FRAME_LIMIT` `1`
- #define `gst_tensor_query_serversink_parent_class` `parent_class`

Enumerations

- enum {
`PROP_0`, `PROP_CONNECT_TYPE`, `PROP_ID`, `PROP_TIMEOUT`,
`PROP_METALESS_FRAME_LIMIT` }

Functions

- `GST_DEBUG_CATEGORY_STATIC` (`gst_tensor_query_serversink_debug`)
- `G_DEFINE_TYPE` (`GstTensorQueryServerSink`, `gst_tensor_query_serversink`, `GST_TYPE_BASE_SINK`)
- static `GstStateChangeReturn` `gst_tensor_query_serversink_change_state` (`GstElement *element`, `GstStateChange transition`)
Change state of query server sink.
- static void `gst_tensor_query_serversink_set_property` (`GObject *object`, `guint prop_id`, `const GValue *value`, `GParamSpec *pspec`)
set property
- static void `gst_tensor_query_serversink_get_property` (`GObject *object`, `guint prop_id`, `GValue *value`, `GParamSpec *pspec`)
get property
- static void `gst_tensor_query_serversink_finalize` (`GObject *object`)
finalize the object
- static `GstFlowReturn` `gst_tensor_query_serversink_render` (`GstBaseSink *bsink`, `GstBuffer *buf`)
render buffer, send buffer to client
- static `gboolean` `gst_tensor_query_serversink_set_caps` (`GstBaseSink *bsink`, `GstCaps *caps`)
An implementation of the set_caps vmethod in GstBaseSinkClass.
- static void `gst_tensor_query_serversink_class_init` (`GstTensorQueryServerSinkClass *klass`)
initialize the class
- static void `gst_tensor_query_serversink_init` (`GstTensorQueryServerSink *sink`)
initialize the new element
- static `gboolean` `_gst_tensor_query_serversink_start` (`GstTensorQueryServerSink *sink`)
start processing of query_serversink
- static `gboolean` `_gst_tensor_query_serversink_playing` (`GstTensorQueryServerSink *sink`)
start processing of query_serversink

Variables

- static `GstStaticPadTemplate` `sinktemplate`
the capabilities of the inputs.

9.121.1 Detailed Description

GStreamer plugin to handle tensor query server sink.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

09 Jul 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.121.2 Macro Definition Documentation

9.121.2.1 DEFAULT_METALESS_FRAME_LIMIT

```
#define DEFAULT_METALESS_FRAME_LIMIT 1
```

Definition at line 23 of file tensor_query_serversink.c.

9.121.2.2 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_query_serversink_debug
```

Definition at line 21 of file tensor_query_serversink.c.

9.121.2.3 gst_tensor_query_serversink_parent_class

```
#define gst_tensor_query_serversink_parent_class parent_class
```

Definition at line 42 of file tensor_query_serversink.c.

9.121.3 Enumeration Type Documentation

9.121.3.1 anonymous enum

```
anonymous enum
```

Enumerator

PROP_0	
PROP_CONNECT_TYPE	
PROP_ID	
PROP_TIMEOUT	
PROP_METALESS_FRAME_LIMIT	

Definition at line 33 of file tensor_query_serversink.c.

9.121.4 Function Documentation

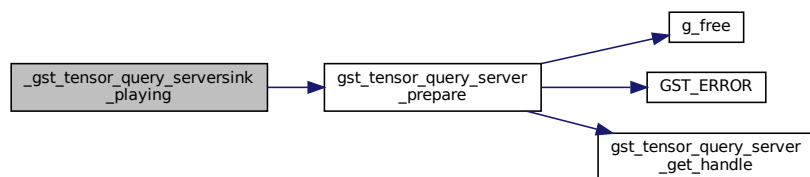
9.121.4.1 `_gst_tensor_query_serversink_playing()`

```
static gboolean _gst_tensor_query_serversink_playing (
    GstTensorQueryServerSink * sink ) [static]
```

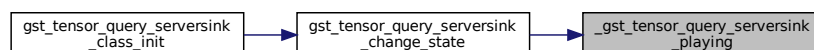
start processing of query_serversink

Definition at line 156 of file tensor_query_serversink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



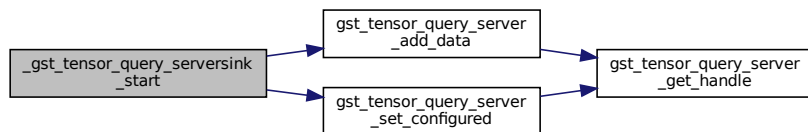
9.121.4.2 `_gst_tensor_query_serversink_start()`

```
static gboolean _gst_tensor_query_serversink_start (
    GstTensorQueryServerSink * sink ) [static]
```

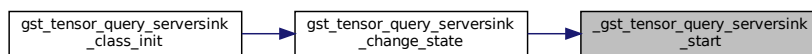
start processing of query_serversink

Definition at line 141 of file tensor_query_serversink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.121.4.3 `G_DEFINE_TYPE()`

```
G_DEFINE_TYPE (
    GstTensorQueryServerSink ,
    gst_tensor_query_serversink ,
    GST_TYPE_BASE_SINK )
```

9.121.4.4 `GST_DEBUG_CATEGORY_STATIC()`

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_query_serversink_debug )
```

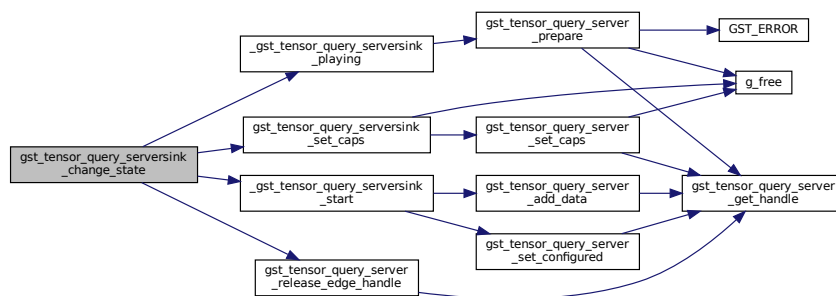
9.121.4.5 `gst_tensor_query_serversink_change_state()`

```
static GstStateChangeReturn gst_tensor_query_serversink_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

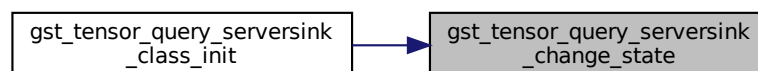
Change state of query server sink.

Definition at line 170 of file `tensor_query_serversink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



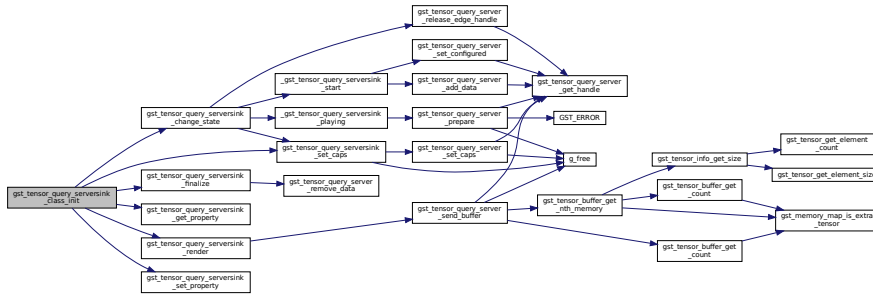
9.121.4.6 `gst_tensor_query_serversink_class_init()`

```
static void gst_tensor_query_serversink_class_init (
    GstTensorQueryServerSinkClass * klass ) [static]
```

initialize the class

Definition at line 62 of file `tensor_query_serversink.c`.

Here is the call graph for this function:



9.121.4.7 gst_tensor_query_serversink_finalize()

```
static void gst_tensor_query_serversink_finalize (
    GObject * object ) [static]
```

finalize the object

Definition at line 130 of file tensor_query_serversink.c.

Here is the call graph for this function:



Here is the caller graph for this function:



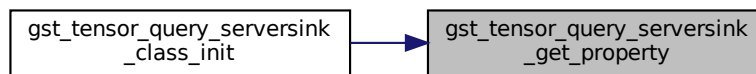
9.121.4.8 `gst_tensor_query_serversink_get_property()`

```
static void gst_tensor_query_serversink_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

get property

Definition at line 248 of file `tensor_query_serversink.c`.

Here is the caller graph for this function:



9.121.4.9 `gst_tensor_query_serversink_init()`

```
static void gst_tensor_query_serversink_init (
    GstTensorQueryServerSink * sink ) [static]
```

initialize the new element

Definition at line 118 of file `tensor_query_serversink.c`.

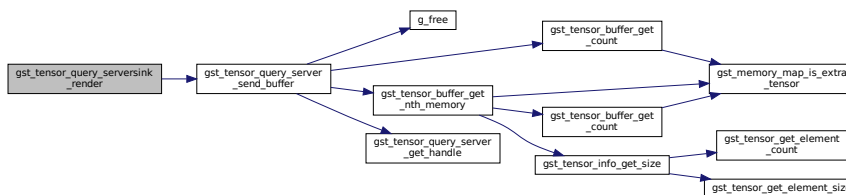
9.121.4.10 `gst_tensor_query_serversink_render()`

```
static GstFlowReturn gst_tensor_query_serversink_render (
    GstBaseSink * bsink,
    GstBuffer * buf ) [static]
```

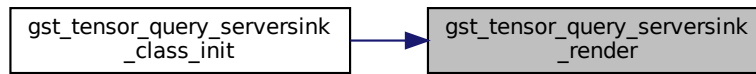
render buffer, send buffer to client

Definition at line 296 of file `tensor_query_serversink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.121.4.11 `gst_tensor_query_serversink_set_caps()`

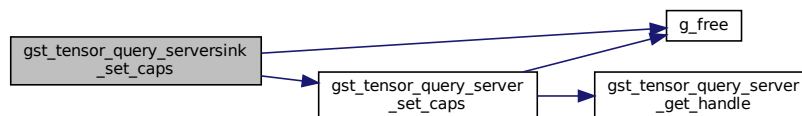
```

static gboolean gst_tensor_query_serversink_set_caps (
    GstBaseSink * basesink,
    GstCaps * caps ) [static]
  
```

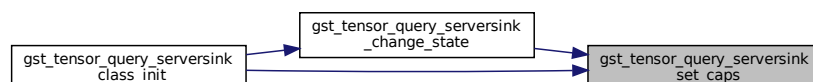
An implementation of the `set_caps` vmethod in `GstBaseSinkClass`.

Definition at line 276 of file `tensor_query_serversink.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



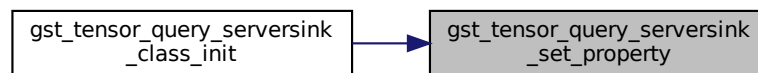
9.121.4.12 `gst_tensor_query_serversink_set_property()`

```
static void gst_tensor_query_serversink_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

set property

Definition at line 220 of file `tensor_query_serversink.c`.

Here is the caller graph for this function:



9.121.5 Variable Documentation

9.121.5.1 `sinktemplate`

```
GstStaticPadTemplate sinktemplate [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("sink",
    GST_PAD_SINK,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS_ANY)
```

the capabilities of the inputs.

Definition at line 28 of file `tensor_query_serversink.c`.

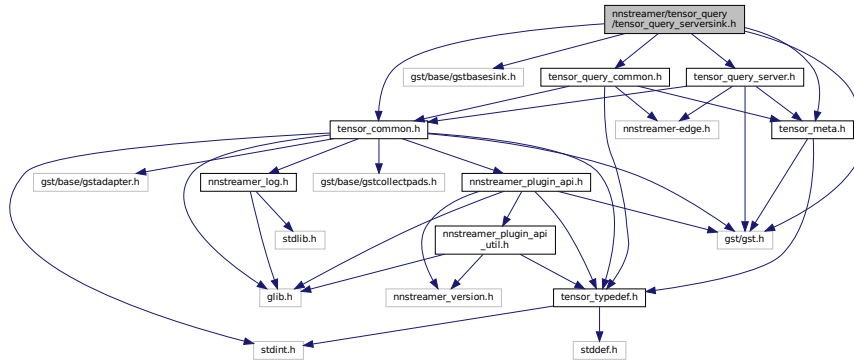
9.122 `nnstreamer/tensor_query/tensor_query_serversink.h` File Reference

GStreamer plugin to handle tensor query_server sink.

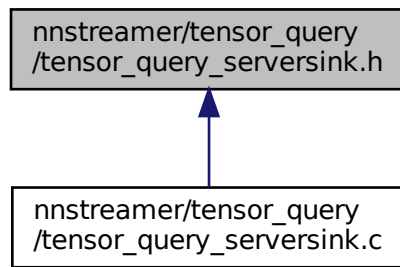
```
#include <gst/gst.h>
#include <gst/base/gstbasesink.h>
#include <tensor_common.h>
#include <tensor_meta.h>
#include "tensor_query_server.h"
```



```
#include "tensor_query_common.h"
Include dependency graph for tensor_query_serversink.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct [_GstTensorQueryServerSink](#)
GstTensorQueryServerSink data structure.
- struct [_GstTensorQueryServerSinkClass](#)
GstTensorQueryServerSinkClass data structure.

Macros

- #define [GST_TYPE_TENSOR_QUERY_SERVERSINK](#) ([gst_tensor_query_serversink_get_type\(\)](#))
- #define [GST_TENSOR_QUERY_SERVERSINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_CAST\(\(obj\), GST_TYPE_TENSOR_QUERY_SERVERSINK, _GstTensorQueryServerSink\)](#))
- #define [GST_TENSOR_QUERY_SERVERSINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_CAST\(\(klass\), GST_TYPE_TENSOR_QUERY_SERVERSINK, _GstTensorQueryServerSinkClass\)](#))
- #define [GST_IS_TENSOR_QUERY_SERVERSINK\(obj\)](#) ([G_TYPE_CHECK_INSTANCE_TYPE\(\(obj\), GST_TYPE_TENSOR_QUERY_SERVERSINK\)](#))
- #define [GST_IS_TENSOR_QUERY_SERVERSINK_CLASS\(klass\)](#) ([G_TYPE_CHECK_CLASS_TYPE\(\(klass\), GST_TYPE_TENSOR_QUERY_SERVERSINK\)](#))
- #define [GST_TENSOR_QUERY_SERVERSINK_CAST\(obj\)](#) ([\(\(GstTensorQueryServerSink *\) \(obj\)\)](#))

Typedefs

- typedef struct [_GstTensorQueryServerSink](#) [GstTensorQueryServerSink](#)
- typedef struct [_GstTensorQueryServerSinkClass](#) [GstTensorQueryServerSinkClass](#)

Functions

- GType [gst_tensor_query_serversink_get_type](#) (void)

9.122.1 Detailed Description

GStreamer plugin to handle tensor query_server sink.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

09 Jul 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.122.2 Macro Definition Documentation

9.122.2.1 GST_IS_TENSOR_QUERY_SERVERSINK

```
#define GST_IS_TENSOR_QUERY_SERVERSINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE((obj), GST_TYPE_TENSOR_QUERY_SERVERSINK))
```

Definition at line 29 of file tensor_query_serversink.h.

9.122.2.2 GST_IS_TENSOR_QUERY_SERVERSINK_CLASS

```
#define GST_IS_TENSOR_QUERY_SERVERSINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_QUERY_SERVERSINK))
```

Definition at line 31 of file tensor_query_serversink.h.

9.122.2.3 GST_TENSOR_QUERY_SERVERSINK

```
#define GST_TENSOR_QUERY_SERVERSINK(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_QUERY_SERVERSINK, GstTensorQueryServerSink))
```

Definition at line 25 of file tensor_query_serversink.h.

9.122.2.4 GST_TENSOR_QUERY_SERVERSINK_CAST

```
#define GST_TENSOR_QUERY_SERVERSINK_CAST(  
    obj ) ((GstTensorQueryServerSink *) (obj))
```

Definition at line 33 of file tensor_query_serversink.h.

9.122.2.5 GST_TENSOR_QUERY_SERVERSINK_CLASS

```
#define GST_TENSOR_QUERY_SERVERSINK_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_QUERY_SERVERSINK, GstTensorQueryServerSink))
```

Definition at line 27 of file tensor_query_serversink.h.

9.122.2.6 GST_TYPE_TENSOR_QUERY_SERVERSINK

```
#define GST_TYPE_TENSOR_QUERY_SERVERSINK (gst_tensor_query_serversink_get_type())
```

Definition at line 23 of file tensor_query_serversink.h.

9.122.3 Typedef Documentation

9.122.3.1 GstTensorQueryServerSink

```
typedef struct _GstTensorQueryServerSink GstTensorQueryServerSink
```

Definition at line 34 of file tensor_query_serversink.h.

9.122.3.2 GstTensorQueryServerSinkClass

```
typedef struct _GstTensorQueryServerSinkClass GstTensorQueryServerSinkClass
```

Definition at line 35 of file tensor_query_serversink.h.

9.122.4 Function Documentation

9.122.4.1 gst_tensor_query_serversink_get_type()

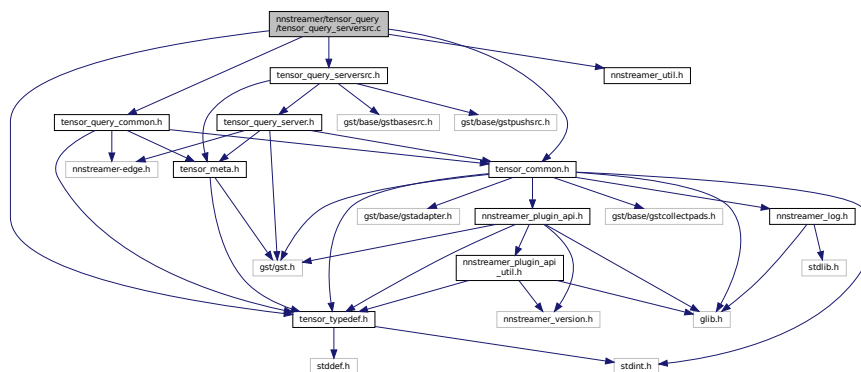
```
GType gst_tensor_query_serversink_get_type (
    void )
```

9.123 nnstreamer/tensor_query/tensor_query_serversrc.c File Reference

GStreamer plugin to handle tensor query_server src.

```
#include <tensor_typedef.h>
#include <tensor_common.h>
#include "tensor_query_serversrc.h"
#include "tensor_query_common.h"
#include "nnstreamer_util.h"
```

Include dependency graph for tensor_query_serversrc.c:



Macros

- #define `GST_CAT_DEFAULT` `gst_tensor_query_serversrc_debug`
- #define `DEFAULT_PORT_SRC` `3000`
- #define `DEFAULT_IS_LIVE` `TRUE`
- #define `DEFAULT_MQTT_HOST` `"127.0.0.1"`
- #define `DEFAULT_MQTT_PORT` `1883`
- #define `DEFAULT_DATA_POP_TIMEOUT` `100000U`
- #define `gst_tensor_query_serversrc_parent_class` `parent_class`

Enumerations

- enum {
[PROP_0](#), [PROP_HOST](#), [PROP_PORT](#), [PROP_DEST_HOST](#),
[PROP_DEST_PORT](#), [PROP_CONNECT_TYPE](#), [PROP_TIMEOUT](#), [PROP_TOPIC](#),
[PROP_ID](#), [PROP_IS_LIVE](#) }
query_serversrc properties

Functions

- [GST_DEBUG_CATEGORY_STATIC](#) ([gst_tensor_query_serversrc_debug](#))
- [G_DEFINE_TYPE](#) ([GstTensorQueryServerSrc](#), [gst_tensor_query_serversrc](#), [GST_TYPE_PUSH_SRC](#))
- static [GstStateChangeReturn](#) [gst_tensor_query_serversrc_change_state](#) ([GstElement](#) *element, [GstStateChange](#) transition)
Change state of query server src.
- static void [gst_tensor_query_serversrc_set_property](#) ([GObject](#) *object, [guint](#) prop_id, [const](#) [GValue](#) *value, [GParamSpec](#) *pspec)
set property of query_serversrc
- static void [gst_tensor_query_serversrc_get_property](#) ([GObject](#) *object, [guint](#) prop_id, [GValue](#) *value, [GParamSpec](#) *pspec)
get property of query_serversrc
- static void [gst_tensor_query_serversrc_finalize](#) ([GObject](#) *object)
finalize the query_serversrc object
- static [GstFlowReturn](#) [gst_tensor_query_serversrc_create](#) ([GstPushSrc](#) *psrc, [GstBuffer](#) **outbuf)
create query_serversrc, wait on socket and receive data
- static [gboolean](#) [gst_tensor_query_serversrc_set_caps](#) ([GstBaseSrc](#) *bsrc, [GstCaps](#) *caps)
An implementation of the set_caps vmethod in GstBaseSrcClass.
- static void [gst_tensor_query_serversrc_class_init](#) ([GstTensorQueryServerSrcClass](#) *klass)
initialize the query_serversrc class
- static void [gst_tensor_query_serversrc_init](#) ([GstTensorQueryServerSrc](#) *src)
initialize the new query_serversrc element
- static [int](#) [_nns_edge_event_cb](#) ([nns_edge_event_h](#) event_h, [void](#) *user_data)
nntreamer-edge event callback.
- static [gboolean](#) [_gst_tensor_query_serversrc_start](#) ([GstTensorQueryServerSrc](#) *src)
start processing of query_serversrc, setting up the server
- static [gboolean](#) [_gst_tensor_query_serversrc_playing](#) ([GstTensorQueryServerSrc](#) *src, [nns_edge_connect_type_e](#) connect_type)
start processing of query_serversrc, setting up the server
- static [GstBuffer](#) * [_gst_tensor_query_serversrc_get_buffer](#) ([GstTensorQueryServerSrc](#) *src)
Get buffer from message queue.

Variables

- static [GstStaticPadTemplate](#) [srctemplate](#)
the capabilities of the outputs

9.123.1 Detailed Description

GStreamer plugin to handle tensor query_server src.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

09 Jul 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.123.2 Macro Definition Documentation

9.123.2.1 DEFAULT_DATA_POP_TIMEOUT

```
#define DEFAULT_DATA_POP_TIMEOUT 100000U
```

Definition at line 30 of file tensor_query_serversrc.c.

9.123.2.2 DEFAULT_IS_LIVE

```
#define DEFAULT_IS_LIVE TRUE
```

Definition at line 27 of file tensor_query_serversrc.c.

9.123.2.3 DEFAULT_MQTT_HOST

```
#define DEFAULT_MQTT_HOST "127.0.0.1"
```

Definition at line 28 of file tensor_query_serversrc.c.

9.123.2.4 DEFAULT_MQTT_PORT

```
#define DEFAULT_MQTT_PORT 1883
```

Definition at line 29 of file tensor_query_serversrc.c.

9.123.2.5 DEFAULT_PORT_SRC

```
#define DEFAULT_PORT_SRC 3000
```

Definition at line 26 of file tensor_query_serversrc.c.

9.123.2.6 GST_CAT_DEFAULT

```
#define GST_CAT_DEFAULT gst_tensor_query_serversrc_debug
```

Definition at line 24 of file tensor_query_serversrc.c.

9.123.2.7 gst_tensor_query_serversrc_parent_class

```
#define gst_tensor_query_serversrc_parent_class parent_class
```

Definition at line 57 of file tensor_query_serversrc.c.

9.123.3 Enumeration Type Documentation

9.123.3.1 anonymous enum

anonymous enum

query_serversrc properties

Enumerator

PROP_0	
PROP_HOST	
PROP_PORT	
PROP_DEST_HOST	
PROP_DEST_PORT	
PROP_CONNECT_TYPE	
PROP_TIMEOUT	
PROP_TOPIC	
PROP_ID	
PROP_IS_LIVE	

Definition at line 43 of file tensor_query_serversrc.c.

9.123.4 Function Documentation

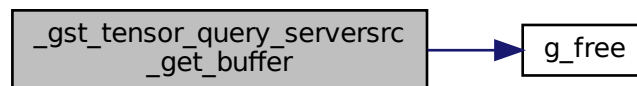
9.123.4.1 `_gst_tensor_query_serversrc_get_buffer()`

```
static GstBuffer* _gst_tensor_query_serversrc_get_buffer (
    GstTensorQueryServerSrc * src ) [static]
```

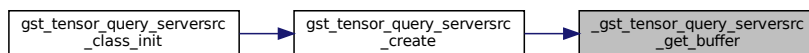
Get buffer from message queue.

Definition at line 434 of file tensor_query_serversrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



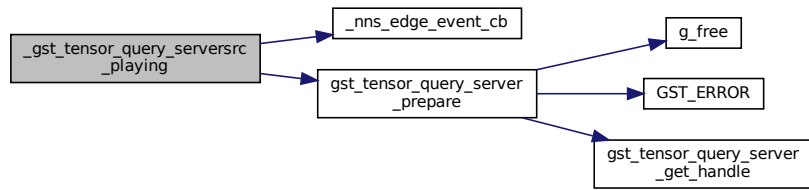
9.123.4.2 `_gst_tensor_query_serversrc_playing()`

```
static gboolean _gst_tensor_query_serversrc_playing (
    GstTensorQueryServerSrc * src,
    nns_edge_connect_type_e connect_type ) [static]
```

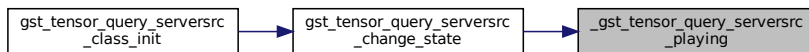
start processing of query_serversrc, setting up the server

Definition at line 252 of file tensor_query_serversrc.c.

Here is the call graph for this function:



Here is the caller graph for this function:



9.123.4.3 `_gst_tensor_query_serversrc_start()`

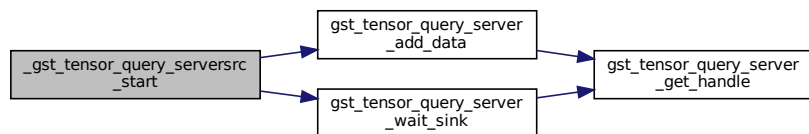
```

static gboolean _gst_tensor_query_serversrc_start (
    GstTensorQueryServerSrc * src ) [static]
  
```

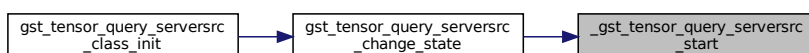
start processing of query_serversrc, setting up the server

Definition at line 235 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.123.4.4 `_nns_edge_event_cb()`

```
static int _nns_edge_event_cb (
    nns_edge_event_h event_h,
    void * user_data ) [static]
```

nnstreamer-edge event callback.

Definition at line 199 of file `tensor_query_serversrc.c`.

Here is the caller graph for this function:



9.123.4.5 `G_DEFINE_TYPE()`

```
G_DEFINE_TYPE (
    GstTensorQueryServerSrc ,
    gst_tensor_query_serversrc ,
    GST_TYPE_PUSH_SRC )
```

9.123.4.6 `GST_DEBUG_CATEGORY_STATIC()`

```
GST_DEBUG_CATEGORY_STATIC (
    gst_tensor_query_serversrc_debug )
```

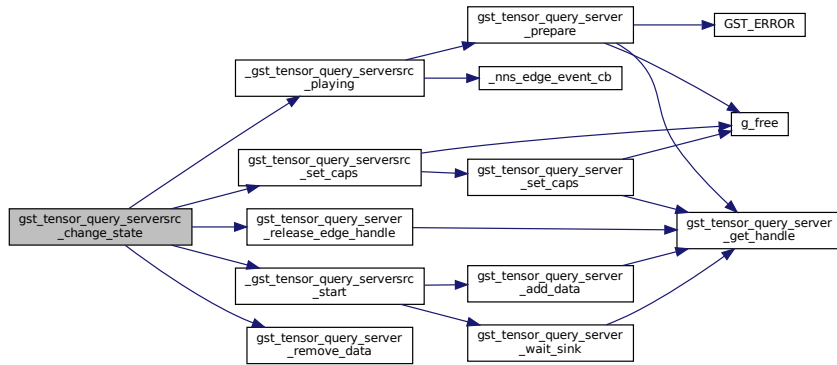
9.123.4.7 `gst_tensor_query_serversrc_change_state()`

```
static GstStateChangeReturn gst_tensor_query_serversrc_change_state (
    GstElement * element,
    GstStateChange transition ) [static]
```

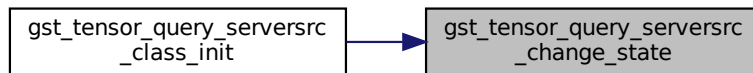
Change state of query server src.

Definition at line 275 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



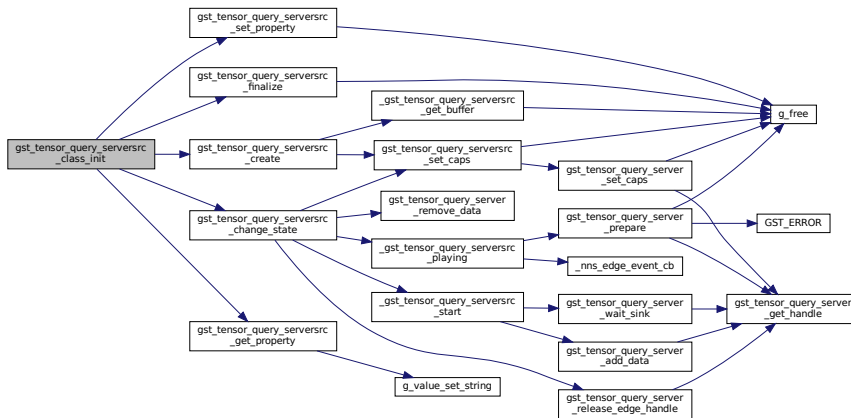
9.123.4.8 `gst_tensor_query_serversrc_class_init()`

```
static void gst_tensor_query_serversrc_class_init (
    GstTensorQueryServerSrcClass * klass ) [static]
```

initialize the query_serversrc class

Definition at line 77 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



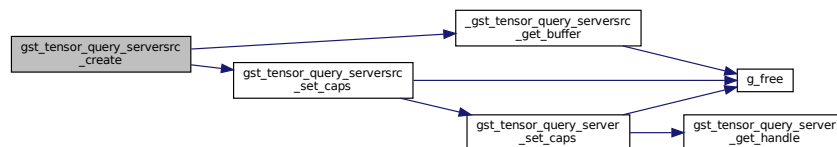
9.123.4.9 `gst_tensor_query_serversrc_create()`

```
static GstFlowReturn gst_tensor_query_serversrc_create (
    GstPushSrc * psrc,
    GstBuffer ** buf ) [static]
```

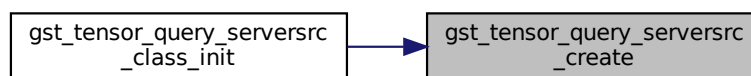
create query_serversrc, wait on socket and receive data

Definition at line 495 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



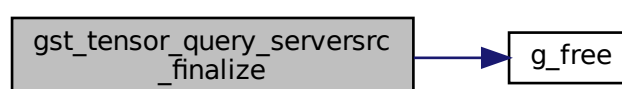
9.123.4.10 `gst_tensor_query_serversrc_finalize()`

```
static void gst_tensor_query_serversrc_finalize (
    GObject * object ) [static]
```

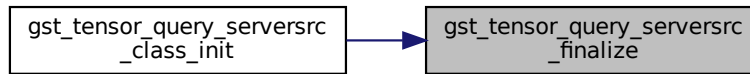
finalize the query_serversrc object

Definition at line 175 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



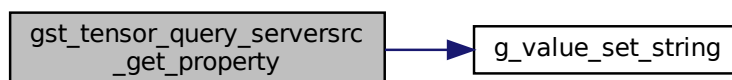
9.123.4.11 `gst_tensor_query_serversrc_get_property()`

```
static void gst_tensor_query_serversrc_get_property (
    GObject * object,
    guint prop_id,
    GValue * value,
    GParamSpec * pspec ) [static]
```

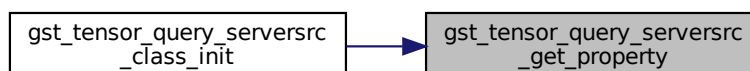
get property of query_serversrc

Definition at line 390 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.123.4.12 `gst_tensor_query_serversrc_init()`

```
static void gst_tensor_query_serversrc_init (
    GstTensorQueryServerSrc * src ) [static]
```

initialize the new `query_serversrc` element

set the timestamps on each buffer

set the source to be live

Definition at line 150 of file `tensor_query_serversrc.c`.

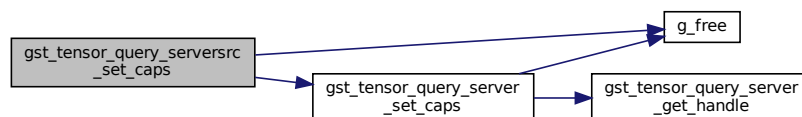
9.123.4.13 `gst_tensor_query_serversrc_set_caps()`

```
static gboolean gst_tensor_query_serversrc_set_caps (
    GstBaseSrc * bsrc,
    GstCaps * caps ) [static]
```

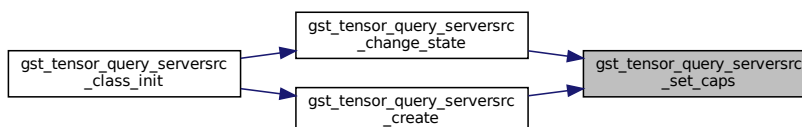
An implementation of the `set_caps` vmethod in `GstBaseSrcClass`.

Definition at line 533 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



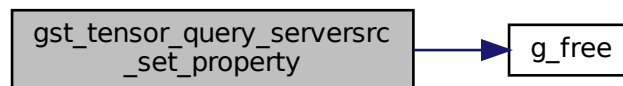
9.123.4.14 `gst_tensor_query_serversrc_set_property()`

```
static void gst_tensor_query_serversrc_set_property (
    GObject * object,
    guint prop_id,
    const GValue * value,
    GParamSpec * pspec ) [static]
```

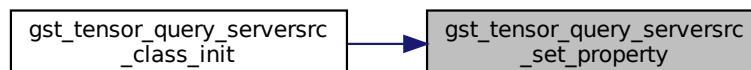
set property of query_serversrc

Definition at line 331 of file `tensor_query_serversrc.c`.

Here is the call graph for this function:



Here is the caller graph for this function:



9.123.5 Variable Documentation

9.123.5.1 `srctemplate`

```
GstStaticPadTemplate srctemplate [static]
```

Initial value:

```
= GST_STATIC_PAD_TEMPLATE ("src",
    GST_PAD_SRC,
    GST_PAD_ALWAYS,
    GST_STATIC_CAPS_ANY)
```

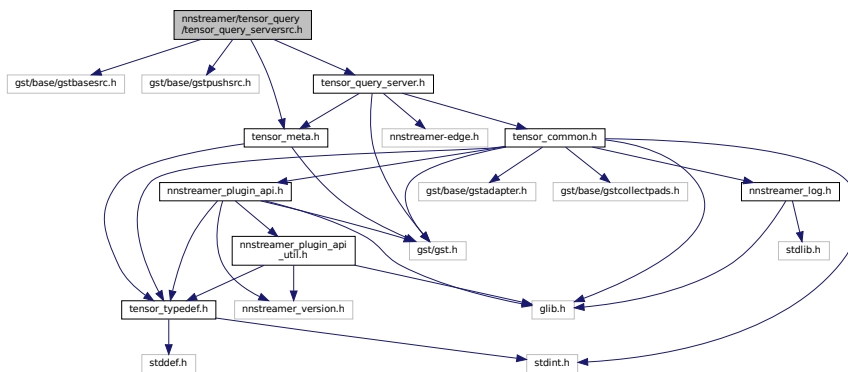
the capabilities of the outputs

Definition at line 35 of file `tensor_query_serversrc.c`.

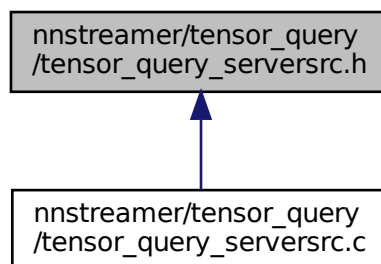
9.124 nnstreamer/tensor_query/tensor_query_serversrc.h File Reference

GStreamer plugin to handle tensor_query_server src.

```
#include <gst/base/gstbasesrc.h>
#include <gst/base/gstpushsrc.h>
#include <tensor_meta.h>
#include "tensor_query_server.h"
Include dependency graph for tensor_query_serversrc.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- [struct _GstTensorQueryServerSrc](#)
GstTensorQueryServerSrc data structure.
- [struct _GstTensorQueryServerSrcClass](#)
GstTensorQueryServerSrcClass data structure.

Macros

- #define [GST_TYPE_TENSOR_QUERY_SERVERSRC](#) ([gst_tensor_query_serversrc_get_type\(\)](#))
- #define [GST_TENSOR_QUERY_SERVERSRC](#)(obj) ([G_TYPE_CHECK_INSTANCE_CAST](#)((obj),[GST_TYPE_TENSOR_QUERY_SERVERSRC](#)))
- #define [GST_TENSOR_QUERY_SERVERSRC_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_CAST](#)((klass),[GST_TYPE_TENSOR_QUERY_SERVERSRC](#)))
- #define [GST_IS_TENSOR_QUERY_SERVERSRC](#)(obj) ([G_TYPE_CHECK_INSTANCE_TYPE](#)((obj),[GST_TYPE_TENSOR_QUERY_SERVERSRC](#)))
- #define [GST_IS_TENSOR_QUERY_SERVERSRC_CLASS](#)(klass) ([G_TYPE_CHECK_CLASS_TYPE](#)((klass),[GST_TYPE_TENSOR_QUERY_SERVERSRC](#)))
- #define [GST_TENSOR_QUERY_SERVERSRC_CAST](#)(obj) (([GstTensorQueryServerSrc](#) *) (obj))

Typedefs

- typedef struct [_GstTensorQueryServerSrc](#) [GstTensorQueryServerSrc](#)
- typedef struct [_GstTensorQueryServerSrcClass](#) [GstTensorQueryServerSrcClass](#)

Functions

- GType [gst_tensor_query_serversrc_get_type](#) (void)

9.124.1 Detailed Description

GStreamer plugin to handle tensor_query_server src.

Copyright (C) 2021 Samsung Electronics Co., Ltd.

Date

09 Jul 2021

Author

Junhwan Kim jejudo.kim@samsung.com

See also

<http://github.com/nnstreamer/nnstreamer>

Bug No known bugs

9.124.2 Macro Definition Documentation

9.124.2.1 GST_IS_TENSOR_QUERY_SERVERSRC

```
#define GST_IS_TENSOR_QUERY_SERVERSRC (
    obj ) (G_TYPE_CHECK_INSTANCE_TYPE ((obj), GST_TYPE_TENSOR_QUERY_SERVERSRC))
```

Definition at line 28 of file [tensor_query_serversrc.h](#).

9.124.2.2 GST_IS_TENSOR_QUERY_SERVERSRC_CLASS

```
#define GST_IS_TENSOR_QUERY_SERVERSRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_TYPE((klass), GST_TYPE_TENSOR_QUERY_SERVERSRC))
```

Definition at line 30 of file tensor_query_serversrc.h.

9.124.2.3 GST_TENSOR_QUERY_SERVERSRC

```
#define GST_TENSOR_QUERY_SERVERSRC(  
    obj ) (G_TYPE_CHECK_INSTANCE_CAST((obj), GST_TYPE_TENSOR_QUERY_SERVERSRC, GstTensorQueryServerSrc))
```

Definition at line 24 of file tensor_query_serversrc.h.

9.124.2.4 GST_TENSOR_QUERY_SERVERSRC_CAST

```
#define GST_TENSOR_QUERY_SERVERSRC_CAST(  
    obj ) ((GstTensorQueryServerSrc *) (obj))
```

Definition at line 32 of file tensor_query_serversrc.h.

9.124.2.5 GST_TENSOR_QUERY_SERVERSRC_CLASS

```
#define GST_TENSOR_QUERY_SERVERSRC_CLASS(  
    klass ) (G_TYPE_CHECK_CLASS_CAST((klass), GST_TYPE_TENSOR_QUERY_SERVERSRC, GstTensorQueryServerSrc))
```

Definition at line 26 of file tensor_query_serversrc.h.

9.124.2.6 GST_TYPE_TENSOR_QUERY_SERVERSRC

```
#define GST_TYPE_TENSOR_QUERY_SERVERSRC (gst_tensor_query_serversrc_get_type())
```

Definition at line 22 of file tensor_query_serversrc.h.

9.124.3 Typedef Documentation

9.124.3.1 GstTensorQueryServerSrc

```
typedef struct _GstTensorQueryServerSrc GstTensorQueryServerSrc
```

Definition at line 33 of file tensor_query_serversrc.h.

9.124.3.2 GstTensorQueryServerSrcClass

```
typedef struct _GstTensorQueryServerSrcClass GstTensorQueryServerSrcClass
```

Definition at line 34 of file tensor_query_serversrc.h.

9.124.4 Function Documentation

9.124.4.1 gst_tensor_query_serversrc_get_type()

```
GType gst_tensor_query_serversrc_get_type (  
    void )
```


Index

- [_CONVERTER_MODE_CUSTOM_CODE](#)
 - [gsttensor_converter.h, 557](#)
- [_CONVERTER_MODE_CUSTOM_SCRIPT](#)
 - [gsttensor_converter.h, 557](#)
- [_CONVERTER_MODE_NONE](#)
 - [gsttensor_converter.h, 557](#)
- [_FilterTransformData, 29](#)
 - [allocate_in_invoke, 30](#)
 - [info, 30](#)
 - [is_flexible, 30](#)
 - [mem, 30](#)
 - [meta, 30](#)
 - [num_tensors, 30](#)
 - [tensors, 31](#)
- [_GTensorFilterSingle, 219](#)
 - [element, 219](#)
 - [priv, 219](#)
- [_GTensorFilterSingleClass, 220](#)
 - [allocate_in_invoke, 220](#)
 - [destroy_notify, 220](#)
 - [input_configured, 221](#)
 - [invoke, 221](#)
 - [output_configured, 221](#)
 - [parent, 221](#)
 - [set_input_info, 221](#)
 - [start, 222](#)
 - [stop, 222](#)
- [_GTensorFilterSinglePrivate, 222](#)
 - [allocate_in_invoke, 223](#)
 - [filter_priv, 223](#)
- [_GstDataRepoSink, 31](#)
 - [cumulative_tensors, 32](#)
 - [data_type, 32](#)
 - [element, 32](#)
 - [fd, 32](#)
 - [fd_offset, 32](#)
 - [filename, 32](#)
 - [fixed_caps, 33](#)
 - [is_static_tensors, 33](#)
 - [json_filename, 33](#)
 - [json_object, 33](#)
 - [sample_offset_array, 33](#)
 - [sample_size, 34](#)
 - [tensor_count_array, 34](#)
 - [tensor_size_array, 34](#)
 - [total_samples, 34](#)
- [_GstDataRepoSinkClass, 35](#)
 - [parent_class, 35](#)
- [_GstDataRepoSrc, 35](#)
 - [array_index, 37](#)
 - [caps, 37](#)
 - [config, 37](#)
 - [current_sample_index, 38](#)
 - [data_type, 38](#)
 - [epochs, 38](#)
 - [fd, 38](#)
 - [fd_offset, 38](#)
 - [file_size, 39](#)
 - [filename, 39](#)
 - [first_epoch_is_done, 39](#)
 - [is_shuffle, 39](#)
 - [is_start, 39](#)
 - [json_filename, 40](#)
 - [last_offset, 40](#)
 - [n_frame, 40](#)
 - [need_changed_caps, 40](#)
 - [num_samples, 40](#)
 - [parent, 41](#)
 - [parser, 41](#)
 - [rate_d, 41](#)
 - [rate_n, 41](#)
 - [read_position, 41](#)
 - [running_time, 42](#)
 - [sample_offset_array, 42](#)
 - [sample_offset_array_len, 42](#)
 - [sample_size, 42](#)
 - [shuffled_index_array, 42](#)
 - [src_pad, 43](#)
 - [start_offset, 43](#)
 - [start_sample_index, 43](#)
 - [stop_sample_index, 43](#)
 - [successful_read, 43](#)
 - [tensor_count_array, 44](#)
 - [tensor_count_array_len, 44](#)
 - [tensor_size_array, 44](#)
 - [tensor_size_array_len, 44](#)
 - [tensors_offset, 44](#)
 - [tensors_seq, 45](#)
 - [tensors_seq_cnt, 45](#)
 - [tensors_seq_str, 45](#)
 - [tensors_size, 45](#)
 - [total_samples, 45](#)
- [_GstDataRepoSrcClass, 46](#)
 - [parent_class, 46](#)
- [_GstEdgeSink, 47](#)
 - [cond, 47](#)
 - [connect_type, 47](#)
 - [connection_timeout, 47](#)

- custom_lib, 48
- dest_host, 48
- dest_port, 48
- edge_h, 48
- element, 48
- host, 48
- is_connected, 49
- lock, 49
- port, 49
- topic, 49
- wait_connection, 49
- _GstEdgeSinkClass, 50
 - parent_class, 50
- _GstEdgeSrc, 50
 - connect_type, 51
 - custom_lib, 51
 - dest_host, 51
 - dest_port, 51
 - edge_h, 51
 - element, 51
 - msg_queue, 52
 - playing, 52
 - topic, 52
- _GstEdgeSrcClass, 52
 - parent_class, 53
- _GstJoin, 53
 - active_sinkpad, 53
 - cond, 54
 - element, 54
 - have_group_id, 54
 - lock, 54
 - n_pads, 54
 - padcount, 54
 - srcpad, 55
- _GstJoinClass, 55
 - parent_class, 55
- _GstJoinPad, 56
 - group_id, 56
 - parent, 56
 - segment, 56
 - segment_seqnum, 56
- _GstJoinPadClass, 57
 - parent, 57
- _GstMQTTMessageHdr, 57
 - _reserved_hdr, 58
 - base_time_epoch, 58
 - dts, 58
 - duration, 59
 - gst_caps_str, 59
 - num_mems, 59
 - pts, 59
 - sent_time_epoch, 59
 - size_mems, 59
- _GstMqttSink, 60
 - base_time_epoch, 61
 - debug, 61
 - err, 61
 - get_epoch_func, 61
 - gquark_err_tag, 62
 - in_caps, 62
 - is_connected, 62
 - max_msg_buf_size, 62
 - mqtt_client_handle, 62
 - mqtt_client_id, 62
 - mqtt_conn_opts, 63
 - mqtt_host_address, 63
 - mqtt_host_port, 63
 - mqtt_msg_buf, 63
 - mqtt_msg_buf_size, 63
 - mqtt_msg_hdr, 63
 - mqtt_ntp_hnames, 64
 - mqtt_ntp_num_srvs, 64
 - mqtt_ntp_ports, 64
 - mqtt_ntp_srvs, 64
 - mqtt_ntp_sync, 64
 - mqtt_pub_wait_timeout, 64
 - mqtt_qos, 65
 - mqtt_respn_opts, 65
 - mqtt_sink_gcond, 65
 - mqtt_sink_mutex, 65
 - mqtt_sink_state, 65
 - mqtt_topic, 65
 - num_buffers, 66
 - parent, 66
- _GstMqttSinkClass, 66
 - parent_class, 67
- _GstMqttSrc, 67
 - aqueue, 68
 - base_time_epoch, 68
 - caps, 68
 - debug, 68
 - err, 68
 - gquark_err_tag, 69
 - is_connected, 69
 - is_live, 69
 - is_subscribed, 69
 - latency, 69
 - mqtt_client_handle, 69
 - mqtt_client_id, 70
 - mqtt_conn_opts, 70
 - mqtt_host_address, 70
 - mqtt_host_port, 70
 - mqtt_qos, 70
 - mqtt_respn_opts, 70
 - mqtt_src_gcond, 71
 - mqtt_src_mutex, 71
 - mqtt_sub_timeout, 71
 - mqtt_topic, 71
 - num_dumped, 71
 - parent, 71
- _GstMqttSrcClass, 72
 - parent_class, 72
- _GstTensorAggregator, 73
 - adapter_table, 74
 - concat, 74
 - element, 74

- frames_dim, 74
- frames_flush, 74
- frames_in, 75
- frames_out, 75
- in_config, 75
- out_config, 75
- silent, 75
- sinkpad, 76
- srcpad, 76
- tensor_configured, 76
- [_GstTensorAggregatorClass](#), 76
 - parent_class, 77
- [_GstTensorConverter](#), 77
 - adapter_table, 78
 - custom, 78
 - do_not_append_header, 79
 - element, 79
 - ext_fw, 79
 - externalConverter, 79
 - frame_size, 79
 - frames_per_tensor, 80
 - have_segment, 80
 - in_media_type, 80
 - mode, 80
 - mode_option, 80
 - need_segment, 81
 - old_timestamp, 81
 - priv_data, 81
 - remove_padding, 81
 - segment, 81
 - set_timestamp, 82
 - silent, 82
 - sinkpad, 82
 - srcpad, 82
 - tensors_config, 82
 - tensors_configured, 83
 - tensors_info, 83
- [_GstTensorConverterClass](#), 83
 - parent_class, 84
- [_GstTensorCrop](#), 84
 - collect, 84
 - element, 85
 - lateness, 85
 - send_stream_start, 85
 - silent, 85
 - sinkpad_info, 85
 - sinkpad_raw, 86
 - srcpad, 86
- [_GstTensorCropClass](#), 86
 - parent_class, 87
- [_GstTensorDebug](#), 87
 - cap_mode, 87
 - element, 88
 - meta_mode, 88
 - output_mode, 88
 - silent, 88
 - sinkpad, 88
 - srcpad, 88
- [_GstTensorDebugClass](#), 89
 - parent_class, 89
- [_GstTensorDecoder](#), 90
 - config_path, 90
 - configured, 91
 - custom, 91
 - decoder, 91
 - element, 91
 - is_custom, 91
 - negotiated, 92
 - option, 92
 - plugin_data, 92
 - silent, 92
 - tensor_config, 92
- [_GstTensorDecoderClass](#), 93
 - parent_class, 93
- [_GstTensorDecoderDef](#), 94
 - decode, 94
 - exit, 95
 - getOutCaps, 95
 - getTransformSize, 95
 - init, 96
 - modename, 96
 - setOption, 96
- [_GstTensorDemux](#), 97
 - element, 98
 - group_id, 98
 - have_group_id, 98
 - num_srcpads, 98
 - silent, 99
 - sinkpad, 99
 - srcpads, 99
 - tensorpick, 99
 - tensors_config, 99
- [_GstTensorDemuxClass](#), 100
 - parent_class, 100
- [_GstTensorFilter](#), 100
 - element, 101
 - prev_ts, 101
 - priv, 101
 - throttling_accum, 101
 - throttling_delay, 102
- [_GstTensorFilterClass](#), 102
 - parent_class, 102
- [_GstTensorFilterCombination](#), 103
 - in_combi, 103
 - in_combi_defined, 103
 - out_combi_i, 104
 - out_combi_i_defined, 104
 - out_combi_o, 104
 - out_combi_o_defined, 104
- [_GstTensorFilterFramework](#), 105
 - allocate_in_invoke, 107
 - allocateInInvoke, 107
 - allow_in_place, 107
 - checkAvailability, 107
 - close, 108
 - destroyNotify, 108

- eventHandler, 109
- getFrameworkInfo, 109
- getInputDimension, 110
- getModelInfo, 110
- getOutputDimension, 111
- handleEvent, 112
- invoke, 112
- invoke_NN, 113
- name, 113
- open, 113
- reloadModel, 114
- run_without_model, 114
- setInputDimension, 114
- statistics, 115
- subplugin_data, 115
- verify_model_path, 115
- version, 116
- _GstTensorFilterFrameworkEventData, 116
 - custom, 117
 - custom_properties, 117
 - data, 118
 - hw, 118
 - hw_list, 118
 - info, 118
 - layout, 118
 - model_files, 119
 - num_hw, 119
 - num_models, 119
- _GstTensorFilterFrameworkInfo, 120
 - accl_auto, 120
 - accl_default, 121
 - allocate_in_invoke, 121
 - allow_in_place, 121
 - hw_list, 121
 - name, 121
 - num_hw, 122
 - run_without_model, 122
 - statistics, 122
 - verify_model_path, 122
- _GstTensorFilterFrameworkStatistics, 123
 - total_invoke_latency, 123
 - total_invoke_num, 123
 - total_overhead_latency, 123
- _GstTensorFilterPrivate, 124
 - combi, 125
 - config_path, 125
 - configured, 125
 - fw, 125
 - in_config, 125
 - info, 126
 - is_updatable, 126
 - latency_mode, 126
 - latency_reported, 126
 - latency_reporting, 126
 - out_config, 127
 - privateData, 127
 - prop, 127
 - silent, 127
 - stat, 127
 - throughput_mode, 128
 - watchdog_h, 128
- _GstTensorFilterProperties, 128
 - accl_str, 129
 - custom_properties, 129
 - fw_opened, 130
 - fwname, 130
 - hw_list, 130
 - input_configured, 130
 - input_layout, 130
 - input_meta, 131
 - input_ranks, 131
 - invoke_dynamic, 131
 - latency, 131
 - model_files, 131
 - num_hw, 132
 - num_models, 132
 - output_configured, 132
 - output_layout, 132
 - output_meta, 132
 - output_ranks, 133
 - shared_tensor_filter_key, 133
 - suspend, 133
 - throughput, 133
- _GstTensorFilterStatistics, 134
 - latency_ignore_count, 134
 - latest_invoke_time, 134
 - old_total_invoke_latency, 134
 - old_total_invoke_num, 135
 - recent_latencies, 135
 - total_invoke_latency, 135
 - total_invoke_num, 135
- _GstTensorIf, 136
 - act_else, 137
 - act_then, 137
 - custom, 137
 - custom_configured, 137
 - cv, 137
 - cv_option, 137
 - element, 138
 - else_option, 138
 - in_config, 138
 - lock, 138
 - num_srcpads, 138
 - op, 139
 - out_config, 139
 - silent, 139
 - sinkpad, 139
 - srcpads, 139
 - sv, 139
 - then_option, 140
- _GstTensorIfClass, 140
 - parent_class, 140
- _GstTensorMerge, 141
 - collect, 142
 - current_time, 142
 - data_linear, 142

- element, 142
- loaded, 142
- mode, 142
- need_segment, 143
- need_set_time, 143
- need_stream_start, 143
- negotiated, 143
- option, 143
- send_stream_start, 143
- silent, 144
- srcpad, 144
- sync, 144
- tensors_config, 144
- [_GstTensorMergeClass](#), 144
 - parent_class, 145
- [_GstTensorMux](#), 145
 - collect, 146
 - current_time, 146
 - element, 146
 - need_segment, 146
 - need_set_time, 147
 - need_stream_start, 147
 - negotiated, 147
 - send_stream_start, 147
 - silent, 147
 - srcpad, 147
 - sync, 148
 - tensors_config, 148
- [_GstTensorMuxClass](#), 148
 - parent_class, 149
- [_GstTensorQueryClient](#), 149
 - config, 150
 - connect_type, 150
 - dest_host, 150
 - dest_port, 151
 - edge_h, 151
 - element, 151
 - host, 151
 - in_caps_str, 151
 - is_tensor, 151
 - max_request, 152
 - msg_queue, 152
 - port, 152
 - requested_num, 152
 - silent, 152
 - sinkpad, 152
 - srcpad, 153
 - timeout, 153
 - topic, 153
- [_GstTensorQueryClientClass](#), 153
 - parent_class, 154
- [_GstTensorQueryServerSink](#), 154
 - connect_type, 155
 - element, 155
 - metaless_frame_count, 155
 - metaless_frame_limit, 155
 - sink_id, 155
 - timeout, 155
- [_GstTensorQueryServerSinkClass](#), 156
 - parent_class, 156
- [_GstTensorQueryServerSrc](#), 157
 - configured, 157
 - connect_type, 157
 - dest_host, 157
 - dest_port, 158
 - element, 158
 - host, 158
 - msg_queue, 158
 - playing, 158
 - port, 158
 - src_id, 159
 - timeout, 159
 - topic, 159
- [_GstTensorQueryServerSrcClass](#), 159
 - parent_class, 160
- [_GstTensorRate](#), 160
 - base_ts, 161
 - drop, 161
 - dup, 161
 - element, 161
 - from_rate_denominator, 161
 - from_rate_numerator, 162
 - in, 162
 - last_ts, 162
 - next_ts, 162
 - out, 162
 - out_frame_count, 163
 - prev_ts, 163
 - prevbuf, 163
 - rate_d, 163
 - rate_n, 163
 - segment, 164
 - sent_qos_on_passthrough, 164
 - silent, 164
 - throttle, 164
 - to_rate_denominator, 164
 - to_rate_numerator, 165
- [_GstTensorRateClass](#), 165
 - parent_class, 165
- [_GstTensorRepoSink](#), 166
 - element, 166
 - in_caps, 166
 - last_render_time, 167
 - myid, 167
 - o_myid, 167
 - set_startid, 167
 - signal_rate, 167
 - silent, 167
- [_GstTensorRepoSinkClass](#), 168
 - parent_class, 168
- [_GstTensorRepoSrc](#), 169
 - caps, 170
 - config, 170
 - fps_d, 170
 - fps_n, 170
 - ini, 170

- myid, 170
- negotiation, 171
- o_myid, 171
- parent, 171
- set_startid, 171
- silent, 171
- _GstTensorRepoSrcClass, 172
 - parent_class, 172
- _GstTensorSink, 172
 - element, 173
 - emit_signal, 173
 - last_render_time, 173
 - mutex, 173
 - signal_rate, 173
 - silent, 174
- _GstTensorSinkClass, 174
 - eos, 174
 - new_data, 175
 - parent_class, 175
 - stream_start, 175
- _GstTensorSparseDec, 176
 - element, 177
 - in_config, 177
 - out_config, 177
 - silent, 177
 - sinkpad, 177
 - srcpad, 177
- _GstTensorSparseDecClass, 178
 - parent_class, 178
- _GstTensorSparseEnc, 179
 - element, 179
 - in_config, 180
 - silent, 180
 - sinkpad, 180
 - srcpad, 180
- _GstTensorSparseEncClass, 181
 - parent_class, 181
- _GstTensorSplit, 182
 - element, 183
 - group_id, 183
 - have_group_id, 183
 - num_srcpads, 183
 - num_tensors, 183
 - silent, 183
 - sink_tensor_conf, 184
 - sinkpad, 184
 - srcpads, 184
 - tensorpick, 184
 - tensorseg, 184
- _GstTensorSplitClass, 185
 - parent_class, 185
- _GstTensorSrcIO, 185
 - base_dir, 187
 - buffer_capacity, 187
 - buffer_data_fp, 187
 - channels, 187
 - channels_enabled, 187
 - configured, 188
 - custom_channel_table, 188
 - default_buffer_capacity, 188
 - default_sampling_frequency, 188
 - default_trigger, 188
 - dev_dir, 189
 - device, 189
 - element, 189
 - is_tensor, 189
 - merge_channels_data, 189
 - mode, 190
 - num_channels_enabled, 190
 - poll_timeout, 190
 - sampling_frequency, 190
 - scan_size, 190
 - silent, 191
 - tensors_config, 191
 - trigger, 191
- _GstTensorSrcIOChannelProperties, 191
 - base_dir, 192
 - base_file, 192
 - big_endian, 192
 - enabled, 193
 - generic_name, 193
 - index, 193
 - is_signed, 193
 - location, 193
 - mask, 194
 - name, 194
 - offset, 194
 - pre_enabled, 194
 - scale, 194
 - shift, 195
 - storage_bits, 195
 - storage_bytes, 195
 - used_bits, 195
- _GstTensorSrcIOClass, 196
 - parent_class, 196
- _GstTensorSrcIODeviceProperties, 196
 - base_dir, 197
 - id, 197
 - name, 197
- _GstTensorTrainer, 198
 - cur_epoch_data_cnt, 199
 - dummy_data_thread, 199
 - element, 199
 - epoch_completion_cond, 199
 - epoch_completion_lock, 199
 - fw, 199
 - fw_created, 200
 - fw_name, 200
 - in_config, 200
 - input_tensors, 200
 - is_epoch_complete, 200
 - is_training_complete, 200
 - notifier, 201
 - out_config, 201
 - output_meta, 201
 - privateData, 201

- prop, 201
- required_sample, 201
- sinkpad, 202
- srcpad, 202
- training_completion_cond, 202
- training_completion_lock, 202
- _GstTensorTrainerClass, 202
 - parent_class, 203
- _GstTensorTrainerEventNotifier, 203
 - notifier, 204
 - version, 204
- _GstTensorTrainerFramework, 204
 - create, 205
 - destroy, 205
 - getFrameworkInfo, 206
 - getStatus, 206
 - push_data, 207
 - start, 207
 - stop, 207
 - version, 209
- _GstTensorTrainerFrameworkInfo, 209
 - name, 209
- _GstTensorTrainerProperties, 210
 - epoch_count, 211
 - input_meta, 211
 - model_config, 211
 - model_load_path, 211
 - model_save_path, 211
 - num_epochs, 212
 - num_inputs, 212
 - num_labels, 212
 - num_training_samples, 212
 - num_validation_samples, 212
 - training_accuracy, 213
 - training_loss, 213
 - validation_accuracy, 213
 - validation_loss, 213
- _GstTensorTransform, 214
 - acceleration, 215
 - apply, 215
 - data_arithmetic, 215
 - data_clamp, 215
 - data_dimchg, 215
 - data_padding, 216
 - data_stand, 216
 - data_transpose, 216
 - data_typecast, 216
 - element, 216
 - in_config, 217
 - loaded, 217
 - mode, 217
 - operators, 217
 - option, 217
 - out_config, 218
 - silent, 218
- _GstTensorTransformClass, 218
 - parent_class, 219
- _NNSTREAMER_ERROR_LENGTH
 - nnstreamer_log.c, 1197
- _NNS_AUDIO
 - tensor_typedef.h, 1154
- _NNS_END
 - tensor_typedef.h, 1155
- _NNS_FLOAT16
 - tensor_typedef.h, 1155
- _NNS_FLOAT32
 - tensor_typedef.h, 1155
- _NNS_FLOAT64
 - tensor_typedef.h, 1155
- _NNS_INT16
 - tensor_typedef.h, 1155
- _NNS_INT32
 - tensor_typedef.h, 1155
- _NNS_INT64
 - tensor_typedef.h, 1155
- _NNS_INT8
 - tensor_typedef.h, 1155
- _NNS_LAYOUT_ANY
 - tensor_typedef.h, 1155
- _NNS_LAYOUT_NCHW
 - tensor_typedef.h, 1155
- _NNS_LAYOUT_NHWC
 - tensor_typedef.h, 1155
- _NNS_LAYOUT_NONE
 - tensor_typedef.h, 1155
- _NNS_MEDIA_ANY
 - tensor_typedef.h, 1154
- _NNS_MEDIA_INVALID
 - tensor_typedef.h, 1154
- _NNS_OCTET
 - tensor_typedef.h, 1154
- _NNS_TENSOR
 - tensor_typedef.h, 1154
- _NNS_TENSOR_FORMAT_END
 - tensor_typedef.h, 1156
- _NNS_TENSOR_FORMAT_FLEXIBLE
 - tensor_typedef.h, 1156
- _NNS_TENSOR_FORMAT_SPARSE
 - tensor_typedef.h, 1156
- _NNS_TENSOR_FORMAT_STATIC
 - tensor_typedef.h, 1156
- _NNS_TEXT
 - tensor_typedef.h, 1154
- _NNS_UINT16
 - tensor_typedef.h, 1155
- _NNS_UINT32
 - tensor_typedef.h, 1155
- _NNS_UINT64
 - tensor_typedef.h, 1155
- _NNS_UINT8
 - tensor_typedef.h, 1155
- _NNS_VIDEO
 - tensor_typedef.h, 1154
- _NNStreamerExternalConverter, 228
 - close, 229
 - convert, 229

- get_out_config, 230
- name, 230
- open, 230
- query_caps, 231
- _NNStreamer_custom_class, 225
 - allocate_invoke, 227
 - destroy_notify, 227
 - exitfunc, 227
 - getInputDim, 227
 - getOutputDim, 227
 - initfunc, 228
 - invoke, 228
 - setInputDim, 228
- _NnstWatchdog, 231
 - cond, 232
 - context, 232
 - lock, 232
 - loop, 232
 - source, 232
 - thread, 232
- _RETURN_ERR_WITH_MSG
 - tensor_filter_support_cc.cc, 1552
- _SANITY_CHECK
 - tensor_filter_support_cc.cc, 1552
- _STR_NULL
 - nnstreamer_plugin_api_util.h, 1052
- __attribute__
 - nnstreamer_log.c, 1197
- __pad0__
 - gsttensor_transform.c, 985
- __write_json
 - gstdataposink.c, 304
- _alloc
 - tensor_allocator.c, 1358
- _append_prev_caps
 - nnstreamer_plugin_api_impl.c, 1209
- _backtrace_to_string
 - nnstreamer_log.c, 1197
 - nnstreamer_log.h, 1204
- _clear_tensorseg
 - gsttensor_split.c, 857
- _close_handle
 - nnstreamer_subplugin.c, 1324
- _compare_rate
 - nnstreamer_plugin_api_util_impl.c, 1255
- _conv_from_f16
 - gsttensor_transform.c, 957
- _conv_to_f16
 - gsttensor_transform.c, 957
- _convert_to_host_byte_order
 - nputil.c, 486
 - nputil.h, 489
- _detect_framework_from_config
 - tensor_filter_common.c, 1443
- _do_init
 - gstdataposink.c, 301
 - gstdataposrc.c, 321
- _double
 - tensor_element, 284
- _extract_mqtt_msg_hdr_from
 - mqttsrc.c, 452
- _fill_in_vstr
 - nnstreamer_conf.c, 1164
- _fill_subplugin_path
 - nnstreamer_conf.c, 1165
- _float
 - tensor_element, 285
- _foreach_custom_property
 - nnstreamer_conf.c, 1165
- _g_list_foreach_vstr_helper
 - nnstreamer_conf.c, 1166
- _g_memdup
 - nnstreamer_util.h, 1121
- _gcd
 - nnstreamer_plugin_api_util_impl.c, 1256
- _get_filenames
 - nnstreamer_conf.c, 1167
- _get_flexible_caps
 - nnstreamer_plugin_api_impl.c, 1210
- _get_subplugin_data
 - nnstreamer_subplugin.c, 1325
- _get_subplugin_with_type
 - nnstreamer_conf.c, 1168
- _get_tensor_caps
 - nnstreamer_plugin_api_impl.c, 1211
- _get_tensors_caps
 - nnstreamer_plugin_api_impl.c, 1212
- _gst_tensor_converter_chain_chunk
 - gsttensor_converter.c, 522
- _gst_tensor_converter_chain_flex_tensor
 - gsttensor_converter.c, 523
- _gst_tensor_converter_chain_octet
 - gsttensor_converter.c, 523
- _gst_tensor_converter_chain_push
 - gsttensor_converter.c, 524
- _gst_tensor_converter_chain_segment
 - gsttensor_converter.c, 525
- _gst_tensor_converter_chain_timestamp
 - gsttensor_converter.c, 526
- _gst_tensor_debug_output
 - gsttensor_debug.c, 593
- _gst_tensor_filter_convert_meta
 - tensor_filter.c, 1404
- _gst_tensor_filter_release_mem_until_idx
 - tensor_filter.c, 1405
- _gst_tensor_filter_transform_check_invoke_result
 - tensor_filter.c, 1405
- _gst_tensor_filter_transform_get_all_input_data
 - tensor_filter.c, 1406
- _gst_tensor_filter_transform_get_invoke_tensors
 - tensor_filter.c, 1406
- _gst_tensor_filter_transform_get_output_data
 - tensor_filter.c, 1407
- _gst_tensor_filter_transform_prepare_output_tensors
 - tensor_filter.c, 1408
- _gst_tensor_filter_transform_update_outbuf

- tensor_filter.c, 1408
- _gst_tensor_filter_transform_validate
 - tensor_filter.c, 1409
- _gst_tensor_query_serversink_playing
 - tensor_query_serversink.c, 1602
- _gst_tensor_query_serversink_start
 - tensor_query_serversink.c, 1602
- _gst_tensor_query_serversrc_get_buffer
 - tensor_query_serversrc.c, 1616
- _gst_tensor_query_serversrc_playing
 - tensor_query_serversrc.c, 1616
- _gst_tensor_query_serversrc_start
 - tensor_query_serversrc.c, 1617
- _gst_tensor_time_sync_buffer_update
 - nnstreamer_plugin_api_impl.c, 1213
- _gst_tensor_time_sync_is_eos
 - nnstreamer_plugin_api_impl.c, 1214
- _gtfc_setprop_ACCELERATOR
 - tensor_filter_common.c, 1443
- _gtfc_setprop_CUSTOM
 - tensor_filter_common.c, 1444
- _gtfc_setprop_DIMENSION
 - tensor_filter_common.c, 1444
- _gtfc_setprop_FRAMEWORK
 - tensor_filter_common.c, 1445
- _gtfc_setprop_INPUTCOMBINATION
 - tensor_filter_common.c, 1446
- _gtfc_setprop_IS_UPDATABLE
 - tensor_filter_common.c, 1446
- _gtfc_setprop_LATENCY
 - tensor_filter_common.c, 1447
- _gtfc_setprop_LAYOUT
 - tensor_filter_common.c, 1447
- _gtfc_setprop_MODEL
 - tensor_filter_common.c, 1448
- _gtfc_setprop_NAME
 - tensor_filter_common.c, 1449
- _gtfc_setprop_OUTPUTCOMBINATION
 - tensor_filter_common.c, 1450
- _gtfc_setprop_PROP_INVOKE_DYNAMIC
 - tensor_filter_common.c, 1450
- _gtfc_setprop_SHARED_TENSOR_FILTER_KEY
 - tensor_filter_common.c, 1450
- _gtfc_setprop_SUSPEND
 - tensor_filter_common.c, 1451
- _gtfc_setprop_THROUGHPUT
 - tensor_filter_common.c, 1451
- _gtfc_setprop_TYPE
 - tensor_filter_common.c, 1452
- _int16_t
 - tensor_element, 285
- _int32_t
 - tensor_element, 285
- _int64_t
 - tensor_element, 285
- _int8_t
 - tensor_element, 285
- _internal_data, 224
 - customFW_private_data, 225
 - methods, 225
 - module, 225
- _is_gst_buffer_timestamp_valid
 - mqttsrc.c, 453
- _is_structure_dimension_same
 - nnstreamer_plugin_api_impl.c, 1214
- _loop_running_cb
 - nnstreamer_watchdog.c, 1346
- _mqtt_set_msg_buf_hdr
 - mqttsink.c, 413
- _mqtt_sink_state_t
 - mqttsink.h, 447
- _nns_edge_event_cb
 - edge_sink.c, 353
 - edge_src.c, 370
 - tensor_query_client.c, 1565
 - tensor_query_serversrc.c, 1617
- _nns_edge_parse_caps
 - tensor_query_client.c, 1566
- _nns_media_type
 - tensor_typedef.h, 1154
- _nns_tensor_layout
 - tensor_typedef.h, 1154
- _nns_tensor_type
 - tensor_typedef.h, 1155
- _nnstreamer_error
 - nnstreamer_log.c, 1198
 - nnstreamer_log.h, 1204
- _nnstreamer_error_clean
 - nnstreamer_log.c, 1198
 - nnstreamer_log.h, 1204
- _nnstreamer_error_write
 - nnstreamer_log.h, 1205
- _nnstreamer_watchdog_thread
 - nnstreamer_watchdog.c, 1346
- _ntp_packet_t, 233
 - li_vn_mode, 234
 - org_ts, 234
 - poll, 234
 - precision, 234
 - recv_ts, 235
 - ref_id, 235
 - ref_ts, 235
 - root_delay, 235
 - root_dispersion, 235
 - stratum, 235
 - xmit_ts, 236
- _ntp_timestamp_t, 236
 - frac, 236
 - sec, 236
- _op_float16
 - gsttensor_transform.c, 957
- _parse_bool_string
 - nnstreamer_conf.c, 1169
- _put_timestamp_on_gst_buf
 - mqttsrc.c, 453
- _put_timestamp_to_msg_buf_hdr

- mqttpsink.c, 414
- _qs_table
 - tensor_query_server.c, 1591
- _repo
 - gsttensor_repo.c, 763
- _reserved_hdr
 - _GstMQTTMessageHdr, 58
- _search_subplugin
 - nnstreamer_subplugin.c, 1325
- _spdata_destroy
 - nnstreamer_subplugin.c, 1326
- _strdup_getenv
 - nnstreamer_conf.c, 1169
- _subscribe
 - mqttpsrc.c, 453
- _tensor_format
 - tensor_typedef.h, 1155
- _tensor_merge_linear, 237
 - direction, 237
- _tensor_sink_signals
 - gsttensor_sink.c, 817
- _tensor_sync_basepad_data, 237
 - duration, 238
 - sink_id, 238
- _tensor_time_sync_data, 238
 - data_basepad, 239
 - mode, 239
 - option, 239
- _tensor_transform_arithmetic, 240
 - ch_dim, 240
 - out_type, 240
 - per_channel_arith, 240
- _tensor_transform_clamp, 241
 - max, 241
 - min, 241
- _tensor_transform_dimchg, 241
 - from, 242
 - to, 242
- _tensor_transform_mode
 - gsttensor_transform.h, 993
- _tensor_transform_padding, 242
 - layout, 243
 - pad, 243
- _tensor_transform_stand, 243
 - mode, 244
 - out_type, 244
 - per_channel, 244
- _tensor_transform_transpose, 244
 - trans_order, 245
- _tensor_transform_typecast, 245
 - to, 245
- _uint16_t
 - tensor_element, 285
- _uint32_t
 - tensor_element, 286
- _uint64_t
 - tensor_element, 286
- _uint8_t
 - tensor_element, 286
- _unsubscribe
 - mqttpsrc.c, 454
- _validate_file
 - nnstreamer_conf.c, 1170
- _wait_connection
 - edge_sink.c, 354
- ABSDIFF
 - gsttensor_rate.c, 732
- acceleration
 - _GstTensorTransform, 215
- ACCL_AUTO
 - nnstreamer_plugin_api_filter.h, 1034
- accl_auto
 - _GstTensorFilterFrameworkInfo, 120
- ACCL_AUTO_STR
 - nnstreamer_plugin_api_filter.h, 1029
- ACCL_CPU
 - nnstreamer_plugin_api_filter.h, 1034
- ACCL_CPU_NEON
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_CPU_NEON_STR
 - nnstreamer_plugin_api_filter.h, 1029
- ACCL_CPU_SIMD
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_CPU_SIMD_STR
 - nnstreamer_plugin_api_filter.h, 1029
- ACCL_CPU_STR
 - nnstreamer_plugin_api_filter.h, 1029
- ACCL_DEFAULT
 - nnstreamer_plugin_api_filter.h, 1034
- accl_default
 - _GstTensorFilterFrameworkInfo, 121
- ACCL_DEFAULT_STR
 - nnstreamer_plugin_api_filter.h, 1029
- ACCL_GPU
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_GPU_STR
 - nnstreamer_plugin_api_filter.h, 1030
- accl_hw
 - nnstreamer_plugin_api_filter.h, 1034
- accl_hw_get_type
 - tensor_filter_common.c, 1452
- ACCL_NONE
 - nnstreamer_plugin_api_filter.h, 1034
- ACCL_NONE_STR
 - nnstreamer_plugin_api_filter.h, 1030
- ACCL_NPU
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_NPU_EDGE_TPU
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_NPU_EDGE_TPU_STR
 - nnstreamer_plugin_api_filter.h, 1030
- ACCL_NPU_MOVIDIUS
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_NPU_MOVIDIUS_STR
 - nnstreamer_plugin_api_filter.h, 1030
- ACCL_NPU_SLSI

- nnstreamer_plugin_api_filter.h, 1035
- ACCL_NPU_SLSI_STR
 - nnstreamer_plugin_api_filter.h, 1030
- ACCL_NPU_SR
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_NPU_SR_STR
 - nnstreamer_plugin_api_filter.h, 1030
- ACCL_NPU_SRCN
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_NPU_SRCN_STR
 - nnstreamer_plugin_api_filter.h, 1031
- ACCL_NPU_STR
 - nnstreamer_plugin_api_filter.h, 1031
- ACCL_NPU_VIVANTE
 - nnstreamer_plugin_api_filter.h, 1035
- ACCL_NPU_VIVANTE_STR
 - nnstreamer_plugin_api_filter.h, 1031
- accl_str
 - _GstTensorFilterProperties, 129
- accumulate_latency
 - tensor_filter.c, 1410
- act_else
 - _GstTensorIf, 137
- act_then
 - _GstTensorIf, 137
- active_sinkpad
 - _GstJoin, 53
- adapter
 - gst_tensor_aggregation_data_s, 251
- adapter_table
 - _GstTensorAggregator, 74
 - _GstTensorConverter, 78
- add_basic_supported_accelerators
 - tensor_filter_common.c, 1453
- AGGREGATION_DEFAULT_KEY
 - nnstreamer_plugin_api_impl.c, 1209
- allocate_in_invoke
 - _FilterTransformData, 30
 - _GTensorFilterSingleClass, 220
 - _GTensorFilterSinglePrivate, 223
 - _GstTensorFilterFramework, 107
 - _GstTensorFilterFrameworkInfo, 121
- allocate_invoke
 - _NNStreamer_custom_class, 227
- allocateInInvoke
 - _GstTensorFilterFramework, 107
- allow_in_place
 - _GstTensorFilterFramework, 107
 - _GstTensorFilterFrameworkInfo, 121
- append_audio_caps_template
 - gsttensor_converter_media_info_audio.h, 558
 - gsttensor_converter_media_no_audio.h, 562
- append_flex_tensor_caps_template
 - gsttensor_converter.c, 519
- append_octet_caps_template
 - gsttensor_converter.c, 519
- append_text_caps_template
 - gsttensor_converter.c, 519
- append_video_caps_template
 - gsttensor_converter_media_info_video.h, 561
 - gsttensor_converter_media_no_video.h, 566
- apply
 - _GstTensorTransform, 215
- applying_ch
 - tensor_transform_operator_s, 290
- aqueue
 - _GstMqttSrc, 68
- array_index
 - _GstDataRepoSrc, 37
- AUDIO_CAPS
 - gstdatareposink.c, 301
- AUDIO_CAPS_STR
 - gsttensor_converter_media_info_audio.h, 559
- auto_accl
 - parse_accl_args, 277
- AVAIL_FREQUENCY_FILE
 - gsttensor_srcio.c, 877
- base
 - dump_buf, 250
- base_dir
 - _GstTensorSrcIO, 187
 - _GstTensorSrcIOChannelProperties, 192
 - _GstTensorSrcIODeviceProperties, 197
- base_file
 - _GstTensorSrcIOChannelProperties, 192
- base_time_epoch
 - _GstMQTTMessageHdr, 58
 - _GstMqttSink, 61
 - _GstMqttSrc, 68
- base_ts
 - _GstTensorRate, 161
- big_endian
 - _GstTensorSrcIOChannelProperties, 192
- BLOCKSIZE
 - gsttensor_srcio.c, 878
- break
 - gsttensor_transform.c, 985
- BUFFER
 - gsttensor_srcio.c, 878
- buffer
 - GstTensorCollectPadData, 255
 - GstTensorRepoData, 271
- buffer_capacity
 - _GstTensorSrcIO, 187
- buffer_data_fp
 - _GstTensorSrcIO, 187
- C_FLAGS
 - gsttensor_debug.c, 590
- cap_mode
 - _GstTensorDebug, 87
- caps
 - _GstDataRepoSrc, 37
 - _GstMqttSrc, 68
 - _GstTensorRepoSrc, 170
 - GstTensorRepoData, 271

CAPS_STRING
 gsttensor_aggregator.c, [494](#)
 gsttensor_debug.c, [590](#)
 gsttensor_decoder.c, [607](#)
 gsttensor_if.c, [656](#)
 gsttensor_merge.c, [687](#)
 gsttensor_rate.c, [732](#)
 gsttensor_reposrc.c, [790](#)
 gsttensor_split.c, [856](#)
 gsttensor_transform.c, [958](#)
 tensor_filter.c, [1402](#)
 CAPS_STRING_SINK
 gsttensor_demux.c, [636](#)
 gsttensor_mux.c, [711](#)
 CAPS_STRING_SRC
 gsttensor_demux.c, [636](#)
 gsttensor_mux.c, [711](#)
 cb
 GstTensorQueryEdgeInfo, [266](#)
 cb_memory_wrapped_destroy
 mqttsrc.c, [455](#)
 cb_mqtt_on_connect
 mqttsink.c, [414](#)
 mqttsrc.c, [455](#)
 cb_mqtt_on_connect_failure
 mqttsink.c, [415](#)
 mqttsrc.c, [456](#)
 cb_mqtt_on_connection_lost
 mqttsink.c, [415](#)
 mqttsrc.c, [457](#)
 cb_mqtt_on_delivery_complete
 mqttsink.c, [416](#)
 cb_mqtt_on_disconnect
 mqttsink.c, [416](#)
 cb_mqtt_on_disconnect_failure
 mqttsink.c, [417](#)
 cb_mqtt_on_message_arrived
 mqttsink.c, [417](#)
 mqttsrc.c, [457](#)
 cb_mqtt_on_send_failure
 mqttsink.c, [418](#)
 cb_mqtt_on_send_success
 mqttsink.c, [418](#)
 cb_mqtt_on_subscribe
 mqttsrc.c, [458](#)
 cb_mqtt_on_subscribe_failure
 mqttsrc.c, [458](#)
 cb_mqtt_on_unsubscribe
 mqttsrc.c, [459](#)
 cb_mqtt_on_unsubscribe_failure
 mqttsrc.c, [459](#)
 ch_dim
 _tensor_transform_arithmetic, [240](#)
 CHANNELS
 gsttensor_srcio.c, [878](#)
 channels
 _GstTensorSrcIO, [187](#)
 channels_enabled
 _GstTensorSrcIO, [187](#)
 CHANNELS_ENABLED_ALL
 gsttensor_srcio.h, [919](#)
 CHANNELS_ENABLED_ALL_CHAR
 gsttensor_srcio.c, [878](#)
 CHANNELS_ENABLED_AUTO
 gsttensor_srcio.h, [919](#)
 CHANNELS_ENABLED_AUTO_CHAR
 gsttensor_srcio.c, [878](#)
 CHANNELS_ENABLED_CUSTOM
 gsttensor_srcio.h, [919](#)
 channels_enabled_options
 gsttensor_srcio.h, [919](#)
 CHECK_HW_AVAILABILITY
 nnstreamer_plugin_api_filter.h, [1035](#)
 checkAvailability
 _GstTensorFilterFramework, [107](#)
 checkGstTensorFilterFrameworkVersion
 nnstreamer_plugin_api_filter.h, [1031](#)
 client_id
 GstMetaQuery, [252](#)
 close
 _GstTensorFilterFramework, [108](#)
 _NNStreamerExternalConverter, [229](#)
 collect
 _GstTensorCrop, [84](#)
 _GstTensorMerge, [142](#)
 _GstTensorMux, [146](#)
 GstTensorCollectPadData, [255](#)
 combi
 _GstTensorFilterPrivate, [125](#)
 concat
 _GstTensorAggregator, [74](#)
 cond
 _GstEdgeSink, [47](#)
 _GstJoin, [54](#)
 _NnstWatchdog, [232](#)
 GstTensorQueryServer, [268](#)
 cond_pull
 GstTensorRepoData, [271](#)
 cond_push
 GstTensorRepoData, [271](#)
 conf
 confdata, [246](#)
 nnstreamer_conf.c, [1178](#)
 CONF_SOURCE_END
 nnstreamer_conf.c, [1164](#)
 CONF_SOURCE_ENVVAR
 nnstreamer_conf.c, [1164](#)
 CONF_SOURCE_EXTRA_CONF
 nnstreamer_conf.c, [1164](#)
 CONF_SOURCE_HARDCODE
 nnstreamer_conf.c, [1164](#)
 CONF_SOURCE_INI
 nnstreamer_conf.c, [1164](#)
 conf_sources
 nnstreamer_conf.c, [1163](#)
 confdata, [246](#)

- conf, 246
- conffile, 246
- enable_envvar, 247
- enable_symlink, 247
- extra_conffile, 247
- loaded, 247
- conffile
 - confdata, 246
- config
 - _GstDataRepoSrc, 37
 - _GstTensorQueryClient, 150
 - _GstTensorRepoSrc, 170
 - GstTensorCropPadData, 256
- config_path
 - _GstTensorDecoder, 90
 - _GstTensorFilterPrivate, 125
- configured
 - _GstTensorDecoder, 91
 - _GstTensorFilterPrivate, 125
 - _GstTensorQueryServerSrc, 157
 - _GstTensorSrcIIO, 188
 - GstTensorQueryServer, 268
- connect_type
 - _GstEdgeSink, 47
 - _GstEdgeSrc, 51
 - _GstTensorQueryClient, 150
 - _GstTensorQueryServerSink, 155
 - _GstTensorQueryServerSrc, 157
- connection_timeout
 - _GstEdgeSink, 47
- context
 - _NnstWatchdog, 232
- convert
 - _NNStreamerExternalConverter, 229
- converter_custom_cb_s, 248
 - data, 248
 - func, 248
- cpu_neon_accel_available
 - hw_accel.c, 996
 - hw_accel.h, 997
- create
 - _GstTensorTrainerFramework, 205
- create_regex
 - tensor_filter_common.c, 1453
- cumulative_tensors
 - _GstDataRepoSink, 32
- cur_epoch_data_cnt
 - _GstTensorTrainer, 199
- current_sample_index
 - _GstDataRepoSrc, 38
- current_time
 - _GstTensorMerge, 142
 - _GstTensorMux, 146
- CURRENT_TRIGGER
 - gsttensor_srcio.c, 878
- cursor
 - vstr_helper, 291
- custom
 - _GstTensorConverter, 78
 - _GstTensorDecoder, 91
 - _GstTensorFilterFrameworkEventData, 117
 - _GstTensorIf, 137
 - custom_allocateInInvoke
 - tensor_filter_custom.c, 1520
 - custom_cb_s, 248
 - data, 249
 - func, 249
 - name, 249
 - custom_channel_table
 - _GstTensorSrcIIO, 188
 - custom_checkAvailability
 - tensor_filter_custom.c, 1520
 - custom_close
 - tensor_filter_custom.c, 1520
 - tensor_filter_custom_easy.c, 1528
 - custom_configured
 - _GstTensorIf, 137
 - custom_destroyNotify
 - tensor_filter_custom.c, 1521
 - custom_eventHandler
 - tensor_filter_custom_easy.c, 1528
 - custom_free_internal_data
 - tensor_filter_custom_easy.c, 1528
 - custom_getFrameworkInfo
 - tensor_filter_custom_easy.c, 1529
 - custom_getInputDim
 - tensor_filter_custom.c, 1521
 - custom_getModelInfo
 - tensor_filter_custom_easy.c, 1529
 - custom_getOutputDim
 - tensor_filter_custom.c, 1522
 - custom_invoke
 - tensor_filter_custom.c, 1522
 - tensor_filter_custom_easy.c, 1530
 - custom_lib
 - _GstEdgeSink, 48
 - _GstEdgeSrc, 51
 - custom_loadlib
 - tensor_filter_custom.c, 1522
 - custom_open
 - tensor_filter_custom.c, 1523
 - tensor_filter_custom_easy.c, 1530
 - CUSTOM_PROP
 - nnstreamer_plugin_api_filter.h, 1035
 - custom_properties
 - _GstTensorFilterFrameworkEventData, 117
 - _GstTensorFilterProperties, 129
 - custom_setInputDim
 - tensor_filter_custom.c, 1524
 - custom_table
 - nnstreamer_conf.c, 1178
 - customFW_private_data
 - _internal_data, 225
- cv
 - _GstTensorIf, 137
 - gsttensor_if.c, 676

- cv_option
 - [_GstTensorIf](#), 137
- data
 - [_GstTensorFilterFrameworkEventData](#), 118
 - [converter_custom_cb_s](#), 248
 - [custom_cb_s](#), 249
 - [decoder_custom_cb_s](#), 250
 - [gsttensor_if.c](#), 677
 - [GstTensorCropPadData](#), 257
 - [GstTensorMemory](#), 261
 - [tensor_data_s](#), 283
 - [tensor_if_sv_s](#), 287
- data_arithmetic
 - [_GstTensorTransform](#), 215
- data_basepad
 - [_tensor_time_sync_data](#), 239
- data_clamp
 - [_GstTensorTransform](#), 215
- data_dimchg
 - [_GstTensorTransform](#), 215
- data_linear
 - [_GstTensorMerge](#), 142
- data_padding
 - [_GstTensorTransform](#), 216
- data_stand
 - [_GstTensorTransform](#), 216
- data_transpose
 - [_GstTensorTransform](#), 216
- data_type
 - [_GstDataRepoSink](#), 32
 - [_GstDataRepoSrc](#), 38
- data_typecast
 - [_GstTensorTransform](#), 216
- [datarepo/gstdatarepo.c](#), 293
- [datarepo/gstdatarepo.h](#), 296
- [datarepo/gstdatareposink.c](#), 298
- [datarepo/gstdatareposink.h](#), 316
- [datarepo/gstdatareposrc.c](#), 319
- [datarepo/gstdatareposrc.h](#), 341
- DBG
 - [gsttensor_aggregator.c](#), 494
 - [gsttensor_converter.c](#), 519
 - [gsttensor_debug.c](#), 590
 - [gsttensor_decoder.c](#), 607
 - [gsttensor_if.c](#), 656
 - [gsttensor_merge.c](#), 687
 - [gsttensor_mux.c](#), 711
 - [gsttensor_rate.c](#), 732
 - [gsttensor_repo.c](#), 754
 - [gsttensor_sink.c](#), 803
 - [gsttensor_sparsedec.c](#), 822
 - [gsttensor_sparseenc.c](#), 836
 - [gsttensor_srcii.c](#), 879
 - [gsttensor_transform.c](#), 958
 - [tensor_filter_common.h](#), 1498
 - [tensor_query_client.c](#), 1563
- debug
 - [_GstMqttSink](#), 61
 - [_GstMqttSrc](#), 68
- decode
 - [_GstTensorDecoderDef](#), 94
- decoder
 - [_GstTensorDecoder](#), 91
- [decoder_custom_cb_s](#), 249
 - [data](#), 250
 - [func](#), 250
- def_accl
 - [parse_accl_args](#), 277
- DEFAULT_ACCELERATION
 - [gsttensor_transform.c](#), 958
- DEFAULT_BUFFER_CAPACITY
 - [gsttensor_srcii.c](#), 879
- default_buffer_capacity
 - [_GstTensorSrcII](#), 188
- DEFAULT_CLIENT_TIMEOUT
 - [tensor_query_client.c](#), 1563
- DEFAULT_CONCAT
 - [gsttensor_aggregator.c](#), 494
- DEFAULT_CONNECT_TYPE
 - [edge_common.h](#), 347
 - [tensor_query_common.h](#), 1582
- DEFAULT_DATA_POP_TIMEOUT
 - [tensor_query_serversrc.c](#), 1614
- DEFAULT_DEBUG
 - [mqttsink.c](#), 413
 - [mqttsrc.c](#), 452
- DEFAULT_EMIT_SIGNAL
 - [gsttensor_sink.c](#), 803
- DEFAULT_EPOCHS
 - [gstdatareposrc.c](#), 322
- DEFAULT_FRAMES_DIMENSION
 - [gsttensor_aggregator.c](#), 494
- DEFAULT_FRAMES_FLUSH
 - [gsttensor_aggregator.c](#), 494
- DEFAULT_FRAMES_IN
 - [gsttensor_aggregator.c](#), 494
- DEFAULT_FRAMES_OUT
 - [gsttensor_aggregator.c](#), 495
- DEFAULT_FRAMES_PER_TENSOR
 - [gsttensor_converter.c](#), 519
- DEFAULT_FREQUENCY
 - [gsttensor_srcii.c](#), 879
- DEFAULT_HOST
 - [edge_common.h](#), 347
 - [tensor_query_common.h](#), 1582
- DEFAULT_INDEX
 - [gstdatareposrc.c](#), 322
 - [gsttensor_reposink.c](#), 774
 - [gsttensor_reposrc.c](#), 790
- DEFAULT_IS_LIVE
 - [mqttsrc.c](#), 452
 - [tensor_query_serversrc.c](#), 1614
- DEFAULT_IS_SHUFFLE
 - [gstdatareposrc.c](#), 322
- DEFAULT_LATENESS
 - [gsttensor_crop.c](#), 571

- DEFAULT_MAX_MSG_BUF_SIZE
 - mqttsink.c, [413](#)
- DEFAULT_MAX_REQUEST
 - tensor_query_client.c, [1563](#)
- DEFAULT_MERGE_CHANNELS
 - gsttensor_srcio.c, [880](#)
- DEFAULT_METALESS_FRAME_LIMIT
 - tensor_query_serversink.c, [1601](#)
- DEFAULT_MQTT_CLIENT_ID
 - mqttsink.c, [441](#)
 - mqttsrc.c, [479](#)
- DEFAULT_MQTT_CLIENT_ID_FORMAT
 - mqttsink.c, [441](#)
 - mqttsrc.c, [479](#)
- DEFAULT_MQTT_CONN_TIMEOUT_SEC
 - mqttcommon.h, [404](#)
- DEFAULT_MQTT_DISCONNECT_TIMEOUT
 - mqttsink.c, [413](#)
- default_mqtt_get_unix_epoch
 - mqttcommon.h, [406](#)
- DEFAULT_MQTT_HOST
 - edge_sink.c, [352](#)
 - tensor_query_serversrc.c, [1614](#)
- DEFAULT_MQTT_HOST_ADDRESS
 - mqttsink.c, [442](#)
 - mqttsrc.c, [479](#)
- DEFAULT_MQTT_HOST_PORT
 - mqttsink.c, [442](#)
 - mqttsrc.c, [479](#)
- DEFAULT_MQTT_NTP_SERVERS
 - mqttsink.c, [442](#)
- DEFAULT_MQTT_NTP_SYNC
 - mqttsink.c, [413](#)
- DEFAULT_MQTT_OPT_CLEANSESSION
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- DEFAULT_MQTT_OPT_KEEP_ALIVE_INTERVAL
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- DEFAULT_MQTT_PORT
 - edge_sink.c, [352](#)
 - tensor_query_serversrc.c, [1614](#)
- DEFAULT_MQTT_PUB_TOPIC
 - mqttsink.c, [442](#)
- DEFAULT_MQTT_PUB_TOPIC_FORMAT
 - mqttsink.c, [442](#)
- DEFAULT_MQTT_PUB_WAIT_TIMEOUT
 - mqttsink.c, [413](#)
- DEFAULT_MQTT_QOS
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- DEFAULT_MQTT_SUB_TIMEOUT
 - mqttsrc.c, [452](#)
- DEFAULT_MQTT_SUB_TIMEOUT_MIN
 - mqttsrc.c, [452](#)
- DEFAULT_NUM_BUFFERS
 - mqttsink.c, [413](#)
- DEFAULT_OPERATING_CHANNELS_ENABLED
 - gsttensor_srcio.c, [880](#)
- DEFAULT_OPERATING_MODE
 - gsttensor_srcio.c, [880](#)
- DEFAULT_POLL_TIMEOUT
 - gsttensor_srcio.c, [880](#)
- DEFAULT_PORT
 - edge_common.h, [347](#)
- DEFAULT_PORT_SRC
 - tensor_query_serversrc.c, [1615](#)
- DEFAULT_PROP_BASE_DIRECTORY
 - gsttensor_srcio.c, [880](#)
- DEFAULT_PROP_DEV_DIRECTORY
 - gsttensor_srcio.c, [880](#)
- DEFAULT_PROP_DEVICE_NUM
 - gsttensor_srcio.c, [881](#)
- DEFAULT_PROP_EPOCHS
 - gsttensor_trainer.c, [923](#)
- DEFAULT_PROP_INPUT_LIST
 - gsttensor_trainer.c, [923](#)
- DEFAULT_PROP_LABEL_LIST
 - gsttensor_trainer.c, [923](#)
- DEFAULT_PROP_SILENT
 - gsttensor_srcio.c, [881](#)
- DEFAULT_PROP_STRING
 - gsttensor_srcio.c, [881](#)
- DEFAULT_PROP_TRAIN_SAMPLES
 - gsttensor_trainer.c, [924](#)
- DEFAULT_PROP_TRIGGER_NUM
 - gsttensor_srcio.c, [881](#)
- DEFAULT_PROP_VALID_SAMPLES
 - gsttensor_trainer.c, [924](#)
- DEFAULT_QOS
 - gsttensor_reposink.c, [774](#)
 - gsttensor_sink.c, [803](#)
 - mqttsink.c, [413](#)
- DEFAULT_QUERY_INFO_TIMEOUT
 - tensor_query_server.h, [1593](#)
- default_sampling_frequency
 - _GstTensorSrcIO, [188](#)
- DEFAULT_SERVER_ID
 - tensor_query_server.h, [1593](#)
- DEFAULT_SET_TIMESTAMP
 - gsttensor_converter.c, [519](#)
- DEFAULT_SIGNAL_RATE
 - gsttensor_reposink.c, [774](#)
 - gsttensor_sink.c, [803](#)
- DEFAULT_SILENT
 - gsttensor_aggregator.c, [495](#)
 - gsttensor_converter.c, [520](#)
 - gsttensor_crop.c, [571](#)
 - gsttensor_debug.c, [591](#)
 - gsttensor_decoder.c, [607](#)
 - gsttensor_rate.c, [733](#)
 - gsttensor_reposink.c, [775](#)
 - gsttensor_reposrc.c, [790](#)
 - gsttensor_sink.c, [803](#)
 - gsttensor_sparsedec.c, [822](#)
 - gsttensor_sparseenc.c, [836](#)

- tensor_query_client.c, 1563
- DEFAULT_STR_PROP_VALUE
 - gsttensor_trainer.c, 924
- DEFAULT_SYNC
 - gsttensor_sink.c, 804
 - mqttsink.c, 413
- DEFAULT_TENSOR_DEBUG_CAP
 - gsttensor_debug.c, 591
- DEFAULT_TENSOR_DEBUG_META_FLAGS
 - gsttensor_debug.c, 591
- DEFAULT_TENSOR_DEBUG_OUTPUT_FLAGS
 - gsttensor_debug.c, 591
- DEFAULT_THROTTLE
 - gsttensor_rate.c, 733
- default_trigger
 - _GstTensorSrcIIO, 188
- dest_host
 - _GstEdgeSink, 48
 - _GstEdgeSrc, 51
 - _GstTensorQueryClient, 150
 - _GstTensorQueryServerSrc, 157
 - GstTensorQueryEdgeInfo, 266
- dest_port
 - _GstEdgeSink, 48
 - _GstEdgeSrc, 51
 - _GstTensorQueryClient, 151
 - _GstTensorQueryServerSrc, 158
 - GstTensorQueryEdgeInfo, 267
- destroy
 - _GstTensorTrainerFramework, 205
- DESTROY_NOTIFY
 - nnstreamer_plugin_api_filter.h, 1035
- destroy_notify
 - _GTensorFilterSingleClass, 220
 - _NNStreamer_custom_class, 227
- destroyNotify
 - _GstTensorFilterFramework, 108
- dev_dir
 - _GstTensorSrcIIO, 189
- DEVICE
 - gsttensor_srciiio.c, 881
- device
 - _GstTensorSrcIIO, 189
- DEVICE_PREFIX
 - gsttensor_srciiio.c, 882
- dimension
 - GstTensorInfo, 260
 - GstTensorMetaInfo, 263
- direction
 - _tensor_merge_linear, 237
- do_not_append_header
 - _GstTensorConverter, 79
- drop
 - _GstTensorRate, 161
- dts
 - _GstMQTTMessageHdr, 58
- dummy_data_thread
 - _GstTensorTrainer, 199
- dump_buf, 250
 - base, 250
 - pos, 251
 - size, 251
- dup
 - _GstTensorRate, 161
- duration
 - _GstMQTTMessageHdr, 59
 - _tensor_sync_basepad_data, 238
- edge/edge_common.c, 344
- edge/edge_common.h, 346
- edge/edge_elements.c, 349
- edge/edge_sink.c, 350
- edge/edge_sink.h, 363
- edge/edge_src.c, 367
- edge/edge_src.h, 379
- edge_common.c
 - gst_edge_get_connect_type, 345
- edge_common.h
 - DEFAULT_CONNECT_TYPE, 347
 - DEFAULT_HOST, 347
 - DEFAULT_PORT, 347
 - GST_EDGE_ELEM_NAME_SINK, 347
 - GST_EDGE_ELEM_NAME_SRC, 348
 - gst_edge_get_connect_type, 348
 - GST_EDGE_PACKAGE, 348
 - GST_TYPE_EDGE_CONNECT_TYPE, 348
- edge_elements.c
 - PACKAGE, 350
 - plugin_init, 350
- edge_h
 - _GstEdgeSink, 48
 - _GstEdgeSrc, 51
 - _GstTensorQueryClient, 151
 - GstTensorQueryServer, 268
- edge_sink.c
 - _nns_edge_event_cb, 353
 - _wait_connection, 354
 - DEFAULT_MQTT_HOST, 352
 - DEFAULT_MQTT_PORT, 352
 - G_DEFINE_TYPE, 354
 - GST_CAT_DEFAULT, 352
 - GST_DEBUG_CATEGORY_STATIC, 354
 - gst_edgesink_class_init, 355
 - gst_edgesink_finalize, 355
 - gst_edgesink_get_connect_type, 356
 - gst_edgesink_get_host, 356
 - gst_edgesink_get_port, 357
 - gst_edgesink_get_property, 357
 - gst_edgesink_init, 358
 - gst_edgesink_parent_class, 353
 - gst_edgesink_render, 358
 - gst_edgesink_set_caps, 359
 - gst_edgesink_set_connect_type, 360
 - gst_edgesink_set_host, 360
 - gst_edgesink_set_port, 361
 - gst_edgesink_set_property, 361
 - gst_edgesink_start, 362

- gst_edgesink_stop, [362](#)
- PROP_0, [353](#)
- PROP_CONNECT_TYPE, [353](#)
- PROP_CONNECTION_TIMEOUT, [353](#)
- PROP_CUSTOM_LIB, [353](#)
- PROP_DEST_HOST, [353](#)
- PROP_DEST_PORT, [353](#)
- PROP_HOST, [353](#)
- PROP_LAST, [353](#)
- PROP_PORT, [353](#)
- PROP_TOPIC, [353](#)
- PROP_WAIT_CONNECTION, [353](#)
- sinktemplate, [363](#)
- edge_sink.h
 - GST_EDGESINK, [365](#)
 - GST_EDGESINK_CAST, [365](#)
 - GST_EDGESINK_CLASS, [365](#)
 - gst_edgesink_get_type, [367](#)
 - GST_IS_EDGESINK, [366](#)
 - GST_IS_EDGESINK_CLASS, [366](#)
 - GST_TYPE_EDGESINK, [366](#)
 - GstEdgeSink, [366](#)
 - GstEdgeSinkClass, [366](#)
- edge_src.c
 - _nns_edge_event_cb, [370](#)
 - G_DEFINE_TYPE, [370](#)
 - GST_CAT_DEFAULT, [369](#)
 - GST_DEBUG_CATEGORY_STATIC, [370](#)
 - gst_edgesrc_change_state, [371](#)
 - gst_edgesrc_class_finalize, [371](#)
 - gst_edgesrc_class_init, [372](#)
 - gst_edgesrc_create, [372](#)
 - gst_edgesrc_get_connect_type, [373](#)
 - gst_edgesrc_get_dest_host, [374](#)
 - gst_edgesrc_get_dest_port, [374](#)
 - gst_edgesrc_get_property, [374](#)
 - gst_edgesrc_init, [375](#)
 - gst_edgesrc_parent_class, [369](#)
 - gst_edgesrc_set_connect_type, [375](#)
 - gst_edgesrc_set_dest_host, [376](#)
 - gst_edgesrc_set_dest_port, [376](#)
 - gst_edgesrc_set_property, [377](#)
 - gst_edgesrc_start, [377](#)
 - gst_edgesrc_stop, [378](#)
 - PROP_0, [370](#)
 - PROP_CONNECT_TYPE, [370](#)
 - PROP_CUSTOM_LIB, [370](#)
 - PROP_DEST_HOST, [370](#)
 - PROP_DEST_PORT, [370](#)
 - PROP_HOST, [370](#)
 - PROP_LAST, [370](#)
 - PROP_PORT, [370](#)
 - PROP_TOPIC, [370](#)
 - srctemplate, [379](#)
- edge_src.h
 - GST_EDGESRC, [380](#)
 - GST_EDGESRC_CAST, [381](#)
 - GST_EDGESRC_CLASS, [381](#)
 - gst_edgesrc_get_type, [382](#)
 - GST_IS_EDGESRC, [381](#)
 - GST_IS_EDGESRC_CLASS, [381](#)
 - GST_TYPE_EDGESRC, [381](#)
 - GstEdgeSrc, [382](#)
 - GstEdgeSrcClass, [382](#)
- element
 - _GTensorFilterSingle, [219](#)
 - _GstDataRepoSink, [32](#)
 - _GstEdgeSink, [48](#)
 - _GstEdgeSrc, [51](#)
 - _GstJoin, [54](#)
 - _GstTensorAggregator, [74](#)
 - _GstTensorConverter, [79](#)
 - _GstTensorCrop, [85](#)
 - _GstTensorDebug, [88](#)
 - _GstTensorDecoder, [91](#)
 - _GstTensorDemux, [98](#)
 - _GstTensorFilter, [101](#)
 - _GstTensorIf, [138](#)
 - _GstTensorMerge, [142](#)
 - _GstTensorMux, [146](#)
 - _GstTensorQueryClient, [151](#)
 - _GstTensorQueryServerSink, [155](#)
 - _GstTensorQueryServerSrc, [158](#)
 - _GstTensorRate, [161](#)
 - _GstTensorRepoSink, [166](#)
 - _GstTensorSink, [173](#)
 - _GstTensorSparseDec, [177](#)
 - _GstTensorSparseEnc, [179](#)
 - _GstTensorSplit, [183](#)
 - _GstTensorSrcIIO, [189](#)
 - _GstTensorTrainer, [199](#)
 - _GstTensorTransform, [216](#)
- else
 - gsttensor_srcio.c, [914](#)
- else_option
 - _GstTensorIf, [138](#)
- emit_signal
 - _GstTensorSink, [173](#)
- EN_SUFFIX
 - gsttensor_srcio.c, [882](#)
- enable_envvar
 - confdata, [247](#)
- enable_symlink
 - confdata, [247](#)
- enabled
 - _GstTensorSrcIIOChannelProperties, [193](#)
- eos
 - _GstTensorSinkClass, [174](#)
 - GstTensorRepoData, [272](#)
- epoch_completion_cond
 - _GstTensorTrainer, [199](#)
- epoch_completion_lock
 - _GstTensorTrainer, [199](#)
- epoch_count
 - _GstTensorTrainerProperties, [211](#)
- epochs

- [_GstDataRepoSrc, 38](#)
- EREMOTEIO
 - [tensor_query_common.c, 1580](#)
- err
 - [_GstMqttSink, 61](#)
 - [_GstMqttSrc, 68](#)
- errmsg
 - [nnstreamer_log.c, 1199](#)
- errmsg_reported
 - [nnstreamer_log.c, 1199](#)
- EVENT_NAME_UPDATE_MODEL
 - [tensor_filter.c, 1402](#)
- event_ops
 - [nnstreamer_plugin_api_filter.h, 1035](#)
- eventHandler
 - [_GstTensorFilterFramework, 109](#)
- exit
 - [_GstTensorDecoderDef, 95](#)
- exitfunc
 - [_NNStreamer_custom_class, 227](#)
- ext_fw
 - [_GstTensorConverter, 79](#)
- externalConverter
 - [_GstTensorConverter, 79](#)
- extra
 - [GstTensorsInfo, 275](#)
- extra_confdata
 - [confdata, 247](#)
- failed
 - [gsttensor_decoder.c, 609](#)
- FALSE
 - [gsttensor_transform.c, 985](#)
 - [nnstreamer.c, 1356](#)
- fd
 - [_GstDataRepoSink, 32](#)
 - [_GstDataRepoSrc, 38](#)
- fd_offset
 - [_GstDataRepoSink, 32](#)
 - [_GstDataRepoSrc, 38](#)
- file_size
 - [_GstDataRepoSrc, 39](#)
- filename
 - [_GstDataRepoSink, 32](#)
 - [_GstDataRepoSrc, 39](#)
- files
 - [subplugin_conf, 280](#)
- filter_priv
 - [_GTensorFilterSinglePrivate, 223](#)
- filter_subplugin_custom
 - [tensor_filter_custom.c, 1525](#)
- filter_supported_accelerators
 - [tensor_filter_common.c, 1454](#)
- FilterTransformData
 - [tensor_filter.c, 1404](#)
- find_key_strv
 - [nnstreamer_plugin_api_util.h, 1053](#)
 - [nnstreamer_plugin_api_util_impl.c, 1256](#)
- findExternalConverter
 - [gsttensor_converter.c, 526](#)
- fini_filter_custom
 - [tensor_filter_custom.c, 1524](#)
- fini_filter_custom_easy
 - [tensor_filter_custom_easy.c, 1531](#)
- fini_queryserver
 - [tensor_query_server.c, 1584](#)
- fini_subplugin
 - [nnstreamer_subplugin.c, 1327](#)
- first_epoch_is_done
 - [_GstDataRepoSrc, 39](#)
- fixed_caps
 - [_GstDataRepoSink, 33](#)
- float16_not_supported
 - [gsttensor_transform.c, 962](#)
- format
 - [GstTensorMetaInfo, 263](#)
 - [GstTensorsInfo, 275](#)
- forward_sticky_events
 - [gstjoin.c, 388](#)
- fps_d
 - [_GstTensorRepoSrc, 170](#)
- fps_n
 - [_GstTensorRepoSrc, 170](#)
- frac
 - [_ntp_timestamp_t, 236](#)
- frame_size
 - [_GstTensorConverter, 79](#)
- frames_dim
 - [_GstTensorAggregator, 74](#)
- frames_flush
 - [_GstTensorAggregator, 74](#)
- frames_in
 - [_GstTensorAggregator, 75](#)
- frames_out
 - [_GstTensorAggregator, 75](#)
- frames_per_tensor
 - [_GstTensorConverter, 80](#)
- from
 - [_tensor_transform_dimchg, 242](#)
- from_rate_denominator
 - [_GstTensorRate, 161](#)
- from_rate_numerator
 - [_GstTensorRate, 162](#)
- func
 - [converter_custom_cb_s, 248](#)
 - [custom_cb_s, 249](#)
 - [decoder_custom_cb_s, 250](#)
- fw
 - [_GstTensorFilterPrivate, 125](#)
 - [_GstTensorTrainer, 199](#)
- fw_created
 - [_GstTensorTrainer, 200](#)
- fw_name
 - [_GstTensorTrainer, 200](#)
- fw_opened
 - [_GstTensorFilterProperties, 130](#)
- fwname

- [_GstTensorFilterProperties](#), 130
- [g_assert](#)
 - [gsttensor_srcii.c](#), 887
- [G_DEFINE_TYPE](#)
 - [edge_sink.c](#), 354
 - [edge_src.c](#), 370
 - [gstjoin.c](#), 388
 - [gsttensor_aggregator.c](#), 496
 - [gsttensor_converter.c](#), 527
 - [gsttensor_crop.c](#), 572
 - [gsttensor_debug.c](#), 593
 - [gsttensor_decoder.c](#), 610
 - [gsttensor_demux.c](#), 637
 - [gsttensor_if.c](#), 658
 - [gsttensor_merge.c](#), 688
 - [gsttensor_mux.c](#), 712
 - [gsttensor_rate.c](#), 735
 - [gsttensor_reposink.c](#), 776
 - [gsttensor_reposrc.c](#), 791
 - [gsttensor_sink.c](#), 805
 - [gsttensor_sparsedec.c](#), 823
 - [gsttensor_sparseenc.c](#), 837
 - [gsttensor_split.c](#), 858
 - [gsttensor_srcii.c](#), 888
 - [gsttensor_trainer.c](#), 926
 - [gsttensor_transform.c](#), 963
 - [mqttSink.c](#), 419
 - [mqttSrc.c](#), 459
 - [tensor_allocator.c](#), 1359
 - [tensor_filter.c](#), 1410
 - [tensor_query_client.c](#), 1566
 - [tensor_query_serversink.c](#), 1603
 - [tensor_query_serversrc.c](#), 1618
- [G_DEFINE_TYPE_WITH_CODE](#)
 - [gstdatareposink.c](#), 304
 - [gstdatareposrc.c](#), 324
 - [gstjoin.c](#), 388
- [G_DEFINE_TYPE_WITH_PRIVATE](#)
 - [tensor_filter_single.c](#), 1537
- [g_free](#)
 - [gsttensor_decoder.c](#), 610
- [g_free_const](#)
 - [tensor_filter_common.c](#), 1440
- [G_IS_TENSOR_FILTER_SINGLE](#)
 - [tensor_filter_single.h](#), 1549
- [G_IS_TENSOR_FILTER_SINGLE_CLASS](#)
 - [tensor_filter_single.h](#), 1549
- [G_LOCK_DEFINE_STATIC](#)
 - [nnstreamer_log.c](#), 1198
 - [nnstreamer_subplugin.c](#), 1327
 - [tensor_filter_common.c](#), 1455
 - [tensor_query_server.c](#), 1584
- [g_strfreev_const](#)
 - [tensor_filter_common.c](#), 1440
- [g_tensor_filter_allocate_in_invoke](#)
 - [tensor_filter_single.c](#), 1537
- [g_tensor_filter_destroy_notify](#)
 - [tensor_filter_single.c](#), 1538
- [g_tensor_filter_input_configured](#)
 - [tensor_filter_single.c](#), 1539
- [g_tensor_filter_output_configured](#)
 - [tensor_filter_single.c](#), 1539
- [g_tensor_filter_set_input_info](#)
 - [tensor_filter_single.c](#), 1539
- [G_TENSOR_FILTER_SINGLE](#)
 - [tensor_filter_single.h](#), 1549
- [G_TENSOR_FILTER_SINGLE_CAST](#)
 - [tensor_filter_single.h](#), 1550
- [G_TENSOR_FILTER_SINGLE_CLASS](#)
 - [tensor_filter_single.h](#), 1550
- [g_tensor_filter_single_class_init](#)
 - [tensor_filter_single.c](#), 1540
- [g_tensor_filter_single_finalize](#)
 - [tensor_filter_single.c](#), 1541
- [g_tensor_filter_single_get_property](#)
 - [tensor_filter_single.c](#), 1542
- [g_tensor_filter_single_get_type](#)
 - [tensor_filter_single.h](#), 1551
- [g_tensor_filter_single_init](#)
 - [tensor_filter_single.c](#), 1543
- [g_tensor_filter_single_invoke](#)
 - [tensor_filter_single.c](#), 1543
- [g_tensor_filter_single_parent_class](#)
 - [tensor_filter_single.c](#), 1537
- [G_TENSOR_FILTER_SINGLE_PRIV](#)
 - [tensor_filter_single.c](#), 1537
- [g_tensor_filter_single_set_property](#)
 - [tensor_filter_single.c](#), 1544
- [g_tensor_filter_single_start](#)
 - [tensor_filter_single.c](#), 1545
- [g_tensor_filter_single_stop](#)
 - [tensor_filter_single.c](#), 1546
- [G_TYPE_TENSOR_FILTER_SINGLE](#)
 - [tensor_filter_single.h](#), 1550
- [g_value_set_string](#)
 - [gsttensor_decoder.c](#), 610
- [generic_name](#)
 - [_GstTensorSrcIOChannelProperties](#), 193
- [get_accl_hw_str](#)
 - [nnstreamer_plugin_api_filter.h](#), 1036
 - [tensor_filter_common.c](#), 1455
- [get_accl_hw_type](#)
 - [nnstreamer_plugin_api_filter.h](#), 1037
 - [tensor_filter_common.c](#), 1456
- [get_all_subplugins](#)
 - [nnstreamer_subplugin.c](#), 1327
 - [nnstreamer_subplugin.h](#), 1337
- [get_epoch_func](#)
 - [_GstMqttSink](#), 61
- [GET_IN_OUT_INFO](#)
 - [nnstreamer_plugin_api_filter.h](#), 1036
- [get_out_config](#)
 - [_NNStreamerExternalConverter](#), 230
- [get_subplugin](#)
 - [nnstreamer_subplugin.c](#), 1328
 - [nnstreamer_subplugin.h](#), 1338

- GET_TFSP_WITH_CHECKS
 - tensor_filter_support_cc.cc, [1553](#)
- getFrameworkInfo
 - _GstTensorFilterFramework, [109](#)
 - _GstTensorTrainerFramework, [206](#)
- getInputDim
 - _NNStreamer_custom_class, [227](#)
- getInputDimension
 - _GstTensorFilterFramework, [110](#)
- getModelInfo
 - _GstTensorFilterFramework, [110](#)
- getOutCaps
 - _GstTensorDecoderDef, [95](#)
- getOutputDim
 - _NNStreamer_custom_class, [227](#)
- getOutputDimension
 - _GstTensorFilterFramework, [111](#)
- getStatus
 - _GstTensorTrainerFramework, [206](#)
- getTransformSize
 - _GstTensorDecoderDef, [95](#)
- gquark_err_tag
 - _GstMqttSink, [62](#)
 - _GstMqttSrc, [69](#)
- group_id
 - _GstJoinPad, [56](#)
 - _GstTensorDemux, [98](#)
 - _GstTensorSplit, [183](#)
- GST_AUDIO_FORMAT_F32
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_FORMAT_F64
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_FORMAT_S16
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_FORMAT_S32
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_FORMAT_S8
 - gsttensor_converter_media_no_audio.h, [564](#)
- gst_audio_format_to_string
 - gsttensor_converter_media_no_audio.h, [563](#)
- GST_AUDIO_FORMAT_U16
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_FORMAT_U32
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_FORMAT_U8
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_FORMAT_UNKNOWN
 - gsttensor_converter_media_no_audio.h, [564](#)
- GST_AUDIO_INFO_BPF
 - gsttensor_converter_media_no_audio.h, [563](#)
- GST_AUDIO_INFO_CHANNELS
 - gsttensor_converter_media_no_audio.h, [563](#)
- GST_AUDIO_INFO_FORMAT
 - gsttensor_converter_media_no_audio.h, [563](#)
- gst_audio_info_from_caps
 - gsttensor_converter_media_no_audio.h, [563](#)
- gst_audio_info_init
 - gsttensor_converter_media_no_audio.h, [563](#)
- GST_AUDIO_INFO_RATE
 - gsttensor_converter_media_no_audio.h, [564](#)
- gst_buffer_add_meta_query
 - tensor_meta.h, [1558](#)
- gst_buffer_get_meta_query
 - tensor_meta.h, [1559](#)
- gst_caps_str
 - _GstMQTTMessageHdr, [59](#)
- GST_CAT_DEFAULT
 - edge_sink.c, [352](#)
 - edge_src.c, [369](#)
 - gstdatareposink.c, [301](#)
 - gstdatareposrc.c, [322](#)
 - gstjoin.c, [385](#)
 - gsttensor_aggregator.c, [495](#)
 - gsttensor_converter.c, [520](#)
 - gsttensor_crop.c, [571](#)
 - gsttensor_debug.c, [591](#)
 - gsttensor_decoder.c, [607](#)
 - gsttensor_demux.c, [637](#)
 - gsttensor_if.c, [656](#)
 - gsttensor_merge.c, [688](#)
 - gsttensor_mux.c, [711](#)
 - gsttensor_rate.c, [733](#)
 - gsttensor_reposink.c, [775](#)
 - gsttensor_reposrc.c, [790](#)
 - gsttensor_sink.c, [804](#)
 - gsttensor_sparsedec.c, [823](#)
 - gsttensor_sparseenc.c, [836](#)
 - gsttensor_split.c, [856](#)
 - gsttensor_srcio.c, [882](#)
 - gsttensor_trainer.c, [924](#)
 - gsttensor_transform.c, [958](#)
 - mqttsink.c, [412](#)
 - mqttsrc.c, [451](#)
 - tensor_filter.c, [1403](#)
 - tensor_query_client.c, [1563](#)
 - tensor_query_serversink.c, [1601](#)
 - tensor_query_serversrc.c, [1615](#)
- GST_DATA_REPO_DATA_AUDIO
 - gstdatarepo.h, [297](#)
- GST_DATA_REPO_DATA_IMAGE
 - gstdatarepo.h, [297](#)
- GST_DATA_REPO_DATA_MAX
 - gstdatarepo.h, [297](#)
- GST_DATA_REPO_DATA_OCTET
 - gstdatarepo.h, [297](#)
- GST_DATA_REPO_DATA_TENSOR
 - gstdatarepo.h, [297](#)
- GST_DATA_REPO_DATA_TEXT
 - gstdatarepo.h, [297](#)
- GST_DATA_REPO_DATA_UNKNOWN
 - gstdatarepo.h, [297](#)
- GST_DATA_REPO_DATA_VIDEO
 - gstdatarepo.h, [297](#)
- gst_data_repo_get_caps_by_tensors_sequence
 - gstdatareposrc.c, [324](#)
- gst_data_repo_get_data_type_from_caps

- gstdatarepo.c, 294
- gstdatarepo.h, 297
- GST_DATA_REPO_SINK
 - gstdatareposink.h, 317
- GST_DATA_REPO_SINK_CAST
 - gstdatareposink.h, 317
- gst_data_repo_sink_change_state
 - gstdatareposink.c, 304
- GST_DATA_REPO_SINK_CLASS
 - gstdatareposink.h, 317
- gst_data_repo_sink_class_init
 - gstdatareposink.c, 305
- gst_data_repo_sink_finalize
 - gstdatareposink.c, 306
- gst_data_repo_sink_get_caps
 - gstdatareposink.c, 307
- gst_data_repo_sink_get_image_filename
 - gstdatareposink.c, 307
- gst_data_repo_sink_get_property
 - gstdatareposink.c, 307
- gst_data_repo_sink_get_type
 - gstdatareposink.h, 318
- gst_data_repo_sink_init
 - gstdatareposink.c, 308
- gst_data_repo_sink_open_file
 - gstdatareposink.c, 308
- gst_data_repo_sink_parent_class
 - gstdatareposink.c, 302
- gst_data_repo_sink_query
 - gstdatareposink.c, 309
- gst_data_repo_sink_render
 - gstdatareposink.c, 309
- gst_data_repo_sink_set_caps
 - gstdatareposink.c, 310
- gst_data_repo_sink_set_is_static_tensors
 - gstdatareposink.c, 311
- gst_data_repo_sink_set_property
 - gstdatareposink.c, 312
- gst_data_repo_sink_stop
 - gstdatareposink.c, 312
- gst_data_repo_sink_write_flexible_or_sparse_tensors
 - gstdatareposink.c, 313
- gst_data_repo_sink_write_json_meta_file
 - gstdatareposink.c, 313
- gst_data_repo_sink_write_multi_images
 - gstdatareposink.c, 314
- gst_data_repo_sink_write_others
 - gstdatareposink.c, 315
- GST_DATA_REPO_SRC
 - gstdatareposrc.h, 342
- gst_data_repo_src_change_state
 - gstdatareposrc.c, 325
- GST_DATA_REPO_SRC_CLASS
 - gstdatareposrc.h, 343
- gst_data_repo_src_class_init
 - gstdatareposrc.c, 326
- gst_data_repo_src_create
 - gstdatareposrc.c, 326
- gst_data_repo_src_epoch_is_done
 - gstdatareposrc.c, 327
- gst_data_repo_src_finalize
 - gstdatareposrc.c, 327
- gst_data_repo_src_get_caps
 - gstdatareposrc.c, 328
- gst_data_repo_src_get_file_offset
 - gstdatareposrc.c, 329
- gst_data_repo_src_get_image_filename
 - gstdatareposrc.c, 329
- gst_data_repo_src_get_num_tensors
 - gstdatareposrc.c, 329
- gst_data_repo_src_get_property
 - gstdatareposrc.c, 330
- gst_data_repo_src_get_type
 - gstdatareposrc.h, 344
- gst_data_repo_src_init
 - gstdatareposrc.c, 330
- gst_data_repo_src_parent_class
 - gstdatareposrc.c, 322
- gst_data_repo_src_parse_caps
 - gstdatareposrc.c, 331
- gst_data_repo_src_read_flexible_or_sparse_tensors
 - gstdatareposrc.c, 332
- gst_data_repo_src_read_json_file
 - gstdatareposrc.c, 332
- gst_data_repo_src_read_multi_images
 - gstdatareposrc.c, 333
- gst_data_repo_src_read_others
 - gstdatareposrc.c, 334
- gst_data_repo_src_read_tensors
 - gstdatareposrc.c, 335
- gst_data_repo_src_set_caps
 - gstdatareposrc.c, 336
- gst_data_repo_src_set_file_path
 - gstdatareposrc.c, 336
- gst_data_repo_src_set_property
 - gstdatareposrc.c, 337
- gst_data_repo_src_set_tensors_sequence
 - gstdatareposrc.c, 338
- gst_data_repo_src_set_timestamp
 - gstdatareposrc.c, 338
- gst_data_repo_src_shuffle_samples_index
 - gstdatareposrc.c, 339
- gst_data_repo_src_start
 - gstdatareposrc.c, 339
- gst_data_repo_src_stop
 - gstdatareposrc.c, 340
- GST_DEBUG_CATEGORY_STATIC
 - edge_sink.c, 354
 - edge_src.c, 370
 - gstdatareposink.c, 315
 - gstdatareposrc.c, 340
 - gstjoin.c, 388
 - gsttensor_aggregator.c, 497
 - gsttensor_converter.c, 527
 - gsttensor_crop.c, 572
 - gsttensor_debug.c, 593

- gsttensor_decoder.c, [611](#)
- gsttensor_demux.c, [638](#)
- gsttensor_if.c, [658](#)
- gsttensor_merge.c, [689](#)
- gsttensor_mux.c, [712](#)
- gsttensor_rate.c, [735](#)
- gsttensor_reposink.c, [776](#)
- gsttensor_reposrc.c, [791](#)
- gsttensor_sink.c, [805](#)
- gsttensor_sparsedec.c, [824](#)
- gsttensor_sparseenc.c, [837](#)
- gsttensor_split.c, [858](#)
- gsttensor_srcio.c, [889](#)
- gsttensor_trainer.c, [926](#)
- gsttensor_transform.c, [963](#)
- mqttpsink.c, [419](#)
- mqttpsrc.c, [460](#)
- tensor_filter.c, [1411](#)
- tensor_query_client.c, [1566](#)
- tensor_query_serversink.c, [1603](#)
- tensor_query_serversrc.c, [1618](#)
- GST_EDGE_ELEM_NAME_SINK
 - edge_common.h, [347](#)
- GST_EDGE_ELEM_NAME_SRC
 - edge_common.h, [348](#)
- gst_edge_get_connect_type
 - edge_common.c, [345](#)
 - edge_common.h, [348](#)
- GST_EDGE_PACKAGE
 - edge_common.h, [348](#)
- GST_EDGESINK
 - edge_sink.h, [365](#)
- GST_EDGESINK_CAST
 - edge_sink.h, [365](#)
- GST_EDGESINK_CLASS
 - edge_sink.h, [365](#)
- gst_edgesink_class_init
 - edge_sink.c, [355](#)
- gst_edgesink_finalize
 - edge_sink.c, [355](#)
- gst_edgesink_get_connect_type
 - edge_sink.c, [356](#)
- gst_edgesink_get_host
 - edge_sink.c, [356](#)
- gst_edgesink_get_port
 - edge_sink.c, [357](#)
- gst_edgesink_get_property
 - edge_sink.c, [357](#)
- gst_edgesink_get_type
 - edge_sink.h, [367](#)
- gst_edgesink_init
 - edge_sink.c, [358](#)
- gst_edgesink_parent_class
 - edge_sink.c, [353](#)
- gst_edgesink_render
 - edge_sink.c, [358](#)
- gst_edgesink_set_caps
 - edge_sink.c, [359](#)
- gst_edgesink_set_connect_type
 - edge_sink.c, [360](#)
- gst_edgesink_set_host
 - edge_sink.c, [360](#)
- gst_edgesink_set_port
 - edge_sink.c, [361](#)
- gst_edgesink_set_property
 - edge_sink.c, [361](#)
- gst_edgesink_start
 - edge_sink.c, [362](#)
- gst_edgesink_stop
 - edge_sink.c, [362](#)
- GST_EDGESRC
 - edge_src.h, [380](#)
- GST_EDGESRC_CAST
 - edge_src.h, [381](#)
- gst_edgesrc_change_state
 - edge_src.c, [371](#)
- GST_EDGESRC_CLASS
 - edge_src.h, [381](#)
- gst_edgesrc_class_finalize
 - edge_src.c, [371](#)
- gst_edgesrc_class_init
 - edge_src.c, [372](#)
- gst_edgesrc_create
 - edge_src.c, [372](#)
- gst_edgesrc_get_connect_type
 - edge_src.c, [373](#)
- gst_edgesrc_get_dest_host
 - edge_src.c, [374](#)
- gst_edgesrc_get_dest_port
 - edge_src.c, [374](#)
- gst_edgesrc_get_property
 - edge_src.c, [374](#)
- gst_edgesrc_get_type
 - edge_src.h, [382](#)
- gst_edgesrc_init
 - edge_src.c, [375](#)
- gst_edgesrc_parent_class
 - edge_src.c, [369](#)
- gst_edgesrc_set_connect_type
 - edge_src.c, [375](#)
- gst_edgesrc_set_dest_host
 - edge_src.c, [376](#)
- gst_edgesrc_set_dest_port
 - edge_src.c, [376](#)
- gst_edgesrc_set_property
 - edge_src.c, [377](#)
- gst_edgesrc_start
 - edge_src.c, [377](#)
- gst_edgesrc_stop
 - edge_src.c, [378](#)
- GST_ELEMENT_ERROR_BTRACE
 - nnstreamer_log.h, [1201](#)
- GST_ERROR
 - nnstreamer.c, [1355](#)
- GST_IS_DATA_REPO_SINK
 - gstdatareposink.h, [317](#)

- GST_IS_DATA_REPO_SINK_CLASS
 - gstdatareposink.h, [317](#)
- GST_IS_DATA_REPO_SRC
 - gstdatareposrc.h, [343](#)
- GST_IS_DATA_REPO_SRC_CLASS
 - gstdatareposrc.h, [343](#)
- GST_IS_EDGESINK
 - edge_sink.h, [366](#)
- GST_IS_EDGESINK_CLASS
 - edge_sink.h, [366](#)
- GST_IS_EDGESRC
 - edge_src.h, [381](#)
- GST_IS_EDGESRC_CLASS
 - edge_src.h, [381](#)
- GST_IS_JOIN
 - gstjoin.h, [401](#)
- GST_IS_JOIN_CLASS
 - gstjoin.h, [401](#)
- GST_IS_JOIN_PAD
 - gstjoin.c, [385](#)
- GST_IS_JOIN_PAD_CLASS
 - gstjoin.c, [385](#)
- GST_IS_MQTT_SINK
 - mqttsink.h, [445](#)
- GST_IS_MQTT_SINK_CLASS
 - mqttsink.h, [445](#)
- GST_IS_MQTT_SRC
 - mqttsrc.h, [482](#)
- GST_IS_MQTT_SRC_CLASS
 - mqttsrc.h, [482](#)
- GST_IS_TENSOR_AGGREGATOR
 - gsttensor_aggregator.h, [513](#)
- GST_IS_TENSOR_AGGREGATOR_CLASS
 - gsttensor_aggregator.h, [513](#)
- GST_IS_TENSOR_CONVERTER
 - gsttensor_converter.h, [555](#)
- GST_IS_TENSOR_CONVERTER_CLASS
 - gsttensor_converter.h, [555](#)
- GST_IS_TENSOR_CROP
 - gsttensor_crop.h, [587](#)
- GST_IS_TENSOR_CROP_CLASS
 - gsttensor_crop.h, [587](#)
- GST_IS_TENSOR_DEBUG
 - gsttensor_debug.h, [601](#)
- GST_IS_TENSOR_DEBUG_CLASS
 - gsttensor_debug.h, [601](#)
- GST_IS_TENSOR_DECODER
 - gsttensor_decoder.h, [632](#)
- GST_IS_TENSOR_DECODER_CLASS
 - gsttensor_decoder.h, [632](#)
- GST_IS_TENSOR_DEMUX
 - gsttensor_demux.h, [651](#)
- GST_IS_TENSOR_DEMUX_CLASS
 - gsttensor_demux.h, [651](#)
- GST_IS_TENSOR_FILTER
 - tensor_filter.h, [1433](#)
- GST_IS_TENSOR_FILTER_CLASS
 - tensor_filter.h, [1434](#)
- GST_IS_TENSOR_IF
 - gsttensor_if.h, [681](#)
- GST_IS_TENSOR_IF_CLASS
 - gsttensor_if.h, [681](#)
- GST_IS_TENSOR_MERGE
 - gsttensor_merge.h, [706](#)
- GST_IS_TENSOR_MERGE_CLASS
 - gsttensor_merge.h, [706](#)
- GST_IS_TENSOR_MUX
 - gsttensor_mux.h, [728](#)
- GST_IS_TENSOR_MUX_CLASS
 - gsttensor_mux.h, [728](#)
- GST_IS_TENSOR_QUERY_CLIENT
 - tensor_query_client.h, [1577](#)
- GST_IS_TENSOR_QUERY_CLIENT_CLASS
 - tensor_query_client.h, [1577](#)
- GST_IS_TENSOR_QUERY_SERVERSINK
 - tensor_query_serversink.h, [1610](#)
- GST_IS_TENSOR_QUERY_SERVERSINK_CLASS
 - tensor_query_serversink.h, [1610](#)
- GST_IS_TENSOR_QUERY_SERVERSRC
 - tensor_query_serversrc.h, [1625](#)
- GST_IS_TENSOR_QUERY_SERVERSRC_CLASS
 - tensor_query_serversrc.h, [1625](#)
- GST_IS_TENSOR_RATE
 - gsttensor_rate.h, [750](#)
- GST_IS_TENSOR_RATE_CLASS
 - gsttensor_rate.h, [750](#)
- GST_IS_TENSOR_REPOSINK
 - gsttensor_reposink.h, [786](#)
- GST_IS_TENSOR_REPOSINK_CLASS
 - gsttensor_reposink.h, [786](#)
- GST_IS_TENSOR_REPOSRC
 - gsttensor_reposrc.h, [798](#)
- GST_IS_TENSOR_REPOSRC_CLASS
 - gsttensor_reposrc.h, [798](#)
- GST_IS_TENSOR_SINK
 - gsttensor_sink.h, [819](#)
- GST_IS_TENSOR_SINK_CLASS
 - gsttensor_sink.h, [819](#)
- GST_IS_TENSOR_SPARSE_DEC
 - gsttensor_sparsedec.h, [832](#)
- GST_IS_TENSOR_SPARSE_DEC_CLASS
 - gsttensor_sparsedec.h, [832](#)
- GST_IS_TENSOR_SPARSE_ENC
 - gsttensor_sparseenc.h, [846](#)
- GST_IS_TENSOR_SPARSE_ENC_CLASS
 - gsttensor_sparseenc.h, [846](#)
- GST_IS_TENSOR_SPLIT
 - gsttensor_split.h, [871](#)
- GST_IS_TENSOR_SPLIT_CLASS
 - gsttensor_split.h, [871](#)
- GST_IS_TENSOR_SRC_IIO
 - gsttensor_srciiio.h, [917](#)
- GST_IS_TENSOR_SRC_IIO_CLASS
 - gsttensor_srciiio.h, [917](#)
- GST_IS_TENSOR_TRAINER
 - gsttensor_trainer.h, [952](#)

`GST_IS_TENSOR_TRAINER_CLASS`
 gsttensor_trainer.h, 952

`GST_IS_TENSOR_TRANSFORM`
 gsttensor_transform.h, 990

`GST_IS_TENSOR_TRANSFORM_CLASS`
 gsttensor_transform.h, 990

`GST_JOIN`
 gstjoin.h, 401

`GST_JOIN_CLASS`
 gstjoin.h, 401

`gst_join_class_init`
 gstjoin.c, 389

`gst_join_dispose`
 gstjoin.c, 389

`gst_join_finalize`
 gstjoin.c, 390

`gst_join_get_active_sinkpad`
 gstjoin.c, 390

`GST_JOIN_GET_COND`
 gstjoin.c, 385

`gst_join_get_linked_pad`
 gstjoin.c, 391

`GST_JOIN_GET_LOCK`
 gstjoin.c, 385

`gst_join_get_property`
 gstjoin.c, 391

`gst_join_get_type`
 gstjoin.h, 402

`gst_join_init`
 gstjoin.c, 392

`GST_JOIN_LOCK`
 gstjoin.c, 385

`GST_JOIN_PAD`
 gstjoin.c, 386

`GST_JOIN_PAD_CAST`
 gstjoin.c, 386

`gst_join_pad_chain`
 gstjoin.c, 392

`GST_JOIN_PAD_CLASS`
 gstjoin.c, 386

`gst_join_pad_class_init`
 gstjoin.c, 393

`gst_join_pad_event`
 gstjoin.c, 393

`gst_join_pad_finalize`
 gstjoin.c, 394

`gst_join_pad_get_type`
 gstjoin.c, 394

`gst_join_pad_init`
 gstjoin.c, 395

`gst_join_pad_iterate_linked_pads`
 gstjoin.c, 395

`gst_join_pad_query`
 gstjoin.c, 396

`gst_join_pad_reset`
 gstjoin.c, 396

`gst_join_parent_class`
 gstjoin.c, 386

`gst_join_request_new_pad`
 gstjoin.c, 397

`gst_join_set_active_pad`
 gstjoin.c, 398

`gst_join_sink_factory`
 gstjoin.c, 399

`gst_join_src_factory`
 gstjoin.c, 399

`GST_JOIN_UNLOCK`
 gstjoin.c, 386

`GST_JOIN_WAIT`
 gstjoin.c, 386

`gst_memory_map_is_extra_tensor`
 nntstreamer_plugin_api_impl.c, 1215

`gst_meta_query_api_get_type`
 tensor_meta.c, 1555
 tensor_meta.h, 1559

`GST_META_QUERY_API_TYPE`
 tensor_meta.h, 1559

`gst_meta_query_free`
 tensor_meta.c, 1555

`gst_meta_query_get_info`
 tensor_meta.c, 1555
 tensor_meta.h, 1559

`GST_META_QUERY_INFO`
 tensor_meta.h, 1559

`gst_meta_query_init`
 tensor_meta.c, 1556

`gst_meta_query_transform`
 tensor_meta.c, 1556

`GST_MQTT_ELEM_NAME_SINK`
 mqttcommon.h, 404

`GST_MQTT_ELEM_NAME_SRC`
 mqttcommon.h, 405

`GST_MQTT_LEN_MSG_HDR`
 mqttcommon.h, 405

`GST_MQTT_MAX_LEN_GST_CAPS_STR`
 mqttcommon.h, 405

`GST_MQTT_MAX_NUM_MEMS`
 mqttcommon.h, 405

`GST_MQTT_PACKAGE`
 mqttcommon.h, 405

`GST_MQTT_SINK`
 mqttsink.h, 445

`GST_MQTT_SINK_CAST`
 mqttsink.h, 446

`gst_mqtt_sink_change_state`
 mqttsink.c, 419

`GST_MQTT_SINK_CLASS`
 mqttsink.h, 446

`gst_mqtt_sink_class_finalize`
 mqttsink.c, 420

`gst_mqtt_sink_class_init`
 mqttsink.c, 421

`gst_mqtt_sink_event`
 mqttsink.c, 422

`gst_mqtt_sink_get_client_id`
 mqttsink.c, 422

`gst_mqtt_sink_get_debug`
mqttpsink.c, [422](#)

`gst_mqtt_sink_get_host_address`
mqttpsink.c, [423](#)

`gst_mqtt_sink_get_host_port`
mqttpsink.c, [423](#)

`gst_mqtt_sink_get_max_msg_buf_size`
mqttpsink.c, [423](#)

`gst_mqtt_sink_get_mqtt_ntp_srvs`
mqttpsink.c, [424](#)

`gst_mqtt_sink_get_mqtt_ntp_sync`
mqttpsink.c, [424](#)

`gst_mqtt_sink_get_mqtt_qos`
mqttpsink.c, [424](#)

`gst_mqtt_sink_get_num_buffers`
mqttpsink.c, [425](#)

`gst_mqtt_sink_get_opt_cleansession`
mqttpsink.c, [425](#)

`gst_mqtt_sink_get_opt_keep_alive_interval`
mqttpsink.c, [425](#)

`gst_mqtt_sink_get_property`
mqttpsink.c, [426](#)

`gst_mqtt_sink_get_pub_topic`
mqttpsink.c, [428](#)

`gst_mqtt_sink_get_pub_wait_timeout`
mqttpsink.c, [428](#)

`gst_mqtt_sink_get_type`
mqttpsink.h, [447](#)

`gst_mqtt_sink_init`
mqttpsink.c, [428](#)

`gst_mqtt_sink_parent_class`
mqttpsink.c, [412](#)

`gst_mqtt_sink_query`
mqttpsink.c, [429](#)

`gst_mqtt_sink_render`
mqttpsink.c, [430](#)

`gst_mqtt_sink_render_list`
mqttpsink.c, [430](#)

`gst_mqtt_sink_set_caps`
mqttpsink.c, [431](#)

`gst_mqtt_sink_set_client_id`
mqttpsink.c, [432](#)

`gst_mqtt_sink_set_debug`
mqttpsink.c, [432](#)

`gst_mqtt_sink_set_host_address`
mqttpsink.c, [433](#)

`gst_mqtt_sink_set_host_port`
mqttpsink.c, [433](#)

`gst_mqtt_sink_set_max_msg_buf_size`
mqttpsink.c, [434](#)

`gst_mqtt_sink_set_mqtt_ntp_srvs`
mqttpsink.c, [434](#)

`gst_mqtt_sink_set_mqtt_ntp_sync`
mqttpsink.c, [435](#)

`gst_mqtt_sink_set_mqtt_qos`
mqttpsink.c, [435](#)

`gst_mqtt_sink_set_num_buffers`
mqttpsink.c, [436](#)

`gst_mqtt_sink_set_opt_cleansession`
mqttpsink.c, [436](#)

`gst_mqtt_sink_set_opt_keep_alive_interval`
mqttpsink.c, [436](#)

`gst_mqtt_sink_set_property`
mqttpsink.c, [437](#)

`gst_mqtt_sink_set_pub_topic`
mqttpsink.c, [438](#)

`gst_mqtt_sink_set_pub_wait_timeout`
mqttpsink.c, [439](#)

`gst_mqtt_sink_start`
mqttpsink.c, [439](#)

`gst_mqtt_sink_stop`
mqttpsink.c, [440](#)

`GST_MQTT_SRC`
mqttpsrc.h, [482](#)

`GST_MQTT_SRC_CAST`
mqttpsrc.h, [482](#)

`gst_mqtt_src_change_state`
mqttpsrc.c, [460](#)

`GST_MQTT_SRC_CLASS`
mqttpsrc.h, [483](#)

`gst_mqtt_src_class_finalize`
mqttpsrc.c, [460](#)

`gst_mqtt_src_class_init`
mqttpsrc.c, [461](#)

`gst_mqtt_src_create`
mqttpsrc.c, [462](#)

`gst_mqtt_src_get_caps`
mqttpsrc.c, [463](#)

`gst_mqtt_src_get_client_id`
mqttpsrc.c, [464](#)

`gst_mqtt_src_get_debug`
mqttpsrc.c, [464](#)

`gst_mqtt_src_get_host_address`
mqttpsrc.c, [464](#)

`gst_mqtt_src_get_host_port`
mqttpsrc.c, [465](#)

`gst_mqtt_src_get_is_live`
mqttpsrc.c, [465](#)

`gst_mqtt_src_get_mqtt_qos`
mqttpsrc.c, [465](#)

`gst_mqtt_src_get_opt_cleansession`
mqttpsrc.c, [466](#)

`gst_mqtt_src_get_opt_keep_alive_interval`
mqttpsrc.c, [466](#)

`gst_mqtt_src_get_property`
mqttpsrc.c, [466](#)

`gst_mqtt_src_get_sub_timeout`
mqttpsrc.c, [468](#)

`gst_mqtt_src_get_sub_topic`
mqttpsrc.c, [468](#)

`gst_mqtt_src_get_times`
mqttpsrc.c, [468](#)

`gst_mqtt_src_get_type`
mqttpsrc.h, [483](#)

`gst_mqtt_src_init`
mqttpsrc.c, [469](#)

- gst_mqtt_src_is_seekable
mqttsrc.c, [469](#)
- gst_mqtt_src_parent_class
mqttsrc.c, [451](#)
- gst_mqtt_src_query
mqttsrc.c, [470](#)
- gst_mqtt_src_renegotiate
mqttsrc.c, [470](#)
- gst_mqtt_src_set_client_id
mqttsrc.c, [471](#)
- gst_mqtt_src_set_debug
mqttsrc.c, [471](#)
- gst_mqtt_src_set_host_address
mqttsrc.c, [472](#)
- gst_mqtt_src_set_host_port
mqttsrc.c, [472](#)
- gst_mqtt_src_set_is_live
mqttsrc.c, [473](#)
- gst_mqtt_src_set_mqtt_qos
mqttsrc.c, [473](#)
- gst_mqtt_src_set_opt_cleansession
mqttsrc.c, [474](#)
- gst_mqtt_src_set_opt_keep_alive_interval
mqttsrc.c, [474](#)
- gst_mqtt_src_set_property
mqttsrc.c, [475](#)
- gst_mqtt_src_set_sub_timeout
mqttsrc.c, [476](#)
- gst_mqtt_src_set_sub_topic
mqttsrc.c, [477](#)
- gst_mqtt_src_start
mqttsrc.c, [477](#)
- gst_mqtt_src_stop
mqttsrc.c, [478](#)
- GST_PLUGIN_DEFINE
nnstreamer.c, [1355](#)
- GST_REPO_BROADCAST
gsttensor_repo.c, [754](#)
- GST_REPO_LOCK
gsttensor_repo.c, [754](#)
- GST_REPO_UNLOCK
gsttensor_repo.c, [755](#)
- GST_REPO_WAIT
gsttensor_repo.c, [755](#)
- gst_structure_get_media_type
nnstreamer_plugin_api.h, [1000](#)
nnstreamer_plugin_api_impl.c, [1216](#)
- gst_structure_is_tensor_stream
nnstreamer_plugin_api.h, [1000](#)
nnstreamer_plugin_api_impl.c, [1217](#)
- gst_tensor_aggregation_add_data
nnstreamer_plugin_api_impl.c, [1218](#)
- gst_tensor_aggregation_clear
nnstreamer_plugin_api_impl.c, [1218](#)
tensor_common.h, [1366](#)
- gst_tensor_aggregation_clear_all
nnstreamer_plugin_api_impl.c, [1219](#)
tensor_common.h, [1367](#)
- gst_tensor_aggregation_clear_internal
nnstreamer_plugin_api_impl.c, [1220](#)
- gst_tensor_aggregation_data_s, [251](#)
adapter, [251](#)
- gst_tensor_aggregation_free_data
nnstreamer_plugin_api_impl.c, [1220](#)
- gst_tensor_aggregation_get_adapter
nnstreamer_plugin_api_impl.c, [1221](#)
tensor_common.h, [1368](#)
- gst_tensor_aggregation_get_data
nnstreamer_plugin_api_impl.c, [1222](#)
- gst_tensor_aggregation_init
nnstreamer_plugin_api_impl.c, [1222](#)
tensor_common.h, [1369](#)
- GST_TENSOR_AGGREGATOR
gsttensor_aggregator.h, [514](#)
- gst_tensor_aggregator_chain
gsttensor_aggregator.c, [497](#)
- gst_tensor_aggregator_change_state
gsttensor_aggregator.c, [498](#)
- gst_tensor_aggregator_check_concat_axis
gsttensor_aggregator.c, [498](#)
- GST_TENSOR_AGGREGATOR_CLASS
gsttensor_aggregator.h, [514](#)
- gst_tensor_aggregator_class_init
gsttensor_aggregator.c, [499](#)
- gst_tensor_aggregator_concat
gsttensor_aggregator.c, [500](#)
- gst_tensor_aggregator_finalize
gsttensor_aggregator.c, [502](#)
- gst_tensor_aggregator_get_adapter
gsttensor_aggregator.c, [503](#)
- gst_tensor_aggregator_get_property
gsttensor_aggregator.c, [504](#)
- gst_tensor_aggregator_get_type
gsttensor_aggregator.h, [515](#)
- gst_tensor_aggregator_init
gsttensor_aggregator.c, [504](#)
- gst_tensor_aggregator_parent_class
gsttensor_aggregator.c, [495](#)
- gst_tensor_aggregator_parse_caps
gsttensor_aggregator.c, [505](#)
- gst_tensor_aggregator_push
gsttensor_aggregator.c, [506](#)
- gst_tensor_aggregator_query_caps
gsttensor_aggregator.c, [506](#)
- gst_tensor_aggregator_reset
gsttensor_aggregator.c, [507](#)
- gst_tensor_aggregator_set_property
gsttensor_aggregator.c, [508](#)
- gst_tensor_aggregator_sink_event
gsttensor_aggregator.c, [508](#)
- gst_tensor_aggregator_sink_query
gsttensor_aggregator.c, [509](#)
- gst_tensor_aggregator_src_query
gsttensor_aggregator.c, [510](#)
- gst_tensor_alloc_init
nnstreamer_plugin_api.h, [1001](#)

- tensor_allocator.c, [1359](#)
- GST_TENSOR_ALLOCATOR
 - tensor_allocator.c, [1358](#)
- gst_tensor_allocator_alignment
 - tensor_allocator.c, [1361](#)
- gst_tensor_allocator_class_init
 - tensor_allocator.c, [1360](#)
- gst_tensor_allocator_get_type
 - tensor_allocator.c, [1360](#)
- gst_tensor_allocator_init
 - tensor_allocator.c, [1360](#)
- gst_tensor_buffer_append_memory
 - nnstreamer_plugin_api.h, [1002](#)
 - nnstreamer_plugin_api_impl.c, [1223](#)
- gst_tensor_buffer_from_config
 - nnstreamer_plugin_api_impl.c, [1225](#)
 - tensor_common.h, [1370](#)
- gst_tensor_buffer_get_count
 - nnstreamer_plugin_api.h, [1003](#)
 - nnstreamer_plugin_api_impl.c, [1226](#)
- gst_tensor_buffer_get_nth_memory
 - nnstreamer_plugin_api.h, [1004](#)
 - nnstreamer_plugin_api_impl.c, [1227](#)
- GST_TENSOR_CAP_DEFAULT
 - tensor_typedef.h, [1149](#)
- gst_tensor_caps_can_intersect
 - nnstreamer_plugin_api.h, [1006](#)
 - nnstreamer_plugin_api_impl.c, [1228](#)
- gst_tensor_caps_from_config
 - nnstreamer_plugin_api.h, [1006](#)
 - nnstreamer_plugin_api_impl.c, [1229](#)
- gst_tensor_caps_update_dimension
 - nnstreamer_plugin_api.h, [1007](#)
 - nnstreamer_plugin_api_impl.c, [1230](#)
- gst_tensor_channel_list_filter_enabled
 - gsttensor_srcio.c, [889](#)
- gst_tensor_channel_list_sort_cmp
 - gsttensor_srcio.c, [889](#)
- GST_TENSOR_CONVERTER
 - gsttensor_converter.h, [556](#)
- gst_tensor_converter_chain
 - gsttensor_converter.c, [527](#)
- gst_tensor_converter_change_state
 - gsttensor_converter.c, [528](#)
- GST_TENSOR_CONVERTER_CLASS
 - gsttensor_converter.h, [556](#)
- gst_tensor_converter_class_init
 - gsttensor_converter.c, [529](#)
- gst_tensor_converter_finalize
 - gsttensor_converter.c, [530](#)
- gst_tensor_converter_get_adapter
 - gsttensor_converter.c, [531](#)
- gst_tensor_converter_get_format_list
 - gsttensor_converter.c, [532](#)
- gst_tensor_converter_get_possible_media_caps
 - gsttensor_converter.c, [532](#)
- gst_tensor_converter_get_property
 - gsttensor_converter.c, [533](#)
- gst_tensor_converter_get_type
 - gsttensor_converter.h, [557](#)
- gst_tensor_converter_init
 - gsttensor_converter.c, [534](#)
- gst_tensor_converter_parent_class
 - gsttensor_converter.c, [520](#)
- gst_tensor_converter_parse_audio
 - gsttensor_converter.c, [535](#)
- gst_tensor_converter_parse_caps
 - gsttensor_converter.c, [536](#)
- gst_tensor_converter_parse_custom
 - gsttensor_converter.c, [537](#)
- gst_tensor_converter_parse_octet
 - gsttensor_converter.c, [538](#)
- gst_tensor_converter_parse_tensor
 - gsttensor_converter.c, [539](#)
- gst_tensor_converter_parse_text
 - gsttensor_converter.c, [540](#)
- gst_tensor_converter_parse_video
 - gsttensor_converter.c, [541](#)
- gst_tensor_converter_query_caps
 - gsttensor_converter.c, [542](#)
- gst_tensor_converter_reset
 - gsttensor_converter.c, [543](#)
- gst_tensor_converter_set_property
 - gsttensor_converter.c, [544](#)
- gst_tensor_converter_sink_event
 - gsttensor_converter.c, [545](#)
- gst_tensor_converter_sink_query
 - gsttensor_converter.c, [546](#)
- gst_tensor_converter_src_query
 - gsttensor_converter.c, [547](#)
- gst_tensor_converter_update_caps
 - gsttensor_converter.c, [548](#)
- gst_tensor_converter_video_stride
 - gsttensor_converter.c, [549](#)
- GST_TENSOR_CROP
 - gsttensor_crop.h, [587](#)
- gst_tensor_crop_chain
 - gsttensor_crop.c, [572](#)
- gst_tensor_crop_change_state
 - gsttensor_crop.c, [573](#)
- GST_TENSOR_CROP_CLASS
 - gsttensor_crop.h, [587](#)
- gst_tensor_crop_class_init
 - gsttensor_crop.c, [574](#)
- gst_tensor_crop_collected
 - gsttensor_crop.c, [574](#)
- gst_tensor_crop_do_cropping
 - gsttensor_crop.c, [575](#)
- gst_tensor_crop_finalize
 - gsttensor_crop.c, [576](#)
- gst_tensor_crop_get_crop_info
 - gsttensor_crop.c, [577](#)
- gst_tensor_crop_get_property
 - gsttensor_crop.c, [578](#)
- gst_tensor_crop_get_type
 - gsttensor_crop.h, [588](#)

[gst_tensor_crop_init](#)
[gsttensor_crop.c](#), 578
[gst_tensor_crop_negotiate](#)
[gsttensor_crop.c](#), 579
[gst_tensor_crop_pad_reset](#)
[gsttensor_crop.c](#), 580
[gst_tensor_crop_parent_class](#)
[gsttensor_crop.c](#), 571
[gst_tensor_crop_prepare_out_meta](#)
[gsttensor_crop.c](#), 580
[gst_tensor_crop_reset](#)
[gsttensor_crop.c](#), 581
[gst_tensor_crop_set_property](#)
[gsttensor_crop.c](#), 582
[gst_tensor_crop_sink_event](#)
[gsttensor_crop.c](#), 582
[gst_tensor_crop_src_event](#)
[gsttensor_crop.c](#), 583
[gst_tensor_data_get](#)
[tensor_data.c](#), 1383
[tensor_data.h](#), 1392
[gst_tensor_data_raw_average](#)
[tensor_data.c](#), 1384
[tensor_data.h](#), 1392
[gst_tensor_data_raw_average_per_channel](#)
[tensor_data.c](#), 1385
[tensor_data.h](#), 1393
[gst_tensor_data_raw_std](#)
[tensor_data.c](#), 1386
[tensor_data.h](#), 1394
[gst_tensor_data_raw_std_per_channel](#)
[tensor_data.c](#), 1387
[tensor_data.h](#), 1395
[gst_tensor_data_raw_typecast](#)
[tensor_data.c](#), 1388
[tensor_data.h](#), 1396
[gst_tensor_data_set](#)
[tensor_data.h](#), 1397
[gst_tensor_data_typecast](#)
[tensor_data.c](#), 1389
[tensor_data.h](#), 1398
[GST_TENSOR_DEBUG](#)
[gsttensor_debug.h](#), 601
[GST_TENSOR_DEBUG_CAST](#)
[gsttensor_debug.h](#), 601
[GST_TENSOR_DEBUG_CLASS](#)
[gsttensor_debug.h](#), 602
[gst_tensor_debug_class_init](#)
[gsttensor_debug.c](#), 594
[gst_tensor_debug_finalize](#)
[gsttensor_debug.c](#), 594
[gst_tensor_debug_fixate_caps](#)
[gsttensor_debug.c](#), 595
[gst_tensor_debug_get_property](#)
[gsttensor_debug.c](#), 595
[gst_tensor_debug_get_type](#)
[gsttensor_debug.h](#), 604
[gst_tensor_debug_init](#)
[gsttensor_debug.c](#), 596
[gst_tensor_debug_parent_class](#)
[gsttensor_debug.c](#), 592
[gst_tensor_debug_set_caps](#)
[gsttensor_debug.c](#), 596
[gst_tensor_debug_set_property](#)
[gsttensor_debug.c](#), 596
[gst_tensor_debug_transform_ip](#)
[gsttensor_debug.c](#), 597
[GST_TENSOR_DECODER](#)
[gsttensor_decoder.h](#), 632
[GST_TENSOR_DECODER_CAST](#)
[gsttensor_decoder.h](#), 633
[GST_TENSOR_DECODER_CLASS](#)
[gsttensor_decoder.h](#), 633
[gst_tensor_decoder_clean_plugin](#)
[gsttensor_decoder.c](#), 608
[GST_TENSOR_DEMUX](#)
[gsttensor_demux.h](#), 651
[GST_TENSOR_DEMUX_CAST](#)
[gsttensor_demux.h](#), 651
[gst_tensor_demux_chain](#)
[gsttensor_demux.c](#), 638
[gst_tensor_demux_change_state](#)
[gsttensor_demux.c](#), 639
[GST_TENSOR_DEMUX_CLASS](#)
[gsttensor_demux.h](#), 651
[gst_tensor_demux_class_init](#)
[gsttensor_demux.c](#), 640
[gst_tensor_demux_combine_flows](#)
[gsttensor_demux.c](#), 640
[gst_tensor_demux_dispose](#)
[gsttensor_demux.c](#), 641
[gst_tensor_demux_event](#)
[gsttensor_demux.c](#), 642
[GST_TENSOR_DEMUX_GET_CLASS](#)
[gsttensor_demux.h](#), 651
[gst_tensor_demux_get_property](#)
[gsttensor_demux.c](#), 642
[gst_tensor_demux_get_tensor_config](#)
[gsttensor_demux.c](#), 643
[gst_tensor_demux_get_tensor_pad](#)
[gsttensor_demux.c](#), 644
[gst_tensor_demux_get_type](#)
[gsttensor_demux.h](#), 652
[gst_tensor_demux_init](#)
[gsttensor_demux.c](#), 645
[gst_tensor_demux_parent_class](#)
[gsttensor_demux.c](#), 637
[gst_tensor_demux_parse_caps](#)
[gsttensor_demux.c](#), 646
[gst_tensor_demux_remove_src_pads](#)
[gsttensor_demux.c](#), 647
[gst_tensor_demux_set_property](#)
[gsttensor_demux.c](#), 648
[gst_tensor_dimension_get_min_rank](#)
[nnstreamer_plugin_api_util.h](#), 1055
[nnstreamer_plugin_api_util_impl.c](#), 1257

- gst_tensor_dimension_get_rank
 - nnstreamer_plugin_api_util.h, [1056](#)
 - nnstreamer_plugin_api_util_impl.c, [1258](#)
- gst_tensor_dimension_is_equal
 - nnstreamer_plugin_api_util.h, [1057](#)
 - nnstreamer_plugin_api_util_impl.c, [1259](#)
- gst_tensor_dimension_is_valid
 - nnstreamer_plugin_api_util.h, [1058](#)
 - nnstreamer_plugin_api_util_impl.c, [1260](#)
- gst_tensor_dimension_string_is_equal
 - nnstreamer_plugin_api_util.h, [1060](#)
 - nnstreamer_plugin_api_util_impl.c, [1262](#)
- gst_tensor_extra_info_init
 - nnstreamer_plugin_api_impl.c, [1231](#)
- GST_TENSOR_FILTER
 - tensor_filter.h, [1434](#)
- gst_tensor_filter_allocate_in_invoke
 - tensor_filter_common.c, [1457](#)
 - tensor_filter_common.h, [1502](#)
- GST_TENSOR_FILTER_API_VERSION_DEFINED
 - nnstreamer_plugin_api_filter.h, [1031](#)
- GST_TENSOR_FILTER_API_VERSION_MAX
 - nnstreamer_plugin_api_filter.h, [1032](#)
- GST_TENSOR_FILTER_API_VERSION_MIN
 - nnstreamer_plugin_api_filter.h, [1032](#)
- GST_TENSOR_FILTER_CAST
 - tensor_filter.h, [1434](#)
- gst_tensor_filter_check_hw_availability
 - nnstreamer_internal.h, [1192](#)
 - tensor_filter_common.c, [1458](#)
 - tensor_filter_common.h, [1503](#)
- gst_tensor_filter_check_throttling_delay
 - tensor_filter.c, [1411](#)
- GST_TENSOR_FILTER_CLASS
 - tensor_filter.h, [1434](#)
- gst_tensor_filter_class_init
 - tensor_filter.c, [1411](#)
- gst_tensor_filter_common_close_fw
 - tensor_filter_common.c, [1458](#)
 - tensor_filter_common.h, [1504](#)
- gst_tensor_filter_common_free_property
 - tensor_filter_common.c, [1459](#)
 - tensor_filter_common.h, [1504](#)
- gst_tensor_filter_common_get_combined_in_info
 - tensor_filter_common.c, [1460](#)
 - tensor_filter_common.h, [1505](#)
- gst_tensor_filter_common_get_combined_out_info
 - tensor_filter_common.c, [1460](#)
 - tensor_filter_common.h, [1506](#)
- gst_tensor_filter_common_get_out_info
 - tensor_filter_common.c, [1461](#)
 - tensor_filter_common.h, [1507](#)
- gst_tensor_filter_common_get_property
 - tensor_filter_common.c, [1462](#)
 - tensor_filter_common.h, [1507](#)
- gst_tensor_filter_common_init_property
 - tensor_filter_common.c, [1463](#)
 - tensor_filter_common.h, [1509](#)
- gst_tensor_filter_common_open_fw
 - tensor_filter_common.c, [1464](#)
 - tensor_filter_common.h, [1509](#)
- gst_tensor_filter_common_set_property
 - tensor_filter_common.c, [1464](#)
 - tensor_filter_common.h, [1510](#)
- gst_tensor_filter_common_unload_fw
 - tensor_filter_common.c, [1465](#)
 - tensor_filter_common.h, [1511](#)
- gst_tensor_filter_configure_tensor
 - tensor_filter.c, [1412](#)
- gst_tensor_filter_destroy_notify
 - tensor_filter.c, [1414](#)
- gst_tensor_filter_destroy_notify_util
 - tensor_filter_common.c, [1466](#)
 - tensor_filter_common.h, [1511](#)
- gst_tensor_filter_detect_framework
 - nnstreamer_internal.h, [1193](#)
 - tensor_filter_common.c, [1467](#)
 - tensor_filter_common.h, [1512](#)
- gst_tensor_filter_finalize
 - tensor_filter.c, [1415](#)
- gst_tensor_filter_fixate_caps
 - tensor_filter.c, [1415](#)
- GST_TENSOR_FILTER_FRAMEWORK_BASE
 - nnstreamer_plugin_api_filter.h, [1032](#)
- gst_tensor_filter_framework_info_init
 - tensor_filter_common.c, [1468](#)
- GST_TENSOR_FILTER_FRAMEWORK_V0
 - nnstreamer_plugin_api_filter.h, [1032](#)
- GST_TENSOR_FILTER_FRAMEWORK_V1
 - nnstreamer_plugin_api_filter.h, [1032](#)
- gst_tensor_filter_get_available_framework
 - tensor_filter_common.c, [1468](#)
- gst_tensor_filter_get_dimension_string
 - tensor_filter_common.c, [1469](#)
- gst_tensor_filter_get_layout_string
 - tensor_filter_common.c, [1470](#)
- gst_tensor_filter_get_name_string
 - tensor_filter_common.c, [1471](#)
- gst_tensor_filter_get_property
 - tensor_filter.c, [1416](#)
- gst_tensor_filter_get_rank_string
 - tensor_filter_common.c, [1472](#)
- gst_tensor_filter_get_tensor_size
 - tensor_filter.c, [1417](#)
- gst_tensor_filter_get_type
 - tensor_filter.h, [1435](#)
- gst_tensor_filter_get_type_string
 - tensor_filter_common.c, [1472](#)
- gst_tensor_filter_get_wrapped_mem
 - tensor_filter.c, [1418](#)
- gst_tensor_filter_init
 - tensor_filter.c, [1419](#)
- gst_tensor_filter_install_properties
 - tensor_filter_common.c, [1473](#)
 - tensor_filter_common.h, [1513](#)
- gst_tensor_filter_load_tensor_info

- tensor_filter_common.c, [1474](#)
- tensor_filter_common.h, [1514](#)
- gst_tensor_filter_parent_class
 - tensor_filter.c, [1403](#)
- gst_tensor_filter_parse_accelerator
 - tensor_filter_common.c, [1475](#)
- gst_tensor_filter_parse_modelpaths_string
 - tensor_filter_common.c, [1476](#)
- gst_tensor_filter_properties_init
 - tensor_filter_common.c, [1477](#)
- gst_tensor_filter_property_to_string
 - tensor_filter_common.c, [1477](#)
- gst_tensor_filter_query
 - tensor_filter.c, [1419](#)
- gst_tensor_filter_set_caps
 - tensor_filter.c, [1420](#)
- gst_tensor_filter_set_property
 - tensor_filter.c, [1421](#)
- gst_tensor_filter_sink_event
 - tensor_filter.c, [1422](#)
- gst_tensor_filter_src_event
 - tensor_filter.c, [1423](#)
- gst_tensor_filter_start
 - tensor_filter.c, [1423](#)
- gst_tensor_filter_statistics_init
 - tensor_filter_common.c, [1478](#)
- gst_tensor_filter_stop
 - tensor_filter.c, [1424](#)
- gst_tensor_filter_transform
 - tensor_filter.c, [1425](#)
- gst_tensor_filter_transform_caps
 - tensor_filter.c, [1426](#)
- gst_tensor_filter_transform_size
 - tensor_filter.c, [1428](#)
- gst_tensor_filter_v0_call
 - tensor_filter_common.h, [1499](#)
- gst_tensor_filter_v1_call
 - tensor_filter_common.h, [1499](#)
- gst_tensor_filter_watchdog_trigger
 - tensor_filter.c, [1428](#)
- GST_TENSOR_FORMAT_ALL
 - tensor_typedef.h, [1149](#)
- gst_tensor_get_dimension_string
 - nnstreamer_plugin_api_util.h, [1061](#)
 - nnstreamer_plugin_api_util_impl.c, [1263](#)
- gst_tensor_get_element_count
 - nnstreamer_plugin_api_util.h, [1063](#)
 - nnstreamer_plugin_api_util_impl.c, [1265](#)
- gst_tensor_get_element_size
 - nnstreamer_plugin_api_util.h, [1064](#)
 - nnstreamer_plugin_api_util_impl.c, [1266](#)
- gst_tensor_get_format
 - nnstreamer_plugin_api_util.h, [1065](#)
 - nnstreamer_plugin_api_util_impl.c, [1267](#)
- gst_tensor_get_format_string
 - nnstreamer_plugin_api_util.h, [1067](#)
 - nnstreamer_plugin_api_util_impl.c, [1269](#)
- gst_tensor_get_layout_string
 - tensor_filter_common.c, [1478](#)
- gst_tensor_get_rank_dimension_string
 - nnstreamer_plugin_api_util.h, [1068](#)
 - nnstreamer_plugin_api_util_impl.c, [1270](#)
- gst_tensor_get_size_from_channels
 - gsttensor_srcio.c, [889](#)
- gst_tensor_get_type
 - nnstreamer_plugin_api_util.h, [1069](#)
 - nnstreamer_plugin_api_util_impl.c, [1271](#)
- gst_tensor_get_type_string
 - nnstreamer_plugin_api_util.h, [1070](#)
 - nnstreamer_plugin_api_util_impl.c, [1272](#)
- GST_TENSOR_IF
 - gsttensor_if.h, [681](#)
- gst_tensor_if_act_get_type
 - gsttensor_if.c, [658](#)
- gst_tensor_if_calculate_cv
 - gsttensor_if.c, [658](#)
- GST_TENSOR_IF_CAST
 - gsttensor_if.h, [682](#)
- gst_tensor_if_chain
 - gsttensor_if.c, [659](#)
- gst_tensor_if_check_condition
 - gsttensor_if.c, [660](#)
- GST_TENSOR_IF_CLASS
 - gsttensor_if.h, [682](#)
- gst_tensor_if_class_init
 - gsttensor_if.c, [661](#)
- gst_tensor_if_combine_flows
 - gsttensor_if.c, [662](#)
- gst_tensor_if_configure_custom_prop
 - gsttensor_if.c, [663](#)
- gst_tensor_if_cv_get_type
 - gsttensor_if.c, [663](#)
- gst_tensor_if_dispose
 - gsttensor_if.c, [663](#)
- gst_tensor_if_event
 - gsttensor_if.c, [664](#)
- GST_TENSOR_IF_GET_CLASS
 - gsttensor_if.h, [682](#)
- gst_tensor_if_get_property
 - gsttensor_if.c, [665](#)
- gst_tensor_if_get_property_supplied_value
 - gsttensor_if.c, [666](#)
- gst_tensor_if_get_tensor_average
 - gsttensor_if.c, [666](#)
- gst_tensor_if_get_tensor_pad
 - gsttensor_if.c, [667](#)
- gst_tensor_if_get_type
 - gsttensor_if.h, [685](#)
- gst_tensor_if_init
 - gsttensor_if.c, [668](#)
- gst_tensor_if_install_properties
 - gsttensor_if.c, [669](#)
- gst_tensor_if_op_get_type
 - gsttensor_if.c, [670](#)
- gst_tensor_if_parent_class
 - gsttensor_if.c, [656](#)

gst_tensor_if_parse_caps
gsttensor_if.c, 670

gst_tensor_if_property_to_string
gsttensor_if.c, 671

gst_tensor_if_remove_src_pads
gsttensor_if.c, 672

gst_tensor_if_set_property
gsttensor_if.c, 672

gst_tensor_if_set_property_cv_option
gsttensor_if.c, 673

gst_tensor_if_set_property_glist
gsttensor_if.c, 674

gst_tensor_if_set_property_supplied_value
gsttensor_if.c, 674

gst_tensor_info_convert_to_meta
nnstreamer_plugin_api_util.h, 1071
nnstreamer_plugin_api_util_impl.c, 1273

gst_tensor_info_copy
nnstreamer_plugin_api_util.h, 1072
nnstreamer_plugin_api_util_impl.c, 1274

gst_tensor_info_copy_n
nnstreamer_plugin_api_util.h, 1074
nnstreamer_plugin_api_util_impl.c, 1276

gst_tensor_info_free
nnstreamer_plugin_api_util.h, 1075
nnstreamer_plugin_api_util_impl.c, 1277

gst_tensor_info_get_rank
nnstreamer_plugin_api_util.h, 1077
nnstreamer_plugin_api_util_impl.c, 1279

gst_tensor_info_get_size
nnstreamer_plugin_api_util.h, 1078
nnstreamer_plugin_api_util_impl.c, 1280

gst_tensor_info_init
nnstreamer_plugin_api_util.h, 1079
nnstreamer_plugin_api_util_impl.c, 1281

gst_tensor_info_is_equal
nnstreamer_plugin_api_util.h, 1080
nnstreamer_plugin_api_util_impl.c, 1282

gst_tensor_info_validate
nnstreamer_plugin_api_util.h, 1081
nnstreamer_plugin_api_util_impl.c, 1283

GST_TENSOR_MERGE
gsttensor_merge.h, 706

GST_TENSOR_MERGE_CAST
gsttensor_merge.h, 706

gst_tensor_merge_change_state
gsttensor_merge.c, 689

GST_TENSOR_MERGE_CLASS
gsttensor_merge.h, 706

gst_tensor_merge_class_init
gsttensor_merge.c, 690

gst_tensor_merge_collect_buffer
gsttensor_merge.c, 690

gst_tensor_merge_collected
gsttensor_merge.c, 691

gst_tensor_merge_finalize
gsttensor_merge.c, 692

gst_tensor_merge_generate_mem
gsttensor_merge.c, 693

GST_TENSOR_MERGE_GET_CLASS
gsttensor_merge.h, 706

gst_tensor_merge_get_merged_config
gsttensor_merge.c, 694

gst_tensor_merge_get_mode
gsttensor_merge.c, 695

gst_tensor_merge_get_property
gsttensor_merge.c, 695

gst_tensor_merge_get_type
gsttensor_merge.h, 708

gst_tensor_merge_init
gsttensor_merge.c, 696

gst_tensor_merge_linear_string
gsttensor_merge.c, 702

gst_tensor_merge_mode_string
gsttensor_merge.c, 702

gst_tensor_merge_parent_class
gsttensor_merge.c, 688

gst_tensor_merge_ready_to_paused
gsttensor_merge.c, 697

gst_tensor_merge_request_new_pad
gsttensor_merge.c, 697

gst_tensor_merge_send_segment_event
gsttensor_merge.c, 698

gst_tensor_merge_set_option_data
gsttensor_merge.c, 698

gst_tensor_merge_set_property
gsttensor_merge.c, 699

gst_tensor_merge_set_src_caps
gsttensor_merge.c, 700

gst_tensor_merge_sink_event
gsttensor_merge.c, 701

gst_tensor_merge_src_event
gsttensor_merge.c, 702

gst_tensor_meta_info_append_header
nnstreamer_plugin_api.h, 1008
nnstreamer_plugin_api_impl.c, 1232

gst_tensor_meta_info_convert
nnstreamer_plugin_api_util.h, 1082
nnstreamer_plugin_api_util_impl.c, 1284

gst_tensor_meta_info_get_data_size
nnstreamer_plugin_api_util.h, 1084
nnstreamer_plugin_api_util_impl.c, 1286

gst_tensor_meta_info_get_header_size
nnstreamer_plugin_api_util.h, 1085
nnstreamer_plugin_api_util_impl.c, 1287

gst_tensor_meta_info_get_version
nnstreamer_plugin_api_util.h, 1086
nnstreamer_plugin_api_util_impl.c, 1288

gst_tensor_meta_info_init
nnstreamer_plugin_api_util.h, 1086
nnstreamer_plugin_api_util_impl.c, 1288

gst_tensor_meta_info_parse_header
nnstreamer_plugin_api_util.h, 1087
nnstreamer_plugin_api_util_impl.c, 1289

gst_tensor_meta_info_parse_memory
nnstreamer_plugin_api.h, 1009

nnstreamer_plugin_api_impl.c, [1233](#)
 gst_tensor_meta_info_update_header
 nnstreamer_plugin_api_util.h, [1088](#)
 nnstreamer_plugin_api_util_impl.c, [1290](#)
 gst_tensor_meta_info_validate
 nnstreamer_plugin_api_util.h, [1089](#)
 nnstreamer_plugin_api_util_impl.c, [1291](#)
 GST_TENSOR_META_IS_V1
 nnstreamer_plugin_api_util_impl.c, [1253](#)
 GST_TENSOR_META_IS_VALID
 nnstreamer_plugin_api_util_impl.c, [1254](#)
 GST_TENSOR_META_MAGIC
 nnstreamer_plugin_api_util_impl.c, [1254](#)
 GST_TENSOR_META_MAGIC_VALID
 nnstreamer_plugin_api_util_impl.c, [1254](#)
 GST_TENSOR_META_MAKE_VERSION
 nnstreamer_plugin_api_util_impl.c, [1254](#)
 GST_TENSOR_META_VERSION
 nnstreamer_plugin_api_util_impl.c, [1254](#)
 GST_TENSOR_META_VERSION_VALID
 nnstreamer_plugin_api_util_impl.c, [1255](#)
 GST_TENSOR_MUX
 gsttensor_mux.h, [728](#)
 GST_TENSOR_MUX_CAST
 gsttensor_mux.h, [728](#)
 gst_tensor_mux_chain_flex_tensor
 gsttensor_mux.c, [713](#)
 gst_tensor_mux_change_state
 gsttensor_mux.c, [714](#)
 GST_TENSOR_MUX_CLASS
 gsttensor_mux.h, [728](#)
 gst_tensor_mux_class_init
 gsttensor_mux.c, [714](#)
 gst_tensor_mux_collect_buffer
 gsttensor_mux.c, [715](#)
 gst_tensor_mux_collected
 gsttensor_mux.c, [716](#)
 gst_tensor_mux_do_clip
 gsttensor_mux.c, [717](#)
 gst_tensor_mux_finalize
 gsttensor_mux.c, [718](#)
 GST_TENSOR_MUX_GET_CLASS
 gsttensor_mux.h, [728](#)
 gst_tensor_mux_get_property
 gsttensor_mux.c, [719](#)
 gst_tensor_mux_get_type
 gsttensor_mux.h, [729](#)
 gst_tensor_mux_init
 gsttensor_mux.c, [719](#)
 gst_tensor_mux_parent_class
 gsttensor_mux.c, [712](#)
 gst_tensor_mux_ready_to_paused
 gsttensor_mux.c, [720](#)
 gst_tensor_mux_request_new_pad
 gsttensor_mux.c, [721](#)
 gst_tensor_mux_send_segment_event
 gsttensor_mux.c, [721](#)
 gst_tensor_mux_set_property
 gsttensor_mux.c, [722](#)
 gst_tensor_mux_set_src_caps
 gsttensor_mux.c, [723](#)
 gst_tensor_mux_set_waiting
 gsttensor_mux.c, [723](#)
 gst_tensor_mux_sink_event
 gsttensor_mux.c, [724](#)
 gst_tensor_mux_src_event
 gsttensor_mux.c, [725](#)
 GST_TENSOR_NUM_TENSORS_RANGE
 tensor_typedef.h, [1149](#)
 gst_tensor_pad_caps_from_config
 nnstreamer_plugin_api_impl.c, [1234](#)
 tensor_common.h, [1371](#)
 gst_tensor_pad_caps_is_flexible
 tensor_common.h, [1364](#)
 gst_tensor_pad_caps_is_sparse
 tensor_common.h, [1364](#)
 gst_tensor_pad_caps_is_static
 tensor_common.h, [1364](#)
 gst_tensor_pad_get_format
 nnstreamer_plugin_api_impl.c, [1236](#)
 tensor_common.h, [1372](#)
 gst_tensor_pad_possible_caps_from_config
 nnstreamer_plugin_api_impl.c, [1237](#)
 tensor_common.h, [1373](#)
 gst_tensor_parse_config_file
 nnstreamer_plugin_api_impl.c, [1238](#)
 tensor_common.h, [1374](#)
 gst_tensor_parse_dimension
 nnstreamer_plugin_api_util.h, [1091](#)
 nnstreamer_plugin_api_util_impl.c, [1293](#)
 gst_tensor_parse_layout_string
 tensor_filter_common.c, [1479](#)
 GST_TENSOR_QUERY_CLIENT
 tensor_query_client.h, [1577](#)
 GST_TENSOR_QUERY_CLIENT_CAST
 tensor_query_client.h, [1577](#)
 gst_tensor_query_client_chain
 tensor_query_client.c, [1567](#)
 GST_TENSOR_QUERY_CLIENT_CLASS
 tensor_query_client.h, [1578](#)
 gst_tensor_query_client_class_init
 tensor_query_client.c, [1567](#)
 gst_tensor_query_client_create_edge_handle
 tensor_query_client.c, [1568](#)
 gst_tensor_query_client_finalize
 tensor_query_client.c, [1569](#)
 gst_tensor_query_client_get_property
 tensor_query_client.c, [1569](#)
 gst_tensor_query_client_get_type
 tensor_query_client.h, [1578](#)
 gst_tensor_query_client_init
 tensor_query_client.c, [1570](#)
 gst_tensor_query_client_parent_class
 tensor_query_client.c, [1563](#)
 gst_tensor_query_client_query_caps
 tensor_query_client.c, [1571](#)

gst_tensor_query_client_set_property
 tensor_query_client.c, [1571](#)

gst_tensor_query_client_sink_event
 tensor_query_client.c, [1572](#)

gst_tensor_query_client_sink_query
 tensor_query_client.c, [1573](#)

gst_tensor_query_client_update_caps
 tensor_query_client.c, [1574](#)

gst_tensor_query_get_connect_type
 tensor_query_common.c, [1580](#)
 tensor_query_common.h, [1582](#)

gst_tensor_query_server_add_data
 tensor_query_server.c, [1585](#)
 tensor_query_server.h, [1593](#)

gst_tensor_query_server_get_handle
 tensor_query_server.c, [1585](#)

gst_tensor_query_server_prepare
 tensor_query_server.c, [1586](#)
 tensor_query_server.h, [1594](#)

gst_tensor_query_server_release_edge_handle
 tensor_query_server.c, [1587](#)
 tensor_query_server.h, [1595](#)

gst_tensor_query_server_remove_data
 tensor_query_server.c, [1587](#)
 tensor_query_server.h, [1596](#)

gst_tensor_query_server_send_buffer
 tensor_query_server.c, [1588](#)
 tensor_query_server.h, [1596](#)

gst_tensor_query_server_set_caps
 tensor_query_server.c, [1588](#)
 tensor_query_server.h, [1597](#)

gst_tensor_query_server_set_configured
 tensor_query_server.c, [1589](#)
 tensor_query_server.h, [1598](#)

gst_tensor_query_server_wait_sink
 tensor_query_server.c, [1590](#)
 tensor_query_server.h, [1598](#)

GST_TENSOR_QUERY_SERVERSINK
 tensor_query_serversink.h, [1610](#)

GST_TENSOR_QUERY_SERVERSINK_CAST
 tensor_query_serversink.h, [1611](#)

gst_tensor_query_serversink_change_state
 tensor_query_serversink.c, [1603](#)

GST_TENSOR_QUERY_SERVERSINK_CLASS
 tensor_query_serversink.h, [1611](#)

gst_tensor_query_serversink_class_init
 tensor_query_serversink.c, [1604](#)

gst_tensor_query_serversink_finalize
 tensor_query_serversink.c, [1605](#)

gst_tensor_query_serversink_get_property
 tensor_query_serversink.c, [1605](#)

gst_tensor_query_serversink_get_type
 tensor_query_serversink.h, [1612](#)

gst_tensor_query_serversink_init
 tensor_query_serversink.c, [1606](#)

gst_tensor_query_serversink_parent_class
 tensor_query_serversink.c, [1601](#)

gst_tensor_query_serversink_render
 tensor_query_serversink.c, [1606](#)

gst_tensor_query_serversink_set_caps
 tensor_query_serversink.c, [1607](#)

gst_tensor_query_serversink_set_property
 tensor_query_serversink.c, [1607](#)

GST_TENSOR_QUERY_SERVERSRC
 tensor_query_serversrc.h, [1626](#)

GST_TENSOR_QUERY_SERVERSRC_CAST
 tensor_query_serversrc.h, [1626](#)

gst_tensor_query_serversrc_change_state
 tensor_query_serversrc.c, [1618](#)

GST_TENSOR_QUERY_SERVERSRC_CLASS
 tensor_query_serversrc.h, [1626](#)

gst_tensor_query_serversrc_class_init
 tensor_query_serversrc.c, [1619](#)

gst_tensor_query_serversrc_create
 tensor_query_serversrc.c, [1620](#)

gst_tensor_query_serversrc_finalize
 tensor_query_serversrc.c, [1620](#)

gst_tensor_query_serversrc_get_property
 tensor_query_serversrc.c, [1621](#)

gst_tensor_query_serversrc_get_type
 tensor_query_serversrc.h, [1627](#)

gst_tensor_query_serversrc_init
 tensor_query_serversrc.c, [1621](#)

gst_tensor_query_serversrc_parent_class
 tensor_query_serversrc.c, [1615](#)

gst_tensor_query_serversrc_set_caps
 tensor_query_serversrc.c, [1622](#)

gst_tensor_query_serversrc_set_property
 tensor_query_serversrc.c, [1622](#)

GST_TENSOR_RATE
 gsttensor_rate.h, [751](#)

GST_TENSOR_RATE_CAST
 gsttensor_rate.h, [751](#)

GST_TENSOR_RATE_CLASS
 gsttensor_rate.h, [751](#)

gst_tensor_rate_class_init
 gsttensor_rate.c, [735](#)

gst_tensor_rate_finalize
 gsttensor_rate.c, [736](#)

gst_tensor_rate_fixate_caps
 gsttensor_rate.c, [736](#)

gst_tensor_rate_flush_prev
 gsttensor_rate.c, [737](#)

GST_TENSOR_RATE_GET_CLASS
 gsttensor_rate.h, [751](#)

gst_tensor_rate_get_property
 gsttensor_rate.c, [737](#)

gst_tensor_rate_get_type
 gsttensor_rate.h, [752](#)

gst_tensor_rate_init
 gsttensor_rate.c, [738](#)

gst_tensor_rate_install_properties
 gsttensor_rate.c, [738](#)

gst_tensor_rate_notify_drop
 gsttensor_rate.c, [739](#)

gst_tensor_rate_notify_duplicate

- gsttensor_rate.c, [739](#)
- gst_tensor_rate_parent_class
 - gsttensor_rate.c, [733](#)
- gst_tensor_rate_push_buffer
 - gsttensor_rate.c, [740](#)
- GST_TENSOR_RATE_RANGE
 - tensor_typedef.h, [1149](#)
- gst_tensor_rate_reset
 - gsttensor_rate.c, [741](#)
- GST_TENSOR_RATE_SCALED_TIME
 - gsttensor_rate.c, [734](#)
- gst_tensor_rate_send_qos_throttle
 - gsttensor_rate.c, [741](#)
- gst_tensor_rate_set_caps
 - gsttensor_rate.c, [742](#)
- gst_tensor_rate_set_property
 - gsttensor_rate.c, [742](#)
- gst_tensor_rate_sink_event
 - gsttensor_rate.c, [743](#)
- gst_tensor_rate_start
 - gsttensor_rate.c, [744](#)
- gst_tensor_rate_stop
 - gsttensor_rate.c, [745](#)
- gst_tensor_rate_swap_prev
 - gsttensor_rate.c, [745](#)
- gst_tensor_rate_transform_caps
 - gsttensor_rate.c, [746](#)
- gst_tensor_rate_transform_ip
 - gsttensor_rate.c, [746](#)
- gst_tensor_repo_add_repodata
 - gsttensor_repo.c, [755](#)
 - gsttensor_repo.h, [765](#)
- gst_tensor_repo_check_changed
 - gsttensor_repo.c, [756](#)
 - gsttensor_repo.h, [766](#)
- gst_tensor_repo_check_eos
 - gsttensor_repo.c, [756](#)
 - gsttensor_repo.h, [766](#)
- gst_tensor_repo_get_buffer
 - gsttensor_repo.c, [757](#)
 - gsttensor_repo.h, [767](#)
- gst_tensor_repo_get_repodata
 - gsttensor_repo.c, [758](#)
 - gsttensor_repo.h, [768](#)
- gst_tensor_repo_init
 - gsttensor_repo.c, [758](#)
 - gsttensor_repo.h, [768](#)
- gst_tensor_repo_release_repodata
 - gsttensor_repo.c, [759](#)
- gst_tensor_repo_remove_repodata
 - gsttensor_repo.c, [760](#)
 - gsttensor_repo.h, [769](#)
- gst_tensor_repo_set_buffer
 - gsttensor_repo.c, [760](#)
 - gsttensor_repo.h, [769](#)
- gst_tensor_repo_set_changed
 - gsttensor_repo.c, [761](#)
 - gsttensor_repo.h, [770](#)
- gst_tensor_repo_set_eos
 - gsttensor_repo.c, [762](#)
 - gsttensor_repo.h, [771](#)
- gst_tensor_repo_wait
 - gsttensor_repo.c, [762](#)
 - gsttensor_repo.h, [772](#)
- GST_TENSOR_REPOSINK
 - gsttensor_reposink.h, [786](#)
- GST_TENSOR_REPOSINK_CLASS
 - gsttensor_reposink.h, [787](#)
- gst_tensor_reposink_class_init
 - gsttensor_reposink.c, [776](#)
- gst_tensor_reposink_dispose
 - gsttensor_reposink.c, [776](#)
- gst_tensor_reposink_event
 - gsttensor_reposink.c, [777](#)
- gst_tensor_reposink_get_caps
 - gsttensor_reposink.c, [777](#)
- gst_tensor_reposink_get_property
 - gsttensor_reposink.c, [778](#)
- gst_tensor_reposink_get_type
 - gsttensor_reposink.h, [787](#)
- gst_tensor_reposink_init
 - gsttensor_reposink.c, [778](#)
- gst_tensor_reposink_parent_class
 - gsttensor_reposink.c, [775](#)
- gst_tensor_reposink_query
 - gsttensor_reposink.c, [779](#)
- gst_tensor_reposink_render
 - gsttensor_reposink.c, [779](#)
- gst_tensor_reposink_render_buffer
 - gsttensor_reposink.c, [780](#)
- gst_tensor_reposink_render_list
 - gsttensor_reposink.c, [781](#)
- gst_tensor_reposink_set_caps
 - gsttensor_reposink.c, [781](#)
- gst_tensor_reposink_set_property
 - gsttensor_reposink.c, [782](#)
- gst_tensor_reposink_start
 - gsttensor_reposink.c, [783](#)
- gst_tensor_reposink_stop
 - gsttensor_reposink.c, [783](#)
- GST_TENSOR_REPOSRC
 - gsttensor_reposrc.h, [799](#)
- GST_TENSOR_REPOSRC_CLASS
 - gsttensor_reposrc.h, [799](#)
- gst_tensor_reposrc_class_init
 - gsttensor_reposrc.c, [791](#)
- gst_tensor_reposrc_create
 - gsttensor_reposrc.c, [792](#)
- gst_tensor_reposrc_dispose
 - gsttensor_reposrc.c, [793](#)
- gst_tensor_reposrc_gen_dummy_buffer
 - gsttensor_reposrc.c, [793](#)
- gst_tensor_reposrc_get_property
 - gsttensor_reposrc.c, [794](#)
- gst_tensor_reposrc_get_type
 - gsttensor_reposrc.h, [800](#)

gst_tensor_reposrc_getcaps
gsttensor_reposrc.c, 795

gst_tensor_reposrc_init
gsttensor_reposrc.c, 795

gst_tensor_reposrc_parent_class
gsttensor_reposrc.c, 790

gst_tensor_reposrc_set_property
gsttensor_reposrc.c, 796

gst_tensor_set_all_channels
gsttensor_srcio.c, 890

GST_TENSOR_SINK
gsttensor_sink.h, 819

GST_TENSOR_SINK_CLASS
gsttensor_sink.h, 819

gst_tensor_sink_class_init
gsttensor_sink.c, 805

gst_tensor_sink_event
gsttensor_sink.c, 806

gst_tensor_sink_finalize
gsttensor_sink.c, 807

gst_tensor_sink_get_emit_signal
gsttensor_sink.c, 808

gst_tensor_sink_get_last_render_time
gsttensor_sink.c, 808

gst_tensor_sink_get_property
gsttensor_sink.c, 808

gst_tensor_sink_get_signal_rate
gsttensor_sink.c, 809

gst_tensor_sink_get_silent
gsttensor_sink.c, 810

gst_tensor_sink_get_type
gsttensor_sink.h, 820

gst_tensor_sink_init
gsttensor_sink.c, 810

gst_tensor_sink_parent_class
gsttensor_sink.c, 804

gst_tensor_sink_query
gsttensor_sink.c, 810

gst_tensor_sink_render
gsttensor_sink.c, 811

gst_tensor_sink_render_buffer
gsttensor_sink.c, 812

gst_tensor_sink_render_list
gsttensor_sink.c, 813

gst_tensor_sink_set_emit_signal
gsttensor_sink.c, 814

gst_tensor_sink_set_last_render_time
gsttensor_sink.c, 814

gst_tensor_sink_set_property
gsttensor_sink.c, 815

gst_tensor_sink_set_signal_rate
gsttensor_sink.c, 816

gst_tensor_sink_set_silent
gsttensor_sink.c, 816

GST_TENSOR_SPARSE_DEC
gsttensor_sparsedec.h, 832

gst_tensor_sparse_dec_chain
gsttensor_sparsedec.c, 824

GST_TENSOR_SPARSE_DEC_CLASS
gsttensor_sparsedec.h, 833

gst_tensor_sparse_dec_class_init
gsttensor_sparsedec.c, 825

gst_tensor_sparse_dec_finalize
gsttensor_sparsedec.c, 825

gst_tensor_sparse_dec_get_property
gsttensor_sparsedec.c, 826

gst_tensor_sparse_dec_get_type
gsttensor_sparsedec.h, 833

gst_tensor_sparse_dec_init
gsttensor_sparsedec.c, 826

gst_tensor_sparse_dec_parent_class
gsttensor_sparsedec.c, 823

gst_tensor_sparse_dec_query_caps
gsttensor_sparsedec.c, 827

gst_tensor_sparse_dec_set_property
gsttensor_sparsedec.c, 828

gst_tensor_sparse_dec_sink_event
gsttensor_sparsedec.c, 828

gst_tensor_sparse_dec_sink_query
gsttensor_sparsedec.c, 829

GST_TENSOR_SPARSE_ENC
gsttensor_sparseenc.h, 846

gst_tensor_sparse_enc_chain
gsttensor_sparseenc.c, 837

GST_TENSOR_SPARSE_ENC_CLASS
gsttensor_sparseenc.h, 847

gst_tensor_sparse_enc_class_init
gsttensor_sparseenc.c, 838

gst_tensor_sparse_enc_finalize
gsttensor_sparseenc.c, 839

gst_tensor_sparse_enc_get_property
gsttensor_sparseenc.c, 839

gst_tensor_sparse_enc_get_type
gsttensor_sparseenc.h, 847

gst_tensor_sparse_enc_init
gsttensor_sparseenc.c, 840

gst_tensor_sparse_enc_parent_class
gsttensor_sparseenc.c, 836

gst_tensor_sparse_enc_parse_caps
gsttensor_sparseenc.c, 841

gst_tensor_sparse_enc_query_caps
gsttensor_sparseenc.c, 841

gst_tensor_sparse_enc_set_property
gsttensor_sparseenc.c, 842

gst_tensor_sparse_enc_sink_event
gsttensor_sparseenc.c, 842

gst_tensor_sparse_enc_sink_query
gsttensor_sparseenc.c, 843

gst_tensor_sparse_from_dense
gsttensor_sparseutil.c, 849

gsttensor_sparseutil.h, 852

gst_tensor_sparse_to_dense
gsttensor_sparseutil.c, 850

gsttensor_sparseutil.h, 853

GST_TENSOR_SPLIT
gsttensor_split.h, 871

GST_TENSOR_SPLIT_CAST
 gsttensor_split.h, 871
 gst_tensor_split_chain
 gsttensor_split.c, 858
 gst_tensor_split_change_state
 gsttensor_split.c, 859
 GST_TENSOR_SPLIT_CLASS
 gsttensor_split.h, 871
 gst_tensor_split_class_init
 gsttensor_split.c, 860
 gst_tensor_split_combine_flows
 gsttensor_split.c, 860
 gst_tensor_split_event
 gsttensor_split.c, 861
 gst_tensor_split_finalize
 gsttensor_split.c, 862
 gst_tensor_split_get_capsparam
 gsttensor_split.c, 862
 GST_TENSOR_SPLIT_GET_CLASS
 gsttensor_split.h, 871
 gst_tensor_split_get_property
 gsttensor_split.c, 863
 gst_tensor_split_get_splitted
 gsttensor_split.c, 864
 gst_tensor_split_get_tensor_pad
 gsttensor_split.c, 865
 gst_tensor_split_get_type
 gsttensor_split.h, 872
 gst_tensor_split_init
 gsttensor_split.c, 866
 gst_tensor_split_parent_class
 gsttensor_split.c, 856
 gst_tensor_split_remove_src_pads
 gsttensor_split.c, 867
 gst_tensor_split_set_property
 gsttensor_split.c, 868
 GST_TENSOR_SRC_IIO
 gsttensor_srciiio.h, 918
 GST_TENSOR_SRC_IIO_CAST
 gsttensor_srciiio.h, 918
 gst_tensor_src_iio_change_state
 gsttensor_srciiio.c, 890
 gst_tensor_src_iio_channel_properties_free
 gsttensor_srciiio.c, 891
 GST_TENSOR_SRC_IIO_CLASS
 gsttensor_srciiio.h, 918
 gst_tensor_src_iio_class_init
 gsttensor_srciiio.c, 892
 gst_tensor_src_iio_create
 gsttensor_srciiio.c, 892
 gst_tensor_src_iio_create_config
 gsttensor_srciiio.c, 893
 gst_tensor_src_iio_device_properties_init
 gsttensor_srciiio.c, 894
 gst_tensor_src_iio_event
 gsttensor_srciiio.c, 894
 gst_tensor_src_iio_fill
 gsttensor_srciiio.c, 895
 gst_tensor_src_iio_finalize
 gsttensor_srciiio.c, 896
 gst_tensor_src_iio_fixate
 gsttensor_srciiio.c, 896
 gst_tensor_src_iio_get_all_channel_info
 gsttensor_srciiio.c, 897
 gst_tensor_src_iio_get_available_frequency
 gsttensor_srciiio.c, 898
 gst_tensor_src_iio_get_caps
 gsttensor_srciiio.c, 898
 gst_tensor_src_iio_get_float_from_file
 gsttensor_srciiio.c, 899
 gst_tensor_src_iio_get_generic_name
 gsttensor_srciiio.c, 900
 gst_tensor_src_iio_get_id_by_name
 gsttensor_srciiio.c, 900
 gst_tensor_src_iio_get_name_by_id
 gsttensor_srciiio.c, 900
 gst_tensor_src_iio_get_property
 gsttensor_srciiio.c, 901
 gst_tensor_src_iio_get_times
 gsttensor_srciiio.c, 902
 gst_tensor_src_iio_get_type
 gsttensor_srciiio.h, 920
 gst_tensor_src_iio_init
 gsttensor_srciiio.c, 902
 gst_tensor_src_iio_is_seekable
 gsttensor_srciiio.c, 903
 gst_tensor_src_iio_parent_class
 gsttensor_srciiio.c, 882
 gst_tensor_src_iio_process_scanned_data
 gsttensor_srciiio.c, 903
 gst_tensor_src_iio_set_caps
 gsttensor_srciiio.c, 904
 gst_tensor_src_iio_set_channel_type
 gsttensor_srciiio.c, 904
 gst_tensor_src_iio_set_property
 gsttensor_srciiio.c, 905
 gst_tensor_src_iio_setup_device_buffer
 gsttensor_srciiio.c, 906
 gst_tensor_src_iio_setup_device_properties
 gsttensor_srciiio.c, 906
 gst_tensor_src_iio_setup_sampling_frequency
 gsttensor_srciiio.c, 907
 gst_tensor_src_iio_setup_scan_channels
 gsttensor_srciiio.c, 907
 gst_tensor_src_iio_setup_trigger_properties
 gsttensor_srciiio.c, 908
 gst_tensor_src_iio_start
 gsttensor_srciiio.c, 909
 gst_tensor_src_iio_stop
 gsttensor_srciiio.c, 909
 gst_tensor_src_merge_tensor_by_type
 gsttensor_srciiio.c, 910
 gst_tensor_src_restore_iio_device
 gsttensor_srciiio.c, 911
 gst_tensor_time_sync_buffer_from_collectpad
 nnstreamer_plugin_api_impl.c, 1239

- tensor_common.h, [1375](#)
- gst_tensor_time_sync_flush
 - nnstreamer_plugin_api_impl.c, [1240](#)
 - tensor_common.h, [1376](#)
- gst_tensor_time_sync_get_current_time
 - nnstreamer_plugin_api_impl.c, [1241](#)
 - tensor_common.h, [1377](#)
- gst_tensor_time_sync_get_mode
 - nnstreamer_plugin_api_impl.c, [1242](#)
 - tensor_common.h, [1378](#)
- gst_tensor_time_sync_get_mode_string
 - nnstreamer_plugin_api_impl.c, [1243](#)
 - tensor_common.h, [1379](#)
- gst_tensor_time_sync_mode_string
 - nnstreamer_plugin_api_impl.c, [1249](#)
- gst_tensor_time_sync_set_option_data
 - nnstreamer_plugin_api_impl.c, [1243](#)
 - tensor_common.h, [1380](#)
- GST_TENSOR_TRAINER
 - gsttensor_trainer.h, [952](#)
- gst_tensor_trainer_chain
 - gsttensor_trainer.c, [926](#)
- gst_tensor_trainer_change_state
 - gsttensor_trainer.c, [927](#)
- gst_tensor_trainer_check_buffer_drop_conditions
 - gsttensor_trainer.c, [928](#)
- gst_tensor_trainer_check_chain_conditions
 - gsttensor_trainer.c, [928](#)
- gst_tensor_trainer_check_invalid_param
 - gsttensor_trainer.c, [929](#)
- GST_TENSOR_TRAINER_CLASS
 - gsttensor_trainer.h, [953](#)
- gst_tensor_trainer_class_init
 - gsttensor_trainer.c, [929](#)
- gst_tensor_trainer_convert_meta
 - gsttensor_trainer.c, [930](#)
- gst_tensor_trainer_create_event_notifier
 - gsttensor_trainer.c, [931](#)
- gst_tensor_trainer_create_framework
 - gsttensor_trainer.c, [931](#)
- gst_tensor_trainer_create_model
 - gsttensor_trainer.c, [931](#)
- gst_tensor_trainer_create_output
 - gsttensor_trainer.c, [932](#)
- gst_tensor_trainer_dummy_data_generation_func
 - gsttensor_trainer.c, [933](#)
- gst_tensor_trainer_epochs_is_complete
 - gsttensor_trainer.c, [933](#)
- gst_tensor_trainer_finalize
 - gsttensor_trainer.c, [934](#)
- gst_tensor_trainer_find_framework
 - gsttensor_trainer.c, [935](#)
- GST_TENSOR_TRAINER_FRAMEWORK_BASE
 - nnstreamer_plugin_api_trainer.h, [1044](#)
- GST_TENSOR_TRAINER_FRAMEWORK_V1
 - nnstreamer_plugin_api_trainer.h, [1044](#)
- gst_tensor_trainer_get_model_stats
 - gsttensor_trainer.c, [935](#)
- gst_tensor_trainer_get_property
 - gsttensor_trainer.c, [936](#)
- gst_tensor_trainer_get_tensor_size
 - gsttensor_trainer.c, [936](#)
- gst_tensor_trainer_get_type
 - gsttensor_trainer.h, [953](#)
- gst_tensor_trainer_init
 - gsttensor_trainer.c, [937](#)
- gst_tensor_trainer_parent_class
 - gsttensor_trainer.c, [924](#)
- gst_tensor_trainer_push_input
 - gsttensor_trainer.c, [938](#)
- gst_tensor_trainer_query_caps
 - gsttensor_trainer.c, [939](#)
- gst_tensor_trainer_set_model_load_path
 - gsttensor_trainer.c, [939](#)
- gst_tensor_trainer_set_model_save_path
 - gsttensor_trainer.c, [940](#)
- gst_tensor_trainer_set_output_meta
 - gsttensor_trainer.c, [941](#)
- gst_tensor_trainer_set_prop_framework
 - gsttensor_trainer.c, [941](#)
- gst_tensor_trainer_set_prop_model_config_file_path
 - gsttensor_trainer.c, [942](#)
- gst_tensor_trainer_set_property
 - gsttensor_trainer.c, [943](#)
- gst_tensor_trainer_sink_event
 - gsttensor_trainer.c, [944](#)
- gst_tensor_trainer_sink_query
 - gsttensor_trainer.c, [945](#)
- gst_tensor_trainer_src_query
 - gsttensor_trainer.c, [946](#)
- gst_tensor_trainer_start_model_training
 - gsttensor_trainer.c, [947](#)
- gst_tensor_trainer_stop_model_training
 - gsttensor_trainer.c, [947](#)
- gst_tensor_trainer_wait_for_epoch_completion
 - gsttensor_trainer.c, [947](#)
- gst_tensor_trainer_wait_for_training_completion
 - gsttensor_trainer.c, [948](#)
- GST_TENSOR_TRANSFORM
 - gsttensor_transform.h, [990](#)
- gst_tensor_transform_arithmetic
 - gsttensor_transform.c, [963](#)
- GST_TENSOR_TRANSFORM_CAST
 - gsttensor_transform.h, [990](#)
- gst_tensor_transform_clamp
 - gsttensor_transform.c, [964](#)
- GST_TENSOR_TRANSFORM_CLASS
 - gsttensor_transform.h, [991](#)
- gst_tensor_transform_class_init
 - gsttensor_transform.c, [965](#)
- gst_tensor_transform_convert_dimension
 - gsttensor_transform.c, [966](#)
- gst_tensor_transform_dimchg
 - gsttensor_transform.c, [967](#)
- gst_tensor_transform_finalize
 - gsttensor_transform.c, [969](#)

- gst_tensor_transform_fixate_caps
 - gsttensor_transform.c, [969](#)
- gst_tensor_transform_get_operator
 - gsttensor_transform.c, [970](#)
- gst_tensor_transform_get_property
 - gsttensor_transform.c, [971](#)
- gst_tensor_transform_get_stand_mode
 - gsttensor_transform.c, [972](#)
- gst_tensor_transform_get_type
 - gsttensor_transform.h, [994](#)
- gst_tensor_transform_init
 - gsttensor_transform.c, [972](#)
- gst_tensor_transform_mode_get_type
 - gsttensor_transform.c, [973](#)
- gst_tensor_transform_operator_string
 - gsttensor_transform.c, [986](#)
- gst_tensor_transform_padding
 - gsttensor_transform.c, [973](#)
- gst_tensor_transform_parent_class
 - gsttensor_transform.c, [959](#)
- gst_tensor_transform_read_caps
 - gsttensor_transform.c, [974](#)
- gst_tensor_transform_set_caps
 - gsttensor_transform.c, [975](#)
- gst_tensor_transform_set_option_data
 - gsttensor_transform.c, [976](#)
- gst_tensor_transform_set_property
 - gsttensor_transform.c, [977](#)
- gst_tensor_transform_stand
 - gsttensor_transform.c, [978](#)
- gst_tensor_transform_stand_string
 - gsttensor_transform.c, [986](#)
- gst_tensor_transform_transform
 - gsttensor_transform.c, [979](#)
- gst_tensor_transform_transform_caps
 - gsttensor_transform.c, [981](#)
- gst_tensor_transform_transform_size
 - gsttensor_transform.c, [982](#)
- gst_tensor_transform_transpose
 - gsttensor_transform.c, [983](#)
- gst_tensor_transform_typecast
 - gsttensor_transform.c, [984](#)
- GST_TENSOR_TYPE_ALL
 - tensor_typedef.h, [1150](#)
- gst_tensor_write_sysfs_int
 - gsttensor_srcio.c, [912](#)
- gst_tensor_write_sysfs_string
 - gsttensor_srcio.c, [913](#)
- gst_tensordec_check_consistency
 - gsttensor_decoder.c, [612](#)
- gst_tensordec_class_finalize
 - gsttensor_decoder.c, [612](#)
- gst_tensordec_class_init
 - gsttensor_decoder.c, [613](#)
- gst_tensordec_configure
 - gsttensor_decoder.c, [614](#)
- gst_tensordec_fixate_caps
 - gsttensor_decoder.c, [615](#)
- gst_tensordec_get_property
 - gsttensor_decoder.c, [615](#)
- gst_tensordec_get_type
 - gsttensor_decoder.h, [634](#)
- gst_tensordec_init
 - gsttensor_decoder.c, [616](#)
- gst_tensordec_media_caps_from_structure
 - gsttensor_decoder.c, [616](#)
- gst_tensordec_media_caps_from_tensor
 - gsttensor_decoder.c, [617](#)
- gst_tensordec_parent_class
 - gsttensor_decoder.c, [608](#)
- gst_tensordec_process_plugin_options
 - gsttensor_decoder.c, [618](#)
- gst_tensordec_set_caps
 - gsttensor_decoder.c, [619](#)
- gst_tensordec_set_property
 - gsttensor_decoder.c, [619](#)
- gst_tensordec_transform
 - gsttensor_decoder.c, [620](#)
- gst_tensordec_transform_caps
 - gsttensor_decoder.c, [621](#)
- gst_tensordec_transform_size
 - gsttensor_decoder.c, [622](#)
- GST_TENSORS_CAP_DEFAULT
 - tensor_typedef.h, [1150](#)
- GST_TENSORS_CAP_MAKE
 - tensor_typedef.h, [1150](#)
- GST_TENSORS_CAP_WITH_NUM
 - tensor_typedef.h, [1150](#)
- gst_tensors_caps_from_config
 - nnstreamer_plugin_api.h, [1010](#)
 - nnstreamer_plugin_api_impl.c, [1244](#)
- gst_tensors_config_copy
 - nnstreamer_plugin_api_util.h, [1092](#)
 - nnstreamer_plugin_api_util_impl.c, [1294](#)
- gst_tensors_config_free
 - nnstreamer_plugin_api_util.h, [1093](#)
 - nnstreamer_plugin_api_util_impl.c, [1295](#)
- gst_tensors_config_from_peer
 - nnstreamer_plugin_api.h, [1011](#)
 - nnstreamer_plugin_api_impl.c, [1245](#)
- gst_tensors_config_from_structure
 - nnstreamer_plugin_api.h, [1013](#)
 - nnstreamer_plugin_api_impl.c, [1246](#)
- gst_tensors_config_init
 - nnstreamer_plugin_api_util.h, [1094](#)
 - nnstreamer_plugin_api_util_impl.c, [1296](#)
- gst_tensors_config_is_equal
 - nnstreamer_plugin_api_util.h, [1096](#)
 - nnstreamer_plugin_api_util_impl.c, [1298](#)
- gst_tensors_config_is_flexible
 - nnstreamer_plugin_api_util.h, [1053](#)
- gst_tensors_config_is_sparse
 - nnstreamer_plugin_api_util.h, [1053](#)
- gst_tensors_config_is_static
 - nnstreamer_plugin_api_util.h, [1053](#)
- gst_tensors_config_to_string

- nnstreamer_plugin_api_util.h, 1097
- nnstreamer_plugin_api_util_impl.c, 1299
- gst_tensors_config_validate
 - nnstreamer_plugin_api_util.h, 1098
 - nnstreamer_plugin_api_util_impl.c, 1300
- GST_TENSORS_FLEX_CAP_DEFAULT
 - tensor_typedef.h, 1151
- gst_tensors_info_copy
 - nnstreamer_plugin_api_util.h, 1100
 - nnstreamer_plugin_api_util_impl.c, 1301
- gst_tensors_info_free
 - nnstreamer_plugin_api_util.h, 1101
 - nnstreamer_plugin_api_util_impl.c, 1302
- gst_tensors_info_get_dimensions_string
 - nnstreamer_plugin_api_util.h, 1103
 - nnstreamer_plugin_api_util_impl.c, 1304
- gst_tensors_info_get_names_string
 - nnstreamer_plugin_api_util.h, 1104
 - nnstreamer_plugin_api_util_impl.c, 1305
- gst_tensors_info_get_nth_info
 - nnstreamer_plugin_api_util.h, 1105
 - nnstreamer_plugin_api_util_impl.c, 1306
- gst_tensors_info_get_rank_dimensions_string
 - nnstreamer_plugin_api_util.h, 1106
 - nnstreamer_plugin_api_util_impl.c, 1307
- gst_tensors_info_get_size
 - nnstreamer_plugin_api_util.h, 1107
 - nnstreamer_plugin_api_util_impl.c, 1308
- gst_tensors_info_get_types_string
 - nnstreamer_plugin_api_util.h, 1108
 - nnstreamer_plugin_api_util_impl.c, 1309
- gst_tensors_info_init
 - nnstreamer_plugin_api_util.h, 1109
 - nnstreamer_plugin_api_util_impl.c, 1310
- gst_tensors_info_is_equal
 - nnstreamer_plugin_api_util.h, 1111
 - nnstreamer_plugin_api_util_impl.c, 1312
- gst_tensors_info_parse_dimensions_string
 - nnstreamer_plugin_api_util.h, 1112
 - nnstreamer_plugin_api_util_impl.c, 1313
- gst_tensors_info_parse_names_string
 - nnstreamer_plugin_api_util.h, 1114
 - nnstreamer_plugin_api_util_impl.c, 1315
- gst_tensors_info_parse_types_string
 - nnstreamer_plugin_api_util.h, 1115
 - nnstreamer_plugin_api_util_impl.c, 1316
- gst_tensors_info_to_string
 - nnstreamer_plugin_api_util.h, 1117
 - nnstreamer_plugin_api_util_impl.c, 1318
- gst_tensors_info_validate
 - nnstreamer_plugin_api_util.h, 1118
 - nnstreamer_plugin_api_util_impl.c, 1319
- gst_tensors_layout_init
 - tensor_filter_common.c, 1479
- gst_tensors_parse_layouts_string
 - tensor_filter_common.c, 1480
- gst_tensors_rank_init
 - tensor_filter_common.c, 1481
- GST_TENSORS_SPARSE_CAP_DEFAULT
 - tensor_typedef.h, 1151
- gst_tensorsinfo_compare_print
 - tensor_filter_common.c, 1481
 - tensor_filter_common.h, 1515
- gst_tensorsinfo_compare_to_string
 - tensor_filter_common.c, 1482
 - tensor_filter_common.h, 1516
- GST_TF_FW_INVOKE_COMPAT
 - tensor_filter_common.h, 1499
- GST_TF_FW_V0
 - tensor_filter_common.h, 1500
- GST_TF_FW_V1
 - tensor_filter_common.h, 1500
- GST_TF_FW_VN
 - tensor_filter_common.h, 1500
- GST_TF_STAT_MAX_RECENT
 - tensor_filter_common.h, 1500
- GST_TYPE_DATA_REPO_SINK
 - gstdataareposink.h, 318
- GST_TYPE_DATA_REPO_SRC
 - gstdataareposrc.h, 343
- GST_TYPE_EDGE_CONNECT_TYPE
 - edge_common.h, 348
- GST_TYPE_EDGESINK
 - edge_sink.h, 366
- GST_TYPE_EDGESRC
 - edge_src.h, 381
- GST_TYPE_JOIN
 - gstjoin.h, 402
- GST_TYPE_JOIN_PAD
 - gstjoin.c, 387
- GST_TYPE_MQTT_SINK
 - mqttsink.h, 446
- GST_TYPE_MQTT_SRC
 - mqttsrc.h, 483
- GST_TYPE_QUERY_CONNECT_TYPE
 - tensor_query_common.h, 1582
- GST_TYPE_TENSOR_AGGREGATOR
 - gsttensor_aggregator.h, 514
- GST_TYPE_TENSOR_CONVERTER
 - gsttensor_converter.h, 556
- GST_TYPE_TENSOR_CROP
 - gsttensor_crop.h, 587
- GST_TYPE_TENSOR_DEBUG
 - gsttensor_debug.h, 602
- GST_TYPE_TENSOR_DECODER
 - gsttensor_decoder.h, 633
- GST_TYPE_TENSOR_DEMUX
 - gsttensor_demux.h, 652
- GST_TYPE_TENSOR_FILTER
 - tensor_filter.h, 1434
- GST_TYPE_TENSOR_IF
 - gsttensor_if.h, 682
- GST_TYPE_TENSOR_IF_ACT
 - gsttensor_if.c, 657
- GST_TYPE_TENSOR_IF_CV
 - gsttensor_if.c, 657

- GST_TYPE_TENSOR_IF_OP
 - gsttensor_if.c, [657](#)
- GST_TYPE_TENSOR_MERGE
 - gsttensor_merge.h, [707](#)
- GST_TYPE_TENSOR_MUX
 - gsttensor_mux.h, [729](#)
- GST_TYPE_TENSOR_QUERY_CLIENT
 - tensor_query_client.h, [1578](#)
- GST_TYPE_TENSOR_QUERY_SERVERSINK
 - tensor_query_serversink.h, [1611](#)
- GST_TYPE_TENSOR_QUERY_SERVERSRC
 - tensor_query_serversrc.h, [1626](#)
- GST_TYPE_TENSOR_RATE
 - gsttensor_rate.h, [751](#)
- GST_TYPE_TENSOR_REPOSINK
 - gsttensor_reposink.h, [787](#)
- GST_TYPE_TENSOR_REPOSRC
 - gsttensor_reposrc.h, [799](#)
- GST_TYPE_TENSOR_SINK
 - gsttensor_sink.h, [819](#)
- GST_TYPE_TENSOR_SPARSE_DEC
 - gsttensor_sparsedec.h, [833](#)
- GST_TYPE_TENSOR_SPARSE_ENC
 - gsttensor_sparseenc.h, [847](#)
- GST_TYPE_TENSOR_SPLIT
 - gsttensor_split.h, [872](#)
- GST_TYPE_TENSOR_SRC_IIO
 - gsttensor_srciiio.h, [918](#)
- GST_TYPE_TENSOR_TRAINER
 - gsttensor_trainer.h, [953](#)
- GST_TYPE_TENSOR_TRANSFORM
 - gsttensor_transform.h, [991](#)
- GST_TYPE_TENSOR_TRANSFORM_MODE
 - gsttensor_transform.c, [959](#)
- GST_US_TO_NS_MULTIPLIER
 - mqttcommon.h, [405](#)
- GST_VIDEO_FORMAT_ABGR
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_ARGB
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_BGR
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_BGRA
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_BGRx
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_GRAY16_BE
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_GRAY16_LE
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_GRAY8
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_I420
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_RGB
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_RGBA
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_RGBx
 - gsttensor_converter_media_no_video.h, [568](#)
- gst_video_format_to_string
 - gsttensor_converter_media_no_video.h, [566](#)
- GST_VIDEO_FORMAT_UNKNOWN
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_xBGR
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_FORMAT_xRGB
 - gsttensor_converter_media_no_video.h, [568](#)
- GST_VIDEO_INFO_FORMAT
 - gsttensor_converter_media_no_video.h, [566](#)
- GST_VIDEO_INFO_FPS_D
 - gsttensor_converter_media_no_video.h, [566](#)
- GST_VIDEO_INFO_FPS_N
 - gsttensor_converter_media_no_video.h, [566](#)
- gst_video_info_from_caps
 - gsttensor_converter_media_no_video.h, [566](#)
- GST_VIDEO_INFO_HEIGHT
 - gsttensor_converter_media_no_video.h, [567](#)
- gst_video_info_init
 - gsttensor_converter_media_no_video.h, [567](#)
- GST_VIDEO_INFO_SIZE
 - gsttensor_converter_media_no_video.h, [567](#)
- GST_VIDEO_INFO_WIDTH
 - gsttensor_converter_media_no_video.h, [567](#)
- GstAudioFormat
 - gsttensor_converter_media_no_audio.h, [564](#)
- GstAudioInfo
 - gsttensor_converter_media_no_audio.h, [564](#)
- gstdatarepo.c
 - gst_data_repo_get_data_type_from_caps, [294](#)
 - PACKAGE, [294](#)
 - plugin_init, [295](#)
- gstdatarepo.h
 - GST_DATA_REPO_DATA_AUDIO, [297](#)
 - GST_DATA_REPO_DATA_IMAGE, [297](#)
 - GST_DATA_REPO_DATA_MAX, [297](#)
 - GST_DATA_REPO_DATA_OCTET, [297](#)
 - GST_DATA_REPO_DATA_TENSOR, [297](#)
 - GST_DATA_REPO_DATA_TEXT, [297](#)
 - GST_DATA_REPO_DATA_UNKNOWN, [297](#)
 - GST_DATA_REPO_DATA_VIDEO, [297](#)
 - gst_data_repo_get_data_type_from_caps, [297](#)
 - GstDataRepoDataType, [297](#)
- GstDataRepoDataType
 - gstdatarepo.h, [297](#)
- GstDataRepoSink
 - gstdatareposink.h, [318](#)
- gstdatareposink.c
 - __write_json, [304](#)
 - _do_init, [301](#)
 - AUDIO_CAPS, [301](#)
 - G_DEFINE_TYPE_WITH_CODE, [304](#)
 - GST_CAT_DEFAULT, [301](#)
 - gst_data_repo_sink_change_state, [304](#)
 - gst_data_repo_sink_class_init, [305](#)
 - gst_data_repo_sink_finalize, [306](#)

- gst_data_repo_sink_get_caps, [307](#)
- gst_data_repo_sink_get_image_filename, [307](#)
- gst_data_repo_sink_get_property, [307](#)
- gst_data_repo_sink_init, [308](#)
- gst_data_repo_sink_open_file, [308](#)
- gst_data_repo_sink_parent_class, [302](#)
- gst_data_repo_sink_query, [309](#)
- gst_data_repo_sink_render, [309](#)
- gst_data_repo_sink_set_caps, [310](#)
- gst_data_repo_sink_set_is_static_tensors, [311](#)
- gst_data_repo_sink_set_property, [312](#)
- gst_data_repo_sink_stop, [312](#)
- gst_data_repo_sink_write_flexible_or_sparse_tensors, [313](#)
- gst_data_repo_sink_write_json_meta_file, [313](#)
- gst_data_repo_sink_write_multi_images, [314](#)
- gst_data_repo_sink_write_others, [315](#)
- GST_DEBUG_CATEGORY_STATIC, [315](#)
- IMAGE_CAPS, [302](#)
- OCTET_CAPS, [302](#)
- PROP_0, [304](#)
- PROP_JSON, [304](#)
- PROP_LOCATION, [304](#)
- sinktemplate, [315](#)
- SUPPORTED_AUDIO_FORMAT, [302](#)
- SUPPORTED_VIDEO_FORMAT, [302](#)
- TENSOR_CAPS, [303](#)
- TEXT_CAPS, [303](#)
- VIDEO_CAPS, [303](#)
- gstdatareposink.h
 - GST_DATA_REPO_SINK, [317](#)
 - GST_DATA_REPO_SINK_CAST, [317](#)
 - GST_DATA_REPO_SINK_CLASS, [317](#)
 - gst_data_repo_sink_get_type, [318](#)
 - GST_IS_DATA_REPO_SINK, [317](#)
 - GST_IS_DATA_REPO_SINK_CLASS, [317](#)
 - GST_TYPE_DATA_REPO_SINK, [318](#)
 - GstDataRepoSink, [318](#)
 - GstDataRepoSinkClass, [318](#)
- GstDataRepoSinkClass
 - gstdatareposink.h, [318](#)
- GstDataRepoSrc
 - gstdatareposrc.h, [343](#)
- gstdatareposrc.c
 - _do_init, [321](#)
 - DEFAULT_EPOCHS, [322](#)
 - DEFAULT_INDEX, [322](#)
 - DEFAULT_IS_SHUFFLE, [322](#)
 - G_DEFINE_TYPE_WITH_CODE, [324](#)
 - GST_CAT_DEFAULT, [322](#)
 - gst_data_repo_get_caps_by_tensors_sequence, [324](#)
 - gst_data_repo_src_change_state, [325](#)
 - gst_data_repo_src_class_init, [326](#)
 - gst_data_repo_src_create, [326](#)
 - gst_data_repo_src_epoch_is_done, [327](#)
 - gst_data_repo_src_finalize, [327](#)
 - gst_data_repo_src_get_caps, [328](#)
 - gst_data_repo_src_get_file_offset, [329](#)
 - gst_data_repo_src_get_image_filename, [329](#)
 - gst_data_repo_src_get_num_tensors, [329](#)
 - gst_data_repo_src_get_property, [330](#)
 - gst_data_repo_src_init, [330](#)
 - gst_data_repo_src_parent_class, [322](#)
 - gst_data_repo_src_parse_caps, [331](#)
 - gst_data_repo_src_read_flexible_or_sparse_tensors, [332](#)
 - gst_data_repo_src_read_json_file, [332](#)
 - gst_data_repo_src_read_multi_images, [333](#)
 - gst_data_repo_src_read_others, [334](#)
 - gst_data_repo_src_read_tensors, [335](#)
 - gst_data_repo_src_set_caps, [336](#)
 - gst_data_repo_src_set_file_path, [336](#)
 - gst_data_repo_src_set_property, [337](#)
 - gst_data_repo_src_set_tensors_sequence, [338](#)
 - gst_data_repo_src_set_timestamp, [338](#)
 - gst_data_repo_src_shuffle_samples_index, [339](#)
 - gst_data_repo_src_start, [339](#)
 - gst_data_repo_src_stop, [340](#)
 - GST_DEBUG_CATEGORY_STATIC, [340](#)
 - O_BINARY, [322](#)
 - PROP_0, [324](#)
 - PROP_CAPS, [324](#)
 - PROP_EPOCHS, [324](#)
 - PROP_IS_SHUFFLE, [324](#)
 - PROP_JSON, [324](#)
 - PROP_LOCATION, [324](#)
 - PROP_START_SAMPLE_INDEX, [324](#)
 - PROP_STOP_SAMPLE_INDEX, [324](#)
 - PROP_TENSORS_SEQUENCE, [324](#)
 - S_ISDIR, [323](#)
 - S_ISREG, [323](#)
 - S_ISSOCK, [323](#)
 - srctemplate, [341](#)
 - struct_stat, [323](#)
- gstdatareposrc.h
 - GST_DATA_REPO_SRC, [342](#)
 - GST_DATA_REPO_SRC_CLASS, [343](#)
 - gst_data_repo_src_get_type, [344](#)
 - GST_IS_DATA_REPO_SRC, [343](#)
 - GST_IS_DATA_REPO_SRC_CLASS, [343](#)
 - GST_TYPE_DATA_REPO_SRC, [343](#)
 - GstDataRepoSrc, [343](#)
 - GstDataRepoSrcClass, [344](#)
- GstDataRepoSrcClass
 - gstdatareposrc.h, [344](#)
- GstEdgeSink
 - edge_sink.h, [366](#)
- GstEdgeSinkClass
 - edge_sink.h, [366](#)
- GstEdgeSrc
 - edge_src.h, [382](#)
- GstEdgeSrcClass
 - edge_src.h, [382](#)
- GstJoin
 - gstjoin.h, [402](#)

- gstjoin.c
 - forward_sticky_events, 388
 - G_DEFINE_TYPE, 388
 - G_DEFINE_TYPE_WITH_CODE, 388
 - GST_CAT_DEFAULT, 385
 - GST_DEBUG_CATEGORY_STATIC, 388
 - GST_IS_JOIN_PAD, 385
 - GST_IS_JOIN_PAD_CLASS, 385
 - gst_join_class_init, 389
 - gst_join_dispose, 389
 - gst_join_finalize, 390
 - gst_join_get_active_sinkpad, 390
 - GST_JOIN_GET_COND, 385
 - gst_join_get_linked_pad, 391
 - GST_JOIN_GET_LOCK, 385
 - gst_join_get_property, 391
 - gst_join_init, 392
 - GST_JOIN_LOCK, 385
 - GST_JOIN_PAD, 386
 - GST_JOIN_PAD_CAST, 386
 - gst_join_pad_chain, 392
 - GST_JOIN_PAD_CLASS, 386
 - gst_join_pad_class_init, 393
 - gst_join_pad_event, 393
 - gst_join_pad_finalize, 394
 - gst_join_pad_get_type, 394
 - gst_join_pad_init, 395
 - gst_join_pad_iterate_linked_pads, 395
 - gst_join_pad_query, 396
 - gst_join_pad_reset, 396
 - gst_join_parent_class, 386
 - gst_join_request_new_pad, 397
 - gst_join_set_active_pad, 398
 - gst_join_sink_factory, 399
 - gst_join_src_factory, 399
 - GST_JOIN_UNLOCK, 386
 - GST_JOIN_WAIT, 386
 - GST_TYPE_JOIN_PAD, 387
 - GstJoinPad, 387
 - GstJoinPadClass, 387
 - PACKAGE, 387
 - plugin_init, 398
 - PROP_0, 388
 - PROP_ACTIVE_PAD, 388
 - PROP_N_PADS, 388
- gstjoin.h
 - GST_IS_JOIN, 401
 - GST_IS_JOIN_CLASS, 401
 - GST_JOIN, 401
 - GST_JOIN_CLASS, 401
 - gst_join_get_type, 402
 - GST_TYPE_JOIN, 402
 - GstJoin, 402
 - GstJoinClass, 402
- GstJoinClass
 - gstjoin.h, 402
- GstJoinPad
 - gstjoin.c, 387
- GstJoinPadClass
 - gstjoin.c, 387
- GstMetaQuery, 252
 - client_id, 252
 - meta, 252
- GstMQTTMessageHdr
 - mqttcommon.h, 406
- GstMqttSink
 - mqttsink.h, 446
- GstMqttSinkClass
 - mqttsink.h, 446
- GstMqttSrc
 - mqttsrc.h, 483
- GstMqttSrcClass
 - mqttsrc.h, 483
- GstSparseTensorInfo, 253
 - nnz, 253
- gsttensor_aggregator.c
 - CAPS_STRING, 494
 - DBG, 494
 - DEFAULT_CONCAT, 494
 - DEFAULT_FRAMES_DIMENSION, 494
 - DEFAULT_FRAMES_FLUSH, 494
 - DEFAULT_FRAMES_IN, 494
 - DEFAULT_FRAMES_OUT, 495
 - DEFAULT_SILENT, 495
 - G_DEFINE_TYPE, 496
 - GST_CAT_DEFAULT, 495
 - GST_DEBUG_CATEGORY_STATIC, 497
 - gst_tensor_aggregator_chain, 497
 - gst_tensor_aggregator_change_state, 498
 - gst_tensor_aggregator_check_concat_axis, 498
 - gst_tensor_aggregator_class_init, 499
 - gst_tensor_aggregator_concat, 500
 - gst_tensor_aggregator_finalize, 502
 - gst_tensor_aggregator_get_adapter, 503
 - gst_tensor_aggregator_get_property, 504
 - gst_tensor_aggregator_init, 504
 - gst_tensor_aggregator_parent_class, 495
 - gst_tensor_aggregator_parse_caps, 505
 - gst_tensor_aggregator_push, 506
 - gst_tensor_aggregator_query_caps, 506
 - gst_tensor_aggregator_reset, 507
 - gst_tensor_aggregator_set_property, 508
 - gst_tensor_aggregator_sink_event, 508
 - gst_tensor_aggregator_sink_query, 509
 - gst_tensor_aggregator_src_query, 510
 - PROP_0, 496
 - PROP_CONCAT, 496
 - PROP_FRAMES_DIMENSION, 496
 - PROP_FRAMES_FLUSH, 496
 - PROP_FRAMES_IN, 496
 - PROP_FRAMES_OUT, 496
 - PROP_SILENT, 496
 - silent_debug_config, 495
 - sink_template, 511
 - src_template, 511
- gsttensor_aggregator.h

- GST_IS_TENSOR_AGGREGATOR, 513
- GST_IS_TENSOR_AGGREGATOR_CLASS, 513
- GST_TENSOR_AGGREGATOR, 514
- GST_TENSOR_AGGREGATOR_CLASS, 514
- gst_tensor_aggregator_get_type, 515
- GST_TYPE_TENSOR_AGGREGATOR, 514
- GstTensorAggregator, 514
- GstTensorAggregatorClass, 514
- gsttensor_converter.c
 - _gst_tensor_converter_chain_chunk, 522
 - _gst_tensor_converter_chain_flex_tensor, 523
 - _gst_tensor_converter_chain_octet, 523
 - _gst_tensor_converter_chain_push, 524
 - _gst_tensor_converter_chain_segment, 525
 - _gst_tensor_converter_chain_timestamp, 526
 - append_flex_tensor_caps_template, 519
 - append_octet_caps_template, 519
 - append_text_caps_template, 519
 - DBG, 519
 - DEFAULT_FRAMES_PER_TENSOR, 519
 - DEFAULT_SET_TIMESTAMP, 519
 - DEFAULT_SILENT, 520
 - findExternalConverter, 526
 - G_DEFINE_TYPE, 527
 - GST_CAT_DEFAULT, 520
 - GST_DEBUG_CATEGORY_STATIC, 527
 - gst_tensor_converter_chain, 527
 - gst_tensor_converter_change_state, 528
 - gst_tensor_converter_class_init, 529
 - gst_tensor_converter_finalize, 530
 - gst_tensor_converter_get_adapter, 531
 - gst_tensor_converter_get_format_list, 532
 - gst_tensor_converter_get_possible_media_caps, 532
 - gst_tensor_converter_get_property, 533
 - gst_tensor_converter_init, 534
 - gst_tensor_converter_parent_class, 520
 - gst_tensor_converter_parse_audio, 535
 - gst_tensor_converter_parse_caps, 536
 - gst_tensor_converter_parse_custom, 537
 - gst_tensor_converter_parse_octet, 538
 - gst_tensor_converter_parse_tensor, 539
 - gst_tensor_converter_parse_text, 540
 - gst_tensor_converter_parse_video, 541
 - gst_tensor_converter_query_caps, 542
 - gst_tensor_converter_reset, 543
 - gst_tensor_converter_set_property, 544
 - gst_tensor_converter_sink_event, 545
 - gst_tensor_converter_sink_query, 546
 - gst_tensor_converter_src_query, 547
 - gst_tensor_converter_update_caps, 548
 - gst_tensor_converter_video_stride, 549
 - nnstreamer_converter_custom_register, 549
 - nnstreamer_converter_custom_unregister, 549
 - nnstreamer_converter_find, 550
 - nnstreamer_converter_set_custom_property_desc, 551
 - nnstreamer_converter_validate, 551
- OCTET_CAPS_STR, 520
- PROP_0, 522
- PROP_FRAMES_PER_TENSOR, 522
- PROP_INPUT_DIMENSION, 522
- PROP_INPUT_TYPE, 522
- PROP_MODE, 522
- PROP_SET_TIMESTAMP, 522
- PROP_SILENT, 522
- PROP_SUBPLUGINS, 522
- registerExternalConverter, 552
- silent_debug_timestamp, 520
- STRING_CUSTOM_MODE, 521
- TEXT_CAPS_STR, 521
- unregisterExternalConverter, 552
- gsttensor_converter.h
 - _CONVERTER_MODE_CUSTOM_CODE, 557
 - _CONVERTER_MODE_CUSTOM_SCRIPT, 557
 - _CONVERTER_MODE_NONE, 557
 - GST_IS_TENSOR_CONVERTER, 555
 - GST_IS_TENSOR_CONVERTER_CLASS, 555
 - GST_TENSOR_CONVERTER, 556
 - GST_TENSOR_CONVERTER_CLASS, 556
 - gst_tensor_converter_get_type, 557
 - GST_TYPE_TENSOR_CONVERTER, 556
 - GstTensorConverter, 556
 - GstTensorConverterClass, 556
 - tensor_converter_mode, 556
- gsttensor_converter_media_info_audio.h
 - append_audio_caps_template, 558
 - AUDIO_CAPS_STR, 559
 - is_audio_supported, 559
- gsttensor_converter_media_info_video.h
 - append_video_caps_template, 561
 - is_video_supported, 561
 - NNS_VIDEO_FORMAT, 561
 - VIDEO_CAPS_STR, 561
- gsttensor_converter_media_no_audio.h
 - append_audio_caps_template, 562
 - GST_AUDIO_FORMAT_F32, 564
 - GST_AUDIO_FORMAT_F64, 564
 - GST_AUDIO_FORMAT_S16, 564
 - GST_AUDIO_FORMAT_S32, 564
 - GST_AUDIO_FORMAT_S8, 564
 - gst_audio_format_to_string, 563
 - GST_AUDIO_FORMAT_U16, 564
 - GST_AUDIO_FORMAT_U32, 564
 - GST_AUDIO_FORMAT_U8, 564
 - GST_AUDIO_FORMAT_UNKNOWN, 564
 - GST_AUDIO_INFO_BPF, 563
 - GST_AUDIO_INFO_CHANNELS, 563
 - GST_AUDIO_INFO_FORMAT, 563
 - gst_audio_info_from_caps, 563
 - gst_audio_info_init, 563
 - GST_AUDIO_INFO_RATE, 564
 - GstAudioFormat, 564
 - GstAudioInfo, 564
 - is_audio_supported, 564
- gsttensor_converter_media_no_video.h

- append_video_caps_template, 566
- GST_VIDEO_FORMAT_ABGR, 568
- GST_VIDEO_FORMAT_ARGB, 568
- GST_VIDEO_FORMAT_BGR, 568
- GST_VIDEO_FORMAT_BGRA, 568
- GST_VIDEO_FORMAT_BGRx, 568
- GST_VIDEO_FORMAT_GRAY16_BE, 568
- GST_VIDEO_FORMAT_GRAY16_LE, 568
- GST_VIDEO_FORMAT_GRAY8, 568
- GST_VIDEO_FORMAT_I420, 568
- GST_VIDEO_FORMAT_RGB, 568
- GST_VIDEO_FORMAT_RGBA, 568
- GST_VIDEO_FORMAT_RGBx, 568
- gst_video_format_to_string, 566
- GST_VIDEO_FORMAT_UNKNOWN, 568
- GST_VIDEO_FORMAT_xBGR, 568
- GST_VIDEO_FORMAT_xRGB, 568
- GST_VIDEO_INFO_FORMAT, 566
- GST_VIDEO_INFO_FPS_D, 566
- GST_VIDEO_INFO_FPS_N, 566
- gst_video_info_from_caps, 566
- GST_VIDEO_INFO_HEIGHT, 567
- gst_video_info_init, 567
- GST_VIDEO_INFO_SIZE, 567
- GST_VIDEO_INFO_WIDTH, 567
- GstVideoFormat, 568
- GstVideoInfo, 567
- is_video_supported, 567
- gsttensor_crop.c
 - DEFAULT_LATENESS, 571
 - DEFAULT_SILENT, 571
 - G_DEFINE_TYPE, 572
 - GST_CAT_DEFAULT, 571
 - GST_DEBUG_CATEGORY_STATIC, 572
 - gst_tensor_crop_chain, 572
 - gst_tensor_crop_change_state, 573
 - gst_tensor_crop_class_init, 574
 - gst_tensor_crop_collected, 574
 - gst_tensor_crop_do_cropping, 575
 - gst_tensor_crop_finalize, 576
 - gst_tensor_crop_get_crop_info, 577
 - gst_tensor_crop_get_property, 578
 - gst_tensor_crop_init, 578
 - gst_tensor_crop_negotiate, 579
 - gst_tensor_crop_pad_reset, 580
 - gst_tensor_crop_parent_class, 571
 - gst_tensor_crop_prepare_out_meta, 580
 - gst_tensor_crop_reset, 581
 - gst_tensor_crop_set_property, 582
 - gst_tensor_crop_sink_event, 582
 - gst_tensor_crop_src_event, 583
 - info_template, 584
 - PROP_0, 572
 - PROP_LATENESS, 572
 - PROP_SILENT, 572
 - raw_template, 584
 - src_template, 584
- gsttensor_crop.h
 - GST_IS_TENSOR_CROP, 587
 - GST_IS_TENSOR_CROP_CLASS, 587
 - GST_TENSOR_CROP, 587
 - GST_TENSOR_CROP_CLASS, 587
 - gst_tensor_crop_get_type, 588
 - GST_TYPE_TENSOR_CROP, 587
 - GstTensorCrop, 588
 - GstTensorCropClass, 588
- gsttensor_debug.c
 - _gst_tensor_debug_output, 593
 - C_FLAGS, 590
 - CAPS_STRING, 590
 - DBG, 590
 - DEFAULT_SILENT, 591
 - DEFAULT_TENSOR_DEBUG_CAP, 591
 - DEFAULT_TENSOR_DEBUG_META_FLAGS, 591
 - DEFAULT_TENSOR_DEBUG_OUTPUT_FLAGS, 591
 - G_DEFINE_TYPE, 593
 - GST_CAT_DEFAULT, 591
 - GST_DEBUG_CATEGORY_STATIC, 593
 - gst_tensor_debug_class_init, 594
 - gst_tensor_debug_finalize, 594
 - gst_tensor_debug_fixate_caps, 595
 - gst_tensor_debug_get_property, 595
 - gst_tensor_debug_init, 596
 - gst_tensor_debug_parent_class, 592
 - gst_tensor_debug_set_caps, 596
 - gst_tensor_debug_set_property, 596
 - gst_tensor_debug_transform_ip, 597
 - PROP_0, 593
 - PROP_CAP, 593
 - PROP_META, 593
 - PROP_OUTPUT, 593
 - PROP_SILENT, 593
 - sink_factory, 598
 - src_factory, 598
 - tensor_debug_cap_get_type, 597
 - tensor_debug_meta_flags_get_type, 598
 - tensor_debug_output_flags_get_type, 598
 - TENSOR_DEBUG_TYPE_CAPS, 592
 - TENSOR_DEBUG_TYPE_META_FLAGS, 592
 - TENSOR_DEBUG_TYPE_OUTPUT_FLAGS, 592
- gsttensor_debug.h
 - GST_IS_TENSOR_DEBUG, 601
 - GST_IS_TENSOR_DEBUG_CLASS, 601
 - GST_TENSOR_DEBUG, 601
 - GST_TENSOR_DEBUG_CAST, 601
 - GST_TENSOR_DEBUG_CLASS, 602
 - gst_tensor_debug_get_type, 604
 - GST_TYPE_TENSOR_DEBUG, 602
 - GstTensorDebug, 602
 - GstTensorDebugClass, 602
 - TDBG_CAP_DISABLED, 603
 - tdbg_cap_mode, 602
 - TDBG_CAP_SHOW_ALWAYS, 603
 - TDBG_CAP_SHOW_UPDATE, 603
 - TDBG_CAP_SHOW_UPDATE_F, 603

- TDBG_META_DISABLED, 603
- tdbg_meta_mode, 603
- TDBG_META_QUERYSERVER, 603
- TDBG_META_TIMESTAMP, 603
- TDBG_OUTPUT_CIRCULARBUF, 603
- TDBG_OUTPUT_CONSOLE_E, 603
- TDBG_OUTPUT_CONSOLE_I, 603
- TDBG_OUTPUT_CONSOLE_W, 603
- TDBG_OUTPUT_DISABLED, 603
- TDBG_OUTPUT_FILEWRITE, 603
- TDBG_OUTPUT_GSTDBG_E, 603
- TDBG_OUTPUT_GSTDBG_I, 603
- TDBG_OUTPUT_GSTDBG_W, 603
- tdbg_output_mode, 603
- gsttensor_decoder.c
 - CAPS_STRING, 607
 - DBG, 607
 - DEFAULT_SILENT, 607
 - failed, 609
 - G_DEFINE_TYPE, 610
 - g_free, 610
 - g_value_set_string, 610
 - GST_CAT_DEFAULT, 607
 - GST_DEBUG_CATEGORY_STATIC, 611
 - gst_tensor_decoder_clean_plugin, 608
 - gst_tensordec_check_consistency, 612
 - gst_tensordec_class_finalize, 612
 - gst_tensordec_class_init, 613
 - gst_tensordec_configure, 614
 - gst_tensordec_fixate_caps, 615
 - gst_tensordec_get_property, 615
 - gst_tensordec_init, 616
 - gst_tensordec_media_caps_from_structure, 616
 - gst_tensordec_media_caps_from_tensor, 617
 - gst_tensordec_parent_class, 608
 - gst_tensordec_process_plugin_options, 618
 - gst_tensordec_set_caps, 619
 - gst_tensordec_set_property, 619
 - gst_tensordec_transform, 620
 - gst_tensordec_transform_caps, 621
 - gst_tensordec_transform_size, 622
 - if, 623
 - nnstreamer_decoder_custom_register, 624
 - nnstreamer_decoder_custom_unregister, 625
 - nnstreamer_decoder_exit, 625
 - nnstreamer_decoder_find, 627
 - nnstreamer_decoder_probe, 628
 - nnstreamer_decoder_set_custom_property_desc, 628
 - nnstreamer_decoder_validate, 629
 - opnum, 629
 - option, 629
 - PROP_0, 609
 - PROP_CONFIG, 609
 - PROP_MODE, 609
 - PROP_MODE_OPTION, 608
 - PROP_MODE_OPTION1, 609
 - PROP_MODE_OPTION2, 609
 - PROP_MODE_OPTION3, 609
 - PROP_MODE_OPTION4, 609
 - PROP_MODE_OPTION5, 609
 - PROP_MODE_OPTION6, 609
 - PROP_MODE_OPTION7, 609
 - PROP_MODE_OPTION8, 609
 - PROP_MODE_OPTION9, 609
 - PROP_READ_OPTION, 608
 - PROP_SILENT, 609
 - PROP_SUBPLUGINS, 609
 - sink_factory, 629
 - src_factory, 630
- gsttensor_decoder.h
 - GST_IS_TENSOR_DECODER, 632
 - GST_IS_TENSOR_DECODER_CLASS, 632
 - GST_TENSOR_DECODER, 632
 - GST_TENSOR_DECODER_CAST, 633
 - GST_TENSOR_DECODER_CLASS, 633
 - gst_tensordec_get_type, 634
 - GST_TYPE_TENSOR_DECODER, 633
 - GstTensorDecoder, 633
 - GstTensorDecoderClass, 634
 - TensorDecMaxOpNum, 633
- gsttensor_demux.c
 - CAPS_STRING_SINK, 636
 - CAPS_STRING_SRC, 636
 - G_DEFINE_TYPE, 637
 - GST_CAT_DEFAULT, 637
 - GST_DEBUG_CATEGORY_STATIC, 638
 - gst_tensor_demux_chain, 638
 - gst_tensor_demux_change_state, 639
 - gst_tensor_demux_class_init, 640
 - gst_tensor_demux_combine_flows, 640
 - gst_tensor_demux_dispose, 641
 - gst_tensor_demux_event, 642
 - gst_tensor_demux_get_property, 642
 - gst_tensor_demux_get_tensor_config, 643
 - gst_tensor_demux_get_tensor_pad, 644
 - gst_tensor_demux_init, 645
 - gst_tensor_demux_parent_class, 637
 - gst_tensor_demux_parse_caps, 646
 - gst_tensor_demux_remove_src_pads, 647
 - gst_tensor_demux_set_property, 648
 - PROP_0, 637
 - PROP_SILENT, 637
 - PROP_TENSORPICK, 637
 - sink_tmpl, 648
 - src_tmpl, 648
- gsttensor_demux.h
 - GST_IS_TENSOR_DEMUX, 651
 - GST_IS_TENSOR_DEMUX_CLASS, 651
 - GST_TENSOR_DEMUX, 651
 - GST_TENSOR_DEMUX_CAST, 651
 - GST_TENSOR_DEMUX_CLASS, 651
 - GST_TENSOR_DEMUX_GET_CLASS, 651
 - gst_tensor_demux_get_type, 652
 - GST_TYPE_TENSOR_DEMUX, 652
 - GstTensorDemux, 652

- GstTensorDemuxClass, 652
- gsttensor_if.c
 - CAPS_STRING, 656
 - cv, 676
 - data, 677
 - DBG, 656
 - G_DEFINE_TYPE, 658
 - GST_CAT_DEFAULT, 656
 - GST_DEBUG_CATEGORY_STATIC, 658
 - gst_tensor_if_act_get_type, 658
 - gst_tensor_if_calculate_cv, 658
 - gst_tensor_if_chain, 659
 - gst_tensor_if_check_condition, 660
 - gst_tensor_if_class_init, 661
 - gst_tensor_if_combine_flows, 662
 - gst_tensor_if_configure_custom_prop, 663
 - gst_tensor_if_cv_get_type, 663
 - gst_tensor_if_dispose, 663
 - gst_tensor_if_event, 664
 - gst_tensor_if_get_property, 665
 - gst_tensor_if_get_property_supplied_value, 666
 - gst_tensor_if_get_tensor_average, 666
 - gst_tensor_if_get_tensor_pad, 667
 - gst_tensor_if_init, 668
 - gst_tensor_if_install_properties, 669
 - gst_tensor_if_op_get_type, 670
 - gst_tensor_if_parent_class, 656
 - gst_tensor_if_parse_caps, 670
 - gst_tensor_if_property_to_string, 671
 - gst_tensor_if_remove_src_pads, 672
 - gst_tensor_if_set_property, 672
 - gst_tensor_if_set_property_cv_option, 673
 - gst_tensor_if_set_property_glist, 674
 - gst_tensor_if_set_property_supplied_value, 674
 - GST_TYPE_TENSOR_IF_ACT, 657
 - GST_TYPE_TENSOR_IF_CV, 657
 - GST_TYPE_TENSOR_IF_OP, 657
 - nnstreamer_if_custom_register, 675
 - nnstreamer_if_custom_unregister, 675
 - operator_func, 657
 - PROP_0, 658
 - PROP_CV, 658
 - PROP_CV_OPTION, 658
 - PROP_ELSE, 658
 - PROP_ELSE_OPTION, 658
 - PROP_OP, 658
 - PROP_SILENT, 658
 - PROP_SV, 658
 - PROP_THEN, 658
 - PROP_THEN_OPTION, 658
 - result, 677
 - sink_factory, 677
 - src_factory, 677
 - svtc_1, 677
 - svtc_2, 678
 - switch, 676
 - TIFOP_NE, 678
 - TRUE, 678
 - type, 678
- gsttensor_if.h
 - GST_IS_TENSOR_IF, 681
 - GST_IS_TENSOR_IF_CLASS, 681
 - GST_TENSOR_IF, 681
 - GST_TENSOR_IF_CAST, 682
 - GST_TENSOR_IF_CLASS, 682
 - GST_TENSOR_IF_GET_CLASS, 682
 - gst_tensor_if_get_type, 685
 - GST_TYPE_TENSOR_IF, 682
 - GstTensorIf, 682
 - GstTensorIfClass, 683
 - tensor_if_behavior, 683
 - tensor_if_compared_value, 683
 - tensor_if_operator, 684
 - tensor_if_srcpads, 684
 - TIFB_END, 683
 - TIFB_FILL_VALUES, 683
 - TIFB_FILL_WITH_FILE, 683
 - TIFB_FILL_WITH_FILE_RPT, 683
 - TIFB_FILL_ZERO, 683
 - TIFB_PASSTHROUGH, 683
 - TIFB_REPEAT_PREVIOUS_FRAME, 683
 - TIFB_SKIP, 683
 - TIFB_TENSORPICK, 683
 - TIFCV_A_VALUE, 684
 - TIFCV_ALL_TENSORS_AVERAGE_VALUE, 684
 - TIFCV_ALL_TENSORS_TOTAL_VALUE, 684
 - TIFCV_CUSTOM, 684
 - TIFCV_END, 684
 - TIFCV_TENSOR_AVERAGE_VALUE, 684
 - TIFCV_TENSOR_TOTAL_VALUE, 684
 - TIFOP_END, 684
 - TIFOP_EQ, 684
 - TIFOP_GE, 684
 - TIFOP_GT, 684
 - TIFOP_LE, 684
 - TIFOP_LT, 684
 - TIFOP_NE, 684
 - TIFOP_NOT_IN_RANGE_EXCLUSIVE, 684
 - TIFOP_NOT_IN_RANGE_INCLUSIVE, 684
 - TIFOP_RANGE_EXCLUSIVE, 684
 - TIFOP_RANGE_INCLUSIVE, 684
 - TIFSP_ELSE_PAD, 685
 - TIFSP_THEN_PAD, 685
- gsttensor_merge.c
 - CAPS_STRING, 687
 - DBG, 687
 - G_DEFINE_TYPE, 688
 - GST_CAT_DEFAULT, 688
 - GST_DEBUG_CATEGORY_STATIC, 689
 - gst_tensor_merge_change_state, 689
 - gst_tensor_merge_class_init, 690
 - gst_tensor_merge_collect_buffer, 690
 - gst_tensor_merge_collected, 691
 - gst_tensor_merge_finalize, 692
 - gst_tensor_merge_generate_mem, 693
 - gst_tensor_merge_get_merged_config, 694

- gst_tensor_merge_get_mode, 695
- gst_tensor_merge_get_property, 695
- gst_tensor_merge_init, 696
- gst_tensor_merge_linear_string, 702
- gst_tensor_merge_mode_string, 702
- gst_tensor_merge_parent_class, 688
- gst_tensor_merge_ready_to_paused, 697
- gst_tensor_merge_request_new_pad, 697
- gst_tensor_merge_send_segment_event, 698
- gst_tensor_merge_set_option_data, 698
- gst_tensor_merge_set_property, 699
- gst_tensor_merge_set_src_caps, 700
- gst_tensor_merge_sink_event, 701
- gst_tensor_merge_src_event, 702
- PROP_0, 688
- PROP_MODE, 688
- PROP_OPTION, 688
- PROP_SILENT, 688
- PROP_SYNC_MODE, 688
- PROP_SYNC_OPTION, 688
- sink_tmpl, 703
- src_tmpl, 703
- gsttensor_merge.h
 - GST_IS_TENSOR_MERGE, 706
 - GST_IS_TENSOR_MERGE_CLASS, 706
 - GST_TENSOR_MERGE, 706
 - GST_TENSOR_MERGE_CAST, 706
 - GST_TENSOR_MERGE_CLASS, 706
 - GST_TENSOR_MERGE_GET_CLASS, 706
 - gst_tensor_merge_get_type, 708
 - GST_TYPE_TENSOR_MERGE, 707
 - GstTensorMerge, 707
 - GstTensorMergeClass, 707
 - GTT_END, 708
 - GTT_LINEAR, 708
 - LINEAR_END, 708
 - LINEAR_FIRST, 708
 - LINEAR_FOURTH, 708
 - LINEAR_SECOND, 708
 - LINEAR_THIRD, 708
 - tensor_merge_linear, 707
 - tensor_merge_linear_mode, 707
 - tensor_merge_mode, 708
- gsttensor_mux.c
 - CAPS_STRING_SINK, 711
 - CAPS_STRING_SRC, 711
 - DBG, 711
 - G_DEFINE_TYPE, 712
 - GST_CAT_DEFAULT, 711
 - GST_DEBUG_CATEGORY_STATIC, 712
 - gst_tensor_mux_chain_flex_tensor, 713
 - gst_tensor_mux_change_state, 714
 - gst_tensor_mux_class_init, 714
 - gst_tensor_mux_collect_buffer, 715
 - gst_tensor_mux_collected, 716
 - gst_tensor_mux_do_clip, 717
 - gst_tensor_mux_finalize, 718
 - gst_tensor_mux_get_property, 719
 - gst_tensor_mux_init, 719
 - gst_tensor_mux_parent_class, 712
 - gst_tensor_mux_ready_to_paused, 720
 - gst_tensor_mux_request_new_pad, 721
 - gst_tensor_mux_send_segment_event, 721
 - gst_tensor_mux_set_property, 722
 - gst_tensor_mux_set_src_caps, 723
 - gst_tensor_mux_set_waiting, 723
 - gst_tensor_mux_sink_event, 724
 - gst_tensor_mux_src_event, 725
 - PROP_0, 712
 - PROP_SILENT, 712
 - PROP_SYNC_MODE, 712
 - PROP_SYNC_OPTION, 712
 - sink_tmpl, 725
 - src_tmpl, 725
- gsttensor_mux.h
 - GST_IS_TENSOR_MUX, 728
 - GST_IS_TENSOR_MUX_CLASS, 728
 - GST_TENSOR_MUX, 728
 - GST_TENSOR_MUX_CAST, 728
 - GST_TENSOR_MUX_CLASS, 728
 - GST_TENSOR_MUX_GET_CLASS, 728
 - gst_tensor_mux_get_type, 729
 - GST_TYPE_TENSOR_MUX, 729
 - GstTensorMux, 729
 - GstTensorMuxClass, 729
- gsttensor_rate.c
 - ABSDIFF, 732
 - CAPS_STRING, 732
 - DBG, 732
 - DEFAULT_SILENT, 733
 - DEFAULT_THROTTLE, 733
 - G_DEFINE_TYPE, 735
 - GST_CAT_DEFAULT, 733
 - GST_DEBUG_CATEGORY_STATIC, 735
 - gst_tensor_rate_class_init, 735
 - gst_tensor_rate_finalize, 736
 - gst_tensor_rate_fixate_caps, 736
 - gst_tensor_rate_flush_prev, 737
 - gst_tensor_rate_get_property, 737
 - gst_tensor_rate_init, 738
 - gst_tensor_rate_install_properties, 738
 - gst_tensor_rate_notify_drop, 739
 - gst_tensor_rate_notify_duplicate, 739
 - gst_tensor_rate_parent_class, 733
 - gst_tensor_rate_push_buffer, 740
 - gst_tensor_rate_reset, 741
 - GST_TENSOR_RATE_SCALED_TIME, 734
 - gst_tensor_rate_send_qos_throttle, 741
 - gst_tensor_rate_set_caps, 742
 - gst_tensor_rate_set_property, 742
 - gst_tensor_rate_sink_event, 743
 - gst_tensor_rate_start, 744
 - gst_tensor_rate_stop, 745
 - gst_tensor_rate_swap_prev, 745
 - gst_tensor_rate_transform_caps, 746
 - gst_tensor_rate_transform_ip, 746

- MAGIC_LIMIT, 734
- PROP_0, 735
- PROP_DROP, 735
- PROP_DUP, 735
- PROP_FRAMERATE, 735
- PROP_IN, 735
- PROP_OUT, 735
- PROP_SILENT, 735
- PROP_THROTTLE, 735
- pspec_drop, 747
- pspec_duplicate, 747
- sink_factory, 748
- src_factory, 748
- THROTTLE_DELAY_RATIO, 734
- gsttensor_rate.h
 - GST_IS_TENSOR_RATE, 750
 - GST_IS_TENSOR_RATE_CLASS, 750
 - GST_TENSOR_RATE, 751
 - GST_TENSOR_RATE_CAST, 751
 - GST_TENSOR_RATE_CLASS, 751
 - GST_TENSOR_RATE_GET_CLASS, 751
 - gst_tensor_rate_get_type, 752
 - GST_TYPE_TENSOR_RATE, 751
 - GstTensorRate, 752
 - GstTensorRateClass, 752
- gsttensor_repo.c
 - _repo, 763
 - DBG, 754
 - GST_REPO_BROADCAST, 754
 - GST_REPO_LOCK, 754
 - GST_REPO_UNLOCK, 755
 - GST_REPO_WAIT, 755
 - gst_tensor_repo_add_repodata, 755
 - gst_tensor_repo_check_changed, 756
 - gst_tensor_repo_check_eos, 756
 - gst_tensor_repo_get_buffer, 757
 - gst_tensor_repo_get_repodata, 758
 - gst_tensor_repo_init, 758
 - gst_tensor_repo_release_repodata, 759
 - gst_tensor_repo_remove_repodata, 760
 - gst_tensor_repo_set_buffer, 760
 - gst_tensor_repo_set_changed, 761
 - gst_tensor_repo_set_eos, 762
 - gst_tensor_repo_wait, 762
- gsttensor_repo.h
 - gst_tensor_repo_add_repodata, 765
 - gst_tensor_repo_check_changed, 766
 - gst_tensor_repo_check_eos, 766
 - gst_tensor_repo_get_buffer, 767
 - gst_tensor_repo_get_repodata, 768
 - gst_tensor_repo_init, 768
 - gst_tensor_repo_remove_repodata, 769
 - gst_tensor_repo_set_buffer, 769
 - gst_tensor_repo_set_changed, 770
 - gst_tensor_repo_set_eos, 771
 - gst_tensor_repo_wait, 772
- gsttensor_reposink.c
 - DEFAULT_INDEX, 774
 - DEFAULT_SIGNAL_RATE, 774
 - DEFAULT_SILENT, 775
 - G_DEFINE_TYPE, 776
 - GST_CAT_DEFAULT, 775
 - GST_DEBUG_CATEGORY_STATIC, 776
 - gst_tensor_reposink_class_init, 776
 - gst_tensor_reposink_dispose, 776
 - gst_tensor_reposink_event, 777
 - gst_tensor_reposink_get_caps, 777
 - gst_tensor_reposink_get_property, 778
 - gst_tensor_reposink_init, 778
 - gst_tensor_reposink_parent_class, 775
 - gst_tensor_reposink_query, 779
 - gst_tensor_reposink_render, 779
 - gst_tensor_reposink_render_buffer, 780
 - gst_tensor_reposink_render_list, 781
 - gst_tensor_reposink_set_caps, 781
 - gst_tensor_reposink_set_property, 782
 - gst_tensor_reposink_start, 783
 - gst_tensor_reposink_stop, 783
 - PROP_0, 775
 - PROP_SIGNAL_RATE, 775
 - PROP_SILENT, 775
 - PROP_SLOT, 775
- gsttensor_reposink.h
 - GST_IS_TENSOR_REPOSINK, 786
 - GST_IS_TENSOR_REPOSINK_CLASS, 786
 - GST_TENSOR_REPOSINK, 786
 - GST_TENSOR_REPOSINK_CLASS, 787
 - gst_tensor_reposink_get_type, 787
 - GST_TYPE_TENSOR_REPOSINK, 787
 - GstTensorRepoSink, 787
 - GstTensorRepoSinkClass, 787
- gsttensor_reposrc.c
 - CAPS_STRING, 790
 - DEFAULT_INDEX, 790
 - DEFAULT_SILENT, 790
 - G_DEFINE_TYPE, 791
 - GST_CAT_DEFAULT, 790
 - GST_DEBUG_CATEGORY_STATIC, 791
 - gst_tensor_reposrc_class_init, 791
 - gst_tensor_reposrc_create, 792
 - gst_tensor_reposrc_dispose, 793
 - gst_tensor_reposrc_gen_dummy_buffer, 793
 - gst_tensor_reposrc_get_property, 794
 - gst_tensor_reposrc_getcaps, 795
 - gst_tensor_reposrc_init, 795
 - gst_tensor_reposrc_parent_class, 790
 - gst_tensor_reposrc_set_property, 796
 - INVALID_INDEX, 790
 - PROP_0, 791
 - PROP_CAPS, 791
 - PROP_SILENT, 791
 - PROP_SLOT_ID, 791
- gsttensor_reposrc.h
 - GST_IS_TENSOR_REPOSRC, 798
 - GST_IS_TENSOR_REPOSRC_CLASS, 798

- GST_TENSOR_REPOSRC, 799
- GST_TENSOR_REPOSRC_CLASS, 799
- gst_tensor_reposrc_get_type, 800
- GST_TYPE_TENSOR_REPOSRC, 799
- GstTensorRepoSrc, 799
- GstTensorRepoSrcClass, 799
- gsttensor_sink.c
 - _tensor_sink_signals, 817
 - DBG, 803
 - DEFAULT_EMIT_SIGNAL, 803
 - DEFAULT_QOS, 803
 - DEFAULT_SIGNAL_RATE, 803
 - DEFAULT_SILENT, 803
 - DEFAULT_SYNC, 804
 - G_DEFINE_TYPE, 805
 - GST_CAT_DEFAULT, 804
 - GST_DEBUG_CATEGORY_STATIC, 805
 - gst_tensor_sink_class_init, 805
 - gst_tensor_sink_event, 806
 - gst_tensor_sink_finalize, 807
 - gst_tensor_sink_get_emit_signal, 808
 - gst_tensor_sink_get_last_render_time, 808
 - gst_tensor_sink_get_property, 808
 - gst_tensor_sink_get_signal_rate, 809
 - gst_tensor_sink_get_silent, 810
 - gst_tensor_sink_init, 810
 - gst_tensor_sink_parent_class, 804
 - gst_tensor_sink_query, 810
 - gst_tensor_sink_render, 811
 - gst_tensor_sink_render_buffer, 812
 - gst_tensor_sink_render_list, 813
 - gst_tensor_sink_set_emit_signal, 814
 - gst_tensor_sink_set_last_render_time, 814
 - gst_tensor_sink_set_property, 815
 - gst_tensor_sink_set_signal_rate, 816
 - gst_tensor_sink_set_silent, 816
 - LAST_SIGNAL, 805
 - PROP_0, 805
 - PROP_EMIT_SIGNAL, 805
 - PROP_SIGNAL_RATE, 805
 - PROP_SILENT, 805
 - SIGNAL_EOS, 805
 - SIGNAL_NEW_DATA, 805
 - SIGNAL_STREAM_START, 805
 - silent_debug_timestamp, 804
- gsttensor_sink.h
 - GST_IS_TENSOR_SINK, 819
 - GST_IS_TENSOR_SINK_CLASS, 819
 - GST_TENSOR_SINK, 819
 - GST_TENSOR_SINK_CLASS, 819
 - gst_tensor_sink_get_type, 820
 - GST_TYPE_TENSOR_SINK, 819
 - GstTensorSink, 820
 - GstTensorSinkClass, 820
- gsttensor_sparsedec.c
 - DBG, 822
 - DEFAULT_SILENT, 822
 - G_DEFINE_TYPE, 823
 - GST_CAT_DEFAULT, 823
 - GST_DEBUG_CATEGORY_STATIC, 824
 - gst_tensor_sparse_dec_chain, 824
 - gst_tensor_sparse_dec_class_init, 825
 - gst_tensor_sparse_dec_finalize, 825
 - gst_tensor_sparse_dec_get_property, 826
 - gst_tensor_sparse_dec_init, 826
 - gst_tensor_sparse_dec_parent_class, 823
 - gst_tensor_sparse_dec_query_caps, 827
 - gst_tensor_sparse_dec_set_property, 828
 - gst_tensor_sparse_dec_sink_event, 828
 - gst_tensor_sparse_dec_sink_query, 829
 - PROP_0, 823
 - PROP_SILENT, 823
 - sink_template, 830
 - src_template, 830
- gsttensor_sparsedec.h
 - GST_IS_TENSOR_SPARSE_DEC, 832
 - GST_IS_TENSOR_SPARSE_DEC_CLASS, 832
 - GST_TENSOR_SPARSE_DEC, 832
 - GST_TENSOR_SPARSE_DEC_CLASS, 833
 - gst_tensor_sparse_dec_get_type, 833
 - GST_TYPE_TENSOR_SPARSE_DEC, 833
 - GstTensorSparseDec, 833
 - GstTensorSparseDecClass, 833
- gsttensor_sparseenc.c
 - DBG, 836
 - DEFAULT_SILENT, 836
 - G_DEFINE_TYPE, 837
 - GST_CAT_DEFAULT, 836
 - GST_DEBUG_CATEGORY_STATIC, 837
 - gst_tensor_sparse_enc_chain, 837
 - gst_tensor_sparse_enc_class_init, 838
 - gst_tensor_sparse_enc_finalize, 839
 - gst_tensor_sparse_enc_get_property, 839
 - gst_tensor_sparse_enc_init, 840
 - gst_tensor_sparse_enc_parent_class, 836
 - gst_tensor_sparse_enc_parse_caps, 841
 - gst_tensor_sparse_enc_query_caps, 841
 - gst_tensor_sparse_enc_set_property, 842
 - gst_tensor_sparse_enc_sink_event, 842
 - gst_tensor_sparse_enc_sink_query, 843
 - PROP_0, 837
 - PROP_SILENT, 837
 - sink_template, 844
 - src_template, 844
- gsttensor_sparseenc.h
 - GST_IS_TENSOR_SPARSE_ENC, 846
 - GST_IS_TENSOR_SPARSE_ENC_CLASS, 846
 - GST_TENSOR_SPARSE_ENC, 846
 - GST_TENSOR_SPARSE_ENC_CLASS, 847
 - gst_tensor_sparse_enc_get_type, 847
 - GST_TYPE_TENSOR_SPARSE_ENC, 847
 - GstTensorSparseEnc, 847
 - GstTensorSparseEncClass, 847
- gsttensor_sparseutil.c
 - gst_tensor_sparse_from_dense, 849
 - gst_tensor_sparse_to_dense, 850

- gsttensor_sparseutil.h
 - gst_tensor_sparse_from_dense, 852
 - gst_tensor_sparse_to_dense, 853
- gsttensor_split.c
 - _clear_tensorseg, 857
 - CAPS_STRING, 856
 - G_DEFINE_TYPE, 858
 - GST_CAT_DEFAULT, 856
 - GST_DEBUG_CATEGORY_STATIC, 858
 - gst_tensor_split_chain, 858
 - gst_tensor_split_change_state, 859
 - gst_tensor_split_class_init, 860
 - gst_tensor_split_combine_flows, 860
 - gst_tensor_split_event, 861
 - gst_tensor_split_finalize, 862
 - gst_tensor_split_get_capsparam, 862
 - gst_tensor_split_get_property, 863
 - gst_tensor_split_get_splitted, 864
 - gst_tensor_split_get_tensor_pad, 865
 - gst_tensor_split_init, 866
 - gst_tensor_split_parent_class, 856
 - gst_tensor_split_remove_src_pads, 867
 - gst_tensor_split_set_property, 868
 - PROP_0, 857
 - PROP_SILENT, 857
 - PROP_TENSORPICK, 857
 - PROP_TENSORSEG, 857
 - sink_tmpl, 868
 - src_tmpl, 868
- gsttensor_split.h
 - GST_IS_TENSOR_SPLIT, 871
 - GST_IS_TENSOR_SPLIT_CLASS, 871
 - GST_TENSOR_SPLIT, 871
 - GST_TENSOR_SPLIT_CAST, 871
 - GST_TENSOR_SPLIT_CLASS, 871
 - GST_TENSOR_SPLIT_GET_CLASS, 871
 - gst_tensor_split_get_type, 872
 - GST_TYPE_TENSOR_SPLIT, 872
 - GstTensorSplit, 872
 - GstTensorSplitClass, 872
- gsttensor_srcio.c
 - AVAIL_FREQUENCY_FILE, 877
 - BLOCKSIZE, 878
 - BUFFER, 878
 - CHANNELS, 878
 - CHANNELS_ENABLED_ALL_CHAR, 878
 - CHANNELS_ENABLED_AUTO_CHAR, 878
 - CURRENT_TRIGGER, 878
 - DBG, 879
 - DEFAULT_BUFFER_CAPACITY, 879
 - DEFAULT_FREQUENCY, 879
 - DEFAULT_MERGE_CHANNELS, 880
 - DEFAULT_OPERATING_CHANNELS_ENABLED, 880
 - DEFAULT_OPERATING_MODE, 880
 - DEFAULT_POLL_TIMEOUT, 880
 - DEFAULT_PROP_BASE_DIRECTORY, 880
 - DEFAULT_PROP_DEV_DIRECTORY, 880
 - DEFAULT_PROP_DEVICE_NUM, 881
 - DEFAULT_PROP_SILENT, 881
 - DEFAULT_PROP_STRING, 881
 - DEFAULT_PROP_TRIGGER_NUM, 881
 - DEVICE, 881
 - DEVICE_PREFIX, 882
 - else, 914
 - EN_SUFFIX, 882
 - g_assert, 887
 - G_DEFINE_TYPE, 888
 - GST_CAT_DEFAULT, 882
 - GST_DEBUG_CATEGORY_STATIC, 889
 - gst_tensor_channel_list_filter_enabled, 889
 - gst_tensor_channel_list_sort_cmp, 889
 - gst_tensor_get_size_from_channels, 889
 - gst_tensor_set_all_channels, 890
 - gst_tensor_src_iio_change_state, 890
 - gst_tensor_src_iio_channel_properties_free, 891
 - gst_tensor_src_iio_class_init, 892
 - gst_tensor_src_iio_create, 892
 - gst_tensor_src_iio_create_config, 893
 - gst_tensor_src_iio_device_properties_init, 894
 - gst_tensor_src_iio_event, 894
 - gst_tensor_src_iio_fill, 895
 - gst_tensor_src_iio_finalize, 896
 - gst_tensor_src_iio_fixate, 896
 - gst_tensor_src_iio_get_all_channel_info, 897
 - gst_tensor_src_iio_get_available_frequency, 898
 - gst_tensor_src_iio_get_caps, 898
 - gst_tensor_src_iio_get_float_from_file, 899
 - gst_tensor_src_iio_get_generic_name, 900
 - gst_tensor_src_iio_get_id_by_name, 900
 - gst_tensor_src_iio_get_name_by_id, 900
 - gst_tensor_src_iio_get_property, 901
 - gst_tensor_src_iio_get_times, 902
 - gst_tensor_src_iio_init, 902
 - gst_tensor_src_iio_is_seekable, 903
 - gst_tensor_src_iio_parent_class, 882
 - gst_tensor_src_iio_process_scanned_data, 903
 - gst_tensor_src_iio_set_caps, 904
 - gst_tensor_src_iio_set_channel_type, 904
 - gst_tensor_src_iio_set_property, 905
 - gst_tensor_src_iio_setup_device_buffer, 906
 - gst_tensor_src_iio_setup_device_properties, 906
 - gst_tensor_src_iio_setup_sampling_frequency, 907
 - gst_tensor_src_iio_setup_scan_channels, 907
 - gst_tensor_src_iio_setup_trigger_properties, 908
 - gst_tensor_src_iio_start, 909
 - gst_tensor_src_iio_stop, 909
 - gst_tensor_src_merge_tensor_by_type, 910
 - gst_tensor_src_restore_iio_device, 911
 - gst_tensor_write_sysfs_int, 912
 - gst_tensor_write_sysfs_string, 913
 - if, 913
 - IIO, 882
 - INDEX_SUFFIX, 883
 - MAX_BUFFER_CAPACITY, 883

- MAX_FREQUENCY, 883
- MAX_POLL_TIMEOUT, 883
- MIN_BUFFER_CAPACITY, 883
- MIN_FREQUENCY, 883
- MIN_POLL_TIMEOUT, 884
- MODE_CONTINUOUS, 884
- MODE_ONE_SHOT, 884
- NAME_FILE, 884
- OFFSET_SUFFIX, 884
- PROCESS_SCANNED_DATA, 885, 913, 914
- prop, 914
- PROP_0, 886
- PROP_BASE_DIRECTORY, 886
- PROP_BUFFER_CAPACITY, 886
- PROP_CHANNELS, 886
- PROP_DEV_DIRECTORY, 886
- PROP_DEVICE, 886
- PROP_DEVICE_NUM, 886
- PROP_FREQUENCY, 886
- PROP_MERGE_CHANNELS, 886
- PROP_MODE, 886
- PROP_POLL_TIMEOUT, 886
- PROP_SILENT, 886
- PROP_TRIGGER, 886
- PROP_TRIGGER_NUM, 886
- SAMPLING_FREQUENCY, 885
- SCALE_SUFFIX, 885
- TIMESTAMP, 885
- TRIGGER, 885
- TRIGGER_PREFIX, 886
- TYPE_SUFFIX, 886
- value_float, 915
- value_unsigned, 915
- gsttensor_srcio.h
 - CHANNELS_ENABLED_ALL, 919
 - CHANNELS_ENABLED_AUTO, 919
 - CHANNELS_ENABLED_CUSTOM, 919
 - channels_enabled_options, 919
 - GST_IS_TENSOR_SRC_IIO, 917
 - GST_IS_TENSOR_SRC_IIO_CLASS, 917
 - GST_TENSOR_SRC_IIO, 918
 - GST_TENSOR_SRC_IIO_CAST, 918
 - GST_TENSOR_SRC_IIO_CLASS, 918
 - gst_tensor_src_iio_get_type, 920
 - GST_TYPE_TENSOR_SRC_IIO, 918
 - GstTensorSrcIIO, 918
 - GstTensorSrcIIOChannelProperties, 919
 - GstTensorSrcIIOClass, 919
 - GstTensorSrcIIODeviceProperties, 919
- gsttensor_trainer.c
 - DEFAULT_PROP_EPOCHS, 923
 - DEFAULT_PROP_INPUT_LIST, 923
 - DEFAULT_PROP_LABEL_LIST, 923
 - DEFAULT_PROP_TRAIN_SAMPLES, 924
 - DEFAULT_PROP_VALID_SAMPLES, 924
 - DEFAULT_STR_PROP_VALUE, 924
 - G_DEFINE_TYPE, 926
 - GST_CAT_DEFAULT, 924
 - GST_DEBUG_CATEGORY_STATIC, 926
 - gst_tensor_trainer_chain, 926
 - gst_tensor_trainer_change_state, 927
 - gst_tensor_trainer_check_buffer_drop_conditions, 928
 - gst_tensor_trainer_check_chain_conditions, 928
 - gst_tensor_trainer_check_invalid_param, 929
 - gst_tensor_trainer_class_init, 929
 - gst_tensor_trainer_convert_meta, 930
 - gst_tensor_trainer_create_event_notifier, 931
 - gst_tensor_trainer_create_framework, 931
 - gst_tensor_trainer_create_model, 931
 - gst_tensor_trainer_create_output, 932
 - gst_tensor_trainer_dummy_data_generation_func, 933
 - gst_tensor_trainer_epochs_is_complete, 933
 - gst_tensor_trainer_finalize, 934
 - gst_tensor_trainer_find_framework, 935
 - gst_tensor_trainer_get_model_stats, 935
 - gst_tensor_trainer_get_property, 936
 - gst_tensor_trainer_get_tensor_size, 936
 - gst_tensor_trainer_init, 937
 - gst_tensor_trainer_parent_class, 924
 - gst_tensor_trainer_push_input, 938
 - gst_tensor_trainer_query_caps, 939
 - gst_tensor_trainer_set_model_load_path, 939
 - gst_tensor_trainer_set_model_save_path, 940
 - gst_tensor_trainer_set_output_meta, 941
 - gst_tensor_trainer_set_prop_framework, 941
 - gst_tensor_trainer_set_prop_model_config_file_path, 942
 - gst_tensor_trainer_set_property, 943
 - gst_tensor_trainer_sink_event, 944
 - gst_tensor_trainer_sink_query, 945
 - gst_tensor_trainer_src_query, 946
 - gst_tensor_trainer_start_model_training, 947
 - gst_tensor_trainer_stop_model_training, 947
 - gst_tensor_trainer_wait_for_epoch_completion, 947
 - gst_tensor_trainer_wait_for_training_completion, 948
 - MODEL_STATS_SIZE, 924
 - nnstreamer_trainer_exit, 948
 - nnstreamer_trainer_notify_event, 949
 - nnstreamer_trainer_probe, 949
 - PROP_0, 926
 - PROP_EPOCHS, 926
 - PROP_FRAMEWORK, 926
 - PROP_MODEL_CONFIG, 926
 - PROP_MODEL_LOAD_PATH, 926
 - PROP_MODEL_SAVE_PATH, 926
 - PROP_NUM_INPUTS, 926
 - PROP_NUM_LABELS, 926
 - PROP_NUM_TRAINING_SAMPLES, 926
 - PROP_NUM_VALIDATION_SAMPLES, 926
 - SINK_CAPS_STRING, 925
 - sink_template, 950
 - SRC_CAPS_STRING, 925

- src_template, [950](#)
- TRAINING_ACCURACY, [925](#)
- TRAINING_LOSS, [925](#)
- VALIDATION_ACCURACY, [925](#)
- VALIDATION_LOSS, [925](#)
- gsttensor_trainer.h
 - GST_IS_TENSOR_TRAINER, [952](#)
 - GST_IS_TENSOR_TRAINER_CLASS, [952](#)
 - GST_TENSOR_TRAINER, [952](#)
 - GST_TENSOR_TRAINER_CLASS, [953](#)
 - gst_tensor_trainer_get_type, [953](#)
 - GST_TYPE_TENSOR_TRAINER, [953](#)
 - GstTensorTrainer, [953](#)
 - GstTensorTrainerClass, [953](#)
- gsttensor_transform.c
 - __pad0__, [985](#)
 - _conv_from_f16, [957](#)
 - _conv_to_f16, [957](#)
 - _op_float16, [957](#)
 - break, [985](#)
 - CAPS_STRING, [958](#)
 - DBG, [958](#)
 - DEFAULT_ACCELERATION, [958](#)
 - FALSE, [985](#)
 - float16_not_supported, [962](#)
 - G_DEFINE_TYPE, [963](#)
 - GST_CAT_DEFAULT, [958](#)
 - GST_DEBUG_CATEGORY_STATIC, [963](#)
 - gst_tensor_transform_arithmetic, [963](#)
 - gst_tensor_transform_clamp, [964](#)
 - gst_tensor_transform_class_init, [965](#)
 - gst_tensor_transform_convert_dimension, [966](#)
 - gst_tensor_transform_dimchg, [967](#)
 - gst_tensor_transform_finalize, [969](#)
 - gst_tensor_transform_fixate_caps, [969](#)
 - gst_tensor_transform_get_operator, [970](#)
 - gst_tensor_transform_get_property, [971](#)
 - gst_tensor_transform_get_stand_mode, [972](#)
 - gst_tensor_transform_init, [972](#)
 - gst_tensor_transform_mode_get_type, [973](#)
 - gst_tensor_transform_operator_string, [986](#)
 - gst_tensor_transform_padding, [973](#)
 - gst_tensor_transform_parent_class, [959](#)
 - gst_tensor_transform_read_caps, [974](#)
 - gst_tensor_transform_set_caps, [975](#)
 - gst_tensor_transform_set_option_data, [976](#)
 - gst_tensor_transform_set_property, [977](#)
 - gst_tensor_transform_stand, [978](#)
 - gst_tensor_transform_stand_string, [986](#)
 - gst_tensor_transform_transform, [979](#)
 - gst_tensor_transform_transform_caps, [981](#)
 - gst_tensor_transform_transform_size, [982](#)
 - gst_tensor_transform_transpose, [983](#)
 - gst_tensor_transform_typecast, [984](#)
 - GST_TYPE_TENSOR_TRANSFORM_MODE, [959](#)
 - GTT_OP_MUL, [986](#)
 - handle_operator, [959](#)
 - NNS_TENSOR_PADDING_RANK_LIMIT, [959](#)
 - NNS_TENSOR_TRANSPOSE_RANK_LIMIT, [959](#)
 - operator%d", oper), [986](#)
 - PROP_0, [962](#)
 - PROP_ACCELERATION, [962](#)
 - PROP_APPLY, [962](#)
 - PROP_MODE, [962](#)
 - PROP_OPTION, [962](#)
 - PROP_SILENT, [962](#)
 - PROP_TRANSPOSE_RANK_LIMIT, [962](#)
 - REGEX_ARITH_OPTION, [960](#)
 - REGEX_ARITH_OPTION_TYPECAST, [960](#)
 - REGEX_CLAMP_OPTION, [960](#)
 - REGEX_DIMCHG_OPTION, [960](#)
 - REGEX_PADDING_OPTION, [960](#)
 - REGEX_STAND_OPTION, [961](#)
 - REGEX_TRANSPOSE_OPTION, [961](#)
 - REGEX_TYPECAST_OPTION, [961](#)
 - sink_factory, [986](#)
 - src_factory, [987](#)
 - transposeloop, [961](#)
 - while, [985](#)
- gsttensor_transform.h
 - _tensor_transform_mode, [993](#)
 - GST_IS_TENSOR_TRANSFORM, [990](#)
 - GST_IS_TENSOR_TRANSFORM_CLASS, [990](#)
 - GST_TENSOR_TRANSFORM, [990](#)
 - GST_TENSOR_TRANSFORM_CAST, [990](#)
 - GST_TENSOR_TRANSFORM_CLASS, [991](#)
 - gst_tensor_transform_get_type, [994](#)
 - GST_TYPE_TENSOR_TRANSFORM, [991](#)
 - GstTensorTransform, [991](#)
 - GstTensorTransformClass, [991](#)
 - GTT_ARITHMETIC, [993](#)
 - GTT_CLAMP, [993](#)
 - GTT_DIMCHG, [993](#)
 - GTT_OP_ADD, [993](#)
 - GTT_OP_DIV, [993](#)
 - GTT_OP_MUL, [993](#)
 - GTT_OP_TYPECAST, [993](#)
 - GTT_OP_UNKNOWN, [993](#)
 - GTT_PADDING, [993](#)
 - GTT_STAND, [993](#)
 - GTT_TRANSPOSE, [993](#)
 - GTT_TYPECAST, [993](#)
 - GTT_UNKNOWN, [993](#)
 - PADDING_BACK, [994](#)
 - PADDING_BOTTOM, [994](#)
 - PADDING_END, [994](#)
 - PADDING_FRONT, [994](#)
 - PADDING_LEFT, [994](#)
 - PADDING_RIGHT, [994](#)
 - PADDING_TOP, [994](#)
 - STAND_DC_AVERAGE, [994](#)
 - STAND_DEFAULT, [994](#)
 - STAND_END, [994](#)
 - tensor_transform_arithmetic, [991](#)
 - tensor_transform_clamp, [992](#)

- tensor_transform_dimchg, 992
- tensor_transform_mode, 992
- tensor_transform_operator, 993
- tensor_transform_padding, 992
- tensor_transform_padding_axis, 994
- tensor_transform_stand, 992
- tensor_transform_stand_mode, 994
- tensor_transform_transpose, 992
- tensor_transform_typecast, 993
- GstTensorAggregator
 - gsttensor_aggregator.h, 514
- GstTensorAggregatorClass
 - gsttensor_aggregator.h, 514
- GstTensorAllocator, 253
 - parent, 254
- GstTensorAllocatorClass, 254
 - parent_class, 254
- GstTensorCollectPadData, 255
 - buffer, 255
 - collect, 255
- GstTensorConverter
 - gsttensor_converter.h, 556
- GstTensorConverterClass
 - gsttensor_converter.h, 556
- GstTensorCrop
 - gsttensor_crop.h, 588
- GstTensorCropClass
 - gsttensor_crop.h, 588
- GstTensorCropPadData, 256
 - config, 256
 - data, 257
- GstTensorDebug
 - gsttensor_debug.h, 602
- GstTensorDebugClass
 - gsttensor_debug.h, 602
- GstTensorDecoder
 - gsttensor_decoder.h, 633
- GstTensorDecoderClass
 - gsttensor_decoder.h, 634
- GstTensorDecoderDef
 - nnstreamer_plugin_api_decoder.h, 1022
- GstTensorDemux
 - gsttensor_demux.h, 652
- GstTensorDemuxClass
 - gsttensor_demux.h, 652
- GstTensorExtraInfo, 257
 - infos, 258
 - magic, 258
 - num_extra_tensors, 258
 - reserved, 258
 - version, 258
- GstTensorFilter
 - tensor_filter.h, 1435
- GstTensorFilterClass
 - tensor_filter.h, 1435
- GstTensorFilterCombination
 - tensor_filter_common.h, 1501
- GstTensorFilterFramework
 - nnstreamer_plugin_api_filter.h, 1033
- GstTensorFilterFrameworkEventData
 - nnstreamer_plugin_api_filter.h, 1033
- GstTensorFilterFrameworkInfo
 - nnstreamer_plugin_api_filter.h, 1033
- GstTensorFilterFrameworkStatistics
 - nnstreamer_plugin_api_filter.h, 1033
- GstTensorFilterPrivate
 - tensor_filter_common.h, 1501
- GstTensorFilterProperties
 - nnstreamer_plugin_api_filter.h, 1033
- GstTensorFilterSharedModelRepresentation, 259
 - referred_list, 259
 - shared_interpreter, 259
- GstTensorFilterStatistics
 - tensor_filter_common.h, 1501
- GstTensorIf
 - gsttensor_if.h, 682
- GstTensorIfClass
 - gsttensor_if.h, 683
- GstTensorInfo, 260
 - dimension, 260
 - name, 260
 - type, 261
- GstTensorMemory, 261
 - data, 261
 - size, 262
- GstTensorMerge
 - gsttensor_merge.h, 707
- GstTensorMergeClass
 - gsttensor_merge.h, 707
- GstTensorMetaInfo, 262
 - dimension, 263
 - format, 263
 - magic, 264
 - media_type, 264
 - sparse_info, 264
 - type, 264
 - version, 264
- GstTensorMux
 - gsttensor_mux.h, 729
- GstTensorMuxClass
 - gsttensor_mux.h, 729
- GstTensorPad, 265
 - last_ret, 265
 - last_ts, 265
 - nth, 265
 - pad, 265
- GstTensorQueryClient
 - tensor_query_client.h, 1578
- GstTensorQueryClientClass
 - tensor_query_client.h, 1578
- GstTensorQueryEdgeInfo, 266
 - cb, 266
 - dest_host, 266
 - dest_port, 267
 - host, 267
 - pdata, 267

- port, [267](#)
- topic, [267](#)
- GstTensorQueryServer, [268](#)
 - cond, [268](#)
 - configured, [268](#)
 - edge_h, [268](#)
 - id, [268](#)
 - lock, [269](#)
- GstTensorQueryServerSink
 - tensor_query_serversink.h, [1611](#)
- GstTensorQueryServerSinkClass
 - tensor_query_serversink.h, [1611](#)
- GstTensorQueryServerSrc
 - tensor_query_serversrc.h, [1626](#)
- GstTensorQueryServerSrcClass
 - tensor_query_serversrc.h, [1627](#)
- GstTensorRate
 - gsttensor_rate.h, [752](#)
- GstTensorRateClass
 - gsttensor_rate.h, [752](#)
- GstTensorRepo, [269](#)
 - hash, [269](#)
 - initialized, [270](#)
 - num_data, [270](#)
 - repo_cond, [270](#)
 - repo_lock, [270](#)
- GstTensorRepoData, [270](#)
 - buffer, [271](#)
 - caps, [271](#)
 - cond_pull, [271](#)
 - cond_push, [271](#)
 - eos, [272](#)
 - lock, [272](#)
 - pushed, [272](#)
 - sink_changed, [272](#)
 - sink_id, [272](#)
 - src_changed, [272](#)
 - src_id, [273](#)
- GstTensorRepoSink
 - gsttensor_reposink.h, [787](#)
- GstTensorRepoSinkClass
 - gsttensor_reposink.h, [787](#)
- GstTensorRepoSrc
 - gsttensor_reposrc.h, [799](#)
- GstTensorRepoSrcClass
 - gsttensor_reposrc.h, [799](#)
- GstTensorsConfig, [273](#)
 - info, [274](#)
 - rate_d, [274](#)
 - rate_n, [274](#)
- GstTensorsInfo, [275](#)
 - extra, [275](#)
 - format, [275](#)
 - info, [276](#)
 - num_tensors, [276](#)
- GstTensorSink
 - gsttensor_sink.h, [820](#)
- GstTensorSinkClass
 - gsttensor_sink.h, [820](#)
- GstTensorSparseDec
 - gsttensor_sparsedec.h, [833](#)
- GstTensorSparseDecClass
 - gsttensor_sparsedec.h, [833](#)
- GstTensorSparseEnc
 - gsttensor_sparseenc.h, [847](#)
- GstTensorSparseEncClass
 - gsttensor_sparseenc.h, [847](#)
- GstTensorSplit
 - gsttensor_split.h, [872](#)
- GstTensorSplitClass
 - gsttensor_split.h, [872](#)
- GstTensorSrcIO
 - gsttensor_srcio.h, [918](#)
- GstTensorSrcIOChannelProperties
 - gsttensor_srcio.h, [919](#)
- GstTensorSrcIOClass
 - gsttensor_srcio.h, [919](#)
- GstTensorSrcIODeviceProperties
 - gsttensor_srcio.h, [919](#)
- GstTensorTrainer
 - gsttensor_trainer.h, [953](#)
- GstTensorTrainerClass
 - gsttensor_trainer.h, [953](#)
- GstTensorTrainerEventNotifier
 - nnstreamer_plugin_api_trainer.h, [1045](#)
- GstTensorTrainerEventType
 - nnstreamer_plugin_api_trainer.h, [1045](#)
- GstTensorTrainerFramework
 - nnstreamer_plugin_api_trainer.h, [1045](#)
- GstTensorTrainerFrameworkInfo
 - nnstreamer_plugin_api_trainer.h, [1045](#)
- GstTensorTrainerProperties
 - nnstreamer_plugin_api_trainer.h, [1045](#)
- GstTensorTransform
 - gsttensor_transform.h, [991](#)
- GstTensorTransformClass
 - gsttensor_transform.h, [991](#)
- GstVideoFormat
 - gsttensor_converter_media_no_video.h, [568](#)
- GstVideoInfo
 - gsttensor_converter_media_no_video.h, [567](#)
- GTensorFilterSingle
 - tensor_filter_single.h, [1550](#)
- GTensorFilterSingleClass
 - tensor_filter_single.h, [1550](#)
- GTensorFilterSinglePrivate
 - tensor_filter_single.c, [1537](#)
- GTT_ARITHMETIC
 - gsttensor_transform.h, [993](#)
- GTT_CLAMP
 - gsttensor_transform.h, [993](#)
- GTT_DIMCHG
 - gsttensor_transform.h, [993](#)
- GTT_END
 - gsttensor_merge.h, [708](#)
- GTT_LINEAR

- gsttensor_merge.h, 708
- GTT_OP_ADD
 - gsttensor_transform.h, 993
- GTT_OP_DIV
 - gsttensor_transform.h, 993
- GTT_OP_MUL
 - gsttensor_transform.c, 986
 - gsttensor_transform.h, 993
- GTT_OP_TYPECAST
 - gsttensor_transform.h, 993
- GTT_OP_UNKNOWN
 - gsttensor_transform.h, 993
- GTT_PADDING
 - gsttensor_transform.h, 993
- GTT_STAND
 - gsttensor_transform.h, 993
- GTT_TRANSPOSE
 - gsttensor_transform.h, 993
- GTT_TYPECAST
 - gsttensor_transform.h, 993
- GTT_UNKNOWN
 - gsttensor_transform.h, 993
- h
 - tensor_region_s, 288
- handle_operator
 - gsttensor_transform.c, 959
- handleEvent
 - _GstTensorFilterFramework, 112
- handles
 - nnstreamer_subplugin.c, 1334
- hash
 - GstTensorRepo, 269
- have_group_id
 - _GstJoin, 54
 - _GstTensorDemux, 98
 - _GstTensorSplit, 183
- have_segment
 - _GstTensorConverter, 80
- host
 - _GstEdgeSink, 48
 - _GstTensorQueryClient, 151
 - _GstTensorQueryServerSrc, 158
 - GstTensorQueryEdgeInfo, 267
- hw
 - _GstTensorFilterFrameworkEventData, 118
- hw_accel.c
 - cpu_neon_accel_available, 996
- hw_accel.h
 - cpu_neon_accel_available, 997
- hw_list
 - _GstTensorFilterFrameworkEventData, 118
 - _GstTensorFilterFrameworkInfo, 121
 - _GstTensorFilterProperties, 130
- id
 - _GstTensorSrcIIODeviceProperties, 197
 - GstTensorQueryServer, 268
- if
 - gsttensor_decoder.c, 623
 - gsttensor_srcio.c, 913
- IIO
 - gsttensor_srcio.c, 882
- IMAGE_CAPS
 - gstdatareposink.c, 302
- in
 - _GstTensorRate, 162
- in_accl
 - parse_accl_args, 277
- in_caps
 - _GstMqttSink, 62
 - _GstTensorRepoSink, 166
- in_caps_str
 - _GstTensorQueryClient, 151
- in_combi
 - _GstTensorFilterCombination, 103
- in_combi_defined
 - _GstTensorFilterCombination, 103
- in_config
 - _GstTensorAggregator, 75
 - _GstTensorFilterPrivate, 125
 - _GstTensorIf, 138
 - _GstTensorSparseDec, 177
 - _GstTensorSparseEnc, 180
 - _GstTensorTrainer, 200
 - _GstTensorTransform, 217
- in_media_type
 - _GstTensorConverter, 80
- index
 - _GstTensorSrcIIOChannelProperties, 193
- INDEX_SUFFIX
 - gsttensor_srcio.c, 883
- info
 - _FilterTransformData, 30
 - _GstTensorFilterFrameworkEventData, 118
 - _GstTensorFilterPrivate, 126
 - GstTensorsConfig, 274
 - GstTensorsInfo, 276
- info_template
 - gsttensor_crop.c, 584
- infos
 - GstTensorExtraInfo, 258
- ini
 - _GstTensorRepoSrc, 170
- init
 - _GstTensorDecoderDef, 96
- init_filter_custom
 - tensor_filter_custom.c, 1524
- init_filter_custom_easy
 - tensor_filter_custom_easy.c, 1531
- init_queryserver
 - tensor_query_server.c, 1590
- init_subplugin
 - nnstreamer_subplugin.c, 1329
- initfunc
 - _NNStreamer_custom_class, 228
- initialized

- GstTensorRepo, 270
- input_configured
 - _GstTensorFilterSingleClass, 221
 - _GstTensorFilterProperties, 130
- input_layout
 - _GstTensorFilterProperties, 130
- input_meta
 - _GstTensorFilterProperties, 131
 - _GstTensorTrainerProperties, 211
- input_ranks
 - _GstTensorFilterProperties, 131
- input_tensors
 - _GstTensorTrainer, 200
- internal_data
 - tensor_filter_custom.c, 1519
 - tensor_filter_custom_easy.c, 1534
- INVALID_INDEX
 - gsttensor_reposrc.c, 790
- invoke
 - _GstTensorFilterSingleClass, 221
 - _GstTensorFilterFramework, 112
 - _NNStreamer_custom_class, 228
- invoke_dynamic
 - _GstTensorFilterProperties, 131
- invoke_NN
 - _GstTensorFilterFramework, 113
- is_audio_supported
 - gsttensor_converter_media_info_audio.h, 559
 - gsttensor_converter_media_no_audio.h, 564
- is_connected
 - _GstEdgeSink, 49
 - _GstMqttSink, 62
 - _GstMqttSrc, 69
- is_custom
 - _GstTensorDecoder, 91
- is_epoch_complete
 - _GstTensorTrainer, 200
- is_flexible
 - _FilterTransformData, 30
- is_live
 - _GstMqttSrc, 69
- is_shuffle
 - _GstDataRepoSrc, 39
- is_signed
 - _GstTensorSrcIOChannelProperties, 193
- is_start
 - _GstDataRepoSrc, 39
- is_static_tensors
 - _GstDataRepoSink, 33
- is_subscribed
 - _GstMqttSrc, 69
- is_tensor
 - _GstTensorQueryClient, 151
 - _GstTensorSrcIO, 189
- is_training_complete
 - _GstTensorTrainer, 200
- is_updatable
 - _GstTensorFilterPrivate, 126
- is_video_supported
 - gsttensor_converter_media_info_video.h, 561
 - gsttensor_converter_media_no_video.h, 567
- join/gstjoin.c, 382
- join/gstjoin.h, 399
- json_filename
 - _GstDataRepoSink, 33
 - _GstDataRepoSrc, 40
- JSON_KEY_MODEL_PATH
 - ml_agent.c, 1158
- json_object
 - _GstDataRepoSink, 33
- last_offset
 - _GstDataRepoSrc, 40
- last_render_time
 - _GstTensorRepoSink, 167
 - _GstTensorSink, 173
- last_ret
 - GstTensorPad, 265
- LAST_SIGNAL
 - gsttensor_sink.c, 805
- last_ts
 - _GstTensorRate, 162
 - GstTensorPad, 265
- latency
 - _GstMqttSrc, 69
 - _GstTensorFilterProperties, 131
- latency_ignore_count
 - _GstTensorFilterStatistics, 134
- latency_mode
 - _GstTensorFilterPrivate, 126
- LATENCY_REPORT_HEADROOM
 - tensor_filter.c, 1403
- LATENCY_REPORT_THRESHOLD
 - tensor_filter.c, 1403
- latency_reported
 - _GstTensorFilterPrivate, 126
- latency_reporting
 - _GstTensorFilterPrivate, 126
- lateness
 - _GstTensorCrop, 85
- latest_invoke_time
 - _GstTensorFilterStatistics, 134
- layout
 - _GstTensorFilterFrameworkEventData, 118
 - _tensor_transform_padding, 243
- li_vn_mode
 - _ntp_packet_t, 234
- LINEAR_END
 - gsttensor_merge.h, 708
- LINEAR_FIRST
 - gsttensor_merge.h, 708
- LINEAR_FOURTH
 - gsttensor_merge.h, 708
- LINEAR_SECOND
 - gsttensor_merge.h, 708
- LINEAR_THIRD

- gsttensor_merge.h, 708
- loaded
 - _GstTensorMerge, 142
 - _GstTensorTransform, 217
 - confdata, 247
- location
 - _GstTensorSrcIIOChannelProperties, 193
- lock
 - _GstEdgeSink, 49
 - _GstJoin, 54
 - _GstTensorIf, 138
 - _NnstWatchdog, 232
 - GstTensorQueryServer, 269
 - GstTensorRepoData, 272
- loop
 - _NnstWatchdog, 232
- magic
 - GstTensorExtraInfo, 258
 - GstTensorMetaInfo, 264
- MAGIC_LIMIT
 - gsttensor_rate.c, 734
- mask
 - _GstTensorSrcIIOChannelProperties, 194
- max
 - _tensor_transform_clamp, 241
- MAX_BUFFER_CAPACITY
 - gsttensor_srcii.c, 883
- MAX_FREQUENCY
 - gsttensor_srcii.c, 883
- MAX_LEN_PROP_NTP_SRVS
 - mqttSink.c, 413
- max_msg_buf_size
 - _GstMqttSink, 62
- MAX_POLL_TIMEOUT
 - gsttensor_srcii.c, 883
- max_request
 - _GstTensorQueryClient, 152
- media_type
 - GstTensorMetaInfo, 264
 - tensor_typedef.h, 1153
- mem
 - _FilterTransformData, 30
- merge_channels_data
 - _GstTensorSrcIIO, 189
- meta
 - _FilterTransformData, 30
 - GstMetaQuery, 252
- meta_mode
 - _GstTensorDebug, 88
- metaless_frame_count
 - _GstTensorQueryServerSink, 155
- metaless_frame_limit
 - _GstTensorQueryServerSink, 155
- methods
 - _internal_data, 225
- min
 - _tensor_transform_clamp, 241
- MIN_BUFFER_CAPACITY
 - gsttensor_srcii.c, 883
- MIN_FREQUENCY
 - gsttensor_srcii.c, 883
- MIN_POLL_TIMEOUT
 - gsttensor_srcii.c, 884
- ml_agent.c
 - JSON_KEY_MODEL_PATH, 1158
 - mlagent_get_model_path_from, 1157
 - URI_KEYWORD_MODEL, 1158
 - URI_SCHEME, 1158
- ml_agent.h
 - mlagent_get_model_path_from, 1160
- ml_log_stacktrace
 - nnstreamer_log.h, 1201
- ml_logd
 - nnstreamer_log.h, 1201
- ml_loge
 - nnstreamer_log.h, 1201
- ml_loge_stacktrace
 - nnstreamer_log.h, 1202
- ml_logf
 - nnstreamer_log.h, 1202
- ml_logf_stacktrace
 - nnstreamer_log.h, 1202
- ml_logi
 - nnstreamer_log.h, 1202
- ml_logw
 - nnstreamer_log.h, 1202
- mlagent_get_model_path_from
 - ml_agent.c, 1157
 - ml_agent.h, 1160
- mode
 - _GstTensorConverter, 80
 - _GstTensorMerge, 142
 - _GstTensorSrcIIO, 190
 - _GstTensorTransform, 217
 - _tensor_time_sync_data, 239
 - _tensor_transform_stand, 244
- MODE_CONTINUOUS
 - gsttensor_srcii.c, 884
- MODE_ONE_SHOT
 - gsttensor_srcii.c, 884
- mode_option
 - _GstTensorConverter, 80
- model
 - runtime_data, 280
- model_config
 - _GstTensorTrainerProperties, 211
- model_files
 - _GstTensorFilterFrameworkEventData, 119
 - _GstTensorFilterProperties, 131
- model_info_ops
 - nnstreamer_plugin_api_filter.h, 1035
- model_load_path
 - _GstTensorTrainerProperties, 211
- model_save_path
 - _GstTensorTrainerProperties, 211
- MODEL_STATS_SIZE

- gsttensor_trainer.c, 924
- modename
 - _GstTensorDecoderDef, 96
- module
 - _internal_data, 225
- mqtt/mqttcommon.h, 403
- mqtt/mqttelements.c, 407
- mqtt/mqttsink.c, 408
- mqtt/mqttsink.h, 443
- mqtt/mqttsrc.c, 448
- mqtt/mqttsrc.h, 480
- mqtt/nputil.c, 484
- mqtt/nputil.h, 488
- mqtt_client_handle
 - _GstMqttSink, 62
 - _GstMqttSrc, 69
- mqtt_client_id
 - _GstMqttSink, 62
 - _GstMqttSrc, 70
- mqtt_conn_opts
 - _GstMqttSink, 63
 - _GstMqttSrc, 70
- MQTT_CONNECT_FAILURE
 - mqttsink.h, 447
- MQTT_CONNECTED
 - mqttsink.h, 447
- MQTT_CONNECTION_LOST
 - mqttsink.h, 447
- MQTT_DISCONNECT_FAILED
 - mqttsink.h, 447
- MQTT_DISCONNECTED
 - mqttsink.h, 447
- mqtt_get_unix_epoch
 - mqttcommon.h, 406
- mqtt_host_address
 - _GstMqttSink, 63
 - _GstMqttSrc, 70
- mqtt_host_port
 - _GstMqttSink, 63
 - _GstMqttSrc, 70
- mqtt_msg_buf
 - _GstMqttSink, 63
- mqtt_msg_buf_size
 - _GstMqttSink, 63
- mqtt_msg_hdr
 - _GstMqttSink, 63
- mqtt_ntp_hnames
 - _GstMqttSink, 64
- mqtt_ntp_num_srvs
 - _GstMqttSink, 64
- mqtt_ntp_ports
 - _GstMqttSink, 64
- mqtt_ntp_srvs
 - _GstMqttSink, 64
- mqtt_ntp_sync
 - _GstMqttSink, 64
- mqtt_pub_wait_timeout
 - _GstMqttSink, 64
- mqtt_qos
 - _GstMqttSink, 65
 - _GstMqttSrc, 70
- mqtt_respn_opts
 - _GstMqttSink, 65
 - _GstMqttSrc, 70
- mqtt_sink_gcond
 - _GstMqttSink, 65
- mqtt_sink_mutex
 - _GstMqttSink, 65
- mqtt_sink_state
 - _GstMqttSink, 65
- mqtt_sink_state_t
 - mqttsink.h, 447
- mqtt_src_gcond
 - _GstMqttSrc, 71
- mqtt_src_mutex
 - _GstMqttSrc, 71
- mqtt_sub_timeout
 - _GstMqttSrc, 71
- mqtt_topic
 - _GstMqttSink, 65
 - _GstMqttSrc, 71
- mqttcommon.h
 - DEFAULT_MQTT_CONN_TIMEOUT_SEC, 404
 - default_mqtt_get_unix_epoch, 406
 - GST_MQTT_ELEM_NAME_SINK, 404
 - GST_MQTT_ELEM_NAME_SRC, 405
 - GST_MQTT_LEN_MSG_HDR, 405
 - GST_MQTT_MAX_LEN_GST_CAPS_STR, 405
 - GST_MQTT_MAX_NUM_MEMS, 405
 - GST_MQTT_PACKAGE, 405
 - GST_US_TO_NS_MULTIPLIER, 405
 - GstMQTTMessageHdr, 406
 - mqtt_get_unix_epoch, 406
 - UNUSED, 406
- mqttelements.c
 - PACKAGE, 408
 - plugin_init, 408
- mqttsink.c
 - _mqtt_set_msg_buf_hdr, 413
 - _put_timestamp_to_msg_buf_hdr, 414
 - cb_mqtt_on_connect, 414
 - cb_mqtt_on_connect_failure, 415
 - cb_mqtt_on_connection_lost, 415
 - cb_mqtt_on_delivery_complete, 416
 - cb_mqtt_on_disconnect, 416
 - cb_mqtt_on_disconnect_failure, 417
 - cb_mqtt_on_message_arrived, 417
 - cb_mqtt_on_send_failure, 418
 - cb_mqtt_on_send_success, 418
 - DEFAULT_DEBUG, 413
 - DEFAULT_MAX_MSG_BUF_SIZE, 413
 - DEFAULT_MQTT_CLIENT_ID, 441
 - DEFAULT_MQTT_CLIENT_ID_FORMAT, 441
 - DEFAULT_MQTT_DISCONNECT_TIMEOUT, 413
 - DEFAULT_MQTT_HOST_ADDRESS, 442
 - DEFAULT_MQTT_HOST_PORT, 442

- DEFAULT_MQTT_NTP_SERVERS, 442
- DEFAULT_MQTT_NTP_SYNC, 413
- DEFAULT_MQTT_OPT_CLEANSSESSION, 413
- DEFAULT_MQTT_OPT_KEEP_ALIVE_INTERVAL, 413
- DEFAULT_MQTT_PUB_TOPIC, 442
- DEFAULT_MQTT_PUB_TOPIC_FORMAT, 442
- DEFAULT_MQTT_PUB_WAIT_TIMEOUT, 413
- DEFAULT_MQTT_QOS, 413
- DEFAULT_NUM_BUFFERS, 413
- DEFAULT_QOS, 413
- DEFAULT_SYNC, 413
- G_DEFINE_TYPE, 419
- GST_CAT_DEFAULT, 412
- GST_DEBUG_CATEGORY_STATIC, 419
- gst_mqtt_sink_change_state, 419
- gst_mqtt_sink_class_finalize, 420
- gst_mqtt_sink_class_init, 421
- gst_mqtt_sink_event, 422
- gst_mqtt_sink_get_client_id, 422
- gst_mqtt_sink_get_debug, 422
- gst_mqtt_sink_get_host_address, 423
- gst_mqtt_sink_get_host_port, 423
- gst_mqtt_sink_get_max_msg_buf_size, 423
- gst_mqtt_sink_get_mqtt_ntp_srvs, 424
- gst_mqtt_sink_get_mqtt_ntp_sync, 424
- gst_mqtt_sink_get_mqtt_qos, 424
- gst_mqtt_sink_get_num_buffers, 425
- gst_mqtt_sink_get_opt_cleansession, 425
- gst_mqtt_sink_get_opt_keep_alive_interval, 425
- gst_mqtt_sink_get_property, 426
- gst_mqtt_sink_get_pub_topic, 428
- gst_mqtt_sink_get_pub_wait_timeout, 428
- gst_mqtt_sink_init, 428
- gst_mqtt_sink_parent_class, 412
- gst_mqtt_sink_query, 429
- gst_mqtt_sink_render, 430
- gst_mqtt_sink_render_list, 430
- gst_mqtt_sink_set_caps, 431
- gst_mqtt_sink_set_client_id, 432
- gst_mqtt_sink_set_debug, 432
- gst_mqtt_sink_set_host_address, 433
- gst_mqtt_sink_set_host_port, 433
- gst_mqtt_sink_set_max_msg_buf_size, 434
- gst_mqtt_sink_set_mqtt_ntp_srvs, 434
- gst_mqtt_sink_set_mqtt_ntp_sync, 435
- gst_mqtt_sink_set_mqtt_qos, 435
- gst_mqtt_sink_set_num_buffers, 436
- gst_mqtt_sink_set_opt_cleansession, 436
- gst_mqtt_sink_set_opt_keep_alive_interval, 436
- gst_mqtt_sink_set_property, 437
- gst_mqtt_sink_set_pub_topic, 438
- gst_mqtt_sink_set_pub_wait_timeout, 439
- gst_mqtt_sink_start, 439
- gst_mqtt_sink_stop, 440
- MAX_LEN_PROP_NTP_SRVS, 413
- PROP_0, 413
- PROP_DEBUG, 413
- PROP_LAST, 413
- PROP_MAX_MSG_BUF_SIZE, 413
- PROP_MQTT_CLIENT_ID, 413
- PROP_MQTT_HOST_ADDRESS, 413
- PROP_MQTT_HOST_PORT, 413
- PROP_MQTT_NTP_SRVS, 413
- PROP_MQTT_NTP_SYNC, 413
- PROP_MQTT_OPT_CLEANSSESSION, 413
- PROP_MQTT_OPT_KEEP_ALIVE_INTERVAL, 413
- PROP_MQTT_PUB_TOPIC, 413
- PROP_MQTT_PUB_WAIT_TIMEOUT, 413
- PROP_MQTT_QOS, 413
- PROP_NUM_BUFFERS, 413
- sink_client_id, 442
- sink_pad_template, 443
- TAG_ERR_MQTT_SINK, 443
- mqttsink.h
 - _mqtt_sink_state_t, 447
 - GST_IS_MQTT_SINK, 445
 - GST_IS_MQTT_SINK_CLASS, 445
 - GST_MQTT_SINK, 445
 - GST_MQTT_SINK_CAST, 446
 - GST_MQTT_SINK_CLASS, 446
 - gst_mqtt_sink_get_type, 447
 - GST_TYPE_MQTT_SINK, 446
 - GstMqttSink, 446
 - GstMqttSinkClass, 446
 - MQTT_CONNECT_FAILURE, 447
 - MQTT_CONNECTED, 447
 - MQTT_CONNECTION_LOST, 447
 - MQTT_DISCONNECT_FAILED, 447
 - MQTT_DISCONNECTED, 447
 - mqtt_sink_state_t, 447
 - SINK_INITIALIZING, 447
 - SINK_RENDER_EOS, 447
 - SINK_RENDER_ERROR, 447
 - SINK_RENDER_STOPPED, 447
- mqttsrc.c
 - _extract_mqtt_msg_hdr_from, 452
 - _is_gst_buffer_timestamp_valid, 453
 - _put_timestamp_on_gst_buf, 453
 - _subscribe, 453
 - _unsubscribe, 454
 - cb_memory_wrapped_destroy, 455
 - cb_mqtt_on_connect, 455
 - cb_mqtt_on_connect_failure, 456
 - cb_mqtt_on_connection_lost, 457
 - cb_mqtt_on_message_arrived, 457
 - cb_mqtt_on_subscribe, 458
 - cb_mqtt_on_subscribe_failure, 458
 - cb_mqtt_on_unsubscribe, 459
 - cb_mqtt_on_unsubscribe_failure, 459
 - DEFAULT_DEBUG, 452
 - DEFAULT_IS_LIVE, 452
 - DEFAULT_MQTT_CLIENT_ID, 479
 - DEFAULT_MQTT_CLIENT_ID_FORMAT, 479
 - DEFAULT_MQTT_HOST_ADDRESS, 479

- DEFAULT_MQTT_HOST_PORT, 479
- DEFAULT_MQTT_OPT_CLEANSESSION, 452
- DEFAULT_MQTT_OPT_KEEP_ALIVE_INTERVAL, 452
- DEFAULT_MQTT_QOS, 452
- DEFAULT_MQTT_SUB_TIMEOUT, 452
- DEFAULT_MQTT_SUB_TIMEOUT_MIN, 452
- G_DEFINE_TYPE, 459
- GST_CAT_DEFAULT, 451
- GST_DEBUG_CATEGORY_STATIC, 460
- gst_mqtt_src_change_state, 460
- gst_mqtt_src_class_finalize, 460
- gst_mqtt_src_class_init, 461
- gst_mqtt_src_create, 462
- gst_mqtt_src_get_caps, 463
- gst_mqtt_src_get_client_id, 464
- gst_mqtt_src_get_debug, 464
- gst_mqtt_src_get_host_address, 464
- gst_mqtt_src_get_host_port, 465
- gst_mqtt_src_get_is_live, 465
- gst_mqtt_src_get_mqtt_qos, 465
- gst_mqtt_src_get_opt_cleansession, 466
- gst_mqtt_src_get_opt_keep_alive_interval, 466
- gst_mqtt_src_get_property, 466
- gst_mqtt_src_get_sub_timeout, 468
- gst_mqtt_src_get_sub_topic, 468
- gst_mqtt_src_get_times, 468
- gst_mqtt_src_init, 469
- gst_mqtt_src_is_seekable, 469
- gst_mqtt_src_parent_class, 451
- gst_mqtt_src_query, 470
- gst_mqtt_src_renegotiate, 470
- gst_mqtt_src_set_client_id, 471
- gst_mqtt_src_set_debug, 471
- gst_mqtt_src_set_host_address, 472
- gst_mqtt_src_set_host_port, 472
- gst_mqtt_src_set_is_live, 473
- gst_mqtt_src_set_mqtt_qos, 473
- gst_mqtt_src_set_opt_cleansession, 474
- gst_mqtt_src_set_opt_keep_alive_interval, 474
- gst_mqtt_src_set_property, 475
- gst_mqtt_src_set_sub_timeout, 476
- gst_mqtt_src_set_sub_topic, 477
- gst_mqtt_src_start, 477
- gst_mqtt_src_stop, 478
- PROP_0, 452
- PROP_DEBUG, 452
- PROP_IS_LIVE, 452
- PROP_LAST, 452
- PROP_MQTT_CLIENT_ID, 452
- PROP_MQTT_HOST_ADDRESS, 452
- PROP_MQTT_HOST_PORT, 452
- PROP_MQTT_OPT_CLEANSESSION, 452
- PROP_MQTT_OPT_KEEP_ALIVE_INTERVAL, 452
- PROP_MQTT_QOS, 452
- PROP_MQTT_SUB_TIMEOUT, 452
- PROP_MQTT_SUB_TOPIC, 452
- src_client_id, 479
- src_pad_template, 480
- TAG_ERR_MQTTSRC, 480
- mqttsrc.h
 - GST_IS_MQTT_SRC, 482
 - GST_IS_MQTT_SRC_CLASS, 482
 - GST_MQTT_SRC, 482
 - GST_MQTT_SRC_CAST, 482
 - GST_MQTT_SRC_CLASS, 483
 - gst_mqtt_src_get_type, 483
 - GST_TYPE_MQTT_SRC, 483
 - GstMqttSrc, 483
 - GstMqttSrcClass, 483
- msg_queue
 - _GstEdgeSrc, 52
 - _GstTensorQueryClient, 152
 - _GstTensorQueryServerSrc, 158
- mutex
 - _GstTensorSink, 173
- myid
 - _GstTensorRepoSink, 167
 - _GstTensorRepoSrc, 170
- n_frame
 - _GstDataRepoSrc, 40
- n_pads
 - _GstJoin, 54
- name
 - _GstTensorFilterFramework, 113
 - _GstTensorFilterFrameworkInfo, 121
 - _GstTensorSrcIIOChannelProperties, 194
 - _GstTensorSrcIIODeviceProperties, 197
 - _GstTensorTrainerFrameworkInfo, 209
 - _NNStreamerExternalConverter, 230
 - custom_cb_s, 249
 - GstTensorInfo, 260
- NAME_FILE
 - gsttensor_srcio.c, 884
- names
 - subplugin_conf, 280
 - subplugin_info_s, 281
- need_changed_caps
 - _GstDataRepoSrc, 40
- need_segment
 - _GstTensorConverter, 81
 - _GstTensorMerge, 143
 - _GstTensorMux, 146
- need_set_time
 - _GstTensorMerge, 143
 - _GstTensorMux, 147
- need_stream_start
 - _GstTensorMerge, 143
 - _GstTensorMux, 147
- negotiated
 - _GstTensorDecoder, 92
 - _GstTensorMerge, 143
 - _GstTensorMux, 147
- negotiation
 - _GstTensorRepoSrc, 171

new_data
 _GstTensorSinkClass, 175
 next_ts
 _GstTensorRate, 162
 NNS_custom_allocate_invoke
 tensor_filter_custom.h, 1132
 NNS_CUSTOM_CONVERTER
 nnstreamer_subplugin.h, 1337
 NNS_CUSTOM_DECODER
 nnstreamer_subplugin.h, 1337
 NNS_custom_destroy_notify
 tensor_filter_custom.h, 1132
 NNS_custom_easy_dynamic_register
 tensor_filter_custom_easy.c, 1532
 tensor_filter_custom_easy.h, 1139
 NNS_custom_easy_register
 tensor_filter_custom_easy.c, 1532
 tensor_filter_custom_easy.h, 1140
 NNS_custom_easy_unregister
 tensor_filter_custom_easy.c, 1533
 tensor_filter_custom_easy.h, 1141
 NNS_custom_exit_func
 tensor_filter_custom.h, 1133
 NNS_custom_get_input_dimension
 tensor_filter_custom.h, 1133
 NNS_custom_get_output_dimension
 tensor_filter_custom.h, 1133
 NNS_custom_init_func
 tensor_filter_custom.h, 1134
 NNS_custom_invoke
 tensor_filter_custom.h, 1134
 NNS_custom_invoke_dynamic
 tensor_filter_custom_easy.h, 1138
 NNS_custom_set_input_dimension
 tensor_filter_custom.h, 1135
 NNS_EASY_CUSTOM_FILTER
 nnstreamer_subplugin.h, 1337
 NNS_IF_CUSTOM
 nnstreamer_subplugin.h, 1337
 nns_logd
 nnstreamer_log.h, 1203
 nns_loge
 nnstreamer_log.h, 1203
 nns_logf
 nnstreamer_log.h, 1203
 nns_logi
 nnstreamer_log.h, 1203
 nns_logw
 nnstreamer_log.h, 1203
 nns_memcpy
 tensor_common.h, 1364
 nns_memset
 tensor_common.h, 1365
 NNS_MIMETYPE_TENSOR
 tensor_typedef.h, 1151
 NNS_MIMETYPE_TENSORS
 tensor_typedef.h, 1151
 NNS_SEARCH_FILENAME
 nnstreamer_subplugin.c, 1324
 NNS_SEARCH_GETALL
 nnstreamer_subplugin.c, 1324
 NNS_SEARCH_NO_OP
 nnstreamer_subplugin.c, 1324
 NNS_SUBPLUGIN_CHECKER
 nnstreamer_subplugin.h, 1337
 NNS_SUBPLUGIN_CONVERTER
 nnstreamer_subplugin.h, 1337
 NNS_SUBPLUGIN_DECODER
 nnstreamer_subplugin.h, 1337
 NNS_SUBPLUGIN_END
 nnstreamer_subplugin.h, 1337
 NNS_SUBPLUGIN_FILTER
 nnstreamer_subplugin.h, 1337
 NNS_SUBPLUGIN_TRAINER
 nnstreamer_subplugin.h, 1337
 NNS_support_custom
 tensor_filter_custom.c, 1525
 NNS_support_custom_easy
 tensor_filter_custom_easy.c, 1534
 NNS_TENSOR_EXTRA_MAGIC
 nnstreamer_plugin_api_impl.c, 1209
 NNS_TENSOR_MEMORY_MAX
 tensor_typedef.h, 1152
 NNS_TENSOR_PADDING_RANK_LIMIT
 gsttensor_transform.c, 959
 NNS_TENSOR_RANK_LIMIT
 tensor_typedef.h, 1152
 NNS_TENSOR_RANK_LIMIT_PREV
 nnstreamer_plugin_api_impl.c, 1209
 NNS_TENSOR_SIZE_EXTRA_LIMIT
 tensor_typedef.h, 1152
 NNS_TENSOR_SIZE_LIMIT
 tensor_typedef.h, 1152
 NNS_TENSOR_SIZE_LIMIT_STR
 tensor_typedef.h, 1152
 NNS_TENSOR_TRANSPOSE_RANK_LIMIT
 gsttensor_transform.c, 959
 NNS_VIDEO_FORMAT
 gsttensor_converter_media_info_video.h, 561
 nns_watchdog_h
 nnstreamer_watchdog.h, 1351
 nnsconf_dump
 nnstreamer_conf.c, 1171
 nnstreamer_conf.h, 1183
 nnsconf_get_custom_value_bool
 nnstreamer_conf.c, 1171
 nnstreamer_conf.h, 1184
 nnstreamer_internal.h, 1193
 nnsconf_get_custom_value_string
 nnstreamer_conf.c, 1172
 nnstreamer_conf.h, 1184
 nnstreamer_internal.h, 1194
 nnsconf_get_fullpath
 nnstreamer_conf.c, 1173
 nnstreamer_conf.h, 1185
 nnsconf_get_subplugin_info

- nnstreamer_conf.c, 1174
- nnstreamer_conf.h, 1186
- nnsconf_get_subplugin_name_prefix
 - nnstreamer_conf.c, 1175
 - nnstreamer_conf.h, 1187
- nnsconf_loadconf
 - nnstreamer_conf.c, 1175
 - nnstreamer_conf.h, 1188
- NNSCONF_PATH_CONVERTERS
 - nnstreamer_conf.h, 1183
- NNSCONF_PATH_CUSTOM_FILTERS
 - nnstreamer_conf.h, 1183
- NNSCONF_PATH_DECODERS
 - nnstreamer_conf.h, 1183
- NNSCONF_PATH_EASY_CUSTOM_FILTERS
 - nnstreamer_conf.h, 1183
- NNSCONF_PATH_END
 - nnstreamer_conf.h, 1183
- NNSCONF_PATH_FILTERS
 - nnstreamer_conf.h, 1183
- NNSCONF_PATH_TRAINERS
 - nnstreamer_conf.h, 1183
- nnsconf_subplugin_dump
 - nnstreamer_conf.c, 1176
 - nnstreamer_conf.h, 1189
- nnsconf_type_path
 - nnstreamer_conf.h, 1182
- nnsconf_validate_file
 - nnstreamer_conf.c, 1177
 - nnstreamer_conf.h, 1189
- nnstreamer, 27
- nnstreamer.c
 - FALSE, 1356
 - GST_ERROR, 1355
 - GST_PLUGIN_DEFINE, 1355
 - NNSTREAMER_INIT, 1355
 - PACKAGE, 1355
 - while, 1356
- nnstreamer/elements/gsttensor_aggregator.c, 491
- nnstreamer/elements/gsttensor_aggregator.h, 512
- nnstreamer/elements/gsttensor_converter.c, 515
- nnstreamer/elements/gsttensor_converter.h, 553
- nnstreamer/elements/gsttensor_converter_media_info_audio_host
 - 557
- nnstreamer/elements/gsttensor_converter_media_info_video_host
 - 559
- nnstreamer/elements/gsttensor_converter_media_no_audio_host
 - 561
- nnstreamer/elements/gsttensor_converter_media_no_video_host
 - 565
- nnstreamer/elements/gsttensor_crop.c, 568
- nnstreamer/elements/gsttensor_crop.h, 585
- nnstreamer/elements/gsttensor_debug.c, 588
- nnstreamer/elements/gsttensor_debug.h, 599
- nnstreamer/elements/gsttensor_decoder.c, 604
- nnstreamer/elements/gsttensor_decoder.h, 630
- nnstreamer/elements/gsttensor_demux.c, 634
- nnstreamer/elements/gsttensor_demux.h, 649
- nnstreamer/elements/gsttensor_if.c, 653
- nnstreamer/elements/gsttensor_if.h, 679
- nnstreamer/elements/gsttensor_merge.c, 685
- nnstreamer/elements/gsttensor_merge.h, 704
- nnstreamer/elements/gsttensor_mux.c, 708
- nnstreamer/elements/gsttensor_mux.h, 726
- nnstreamer/elements/gsttensor_rate.c, 730
- nnstreamer/elements/gsttensor_rate.h, 748
- nnstreamer/elements/gsttensor_repo.c, 752
- nnstreamer/elements/gsttensor_repo.h, 763
- nnstreamer/elements/gsttensor_reposink.c, 772
- nnstreamer/elements/gsttensor_reposink.h, 784
- nnstreamer/elements/gsttensor_reposrc.c, 788
- nnstreamer/elements/gsttensor_reposrc.h, 797
- nnstreamer/elements/gsttensor_sink.c, 800
- nnstreamer/elements/gsttensor_sink.h, 817
- nnstreamer/elements/gsttensor_sparsedec.c, 820
- nnstreamer/elements/gsttensor_sparsedec.h, 831
- nnstreamer/elements/gsttensor_sparseenc.c, 834
- nnstreamer/elements/gsttensor_sparseenc.h, 845
- nnstreamer/elements/gsttensor_sparseutil.c, 848
- nnstreamer/elements/gsttensor_sparseutil.h, 851
- nnstreamer/elements/gsttensor_split.c, 854
- nnstreamer/elements/gsttensor_split.h, 869
- nnstreamer/elements/gsttensor_srcio.c, 873
- nnstreamer/elements/gsttensor_srcio.h, 915
- nnstreamer/elements/gsttensor_trainer.c, 920
- nnstreamer/elements/gsttensor_trainer.h, 951
- nnstreamer/elements/gsttensor_transform.c, 954
- nnstreamer/elements/gsttensor_transform.h, 987
- nnstreamer/elements/ignore_warning.sh, 995
- nnstreamer/hw_accel.c, 995
- nnstreamer/hw_accel.h, 996
- nnstreamer/include/nnstreamer_plugin_api.h, 998
- nnstreamer/include/nnstreamer_plugin_api_converter.h, 1015
- nnstreamer/include/nnstreamer_plugin_api_decoder.h, 1020
- nnstreamer/include/nnstreamer_plugin_api_filter.h, 1025
- nnstreamer/include/nnstreamer_plugin_api_trainer.h, 1042
- nnstreamer/include/nnstreamer_plugin_api_util.h, 1049
- nnstreamer/include/nnstreamer_util.h, 1120
- nnstreamer/include/tensor_converter_custom.h, 1122
- nnstreamer/include/tensor_decoder_custom.h, 1125
- nnstreamer/include/tensor_filter_custom.h, 1129
- nnstreamer/include/tensor_filter_custom_easy.h, 1136
- nnstreamer/include/tensor_if.h, 1142
- nnstreamer/include/tensor_typedef.h, 1146
- nnstreamer/ml_agent.c, 1156
- nnstreamer/ml_agent.h, 1159
- nnstreamer/nnstreamer_conf.c, 1160
- nnstreamer/nnstreamer_conf.h, 1179
- nnstreamer/nnstreamer_internal.h, 1190
- nnstreamer/nnstreamer_log.c, 1196
- nnstreamer/nnstreamer_log.h, 1199
- nnstreamer/nnstreamer_plugin_api_impl.c, 1206

- nnstreamer/nnstreamer_plugin_api_util_impl.c, 1250
- nnstreamer/nnstreamer_subplugin.c, 1322
- nnstreamer/nnstreamer_subplugin.h, 1335
- nnstreamer/nnstreamer_watchdog.c, 1344
- nnstreamer/nnstreamer_watchdog.h, 1349
- nnstreamer/registerer/nnstreamer.c, 1353
- nnstreamer/tensor_allocator.c, 1356
- nnstreamer/tensor_common.h, 1361
- nnstreamer/tensor_data.c, 1380
- nnstreamer/tensor_data.h, 1390
- nnstreamer/tensor_filter/tensor_filter.c, 1399
- nnstreamer/tensor_filter/tensor_filter.h, 1432
- nnstreamer/tensor_filter/tensor_filter_common.c, 1435
- nnstreamer/tensor_filter/tensor_filter_common.h, 1495
- nnstreamer/tensor_filter/tensor_filter_custom.c, 1517
- nnstreamer/tensor_filter/tensor_filter_custom_easy.c, 1526
- nnstreamer/tensor_filter/tensor_filter_single.c, 1535
- nnstreamer/tensor_filter/tensor_filter_single.h, 1547
- nnstreamer/tensor_filter/tensor_filter_support_cc.cc, 1551
- nnstreamer/tensor_meta.c, 1553
- nnstreamer/tensor_meta.h, 1557
- nnstreamer/tensor_query/tensor_query_client.c, 1560
- nnstreamer/tensor_query/tensor_query_client.h, 1575
- nnstreamer/tensor_query/tensor_query_common.c, 1579
- nnstreamer/tensor_query/tensor_query_common.h, 1580
- nnstreamer/tensor_query/tensor_query_server.c, 1583
- nnstreamer/tensor_query/tensor_query_server.h, 1591
- nnstreamer/tensor_query/tensor_query_serversink.c, 1599
- nnstreamer/tensor_query/tensor_query_serversink.h, 1608
- nnstreamer/tensor_query/tensor_query_serversrc.c, 1612
- nnstreamer/tensor_query/tensor_query_serversrc.h, 1624
- nnstreamer_conf.c
 - _fill_in_vstr, 1164
 - _fill_subplugin_path, 1165
 - _foreach_custom_property, 1165
 - _g_list_foreach_vstr_helper, 1166
 - _get_filenames, 1167
 - _get_subplugin_with_type, 1168
 - _parse_bool_string, 1169
 - _strdup_getenv, 1169
 - _validate_file, 1170
 - conf, 1178
 - CONF_SOURCE_END, 1164
 - CONF_SOURCE_ENVVAR, 1164
 - CONF_SOURCE_EXTRA_CONF, 1164
 - CONF_SOURCE_HARDCODE, 1164
 - CONF_SOURCE_INI, 1164
 - conf_sources, 1163
 - custom_table, 1178
 - nnsconf_dump, 1171
- nnsconf_get_custom_value_bool, 1171
- nnsconf_get_custom_value_string, 1172
- nnsconf_get_fullpath, 1173
- nnsconf_get_subplugin_info, 1174
- nnsconf_get_subplugin_name_prefix, 1175
- nnsconf_loadconf, 1175
- nnsconf_subplugin_dump, 1176
- nnsconf_validate_file, 1177
- NNSTREAMER_ENVVAR, 1178
- NNSTREAMER_PATH, 1178
- NNSTREAMER_PREFIX_CONVERTER, 1163
- NNSTREAMER_PREFIX_CUSTOMFILTERS, 1163
- NNSTREAMER_PREFIX_DECODER, 1163
- NNSTREAMER_PREFIX_FILTER, 1163
- NNSTREAMER_PREFIX_TRAINER, 1163
- STR_BOOL, 1163
- subplugin_prefixes, 1179
- nnstreamer_conf.h
 - nnsconf_dump, 1183
 - nnsconf_get_custom_value_bool, 1184
 - nnsconf_get_custom_value_string, 1184
 - nnsconf_get_fullpath, 1185
 - nnsconf_get_subplugin_info, 1186
 - nnsconf_get_subplugin_name_prefix, 1187
 - nnsconf_loadconf, 1188
 - NNSCONF_PATH_CONVERTERS, 1183
 - NNSCONF_PATH_CUSTOM_FILTERS, 1183
 - NNSCONF_PATH_DECODERS, 1183
 - NNSCONF_PATH_EASY_CUSTOM_FILTERS, 1183
 - NNSCONF_PATH_END, 1183
 - NNSCONF_PATH_FILTERS, 1183
 - NNSCONF_PATH_TRAINERS, 1183
 - nnsconf_subplugin_dump, 1189
 - nnsconf_type_path, 1182
 - nnsconf_validate_file, 1189
 - NNSTREAMER_CONF_FILE, 1182
 - NNSTREAMER_DEFAULT_CONF_FILE, 1182
 - NNSTREAMER_ENVVAR_CONF_FILE, 1182
 - NNSTREAMER_SO_FILE_EXTENSION, 1182
 - NNSTREAMER_SYS_ROOT_PATH_PREFIX, 1182
 - NNSTREAMER_CONF_FILE
 - nnstreamer_conf.h, 1182
- nnstreamer_converter_custom_register
 - gsttensor_converter.c, 549
 - tensor_converter_custom.h, 1123
- nnstreamer_converter_custom_unregister
 - gsttensor_converter.c, 549
 - tensor_converter_custom.h, 1124
- nnstreamer_converter_find
 - gsttensor_converter.c, 550
 - nnstreamer_plugin_api_converter.h, 1018
- nnstreamer_converter_set_custom_property_desc
 - gsttensor_converter.c, 551
 - nnstreamer_plugin_api_converter.h, 1019
- nnstreamer_converter_validate

- gsttensor_converter.c, 551
- NNStreamer_custom
 - tensor_filter_custom.h, 1135
- NNStreamer_custom_class
 - tensor_filter_custom.h, 1135
- nnstreamer_decoder_custom_register
 - gsttensor_decoder.c, 624
 - tensor_decoder_custom.h, 1127
- nnstreamer_decoder_custom_unregister
 - gsttensor_decoder.c, 625
 - tensor_decoder_custom.h, 1128
- nnstreamer_decoder_exit
 - gsttensor_decoder.c, 625
 - nnstreamer_plugin_api_decoder.h, 1023
- nnstreamer_decoder_find
 - gsttensor_decoder.c, 627
 - nnstreamer_plugin_api_decoder.h, 1023
- nnstreamer_decoder_probe
 - gsttensor_decoder.c, 628
 - nnstreamer_plugin_api_decoder.h, 1024
- nnstreamer_decoder_set_custom_property_desc
 - gsttensor_decoder.c, 628
 - nnstreamer_plugin_api_decoder.h, 1025
- nnstreamer_decoder_validate
 - gsttensor_decoder.c, 629
- NNSTREAMER_DEFAULT_CONF_FILE
 - nnstreamer_conf.h, 1182
- NNSTREAMER_ENVVAR
 - nnstreamer_conf.c, 1178
- NNSTREAMER_ENVVAR_CONF_FILE
 - nnstreamer_conf.h, 1182
- nnstreamer_filter_exit
 - nnstreamer_plugin_api_filter.h, 1037
 - tensor_filter_common.c, 1483
- nnstreamer_filter_find
 - nnstreamer_plugin_api_filter.h, 1037
 - tensor_filter_common.c, 1483
- nnstreamer_filter_find_best_fit
 - tensor_filter_common.c, 1484
- nnstreamer_filter_probe
 - nnstreamer_plugin_api_filter.h, 1038
 - tensor_filter_common.c, 1485
- nnstreamer_filter_set_custom_property_desc
 - nnstreamer_plugin_api_filter.h, 1039
 - tensor_filter_common.c, 1486
- nnstreamer_filter_shared_model_get
 - nnstreamer_plugin_api_filter.h, 1039
 - tensor_filter_common.c, 1486
- nnstreamer_filter_shared_model_insert_and_get
 - nnstreamer_plugin_api_filter.h, 1040
 - tensor_filter_common.c, 1487
- nnstreamer_filter_shared_model_remove
 - nnstreamer_plugin_api_filter.h, 1040
 - tensor_filter_common.c, 1487
- nnstreamer_filter_shared_model_replace
 - nnstreamer_plugin_api_filter.h, 1041
 - tensor_filter_common.c, 1488
- nnstreamer_filter_validate
 - tensor_filter_common.c, 1488
- nnstreamer_if_custom_register
 - gsttensor_if.c, 675
 - tensor_if.h, 1144
- nnstreamer_if_custom_unregister
 - gsttensor_if.c, 675
 - tensor_if.h, 1144
- NNSTREAMER_INIT
 - nnstreamer.c, 1355
- nnstreamer_internal.h
 - gst_tensor_filter_check_hw_availability, 1192
 - gst_tensor_filter_detect_framework, 1193
 - nnsconf_get_custom_value_bool, 1193
 - nnsconf_get_custom_value_string, 1194
- nnstreamer_log.c
 - _NNSTREAMER_ERROR_LENGTH, 1197
 - __attribute__, 1197
 - _backtrace_to_string, 1197
 - _nnstreamer_error, 1198
 - _nnstreamer_error_clean, 1198
 - errmsg, 1199
 - errmsg_reported, 1199
 - G_LOCK_DEFINE_STATIC, 1198
- nnstreamer_log.h
 - _backtrace_to_string, 1204
 - _nnstreamer_error, 1204
 - _nnstreamer_error_clean, 1204
 - _nnstreamer_error_write, 1205
 - GST_ELEMENT_ERROR_BTRACE, 1201
 - m_log_stacktrace, 1201
 - m_logd, 1201
 - m_loge, 1201
 - m_loge_stacktrace, 1202
 - m_logf, 1202
 - m_logf_stacktrace, 1202
 - m_logi, 1202
 - m_logw, 1202
 - nns_logd, 1203
 - nns_loge, 1203
 - nns_logf, 1203
 - nns_logi, 1203
 - nns_logw, 1203
 - TAG_NAME, 1203
- NNSTREAMER_PATH
 - nnstreamer_conf.c, 1178
- nnstreamer_plugin_api.h
 - gst_structure_get_media_type, 1000
 - gst_structure_is_tensor_stream, 1000
 - gst_tensor_alloc_init, 1001
 - gst_tensor_buffer_append_memory, 1002
 - gst_tensor_buffer_get_count, 1003
 - gst_tensor_buffer_get_nth_memory, 1004
 - gst_tensor_caps_can_intersect, 1006
 - gst_tensor_caps_from_config, 1006
 - gst_tensor_caps_update_dimension, 1007
 - gst_tensor_meta_info_append_header, 1008
 - gst_tensor_meta_info_parse_memory, 1009
 - gst_tensors_caps_from_config, 1010

- gst_tensors_config_from_peer, [1011](#)
- gst_tensors_config_from_structure, [1013](#)
- nnstreamer_plugin_api_converter.h
 - nnstreamer_converter_find, [1018](#)
 - nnstreamer_converter_set_custom_property_desc, [1019](#)
 - NNStreamerExternalConverter, [1017](#)
 - registerExternalConverter, [1019](#)
 - unregisterExternalConverter, [1020](#)
- nnstreamer_plugin_api_decoder.h
 - GstTensorDecoderDef, [1022](#)
 - nnstreamer_decoder_exit, [1023](#)
 - nnstreamer_decoder_find, [1023](#)
 - nnstreamer_decoder_probe, [1024](#)
 - nnstreamer_decoder_set_custom_property_desc, [1025](#)
- nnstreamer_plugin_api_filter.h
 - ACCL_AUTO, [1034](#)
 - ACCL_AUTO_STR, [1029](#)
 - ACCL_CPU, [1034](#)
 - ACCL_CPU_NEON, [1035](#)
 - ACCL_CPU_NEON_STR, [1029](#)
 - ACCL_CPU_SIMD, [1035](#)
 - ACCL_CPU_SIMD_STR, [1029](#)
 - ACCL_CPU_STR, [1029](#)
 - ACCL_DEFAULT, [1034](#)
 - ACCL_DEFAULT_STR, [1029](#)
 - ACCL_GPU, [1035](#)
 - ACCL_GPU_STR, [1030](#)
 - accl_hw, [1034](#)
 - ACCL_NONE, [1034](#)
 - ACCL_NONE_STR, [1030](#)
 - ACCL_NPU, [1035](#)
 - ACCL_NPU_EDGE_TPU, [1035](#)
 - ACCL_NPU_EDGE_TPU_STR, [1030](#)
 - ACCL_NPU_MOVIDIUS, [1035](#)
 - ACCL_NPU_MOVIDIUS_STR, [1030](#)
 - ACCL_NPU_SLSI, [1035](#)
 - ACCL_NPU_SLSI_STR, [1030](#)
 - ACCL_NPU_SR, [1035](#)
 - ACCL_NPU_SR_STR, [1030](#)
 - ACCL_NPU_SRCN, [1035](#)
 - ACCL_NPU_SRCN_STR, [1031](#)
 - ACCL_NPU_STR, [1031](#)
 - ACCL_NPU_VIVANTE, [1035](#)
 - ACCL_NPU_VIVANTE_STR, [1031](#)
 - CHECK_HW_AVAILABILITY, [1035](#)
 - checkGstTensorFilterFrameworkVersion, [1031](#)
 - CUSTOM_PROP, [1035](#)
 - DESTROY_NOTIFY, [1035](#)
 - event_ops, [1035](#)
 - get_accl_hw_str, [1036](#)
 - get_accl_hw_type, [1037](#)
 - GET_IN_OUT_INFO, [1036](#)
 - GST_TENSOR_FILTER_API_VERSION_DEFINED, [1031](#)
 - GST_TENSOR_FILTER_API_VERSION_MAX, [1032](#)
 - GST_TENSOR_FILTER_API_VERSION_MIN, [1032](#)
 - GST_TENSOR_FILTER_FRAMEWORK_BASE, [1032](#)
 - GST_TENSOR_FILTER_FRAMEWORK_V0, [1032](#)
 - GST_TENSOR_FILTER_FRAMEWORK_V1, [1032](#)
 - GstTensorFilterFramework, [1033](#)
 - GstTensorFilterFrameworkEventData, [1033](#)
 - GstTensorFilterFrameworkInfo, [1033](#)
 - GstTensorFilterFrameworkStatistics, [1033](#)
 - GstTensorFilterProperties, [1033](#)
 - model_info_ops, [1035](#)
 - nnstreamer_filter_exit, [1037](#)
 - nnstreamer_filter_find, [1037](#)
 - nnstreamer_filter_probe, [1038](#)
 - nnstreamer_filter_set_custom_property_desc, [1039](#)
 - nnstreamer_filter_shared_model_get, [1039](#)
 - nnstreamer_filter_shared_model_insert_and_get, [1040](#)
 - nnstreamer_filter_shared_model_remove, [1040](#)
 - nnstreamer_filter_shared_model_replace, [1041](#)
 - parse_accl_hw, [1032](#)
 - parse_accl_hw_fill, [1041](#)
 - RELOAD_MODEL, [1035](#)
 - SET_ACCELERATOR, [1035](#)
 - SET_INPUT_INFO, [1036](#)
 - SET_INPUT_PROP, [1035](#)
 - SET_OUTPUT_PROP, [1035](#)
 - tensors_layout, [1034](#)
- nnstreamer_plugin_api_impl.c
 - _append_prev_caps, [1209](#)
 - _get_flexible_caps, [1210](#)
 - _get_tensor_caps, [1211](#)
 - _get_tensors_caps, [1212](#)
 - _gst_tensor_time_sync_buffer_update, [1213](#)
 - _gst_tensor_time_sync_is_eos, [1214](#)
 - _is_structure_dimension_same, [1214](#)
 - AGGREGATION_DEFAULT_KEY, [1209](#)
 - gst_memory_map_is_extra_tensor, [1215](#)
 - gst_structure_get_media_type, [1216](#)
 - gst_structure_is_tensor_stream, [1217](#)
 - gst_tensor_aggregation_add_data, [1218](#)
 - gst_tensor_aggregation_clear, [1218](#)
 - gst_tensor_aggregation_clear_all, [1219](#)
 - gst_tensor_aggregation_clear_internal, [1220](#)
 - gst_tensor_aggregation_free_data, [1220](#)
 - gst_tensor_aggregation_get_adapter, [1221](#)
 - gst_tensor_aggregation_get_data, [1222](#)
 - gst_tensor_aggregation_init, [1222](#)
 - gst_tensor_buffer_append_memory, [1223](#)
 - gst_tensor_buffer_from_config, [1225](#)
 - gst_tensor_buffer_get_count, [1226](#)
 - gst_tensor_buffer_get_nth_memory, [1227](#)
 - gst_tensor_caps_can_intersect, [1228](#)
 - gst_tensor_caps_from_config, [1229](#)
 - gst_tensor_caps_update_dimension, [1230](#)
 - gst_tensor_extra_info_init, [1231](#)

- gst_tensor_meta_info_append_header, 1232
- gst_tensor_meta_info_parse_memory, 1233
- gst_tensor_pad_caps_from_config, 1234
- gst_tensor_pad_get_format, 1236
- gst_tensor_pad_possible_caps_from_config, 1237
- gst_tensor_parse_config_file, 1238
- gst_tensor_time_sync_buffer_from_collectpad, 1239
- gst_tensor_time_sync_flush, 1240
- gst_tensor_time_sync_get_current_time, 1241
- gst_tensor_time_sync_get_mode, 1242
- gst_tensor_time_sync_get_mode_string, 1243
- gst_tensor_time_sync_mode_string, 1249
- gst_tensor_time_sync_set_option_data, 1243
- gst_tensors_caps_from_config, 1244
- gst_tensors_config_from_peer, 1245
- gst_tensors_config_from_structure, 1246
- NNS_TENSOR_EXTRA_MAGIC, 1209
- NNS_TENSOR_RANK_LIMIT_PREV, 1209
- set_property_value, 1248
- nnstreamer_plugin_api_trainer.h
 - GST_TENSOR_TRAINER_FRAMEWORK_BASE, 1044
 - GST_TENSOR_TRAINER_FRAMEWORK_V1, 1044
 - GstTensorTrainerEventNotifier, 1045
 - GstTensorTrainerEventType, 1045
 - GstTensorTrainerFramework, 1045
 - GstTensorTrainerFrameworkInfo, 1045
 - GstTensorTrainerProperties, 1045
 - nnstreamer_trainer_exit, 1046
 - nnstreamer_trainer_notify_event, 1046
 - nnstreamer_trainer_probe, 1048
 - TRAINER_EVENT_EPOCH_COMPLETION, 1046
 - TRAINER_EVENT_TRAINING_COMPLETION, 1046
 - TRAINER_EVENT_UNKNOWN, 1046
- nnstreamer_plugin_api_util.h
 - _STR_NULL, 1052
 - find_key_strv, 1053
 - gst_tensor_dimension_get_min_rank, 1055
 - gst_tensor_dimension_get_rank, 1056
 - gst_tensor_dimension_is_equal, 1057
 - gst_tensor_dimension_is_valid, 1058
 - gst_tensor_dimension_string_is_equal, 1060
 - gst_tensor_get_dimension_string, 1061
 - gst_tensor_get_element_count, 1063
 - gst_tensor_get_element_size, 1064
 - gst_tensor_get_format, 1065
 - gst_tensor_get_format_string, 1067
 - gst_tensor_get_rank_dimension_string, 1068
 - gst_tensor_get_type, 1069
 - gst_tensor_get_type_string, 1070
 - gst_tensor_info_convert_to_meta, 1071
 - gst_tensor_info_copy, 1072
 - gst_tensor_info_copy_n, 1074
 - gst_tensor_info_free, 1075
 - gst_tensor_info_get_rank, 1077
 - gst_tensor_info_get_size, 1078
 - gst_tensor_info_init, 1079
 - gst_tensor_info_is_equal, 1080
 - gst_tensor_info_validate, 1081
 - gst_tensor_meta_info_convert, 1082
 - gst_tensor_meta_info_get_data_size, 1084
 - gst_tensor_meta_info_get_header_size, 1085
 - gst_tensor_meta_info_get_version, 1086
 - gst_tensor_meta_info_init, 1086
 - gst_tensor_meta_info_parse_header, 1087
 - gst_tensor_meta_info_update_header, 1088
 - gst_tensor_meta_info_validate, 1089
 - gst_tensor_parse_dimension, 1091
 - gst_tensors_config_copy, 1092
 - gst_tensors_config_free, 1093
 - gst_tensors_config_init, 1094
 - gst_tensors_config_is_equal, 1096
 - gst_tensors_config_is_flexible, 1053
 - gst_tensors_config_is_sparse, 1053
 - gst_tensors_config_is_static, 1053
 - gst_tensors_config_to_string, 1097
 - gst_tensors_config_validate, 1098
 - gst_tensors_info_copy, 1100
 - gst_tensors_info_free, 1101
 - gst_tensors_info_get_dimensions_string, 1103
 - gst_tensors_info_get_names_string, 1104
 - gst_tensors_info_get_nth_info, 1105
 - gst_tensors_info_get_rank_dimensions_string, 1106
 - gst_tensors_info_get_size, 1107
 - gst_tensors_info_get_types_string, 1108
 - gst_tensors_info_init, 1109
 - gst_tensors_info_is_equal, 1111
 - gst_tensors_info_parse_dimensions_string, 1112
 - gst_tensors_info_parse_names_string, 1114
 - gst_tensors_info_parse_types_string, 1115
 - gst_tensors_info_to_string, 1117
 - gst_tensors_info_validate, 1118
 - nnstreamer_version_fetch, 1119
 - nnstreamer_version_string, 1120
- nnstreamer_plugin_api_util_impl.c
 - _compare_rate, 1255
 - _gcd, 1256
 - find_key_strv, 1256
 - gst_tensor_dimension_get_min_rank, 1257
 - gst_tensor_dimension_get_rank, 1258
 - gst_tensor_dimension_is_equal, 1259
 - gst_tensor_dimension_is_valid, 1260
 - gst_tensor_dimension_string_is_equal, 1262
 - gst_tensor_get_dimension_string, 1263
 - gst_tensor_get_element_count, 1265
 - gst_tensor_get_element_size, 1266
 - gst_tensor_get_format, 1267
 - gst_tensor_get_format_string, 1269
 - gst_tensor_get_rank_dimension_string, 1270
 - gst_tensor_get_type, 1271
 - gst_tensor_get_type_string, 1272
 - gst_tensor_info_convert_to_meta, 1273

- gst_tensor_info_copy, [1274](#)
- gst_tensor_info_copy_n, [1276](#)
- gst_tensor_info_free, [1277](#)
- gst_tensor_info_get_rank, [1279](#)
- gst_tensor_info_get_size, [1280](#)
- gst_tensor_info_init, [1281](#)
- gst_tensor_info_is_equal, [1282](#)
- gst_tensor_info_validate, [1283](#)
- gst_tensor_meta_info_convert, [1284](#)
- gst_tensor_meta_info_get_data_size, [1286](#)
- gst_tensor_meta_info_get_header_size, [1287](#)
- gst_tensor_meta_info_get_version, [1288](#)
- gst_tensor_meta_info_init, [1288](#)
- gst_tensor_meta_info_parse_header, [1289](#)
- gst_tensor_meta_info_update_header, [1290](#)
- gst_tensor_meta_info_validate, [1291](#)
- GST_TENSOR_META_IS_V1, [1253](#)
- GST_TENSOR_META_IS_VALID, [1254](#)
- GST_TENSOR_META_MAGIC, [1254](#)
- GST_TENSOR_META_MAGIC_VALID, [1254](#)
- GST_TENSOR_META_MAKE_VERSION, [1254](#)
- GST_TENSOR_META_VERSION, [1254](#)
- GST_TENSOR_META_VERSION_VALID, [1255](#)
- gst_tensor_parse_dimension, [1293](#)
- gst_tensors_config_copy, [1294](#)
- gst_tensors_config_free, [1295](#)
- gst_tensors_config_init, [1296](#)
- gst_tensors_config_is_equal, [1298](#)
- gst_tensors_config_to_string, [1299](#)
- gst_tensors_config_validate, [1300](#)
- gst_tensors_info_copy, [1301](#)
- gst_tensors_info_free, [1302](#)
- gst_tensors_info_get_dimensions_string, [1304](#)
- gst_tensors_info_get_names_string, [1305](#)
- gst_tensors_info_get_nth_info, [1306](#)
- gst_tensors_info_get_rank_dimensions_string, [1307](#)
- gst_tensors_info_get_size, [1308](#)
- gst_tensors_info_get_types_string, [1309](#)
- gst_tensors_info_init, [1310](#)
- gst_tensors_info_is_equal, [1312](#)
- gst_tensors_info_parse_dimensions_string, [1313](#)
- gst_tensors_info_parse_names_string, [1315](#)
- gst_tensors_info_parse_types_string, [1316](#)
- gst_tensors_info_to_string, [1318](#)
- gst_tensors_info_validate, [1319](#)
- nnstreamer_version_fetch, [1320](#)
- nnstreamer_version_string, [1321](#)
- tensor_element_size, [1321](#)
- tensor_element_typename, [1321](#)
- tensor_format_name, [1322](#)
- NNSTREAMER_PREFIX_CONVERTER
 - nnstreamer_conf.c, [1163](#)
- NNSTREAMER_PREFIX_CUSTOMFILTERS
 - nnstreamer_conf.c, [1163](#)
- NNSTREAMER_PREFIX_DECODER
 - nnstreamer_conf.c, [1163](#)
- NNSTREAMER_PREFIX_FILTER
 - nnstreamer_conf.c, [1163](#)
- NNSTREAMER_PREFIX_TRAINER
 - nnstreamer_conf.c, [1163](#)
- NNSTREAMER_SO_FILE_EXTENSION
 - nnstreamer_conf.h, [1182](#)
- nnstreamer_subplugin.c
 - _close_handle, [1324](#)
 - _get_subplugin_data, [1325](#)
 - _search_subplugin, [1325](#)
 - _spdata_destroy, [1326](#)
 - fini_subplugin, [1327](#)
 - G_LOCK_DEFINE_STATIC, [1327](#)
 - get_all_subplugins, [1327](#)
 - get_subplugin, [1328](#)
 - handles, [1334](#)
 - init_subplugin, [1329](#)
 - NNS_SEARCH_FILENAME, [1324](#)
 - NNS_SEARCH_GETALL, [1324](#)
 - NNS_SEARCH_NO_OP, [1324](#)
 - register_subplugin, [1329](#)
 - searchAlgorithm, [1334](#)
 - subplugin_get_custom_property_desc, [1331](#)
 - subplugin_set_custom_property_desc, [1332](#)
 - subpluginData, [1334](#)
 - subplugins, [1334](#)
 - subpluginSearchLogic, [1324](#)
 - unregister_subplugin, [1333](#)
- nnstreamer_subplugin.h
 - get_all_subplugins, [1337](#)
 - get_subplugin, [1338](#)
 - NNS_CUSTOM_CONVERTER, [1337](#)
 - NNS_CUSTOM_DECODER, [1337](#)
 - NNS_EASY_CUSTOM_FILTER, [1337](#)
 - NNS_IF_CUSTOM, [1337](#)
 - NNS_SUBPLUGIN_CHECKER, [1337](#)
 - NNS_SUBPLUGIN_CONVERTER, [1337](#)
 - NNS_SUBPLUGIN_DECODER, [1337](#)
 - NNS_SUBPLUGIN_END, [1337](#)
 - NNS_SUBPLUGIN_FILTER, [1337](#)
 - NNS_SUBPLUGIN_TRAINER, [1337](#)
 - register_subplugin, [1339](#)
 - subplugin_get_custom_property_desc, [1341](#)
 - subplugin_set_custom_property_desc, [1342](#)
 - subpluginType, [1337](#)
 - unregister_subplugin, [1343](#)
- NNSTREAMER_SYS_ROOT_PATH_PREFIX
 - nnstreamer_conf.h, [1182](#)
- nnstreamer_trainer_exit
 - gsttensor_trainer.c, [948](#)
 - nnstreamer_plugin_api_trainer.h, [1046](#)
- nnstreamer_trainer_notify_event
 - gsttensor_trainer.c, [949](#)
 - nnstreamer_plugin_api_trainer.h, [1046](#)
- nnstreamer_trainer_probe
 - gsttensor_trainer.c, [949](#)
 - nnstreamer_plugin_api_trainer.h, [1048](#)
- nnstreamer_util.h
 - _g_memdup, [1121](#)

- UNUSED, 1121
- nnstreamer_version_fetch
 - nnstreamer_plugin_api_util.h, 1119
 - nnstreamer_plugin_api_util_impl.c, 1320
- nnstreamer_version_string
 - nnstreamer_plugin_api_util.h, 1120
 - nnstreamer_plugin_api_util_impl.c, 1321
- nnstreamer_watchdog.c
 - _loop_running_cb, 1346
 - _nnstreamer_watchdog_thread, 1346
 - nnstreamer_watchdog_create, 1346
 - nnstreamer_watchdog_destroy, 1347
 - nnstreamer_watchdog_feed, 1348
 - nnstreamer_watchdog_release, 1348
 - NnstWatchdog, 1345
- nnstreamer_watchdog.h
 - nns_watchdog_h, 1351
 - nnstreamer_watchdog_create, 1351
 - nnstreamer_watchdog_destroy, 1351
 - nnstreamer_watchdog_feed, 1352
 - nnstreamer_watchdog_release, 1353
- nnstreamer_watchdog_create
 - nnstreamer_watchdog.c, 1346
 - nnstreamer_watchdog.h, 1351
- nnstreamer_watchdog_destroy
 - nnstreamer_watchdog.c, 1347
 - nnstreamer_watchdog.h, 1351
- nnstreamer_watchdog_feed
 - nnstreamer_watchdog.c, 1348
 - nnstreamer_watchdog.h, 1352
- nnstreamer_watchdog_release
 - nnstreamer_watchdog.c, 1348
 - nnstreamer_watchdog.h, 1353
- NNStreamerExternalConverter
 - nnstreamer_plugin_api_converter.h, 1017
- NnstWatchdog
 - nnstreamer_watchdog.c, 1345
- nnz
 - GstSparseTensorInfo, 253
- NO_ANONYMOUS_NESTED_STRUCT
 - tensor_filter_support_cc.cc, 1553
- notifier
 - _GstTensorTrainer, 201
 - _GstTensorTrainerEventNotifier, 204
- nth
 - GstTensorPad, 265
- ntp_packet_t
 - ntputil.c, 485
- ntp_timestamp_t
 - ntputil.c, 485
- ntputil.c
 - _convert_to_host_byte_order, 486
 - ntp_packet_t, 485
 - ntp_timestamp_t, 485
 - NTPUTIL_DEFAULT_HNAME, 487
 - NTPUTIL_DEFAULT_PORT, 487
 - ntputil_get_epoch, 486
 - NTPUTIL_MAX_FRAC_DOUBLE, 488
 - NTPUTIL_SEC_TO_USEC_MULTIPLIER, 488
 - NTPUTIL_TIMESTAMP_DELTA, 488
- ntputil.h
 - _convert_to_host_byte_order, 489
 - ntputil_get_epoch, 490
- NTPUTIL_DEFAULT_HNAME
 - ntputil.c, 487
- NTPUTIL_DEFAULT_PORT
 - ntputil.c, 487
- ntputil_get_epoch
 - ntputil.c, 486
 - ntputil.h, 490
- NTPUTIL_MAX_FRAC_DOUBLE
 - ntputil.c, 488
- NTPUTIL_SEC_TO_USEC_MULTIPLIER
 - ntputil.c, 488
- NTPUTIL_TIMESTAMP_DELTA
 - ntputil.c, 488
- num
 - tensor_crop_info_s, 282
 - tensor_if_sv_s, 287
- num_buffers
 - _GstMqttSink, 66
- num_channels_enabled
 - _GstTensorSrcIIO, 190
- num_data
 - GstTensorRepo, 270
- num_dumped
 - _GstMqttSrc, 71
- num_epochs
 - _GstTensorTrainerProperties, 212
- num_extra_tensors
 - GstTensorExtraInfo, 258
- num_hw
 - _GstTensorFilterFrameworkEventData, 119
 - _GstTensorFilterFrameworkInfo, 122
 - _GstTensorFilterProperties, 132
- num_inputs
 - _GstTensorTrainerProperties, 212
- num_labels
 - _GstTensorTrainerProperties, 212
- num_mems
 - _GstMQTTMessageHdr, 59
- num_models
 - _GstTensorFilterFrameworkEventData, 119
 - _GstTensorFilterProperties, 132
- num_samples
 - _GstDataRepoSrc, 40
- num_srcpads
 - _GstTensorDemux, 98
 - _GstTensorIf, 138
 - _GstTensorSplit, 183
- num_tensors
 - _FilterTransformData, 30
 - _GstTensorSplit, 183
 - GstTensorsInfo, 276
- num_training_samples
 - _GstTensorTrainerProperties, 212

- num_validation_samples
 - [_GstTensorTrainerProperties](#), 212
- O_BINARY
 - [gstdatareposrc.c](#), 322
- o_myid
 - [_GstTensorRepoSink](#), 167
 - [_GstTensorRepoSrc](#), 171
- OCTET_CAPS
 - [gstdatareposink.c](#), 302
- OCTET_CAPS_STR
 - [gsttensor_converter.c](#), 520
- offset
 - [_GstTensorSrcIOChannelProperties](#), 194
- OFFSET_SUFFIX
 - [gsttensor_srcio.c](#), 884
- old_timestamp
 - [_GstTensorConverter](#), 81
- old_total_invoke_latency
 - [_GstTensorFilterStatistics](#), 134
- old_total_invoke_num
 - [_GstTensorFilterStatistics](#), 135
- op
 - [_GstTensorIf](#), 139
 - [tensor_transform_operator_s](#), 290
- open
 - [_GstTensorFilterFramework](#), 113
 - [_NNStreamerExternalConverter](#), 230
- operator%d", (oper)
 - [gsttensor_transform.c](#), 986
- operator_func
 - [gsttensor_if.c](#), 657
- operators
 - [_GstTensorTransform](#), 217
- opnum
 - [gsttensor_decoder.c](#), 629
- option
 - [_GstTensorDecoder](#), 92
 - [_GstTensorMerge](#), 143
 - [_GstTensorTransform](#), 217
 - [_tensor_time_sync_data](#), 239
 - [gsttensor_decoder.c](#), 629
- org_ts
 - [_ntp_packet_t](#), 234
- out
 - [_GstTensorRate](#), 162
- out_combi_i
 - [_GstTensorFilterCombination](#), 104
- out_combi_i_defined
 - [_GstTensorFilterCombination](#), 104
- out_combi_o
 - [_GstTensorFilterCombination](#), 104
- out_combi_o_defined
 - [_GstTensorFilterCombination](#), 104
- out_config
 - [_GstTensorAggregator](#), 75
 - [_GstTensorFilterPrivate](#), 127
 - [_GstTensorIf](#), 139
 - [_GstTensorSparseDec](#), 177
 - [_GstTensorTrainer](#), 201
 - [_GstTensorTransform](#), 218
- out_frame_count
 - [_GstTensorRate](#), 163
- out_type
 - [_tensor_transform_arithmetic](#), 240
 - [_tensor_transform_stand](#), 244
- output_configured
 - [_GTensorFilterSingleClass](#), 221
 - [_GstTensorFilterProperties](#), 132
- output_layout
 - [_GstTensorFilterProperties](#), 132
- output_meta
 - [_GstTensorFilterProperties](#), 132
 - [_GstTensorTrainer](#), 201
- output_mode
 - [_GstTensorDebug](#), 88
- output_ranks
 - [_GstTensorFilterProperties](#), 133
- PACKAGE
 - [edge_elements.c](#), 350
 - [gstdatarepo.c](#), 294
 - [gstjoin.c](#), 387
 - [mqttelements.c](#), 408
 - [nnstreamer.c](#), 1355
- pad
 - [_tensor_transform_padding](#), 243
 - [GstTensorPad](#), 265
- padcount
 - [_GstJoin](#), 54
- PADDING_BACK
 - [gsttensor_transform.h](#), 994
- PADDING_BOTTOM
 - [gsttensor_transform.h](#), 994
- PADDING_END
 - [gsttensor_transform.h](#), 994
- PADDING_FRONT
 - [gsttensor_transform.h](#), 994
- PADDING_LEFT
 - [gsttensor_transform.h](#), 994
- PADDING_RIGHT
 - [gsttensor_transform.h](#), 994
- PADDING_TOP
 - [gsttensor_transform.h](#), 994
- parent
 - [_GTensorFilterSingleClass](#), 221
 - [_GstDataRepoSrc](#), 41
 - [_GstJoinPad](#), 56
 - [_GstJoinPadClass](#), 57
 - [_GstMqttSink](#), 66
 - [_GstMqttSrc](#), 71
 - [_GstTensorRepoSrc](#), 171
 - [GstTensorAllocator](#), 254
- parent_class
 - [_GstDataRepoSinkClass](#), 35
 - [_GstDataRepoSrcClass](#), 46
 - [_GstEdgeSinkClass](#), 50
 - [_GstEdgeSrcClass](#), 53

- [_GstJoinClass](#), 55
- [_GstMqttSinkClass](#), 67
- [_GstMqttSrcClass](#), 72
- [_GstTensorAggregatorClass](#), 77
- [_GstTensorConverterClass](#), 84
- [_GstTensorCropClass](#), 87
- [_GstTensorDebugClass](#), 89
- [_GstTensorDecoderClass](#), 93
- [_GstTensorDemuxClass](#), 100
- [_GstTensorFilterClass](#), 102
- [_GstTensorIfClass](#), 140
- [_GstTensorMergeClass](#), 145
- [_GstTensorMuxClass](#), 149
- [_GstTensorQueryClientClass](#), 154
- [_GstTensorQueryServerSinkClass](#), 156
- [_GstTensorQueryServerSrcClass](#), 160
- [_GstTensorRateClass](#), 165
- [_GstTensorRepoSinkClass](#), 168
- [_GstTensorRepoSrcClass](#), 172
- [_GstTensorSinkClass](#), 175
- [_GstTensorSparseDecClass](#), 178
- [_GstTensorSparseEncClass](#), 181
- [_GstTensorSplitClass](#), 185
- [_GstTensorSrcIOClass](#), 196
- [_GstTensorTrainerClass](#), 203
- [_GstTensorTransformClass](#), 219
- [GstTensorAllocatorClass](#), 254
- [parse_accl_args](#), 276
 - [auto_accl](#), 277
 - [def_accl](#), 277
 - [in_accl](#), 277
 - [sup_accl](#), 277
- [parse_accl_hw](#)
 - [nnstreamer_plugin_api_filter.h](#), 1032
- [parse_accl_hw_all](#)
 - [tensor_filter_common.c](#), 1489
- [parse_accl_hw_fill](#)
 - [nnstreamer_plugin_api_filter.h](#), 1041
 - [tensor_filter_common.c](#), 1490
- [parse_accl_hw_util](#)
 - [tensor_filter_common.c](#), 1490
- [parser](#)
 - [_GstDataRepoSrc](#), 41
- [path](#)
 - [subplugin_conf](#), 280
- [paths](#)
 - [subplugin_info_s](#), 281
- [pdata](#)
 - [GstTensorQueryEdgeInfo](#), 267
- [per_channel](#)
 - [_tensor_transform_stand](#), 244
- [per_channel_arith](#)
 - [_tensor_transform_arithmetic](#), 240
- [playing](#)
 - [_GstEdgeSrc](#), 52
 - [_GstTensorQueryServerSrc](#), 158
- [plugin_data](#)
 - [_GstTensorDecoder](#), 92
- [plugin_init](#)
 - [edge_elements.c](#), 350
 - [gstdatarepo.c](#), 295
 - [gstjoin.c](#), 398
 - [mqttelements.c](#), 408
- [poll](#)
 - [_ntp_packet_t](#), 234
- [poll_timeout](#)
 - [_GstTensorSrcIO](#), 190
- [port](#)
 - [_GstEdgeSink](#), 49
 - [_GstTensorQueryClient](#), 152
 - [_GstTensorQueryServerSrc](#), 158
 - [GstTensorQueryEdgeInfo](#), 267
- [pos](#)
 - [dump_buf](#), 251
- [pre_enabled](#)
 - [_GstTensorSrcIOChannelProperties](#), 194
- [precision](#)
 - [_ntp_packet_t](#), 234
- [prepare_statistics](#)
 - [tensor_filter.c](#), 1429
- [prev_ts](#)
 - [_GstTensorFilter](#), 101
 - [_GstTensorRate](#), 163
- [prevbuf](#)
 - [_GstTensorRate](#), 163
- [priv](#)
 - [_GstTensorFilterSingle](#), 219
 - [_GstTensorFilter](#), 101
- [priv_data](#)
 - [_GstTensorConverter](#), 81
- [privateData](#)
 - [_GstTensorFilterPrivate](#), 127
 - [_GstTensorTrainer](#), 201
- [PROCESS_SCANNED_DATA](#)
 - [gsttensor_srcio.c](#), 885, 913, 914
- [prop](#)
 - [_GstTensorFilterPrivate](#), 127
 - [_GstTensorTrainer](#), 201
 - [gsttensor_srcio.c](#), 914
- [PROP_0](#)
 - [edge_sink.c](#), 353
 - [edge_src.c](#), 370
 - [gstdatareposink.c](#), 304
 - [gstdatareposrc.c](#), 324
 - [gstjoin.c](#), 388
 - [gsttensor_aggregator.c](#), 496
 - [gsttensor_converter.c](#), 522
 - [gsttensor_crop.c](#), 572
 - [gsttensor_debug.c](#), 593
 - [gsttensor_decoder.c](#), 609
 - [gsttensor_demux.c](#), 637
 - [gsttensor_if.c](#), 658
 - [gsttensor_merge.c](#), 688
 - [gsttensor_mux.c](#), 712
 - [gsttensor_rate.c](#), 735
 - [gsttensor_reposink.c](#), 775

- gsttensor_reposrc.c, 791
- gsttensor_sink.c, 805
- gsttensor_sparsedec.c, 823
- gsttensor_sparseenc.c, 837
- gsttensor_split.c, 857
- gsttensor_srcii.c, 886
- gsttensor_trainer.c, 926
- gsttensor_transform.c, 962
- mqttsink.c, 413
- mqttsrc.c, 452
- tensor_filter_common.h, 1502
- tensor_query_client.c, 1565
- tensor_query_serversink.c, 1602
- tensor_query_serversrc.c, 1615
- PROP_ACCELERATION
 - gsttensor_transform.c, 962
- PROP_ACCELERATOR
 - tensor_filter_common.h, 1502
- PROP_ACTIVE_PAD
 - gstjoin.c, 388
- PROP_APPLY
 - gsttensor_transform.c, 962
- PROP_BASE_DIRECTORY
 - gsttensor_srcii.c, 886
- PROP_BUFFER_CAPACITY
 - gsttensor_srcii.c, 886
- PROP_CAP
 - gsttensor_debug.c, 593
- PROP_CAPS
 - gstdataareposrc.c, 324
 - gsttensor_reposrc.c, 791
- PROP_CHANNELS
 - gsttensor_srcii.c, 886
- PROP_CONCAT
 - gsttensor_aggregator.c, 496
- PROP_CONFIG
 - gsttensor_decoder.c, 609
 - tensor_filter_common.h, 1502
- PROP_CONNECT_TYPE
 - edge_sink.c, 353
 - edge_src.c, 370
 - tensor_query_client.c, 1565
 - tensor_query_serversink.c, 1602
 - tensor_query_serversrc.c, 1615
- PROP_CONNECTION_TIMEOUT
 - edge_sink.c, 353
- PROP_CUSTOM
 - tensor_filter_common.h, 1502
- PROP_CUSTOM_LIB
 - edge_sink.c, 353
 - edge_src.c, 370
- PROP_CV
 - gsttensor_if.c, 658
- PROP_CV_OPTION
 - gsttensor_if.c, 658
- PROP_DEBUG
 - mqttsink.c, 413
 - mqttsrc.c, 452
- PROP_DEST_HOST
 - edge_sink.c, 353
 - edge_src.c, 370
 - tensor_query_client.c, 1565
 - tensor_query_serversrc.c, 1615
- PROP_DEST_PORT
 - edge_sink.c, 353
 - edge_src.c, 370
 - tensor_query_client.c, 1565
 - tensor_query_serversrc.c, 1615
- PROP_DEV_DIRECTORY
 - gsttensor_srcii.c, 886
- PROP_DEVICE
 - gsttensor_srcii.c, 886
- PROP_DEVICE_NUM
 - gsttensor_srcii.c, 886
- PROP_DROP
 - gsttensor_rate.c, 735
- PROP_DUP
 - gsttensor_rate.c, 735
- PROP_ELSE
 - gsttensor_if.c, 658
- PROP_ELSE_OPTION
 - gsttensor_if.c, 658
- PROP_EMIT_SIGNAL
 - gsttensor_sink.c, 805
- PROP_EPOCHS
 - gstdataareposrc.c, 324
 - gsttensor_trainer.c, 926
- PROP_FRAMERATE
 - gsttensor_rate.c, 735
- PROP_FRAMES_DIMENSION
 - gsttensor_aggregator.c, 496
- PROP_FRAMES_FLUSH
 - gsttensor_aggregator.c, 496
- PROP_FRAMES_IN
 - gsttensor_aggregator.c, 496
- PROP_FRAMES_OUT
 - gsttensor_aggregator.c, 496
- PROP_FRAMES_PER_TENSOR
 - gsttensor_converter.c, 522
- PROP_FRAMEWORK
 - gsttensor_trainer.c, 926
 - tensor_filter_common.h, 1502
- PROP_FREQUENCY
 - gsttensor_srcii.c, 886
- PROP_HOST
 - edge_sink.c, 353
 - edge_src.c, 370
 - tensor_query_client.c, 1565
 - tensor_query_serversrc.c, 1615
- PROP_ID
 - tensor_query_serversink.c, 1602
 - tensor_query_serversrc.c, 1615
- PROP_IN
 - gsttensor_rate.c, 735
- PROP_INPUT
 - tensor_filter_common.h, 1502

- PROP_INPUT_DIMENSION
 - gsttensor_converter.c, [522](#)
- PROP_INPUT_TYPE
 - gsttensor_converter.c, [522](#)
- PROP_INPUTCOMBINATION
 - tensor_filter_common.h, [1502](#)
- PROP_INPUTLAYOUT
 - tensor_filter_common.h, [1502](#)
- PROP_INPUTNAME
 - tensor_filter_common.h, [1502](#)
- PROP_INPUTRANKS
 - tensor_filter_common.h, [1502](#)
- PROP_INPUTTYPE
 - tensor_filter_common.h, [1502](#)
- PROP_INVOKE_DYNAMIC
 - tensor_filter_common.h, [1502](#)
- PROP_IS_LIVE
 - mqttsrc.c, [452](#)
 - tensor_query_serversrc.c, [1615](#)
- PROP_IS_SHUFFLE
 - gstdataareposrc.c, [324](#)
- PROP_IS_UPDATABLE
 - tensor_filter_common.h, [1502](#)
- PROP_JSON
 - gstdataareposink.c, [304](#)
 - gstdataareposrc.c, [324](#)
- PROP_LAST
 - edge_sink.c, [353](#)
 - edge_src.c, [370](#)
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- PROP_LATENCY
 - tensor_filter_common.h, [1502](#)
- PROP_LATENCY_REPORT
 - tensor_filter_common.h, [1502](#)
- PROP_LATENESS
 - gsttensor_crop.c, [572](#)
- PROP_LOCATION
 - gstdataareposink.c, [304](#)
 - gstdataareposrc.c, [324](#)
- PROP_MAX_MSG_BUF_SIZE
 - mqttsink.c, [413](#)
- PROP_MAX_REQUEST
 - tensor_query_client.c, [1565](#)
- PROP_MERGE_CHANNELS
 - gsttensor_srciiio.c, [886](#)
- PROP_META
 - gsttensor_debug.c, [593](#)
- PROP_METALESS_FRAME_LIMIT
 - tensor_query_serversink.c, [1602](#)
- PROP_MODE
 - gsttensor_converter.c, [522](#)
 - gsttensor_decoder.c, [609](#)
 - gsttensor_merge.c, [688](#)
 - gsttensor_srciiio.c, [886](#)
 - gsttensor_transform.c, [962](#)
- PROP_MODE_OPTION
 - gsttensor_decoder.c, [608](#)
- PROP_MODE_OPTION1
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION2
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION3
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION4
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION5
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION6
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION7
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION8
 - gsttensor_decoder.c, [609](#)
- PROP_MODE_OPTION9
 - gsttensor_decoder.c, [609](#)
- PROP_MODEL
 - tensor_filter_common.h, [1502](#)
- PROP_MODEL_CONFIG
 - gsttensor_trainer.c, [926](#)
- PROP_MODEL_LOAD_PATH
 - gsttensor_trainer.c, [926](#)
- PROP_MODEL_SAVE_PATH
 - gsttensor_trainer.c, [926](#)
- PROP_MQTT_CLIENT_ID
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- PROP_MQTT_HOST_ADDRESS
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- PROP_MQTT_HOST_PORT
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- PROP_MQTT_NTP_SRVS
 - mqttsink.c, [413](#)
- PROP_MQTT_NTP_SYNC
 - mqttsink.c, [413](#)
- PROP_MQTT_OPT_CLEANSESSION
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- PROP_MQTT_OPT_KEEP_ALIVE_INTERVAL
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- PROP_MQTT_PUB_TOPIC
 - mqttsink.c, [413](#)
- PROP_MQTT_PUB_WAIT_TIMEOUT
 - mqttsink.c, [413](#)
- PROP_MQTT_QOS
 - mqttsink.c, [413](#)
 - mqttsrc.c, [452](#)
- PROP_MQTT_SUB_TIMEOUT
 - mqttsrc.c, [452](#)
- PROP_MQTT_SUB_TOPIC
 - mqttsrc.c, [452](#)
- PROP_N_PADS
 - gstjoin.c, [388](#)

- PROP_NUM_BUFFERS
 - mqttpsink.c, [413](#)
- PROP_NUM_INPUTS
 - gsttensor_trainer.c, [926](#)
- PROP_NUM_LABELS
 - gsttensor_trainer.c, [926](#)
- PROP_NUM_TRAINING_SAMPLES
 - gsttensor_trainer.c, [926](#)
- PROP_NUM_VALIDATION_SAMPLES
 - gsttensor_trainer.c, [926](#)
- PROP_OP
 - gsttensor_if.c, [658](#)
- PROP_OPTION
 - gsttensor_merge.c, [688](#)
 - gsttensor_transform.c, [962](#)
- PROP_OUT
 - gsttensor_rate.c, [735](#)
- PROP_OUTPUT
 - gsttensor_debug.c, [593](#)
 - tensor_filter_common.h, [1502](#)
- PROP_OUTPUTCOMBINATION
 - tensor_filter_common.h, [1502](#)
- PROP_OUTPUTLAYOUT
 - tensor_filter_common.h, [1502](#)
- PROP_OUTPUTNAME
 - tensor_filter_common.h, [1502](#)
- PROP_OUTPUTRANKS
 - tensor_filter_common.h, [1502](#)
- PROP_OUTPUTTYPE
 - tensor_filter_common.h, [1502](#)
- PROP_POLL_TIMEOUT
 - gsttensor_srcio.c, [886](#)
- PROP_PORT
 - edge_sink.c, [353](#)
 - edge_src.c, [370](#)
 - tensor_query_client.c, [1565](#)
 - tensor_query_serversrc.c, [1615](#)
- PROP_READ_OPTION
 - gsttensor_decoder.c, [608](#)
- PROP_SET_TIMESTAMP
 - gsttensor_converter.c, [522](#)
- PROP_SHARED_TENSOR_FILTER_KEY
 - tensor_filter_common.h, [1502](#)
- PROP_SIGNAL_RATE
 - gsttensor_reposink.c, [775](#)
 - gsttensor_sink.c, [805](#)
- PROP_SILENT
 - gsttensor_aggregator.c, [496](#)
 - gsttensor_converter.c, [522](#)
 - gsttensor_crop.c, [572](#)
 - gsttensor_debug.c, [593](#)
 - gsttensor_decoder.c, [609](#)
 - gsttensor_demux.c, [637](#)
 - gsttensor_if.c, [658](#)
 - gsttensor_merge.c, [688](#)
 - gsttensor_mux.c, [712](#)
 - gsttensor_rate.c, [735](#)
 - gsttensor_reposink.c, [775](#)
 - gsttensor_reposrc.c, [791](#)
 - gsttensor_sink.c, [805](#)
 - gsttensor_sparsedec.c, [823](#)
 - gsttensor_sparseenc.c, [837](#)
 - gsttensor_split.c, [857](#)
 - gsttensor_srcio.c, [886](#)
 - gsttensor_transform.c, [962](#)
 - tensor_filter_common.h, [1502](#)
 - tensor_query_client.c, [1565](#)
- PROP_SLOT
 - gsttensor_reposink.c, [775](#)
- PROP_SLOT_ID
 - gsttensor_reposrc.c, [791](#)
- PROP_START_SAMPLE_INDEX
 - gstdataareposrc.c, [324](#)
- PROP_STOP_SAMPLE_INDEX
 - gstdataareposrc.c, [324](#)
- PROP_SUBPLUGINS
 - gsttensor_converter.c, [522](#)
 - gsttensor_decoder.c, [609](#)
 - tensor_filter_common.h, [1502](#)
- PROP_SUSPEND
 - tensor_filter_common.h, [1502](#)
- PROP_SV
 - gsttensor_if.c, [658](#)
- PROP_SYNC_MODE
 - gsttensor_merge.c, [688](#)
 - gsttensor_mux.c, [712](#)
- PROP_SYNC_OPTION
 - gsttensor_merge.c, [688](#)
 - gsttensor_mux.c, [712](#)
- PROP_TENSORPICK
 - gsttensor_demux.c, [637](#)
 - gsttensor_split.c, [857](#)
- PROP_TENSORS_SEQUENCE
 - gstdataareposrc.c, [324](#)
- PROP_TENSORSEG
 - gsttensor_split.c, [857](#)
- PROP_THEN
 - gsttensor_if.c, [658](#)
- PROP_THEN_OPTION
 - gsttensor_if.c, [658](#)
- PROP_THROTTLE
 - gsttensor_rate.c, [735](#)
- PROP_THROUGHPUT
 - tensor_filter_common.h, [1502](#)
- PROP_TIMEOUT
 - tensor_query_client.c, [1565](#)
 - tensor_query_serversink.c, [1602](#)
 - tensor_query_serversrc.c, [1615](#)
- PROP_TOPIC
 - edge_sink.c, [353](#)
 - edge_src.c, [370](#)
 - tensor_query_client.c, [1565](#)
 - tensor_query_serversrc.c, [1615](#)
- PROP_TRANSPOSE_RANK_LIMIT
 - gsttensor_transform.c, [962](#)
- PROP_TRIGGER

- gsttensor_srcii.c, 886
- PROP_TRIGGER_NUM
 - gsttensor_srcii.c, 886
- PROP_WAIT_CONNECTION
 - edge_sink.c, 353
- pspec_drop
 - gsttensor_rate.c, 747
- pspec_duplicate
 - gsttensor_rate.c, 747
- pts
 - _GstMQTTMessageHdr, 59
- push_data
 - _GstTensorTrainerFramework, 207
- pushed
 - GstTensorRepoData, 272
- query_caps
 - _NNStreamerExternalConverter, 231
- query_client_id_t
 - tensor_meta.h, 1560
- QUERY_DEFAULT_TIMEOUT_SEC
 - tensor_query_common.h, 1582
- rate_d
 - _GstDataRepoSrc, 41
 - _GstTensorRate, 163
 - GstTensorsConfig, 274
- rate_n
 - _GstDataRepoSrc, 41
 - _GstTensorRate, 163
 - GstTensorsConfig, 274
- raw_template
 - gsttensor_crop.c, 584
- read_position
 - _GstDataRepoSrc, 41
- recent_latencies
 - _GstTensorFilterStatistics, 135
- record_statistics
 - tensor_filter.c, 1429
- recv_ts
 - _ntp_packet_t, 235
- ref_id
 - _ntp_packet_t, 235
- ref_ts
 - _ntp_packet_t, 235
- referred_list
 - GstTensorFilterSharedModelRepresentation, 259
- REGEX_ACCL_DELIMITER
 - tensor_filter_common.c, 1440
- REGEX_ACCL_ELEM_DELIMITER
 - tensor_filter_common.c, 1441
- REGEX_ACCL_ELEM_END
 - tensor_filter_common.c, 1441
- REGEX_ACCL_ELEM_PREFIX
 - tensor_filter_common.c, 1441
- REGEX_ACCL_ELEM_START
 - tensor_filter_common.c, 1441
- REGEX_ACCL_ELEM_SUFFIX
 - tensor_filter_common.c, 1441
- regex_accl_elem_utils
 - tensor_filter_common.c, 1494
- REGEX_ACCL_END
 - tensor_filter_common.c, 1441
- REGEX_ACCL_PREFIX
 - tensor_filter_common.c, 1442
- REGEX_ACCL_START
 - tensor_filter_common.c, 1442
- REGEX_ACCL_SUFFIX
 - tensor_filter_common.c, 1442
- regex_accl_utils
 - tensor_filter_common.c, 1495
- REGEX_ARITH_OPTION
 - gsttensor_transform.c, 960
- REGEX_ARITH_OPTION_TYPECAST
 - gsttensor_transform.c, 960
- REGEX_CLAMP_OPTION
 - gsttensor_transform.c, 960
- REGEX_DIMCHG_OPTION
 - gsttensor_transform.c, 960
- REGEX_PADDING_OPTION
 - gsttensor_transform.c, 960
- REGEX_STAND_OPTION
 - gsttensor_transform.c, 961
- REGEX_TRANSPOSE_OPTION
 - gsttensor_transform.c, 961
- REGEX_TYPECAST_OPTION
 - gsttensor_transform.c, 961
- region
 - tensor_crop_info_s, 282
- register_subplugin
 - nnstreamer_subplugin.c, 1329
 - nnstreamer_subplugin.h, 1339
- registerExternalConverter
 - gsttensor_converter.c, 552
 - nnstreamer_plugin_api_converter.h, 1019
- RELOAD_MODEL
 - nnstreamer_plugin_api_filter.h, 1035
- reloadModel
 - _GstTensorFilterFramework, 114
- remove_padding
 - _GstTensorConverter, 81
- repo_cond
 - GstTensorRepo, 270
- repo_lock
 - GstTensorRepo, 270
- requested_num
 - _GstTensorQueryClient, 152
- required_sample
 - _GstTensorTrainer, 201
- reserved
 - GstTensorExtraInfo, 258
- result
 - gsttensor_if.c, 677
- root_delay
 - _ntp_packet_t, 235
- root_dispersion
 - _ntp_packet_t, 235

- run_without_model
 - _GstTensorFilterFramework, 114
 - _GstTensorFilterFrameworkInfo, 122
- running_time
 - _GstDataRepoSrc, 42
- runtime_check_supported_accelerator
 - tensor_filter_common.c, 1492
- runtime_data, 278
 - model, 280
- S_ISDIR
 - gstdatareposrc.c, 323
- S_ISREG
 - gstdatareposrc.c, 323
- S_ISSOCK
 - gstdatareposrc.c, 323
- sample_offset_array
 - _GstDataRepoSink, 33
 - _GstDataRepoSrc, 42
- sample_offset_array_len
 - _GstDataRepoSrc, 42
- sample_size
 - _GstDataRepoSink, 34
 - _GstDataRepoSrc, 42
- SAMPLING_FREQUENCY
 - gsttensor_srcii.c, 885
- sampling_frequency
 - _GstTensorSrcIIO, 190
- scale
 - _GstTensorSrcIIOChannelProperties, 194
- SCALE_SUFFIX
 - gsttensor_srcii.c, 885
- scan_size
 - _GstTensorSrcIIO, 190
- searchAlgorithm
 - nnstreamer_subplugin.c, 1334
- sec
 - _ntp_timestamp_t, 236
- segment
 - _GstJoinPad, 56
 - _GstTensorConverter, 81
 - _GstTensorRate, 164
- segment_seqnum
 - _GstJoinPad, 56
- send_stream_start
 - _GstTensorCrop, 85
 - _GstTensorMerge, 143
 - _GstTensorMux, 147
- sent_qos_on_passthrough
 - _GstTensorRate, 164
- sent_time_epoch
 - _GstMQTTMessageHdr, 59
- SET_ACCELERATOR
 - nnstreamer_plugin_api_filter.h, 1035
- SET_INPUT_INFO
 - nnstreamer_plugin_api_filter.h, 1036
- set_input_info
 - _GstTensorFilterSingleClass, 221
- SET_INPUT_PROP
 - nnstreamer_plugin_api_filter.h, 1035
- SET_OUTPUT_PROP
 - nnstreamer_plugin_api_filter.h, 1035
- set_property_value
 - nnstreamer_plugin_api_impl.c, 1248
- set_startid
 - _GstTensorRepoSink, 167
 - _GstTensorRepoSrc, 171
- set_timestamp
 - _GstTensorConverter, 82
- setInputDim
 - _NNStreamer_custom_class, 228
- setInputDimension
 - _GstTensorFilterFramework, 114
- setOption
 - _GstTensorDecoderDef, 96
- shared_interpreter
 - GstTensorFilterSharedModelRepresentation, 259
- shared_model_table
 - tensor_filter_common.c, 1495
- shared_tensor_filter_key
 - _GstTensorFilterProperties, 133
- shift
 - _GstTensorSrcIIOChannelProperties, 195
- shuffled_index_array
 - _GstDataRepoSrc, 42
- SIGNAL_EOS
 - gsttensor_sink.c, 805
- SIGNAL_NEW_DATA
 - gsttensor_sink.c, 805
- signal_rate
 - _GstTensorRepoSink, 167
 - _GstTensorSink, 173
- SIGNAL_STREAM_START
 - gsttensor_sink.c, 805
- silent
 - _GstTensorAggregator, 75
 - _GstTensorConverter, 82
 - _GstTensorCrop, 85
 - _GstTensorDebug, 88
 - _GstTensorDecoder, 92
 - _GstTensorDemux, 99
 - _GstTensorFilterPrivate, 127
 - _GstTensorIf, 139
 - _GstTensorMerge, 144
 - _GstTensorMux, 147
 - _GstTensorQueryClient, 152
 - _GstTensorRate, 164
 - _GstTensorRepoSink, 167
 - _GstTensorRepoSrc, 171
 - _GstTensorSink, 174
 - _GstTensorSparseDec, 177
 - _GstTensorSparseEnc, 180
 - _GstTensorSplit, 183
 - _GstTensorSrcIIO, 191
 - _GstTensorTransform, 218
- silent_debug
 - tensor_common.h, 1365

- silent_debug_caps
 - tensor_common.h, 1365
- silent_debug_config
 - gsttensor_aggregator.c, 495
- silent_debug_info
 - tensor_filter_common.c, 1442
- silent_debug_timestamp
 - gsttensor_converter.c, 520
 - gsttensor_sink.c, 804
- SINK_CAPS_STRING
 - gsttensor_trainer.c, 925
- sink_changed
 - GstTensorRepoData, 272
- sink_client_id
 - mqttsink.c, 442
- sink_factory
 - gsttensor_debug.c, 598
 - gsttensor_decoder.c, 629
 - gsttensor_if.c, 677
 - gsttensor_rate.c, 748
 - gsttensor_transform.c, 986
 - tensor_filter.c, 1431
- sink_id
 - _GstTensorQueryServerSink, 155
 - _tensor_sync_basepad_data, 238
 - GstTensorRepoData, 272
- SINK_INITIALIZING
 - mqttsink.h, 447
- sink_pad_template
 - mqttsink.c, 443
- SINK_RENDER_EOS
 - mqttsink.h, 447
- SINK_RENDER_ERROR
 - mqttsink.h, 447
- SINK_RENDER_STOPPED
 - mqttsink.h, 447
- sink_templ
 - gsttensor_demux.c, 648
 - gsttensor_merge.c, 703
 - gsttensor_mux.c, 725
 - gsttensor_split.c, 868
- sink_template
 - gsttensor_aggregator.c, 511
 - gsttensor_sparsedec.c, 830
 - gsttensor_sparseenc.c, 844
 - gsttensor_trainer.c, 950
- sink_tensor_conf
 - _GstTensorSplit, 184
- sinkpad
 - _GstTensorAggregator, 76
 - _GstTensorConverter, 82
 - _GstTensorDebug, 88
 - _GstTensorDemux, 99
 - _GstTensorIf, 139
 - _GstTensorQueryClient, 152
 - _GstTensorSparseDec, 177
 - _GstTensorSparseEnc, 180
 - _GstTensorSplit, 184
 - _GstTensorTrainer, 202
- sinkpad_info
 - _GstTensorCrop, 85
- sinkpad_raw
 - _GstTensorCrop, 86
- sinktemplate
 - edge_sink.c, 363
 - gstdatareposink.c, 315
 - tensor_query_client.c, 1575
 - tensor_query_serversink.c, 1608
- size
 - dump_buf, 251
 - GstTensorMemory, 262
 - vstr_helper, 291
- size_mems
 - _GstMQTTMessageHdr, 59
- source
 - _NnstWatchdog, 232
- sparse_info
 - GstTensorMetaInfo, 264
- SRC_CAPS_STRING
 - gsttensor_trainer.c, 925
- src_changed
 - GstTensorRepoData, 272
- src_client_id
 - mqttsrc.c, 479
- src_factory
 - gsttensor_debug.c, 598
 - gsttensor_decoder.c, 630
 - gsttensor_if.c, 677
 - gsttensor_rate.c, 748
 - gsttensor_transform.c, 987
 - tensor_filter.c, 1431
- src_id
 - _GstTensorQueryServerSrc, 159
 - GstTensorRepoData, 273
- src_pad
 - _GstDataRepoSrc, 43
- src_pad_template
 - mqttsrc.c, 480
- src_templ
 - gsttensor_demux.c, 648
 - gsttensor_merge.c, 703
 - gsttensor_mux.c, 725
 - gsttensor_split.c, 868
- src_template
 - gsttensor_aggregator.c, 511
 - gsttensor_crop.c, 584
 - gsttensor_sparsedec.c, 830
 - gsttensor_sparseenc.c, 844
 - gsttensor_trainer.c, 950
- srcpad
 - _GstJoin, 55
 - _GstTensorAggregator, 76
 - _GstTensorConverter, 82
 - _GstTensorCrop, 86
 - _GstTensorDebug, 88
 - _GstTensorMerge, 144

- [_GstTensorMux](#), 147
 - [_GstTensorQueryClient](#), 153
 - [_GstTensorSparseDec](#), 177
 - [_GstTensorSparseEnc](#), 180
 - [_GstTensorTrainer](#), 202
- srcpads
 - [_GstTensorDemux](#), 99
 - [_GstTensorIf](#), 139
 - [_GstTensorSplit](#), 184
- srctemplate
 - [edge_src.c](#), 379
 - [gstdatareposrc.c](#), 341
 - [tensor_query_client.c](#), 1575
 - [tensor_query_serversrc.c](#), 1623
- STAND_DC_AVERAGE
 - [gsttensor_transform.h](#), 994
- STAND_DEFAULT
 - [gsttensor_transform.h](#), 994
- STAND_END
 - [gsttensor_transform.h](#), 994
- start
 - [_GTensorFilterSingleClass](#), 222
 - [_GstTensorTrainerFramework](#), 207
- start_offset
 - [_GstDataRepoSrc](#), 43
- start_sample_index
 - [_GstDataRepoSrc](#), 43
- stat
 - [_GstTensorFilterPrivate](#), 127
- statistics
 - [_GstTensorFilterFramework](#), 115
 - [_GstTensorFilterFrameworkInfo](#), 122
- stop
 - [_GTensorFilterSingleClass](#), 222
 - [_GstTensorTrainerFramework](#), 207
- stop_sample_index
 - [_GstDataRepoSrc](#), 43
- storage_bits
 - [_GstTensorSrcIOChannelProperties](#), 195
- storage_bytes
 - [_GstTensorSrcIOChannelProperties](#), 195
- STR_BOOL
 - [nnstreamer_conf.c](#), 1163
- stratum
 - [_ntp_packet_t](#), 235
- strcpy2
 - [tensor_filter_common.c](#), 1493
- stream_start
 - [_GstTensorSinkClass](#), 175
- STRING_CUSTOM_MODE
 - [gsttensor_converter.c](#), 521
- struct_stat
 - [gstdatareposrc.c](#), 323
- subplugin_conf, 280
 - files, 280
 - names, 280
 - path, 280
- subplugin_data
 - [_GstTensorFilterFramework](#), 115
 - [subplugin_get_custom_property_desc](#)
 - [nnstreamer_subplugin.c](#), 1331
 - [nnstreamer_subplugin.h](#), 1341
 - [subplugin_info_s](#), 281
 - names, 281
 - paths, 281
 - [subplugin_prefixes](#)
 - [nnstreamer_conf.c](#), 1179
 - [subplugin_set_custom_property_desc](#)
 - [nnstreamer_subplugin.c](#), 1332
 - [nnstreamer_subplugin.h](#), 1342
 - [subpluginData](#)
 - [nnstreamer_subplugin.c](#), 1334
 - [subplugins](#)
 - [nnstreamer_subplugin.c](#), 1334
 - [subpluginSearchLogic](#)
 - [nnstreamer_subplugin.c](#), 1324
 - [subpluginType](#)
 - [nnstreamer_subplugin.h](#), 1337
 - [successful_read](#)
 - [_GstDataRepoSrc](#), 43
 - [sup_accl](#)
 - [parse_accl_args](#), 277
 - SUPPORTED_AUDIO_FORMAT
 - [gstdatareposink.c](#), 302
 - SUPPORTED_VIDEO_FORMAT
 - [gstdatareposink.c](#), 302
 - [suspend](#)
 - [_GstTensorFilterProperties](#), 133
 - sv
 - [_GstTensorIf](#), 139
 - svtc_1
 - [gsttensor_if.c](#), 677
 - svtc_2
 - [gsttensor_if.c](#), 678
 - switch
 - [gsttensor_if.c](#), 676
 - sync
 - [_GstTensorMerge](#), 144
 - [_GstTensorMux](#), 148
 - SYNC_BASEPAD
 - [tensor_common.h](#), 1366
 - SYNC_END
 - [tensor_common.h](#), 1366
 - SYNC_NOSYNC
 - [tensor_common.h](#), 1366
 - SYNC_REFRESH
 - [tensor_common.h](#), 1366
 - SYNC_SLOWEST
 - [tensor_common.h](#), 1366
 - TAG_ERR_MQTTSINK
 - [mqtttsink.c](#), 443
 - TAG_ERR_MQTTSRC
 - [mqtttsrc.c](#), 480
 - TAG_NAME
 - [nnstreamer_log.h](#), 1203
 - TCP_DEFAULT_CLIENT_SRC_PORT

- tensor_query_client.c, 1564
- TCP_DEFAULT_HOST
 - tensor_query_client.c, 1564
- TCP_DEFAULT_SRV_SRC_PORT
 - tensor_query_client.c, 1564
- TCP_HIGHEST_PORT
 - tensor_query_client.c, 1564
- td_get_data
 - tensor_data.c, 1382
- td_set_data
 - tensor_data.c, 1382
- td_typecast
 - tensor_data.c, 1382
- td_typecast_to
 - tensor_data.c, 1383
- td_typecast_to_fromf16
 - tensor_data.c, 1383
- TDBG_CAP_DISABLED
 - gsttensor_debug.h, 603
- tdbg_cap_mode
 - gsttensor_debug.h, 602
- TDBG_CAP_SHOW_ALWAYS
 - gsttensor_debug.h, 603
- TDBG_CAP_SHOW_UPDATE
 - gsttensor_debug.h, 603
- TDBG_CAP_SHOW_UPDATE_F
 - gsttensor_debug.h, 603
- TDBG_META_DISABLED
 - gsttensor_debug.h, 603
- tdbg_meta_mode
 - gsttensor_debug.h, 603
- TDBG_META_QUERYSERVER
 - gsttensor_debug.h, 603
- TDBG_META_TIMESTAMP
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_CIRCULARBUF
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_CONSOLE_E
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_CONSOLE_I
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_CONSOLE_W
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_DISABLED
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_FILEWRITE
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_GSTDBG_E
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_GSTDBG_I
 - gsttensor_debug.h, 603
- TDBG_OUTPUT_GSTDBG_W
 - gsttensor_debug.h, 603
- tdbg_output_mode
 - gsttensor_debug.h, 603
- tensor_allocator.c
 - _alloc, 1358
 - G_DEFINE_TYPE, 1359
 - gst_tensor_alloc_init, 1359
 - GST_TENSOR_ALLOCATOR, 1358
 - gst_tensor_allocator_alignment, 1361
 - gst_tensor_allocator_class_init, 1360
 - gst_tensor_allocator_get_type, 1360
 - gst_tensor_allocator_init, 1360
- TensorCaps
 - gstdatarepositor.c, 303
- tensor_common.h
 - gst_tensor_aggregation_clear, 1366
 - gst_tensor_aggregation_clear_all, 1367
 - gst_tensor_aggregation_get_adapter, 1368
 - gst_tensor_aggregation_init, 1369
 - gst_tensor_buffer_from_config, 1370
 - gst_tensor_pad_caps_from_config, 1371
 - gst_tensor_pad_caps_is_flexible, 1364
 - gst_tensor_pad_caps_is_sparse, 1364
 - gst_tensor_pad_caps_is_static, 1364
 - gst_tensor_pad_get_format, 1372
 - gst_tensor_pad_possible_caps_from_config, 1373
 - gst_tensor_parse_config_file, 1374
 - gst_tensor_time_sync_buffer_from_collectpad, 1375
 - gst_tensor_time_sync_flush, 1376
 - gst_tensor_time_sync_get_current_time, 1377
 - gst_tensor_time_sync_get_mode, 1378
 - gst_tensor_time_sync_get_mode_string, 1379
 - gst_tensor_time_sync_set_option_data, 1380
 - nns_memcpy, 1364
 - nns_memset, 1365
 - silent_debug, 1365
 - silent_debug_caps, 1365
 - SYNC_BASEPAD, 1366
 - SYNC_END, 1366
 - SYNC_NOSYNC, 1366
 - SYNC_REFRESH, 1366
 - SYNC_SLOWEST, 1366
 - tensor_sync_basepad_data, 1366
 - tensor_time_sync_data, 1366
 - tensor_time_sync_mode, 1366
- tensor_config
 - _GstTensorDecoder, 92
- tensor_configured
 - _GstTensorAggregator, 76
- tensor_converter_custom
 - tensor_converter_custom.h, 1125
- tensor_converter_custom.h
 - nstreamer_converter_custom_register, 1123
 - nstreamer_converter_custom_unregister, 1124
 - tensor_converter_custom, 1125
- tensor_converter_mode
 - gsttensor_converter.h, 556
- tensor_count_array
 - _GstDataRepoSink, 34
 - _GstDataRepoSrc, 44
- tensor_count_array_len
 - _GstDataRepoSrc, 44
- tensor_crop_info_s, 282

- num, [282](#)
- region, [282](#)
- tensor_data.c
 - gst_tensor_data_get, [1383](#)
 - gst_tensor_data_raw_average, [1384](#)
 - gst_tensor_data_raw_average_per_channel, [1385](#)
 - gst_tensor_data_raw_std, [1386](#)
 - gst_tensor_data_raw_std_per_channel, [1387](#)
 - gst_tensor_data_raw_typecast, [1388](#)
 - gst_tensor_data_typecast, [1389](#)
 - td_get_data, [1382](#)
 - td_set_data, [1382](#)
 - td_typecast, [1382](#)
 - td_typecast_to, [1383](#)
 - td_typecast_to_fromf16, [1383](#)
 - while, [1390](#)
- tensor_data.h
 - gst_tensor_data_get, [1392](#)
 - gst_tensor_data_raw_average, [1392](#)
 - gst_tensor_data_raw_average_per_channel, [1393](#)
 - gst_tensor_data_raw_std, [1394](#)
 - gst_tensor_data_raw_std_per_channel, [1395](#)
 - gst_tensor_data_raw_typecast, [1396](#)
 - gst_tensor_data_set, [1397](#)
 - gst_tensor_data_typecast, [1398](#)
- tensor_data_s, [283](#)
 - data, [283](#)
 - type, [284](#)
- tensor_debug_cap_get_type
 - gsttensor_debug.c, [597](#)
- tensor_debug_meta_flags_get_type
 - gsttensor_debug.c, [598](#)
- tensor_debug_output_flags_get_type
 - gsttensor_debug.c, [598](#)
- TENSOR_DEBUG_TYPE_CAPS
 - gsttensor_debug.c, [592](#)
- TENSOR_DEBUG_TYPE_META_FLAGS
 - gsttensor_debug.c, [592](#)
- TENSOR_DEBUG_TYPE_OUTPUT_FLAGS
 - gsttensor_debug.c, [592](#)
- tensor_decoder_custom
 - tensor_decoder_custom.h, [1129](#)
- tensor_decoder_custom.h
 - nnstreamer_decoder_custom_register, [1127](#)
 - nnstreamer_decoder_custom_unregister, [1128](#)
 - tensor_decoder_custom, [1129](#)
- tensor_dim
 - tensor_typedef.h, [1153](#)
- tensor_element, [284](#)
 - _double, [284](#)
 - _float, [285](#)
 - _int16_t, [285](#)
 - _int32_t, [285](#)
 - _int64_t, [285](#)
 - _int8_t, [285](#)
 - _uint16_t, [285](#)
 - _uint32_t, [286](#)
 - _uint64_t, [286](#)
 - _uint8_t, [286](#)
- tensor_element_size
 - nnstreamer_plugin_api_util_impl.c, [1321](#)
- tensor_element_typename
 - nnstreamer_plugin_api_util_impl.c, [1321](#)
- tensor_filter.c
 - _gst_tensor_filter_convert_meta, [1404](#)
 - _gst_tensor_filter_release_mem_until_idx, [1405](#)
 - _gst_tensor_filter_transform_check_invoke_result, [1405](#)
 - _gst_tensor_filter_transform_get_all_input_data, [1406](#)
 - _gst_tensor_filter_transform_get_invoke_tensors, [1406](#)
 - _gst_tensor_filter_transform_get_output_data, [1407](#)
 - _gst_tensor_filter_transform_prepare_output_tensors, [1408](#)
 - _gst_tensor_filter_transform_update_outbuf, [1408](#)
 - _gst_tensor_filter_transform_validate, [1409](#)
 - accumulate_latency, [1410](#)
 - CAPS_STRING, [1402](#)
 - EVENT_NAME_UPDATE_MODEL, [1402](#)
 - FilterTransformData, [1404](#)
 - G_DEFINE_TYPE, [1410](#)
 - GST_CAT_DEFAULT, [1403](#)
 - GST_DEBUG_CATEGORY_STATIC, [1411](#)
 - gst_tensor_filter_check_throttling_delay, [1411](#)
 - gst_tensor_filter_class_init, [1411](#)
 - gst_tensor_filter_configure_tensor, [1412](#)
 - gst_tensor_filter_destroy_notify, [1414](#)
 - gst_tensor_filter_finalize, [1415](#)
 - gst_tensor_filter_fixate_caps, [1415](#)
 - gst_tensor_filter_get_property, [1416](#)
 - gst_tensor_filter_get_tensor_size, [1417](#)
 - gst_tensor_filter_get_wrapped_mem, [1418](#)
 - gst_tensor_filter_init, [1419](#)
 - gst_tensor_filter_parent_class, [1403](#)
 - gst_tensor_filter_query, [1419](#)
 - gst_tensor_filter_set_caps, [1420](#)
 - gst_tensor_filter_set_property, [1421](#)
 - gst_tensor_filter_sink_event, [1422](#)
 - gst_tensor_filter_src_event, [1423](#)
 - gst_tensor_filter_start, [1423](#)
 - gst_tensor_filter_stop, [1424](#)
 - gst_tensor_filter_transform, [1425](#)
 - gst_tensor_filter_transform_caps, [1426](#)
 - gst_tensor_filter_transform_size, [1428](#)
 - gst_tensor_filter_watchdog_trigger, [1428](#)
 - LATENCY_REPORT_HEADROOM, [1403](#)
 - LATENCY_REPORT_THRESHOLD, [1403](#)
 - prepare_statistics, [1429](#)
 - record_statistics, [1429](#)
 - sink_factory, [1431](#)
 - src_factory, [1431](#)
 - TF_MODELNAME, [1404](#)
 - THRESHOLD_CACHE_OLD, [1404](#)
 - THRESHOLD_DROP_OLD, [1404](#)

- track_latency, 1430
- tensor_filter.h
 - GST_IS_TENSOR_FILTER, 1433
 - GST_IS_TENSOR_FILTER_CLASS, 1434
 - GST_TENSOR_FILTER, 1434
 - GST_TENSOR_FILTER_CAST, 1434
 - GST_TENSOR_FILTER_CLASS, 1434
 - gst_tensor_filter_get_type, 1435
 - GST_TYPE_TENSOR_FILTER, 1434
 - GstTensorFilter, 1435
 - GstTensorFilterClass, 1435
- tensor_filter_common.c
 - _detect_framework_from_config, 1443
 - _gtfc_setprop_ACCELERATOR, 1443
 - _gtfc_setprop_CUSTOM, 1444
 - _gtfc_setprop_DIMENSION, 1444
 - _gtfc_setprop_FRAMEWORK, 1445
 - _gtfc_setprop_INPUTCOMBINATION, 1446
 - _gtfc_setprop_IS_UPDATABLE, 1446
 - _gtfc_setprop_LATENCY, 1447
 - _gtfc_setprop_LAYOUT, 1447
 - _gtfc_setprop_MODEL, 1448
 - _gtfc_setprop_NAME, 1449
 - _gtfc_setprop_OUTPUTCOMBINATION, 1450
 - _gtfc_setprop_PROP_INVOKE_DYNAMIC, 1450
 - _gtfc_setprop_SHARED_TENSOR_FILTER_KEY, 1450
 - _gtfc_setprop_SUSPEND, 1451
 - _gtfc_setprop_THROUGHPUT, 1451
 - _gtfc_setprop_TYPE, 1452
 - accl_hw_get_type, 1452
 - add_basic_supported_accelerators, 1453
 - create_regex, 1453
 - filter_supported_accelerators, 1454
 - g_free_const, 1440
 - G_LOCK_DEFINE_STATIC, 1455
 - g_strfreev_const, 1440
 - get_accl_hw_str, 1455
 - get_accl_hw_type, 1456
 - gst_tensor_filter_allocate_in_invoke, 1457
 - gst_tensor_filter_check_hw_availability, 1458
 - gst_tensor_filter_common_close_fw, 1458
 - gst_tensor_filter_common_free_property, 1459
 - gst_tensor_filter_common_get_combined_in_info, 1460
 - gst_tensor_filter_common_get_combined_out_info, 1460
 - gst_tensor_filter_common_get_out_info, 1461
 - gst_tensor_filter_common_get_property, 1462
 - gst_tensor_filter_common_init_property, 1463
 - gst_tensor_filter_common_open_fw, 1464
 - gst_tensor_filter_common_set_property, 1464
 - gst_tensor_filter_common_unload_fw, 1465
 - gst_tensor_filter_destroy_notify_util, 1466
 - gst_tensor_filter_detect_framework, 1467
 - gst_tensor_filter_framework_info_init, 1468
 - gst_tensor_filter_get_available_framework, 1468
 - gst_tensor_filter_get_dimension_string, 1469
 - gst_tensor_filter_get_layout_string, 1470
 - gst_tensor_filter_get_name_string, 1471
 - gst_tensor_filter_get_rank_string, 1472
 - gst_tensor_filter_get_type_string, 1472
 - gst_tensor_filter_install_properties, 1473
 - gst_tensor_filter_load_tensor_info, 1474
 - gst_tensor_filter_parse_accelerator, 1475
 - gst_tensor_filter_parse_modelpaths_string, 1476
 - gst_tensor_filter_properties_init, 1477
 - gst_tensor_filter_property_to_string, 1477
 - gst_tensor_filter_statistics_init, 1478
 - gst_tensor_get_layout_string, 1478
 - gst_tensor_parse_layout_string, 1479
 - gst_tensors_layout_init, 1479
 - gst_tensors_parse_layouts_string, 1480
 - gst_tensors_rank_init, 1481
 - gst_tensorsinfo_compare_print, 1481
 - gst_tensorsinfo_compare_to_string, 1482
 - nnstreamer_filter_exit, 1483
 - nnstreamer_filter_find, 1483
 - nnstreamer_filter_find_best_fit, 1484
 - nnstreamer_filter_probe, 1485
 - nnstreamer_filter_set_custom_property_desc, 1486
 - nnstreamer_filter_shared_model_get, 1486
 - nnstreamer_filter_shared_model_insert_and_get, 1487
 - nnstreamer_filter_shared_model_remove, 1487
 - nnstreamer_filter_shared_model_replace, 1488
 - nnstreamer_filter_validate, 1488
 - parse_accl_hw_all, 1489
 - parse_accl_hw_fill, 1490
 - parse_accl_hw_util, 1490
 - REGEX_ACCL_DELIMITER, 1440
 - REGEX_ACCL_ELEM_DELIMITER, 1441
 - REGEX_ACCL_ELEM_END, 1441
 - REGEX_ACCL_ELEM_PREFIX, 1441
 - REGEX_ACCL_ELEM_START, 1441
 - REGEX_ACCL_ELEM_SUFFIX, 1441
 - regex_accl_elem_utils, 1494
 - REGEX_ACCL_END, 1441
 - REGEX_ACCL_PREFIX, 1442
 - REGEX_ACCL_START, 1442
 - REGEX_ACCL_SUFFIX, 1442
 - regex_accl_utils, 1495
 - runtime_check_supported_accelerator, 1492
 - shared_model_table, 1495
 - silent_debug_info, 1442
 - strcpy2, 1493
 - verify_model_path, 1494
- tensor_filter_common.h
 - DBG, 1498
 - gst_tensor_filter_allocate_in_invoke, 1502
 - gst_tensor_filter_check_hw_availability, 1503
 - gst_tensor_filter_common_close_fw, 1504
 - gst_tensor_filter_common_free_property, 1504
 - gst_tensor_filter_common_get_combined_in_info, 1505

- gst_tensor_filter_common_get_combined_out_info, 1506
- gst_tensor_filter_common_get_out_info, 1507
- gst_tensor_filter_common_get_property, 1507
- gst_tensor_filter_common_init_property, 1509
- gst_tensor_filter_common_open_fw, 1509
- gst_tensor_filter_common_set_property, 1510
- gst_tensor_filter_common_unload_fw, 1511
- gst_tensor_filter_destroy_notify_util, 1511
- gst_tensor_filter_detect_framework, 1512
- gst_tensor_filter_install_properties, 1513
- gst_tensor_filter_load_tensor_info, 1514
- gst_tensor_filter_v0_call, 1499
- gst_tensor_filter_v1_call, 1499
- gst_tensorsinfo_compare_print, 1515
- gst_tensorsinfo_compare_to_string, 1516
- GST_TF_FW_INVOKE_COMPAT, 1499
- GST_TF_FW_V0, 1500
- GST_TF_FW_V1, 1500
- GST_TF_FW_VN, 1500
- GST_TF_STAT_MAX_RECENT, 1500
- GstTensorFilterCombination, 1501
- GstTensorFilterPrivate, 1501
- GstTensorFilterStatistics, 1501
- PROP_0, 1502
- PROP_ACCELERATOR, 1502
- PROP_CONFIG, 1502
- PROP_CUSTOM, 1502
- PROP_FRAMEWORK, 1502
- PROP_INPUT, 1502
- PROP_INPUTCOMBINATION, 1502
- PROP_INPUTLAYOUT, 1502
- PROP_INPUTNAME, 1502
- PROP_INPUTRANKS, 1502
- PROP_INPUTTYPE, 1502
- PROP_INVOKE_DYNAMIC, 1502
- PROP_IS_UPDATABLE, 1502
- PROP_LATENCY, 1502
- PROP_LATENCY_REPORT, 1502
- PROP_MODEL, 1502
- PROP_OUTPUT, 1502
- PROP_OUTPUTCOMBINATION, 1502
- PROP_OUTPUTLAYOUT, 1502
- PROP_OUTPUTNAME, 1502
- PROP_OUTPUTRANKS, 1502
- PROP_OUTPUTTYPE, 1502
- PROP_SHARED_TENSOR_FILTER_KEY, 1502
- PROP_SILENT, 1502
- PROP_SUBPLUGINS, 1502
- PROP_SUSPEND, 1502
- PROP_THROUGHPUT, 1502
- tensor_filter_custom.c
 - custom_allocateInInvoke, 1520
 - custom_checkAvailability, 1520
 - custom_close, 1520
 - custom_destroyNotify, 1521
 - custom_getInputDim, 1521
 - custom_getOutputDim, 1522
 - custom_invoke, 1522
 - custom_loadlib, 1522
 - custom_open, 1523
 - custom_setInputDim, 1524
- filter_subplugin_custom, 1525
- fini_filter_custom, 1524
- init_filter_custom, 1524
- internal_data, 1519
- NNS_support_custom, 1525
- tensor_filter_custom.h
 - NNS_custom_allocate_invoke, 1132
 - NNS_custom_destroy_notify, 1132
 - NNS_custom_exit_func, 1133
 - NNS_custom_get_input_dimension, 1133
 - NNS_custom_get_output_dimension, 1133
 - NNS_custom_init_func, 1134
 - NNS_custom_invoke, 1134
 - NNS_custom_set_input_dimension, 1135
 - NNStreamer_custom, 1135
 - NNStreamer_custom_class, 1135
- tensor_filter_custom_easy.c
 - custom_close, 1528
 - custom_eventHandler, 1528
 - custom_free_internal_data, 1528
 - custom_getFrameworkInfo, 1529
 - custom_getModelInfo, 1529
 - custom_invoke, 1530
 - custom_open, 1530
 - fini_filter_custom_easy, 1531
 - init_filter_custom_easy, 1531
 - internal_data, 1534
 - NNS_custom_easy_dynamic_register, 1532
 - NNS_custom_easy_register, 1532
 - NNS_custom_easy_unregister, 1533
 - NNS_support_custom_easy, 1534
- tensor_filter_custom_easy.h
 - NNS_custom_easy_dynamic_register, 1139
 - NNS_custom_easy_register, 1140
 - NNS_custom_easy_unregister, 1141
 - NNS_custom_invoke_dynamic, 1138
- tensor_filter_single.c
 - G_DEFINE_TYPE_WITH_PRIVATE, 1537
 - g_tensor_filter_allocate_in_invoke, 1537
 - g_tensor_filter_destroy_notify, 1538
 - g_tensor_filter_input_configured, 1539
 - g_tensor_filter_output_configured, 1539
 - g_tensor_filter_set_input_info, 1539
 - g_tensor_filter_single_class_init, 1540
 - g_tensor_filter_single_finalize, 1541
 - g_tensor_filter_single_get_property, 1542
 - g_tensor_filter_single_init, 1543
 - g_tensor_filter_single_invoke, 1543
 - g_tensor_filter_single_parent_class, 1537
 - G_TENSOR_FILTER_SINGLE_PRIV, 1537
 - g_tensor_filter_single_set_property, 1544
 - g_tensor_filter_single_start, 1545
 - g_tensor_filter_single_stop, 1546
 - GTensorFilterSinglePrivate, 1537

- tensor_filter_single.h
 - G_IS_TENSOR_FILTER_SINGLE, 1549
 - G_IS_TENSOR_FILTER_SINGLE_CLASS, 1549
 - G_TENSOR_FILTER_SINGLE, 1549
 - G_TENSOR_FILTER_SINGLE_CAST, 1550
 - G_TENSOR_FILTER_SINGLE_CLASS, 1550
 - g_tensor_filter_single_get_type, 1551
 - G_TYPE_TENSOR_FILTER_SINGLE, 1550
 - GTensorFilterSingle, 1550
 - GTensorFilterSingleClass, 1550
- tensor_filter_support_cc.cc
 - _RETURN_ERR_WITH_MSG, 1552
 - _SANITY_CHECK, 1552
 - GET_TFSP_WITH_CHECKS, 1553
 - NO_ANONYMOUS_NESTED_STRUCT, 1553
- tensor_format
 - tensor_typedef.h, 1153
- tensor_format_name
 - nnstreamer_plugin_api_util_impl.c, 1322
- tensor_if.h
 - nnstreamer_if_custom_register, 1144
 - nnstreamer_if_custom_unregister, 1144
 - tensor_if_custom, 1145
- tensor_if_behavior
 - gsttensor_if.h, 683
- tensor_if_compared_value
 - gsttensor_if.h, 683
- tensor_if_custom
 - tensor_if.h, 1145
- tensor_if_operator
 - gsttensor_if.h, 684
- tensor_if_srcpads
 - gsttensor_if.h, 684
- tensor_if_sv_s, 286
 - data, 287
 - num, 287
 - type, 287
- tensor_layout
 - tensor_typedef.h, 1153
- tensor_merge_linear
 - gsttensor_merge.h, 707
- tensor_merge_linear_mode
 - gsttensor_merge.h, 707
- tensor_merge_mode
 - gsttensor_merge.h, 708
- tensor_meta.c
 - gst_meta_query_api_get_type, 1555
 - gst_meta_query_free, 1555
 - gst_meta_query_get_info, 1555
 - gst_meta_query_init, 1556
 - gst_meta_query_transform, 1556
- tensor_meta.h
 - gst_buffer_add_meta_query, 1558
 - gst_buffer_get_meta_query, 1559
 - gst_meta_query_api_get_type, 1559
 - GST_META_QUERY_API_TYPE, 1559
 - gst_meta_query_get_info, 1559
 - GST_META_QUERY_INFO, 1559
 - query_client_id_t, 1560
- tensor_query_client.c
 - _nns_edge_event_cb, 1565
 - _nns_edge_parse_caps, 1566
 - DBG, 1563
 - DEFAULT_CLIENT_TIMEOUT, 1563
 - DEFAULT_MAX_REQUEST, 1563
 - DEFAULT_SILENT, 1563
 - G_DEFINE_TYPE, 1566
 - GST_CAT_DEFAULT, 1563
 - GST_DEBUG_CATEGORY_STATIC, 1566
 - gst_tensor_query_client_chain, 1567
 - gst_tensor_query_client_class_init, 1567
 - gst_tensor_query_client_create_edge_handle, 1568
 - gst_tensor_query_client_finalize, 1569
 - gst_tensor_query_client_get_property, 1569
 - gst_tensor_query_client_init, 1570
 - gst_tensor_query_client_parent_class, 1563
 - gst_tensor_query_client_query_caps, 1571
 - gst_tensor_query_client_set_property, 1571
 - gst_tensor_query_client_sink_event, 1572
 - gst_tensor_query_client_sink_query, 1573
 - gst_tensor_query_client_update_caps, 1574
 - PROP_0, 1565
 - PROP_CONNECT_TYPE, 1565
 - PROP_DEST_HOST, 1565
 - PROP_DEST_PORT, 1565
 - PROP_HOST, 1565
 - PROP_MAX_REQUEST, 1565
 - PROP_PORT, 1565
 - PROP_SILENT, 1565
 - PROP_TIMEOUT, 1565
 - PROP_TOPIC, 1565
 - sinktemplate, 1575
 - srctemplate, 1575
 - TCP_DEFAULT_CLIENT_SRC_PORT, 1564
 - TCP_DEFAULT_HOST, 1564
 - TCP_DEFAULT_SRV_SRC_PORT, 1564
 - TCP_HIGHEST_PORT, 1564
- tensor_query_client.h
 - GST_IS_TENSOR_QUERY_CLIENT, 1577
 - GST_IS_TENSOR_QUERY_CLIENT_CLASS, 1577
 - GST_TENSOR_QUERY_CLIENT, 1577
 - GST_TENSOR_QUERY_CLIENT_CAST, 1577
 - GST_TENSOR_QUERY_CLIENT_CLASS, 1578
 - gst_tensor_query_client_get_type, 1578
 - GST_TYPE_TENSOR_QUERY_CLIENT, 1578
 - GstTensorQueryClient, 1578
 - GstTensorQueryClientClass, 1578
- tensor_query_common.c
 - EREMOTEIO, 1580
 - gst_tensor_query_get_connect_type, 1580
- tensor_query_common.h
 - DEFAULT_CONNECT_TYPE, 1582
 - DEFAULT_HOST, 1582
 - gst_tensor_query_get_connect_type, 1582

- GST_TYPE_QUERY_CONNECT_TYPE, 1582
- QUERY_DEFAULT_TIMEOUT_SEC, 1582
- tensor_query_server.c
 - _qs_table, 1591
 - fini_queryserver, 1584
 - G_LOCK_DEFINE_STATIC, 1584
 - gst_tensor_query_server_add_data, 1585
 - gst_tensor_query_server_get_handle, 1585
 - gst_tensor_query_server_prepare, 1586
 - gst_tensor_query_server_release_edge_handle, 1587
 - gst_tensor_query_server_remove_data, 1587
 - gst_tensor_query_server_send_buffer, 1588
 - gst_tensor_query_server_set_caps, 1588
 - gst_tensor_query_server_set_configured, 1589
 - gst_tensor_query_server_wait_sink, 1590
 - init_queryserver, 1590
- tensor_query_server.h
 - DEFAULT_QUERY_INFO_TIMEOUT, 1593
 - DEFAULT_SERVER_ID, 1593
 - gst_tensor_query_server_add_data, 1593
 - gst_tensor_query_server_prepare, 1594
 - gst_tensor_query_server_release_edge_handle, 1595
 - gst_tensor_query_server_remove_data, 1596
 - gst_tensor_query_server_send_buffer, 1596
 - gst_tensor_query_server_set_caps, 1597
 - gst_tensor_query_server_set_configured, 1598
 - gst_tensor_query_server_wait_sink, 1598
- tensor_query_serversink.c
 - _gst_tensor_query_serversink_playing, 1602
 - _gst_tensor_query_serversink_start, 1602
 - DEFAULT_METALESS_FRAME_LIMIT, 1601
 - G_DEFINE_TYPE, 1603
 - GST_CAT_DEFAULT, 1601
 - GST_DEBUG_CATEGORY_STATIC, 1603
 - gst_tensor_query_serversink_change_state, 1603
 - gst_tensor_query_serversink_class_init, 1604
 - gst_tensor_query_serversink_finalize, 1605
 - gst_tensor_query_serversink_get_property, 1605
 - gst_tensor_query_serversink_init, 1606
 - gst_tensor_query_serversink_parent_class, 1601
 - gst_tensor_query_serversink_render, 1606
 - gst_tensor_query_serversink_set_caps, 1607
 - gst_tensor_query_serversink_set_property, 1607
 - PROP_0, 1602
 - PROP_CONNECT_TYPE, 1602
 - PROP_ID, 1602
 - PROP_METALESS_FRAME_LIMIT, 1602
 - PROP_TIMEOUT, 1602
 - sinktemplate, 1608
- tensor_query_serversink.h
 - GST_IS_TENSOR_QUERY_SERVERSINK, 1610
 - GST_IS_TENSOR_QUERY_SERVERSINK_CLASS, 1610
 - GST_TENSOR_QUERY_SERVERSINK, 1610
 - GST_TENSOR_QUERY_SERVERSINK_CAST, 1611
- GST_TENSOR_QUERY_SERVERSINK_CLASS, 1611
- gst_tensor_query_serversink_get_type, 1612
- GST_TYPE_TENSOR_QUERY_SERVERSINK, 1611
- GstTensorQueryServerSink, 1611
- GstTensorQueryServerSinkClass, 1611
- tensor_query_serversrc.c
 - _gst_tensor_query_serversrc_get_buffer, 1616
 - _gst_tensor_query_serversrc_playing, 1616
 - _gst_tensor_query_serversrc_start, 1617
 - _nns_edge_event_cb, 1617
 - DEFAULT_DATA_POP_TIMEOUT, 1614
 - DEFAULT_IS_LIVE, 1614
 - DEFAULT_MQTT_HOST, 1614
 - DEFAULT_MQTT_PORT, 1614
 - DEFAULT_PORT_SRC, 1615
 - G_DEFINE_TYPE, 1618
 - GST_CAT_DEFAULT, 1615
 - GST_DEBUG_CATEGORY_STATIC, 1618
 - gst_tensor_query_serversrc_change_state, 1618
 - gst_tensor_query_serversrc_class_init, 1619
 - gst_tensor_query_serversrc_create, 1620
 - gst_tensor_query_serversrc_finalize, 1620
 - gst_tensor_query_serversrc_get_property, 1621
 - gst_tensor_query_serversrc_init, 1621
 - gst_tensor_query_serversrc_parent_class, 1615
 - gst_tensor_query_serversrc_set_caps, 1622
 - gst_tensor_query_serversrc_set_property, 1622
 - PROP_0, 1615
 - PROP_CONNECT_TYPE, 1615
 - PROP_DEST_HOST, 1615
 - PROP_DEST_PORT, 1615
 - PROP_HOST, 1615
 - PROP_ID, 1615
 - PROP_IS_LIVE, 1615
 - PROP_PORT, 1615
 - PROP_TIMEOUT, 1615
 - PROP_TOPIC, 1615
 - srctemplate, 1623
- tensor_query_serversrc.h
 - GST_IS_TENSOR_QUERY_SERVERSRC, 1625
 - GST_IS_TENSOR_QUERY_SERVERSRC_CLASS, 1625
 - GST_TENSOR_QUERY_SERVERSRC, 1626
 - GST_TENSOR_QUERY_SERVERSRC_CAST, 1626
 - GST_TENSOR_QUERY_SERVERSRC_CLASS, 1626
 - gst_tensor_query_serversrc_get_type, 1627
 - GST_TYPE_TENSOR_QUERY_SERVERSRC, 1626
 - GstTensorQueryServerSrc, 1626
 - GstTensorQueryServerSrcClass, 1627
- tensor_region_s, 287
 - h, 288
 - w, 288
 - x, 288

- y, [289](#)
- tensor_size_array
 - [_GstDataRepoSink](#), [34](#)
 - [_GstDataRepoSrc](#), [44](#)
- tensor_size_array_len
 - [_GstDataRepoSrc](#), [44](#)
- tensor_sync_basepad_data
 - [tensor_common.h](#), [1366](#)
- tensor_time_sync_data
 - [tensor_common.h](#), [1366](#)
- tensor_time_sync_mode
 - [tensor_common.h](#), [1366](#)
- tensor_transform_arithmetic
 - [gsttensor_transform.h](#), [991](#)
- tensor_transform_clamp
 - [gsttensor_transform.h](#), [992](#)
- tensor_transform_dimchg
 - [gsttensor_transform.h](#), [992](#)
- tensor_transform_mode
 - [gsttensor_transform.h](#), [992](#)
- tensor_transform_operator
 - [gsttensor_transform.h](#), [993](#)
- tensor_transform_operator_s, [289](#)
 - [applying_ch](#), [290](#)
 - [op](#), [290](#)
 - [value](#), [290](#)
- tensor_transform_padding
 - [gsttensor_transform.h](#), [992](#)
- tensor_transform_padding_axis
 - [gsttensor_transform.h](#), [994](#)
- tensor_transform_stand
 - [gsttensor_transform.h](#), [992](#)
- tensor_transform_stand_mode
 - [gsttensor_transform.h](#), [994](#)
- tensor_transform_transpose
 - [gsttensor_transform.h](#), [992](#)
- tensor_transform_typecast
 - [gsttensor_transform.h](#), [993](#)
- tensor_type
 - [tensor_typedef.h](#), [1153](#)
- tensor_typedef.h
 - [_NNS_AUDIO](#), [1154](#)
 - [_NNS_END](#), [1155](#)
 - [_NNS_FLOAT16](#), [1155](#)
 - [_NNS_FLOAT32](#), [1155](#)
 - [_NNS_FLOAT64](#), [1155](#)
 - [_NNS_INT16](#), [1155](#)
 - [_NNS_INT32](#), [1155](#)
 - [_NNS_INT64](#), [1155](#)
 - [_NNS_INT8](#), [1155](#)
 - [_NNS_LAYOUT_ANY](#), [1155](#)
 - [_NNS_LAYOUT_NCHW](#), [1155](#)
 - [_NNS_LAYOUT_NHWC](#), [1155](#)
 - [_NNS_LAYOUT_NONE](#), [1155](#)
 - [_NNS_MEDIA_ANY](#), [1154](#)
 - [_NNS_MEDIA_INVALID](#), [1154](#)
 - [_NNS_OCTET](#), [1154](#)
 - [_NNS_TENSOR](#), [1154](#)
 - [_NNS_TENSOR_FORMAT_END](#), [1156](#)
 - [_NNS_TENSOR_FORMAT_FLEXIBLE](#), [1156](#)
 - [_NNS_TENSOR_FORMAT_SPARSE](#), [1156](#)
 - [_NNS_TENSOR_FORMAT_STATIC](#), [1156](#)
 - [_NNS_TEXT](#), [1154](#)
 - [_NNS_UINT16](#), [1155](#)
 - [_NNS_UINT32](#), [1155](#)
 - [_NNS_UINT64](#), [1155](#)
 - [_NNS_UINT8](#), [1155](#)
 - [_NNS_VIDEO](#), [1154](#)
 - [_nns_media_type](#), [1154](#)
 - [_nns_tensor_layout](#), [1154](#)
 - [_nns_tensor_type](#), [1155](#)
 - [_tensor_format](#), [1155](#)
 - [GST_TENSOR_CAP_DEFAULT](#), [1149](#)
 - [GST_TENSOR_FORMAT_ALL](#), [1149](#)
 - [GST_TENSOR_NUM_TENSORS_RANGE](#), [1149](#)
 - [GST_TENSOR_RATE_RANGE](#), [1149](#)
 - [GST_TENSOR_TYPE_ALL](#), [1150](#)
 - [GST_TENSORS_CAP_DEFAULT](#), [1150](#)
 - [GST_TENSORS_CAP_MAKE](#), [1150](#)
 - [GST_TENSORS_CAP_WITH_NUM](#), [1150](#)
 - [GST_TENSORS_FLEX_CAP_DEFAULT](#), [1151](#)
 - [GST_TENSORS_SPARSE_CAP_DEFAULT](#), [1151](#)
 - [media_type](#), [1153](#)
 - [NNS_MIMETYPE_TENSOR](#), [1151](#)
 - [NNS_MIMETYPE_TENSORS](#), [1151](#)
 - [NNS_TENSOR_MEMORY_MAX](#), [1152](#)
 - [NNS_TENSOR_RANK_LIMIT](#), [1152](#)
 - [NNS_TENSOR_SIZE_EXTRA_LIMIT](#), [1152](#)
 - [NNS_TENSOR_SIZE_LIMIT](#), [1152](#)
 - [NNS_TENSOR_SIZE_LIMIT_STR](#), [1152](#)
 - [tensor_dim](#), [1153](#)
 - [tensor_format](#), [1153](#)
 - [tensor_layout](#), [1153](#)
 - [tensor_type](#), [1153](#)
- TensorDecMaxOpNum
 - [gsttensor_decoder.h](#), [633](#)
- tensorpick
 - [_GstTensorDemux](#), [99](#)
 - [_GstTensorSplit](#), [184](#)
- tensors
 - [_FilterTransformData](#), [31](#)
- tensors_config
 - [_GstTensorConverter](#), [82](#)
 - [_GstTensorDemux](#), [99](#)
 - [_GstTensorMerge](#), [144](#)
 - [_GstTensorMux](#), [148](#)
 - [_GstTensorSrcIIO](#), [191](#)
- tensors_configured
 - [_GstTensorConverter](#), [83](#)
- tensors_info
 - [_GstTensorConverter](#), [83](#)
- tensors_layout
 - [nnstreamer_plugin_api_filter.h](#), [1034](#)
- tensors_offset
 - [_GstDataRepoSrc](#), [44](#)
- tensors_seq

- [_GstDataRepoSrc, 45](#)
- tensors_seq_cnt
 - [_GstDataRepoSrc, 45](#)
- tensors_seq_str
 - [_GstDataRepoSrc, 45](#)
- tensors_size
 - [_GstDataRepoSrc, 45](#)
- tensorseg
 - [_GstTensorSplit, 184](#)
- TEXT_CAPS
 - [gstdatareposink.c, 303](#)
- TEXT_CAPS_STR
 - [gsttensor_converter.c, 521](#)
- TF_MODELNAME
 - [tensor_filter.c, 1404](#)
- then_option
 - [_GstTensorIf, 140](#)
- thread
 - [_NnstWatchdog, 232](#)
- THRESHOLD_CACHE_OLD
 - [tensor_filter.c, 1404](#)
- THRESHOLD_DROP_OLD
 - [tensor_filter.c, 1404](#)
- throttle
 - [_GstTensorRate, 164](#)
- THROTTLE_DELAY_RATIO
 - [gsttensor_rate.c, 734](#)
- throttling_accum
 - [_GstTensorFilter, 101](#)
- throttling_delay
 - [_GstTensorFilter, 102](#)
- throughput
 - [_GstTensorFilterProperties, 133](#)
- throughput_mode
 - [_GstTensorFilterPrivate, 128](#)
- TIFB_END
 - [gsttensor_if.h, 683](#)
- TIFB_FILL_VALUES
 - [gsttensor_if.h, 683](#)
- TIFB_FILL_WITH_FILE
 - [gsttensor_if.h, 683](#)
- TIFB_FILL_WITH_FILE_RPT
 - [gsttensor_if.h, 683](#)
- TIFB_FILL_ZERO
 - [gsttensor_if.h, 683](#)
- TIFB_PASSTHROUGH
 - [gsttensor_if.h, 683](#)
- TIFB_REPEAT_PREVIOUS_FRAME
 - [gsttensor_if.h, 683](#)
- TIFB_SKIP
 - [gsttensor_if.h, 683](#)
- TIFB_TENSORPICK
 - [gsttensor_if.h, 683](#)
- TIFCV_A_VALUE
 - [gsttensor_if.h, 684](#)
- TIFCV_ALL_TENSORS_AVERAGE_VALUE
 - [gsttensor_if.h, 684](#)
- TIFCV_ALL_TENSORS_TOTAL_VALUE
 - [gsttensor_if.h, 684](#)
- TIFCV_CUSTOM
 - [gsttensor_if.h, 684](#)
- TIFCV_END
 - [gsttensor_if.h, 684](#)
- TIFCV_TENSOR_AVERAGE_VALUE
 - [gsttensor_if.h, 684](#)
- TIFCV_TENSOR_TOTAL_VALUE
 - [gsttensor_if.h, 684](#)
- TIFOP_END
 - [gsttensor_if.h, 684](#)
- TIFOP_EQ
 - [gsttensor_if.h, 684](#)
- TIFOP_GE
 - [gsttensor_if.h, 684](#)
- TIFOP_GT
 - [gsttensor_if.h, 684](#)
- TIFOP_LE
 - [gsttensor_if.h, 684](#)
- TIFOP_LT
 - [gsttensor_if.h, 684](#)
- TIFOP_NE
 - [gsttensor_if.c, 678](#)
 - [gsttensor_if.h, 684](#)
- TIFOP_NOT_IN_RANGE_EXCLUSIVE
 - [gsttensor_if.h, 684](#)
- TIFOP_NOT_IN_RANGE_INCLUSIVE
 - [gsttensor_if.h, 684](#)
- TIFOP_RANGE_EXCLUSIVE
 - [gsttensor_if.h, 684](#)
- TIFOP_RANGE_INCLUSIVE
 - [gsttensor_if.h, 684](#)
- TIFSP_ELSE_PAD
 - [gsttensor_if.h, 685](#)
- TIFSP_THEN_PAD
 - [gsttensor_if.h, 685](#)
- timeout
 - [_GstTensorQueryClient, 153](#)
 - [_GstTensorQueryServerSink, 155](#)
 - [_GstTensorQueryServerSrc, 159](#)
- TIMESTAMP
 - [gsttensor_srcio.c, 885](#)
- to
 - [_tensor_transform_dimchg, 242](#)
 - [_tensor_transform_typecast, 245](#)
- to_rate_denominator
 - [_GstTensorRate, 164](#)
- to_rate_numerator
 - [_GstTensorRate, 165](#)
- topic
 - [_GstEdgeSink, 49](#)
 - [_GstEdgeSrc, 52](#)
 - [_GstTensorQueryClient, 153](#)
 - [_GstTensorQueryServerSrc, 159](#)
 - [GstTensorQueryEdgeInfo, 267](#)
- total_invoke_latency
 - [_GstTensorFilterFrameworkStatistics, 123](#)
 - [_GstTensorFilterStatistics, 135](#)

- total_invoke_num
 - _GstTensorFilterFrameworkStatistics, 123
 - _GstTensorFilterStatistics, 135
- total_overhead_latency
 - _GstTensorFilterFrameworkStatistics, 123
- total_samples
 - _GstDataRepoSink, 34
 - _GstDataRepoSrc, 45
- track_latency
 - tensor_filter.c, 1430
- TRAINER_EVENT_EPOCH_COMPLETION
 - nnstreamer_plugin_api_trainer.h, 1046
- TRAINER_EVENT_TRAINING_COMPLETION
 - nnstreamer_plugin_api_trainer.h, 1046
- TRAINER_EVENT_UNKNOWN
 - nnstreamer_plugin_api_trainer.h, 1046
- TRAINING_ACCURACY
 - gsttensor_trainer.c, 925
- training_accuracy
 - _GstTensorTrainerProperties, 213
- training_completion_cond
 - _GstTensorTrainer, 202
- training_completion_lock
 - _GstTensorTrainer, 202
- TRAINING_LOSS
 - gsttensor_trainer.c, 925
- training_loss
 - _GstTensorTrainerProperties, 213
- trans_order
 - _tensor_transform_transpose, 245
- transposeloop
 - gsttensor_transform.c, 961
- TRIGGER
 - gsttensor_srcii.c, 885
- trigger
 - _GstTensorSrcIIO, 191
- TRIGGER_PREFIX
 - gsttensor_srcii.c, 886
- TRUE
 - gsttensor_if.c, 678
- type
 - gsttensor_if.c, 678
 - GstTensorInfo, 261
 - GstTensorMetaInfo, 264
 - tensor_data_s, 284
 - tensor_if_sv_s, 287
- TYPE_SUFFIX
 - gsttensor_srcii.c, 886
- unregister_subplugin
 - nnstreamer_subplugin.c, 1333
 - nnstreamer_subplugin.h, 1343
- unregisterExternalConverter
 - gsttensor_converter.c, 552
 - nnstreamer_plugin_api_converter.h, 1020
- UNUSED
 - mqttcommon.h, 406
 - nnstreamer_util.h, 1121
- URI_KEYWORD_MODEL
 - ml_agent.c, 1158
- URI_SCHEME
 - ml_agent.c, 1158
- used_bits
 - _GstTensorSrcIIOChannelProperties, 195
- VALIDATION_ACCURACY
 - gsttensor_trainer.c, 925
- validation_accuracy
 - _GstTensorTrainerProperties, 213
- VALIDATION_LOSS
 - gsttensor_trainer.c, 925
- validation_loss
 - _GstTensorTrainerProperties, 213
- value
 - tensor_transform_operator_s, 290
- value_float
 - gsttensor_srcii.c, 915
- value_unsigned
 - gsttensor_srcii.c, 915
- verify_model_path
 - _GstTensorFilterFramework, 115
 - _GstTensorFilterFrameworkInfo, 122
 - tensor_filter_common.c, 1494
- version
 - _GstTensorFilterFramework, 116
 - _GstTensorTrainerEventNotifier, 204
 - _GstTensorTrainerFramework, 209
 - GstTensorExtraInfo, 258
 - GstTensorMetaInfo, 264
- VIDEO_CAPS
 - gstdataareposink.c, 303
- VIDEO_CAPS_STR
 - gsttensor_converter_media_info_video.h, 561
- vstr
 - vstr_helper, 291
- vstr_helper, 290
 - cursor, 291
 - size, 291
 - vstr, 291
- w
 - tensor_region_s, 288
- wait_connection
 - _GstEdgeSink, 49
- watchdog_h
 - _GstTensorFilterPrivate, 128
- while
 - gsttensor_transform.c, 985
 - nnstreamer.c, 1356
 - tensor_data.c, 1390
- x
 - tensor_region_s, 288
- xmit_ts
 - _ntp_packet_t, 236
- y
 - tensor_region_s, 289