

vxgcloudagent

1.2.34

Generated by Doxygen 1.8.17



<b>1 VXG Cloud Agent Library</b>	<b>1</b>
<b>2 Build System</b>	<b>3</b>
2.0.1 Overview	3
2.0.2 C++ Toolchain Requirements	3
2.0.3 Build system installation	3
<b>3 Application Development</b>	<b>5</b>
3.1 Overview	5
3.2 Examples	5
3.2.1 Minimal application example	5
3.2.2 Complete application example	7
3.2.3 Linking application against the VXG Agent Cloud Library	10
<b>4 Library Compilation Guide</b>	<b>13</b>
4.0.1 Library build process	13
4.0.2 Cross-compilation	13
<b>5 Deprecated List</b>	<b>15</b>
<b>6 Hierarchical Index</b>	<b>17</b>
6.1 Class Hierarchy	17
<b>7 Data Structure Index</b>	<b>19</b>
7.1 Data Structures	19
<b>8 File Index</b>	<b>23</b>
8.1 File List	23
<b>9 Namespace Documentation</b>	<b>25</b>
9.1 nlohmann Namespace Reference	25
9.2 std Namespace Reference	25
9.2.1 Function Documentation	43
9.2.1.1 make_unique()	43
9.3 vxg Namespace Reference	43
9.4 vxg::cloud Namespace Reference	44
9.4.1 Typedef Documentation	44
9.4.1.1 duration	44
9.4.1.2 time	44
9.5 vxg::cloud::agent Namespace Reference	44
9.5.1 Detailed Description	45
9.5.2 Function Documentation	45
9.5.2.1 version()	46
9.6 vxg::cloud::agent::media Namespace Reference	46
9.7 vxg::cloud::agent::proto Namespace Reference	46

9.7.1 Typedef Documentation	48
9.7.1.1 wifi_list	48
9.7.2 Enumeration Type Documentation	48
9.7.2.1 audio_file_format	48
9.7.2.2 audio_format	48
9.7.2.3 event_status	49
9.7.2.4 event_type	49
9.7.2.5 memorycard_status	50
9.7.2.6 mode	50
9.7.2.7 motion_region_shape	50
9.7.2.8 motion_sensitivity	51
9.7.2.9 ptz_action	51
9.7.2.10 ptz_preset_action	51
9.7.2.11 time_format_n	52
9.7.2.12 video_format	52
9.7.2.13 wifi_encryption	52
9.7.2.14 wifi_network_state	53
9.7.3 Function Documentation	53
9.7.3.1 name()	53
9.8 vxg::cloud::time_spec Namespace Reference	53
9.8.1 Detailed Description	54
9.8.2 Typedef Documentation	54
9.8.2.1 duration	54
9.8.2.2 precision	54
9.9 vxg::cloud::utils Namespace Reference	54
9.9.1 Function Documentation	55
9.9.1.1 dirname()	55
9.9.1.2 set_thread_name()	55
9.9.1.3 string_contains()	55
9.9.1.4 string_endswith()	56
9.9.1.5 string_format()	56
9.9.1.6 string_replace()	56
9.9.1.7 string_split()	56
9.9.1.8 string_startswith()	56
9.9.1.9 string_tolower()	56
9.9.1.10 string_toupper()	57
9.9.1.11 string_trim() [1/2]	57
9.9.1.12 string_trim() [2/2]	57
9.9.1.13 string_urldecode()	57
9.9.1.14 string_urlencode()	57
9.10 vxg::cloud::utils::gcc_abi Namespace Reference	57
9.10.1 Function Documentation	57

9.10.1.1 demangle()	58
9.11 vxg::cloud::utils::motion Namespace Reference	58
9.12 vxg::cloud::utils::time Namespace Reference	58
9.12.1 Function Documentation	58
9.12.1.1 from_double()	58
9.12.1.2 from_iso()	59
9.12.1.3 from_iso2()	59
9.12.1.4 from_iso_packed()	59
9.12.1.5 is_iso()	59
9.12.1.6 is_iso_packed()	59
9.12.1.7 ISO8601_to_time()	59
9.12.1.8 iso_time_valid()	59
9.12.1.9 max()	60
9.12.1.10 now()	60
9.12.1.11 now_ISO8601.UTC()	60
9.12.1.12 now_ISO8601.UTC_packed()	60
9.12.1.13 now_time.UTC()	60
9.12.1.14 null()	60
9.12.1.15 time_to_ISO8601()	60
9.12.1.16 time_to_ISO8601_packed()	61
9.12.1.17 to_double()	61
9.12.1.18 to_iso()	61
9.12.1.19 to_iso2()	61
9.12.1.20 to_iso_8601()	61
9.12.1.21 to_iso_local()	61
9.12.1.22 to_iso_packed()	61
9.13 vxg::media Namespace Reference	62
9.14 vxg::media::ffmpeg Namespace Reference	62
9.15 vxg::media::Streamer Namespace Reference	62
9.15.1 Enumeration Type Documentation	63
9.15.1.1 DropDirection	63
9.15.1.2 MediaType	63
9.15.1.3 StreamError	64
9.15.2 Variable Documentation	64
9.15.2.1 SINK_THREAD_PRIO	64
9.15.2.2 SRC_THREAD_PRIO	64
<b>10 Data Structure Documentation</b>	<b>65</b>
10.1 vxg::cloud::agent::access_token Struct Reference	65
10.1.1 Detailed Description	65
10.1.2 Member Typedef Documentation	66
10.1.2.1 ptr	66

10.1.3 Member Function Documentation	66
10.1.3.1 api_uri()	66
10.1.3.2 pack()	66
10.1.3.3 parse()	66
10.2 alter_bool Struct Reference	66
10.2.1 Detailed Description	67
10.2.2 Member Enumeration Documentation	67
10.2.2.1 n_alter_bool	67
10.2.3 Constructor & Destructor Documentation	68
10.2.3.1 alter_bool() [1/2]	68
10.2.3.2 alter_bool() [2/2]	68
10.2.4 Member Function Documentation	68
10.2.4.1 operator bool()	68
10.2.4.2 operator=()	68
10.2.5 Friends And Related Function Documentation	68
10.2.5.1 from_json	69
10.2.5.2 to_json	69
10.2.6 Field Documentation	69
10.2.6.1 val	69
10.3 vxg::cloud::agent::proto::audio_caps Struct Reference	69
10.3.1 Detailed Description	70
10.3.2 Field Documentation	70
10.3.2.1 audio_file_formats	70
10.3.2.2 backward	70
10.3.2.3 backward_formats	71
10.3.2.4 echo_cancel	71
10.3.2.5 mic	71
10.3.2.6 spkr	71
10.4 vxg::cloud::agent::audio_config Struct Reference	72
10.4.1 Detailed Description	72
10.4.2 Field Documentation	72
10.4.2.1 caps	73
10.4.2.2 echo_cancel	73
10.4.2.3 mic_gain	73
10.4.2.4 mic_mute	73
10.4.2.5 spkr_mute	73
10.4.2.6 spkr_vol	74
10.5 vxg::cloud::agent::proto::audio_stream_config Struct Reference	74
10.5.1 Detailed Description	75
10.5.2 Field Documentation	75
10.5.2.1 brt	75
10.5.2.2 format	75

10.5.2.3 srt	75
10.5.2.4 stream	75
10.6 vxg::media::Streamer::StreamInfo::AudioInfo Struct Reference	76
10.6.1 Detailed Description	76
10.6.2 Field Documentation	76
10.6.2.1 bitrate	77
10.6.2.2 channels	77
10.6.2.3 codec	77
10.6.2.4 extradata	77
10.6.2.5 samplerate	77
10.6.2.6 timebase	78
10.7 vxg::cloud::agent::callback Class Reference	78
10.7.1 Detailed Description	79
10.7.2 Member Typedef Documentation	79
10.7.2.1 ptr	79
10.7.3 Member Function Documentation	80
10.7.3.1 on_audio_file_play()	80
10.7.3.2 on_bye()	80
10.7.3.3 on_cam_ptz()	80
10.7.3.4 on_cam_ptz_preset()	81
10.7.3.5 on_cam_upgrade_firmware()	81
10.7.3.6 on_get_cam_audio_config()	82
10.7.3.7 on_get_cam_events_config()	82
10.7.3.8 on_get_cam_video_config()	82
10.7.3.9 on_get_log()	83
10.7.3.10 on_get_memorycard_info()	83
10.7.3.11 on_get_motion_detection_config()	84
10.7.3.12 on_get_osd_config()	84
10.7.3.13 on_get_ptz_config()	85
10.7.3.14 on_get_timezone()	85
10.7.3.15 on_get_wifi_config()	85
10.7.3.16 on_raw_msg()	86
10.7.3.17 on_registered()	86
10.7.3.18 on_set_cam_audio_config()	87
10.7.3.19 on_set_cam_events_config()	87
10.7.3.20 on_set_cam_video_config()	87
10.7.3.21 on_set_motion_detection_config()	88
10.7.3.22 on_set_osd_config()	88
10.7.3.23 on_set_timezone()	89
10.7.3.24 on_set_wifi_config()	89
10.7.3.25 on_start_backward_audio()	89
10.7.3.26 on_stop_backward_audio()	90

10.7.3.27 on_trigger_event()	90
10.8 vxg::cloud::agent::proto::stream_caps::caps_audio_object Struct Reference	91
10.8.1 Detailed Description	91
10.8.2 Field Documentation	91
10.8.2.1 brt	92
10.8.2.2 formats	92
10.8.2.3 srt	92
10.8.2.4 streams	92
10.9 vxg::cloud::agent::proto::stream_caps::caps_video_object Struct Reference	93
10.9.1 Detailed Description	94
10.9.2 Field Documentation	94
10.9.2.1 brt	94
10.9.2.2 formats	94
10.9.2.3 fps	94
10.9.2.4 gop	95
10.9.2.5 profiles	95
10.9.2.6 quality	95
10.9.2.7 resolutions	95
10.9.2.8 smoothing	96
10.9.2.9 streams	96
10.9.2.10 vbr	96
10.9.2.11 vbr_brt	96
10.10 vxg::cloud::agent::proto::event_caps Struct Reference	96
10.10.1 Detailed Description	97
10.10.2 Field Documentation	97
10.10.2.1 periodic	97
10.10.2.2 snapshot	97
10.10.2.3 statefull	97
10.10.2.4 stream	98
10.10.2.5 trigger	98
10.11 vxg::cloud::agent::event_config Struct Reference	98
10.11.1 Detailed Description	99
10.11.2 Member Function Documentation	99
10.11.2.1 caps_eq()	99
10.11.2.2 name()	100
10.11.2.3 name_eq()	100
10.11.3 Field Documentation	100
10.11.3.1 active	100
10.11.3.2 caps	100
10.11.3.3 custom_event_name	101
10.11.3.4 event	101
10.11.3.5 period	101



10.11.3.6 snapshot	101
10.11.3.7 stream	101
10.12 vxg::cloud::agent::manager::event_state::event_state_caps Struct Reference	102
10.12.1 Detailed Description	102
10.12.2 Field Documentation	102
10.12.2.1 need_clip	102
10.12.2.2 need_snapshot	102
10.12.2.3 stateful	102
10.13 vxg::cloud::agent::event_stream Class Reference	103
10.13.1 Detailed Description	103
10.13.2 Member Typedef Documentation	103
10.13.2.1 ptr	104
10.13.3 Constructor & Destructor Documentation	104
10.13.3.1 event_stream()	104
10.13.3.2 ~event_stream()	104
10.13.4 Member Function Documentation	104
10.13.4.1 finit()	104
10.13.4.2 get_events()	104
10.13.4.3 init()	105
10.13.4.4 notify()	105
10.13.4.5 set_events()	105
10.13.4.6 set_trigger_recording()	106
10.13.4.7 start()	106
10.13.4.8 stop()	107
10.13.4.9 trigger_event()	107
10.14 vxg::cloud::agent::events_config Struct Reference	107
10.14.1 Detailed Description	108
10.14.2 Member Function Documentation	108
10.14.2.1 get_event_config()	108
10.14.3 Field Documentation	109
10.14.3.1 enabled	109
10.14.3.2 events	109
10.15 vxg::media::Streamer::ISink Class Reference	110
10.15.1 Detailed Description	111
10.15.2 Member Typedef Documentation	111
10.15.2.1 ptr	111
10.15.2.2 PtrU	111
10.15.3 Constructor & Destructor Documentation	111
10.15.3.1 ISink()	111
10.15.3.2 ~ISink()	112
10.15.4 Member Function Documentation	112
10.15.4.1 droppable()	112

10.15.4.2 duration()	112
10.15.4.3 error()	112
10.15.4.4 finit()	113
10.15.4.5 init()	113
10.15.4.6 name()	113
10.15.4.7 negotiate()	114
10.15.4.8 process()	114
10.15.4.9 set_eos()	115
10.15.4.10 set_eos_cb()	115
10.16 vxg::media::Streamer::ISource Class Reference	115
10.16.1 Detailed Description	116
10.16.2 Member Typedef Documentation	116
10.16.2.1 ptr	116
10.16.3 Member Enumeration Documentation	116
10.16.3.1 Mode	116
10.16.4 Constructor & Destructor Documentation	117
10.16.4.1 ISource()	117
10.16.5 Member Function Documentation	117
10.16.5.1 error()	117
10.16.5.2 finit()	118
10.16.5.3 init()	118
10.16.5.4 name()	118
10.16.5.5 negotiate()	119
10.16.5.6 pullFrame()	119
10.16.5.7 pushFrame()	119
10.16.6 Field Documentation	120
10.16.6.1 mode_	120
10.17 vxg::logger Class Reference	120
10.17.1 Detailed Description	121
10.17.2 Member Typedef Documentation	121
10.17.2.1 logger_ptr	121
10.17.3 Member Enumeration Documentation	121
10.17.3.1 loglevel	121
10.17.4 Member Function Documentation	122
10.17.4.1 debug() [1/2]	122
10.17.4.2 debug() [2/2]	122
10.17.4.3 error() [1/2]	122
10.17.4.4 error() [2/2]	122
10.17.4.5 info() [1/2]	122
10.17.4.6 info() [2/2]	123
10.17.4.7 instance()	123
10.17.4.8 reset() [1/2]	124

10.17.4.9 reset() [2/2]	124
10.17.4.10 set_level()	124
10.17.4.11 trace() [1/2]	125
10.17.4.12 trace() [2/2]	125
10.17.4.13 warn() [1/2]	125
10.17.4.14 warn() [2/2]	125
10.18 vxg::cloud::agent::manager Class Reference	126
10.18.1 Detailed Description	128
10.18.2 Member Typedef Documentation	128
10.18.2.1 ptr	129
10.18.3 Member Function Documentation	129
10.18.3.1 __notify_record_event()	129
10.18.3.2 __trigger_periodic_event()	129
10.18.3.3 _append_internal_custom_events()	129
10.18.3.4 _cancel_direct_uploads_by_ticket()	129
10.18.3.5 _cancel_periodic_event()	129
10.18.3.6 _cancel_periodic_events()	130
10.18.3.7 _current_delivery_mode()	130
10.18.3.8 _handle_stream_stateful_event()	130
10.18.3.9 _handle_stream_stateless_event()	130
10.18.3.10 _init_events_states()	130
10.18.3.11 _load_events_configs()	130
10.18.3.12 _lookup_event_stream()	130
10.18.3.13 _lookup_event_stream_by_event()	131
10.18.3.14 _request_direct_upload_snapshot()	131
10.18.3.15 _request_direct_upload_video()	131
10.18.3.16 _schedule_direct_upload()	131
10.18.3.17 _schedule_periodic_event()	131
10.18.3.18 _schedule_periodic_events()	131
10.18.3.19 _stop_all_event_streams()	131
10.18.3.20 _stop_all_streams()	132
10.18.3.21 _stop_stream()	132
10.18.3.22 _update_direct_upload_queue_latency()	132
10.18.3.23 _update_event_stream_configs()	132
10.18.3.24 _update_events_configs()	132
10.18.3.25 _update_storage_status()	132
10.18.3.26 create()	132
10.18.3.27 direct_upload_sync_cb()	133
10.18.3.28 handle_event_meta_file()	133
10.18.3.29 handle_event_snapshot()	133
10.18.3.30 handle_stream_event()	133
10.18.3.31 lookup_stream()	134

10.18.3.32 notify_event()	134
10.18.3.33 on_audio_file_play()	134
10.18.3.34 on_cam_memorycard_recording()	134
10.18.3.35 on_cam_memorycard_synchronize()	134
10.18.3.36 on_cam_memorycard_synchronize_cancel()	134
10.18.3.37 on_cam_ptz()	135
10.18.3.38 on_cam_ptz_preset()	135
10.18.3.39 on_cam_upgrade_firmware()	135
10.18.3.40 on_closed()	135
10.18.3.41 on_direct_upload_url()	135
10.18.3.42 on_get_cam_audio_config()	135
10.18.3.43 on_get_cam_events_config()	136
10.18.3.44 on_get_cam_memorycard_timeline()	136
10.18.3.45 on_get_cam_video_config()	136
10.18.3.46 on_get_log()	136
10.18.3.47 on_get_motion_detection_config()	136
10.18.3.48 on_get_osd_config()	136
10.18.3.49 on_get_ptz_config()	136
10.18.3.50 on_get_stream_by_event()	137
10.18.3.51 on_get_stream_caps()	137
10.18.3.52 on_get_stream_config()	137
10.18.3.53 on_get_supported_streams()	137
10.18.3.54 on_get_timezone()	137
10.18.3.55 on_get_wifi_config()	137
10.18.3.56 on_prepared()	137
10.18.3.57 on_raw_message()	138
10.18.3.58 on_registered()	138
10.18.3.59 on_set_activity()	138
10.18.3.60 on_set_cam_audio_config()	138
10.18.3.61 on_set_cam_events_config()	138
10.18.3.62 on_set_cam_video_config()	138
10.18.3.63 on_set_log_enable()	138
10.18.3.64 on_set_motion_detection_config()	139
10.18.3.65 on_set_osd_config()	139
10.18.3.66 on_set_periodic_events()	139
10.18.3.67 on_set_stream_by_event()	139
10.18.3.68 on_set_stream_config()	139
10.18.3.69 on_set_timezone()	139
10.18.3.70 on_set_wifi_config()	139
10.18.3.71 on_start_backward()	140
10.18.3.72 on_stop_backward()	140
10.18.3.73 on_stream_start()	140

10.18.3.74 on_stream_stop()	140
10.18.3.75 on_trigger_event()	140
10.18.3.76 on_update_preview()	140
10.18.3.77 start()	141
10.18.3.78 stop()	141
10.19 vxg::cloud::utils::motion::map Struct Reference	141
10.19.1 Detailed Description	142
10.19.2 Constructor & Destructor Documentation	142
10.19.2.1 map() [1/2]	142
10.19.2.2 map() [2/2]	142
10.19.3 Member Function Documentation	143
10.19.3.1 operator=()	143
10.19.3.2 pack()	143
10.19.3.3 unpack()	143
10.20 vxg::media::Streamer::MediaFrame Struct Reference	143
10.20.1 Detailed Description	144
10.20.2 Member Function Documentation	144
10.20.2.1 operator<()	144
10.20.3 Field Documentation	145
10.20.3.1 data	145
10.20.3.2 dts	145
10.20.3.3 duration	145
10.20.3.4 is_key	146
10.20.3.5 len	146
10.20.3.6 NO_PTS	146
10.20.3.7 pts	146
10.20.3.8 time_realtime	146
10.20.3.9 timescale	147
10.20.3.10 type	147
10.21 vxg::cloud::agent::proto::motion_detection_caps Struct Reference	147
10.21.1 Detailed Description	147
10.21.2 Field Documentation	147
10.21.2.1 max_regions	148
10.21.2.2 region_shape	148
10.21.2.3 sensitivity	148
10.22 vxg::cloud::agent::proto::motion_detection_config Struct Reference	148
10.22.1 Detailed Description	149
10.22.2 Field Documentation	149
10.22.2.1 caps	149
10.22.2.2 columns	149
10.22.2.3 regions	149
10.22.2.4 rows	150

10.23 vxg::cloud::agent::proto::motion_region Struct Reference	150
10.23.1 Detailed Description	151
10.23.2 Field Documentation	151
10.23.2.1 enabled	151
10.23.2.2 map	151
10.23.2.3 region	151
10.23.2.4 sensitivity	152
10.24 vxg::logger::options Struct Reference	152
10.24.1 Detailed Description	153
10.24.2 Field Documentation	153
10.24.2.1 crash_logfile_path	153
10.24.2.2 default_loglevel	153
10.24.2.3 log_pattern	153
10.24.2.4 logfile_max_files	153
10.24.2.5 logfile_max_size	153
10.24.2.6 logfile_path	154
10.24.2.7 syslog_ident	154
10.24.2.8 tcp_logsink_enabled	154
10.24.2.9 tcp_logsink_host	154
10.24.2.10 tcp_logsink_port	154
10.25 vxg::cloud::agent::proto::osd_caps Struct Reference	155
10.25.1 Detailed Description	155
10.25.2 Field Documentation	156
10.25.2.1 alignment	156
10.25.2.2 bkg_color	156
10.25.2.3 bkg_transp	156
10.25.2.4 date	156
10.25.2.5 date_format	157
10.25.2.6 font_color	157
10.25.2.7 font_size	157
10.25.2.8 system_id	157
10.25.2.9 system_id_text	158
10.25.2.10 time	158
10.25.2.11 time_format	158
10.26 vxg::cloud::agent::osd_config Struct Reference	158
10.26.1 Detailed Description	159
10.26.2 Field Documentation	159
10.26.2.1 alignment	159
10.26.2.2 bkg_color	160
10.26.2.3 bkg_transp	160
10.26.2.4 caps	160
10.26.2.5 date	160

10.26.2.6 date_format . . . . .	160
10.26.2.7 font_color . . . . .	161
10.26.2.8 font_size . . . . .	161
10.26.2.9 system_id . . . . .	161
10.26.2.10 system_id_text . . . . .	161
10.26.2.11 time . . . . .	161
10.26.2.12 time_format . . . . .	162
10.27 vxg::cloud::agent::access_token::proxy_config Struct Reference . . . . .	162
10.27.1 Detailed Description . . . . .	162
10.27.2 Field Documentation . . . . .	163
10.27.2.1 socks4 . . . . .	163
10.27.2.2 socks5 . . . . .	163
10.28 vxg::cloud::agent::ptz_command Struct Reference . . . . .	163
10.28.1 Detailed Description . . . . .	163
10.28.2 Field Documentation . . . . .	164
10.28.2.1 action . . . . .	164
10.28.2.2 tm . . . . .	164
10.29 vxg::cloud::agent::ptz_config Struct Reference . . . . .	164
10.29.1 Detailed Description . . . . .	165
10.29.2 Field Documentation . . . . .	165
10.29.2.1 actions . . . . .	165
10.29.2.2 maximum_number_of_presets . . . . .	165
10.29.2.3 presets . . . . .	166
10.30 vxg::cloud::agent::ptz_preset Struct Reference . . . . .	166
10.30.1 Detailed Description . . . . .	166
10.30.2 Field Documentation . . . . .	167
10.30.2.1 action . . . . .	167
10.30.2.2 name . . . . .	167
10.30.2.3 token . . . . .	167
10.31 vxg::media::rtmp_sink Class Reference . . . . .	168
10.31.1 Detailed Description . . . . .	169
10.31.2 Constructor & Destructor Documentation . . . . .	169
10.31.2.1 rtmp_sink() . . . . .	169
10.31.3 Member Function Documentation . . . . .	169
10.31.3.1 droppable() . . . . .	170
10.31.3.2 error() . . . . .	170
10.31.3.3 init() . . . . .	170
10.31.3.4 name() . . . . .	171
10.31.3.5 negotiate() . . . . .	171
10.32 vxg::media::rtmp_source Class Reference . . . . .	172
10.32.1 Detailed Description . . . . .	173
10.32.2 Member Function Documentation . . . . .	173

10.32.2.1 init()	173
10.33 vxg::media::rtsp_source Class Reference	173
10.33.1 Detailed Description	175
10.33.2 Constructor & Destructor Documentation	175
10.33.2.1 rtsp_source() [1/2]	175
10.33.2.2 rtsp_source() [2/2]	175
10.33.3 Member Function Documentation	177
10.33.3.1 init()	177
10.33.3.2 name()	177
10.33.4 Field Documentation	178
10.33.4.1 ffmpeg_opts_	178
10.34 vxg::cloud::agent::media::rtsp_stream Class Reference	178
10.34.1 Detailed Description	180
10.34.2 Member Typedef Documentation	180
10.34.2.1 ptr	180
10.34.3 Constructor & Destructor Documentation	180
10.34.3.1 rtsp_stream()	180
10.34.3.2 ~rtsp_stream()	180
10.34.4 Member Function Documentation	181
10.34.4.1 get_snapshot()	181
10.34.4.2 get_stream_caps()	181
10.34.4.3 get_stream_config()	181
10.34.4.4 get_supported_stream()	182
10.34.4.5 record_export()	182
10.34.4.6 record_get_list()	182
10.34.4.7 set_stream_config()	183
10.34.4.8 start()	183
10.34.4.9 start_record()	184
10.34.4.10 stop_record()	184
10.35 vxg::media::ffmpeg::Sink Class Reference	185
10.35.1 Detailed Description	186
10.35.2 Constructor & Destructor Documentation	186
10.35.2.1 Sink()	186
10.35.2.2 ~Sink()	186
10.35.3 Member Function Documentation	186
10.35.3.1 droppable()	187
10.35.3.2 duration()	187
10.35.3.3 error()	187
10.35.3.4 finit()	188
10.35.3.5 init() [1/2]	188
10.35.3.6 init() [2/2]	188
10.35.3.7 name()	189



10.35.3.8 negotiate()	189
10.35.3.9 stop()	190
10.36 vxg::media::ffmpeg::Source Class Reference	190
10.36.1 Detailed Description	191
10.36.2 Constructor & Destructor Documentation	191
10.36.2.1 Source()	192
10.36.2.2 ~Source()	192
10.36.3 Member Function Documentation	192
10.36.3.1 finit()	192
10.36.3.2 init() [1/3]	192
10.36.3.3 init() [2/3]	193
10.36.3.4 init() [3/3]	193
10.36.3.5 name()	194
10.36.3.6 negotiate()	194
10.36.3.7 pullFrame()	194
10.36.3.8 stop()	195
10.37 vxg::cloud::agent::media::stream Class Reference	195
10.37.1 Detailed Description	197
10.37.2 Member Typedef Documentation	197
10.37.2.1 ptr	197
10.37.3 Constructor & Destructor Documentation	197
10.37.3.1 stream()	197
10.37.3.2 ~stream()	198
10.37.4 Member Function Documentation	198
10.37.4.1 get_snapshot()	198
10.37.4.2 get_stream_caps()	198
10.37.4.3 get_stream_config()	199
10.37.4.4 get_supported_stream()	199
10.37.4.5 record_export()	199
10.37.4.6 record_get_list()	200
10.37.4.7 record_needs_source()	200
10.37.4.8 set_stream_config()	201
10.37.4.9 start_record()	201
10.37.4.10 stop_record()	201
10.38 vxg::media::stream Class Reference	202
10.38.1 Detailed Description	203
10.38.2 Member Typedef Documentation	203
10.38.2.1 ptr	203
10.38.3 Constructor & Destructor Documentation	203
10.38.3.1 stream()	203
10.38.3.2 ~stream()	204
10.38.4 Member Function Documentation	204

10.38.4.1	<a href="#">finit_sink()</a>	204
10.38.4.2	<a href="#">finit_source()</a>	204
10.38.4.3	<a href="#">init_sink()</a>	204
10.38.4.4	<a href="#">init_source()</a>	205
10.38.5	Field Documentation	205
10.38.5.1	<a href="#">sink_</a>	205
10.38.5.2	<a href="#">source_</a>	206
10.39	<a href="#">vxg::cloud::agent::proto::stream_caps Struct Reference</a>	206
10.39.1	Detailed Description	207
10.39.2	Field Documentation	207
10.39.2.1	<a href="#">caps_audio</a>	207
10.39.2.2	<a href="#">caps_video</a>	207
10.40	<a href="#">vxg::cloud::agent::proto::stream_config Struct Reference</a>	207
10.40.1	Detailed Description	208
10.40.2	Field Documentation	208
10.40.2.1	<a href="#">audio</a>	208
10.40.2.2	<a href="#">video</a>	208
10.41	<a href="#">vxg::media::Streamer::StreamInfo Struct Reference</a>	209
10.41.1	Detailed Description	210
10.41.2	Member Enumeration Documentation	210
10.41.2.1	<a href="#">AudioCodec</a>	210
10.41.2.2	<a href="#">DataCodec</a>	210
10.41.2.3	<a href="#">StreamType</a>	210
10.41.2.4	<a href="#">VideoCodec</a>	211
10.41.3	Field Documentation	211
10.41.3.1	<a href="#">audio</a>	211
10.41.3.2	<a href="#">type</a>	211
10.41.3.3	<a href="#">video</a>	212
10.42	<a href="#">vxg::cloud::agent::supported_stream_config Struct Reference</a>	212
10.42.1	Detailed Description	212
10.42.2	Field Documentation	213
10.42.2.1	<a href="#">audio</a>	213
10.42.2.2	<a href="#">id</a>	213
10.42.2.3	<a href="#">video</a>	213
10.43	<a href="#">vxg::cloud::agent::supported_streams_config Struct Reference</a>	213
10.43.1	Detailed Description	214
10.43.2	Field Documentation	214
10.43.2.1	<a href="#">audio_es</a>	214
10.43.2.2	<a href="#">streams</a>	214
10.43.2.3	<a href="#">video_es</a>	214
10.44	<a href="#">vxg::cloud::utils::uri Struct Reference</a>	215
10.44.1	Detailed Description	215

10.44.2 Member Function Documentation	215
10.44.2.1 parse()	216
10.44.3 Field Documentation	216
10.44.3.1 fragment	216
10.44.3.2 host	216
10.44.3.3 password	216
10.44.3.4 path	216
10.44.3.5 port	217
10.44.3.6 query	217
10.44.3.7 scheme	217
10.44.3.8 user	217
10.45 vxg::cloud::agent::proto::video_caps Struct Reference	217
10.45.1 Detailed Description	218
10.45.2 Field Documentation	218
10.45.2.1 brightness	218
10.45.2.2 contrast	219
10.45.2.3 horz_flip	219
10.45.2.4 ir_light	219
10.45.2.5 nr_level	219
10.45.2.6 nr_type	219
10.45.2.7 pwr_frequency	220
10.45.2.8 saturation	220
10.45.2.9 sharpness	220
10.45.2.10 tdn	220
10.45.2.11 vert_flip	220
10.45.2.12 wb_type	221
10.46 vxg::cloud::agent::proto::video_clip_info Struct Reference	221
10.46.1 Detailed Description	222
10.46.2 Field Documentation	222
10.46.2.1 data	222
10.46.2.2 local_start	222
10.46.2.3 local_stop	222
10.46.2.4 tp_start	222
10.46.2.5 tp_stop	223
10.46.2.6 video_height	223
10.46.2.7 video_width	223
10.47 vxg::cloud::agent::proto::video_config Struct Reference	223
10.47.1 Detailed Description	225
10.47.2 Field Documentation	225
10.47.2.1 brightness	225
10.47.2.2 caps	225
10.47.2.3 contrast	225

10.47.2.4 horz_flip . . . . .	226
10.47.2.5 ir_light . . . . .	226
10.47.2.6 nr_level . . . . .	226
10.47.2.7 nr_type . . . . .	226
10.47.2.8 pwr_frequency . . . . .	226
10.47.2.9 saturation . . . . .	227
10.47.2.10 sharpness . . . . .	227
10.47.2.11 tdn . . . . .	227
10.47.2.12 vert_flip . . . . .	227
10.47.2.13 wb_type . . . . .	227
10.48 vxg::cloud::agent::proto::video_stream_config Struct Reference . . . . .	228
10.48.1 Detailed Description . . . . .	229
10.48.2 Field Documentation . . . . .	229
10.48.2.1 brt . . . . .	229
10.48.2.2 format . . . . .	229
10.48.2.3 fps . . . . .	229
10.48.2.4 gop . . . . .	229
10.48.2.5 horz . . . . .	230
10.48.2.6 profile . . . . .	230
10.48.2.7 quality . . . . .	230
10.48.2.8 smoothing . . . . .	230
10.48.2.9 stream . . . . .	230
10.48.2.10 vbr . . . . .	231
10.48.2.11 vbr_brt . . . . .	231
10.48.2.12 vert . . . . .	231
10.49 vxg::media::Streamer::StreamInfo::VideoInfo Struct Reference . . . . .	231
10.49.1 Detailed Description . . . . .	232
10.49.2 Field Documentation . . . . .	232
10.49.2.1 bitrate . . . . .	232
10.49.2.2 codec . . . . .	232
10.49.2.3 extradata . . . . .	233
10.49.2.4 framerate . . . . .	233
10.49.2.5 height . . . . .	233
10.49.2.6 timebase . . . . .	233
10.49.2.7 width . . . . .	233
10.50 vxg::cloud::agent::proto::wifi_config Struct Reference . . . . .	234
10.50.1 Detailed Description . . . . .	234
10.50.2 Field Documentation . . . . .	234
10.50.2.1 networks . . . . .	234
10.51 vxg::cloud::agent::proto::wifi_network Struct Reference . . . . .	235
10.51.1 Detailed Description . . . . .	235
10.51.2 Field Documentation . . . . .	235

10.51.2.1 encryption	236
10.51.2.2 encryption_caps	236
10.51.2.3 mac	236
10.51.2.4 password	236
10.51.2.5 signal	236
10.51.2.6 ssid	236
<b>11 File Documentation</b>	<b>237</b>
11.1 app-dev.md File Reference	237
11.2 arm-example.txt File Reference	237
11.3 base_streamer.h File Reference	237
11.3.1 Macro Definition Documentation	239
11.3.1.1 __BASE_STREAMER_H	239
11.4 build-system.md File Reference	239
11.5 callback.h File Reference	239
11.6 caps.h File Reference	240
11.6.1 Macro Definition Documentation	242
11.6.1.1 ignore_exception	243
11.6.2 Typedef Documentation	243
11.6.2.1 json	243
11.7 cloud-agent-minimal.cc File Reference	243
11.7.1 Function Documentation	244
11.7.1.1 main()	244
11.7.1.2 parse_args()	244
11.7.1.3 signal_handler()	244
11.7.2 Variable Documentation	244
11.7.2.1 props	244
11.7.2.2 quit	245
11.7.2.3 rtsp_url	245
11.7.2.4 vxg_cloud_token	245
11.8 cloud-agent.cc File Reference	245
11.8.1 Function Documentation	246
11.8.1.1 main()	246
11.8.1.2 parse_args()	246
11.8.1.3 signal_handler()	246
11.8.2 Variable Documentation	246
11.8.2.1 quit	247
11.8.2.2 rtsp_url	247
11.8.2.3 vxg_cloud_token	247
11.9 compile.md File Reference	247
11.10 config.h File Reference	247
11.10.1 Detailed Description	250

11.10.2 Macro Definition Documentation	250
11.10.2.1 __CONFIG_H	250
11.11 event-stream.h File Reference	251
11.12 ffmpeg_sink.h File Reference	252
11.13 ffmpeg_source.cc File Reference	253
11.14 ffmpeg_source.h File Reference	253
11.15 logging.h File Reference	254
11.16 mainpage.md File Reference	255
11.17 manager.h File Reference	255
11.18 meson.build File Reference	256
11.19 rtmp_sink.h File Reference	256
11.19.1 Detailed Description	257
11.20 rtmp_source.h File Reference	257
11.20.1 Detailed Description	258
11.21 rtsp-stream.h File Reference	258
11.22 rtsp_source.h File Reference	259
11.22.1 Detailed Description	260
11.23 stream.h File Reference	260
11.24 stream.h File Reference	261
11.25 unset-helper.h File Reference	262
11.25.1 Function Documentation	264
11.25.1.1 __is_unset() [1/2]	264
11.25.1.2 __is_unset() [2/2]	265
11.25.1.3 __is_unset< alter_bool >()	265
11.25.1.4 __is_unset< double >()	265
11.25.1.5 __is_unset< int >()	265
11.25.1.6 __is_unset< nlohmann::json >()	266
11.25.1.7 __is_unset< std::nullptr_t >()	266
11.25.1.8 __is_unset< std::string >()	266
11.25.1.9 __is_unset< vxg::cloud::duration >()	267
11.25.1.10 __is_unset< vxg::cloud::time >()	267
11.25.1.11 unset_value_for()	267
11.25.1.12 unset_value_for_impl() [1/10]	267
11.25.1.13 unset_value_for_impl() [2/10]	268
11.25.1.14 unset_value_for_impl() [3/10]	268
11.25.1.15 unset_value_for_impl() [4/10]	268
11.25.1.16 unset_value_for_impl() [5/10]	268
11.25.1.17 unset_value_for_impl() [6/10]	268
11.25.1.18 unset_value_for_impl() [7/10]	269
11.25.1.19 unset_value_for_impl() [8/10]	269
11.25.1.20 unset_value_for_impl() [9/10]	269
11.25.1.21 unset_value_for_impl() [10/10]	269

---

11.25.2 Variable Documentation . . . . .	269
11.25.2.1 UnsetDouble . . . . .	269
11.25.2.2 UnsetDuration . . . . .	270
11.25.2.3 UnsetFloat . . . . .	270
11.25.2.4 UnsetInt . . . . .	270
11.25.2.5 UnsetInt64 . . . . .	270
11.25.2.6 UnsetString . . . . .	270
11.25.2.7 UnsetTime . . . . .	270
11.25.2.8 UnsetUInt64 . . . . .	271
11.26 utils.h File Reference . . . . .	271
<b>Index</b>	<b>275</b>





## Chapter 1

# VXG Cloud Agent Library

1. [Build system](#)
2. [Library compilation](#)
3. [Application development](#)
4. [API reference](#)



## Chapter 2

# Build System

### 2.0.1 Overview

VXG Cloud Agent library uses [Meson](#) build system as a modern, fast and flexible build system that supports easy to set up and maintain a cross-compilation process.

It's recommended to refer to the [Meson](#) guide.

### 2.0.2 C++ Toolchain Requirements

**IMPORTANT: This projects requires C++ toolchain with C++11 support**

VXG Cloud Agent Library requires modern C++11 so in order to build and use this library the user needs a compiler with C++11 support.

GCC [supports](#) C++11 since version 4.8.1 released on May 31, 2013.

#### C++11 Support in GCC

GCC 4.8.1 was the first feature-complete implementation of the 2011 C++ standard, previously known as C++0x.

This mode can be selected with the `-std=c++11` command-line flag, or `-std=gnu++11` to enable GNU extensions as well.

### 2.0.3 Build system installation

**IMPORTANT: This projects requires Meson version  $\geq$  0.56.0**

It's recommended to use [Ubuntu 20.04 LTS](#) distribution in development process but other distributions or operation systems are also supported by [Meson](#).

Please refer to [Meson installation guide](#) to get and install Meson, preferable way to install Meson is `pip` method.

Quick install guide for Ubuntu 20.04. If you have an old version of meson already installed please remove it first.

```
sudo apt-get update
sudo apt-get install -y python3-pip git ninja-build curl tzdata python3-tz
pip3 install git+https://github.com/mesonbuild/meson@0.56.0
# pip3 puts meson main script into the $HOME/.local/bin/ directory, you need to
# add $HOME/.local/bin/ into your PATH environment variable, for bash shell you
# can run the following command and restart the shell session.
echo 'export PATH=$HOME/.local/bin:$PATH' » $HOME/.bashrc
# Check currently installed meson version
meson -v
```



## Chapter 3

# Application Development

### 3.1 Overview

An application that uses VXG Cloud Agent Library should implement 3 classes derived from the base classes provided by the library:

- `agent::callback` - common callbacks class, only `on_bye` callback is mandatory for implementation
- `agent::media::stream` class, abstract class for media streams, library provides basic `media::rtsp_stream` implementation which retransmits RTSP source stream to the endpoint of the VXG Cloud, all callbacks are stubbed. Developer normally should implement own class derived from the `media::stream` with own `vvg::media::Streamer::ISource` implementation(`vvg::media::ffmpeg::Source` class implementation from the `ffmpeg_source.cc` can be used as a reference), or if RTSP source is acceptable developer can implement own class derived from the `media::rtsp_stream` but with callbacks implemented.
- `agent::event_stream` class, abstract class for events generation.

Any callback implementation as well as `ISource::init` and `ISource::finit` implementations should be non-blocking, VXG Cloud messages processing is single-threaded which means any VXG Cloud messages are handled sequentially hence no new message will be processed until the callback triggered by the previous message is returned.

The library provides the stub implementation for most of the virtual methods of these classes, the stub implementation prints a log message about this method is not implemented and returns an error, the final application should implement all virtual methods on its own.

Most of the callbacks are just getter/setter for the library's `objects`.

### 3.2 Examples

#### 3.2.1 Minimal application example

Headers and namespaces:

```
#include <agent/manager.h>
#include <agent/rtsp-stream.h>
#include <utils/logging.h>
#include <utils/properties.h>
using namespace vvg::cloud;
using namespace vvg::cloud::agent;
```

Common callbacks class, minimal implementation derived from the `agent::callback` class:

```
using namespace vxg::cloud;
class agent_callback_minimal : public agent::callback {
public:
    virtual void on_bye(proto::bye_reason reason) override {
        vxg::logger::warn("Connection close {}", json(reason).dump());
    }
    virtual void on_registered(const std::string& sid) override {
        // Save Cloud registration session id in the local properties file.
        // This is required for the fast reconnection to the Cloud.
        props.set("prev_sid", sid);
    }
};
```

Create and start agent object `agent::manager` with one basic media stream `agent::media::rtsp_stream`

```
using namespace vxg::cloud::agent;
// Agent
manager::ptr agent;
// VXG Cloud token
proto::access_token::ptr access_token =
    proto::access_token::parse(vxg_cloud_token);
// Agent callback
callback::ptr cb = std::make_unique<agent_callback_minimal>();
// Media stream
std::vector<agent::media::stream::ptr> streams;
media::stream::ptr stream =
    std::make_shared<media::rtsp_stream>(rtsp_url, "DemoStream");
streams.push_back(stream);
// Create agent
if ((agent = agent::manager::create(std::move(cb), access_token,
                                    streams)) == nullptr) {
    vxg::logger::error("Failed to create agent");
    return EXIT_FAILURE;
}
if (!quit && !agent->start())
    quit = true;
```

Complete minimal example:

```
#include <signal.h>
#include <args.hxx>
#include <agent/manager.h>
#include <agent/rtsp-stream.h>
#include <utils/logging.h>
#include <utils/properties.h>
using namespace vxg::cloud;
using namespace vxg::cloud::agent;
static bool quit = 0;
static vxg::properties props;
#if !defined(_WIN32)
static void signal_handler(int sig) {
    if (sig == SIGINT || sig == SIGTERM) {
        fprintf(stderr, "\nSIGTERM received\n\n");
        quit = true;
    }
}
#endif
using namespace vxg::cloud;
class agent_callback_minimal : public agent::callback {
public:
    virtual void on_bye(proto::bye_reason reason) override {
        vxg::logger::warn("Connection close {}", json(reason).dump());
    }
    virtual void on_registered(const std::string& sid) override {
        // Save Cloud registration session id in the local properties file.
        // This is required for the fast reconnection to the Cloud.
        props.set("prev_sid", sid);
    }
};
std::string vxg_cloud_token;
std::string rtsp_url;
bool parse_args(int argc, char** argv) {
    args::ArgumentParser parser("This is a test program.", "");
    args::HelpFlag help(parser, "help", "Display this help menu",
                        {'h', "help"});
    args::CompletionFlag completion(parser, {"complete"});
    args::Positional<std::string> token(parser, "vxg_cloud_token",
                                       "VXG Cloud Access Token", "",
                                       args::Options::Required);
    args::Positional<std::string> url(parser, "rtsp_url", "RTSP stream url", "",
                                       args::Options::Required);
    args::Flag secure_connection_arg(
        parser, "",
        "Use secure cloud connection(enables encryption, cloud agent library "
        "must be compiled with openssl support enabled)",
        {"secure-channel", 's'});
```

```

try {
    parser.ParseCLI(argc, argv);
    vxg_cloud_token = args::get(token);
    rtsp_url = args::get(url);
    profile::global::instance().insecure_cloud_channel =
        !args::get(secure_connection_arg);
} catch (const args::RequiredError& e) {
    std::cout << e.what() << std::endl;
    return false;
} catch (const args::Completion& e) {
    std::cout << e.what();
    return false;
} catch (const args::Help&) {
    std::cout << parser;
    return false;
} catch (const args::ParseError& e) {
    std::cerr << e.what() << std::endl;
    std::cerr << parser;
    return false;
}
return true;
}

int main(int argc, char** argv) {
    vxg::properties::reset("agent-test.props");
    // Try to load and set previously saved session id.
    // This is required for the fast reconnection to the Cloud.
    if (!props.get("prev_sid").empty())
        profile::global::instance().cm_registration_sid = props.get("prev_sid");
    // Parse args and retrieve token and rtsp url
    if (!parse_args(argc, argv))
        return EXIT_FAILURE;
#ifdef _WIN32
    // Catch signal
    signal(SIGINT, signal_handler);
    signal(SIGTERM, signal_handler);
    signal(SIGPIPE, signal_handler);
#endif
    vxg::logger::info("VXG Cloud Agent Library Version: {}",
        vxg::cloud::agent::version());
    using namespace vxg::cloud::agent;
    // Agent
    manager::ptr agent;
    // VXG Cloud token
    proto::access_token::ptr access_token =
        proto::access_token::parse(vxg_cloud_token);
    // Agent callback
    callback::ptr cb = std::make_unique<agent_callback_minimal>();
    // Media stream
    std::vector<agent::media::stream::ptr> streams;
    media::stream::ptr stream =
        std::make_shared<media::rtsp_stream>(rtsp_url, "DemoStream");
    streams.push_back(stream);
    // Create agent
    if ((agent = agent::manager::create(std::move(cb), access_token,
        streams)) == nullptr) {
        vxg::logger::error("Failed to create agent");
        return EXIT_FAILURE;
    }
    if (!quit && !agent->start())
        quit = true;
    // Spin main thread until stopped
    while (!quit) {
        std::this_thread::sleep_for(std::chrono::seconds(1));
    }
    agent->stop();
    agent = NULL;
    vxg::logger::info("Agent stopped");
    return EXIT_SUCCESS;
}

```

### 3.2.2 Complete application example

**Common callback class:** derived from `agent::callback`

```

using namespace vxg::cloud;
class my_agent_callback : public agent::callback {
public:
    virtual void on_bye(proto::bye_reason reason) override {
        vxg::logger::error("Error {}", json(reason).dump());
    }
    virtual bool on_raw_msg(std::string client_id, std::string& data) override {
        vxg::logger::info("Raw message {} from client '{}'", data, client_id);
        // Reply json
    }
}

```

```

        data = "{\"reply\": \"OK\"}";
        return true;
    }
    virtual bool on_get_log(std::string& log_data) override {
        log_data = "log messages...";
        vxg::logger::warn("{} not implemented", __func__);
        return true;
    }
    virtual bool on_start_backward_audio(std::string url) override {
        // Start backward audio playback from url
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_stop_backward_audio(std::string url) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_cam_video_config(proto::video_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_set_cam_video_config(
        const proto::video_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_cam_audio_config(proto::audio_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_set_cam_audio_config(
        const proto::audio_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_ptz_config(proto::ptz_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_cam_ptz(proto::ptz_command& command) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_osd_config(proto::osd_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_set_osd_config(const proto::osd_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_wifi_config(proto::wifi_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_set_wifi_config(
        const proto::wifi_network& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_motion_detection_config(
        proto::motion_detection_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_set_motion_detection_config(
        const proto::motion_detection_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_cam_events_config(
        proto::events_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_set_cam_events_config(
        const proto::events_config& config) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_get_timezone(std::string& timezone) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool on_set_timezone(std::string timezone) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }

```



```

    }
    virtual bool on_get_memorycard_info(
        proto::event_object::memorycard_info_object& info) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
};

```

#### Media stream callback class: derived from `agent::media::stream`

```

class my_media_stream : public media::rtsp_stream {
public:
    my_media_stream(std::string url, std::string name)
        : media::rtsp_stream(url, name) {}
    bool get_supported_stream(proto::supported_stream_config& config) override {
        vxg::logger::warn("{} default implementation should be overridden",
            __func__);
        config.id = cloud_name();
        config.video = "Video" + std::to_string(0);
        // config.audio = "Audio" + std::to_string(0);
        return true;
    }
    virtual bool get_stream_caps(proto::stream_caps& caps) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool get_stream_config(
        proto::stream_config& streamConfig) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool set_stream_config(
        const proto::stream_config& streamConfig) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool get_snapshot(
        proto::event_object::snapshot_info_object& snapshot) override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual std::vector<proto::video_clip_info> record_get_list(
        vxg::cloud::time begin,
        vxg::cloud::time end,
        bool align) override {
        std::vector<proto::video_clip_info> empty_vector(0);
        vxg::logger::warn("{} not implemented", __func__);
        return empty_vector;
    }
    virtual proto::video_clip_info record_export(
        vxg::cloud::time begin,
        vxg::cloud::time end) override {
        proto::video_clip_info clip;
        vxg::logger::warn("{} not implemented", __func__);
        // empty clip
        return clip;
    }
    virtual bool start_record() override {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool stop_record() override {
        vxg::logger::warn("{} not implemented", __func__);
        return true;
    }
};

```

#### Event stream callback class: derived from `agent::media::event_stream`

```

class my_event_stream : public agent::event_stream {
public:
    my_event_stream(std::string name) : agent::event_stream(name) {}
    virtual bool start() {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual void stop() { vxg::logger::warn("{} not implemented", __func__); }
    virtual bool init() {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual void finit() { vxg::logger::warn("{} not implemented", __func__); }
    virtual bool set_trigger_recording(bool enabled, int pre, int post) {
        vxg::logger::warn("{} not implemented", __func__);
        return false;
    }
    virtual bool get_events(std::vector<proto::event_config>& configs) {

```

```

        return false;
    }
    virtual bool set_events(const std::vector<proto::event_config>& config) {
        return false;
    }
};

```

### Creating and start agent instance with all callbacks:

```

using namespace vxg::cloud::agent;
// Agent
manager::ptr agent;
// VXG Cloud token
proto::access_token::ptr access_token =
    proto::access_token::parse(vxg_cloud_token);
// Agent callback
callback::ptr cb = std::make_unique<my_agent_callback>();
// Media stream
std::vector<agent::media::stream::ptr> streams;
media::stream::ptr stream =
    std::make_shared<my_media_stream>(rtsp_url, "MyMediaStream");
streams.push_back(stream);
// Event stream
std::vector<agent::event_stream::ptr> event_streams;
event_stream::ptr event_stream =
    std::make_shared<my_event_stream>("MyEventStream");
event_streams.push_back(event_stream);
// Create agent
if ((agent = agent::manager::create(std::move(cb), access_token, streams,
                                     event_streams)) == nullptr) {
    vxg::logger::error("Failed to create agent");
    return EXIT_FAILURE;
}
if (!quit && !agent->start())
    quit = true;

```

## 3.2.3 Linking application against the VXG Agent Cloud Library

There are 3 possible ways of how to build and link your application

1. Building the application inside the VXG CCloud Agent library's Meson project, the app will be assembled during the library project compilation in this case.

You need to add a new executable target into the main `meson.build` file, please refer to the example app build target declaration:

```

cloud_agent_minimal = executable('cloud-agent-minimal', 'src/cloud-agent-minimal.cc',
    install : true, dependencies: dep)

```

User must declare own executable target with a list of sources and dependencies, user may need to declare own dependencies if application requires it.

**This method is not recommended as it makes updating of the VXG Cloud Agent library mostly not possible or very difficult for application developer**

2. Building your app using your own build system and linking against the installed library.  
Running the `install` step from the `compile` section installs the binary libraries and headers into the directory you specified during the `setup` step, it also puts the `pkg-config's .pc` files into the prefix directory which could be used by your own build system.
3. Preferred and recommended way of application development is to hold the app as a separate Meson project and use the VXG Cloud Agent library as a Meson subproject of the application's Meson project.

Using this approach gives the most flexible and convenient workflow for updating the VXG Cloud Library, all library dependencies will be promoted to the main project and will be also accessible by the application.

#### How does it work

- Assuming you have a Meson build system [installed](#)
- Start a new Meson project with a following command:  

```
meson init -l cpp -n your-project-name
```

- As a result of this command you should have the following files tree:  

```
|-- meson.build
|-- your_project_name.cpp
```
- Add VXG Cloud Agent library as a Meson subproject  
All subprojects should be located in the subprojects directory so you have to create it first  

```
mkdir subprojects
```

Now you have 2 options depending on how you want to store the VXG Cloud Agent library sources:

(a) If you want to store the VXG Cloud Agent library as a files tree locally.

- Create a symlink to the library path inside the subprojects dir:  

```
ln -s path/to/vxgcloudagent subprojects/vxgcloudagent
```

Or you can just move vxgcloudagent directory inside the subprojects dir.

- Create a library's Meson wrap file inside the subprojects dir, the name of the file should be the same as symlink you created in 1.1 and the content of the file should be:

```
[wrap-file]
directory = vxgcloudagent
[provide]
vxgcloudagent = vxgcloudagent_dep
```

(b) If you want to store the library in a git repository you just need to create a wrap file with the content like below:

```
[wrap-git]
url=https://your-git-repo-url.com/path/vxgcloudagent.git
# You can specify tag, branch or commit hash as revision
revision=master
[provide]
vxgcloudagent = vxgcloudagent_dep
```

You can find the example app Meson project in the example/app directory of the VXG Cloud library sources package.



## Chapter 4

# Library Compilation Guide

### 4.0.1 Library build process

Here is a compilation quickstart guide:

- First of all you need to have a build system and toolchain [installed](#)
- **Setup the build directory**  

```
meson setup --prefix=path/to/install --strip -Dbuildtype=debug builddir/  
# --prefix=path specifies the installation path  
# --strip indicates that final binaries should be stripped  
# -Dbuildtype= specifies the debug/release build type, please check the Meson docs about full list of  
the build types.
```

- **Build**  

```
meson compile -C builddir  
# Or  
ninja -C builddir
```

- **Install**  

```
meson install -C builddir  
# Or  
ninja -C builddir/ install
```

As a result of the `install` step you should have the library compiled and installed into the prefix directory you specified during the `setup` step.

- **Clean**  

```
ninja -C builddir clean
```

Or you can just delete the `builddir`, you will need to `setup` it again in this case.  

```
rm -rf builddir
```

### 4.0.2 Cross-compilation

- By default `Meson` builds project for the host platform, but it's also possible to cross-compile the library and your application using `Meson`.
- Full `Meson` cross-compilation documentation can be found [here](#).
- The difference between the host compilation described above and the cross-compilation is the additional `--cross-file=path/to/cross-file.txt` flag for the Meson Setup step, the Setup command should look like below:

```
meson setup --prefix=path/to/install --strip -Dbuildtype=debug --cross-file=path/to/cross-file.txt  
builddir/
```

`cross-file.txt` is the target platform description which in terms of Meson called a `cross-file`.

- cross-file example below is for the Debian provided arm-linux-gnueabihf toolchain installable using the Ubuntu's package manager command  
`sudo apt install g++-arm-linux-gnueabihf`

- Example of the ARMv7 cross-file:

```
[host_machine]
system = 'linux'
cpu_family = 'arm'
cpu = 'armv7-a'
endian = 'little'
[built-in options]
# Example of platform specific CFLAGS and CXXFLAGS
c_args = ['-mfloat-abi=hard', '-march=armv7-a+vfpv3']
cpp_args = c_args
default_library = 'static'
[properties]
# If your toolchain requires specifying the sysroot dir you can setup it like below, sysroot_dir is a
# constant declared in [constants] section of the cross-file
#sys_root = sysroot_dir
# Meson uses pkg-config and cmake to detect external dependencies
# Set the correct path to your cross-compilation pkgconfig directory if your app depends on some
# external dependencies like platform specific libs.
#pkg_config_libdir = sysroot_dir / 'usr/lib/pkgconfig/'
[constants]
cross_prefix = 'arm-linux-gnueabihf-'
#sysroot_dir = '/opt/arm-linux-gnueabihf/sysroot/'
[binaries]
c = cross_prefix + 'gcc'
cpp = cross_prefix + 'g++'
ar = cross_prefix + 'ar'
strip = cross_prefix + 'strip'
# You should specify your platform toolchain pkg-config binary here
#pkgconfig = '/opt/arm-linux-gnueabihf/bin/pkg-config'
```

## Chapter 5

# Deprecated List

Global [vxg::logger::reset](#) (int argc, char \*\*argv, loglevel l, std::string syslog\_ident="VXGCloudAgent↵  
Default", std::string crash\_logfile\_path="", std::string logfile\_path="", size\_t logfile\_max\_size=(1024  
\*1024), size\_t logfile\_max\_files=3)

Use [reset\(const options& opts\)](#)





## Chapter 6

# Hierarchical Index

### 6.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

vxg::cloud::agent::access_token . . . . .	65
alter_bool . . . . .	66
vxg::cloud::agent::proto::audio_caps . . . . .	69
vxg::cloud::agent::audio_config . . . . .	72
vxg::cloud::agent::proto::audio_stream_config . . . . .	74
vxg::media::Streamer::StreamInfo::AudioInfo . . . . .	76
vxg::cloud::agent::callback . . . . .	78
vxg::cloud::agent::proto::stream_caps::caps_audio_object . . . . .	91
vxg::cloud::agent::proto::stream_caps::caps_video_object . . . . .	93
command_handler	
vxg::cloud::agent::manager . . . . .	126
common	
vxg::media::ffmpeg::Sink . . . . .	185
vxg::media::rtmp_sink . . . . .	168
vxg::media::ffmpeg::Source . . . . .	190
vxg::media::rtmp_source . . . . .	172
vxg::media::rtsp_source . . . . .	173
vxg::cloud::agent::proto::event_caps . . . . .	96
vxg::cloud::agent::event_config . . . . .	98
vxg::cloud::agent::manager::event_state::event_state_caps . . . . .	102
vxg::cloud::agent::event_stream . . . . .	103
vxg::cloud::agent::events_config . . . . .	107
vxg::media::Streamer::ISink . . . . .	110
vxg::media::ffmpeg::Sink . . . . .	185
vxg::media::Streamer::ISource . . . . .	115
vxg::media::ffmpeg::Source . . . . .	190
vxg::logger . . . . .	120
vxg::media::Streamer::MediaFrame . . . . .	143
vxg::cloud::agent::proto::motion_detection_caps . . . . .	147
vxg::cloud::agent::proto::motion_detection_config . . . . .	148
vxg::cloud::agent::proto::motion_region . . . . .	150
vxg::logger::options . . . . .	152
vxg::cloud::agent::proto::osd_caps . . . . .	155
vxg::cloud::agent::osd_config . . . . .	158

vxg::cloud::agent::access_token::proxy_config . . . . .	162
vxg::cloud::agent::ptz_command . . . . .	163
vxg::cloud::agent::ptz_config . . . . .	164
vxg::cloud::agent::ptz_preset . . . . .	166
vxg::media::stream . . . . .	202
vxg::cloud::agent::media::stream . . . . .	195
vxg::cloud::agent::media::rtsp_stream . . . . .	178
vxg::cloud::agent::proto::stream_caps . . . . .	206
vxg::cloud::agent::proto::stream_config . . . . .	207
vxg::media::Streamer::StreamInfo . . . . .	209
std::string[external]	
vxg::cloud::utils::motion::map . . . . .	141
vxg::cloud::agent::supported_stream_config . . . . .	212
vxg::cloud::agent::supported_streams_config . . . . .	213
vxg::cloud::utils::uri . . . . .	215
vxg::cloud::agent::proto::video_caps . . . . .	217
vxg::cloud::agent::proto::video_clip_info . . . . .	221
vxg::cloud::agent::proto::video_config . . . . .	223
vxg::cloud::agent::proto::video_stream_config . . . . .	228
vxg::media::Streamer::StreamInfo::VideoInfo . . . . .	231
vxg::cloud::agent::proto::wifi_config . . . . .	234
vxg::cloud::agent::proto::wifi_network . . . . .	235

## Chapter 7

# Data Structure Index

### 7.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">vxd::cloud::agent::access_token</a>	65
VXD Cloud access token . . . . .	
<a href="#">alter_bool</a>	66
Alternative bool class Standard bool type has two states, this class adds 3rd state - undefined . . . . .	
<a href="#">vxd::cloud::agent::proto::audio_caps</a>	69
Audio capabilities . . . . .	
<a href="#">vxd::cloud::agent::audio_config</a>	72
Audio config . . . . .	
<a href="#">vxd::cloud::agent::proto::audio_stream_config</a>	74
Audio media stream config . . . . .	
<a href="#">vxd::media::Streamer::StreamInfo::AudioInfo</a>	76
Audio stream info . . . . .	
<a href="#">vxd::cloud::agent::callback</a>	78
VXD Cloud manager common callbacks class . . . . .	
<a href="#">vxd::cloud::agent::proto::stream_caps::caps_audio_object</a>	91
Audio streams capabilities . . . . .	
<a href="#">vxd::cloud::agent::proto::stream_caps::caps_video_object</a>	93
Video streams capabilities . . . . .	
<a href="#">vxd::cloud::agent::proto::event_caps</a>	96
Events capabilities . . . . .	
<a href="#">vxd::cloud::agent::event_config</a>	98
Event config . . . . .	
<a href="#">vxd::cloud::agent::manager::event_state::event_state_caps</a>	102
<a href="#">vxd::cloud::agent::event_stream</a>	103
Event stream, abstract class for event generation . . . . .	
<a href="#">vxd::cloud::agent::events_config</a>	107
Events config, list of <a href="#">event_config</a> objects . . . . .	
<a href="#">vxd::media::Streamer::ISink</a>	110
<a href="#">vxd::media::Streamer::ISource</a>	115
ISource interface class . . . . .	
<a href="#">vxd::logger</a>	120
Logger class, current implementation based on spdlog . . . . .	
<a href="#">vxd::cloud::agent::manager</a>	126
VXD Cloud agent manager class . . . . .	
<a href="#">vxd::cloud::utils::motion::map</a>	141

<a href="#">vxg::media::Streamer::MediaFrame</a>	
Media frame container	143
<a href="#">vxg::cloud::agent::proto::motion_detection_caps</a>	
Motion detection capabilities camera capabilities that limit possible motion detection configuration	147
<a href="#">vxg::cloud::agent::proto::motion_detection_config</a>	
Motion detection config	148
<a href="#">vxg::cloud::agent::proto::motion_region</a>	
Motion detection related structs	150
<a href="#">vxg::logger::options</a>	152
<a href="#">vxg::cloud::agent::proto::osd_caps</a>	
OSD capabilities	155
<a href="#">vxg::cloud::agent::osd_config</a>	
OSD config	158
<a href="#">vxg::cloud::agent::access_token::proxy_config</a>	
Socks proxy settings	162
<a href="#">vxg::cloud::agent::ptz_command</a>	
PTZ command	163
<a href="#">vxg::cloud::agent::ptz_config</a>	
PTZ config	164
<a href="#">vxg::cloud::agent::ptz_preset</a>	
PTZ preset	166
<a href="#">vxg::media::rtmp_sink</a>	
RTMP sink class	168
<a href="#">vxg::media::rtmp_source</a>	
RTMP source class	172
<a href="#">vxg::media::rtsp_source</a>	
RTSP source class	173
<a href="#">vxg::cloud::agent::media::rtsp_stream</a>	
Implementation of the <a href="#">media::stream</a> with RTSP source and NIY stubs	178
<a href="#">vxg::media::ffmpeg::Sink</a>	
Base ffmpeg sink class	185
<a href="#">vxg::media::ffmpeg::Source</a>	
Base ffmpeg source class	190
<a href="#">vxg::cloud::agent::media::stream</a>	
Cloud agent media stream abstract class	195
<a href="#">vxg::media::stream</a>	
Base media stream abstract class	202
<a href="#">vxg::cloud::agent::proto::stream_caps</a>	
Media stream capabilities	206
<a href="#">vxg::cloud::agent::proto::stream_config</a>	
Media stream config	207
<a href="#">vxg::media::Streamer::StreamInfo</a>	
Stream info description	209
<a href="#">vxg::cloud::agent::supported_stream_config</a>	
Supported stream config	212
<a href="#">vxg::cloud::agent::supported_streams_config</a>	
Supported streams config, list of <a href="#">supported_stream_config</a>	213
<a href="#">vxg::cloud::utils::uri</a>	215
<a href="#">vxg::cloud::agent::proto::video_caps</a>	
Video image capabilities	217
<a href="#">vxg::cloud::agent::proto::video_clip_info</a>	
Video recoding(mp4 file) clip description,	221
<a href="#">vxg::cloud::agent::proto::video_config</a>	
Video image config	223
<a href="#">vxg::cloud::agent::proto::video_stream_config</a>	
Video stream config	228

<a href="#">vxd::media::Streamer::StreamInfo::VideoInfo</a>	
Video stream info . . . . .	231
<a href="#">vxd::cloud::agent::proto::wifi_config</a>	
WiFi config . . . . .	234
<a href="#">vxd::cloud::agent::proto::wifi_network</a>	
WiFi network object . . . . .	235



## Chapter 8

# File Index

### 8.1 File List

Here is a list of all files with brief descriptions:

<a href="#">base_streamer.h</a>	237
<a href="#">callback.h</a>	239
<a href="#">caps.h</a>	240
<a href="#">cloud-agent-minimal.cc</a>	243
<a href="#">cloud-agent.cc</a>	245
<a href="#">config.h</a>	247
<a href="#">event-stream.h</a>	251
<a href="#">ffmpeg_sink.h</a>	252
<a href="#">ffmpeg_source.cc</a>	253
<a href="#">ffmpeg_source.h</a>	253
<a href="#">logging.h</a>	254
<a href="#">manager.h</a>	255
<a href="#">meson.build</a>	256
<a href="#">rtmp_sink.h</a>	256
<a href="#">rtmp_source.h</a>	257
<a href="#">rtsp-stream.h</a>	258
<a href="#">rtsp_source.h</a>	259
<a href="#">streamer/stream.h</a>	260
<a href="#">agent/stream.h</a>	261
<a href="#">unset-helper.h</a>	262
<a href="#">utils.h</a>	271





## Chapter 9

# Namespace Documentation

### 9.1 nlohmann Namespace Reference

### 9.2 std Namespace Reference

#### Namespaces

- `chrono`
- `experimental`
- `regex_constants`
- `rel_ops`
- `this_thread`

#### Data Structures

- class `add_const`
- class `add_cv`
- class `add_lvalue_reference`
- class `add_pointer`
- class `add_rvalue_reference`
- class `add_volatile`
- class `adopt_lock_t`
- class `aligned_storage`
- class `aligned_union`
- class `alignment_of`
- class `allocator`
- class `allocator_arg_t`
- class `allocator_traits`
- class `array`
- class `atomic`
- class `atomic_flag`
- class `auto_ptr`
- class `back_insert_iterator`
- class `bad_alloc`
- class `bad_array_length`
- class `bad_array_new_length`

- class **bad\_cast**
- class **bad\_exception**
- class **bad\_function\_call**
- class **bad\_optional\_access**
- class **bad\_typeid**
- class **bad\_weak\_ptr**
- class **basic\_filebuf**
- class **basic\_fstream**
- class **basic\_ifstream**
- class **basic\_ios**
- class **basic\_iostream**
- class **basic\_istream**
- class **basic\_istreamstream**
- class **basic\_ofstream**
- class **basic\_ostream**
- class **basic\_ostreamstream**
- class **basic\_regex**
- class **basic\_streambuf**
- class **basic\_string**
- class **basic\_stringbuf**
- class **basic\_stringstream**
- class **bernoulli\_distribution**
- class **bidirectional\_iterator\_tag**
- class **binary\_function**
- class **binary\_negate**
- class **binomial\_distribution**
- class **bit\_and**
- class **bit\_not**
- class **bit\_or**
- class **bitset**
- class **cauchy\_distribution**
- class **centi**
- class **cerr**
- class **char\_traits**
- class **chi\_squared\_distribution**
- class **cin**
- class **clock\_t**
- class **clog**
- class **cmatch**
- class **codecvt**
- class **codecvt\_base**
- class **codecvt\_byname**
- class **codecvt\_utf16**
- class **codecvt\_utf8**
- class **codecvt\_utf8\_utf16**
- class **collate**
- class **collate\_byname**
- class **common\_type**
- class **complex**
- class **condition\_variable**
- class **condition\_variable\_any**
- class **conditional**
- class **cout**
- class **cregex\_iterator**
- class **cregex\_token\_iterator**

- class **csub\_match**
- class **ctype**
- class **ctype\_base**
- class **ctype\_byname**
- class **deca**
- class **decay**
- class **deci**
- class **default\_delete**
- class **default\_random\_engine**
- class **defer\_lock\_t**
- class **deque**
- class **discard\_block\_engine**
- class **discrete\_distribution**
- class **divides**
- class **domain\_error**
- class **dynarray**
- class **enable\_if**
- class **enable\_shared\_from\_this**
- class **equal\_to**
- class **errc**
- class **error\_category**
- class **error\_code**
- class **error\_condition**
- class **exa**
- class **exception**
- class **exception\_ptr**
- class **exponential\_distribution**
- class **extent**
- class **extreme\_value\_distribution**
- class **false\_type**
- class **femto**
- class **FILE**
- class **filebuf**
- class **fisher\_f\_distribution**
- class **forward\_iterator\_tag**
- class **forward\_list**
- class **fpos**
- class **fpos\_t**
- class **front\_insert\_iterator**
- class **fstream**
- class **function**
- class **future**
- class **future\_error**
- class **gamma\_distribution**
- class **geometric\_distribution**
- class **giga**
- class **greater**
- class **greater\_equal**
- class **has\_virtual\_destructor**
- class **hash**
- class **hecto**
- class **ifstream**
- class **independent\_bits\_engine**
- class **initializer\_list**
- class **input\_iterator\_tag**

- class **insert\_iterator**
- class **int16\_t**
- class **int32\_t**
- class **int64\_t**
- class **int8\_t**
- class **int\_fast16\_t**
- class **int\_fast32\_t**
- class **int\_fast64\_t**
- class **int\_fast8\_t**
- class **int\_least16\_t**
- class **int\_least32\_t**
- class **int\_least64\_t**
- class **int\_least8\_t**
- class **integer\_sequence**
- class **integral\_constant**
- class **intmax\_t**
- class **intptr\_t**
- class **invalid\_argument**
- class **ios\_base**
- class **iostream**
- class **is\_abstract**
- class **is\_arithmetic**
- class **is\_array**
- class **is\_assignable**
- class **is\_base\_of**
- class **is\_bind\_expression**
- class **is\_class**
- class **is\_compound**
- class **is\_const**
- class **is\_constructible**
- class **is\_convertible**
- class **is\_copy\_assignable**
- class **is\_copy\_constructible**
- class **is\_default\_constructible**
- class **is\_destructible**
- class **is\_empty**
- class **is\_enum**
- class **is\_error\_code\_enum**
- class **is\_error\_condition\_enum**
- class **is\_floating\_point**
- class **is\_function**
- class **is\_fundamental**
- class **is\_integral**
- class **is\_literal\_type**
- class **is\_lvalue\_reference**
- class **is\_member\_function\_pointer**
- class **is\_member\_object\_pointer**
- class **is\_member\_pointer**
- class **is\_move\_assignable**
- class **is\_move\_constructible**
- class **is\_nothrow\_assignable**
- class **is\_nothrow\_constructible**
- class **is\_nothrow\_copy\_assignable**
- class **is\_nothrow\_copy\_constructible**
- class **is\_nothrow\_default\_constructible**

- class **is\_nothrow\_destructible**
- class **is\_nothrow\_move\_assignable**
- class **is\_nothrow\_move\_constructible**
- class **is\_object**
- class **is\_placeholder**
- class **is\_pod**
- class **is\_pointer**
- class **is\_polymorphic**
- class **is\_reference**
- class **is\_rvalue\_reference**
- class **is\_same**
- class **is\_scalar**
- class **is\_signed**
- class **is\_standard\_layout**
- class **is\_trivial**
- class **is\_trivially\_assignable**
- class **is\_trivially\_constructible**
- class **is\_trivially\_copy\_assignable**
- class **is\_trivially\_copy\_constructible**
- class **is\_trivially\_copyable**
- class **is\_trivially\_default\_constructible**
- class **is\_trivially\_destructible**
- class **is\_trivially\_move\_assignable**
- class **is\_trivially\_move\_constructible**
- class **is\_union**
- class **is\_unsigned**
- class **is\_void**
- class **is\_volatile**
- class **istream**
- class **istream\_iterator**
- class **istreambuf\_iterator**
- class **istreamstringstream**
- class **istrstream**
- class **iterator**
- class **iterator\_traits**
- class **jmp\_buf**
- class **kilo**
- class **knuth\_b**
- class **lconv**
- class **length\_error**
- class **less**
- class **less\_equal**
- class **linear\_congruential\_engine**
- class **list**
- class **locale**
- class **lock\_guard**
- class **logic\_error**
- class **logical\_and**
- class **logical\_not**
- class **logical\_or**
- class **lognormal\_distribution**
- class **make\_signed**
- class **make\_unsigned**
- class **map**
- class **match\_results**

- class **max\_align\_t**
- class **mbstate\_t**
- class **mega**
- class **mersenne\_twister\_engine**
- class **messages**
- class **messages\_base**
- class **messages\_byname**
- class **micro**
- class **milli**
- class **minstd\_rand**
- class **minstd\_rand0**
- class **minus**
- class **modulus**
- class **money\_base**
- class **money\_get**
- class **money\_put**
- class **moneypunct**
- class **moneypunct\_byname**
- class **move\_iterator**
- class **mt19937**
- class **mt19937\_64**
- class **multimap**
- class **multiplies**
- class **multiset**
- class **mutex**
- class **nano**
- class **negate**
- class **negative\_binomial\_distribution**
- class **nested\_exception**
- class **new\_handler**
- class **normal\_distribution**
- class **not\_equal\_to**
- class **nothrow\_t**
- class **nullptr\_t**
- class **num\_get**
- class **num\_put**
- class **numeric\_limits**
- class **numpunct**
- class **numpunct\_byname**
- class **ofstream**
- class **once\_flag**
- class **ostream**
- class **ostream\_iterator**
- class **ostreambuf\_iterator**
- class **ostringstream**
- class **ostrstream**
- class **out\_of\_range**
- class **output\_iterator\_tag**
- class **overflow\_error**
- class **owner\_less**
- class **packaged\_task**
- class **pair**
- class **peta**
- class **pico**
- class **piecewise\_constant\_distribution**

- class **piecewise\_construct\_t**
- class **piecewise\_linear\_distribution**
- class **placeholders**
- class **plus**
- class **pointer\_safety**
- class **pointer\_traits**
- class **poisson\_distribution**
- class **priority\_queue**
- class **promise**
- class **ptrdiff\_t**
- class **queue**
- class **random\_access\_iterator\_tag**
- class **random\_device**
- class **range\_error**
- class **rank**
- class **ranlux24**
- class **ranlux24\_base**
- class **ranlux48**
- class **ranlux48\_base**
- class **ratio**
- class **ratio\_add**
- class **ratio\_divide**
- class **ratio\_equal**
- class **ratio\_greater**
- class **ratio\_greater\_equal**
- class **ratio\_less**
- class **ratio\_less\_equal**
- class **ratio\_multiply**
- class **ratio\_not\_equal**
- class **ratio\_subtract**
- class **raw\_storage\_iterator**
- class **recursive\_mutex**
- class **recursive\_timed\_mutex**
- class **reference\_wrapper**
- class **regex**
- class **regex\_error**
- class **regex\_iterator**
- class **regex\_token\_iterator**
- class **regex\_traits**
- class **remove\_all\_extents**
- class **remove\_const**
- class **remove\_cv**
- class **remove\_extent**
- class **remove\_pointer**
- class **remove\_reference**
- class **remove\_volatile**
- class **result\_of**
- class **reverse\_iterator**
- class **runtime\_error**
- class **scoped\_allocator\_adaptor**
- class **seed\_seq**
- class **set**
- class **shared\_future**
- class **shared\_lock**
- class **shared\_ptr**

- class **shared\_timed\_mutex**
- class **shuffle\_order\_engine**
- class **sig\_atomic\_t**
- class **size\_t**
- class **smatch**
- class **sregex\_iterator**
- class **sregex\_token\_iterator**
- class **ssub\_match**
- class **stack**
- class **streambuf**
- class **streamoff**
- class **streampos**
- class **streamsize**
- class **string**
- class **stringbuf**
- class **stringstream**
- class **strstream**
- class **strstreambuf**
- class **student\_t\_distribution**
- class **sub\_match**
- class **subtract\_with\_carry\_engine**
- class **system\_error**
- class **tera**
- class **terminate\_handler**
- class **thread**
- class **time\_base**
- class **time\_get**
- class **time\_get\_byname**
- class **time\_put**
- class **time\_put\_byname**
- class **time\_t**
- class **timed\_mutex**
- class **tm**
- class **true\_type**
- class **try\_to\_lock\_t**
- class **tuple**
- class **type\_index**
- class **type\_info**
- class **u16streampos**
- class **u16string**
- class **u32streampos**
- class **u32string**
- class **uint16\_t**
- class **uint32\_t**
- class **uint64\_t**
- class **uint8\_t**
- class **uint\_fast16\_t**
- class **uint\_fast32\_t**
- class **uint\_fast64\_t**
- class **uint\_fast8\_t**
- class **uint\_least16\_t**
- class **uint\_least32\_t**
- class **uint\_least64\_t**
- class **uint\_least8\_t**
- class **uintmax\_t**



- class **uintptr\_t**
- class **unary\_function**
- class **unary\_negate**
- class **underflow\_error**
- class **underlying\_type**
- class **unexpected\_handler**
- class **uniform\_int\_distribution**
- class **uniform\_real\_distribution**
- class **unique\_lock**
- class **unique\_ptr**
- class **unordered\_map**
- class **unordered\_multimap**
- class **unordered\_multiset**
- class **unordered\_set**
- class **uses\_allocator**
- class **valarray**
- class **vector**
- class **wbuffer\_convert**
- class **wcerr**
- class **wcin**
- class **wclog**
- class **wcmatch**
- class **wcout**
- class **wcregex\_iterator**
- class **wcregex\_token\_iterator**
- class **wcsub\_match**
- class **weak\_ptr**
- class **weibull\_distribution**
- class **wfilebuf**
- class **wfstream**
- class **wifstream**
- class **wiostream**
- class **wistream**
- class **wistringstream**
- class **wofstream**
- class **wostream**
- class **wostringstream**
- class **wregex**
- class **wsmatch**
- class **wsregex\_iterator**
- class **wsregex\_token\_iterator**
- class **wssub\_match**
- class **wstreambuf**
- class **wstreampos**
- class **wstring**
- class **wstring\_convert**
- class **wstringbuf**
- class **wstringstream**
- class **yocto**
- class **yotta**
- class **zetta**

## Functions

- T **atomic\_fetch\_and\_explicit** (T... args)
- T **atomic\_fetch\_xor\_explicit** (T... args)
- T **set\_unexpected** (T... args)
- T **fputs** (T... args)
- T **modf** (T... args)
- T **not2** (T... args)
- T **strlen** (T... args)
- T **exp2** (T... args)
- T **setiosflags** (T... args)
- T **adjacent\_difference** (T... args)
- T **cos** (T... args)
- T **fwscanf** (T... args)
- T **atomic\_init** (T... args)
- T **forward\_as\_tuple** (T... args)
- T **abort** (T... args)
- T **wcsncmp** (T... args)
- T **set\_intersection** (T... args)
- T **atomic\_signal\_fence** (T... args)
- T **llabs** (T... args)
- T **make\_move\_iterator** (T... args)
- T **scanf** (T... args)
- T **nextafter** (T... args)
- T **stol** (T... args)
- T **strcspn** (T... args)
- T **ungetwc** (T... args)
- T **transform** (T... args)
- T **putc** (T... args)
- T **iswdigit** (T... args)
- T **rint** (T... args)
- T **memset** (T... args)
- T **isgraph** (T... args)
- T **replace\_copy\_if** (T... args)
- T **scalbn** (T... args)
- T **partial\_sort\_copy** (T... args)
- T **make\_exception\_ptr** (T... args)
- T **frexp** (T... args)
- T **isxdigit** (T... args)
- T **atomic\_exchange\_explicit** (T... args)
- T **wprintf** (T... args)
- T **fdim** (T... args)
- T **wctype** (T... args)
- T **mbrtoc32** (T... args)
- T **setw** (T... args)
- T **get\_temporary\_buffer** (T... args)
- T **fmax** (T... args)
- T **atomic\_thread\_fence** (T... args)
- T **atomic\_exchange** (T... args)
- T **fgetwc** (T... args)
- T **swprintf** (T... args)
- T **prev\_permutation** (T... args)
- T **max\_element** (T... args)
- T **set\_symmetric\_difference** (T... args)
- T **wcscpy** (T... args)

- T **const\_pointer\_cast** (T... args)
- T **minmax\_element** (T... args)
- T **wcstok** (T... args)
- T **ref** (T... args)
- T **feupdateenv** (T... args)
- T **endl** (T... args)
- T **end** (T... args)
- T **wmemmove** (T... args)
- T **fmin** (T... args)
- T **uninitialized\_fill\_n** (T... args)
- T **nouppercase** (T... args)
- T **noshowpos** (T... args)
- T **ctime** (T... args)
- T **wmemset** (T... args)
- T **iswpunct** (T... args)
- T **pop\_heap** (T... args)
- T **sprintf** (T... args)
- T **fixed** (T... args)
- T **make\_shared** (T... args)
- T **make\_heap** (T... args)
- T **fmod** (T... args)
- T **atol** (T... args)
- T **uninitialized\_copy** (T... args)
- T **dynamic\_pointer\_cast** (T... args)
- T **set\_union** (T... args)
- T **hexfloat** (T... args)
- T **vswprintf** (T... args)
- T **asctime** (T... args)
- T **iswspace** (T... args)
- T **nan** (T... args)
- T **sort** (T... args)
- T **quick\_exit** (T... args)
- T **log10** (T... args)
- T **mbstowcs** (T... args)
- T **isspace** (T... args)
- T **strncat** (T... args)
- T **isinf** (T... args)
- T **atof** (T... args)
- T **erf** (T... args)
- T **is\_sorted\_until** (T... args)
- T **cbrt** (T... args)
- T **log1p** (T... args)
- T **return\_temporary\_buffer** (T... args)
- T **mbsrtowcs** (T... args)
- T **feraiseexcept** (T... args)
- T **fseek** (T... args)
- T **atomic\_fetch\_or\_explicit** (T... args)
- T **log** (T... args)
- T **putchar** (T... args)
- T **make\_tuple** (T... args)
- T **expm1** (T... args)
- T **fma** (T... args)
- T **remove\_copy\_if** (T... args)
- T **showpoint** (T... args)
- T **fscanf** (T... args)

- T **stable\_partition** (T... args)
- T **fill\_n** (T... args)
- T **remove\_copy** (T... args)
- T **atomic\_compare\_exchange\_strong\_explicit** (T... args)
- T **wctomb** (T... args)
- T **fgets** (T... args)
- T **remainder** (T... args)
- T **allocate\_shared** (T... args)
- T **unique** (T... args)
- T **includes** (T... args)
- T **iswalnum** (T... args)
- T **exit** (T... args)
- T **put\_time** (T... args)
- T **to\_string** (T... args)
- T **is\_heap\_until** (T... args)
- T **wcstold** (T... args)
- T **stold** (T... args)
- T **ftell** (T... args)
- T **copy\_backward** (T... args)
- T **wcstoll** (T... args)
- T **perror** (T... args)
- T **vwscanf** (T... args)
- T **stable\_sort** (T... args)
- T **generic\_category** (T... args)
- T **abs(int)** (T... args)
- T **fgetws** (T... args)
- T **showpos** (T... args)
- T **exp** (T... args)
- T **fill** (T... args)
- T **isalpha** (T... args)
- T **lgamma** (T... args)
- T **feclearexcept** (T... args)
- T **wcsncpy** (T... args)
- T **undecclare\_reachable** (T... args)
- T **oct** (T... args)
- T **strspn** (T... args)
- T **realloc** (T... args)
- T **copy** (T... args)
- T **binary\_search** (T... args)
- T **system\_category** (T... args)
- T **mbrtowc** (T... args)
- T **strtouf** (T... args)
- T **mem\_fn** (T... args)
- T **distance** (T... args)
- T **lock** (T... args)
- T **strcmp** (T... args)
- T **tmpfile** (T... args)
- T **hypot** (T... args)
- T **getenv** (T... args)
- T **strrchr** (T... args)
- T **count** (T... args)
- T **tan** (T... args)
- T **strftime** (T... args)
- T **stod** (T... args)
- T **towupper** (T... args)

- T **atoll** (T... args)
- T **atomic\_store** (T... args)
- T **stoi** (T... args)
- T **rethrow\_exception** (T... args)
- T **sin** (T... args)
- T **atomic\_fetch\_sub\_explicit** (T... args)
- T **unexpected** (T... args)
- T **mbtowc** (T... args)
- T **get\_time** (T... args)
- T **partition** (T... args)
- T **next** (T... args)
- T **isfinite** (T... args)
- T **boolalpha** (T... args)
- T **fetestexcept** (T... args)
- T **mbrlen** (T... args)
- T **iswgraph** (T... args)
- T **time** (T... args)
- T **atomic\_compare\_exchange\_strong** (T... args)
- T **wcschr** (T... args)
- T **uppercase** (T... args)
- T **lower\_bound** (T... args)
- T **copy\_if** (T... args)
- T **isnan** (T... args)
- T **has\_facet** (T... args)
- T **kill\_dependency** (T... args)
- T **uninitialized\_copy\_n** (T... args)
- T **feholdexcept** (T... args)
- T **div** (T... args)
- T **at\_quick\_exit** (T... args)
- T **wcspbrk** (T... args)
- T **search** (T... args)
- T **find\_first\_of** (T... args)
- T **iota** (T... args)
- T **declare\_reachable** (T... args)
- T **atomic\_compare\_exchange\_weak** (T... args)
- T **strtod** (T... args)
- T **accumulate** (T... args)
- T **wcsrchr** (T... args)
- T **min\_element** (T... args)
- T **clearerr** (T... args)
- T **random\_shuffle** (T... args)
- T **iswalpha** (T... args)
- T **atomic\_fetch\_and** (T... args)
- T **wmemchr** (T... args)
- T **bsearch** (T... args)
- T **ilogb** (T... args)
- T **unique\_copy** (T... args)
- T **\_Exit** (T... args)
- T **move** (T... args)
- T **find\_end** (T... args)
- T **fesetexceptflag** (T... args)
- T **nth\_element** (T... args)
- T **gets** (T... args)
- T **lexicographical\_compare** (T... args)
- T **nearbyint** (T... args)

- T **memcpy** (T... args)
- T **fwrite** (T... args)
- T **unitbuf** (T... args)
- T **iswlower** (T... args)
- T **mblen** (T... args)
- T **swscanf** (T... args)
- T **wcstoimax** (T... args)
- T **fprintf** (T... args)
- T **find\_if** (T... args)
- T **strtoimax** (T... args)
- T **isalnum** (T... args)
- T **atomic\_fetch\_add\_explicit** (T... args)
- T **push\_heap** (T... args)
- T **min** (T... args)
- T **fwprintf** (T... args)
- T **uncaught\_exception** (T... args)
- T **strtoll** (T... args)
- T **throw\_with\_nested** (T... args)
- T **shuffle** (T... args)
- T **isprint** (T... args)
- T **get\_new\_handler** (T... args)
- T **call\_once** (T... args)
- T **trunc** (T... args)
- T **wcscspn** (T... args)
- T **mbrtoc16** (T... args)
- T **lround** (T... args)
- T **pow** (T... args)
- T **tgamma** (T... args)
- T **erfc** (T... args)
- T **llround** (T... args)
- T **abs(float)** (T... args)
- T **asinh** (T... args)
- T **feof** (T... args)
- T **noskipws** (T... args)
- T **find** (T... args)
- T **atoi** (T... args)
- T **not1** (T... args)
- T **vscanf** (T... args)
- T **stof** (T... args)
- T **regex\_search** (T... args)
- T **rotate\_copy** (T... args)
- T **set\_new\_handler** (T... args)
- T **undecclare\_no\_pointers** (T... args)
- T **async** (T... args)
- T **partition\_point** (T... args)
- T **vsscanf** (T... args)
- T **fesetround** (T... args)
- T **atomic\_is\_lock\_free** (T... args)
- T **tanh** (T... args)
- T **ldiv** (T... args)
- T **setbase** (T... args)
- T **remove** (T... args)
- T **strtol** (T... args)
- T **strpbrk** (T... args)
- T **signbit** (T... args)

- T **wcsncat** (T... args)
- T **get\_money** (T... args)
- T **set\_difference** (T... args)
- T **cref** (T... args)
- T **getline** (T... args)
- T **to\_wstring** (T... args)
- T **system** (T... args)
- T **static\_pointer\_cast** (T... args)
- T **wcstoumax** (T... args)
- T **memmove** (T... args)
- T **getwchar** (T... args)
- T **scientific** (T... args)
- T **wcsftime** (T... args)
- T **begin** (T... args)
- T **ceil** (T... args)
- T **sinh** (T... args)
- T **is\_permutation** (T... args)
- T **generate\_n** (T... args)
- T **acosh** (T... args)
- T **advance** (T... args)
- T **flush** (T... args)
- T **atomic\_fetch\_xor** (T... args)
- T **ws** (T... args)
- T **signal** (T... args)
- T **noshowbase** (T... args)
- T **generate** (T... args)
- T **ldexp** (T... args)
- T **vsprintf** (T... args)
- T **remove\_if** (T... args)
- T **stoull** (T... args)
- T **fegetexceptflag** (T... args)
- T **find\_if\_not** (T... args)
- T **merge** (T... args)
- T **free** (T... args)
- T **count\_if** (T... args)
- T **clock** (T... args)
- T **mktime** (T... args)
- T **inserter** (T... args)
- T **puts** (T... args)
- T **asin** (T... args)
- T **isctrl** (T... args)
- T **difftime** (T... args)
- T **terminate** (T... args)
- T **memcmp** (T... args)
- T **uninitialized\_fill** (T... args)
- T **hex** (T... args)
- T **tie** (T... args)
- T **back\_inserter** (T... args)
- T **upper\_bound** (T... args)
- T **adjacent\_find** (T... args)
- T **use\_facet** (T... args)
- T **vfprintf** (T... args)
- T **atomic\_fetch\_add** (T... args)
- T **fsetpos** (T... args)
- T **malloc** (T... args)

- T **localtime** (T... args)
- T **wcscmp** (T... args)
- T **c32rtomb** (T... args)
- T **isupper** (T... args)
- T **wcstod** (T... args)
- T **tolower** (T... args)
- T **sort\_heap** (T... args)
- T **isdigit** (T... args)
- T **wcslen** (T... args)
- T **wmemcmp** (T... args)
- T **move\_if\_noexcept** (T... args)
- T **declval** (T... args)
- T **fpclassify** (T... args)
- T **iswupper** (T... args)
- T **rand** (T... args)
- T **atomic\_compare\_exchange\_weak\_explicit** (T... args)
- T **partial\_sort** (T... args)
- T **llrint** (T... args)
- T **fclose** (T... args)
- T **reverse** (T... args)
- T **partial\_sum** (T... args)
- T **showbase** (T... args)
- T **vswscanf** (T... args)
- T **atan** (T... args)
- T **atanh** (T... args)
- T **iter\_swap** (T... args)
- T **scalbln** (T... args)
- T **reverse\_copy** (T... args)
- T **forward** (T... args)
- T **getc** (T... args)
- T **equal\_range** (T... args)
- T **atomic\_fetch\_sub** (T... args)
- T **is\_partitioned** (T... args)
- T **next\_permutation** (T... args)
- T **isblank** (T... args)
- T **noshowpoint** (T... args)
- T **atan2** (T... args)
- T **nanf** (T... args)
- T **towctrans** (T... args)
- T **right** (T... args)
- T **fputwc** (T... args)
- T **strtoul** (T... args)
- T **is\_heap** (T... args)
- T **fflush** (T... args)
- T **strtoumax** (T... args)
- T **nexttoward** (T... args)
- T **nounitbuf** (T... args)
- T **ispunct** (T... args)
- T **noboolalpha** (T... args)
- T **make\_pair** (T... args)
- T **iswctype** (T... args)
- T **srand** (T... args)
- T **replace\_copy** (T... args)
- T **future\_category** (T... args)
- T **resetiosflags** (T... args)



- T **vprintf** (T... args)
- T **gmtime** (T... args)
- T **align** (T... args)
- T **tuple\_cat** (T... args)
- T **ends** (T... args)
- T **set\_terminate** (T... args)
- T **lrint** (T... args)
- T **none\_of** (T... args)
- T **wscanf** (T... args)
- T **fputc** (T... args)
- T **dec** (T... args)
- T **strcat** (T... args)
- T **raise** (T... args)
- T **wcsspn** (T... args)
- T **fabs** (T... args)
- T **wmemcpy** (T... args)
- T **copy\_n** (T... args)
- T **rethrow\_if\_nested** (T... args)
- T **setlocale** (T... args)
- T **addressof** (T... args)
- T **calloc** (T... args)
- T **strerror** (T... args)
- T **strcpy** (T... args)
- T **wcstoul** (T... args)
- T **c16rtomb** (T... args)
- T **generate\_canonical** (T... args)
- T **vfprintf** (T... args)
- T **notify\_all\_at\_thread\_exit** (T... args)
- T **rotate** (T... args)
- T **current\_exception** (T... args)
- T **strtok** (T... args)
- T **wscat** (T... args)
- T **strncpy** (T... args)
- T **tolower** (T... args)
- T **floor** (T... args)
- T **left** (T... args)
- T **ferror** (T... args)
- T **atomic\_load\_explicit** (T... args)
- T **swap** (T... args)
- T **acos** (T... args)
- T **wscoll** (T... args)
- T **sqrt** (T... args)
- T **mbsinit** (T... args)
- T **qsort** (T... args)
- T **stoll** (T... args)
- T **put\_money** (T... args)
- T **wcstoul** (T... args)
- T **wcstol** (T... args)
- T **atexit** (T... args)
- T **atomic\_fetch\_or** (T... args)
- T **rewind** (T... args)
- T **wcsxfrm** (T... args)
- T **round** (T... args)
- T **vwprintf** (T... args)
- T **all\_of** (T... args)

- T **replace** (T... args)
- T **remquo** (T... args)
- T **setbuf** (T... args)
- T **strncmp** (T... args)
- T **localeconv** (T... args)
- T **wctrans** (T... args)
- T **any\_of** (T... args)
- T **equal** (T... args)
- T **max** (T... args)
- T **strxfrm** (T... args)
- T **iswxdigit** (T... args)
- T **labs** (T... args)
- T **regex\_match** (T... args)
- T **fputws** (T... args)
- T **wcrtomb** (T... args)
- T **setprecision** (T... args)
- T **setvbuf** (T... args)
- T **regex\_replace** (T... args)
- T **freopen** (T... args)
- T **logb** (T... args)
- T **wctob** (T... args)
- T **atomic\_load** (T... args)
- T **search\_n** (T... args)
- T **toupper** (T... args)
- T **move\_backward** (T... args)
- T **is\_sorted** (T... args)
- T **strtoull** (T... args)
- T **iswblank** (T... args)
- T **get\_pointer\_safety** (T... args)
- T **get\_unexpected** (T... args)
- T **sscanf** (T... args)
- T **fesetenv** (T... args)
- T **atomic\_store\_explicit** (T... args)
- T **strtold** (T... args)
- T **fread** (T... args)
- T **memchr** (T... args)
- T **btowc** (T... args)
- T **replace\_if** (T... args)
- T **strcoll** (T... args)
- T **vsprintf** (T... args)
- T **mismatch** (T... args)
- T **getchar** (T... args)
- T **islower** (T... args)
- T **tmpnam** (T... args)
- T **nanl** (T... args)
- T **fopen** (T... args)
- T **for\_each** (T... args)
- T **fegetround** (T... args)
- T **ungetc** (T... args)
- T **internal** (T... args)
- T **vfwscanf** (T... args)
- T **fgetc** (T... args)
- T **wcstof** (T... args)
- T **bind** (T... args)
- T **skipws** (T... args)

- T **iswprint** (T... args)
- T **wcstombs** (T... args)
- T **inplace\_merge** (T... args)
- T **copysign** (T... args)
- T **putwchar** (T... args)
- T **wcsstr** (T... args)
- T **fegetenv** (T... args)
- T **longjmp** (T... args)
- T **iswcntrl** (T... args)
- T **declare\_no\_pointers** (T... args)
- T **isnormal** (T... args)
- T **swap\_ranges** (T... args)
- T **minmax** (T... args)
- T **defaultfloat** (T... args)
- T **rename** (T... args)
- T **snprintf** (T... args)
- T **try\_lock** (T... args)
- T **stoul** (T... args)
- T **fgetpos** (T... args)
- T **partition\_copy** (T... args)
- T **vscanf** (T... args)
- T **front\_inserter** (T... args)
- T **get\_terminate** (T... args)
- T **cosh** (T... args)
- T **prev** (T... args)
- T **strchr** (T... args)
- T **strstr** (T... args)
- T **printf** (T... args)
- T **setfill** (T... args)
- T **inner\_product** (T... args)
- template<typename T, typename... CONSTRUCTOR\_ARGS>  
**std::unique\_ptr**< T > [make\\_unique](#) (CONSTRUCTOR\_ARGS &&... constructor\_args)

## 9.2.1 Function Documentation

### 9.2.1.1 make\_unique()

```
template<typename T, typename... CONSTRUCTOR_ARGS>
std::unique_ptr<T> std::make_unique (
    CONSTRUCTOR_ARGS &&... constructor_args )
```

Definition at line 192 of file utils.h.

## 9.3 vxg Namespace Reference

### Namespaces

- [cloud](#)
- [media](#)

## Data Structures

- class [logger](#)

*Logger class, current implementation based on spdlog.*

## 9.4 vxg::cloud Namespace Reference

### Namespaces

- [agent](#)

*VXG Cloud Agent namespace.*

- [time\\_spec](#)

*time point*

- [utils](#)

### Typedefs

- using [time](#) = `std::chrono::time_point< std::chrono::system_clock, time\_spec::precision >`
- using [duration](#) = `time_spec::duration< time\_spec::precision >`

### 9.4.1 Typedef Documentation

#### 9.4.1.1 duration

```
typedef time\_spec::duration< time\_spec::precision > vxg::cloud::duration
```

Definition at line 43 of file config.h.

#### 9.4.1.2 time

```
typedef std::chrono::time_point< std::chrono::system_clock, time\_spec::precision > vxg::cloud::time
```

Definition at line 42 of file config.h.

## 9.5 vxg::cloud::agent Namespace Reference

VXG Cloud Agent namespace.

## Namespaces

- [media](#)
- [proto](#)

## Data Structures

- struct [access\\_token](#)  
*VXG Cloud access token.*
- struct [audio\\_config](#)  
*Audio config.*
- class [callback](#)  
*VXG Cloud manager common callbacks class.*
- struct [event\\_config](#)  
*Event config.*
- class [event\\_stream](#)  
*Event stream, abstract class for event generation.*
- struct [events\\_config](#)  
*Events config, list of [event\\_config](#) objects.*
- class [manager](#)  
*VXG Cloud agent manager class.*
- struct [osd\\_config](#)  
*OSD config.*
- struct [ptz\\_command](#)  
*PTZ command.*
- struct [ptz\\_config](#)  
*PTZ config.*
- struct [ptz\\_preset](#)  
*PTZ preset.*
- struct [supported\\_stream\\_config](#)  
*Supported stream config.*
- struct [supported\\_streams\\_config](#)  
*Supported streams config, list of [supported\\_stream\\_config](#).*

## Functions

- `std::string version ()`  
*VXG Cloud Agent library version.*

### 9.5.1 Detailed Description

VXG Cloud Agent namespace.

### 9.5.2 Function Documentation

### 9.5.2.1 version()

```
std::string vxg::cloud::agent::version ( )
```

VXG Cloud Agent library version.

Returns

**std::string** version string

## 9.6 vxg::cloud::agent::media Namespace Reference

### Data Structures

- class [rtsp\\_stream](#)  
*Implementation of the [media::stream](#) with RTSP source and NIY stubs.*
- class [stream](#)  
*Cloud agent media stream abstract class.*

## 9.7 vxg::cloud::agent::proto Namespace Reference

### Data Structures

- struct [audio\\_caps](#)  
*Audio capabilities.*
- struct [audio\\_stream\\_config](#)  
*Audio media stream config.*
- struct [event\\_caps](#)  
*Events capabilities.*
- struct [motion\\_detection\\_caps](#)  
*Motion detection capabilities camera capabilities that limit possible motion detection configuration.*
- struct [motion\\_detection\\_config](#)  
*Motion detection config.*
- struct [motion\\_region](#)  
*Motion detection related structs.*
- struct [osd\\_caps](#)  
*OSD capabilities.*
- struct [stream\\_caps](#)  
*Media stream capabilities.*
- struct [stream\\_config](#)  
*Media stream config.*
- struct [video\\_caps](#)  
*Video image capabilities.*
- struct [video\\_clip\\_info](#)  
*Video recoding(mp4 file) clip description,.*
- struct [video\\_config](#)  
*Video image config.*
- struct [video\\_stream\\_config](#)  
*Video stream config.*
- struct [wifi\\_config](#)  
*WiFi config.*
- struct [wifi\\_network](#)  
*WiFi network object.*

## Typedefs

- typedef [wifi\\_config](#) [wifi\\_list](#)  
*wifi\_config*

## Enumerations

- enum [mode](#) { [M\\_OFF](#), [M\\_ON](#), [M\\_AUTO](#), [M\\_INVALID](#) }  
*Mode on/off.*
- enum [video\\_format](#) { [VF\\_H264](#), [VF\\_H265](#), [VF\\_MJPEG](#), [VF\\_INVALID](#) }  
*Video codec format.*
- enum [audio\\_format](#) {  
[AF\\_G711A](#), [AF\\_G711U](#), [AF\\_RAW](#), [AF\\_ADPCM](#),  
[AF\\_MP3](#), [AF\\_NELLY8](#), [AF\\_NELLY16](#), [AF\\_NELLY](#),  
[AF\\_OPUS](#), [AF\\_AAC](#), [AF\\_SPEEX](#), [AF\\_INVALID](#) }  
*Audio codec format.*
- enum [audio\\_file\\_format](#) { [AFF\\_AU\\_G711U](#), [AFF\\_MP3](#), [AFF\\_WAV\\_PCM](#), [AFF\\_INVALID](#) }  
*Audio file format.*
- enum [motion\\_sensitivity](#) { [MS\\_REGION](#), [MS\\_FRAME](#), [MS\\_INVALID](#) }  
*Motion sensitivity.*
- enum [motion\\_region\\_shape](#) { [MR\\_RECTANGLE](#), [MR\\_ANY](#), [MR\\_INVALID](#) }  
*Motion region shape.*
- enum [ptz\\_action](#) {  
[A\\_LEFT](#), [A\\_RIGHT](#), [A\\_TOP](#), [A\\_BOTTOM](#),  
[A\\_ZOOM\\_IN](#), [A\\_ZOOM\\_OUT](#), [A\\_STOP](#), [A\\_INVALID](#) }  
*PTZ actions.*
- enum [ptz\\_preset\\_action](#) {  
[PA\\_CREATE](#), [PA\\_DELETE](#), [PA\\_GOTO](#), [PA\\_UPDATE](#),  
[PA\\_INVALID](#) }  
*PTZ preset action.*
- enum [time\\_format\\_n](#) { [TF\\_12H](#), [TF\\_24H](#), [TF\\_INVALID](#) }  
*3.34 get\_osd\_conf (SRV) 3.35 osd\_conf (CM) 3.36 set\_osd\_conf (SRV)*
- enum [event\\_status](#) { [ES\\_OK](#), [ES\\_ERROR](#), [ES\\_INVALID](#) }  
*Event status.*
- enum [event\\_type](#) {  
[ET\\_MOTION](#), [ET\\_SOUND](#), [ET\\_NET](#), [ET\\_RECORD](#),  
[ET\\_MEMORYCARD](#), [ET\\_WIFI](#), [ET\\_CUSTOM](#), [ET\\_INVALID](#) }  
*Types of events.*
- enum [memorycard\\_status](#) {  
[MCS\\_NONE](#), [MCS\\_NORMAL](#), [MCS\\_NEED\\_FORMAT](#), [MCS\\_FORMATTING](#),  
[MCS\\_INITIALIZATION](#), [MCS\\_INVALID](#) }  
*Memory card status.*
- enum [wifi\\_encryption](#) {  
[WFE\\_OPEN](#), [WFE\\_WEP](#), [WFE\\_WPA](#), [WFE\\_WPA2](#),  
[WFE\\_WPA\\_ENTERPRISE](#), [WFE\\_WPA2\\_ENTERPRISE](#), [WFE\\_INVALID](#) }  
*WiFi encryption type.*
- enum [wifi\\_network\\_state](#) {  
[WNS\\_UNKNOWN](#), [WNS\\_INITIALIZE\\_0](#), [WNS\\_INITIALIZE\\_1](#), [WNS\\_TRY\\_CONNECT](#),  
[WNS\\_RECEIVING\\_IP](#), [WNS\\_CONNECTED](#), [WNS\\_INVALID](#) }  
*WiFi connection state.*

## Functions

- `std::string name ()`

## 9.7.1 Typedef Documentation

### 9.7.1.1 wifi\_list

```
typedef wifi\_config vxg::cloud::agent::proto::wifi_list
```

[wifi\\_config](#)

Definition at line 597 of file config.h.

## 9.7.2 Enumeration Type Documentation

### 9.7.2.1 audio\_file\_format

```
enum vxg::cloud::agent::proto::audio\_file\_format
```

Audio file format.

Enumerator

<code>AFF_AU_G711U</code>	AU file format, encoded in mu-law and sampled with 8 or 16 kHz;.
<code>AFF_MP3</code>	MP3 file format, in mono or stereo with bitrate of 64 kbps to 320 kbps and sample rate of 8 to 48 kHz.
<code>AFF_WAV_PCM</code>	WAV file format, encoded in PCM audio that depends on what the product supports. It may support encoded as 8 or 16-bit mono or stereo and sample rate of 8 to 48 kHz;
<code>AFF_INVALID</code>	Invalid value.

Definition at line 147 of file caps.h.

### 9.7.2.2 audio\_format

```
enum vxg::cloud::agent::proto::audio\_format
```

Audio codec format.



## Enumerator

AF_G711A	G711A - PCMA, A-Law.
AF_G711U	G711U - PCMU, U-Law.
AF_RAW	PCM.
AF_ADPCM	G726LE.
AF_MP3	
AF_NELLY8	
AF_NELLY16	
AF_NELLY	
AF_OPUS	
AF_AAC	AAC.
AF_SPEEX	
AF_INVALID	Invalid value.

Definition at line 106 of file caps.h.

### 9.7.2.3 event\_status

```
enum vxg::cloud::agent::proto::event_status
```

Event status.

## Enumerator

ES_OK	Ok.
ES_ERROR	Error.
ES_INVALID	Default status, invalid.

Definition at line 381 of file config.h.

### 9.7.2.4 event\_type

```
enum vxg::cloud::agent::proto::event_type
```

Types of events.

## Enumerator

ET_MOTION	"motion" for motion detection events
ET_SOUND	"sound" for audio detection
ET_NET	"net" for the camera network status change
ET_RECORD	"record" CM informs server about necessity of changing of recording state
ET_MEMORYCARD	"memorycard" camera's memory-card status change
ET_WIFI	"wifi" status of camera's currently used Wi-Fi
ET_CUSTOM	Custom event.
ET_INVALID	Invalid event type.

Definition at line 404 of file config.h.

### 9.7.2.5 memorycard\_status

```
enum vxg::cloud::agent::proto::memorycard_status
```

Memory card status.

Enumerator

MCS_NONE	No memorycard.
MCS_NORMAL	Memorycard is OK.
MCS_NEED_FORMAT	Need formatting.
MCS_FORMATTING	Formatting ongoing.
MCS_INITIALIZATION	Initialization, not mounted yet for example.
MCS_INVALID	Invalid value.

Definition at line 484 of file config.h.

### 9.7.2.6 mode

```
enum vxg::cloud::agent::proto::mode
```

Mode on/off.

Enumerator

M_OFF	
M_ON	
M_AUTO	
M_INVALID	

Definition at line 30 of file caps.h.

### 9.7.2.7 motion\_region\_shape

```
enum vxg::cloud::agent::proto::motion_region_shape
```

Motion region shape.

Enumerator

MR_RECTANGLE	Rectangle.
MR_ANY	Any shape.
MR_INVALID	Invalid.

Definition at line 313 of file caps.h.

### 9.7.2.8 motion\_sensitivity

```
enum vxg::cloud::agent::proto::motion_sensitivity
```

Motion sensitivity.

#### Enumerator

MS_REGION	Indicates if sensitivity can be set for region.
MS_FRAME	Indicates if sensitivity can be only for the full frame.
MS_INVALID	Invalid value.

Definition at line 291 of file caps.h.

### 9.7.2.9 ptz\_action

```
enum vxg::cloud::agent::proto::ptz_action
```

PTZ actions.

#### Enumerator

A_LEFT	Go left.
A_RIGHT	Go right.
A_TOP	Go tip.
A_BOTTOM	Go bottom.
A_ZOOM_IN	Zoom in.
A_ZOOM_OUT	Zoom out.
A_STOP	Stop current action.
A_INVALID	Invalid value.

Definition at line 527 of file caps.h.

### 9.7.2.10 ptz\_preset\_action

```
enum vxg::cloud::agent::proto::ptz_preset_action
```

PTZ preset action.

**Enumerator**

PA_CREATE	
PA_DELETE	
PA_GOTO	
PA_UPDATE	
PA_INVALID	

Definition at line 563 of file caps.h.

**9.7.2.11 time\_format\_n**

```
enum vsg::cloud::agent::proto::time_format_n
```

3.34 get\_osd\_conf (SRV) 3.35 osd\_conf (CM) 3.36 set\_osd\_conf (SRV)

Time format

**Enumerator**

TF_12H	12 hours
TF_24H	24 hours
TF_INVALID	Invalid value.

Definition at line 592 of file caps.h.

**9.7.2.12 video\_format**

```
enum vsg::cloud::agent::proto::video_format
```

Video codec format.

**Enumerator**

VF_H264	H264 (AVC)
VF_H265	H265 (HEVC)
VF_MJPEG	Motion JPEG.
VF_INVALID	Invalid value.

Definition at line 81 of file caps.h.

**9.7.2.13 wifi\_encryption**

```
enum vsg::cloud::agent::proto::wifi_encryption
```

WiFi encryption type.

#### Enumerator

WFE_OPEN	No encryption.
WFE_WEP	WEP.
WFE_WPA	WPA-PSK.
WFE_WPA2	WPA2-PSK.
WFE_WPA_ENTERPRISE	WPA-Enterprise.
WFE_WPA2_ENTERPRISE	WPA2-Enterprise.
WFE_INVALID	Default, invalid value.

Definition at line 520 of file config.h.

#### 9.7.2.14 wifi\_network\_state

```
enum vxg::cloud::agent::proto::wifi_network_state
```

WiFi connection state.

#### Enumerator

WNS_UNKNOWN	
WNS_INITIALIZE_0	
WNS_INITIALIZE_1	
WNS_TRY_CONNECT	
WNS_RECEIVING_IP	
WNS_CONNECTED	
WNS_INVALID	Invalid value.

Definition at line 600 of file config.h.

### 9.7.3 Function Documentation

#### 9.7.3.1 name()

```
std::string vxg::cloud::agent::proto::name ( )
```

Definition at line 887 of file config.h.

## 9.8 vxg::cloud::time\_spec Namespace Reference

time point

## Typedefs

- using [precision](#) = `std::chrono::nanoseconds`
- template<typename T >  
using [duration](#) = typename `std::conditional< std::is_same< T, precision >::value, precision, std::`  
`::chrono::duration< T > >::type`

### 9.8.1 Detailed Description

time point

### 9.8.2 Typedef Documentation

#### 9.8.2.1 duration

```
template<typename T >
using vxg::cloud::time\_spec::duration = typedef typename std::conditional< std::is_same<T,  
precision>::value, precision, std::chrono::duration<T> >::type
```

Definition at line 39 of file config.h.

#### 9.8.2.2 precision

```
typedef std::chrono::nanoseconds vxg::cloud::time\_spec::precision
```

Definition at line 35 of file config.h.

## 9.9 vxg::cloud::utils Namespace Reference

### Namespaces

- [gcc\\_abi](#)
- [motion](#)
- [time](#)

### Data Structures

- struct [uri](#)

## Functions

- void [set\\_thread\\_name](#) ( **std::string** name)
- template<typename... Args>  
**std::string** [string\\_format](#) (const **std::string** &format, Args... args)
- **std::string** [string\\_trim](#) (const **std::string** &name, **std::regex** regx)
- **std::string** [string\\_trim](#) (const **std::string** &name)
- **std::vector**< **std::string** > [string\\_split](#) (const **std::string** &s, char delimiter)
- bool [string\\_startswith](#) ( **std::string** const &fullString, **std::string** const &start)
- bool [string\\_endswith](#) ( **std::string** const &fullString, **std::string** const &ending)
- bool [string\\_replace](#) ( **std::string** &str, const **std::string** &from, const **std::string** &to)
- **std::string** [string\\_urlencode](#) (const **std::string** &value)
- **std::string** [string\\_urldecode](#) (const **std::string** &text)
- **std::string** [string\\_tolower](#) (const **std::string** &s)
- **std::string** [string\\_toupper](#) (const **std::string** &s)
- bool [string\\_contains](#) ( **std::string** s, char c)
- **std::string** [dirname](#) (const **std::string** &filepath)

### 9.9.1 Function Documentation

#### 9.9.1.1 [dirname\(\)](#)

```
std::string vxg::cloud::utils::dirname (
    const std::string & filepath )
```

#### 9.9.1.2 [set\\_thread\\_name\(\)](#)

```
void vxg::cloud::utils::set_thread_name (
    std::string name )
```

#### 9.9.1.3 [string\\_contains\(\)](#)

```
bool vxg::cloud::utils::string_contains (
    std::string s,
    char c ) [inline]
```

Definition at line 172 of file `utils.h`.

#### 9.9.1.4 string\_endswith()

```
bool vxg::cloud::utils::string_endswith (
    std::string const & fullString,
    std::string const & ending )
```

#### 9.9.1.5 string\_format()

```
template<typename... Args>
std::string vxg::cloud::utils::string_format (
    const std::string & format,
    Args... args )
```

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

#### 9.9.1.6 string\_replace()

```
bool vxg::cloud::utils::string_replace (
    std::string & str,
    const std::string & from,
    const std::string & to )
```

#### 9.9.1.7 string\_split()

```
std::vector< std::string> vxg::cloud::utils::string_split (
    const std::string & s,
    char delimiter )
```

#### 9.9.1.8 string\_startswith()

```
bool vxg::cloud::utils::string_startswith (
    std::string const & fullString,
    std::string const & start )
```

#### 9.9.1.9 string\_tolower()

```
std::string vxg::cloud::utils::string_tolower (
    const std::string & s )
```



#### 9.9.1.10 string\_toupper()

```
std::string vxg::cloud::utils::string_toupper (
    const std::string & s )
```

#### 9.9.1.11 string\_trim() [1/2]

```
std::string vxg::cloud::utils::string_trim (
    const std::string & name )
```

#### 9.9.1.12 string\_trim() [2/2]

```
std::string vxg::cloud::utils::string_trim (
    const std::string & name,
    std::regex regx )
```

#### 9.9.1.13 string\_urldecode()

```
std::string vxg::cloud::utils::string_urldecode (
    const std::string & text )
```

#### 9.9.1.14 string\_urlencode()

```
std::string vxg::cloud::utils::string_urlencode (
    const std::string & value )
```

## 9.10 vxg::cloud::utils::gcc\_abi Namespace Reference

### Functions

- std::string [demangle](#) ( std::string name)

#### 9.10.1 Function Documentation

### 9.10.1.1 demangle()

```
std::string vxg::cloud::utils::gcc_abi::demangle (
    std::string name )
```

## 9.11 vxg::cloud::utils::motion Namespace Reference

### Data Structures

- struct [map](#)

## 9.12 vxg::cloud::utils::time Namespace Reference

### Functions

- [cloud::time now](#) ()
- [std::string time\\_to\\_ISO8601](#) ( [std::time\\_t](#) )
- [std::string time\\_to\\_ISO8601\\_packed](#) ( [std::time\\_t](#) )
- [std::string now\\_ISO8601\\_UTC](#) ()
- [std::string now\\_ISO8601\\_UTC\\_packed](#) ()
- [std::time\\_t now\\_time\\_UTC](#) ()
- [std::time\\_t ISO8601\\_to\\_time](#) (const [std::string](#) &input)
- [std::string to\\_iso\\_8601](#) ([cloud::time t](#))
- [std::string to\\_iso](#) ([cloud::time t](#))
- [std::string to\\_iso2](#) ([cloud::time t](#))
- [std::string to\\_iso\\_packed](#) ([cloud::time t](#))
- [std::string to\\_iso\\_local](#) ([cloud::time t](#))
- [cloud::time from\\_double](#) (double t)
- double [to\\_double](#) ([cloud::time t](#))
- [cloud::time from\\_iso](#) ( [std::string st](#) )
- [cloud::time from\\_iso2](#) ( [std::string st](#) )
- [cloud::time from\\_iso\\_packed](#) ( [std::string st](#) )
- bool [iso\\_time\\_valid](#) (const [std::string](#) &s)
- [cloud::time null](#) ()
- [cloud::time max](#) ()
- bool [is\\_iso\\_packed](#) (const [std::string](#) &s)
- bool [is\\_iso](#) (const [std::string](#) &s)

### 9.12.1 Function Documentation

#### 9.12.1.1 from\_double()

```
cloud::time vxg::cloud::utils::time::from_double (
    double t )
```

### 9.12.1.2 from\_iso()

```
cloud::time vxg::cloud::utils::time::from_iso (
    std::string st )
```

### 9.12.1.3 from\_iso2()

```
cloud::time vxg::cloud::utils::time::from_iso2 (
    std::string st )
```

### 9.12.1.4 from\_iso\_packed()

```
cloud::time vxg::cloud::utils::time::from_iso_packed (
    std::string st )
```

### 9.12.1.5 is\_iso()

```
bool vxg::cloud::utils::time::is_iso (
    const std::string & s )
```

### 9.12.1.6 is\_iso\_packed()

```
bool vxg::cloud::utils::time::is_iso_packed (
    const std::string & s )
```

### 9.12.1.7 ISO8601\_to\_time()

```
std::time_t vxg::cloud::utils::time::ISO8601_to_time (
    const std::string & input )
```

### 9.12.1.8 iso\_time\_valid()

```
bool vxg::cloud::utils::time::iso_time_valid (
    const std::string & s )
```

#### 9.12.1.9 max()

```
cloud::time vxg::cloud::utils::time::max ( ) [inline]
```

Definition at line 57 of file utils.h.

#### 9.12.1.10 now()

```
cloud::time vxg::cloud::utils::time::now ( ) [inline]
```

Definition at line 30 of file utils.h.

#### 9.12.1.11 now\_ISO8601.UTC()

```
std::string vxg::cloud::utils::time::now_ISO8601.UTC ( )
```

#### 9.12.1.12 now\_ISO8601.UTC\_packed()

```
std::string vxg::cloud::utils::time::now_ISO8601.UTC_packed ( )
```

#### 9.12.1.13 now\_time.UTC()

```
std::time_t vxg::cloud::utils::time::now_time.UTC ( )
```

#### 9.12.1.14 null()

```
cloud::time vxg::cloud::utils::time::null ( ) [inline]
```

Definition at line 53 of file utils.h.

#### 9.12.1.15 time\_to\_ISO8601()

```
std::string vxg::cloud::utils::time::time_to_ISO8601 (
    std::time_t )
```

#### 9.12.1.16 time\_to\_ISO8601\_packed()

```
std::string vxg::cloud::utils::time::time_to_ISO8601_packed (
    std::time_t )
```

#### 9.12.1.17 to\_double()

```
double vxg::cloud::utils::time::to_double (
    cloud::time t )
```

#### 9.12.1.18 to\_iso()

```
std::string vxg::cloud::utils::time::to_iso (
    cloud::time t )
```

#### 9.12.1.19 to\_iso2()

```
std::string vxg::cloud::utils::time::to_iso2 (
    cloud::time t )
```

#### 9.12.1.20 to\_iso\_8601()

```
std::string vxg::cloud::utils::time::to_iso_8601 (
    cloud::time t )
```

#### 9.12.1.21 to\_iso\_local()

```
std::string vxg::cloud::utils::time::to_iso_local (
    cloud::time t )
```

#### 9.12.1.22 to\_iso\_packed()

```
std::string vxg::cloud::utils::time::to_iso_packed (
    cloud::time t )
```

## 9.13 vxg::media Namespace Reference

### Namespaces

- [ffmpeg](#)
- [Streamer](#)

### Data Structures

- class [rtmp\\_sink](#)  
*RTMP sink class.*
- class [rtmp\\_source](#)  
*RTMP source class.*
- class [rtsp\\_source](#)  
*RTSP source class.*
- class [stream](#)  
*base media stream abstract class*

## 9.14 vxg::media::ffmpeg Namespace Reference

### Data Structures

- class [Sink](#)  
*Base ffmpeg sink class.*
- class [Source](#)  
*Base ffmpeg source class.*

## 9.15 vxg::media::Streamer Namespace Reference

### Data Structures

- class [ISink](#)
- class [ISource](#)  
*ISource interface class.*
- struct [MediaFrame](#)  
*Media frame container.*
- struct [StreamInfo](#)  
*Stream info description.*

### Enumerations

- enum [DropDirection](#) { [DROP\\_FRONT](#), [DROP\\_BACK](#) }
- enum [StreamError](#) { [E\\_NONE](#), [E\\_FATAL](#), [E\\_EOS](#) }  
*Stream error.*
- enum [MediaType](#) {  
[UNKNOWN](#), [VIDEO](#), [VIDEO\\_AVC\\_SPS](#), [VIDEO\\_AVC\\_PPS](#),  
[VIDEO\\_SEQ\\_HDR](#), [AUDIO](#), [AUDIO\\_SEQ\\_HDR](#), [FLV](#),  
[DATA](#), [MAX](#) }  
*Media frame type.*

## Variables

- constexpr int [SINK\\_THREAD\\_PRIO](#)
- constexpr int [SRC\\_THREAD\\_PRIO](#)

## 9.15.1 Enumeration Type Documentation

### 9.15.1.1 DropDirection

```
enum vxg::media::Streamer::DropDirection
```

Enumerator

DROP_FRONT	
DROP_BACK	

Definition at line 27 of file base\_streamer.h.

### 9.15.1.2 MediaType

```
enum vxg::media::Streamer::MediaType
```

Media frame type.

Used to indicate when type of frame was passed from source to sink.

Enumerator

UNKNOWN	
VIDEO	
VIDEO_AVC_SPS	
VIDEO_AVC_PPS	
VIDEO_SEQ_HDR	
AUDIO	
AUDIO_SEQ_HDR	
FLV	
DATA	
MAX	

Definition at line 389 of file base\_streamer.h.

### 9.15.1.3 StreamError

```
enum vxg::media::Streamer::StreamError
```

Stream error.

Enumerator

E_NONE	
E_FATAL	
E_EOS	

Definition at line 33 of file base\_streamer.h.

## 9.15.2 Variable Documentation

### 9.15.2.1 SINK\_THREAD\_PRIO

```
constexpr int vxg::media::Streamer::SINK_THREAD_PRIO [constexpr]
```

Definition at line 25 of file base\_streamer.h.

### 9.15.2.2 SRC\_THREAD\_PRIO

```
constexpr int vxg::media::Streamer::SRC_THREAD_PRIO [constexpr]
```

Definition at line 26 of file base\_streamer.h.



## Chapter 10

# Data Structure Documentation

### 10.1 vxg::cloud::agent::access\_token Struct Reference

VXG Cloud access token.

```
#include <agent-proto/objects/config.h>
```

#### Data Structures

- struct [proxy\\_config](#)  
*Socks proxy settings.*

#### Public Types

- typedef **std::shared\_ptr**< [access\\_token](#) > [ptr](#)

#### Public Member Functions

- **std::string** [api\\_uri](#) (bool secure=true)
- **std::string** [pack](#) ()

#### Static Public Member Functions

- static [access\\_token::ptr](#) [parse](#) ( **std::string** packed\_token)

#### 10.1.1 Detailed Description

VXG Cloud access token.

Definition at line 1192 of file config.h.

## 10.1.2 Member Typedef Documentation

### 10.1.2.1 ptr

```
typedef std::shared_ptr<access_token> vxg::cloud::agent::access_token::ptr
```

Definition at line 1193 of file config.h.

## 10.1.3 Member Function Documentation

### 10.1.3.1 api\_uri()

```
std::string vxg::cloud::agent::access_token::api_uri (
    bool secure = true ) [inline]
```

Definition at line 1242 of file config.h.

### 10.1.3.2 pack()

```
std::string vxg::cloud::agent::access_token::pack ( ) [inline]
```

Definition at line 1250 of file config.h.

### 10.1.3.3 parse()

```
static access_token::ptr vxg::cloud::agent::access_token::parse (
    std::string packed_token ) [inline], [static]
```

Definition at line 1252 of file config.h.

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

- [config.h](#)

## 10.2 alter\_bool Struct Reference

alternative bool class Standard bool type has two states, this class adds 3rd state - undefined.

```
#include <agent-proto/command/unset-helper.h>
```

## Public Types

- enum `n_alter_bool` { `B_FALSE`, `B_TRUE`, `B_INVALID` }  
*Internal boolean values.*

## Public Member Functions

- `alter_bool` (const `n_alter_bool` &*v*)
- `alter_bool` (const bool &*v*)
- `alter_bool operator=` (const bool &*b*)
- `operator bool` () const

## Data Fields

- `n_alter_bool` *val*

## Friends

- void `from_json` (const `nlohmann::json` &*j*, `alter_bool` &*c*)
- void `to_json` (`nlohmann::json` &*j*, const `alter_bool` &*c*)

### 10.2.1 Detailed Description

alternative bool class Standard bool type has two states, this class adds 3rd state - undefined.

This class used for json boolean => C++ bool type reflection. The `B_INVALID` value of the C++ data indicates that source json has no such field.

Definition at line 168 of file unset-helper.h.

### 10.2.2 Member Enumeration Documentation

#### 10.2.2.1 `n_alter_bool`

```
enum alter_bool::n_alter_bool
```

Internal boolean values.

#### Enumerator

<code>B_FALSE</code>	false
<code>B_TRUE</code>	true
<code>B_INVALID</code>	Undefined, i.e. if the object was constructed from the json object this value means that original json had no such field.

Definition at line 170 of file unset-helper.h.

## 10.2.3 Constructor & Destructor Documentation

### 10.2.3.1 `alter_bool()` [1/2]

```
alter_bool::alter_bool (
    const n_alter_bool & v ) [inline]
```

Definition at line 180 of file unset-helper.h.

### 10.2.3.2 `alter_bool()` [2/2]

```
alter_bool::alter_bool (
    const bool & v ) [inline]
```

Definition at line 182 of file unset-helper.h.

## 10.2.4 Member Function Documentation

### 10.2.4.1 `operator bool()`

```
alter_bool::operator bool ( ) const [inline]
```

Definition at line 196 of file unset-helper.h.

### 10.2.4.2 `operator=()`

```
alter_bool alter_bool::operator= (
    const bool & b ) [inline]
```

Definition at line 189 of file unset-helper.h.

## 10.2.5 Friends And Related Function Documentation

### 10.2.5.1 from\_json

```
void from_json (
    const nlohmann::json & j,
    alter_bool & c ) [friend]
```

Definition at line 202 of file unset-helper.h.

### 10.2.5.2 to\_json

```
void to_json (
    nlohmann::json & j,
    const alter_bool & c ) [friend]
```

Definition at line 209 of file unset-helper.h.

## 10.2.6 Field Documentation

### 10.2.6.1 val

```
n_alter_bool alter_bool::val
```

Definition at line 216 of file unset-helper.h.

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

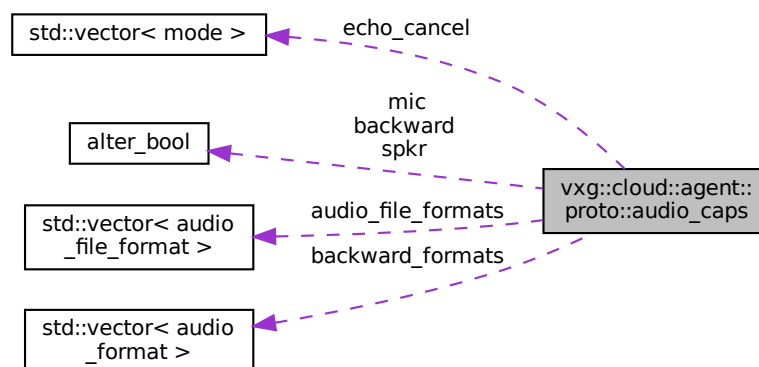
- [unset-helper.h](#)

## 10.3 vxg::cloud::agent::proto::audio\_caps Struct Reference

Audio capabilities.

```
#include <agent-proto/objects/caps.h>
```

Collaboration diagram for vxg::cloud::agent::proto::audio\_caps:



## Data Fields

- [alter\\_bool mic](#)  
*mic: bool, microphone is supported*
- [alter\\_bool spkr](#)  
*spkr: bool, speaker is supported*
- **std::vector**< [mode](#) > [echo\\_cancel](#)  
*echo\_cancel: list of string, echo cancellation modes, empty or absent means not supported*
- [alter\\_bool backward](#)  
*backward: bool, backward audio supported.*
- **std::vector**< [audio\\_format](#) > [backward\\_formats](#)  
*backward\_formats: list of audio\_format, list of supported backward formats.*
- **std::vector**< [audio\\_file\\_format](#) > [audio\\_file\\_formats](#)  
*audio\_file\_formats: list of string, list of supported formats of audio files.*

### 10.3.1 Detailed Description

Audio capabilities.

Definition at line 484 of file caps.h.

### 10.3.2 Field Documentation

#### 10.3.2.1 audio\_file\_formats

```
std::vector<audio\_file\_format> vxg::cloud::agent::proto::audio_caps::audio_file_formats
```

[audio\\_file\\_formats](#): list of string, list of supported formats of audio files.

Definition at line 507 of file caps.h.

#### 10.3.2.2 backward

```
alter\_bool vxg::cloud::agent::proto::audio_caps::backward
```

[backward](#): bool, backward audio supported.

Obsolete. Server will ignore it when [backward\\_formats](#) exists. If true and [backward\\_formats](#) is missed, server will interpret supported formats list as ["UNKNOWN"]

Definition at line 497 of file caps.h.

### 10.3.2.3 backward\_formats

```
std::vector<audio_format> vxg::cloud::agent::proto::audio_caps::backward_formats
```

backward\_formats: list of audio\_format, list of supported backward formats.

Supported values: ["RAW", "ADPCM", "MP3", "NELLY8", "NELLY16", "NELLY", "G711A", "G711U", "AAC", "SPEEX", "UNKNOWN"]. Empty list or missing parameter – camera doesn't support back audio channel.

Definition at line 503 of file caps.h.

### 10.3.2.4 echo\_cancel

```
std::vector<mode> vxg::cloud::agent::proto::audio_caps::echo_cancel
```

echo\_cancel: list of string, echo cancellation modes, empty or absent means not supported

Definition at line 492 of file caps.h.

### 10.3.2.5 mic

```
alter_bool vxg::cloud::agent::proto::audio_caps::mic
```

mic: bool, microphone is supported

Definition at line 486 of file caps.h.

### 10.3.2.6 spkr

```
alter_bool vxg::cloud::agent::proto::audio_caps::spkr
```

spkr: bool, speaker is supported

Definition at line 489 of file caps.h.

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

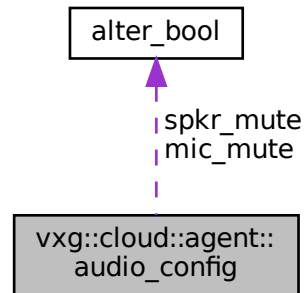
- [caps.h](#)

## 10.4 vxg::cloud::agent::audio\_config Struct Reference

Audio config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::audio\_config:



### Data Fields

- int [mic\\_gain](#)  
*mic\_gain: optional int range 0-100, microphone gain*
- [alter\\_bool](#) [mic\\_mute](#)  
*mic\_mute: optional bool, microphone mute*
- int [spkr\\_vol](#)  
*spkr\_vol: optional int range 0-100, speaker volume*
- [alter\\_bool](#) [spkr\\_mute](#)  
*spkr\_mute: optional bool, speaker mute*
- mode [echo\\_cancel](#)  
*echo\_cancel: optional string, echo cancellation mode, "" means off*
- audio\_caps [caps](#)  
*caps*

### 10.4.1 Detailed Description

Audio config.

Definition at line 1036 of file config.h.

### 10.4.2 Field Documentation



#### 10.4.2.1 caps

`audio_caps vxg::cloud::agent::audio_config::caps`

`caps`

Definition at line 1049 of file `config.h`.

#### 10.4.2.2 echo\_cancel

`mode vxg::cloud::agent::audio_config::echo_cancel`

`echo_cancel`: optional string, echo cancellation mode, "" means off

Definition at line 1046 of file `config.h`.

#### 10.4.2.3 mic\_gain

`int vxg::cloud::agent::audio_config::mic_gain`

`mic_gain`: optional int range 0-100, microphone gain

Definition at line 1038 of file `config.h`.

#### 10.4.2.4 mic\_mute

`alter_bool vxg::cloud::agent::audio_config::mic_mute`

`mic_mute`: optional bool, microphone mute

Definition at line 1040 of file `config.h`.

#### 10.4.2.5 spkr\_mute

`alter_bool vxg::cloud::agent::audio_config::spkr_mute`

`spkr_mute`: optional bool, speaker mute

Definition at line 1044 of file `config.h`.

### 10.4.2.6 spkr\_vol

```
int vxg::cloud::agent::audio_config::spkr_vol
```

spkr\_vol: optional int range 0-100, speaker volume

Definition at line 1042 of file config.h.

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

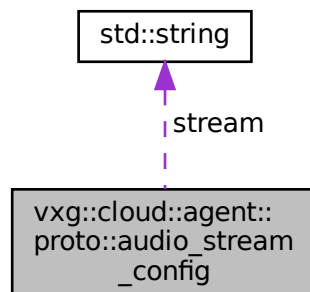
- [config.h](#)

## 10.5 vxg::cloud::agent::proto::audio\_stream\_config Struct Reference

Audio media stream config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::audio\_stream\_config:



### Data Fields

- **std::string** [stream](#)  
*Mandatory: audio ES to use.*
- [audio\\_format](#) **format**  
*Mandatory: audio encoding format.*
- int [brt](#)  
*Mandatory: bitrate, kbps.*
- double [srt](#)  
*Mandatory: samplerate, KHz.*

## 10.5.1 Detailed Description

Audio media stream config.

Definition at line 182 of file config.h.

## 10.5.2 Field Documentation

### 10.5.2.1 brt

```
int vxg::cloud::agent::proto::audio_stream_config::brt
```

Mandatory: bitrate, kbps.

Definition at line 193 of file config.h.

### 10.5.2.2 format

```
audio_format vxg::cloud::agent::proto::audio_stream_config::format
```

Mandatory: audio encoding format.

Definition at line 189 of file config.h.

### 10.5.2.3 srt

```
double vxg::cloud::agent::proto::audio_stream_config::srt
```

Mandatory: samplerate, KHz.

Definition at line 197 of file config.h.

### 10.5.2.4 stream

```
std::string vxg::cloud::agent::proto::audio_stream_config::stream
```

Mandatory: audio ES to use.

Definition at line 185 of file config.h.

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

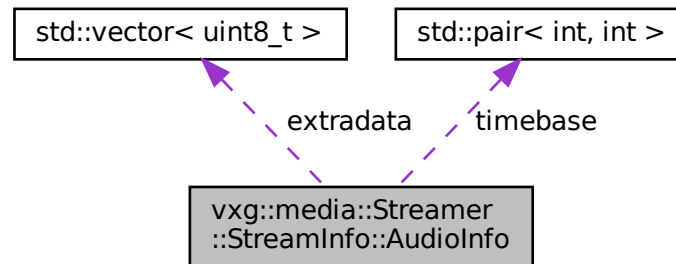
- [config.h](#)

## 10.6 vxg::media::Streamer::StreamInfo::AudioInfo Struct Reference

Audio stream info.

```
#include <streamer/base_streamer.h>
```

Collaboration diagram for vxg::media::Streamer::StreamInfo::AudioInfo:



### Data Fields

- [AudioCodec codec](#)  
*Audio codec.*
- int [channels](#)  
*Audio channels.*
- int [samplerate](#)  
*Audio samplerate.*
- int [bitrate](#)  
*Audio bitrate.*
- **std::pair**< int, int > [timebase](#)  
*Audio timestamps timescale.*
- **std::vector**< uint8\_t > [extradata](#)  
*Audio extradata. AAC requires one.*

### 10.6.1 Detailed Description

Audio stream info.

Definition at line 349 of file base\_streamer.h.

### 10.6.2 Field Documentation

### 10.6.2.1 bitrate

```
int vxg::media::Streamer::StreamInfo::AudioInfo::bitrate
```

Audio bitrate.

Definition at line 357 of file base\_streamer.h.

### 10.6.2.2 channels

```
int vxg::media::Streamer::StreamInfo::AudioInfo::channels
```

Audio channels.

Definition at line 353 of file base\_streamer.h.

### 10.6.2.3 codec

```
AudioCodec vxg::media::Streamer::StreamInfo::AudioInfo::codec
```

Audio codec.

Definition at line 351 of file base\_streamer.h.

### 10.6.2.4 extradata

```
std::vector<uint8_t> vxg::media::Streamer::StreamInfo::AudioInfo::extradata
```

Audio extradata. AAC requires one.

Definition at line 361 of file base\_streamer.h.

### 10.6.2.5 samplerate

```
int vxg::media::Streamer::StreamInfo::AudioInfo::samplerate
```

Audio samplerate.

Definition at line 355 of file base\_streamer.h.

### 10.6.2.6 timebase

```
std::pair<int, int> vxg::media::Streamer::StreamInfo::AudioInfo::timebase
```

Audio timestamps timescale.

Definition at line 359 of file base\_streamer.h.

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

- [base\\_streamer.h](#)

## 10.7 vxg::cloud::agent::callback Class Reference

VXG Cloud manager common callbacks class.

```
#include <agent/callback.h>
```

### Public Types

- typedef **std::unique\_ptr**< [callback](#) > [ptr](#)  
*std::unique\_ptr to callback*

### Public Member Functions

- virtual void [on\\_bye](#) (proto::command::bye\_reason reason)=0  
*VXG Cloud Bye command callback.*
- virtual void [on\\_registered](#) (const **std::string** &sid)  
*Registration on the Cloud has passed callback.*
- virtual bool [on\\_raw\\_msg](#) ( **std::string** client\_id, **std::string** &data)  
*raw message callback*
- virtual bool [on\\_get\\_log](#) ( **std::string** &log\_data)  
*Get logging data callback.*
- virtual bool [on\\_start\\_backward\\_audio](#) ( **std::string** url)  
*Start backward audio stream.*
- virtual bool [on\\_stop\\_backward\\_audio](#) ( **std::string** url)  
*Stop backward audio.*
- virtual bool [on\\_get\\_cam\\_video\\_config](#) (proto::video\_config &config)  
*Get video image config.*
- virtual bool [on\\_set\\_cam\\_video\\_config](#) (const proto::video\_config &config)  
*Set video input config.*
- virtual bool [on\\_get\\_cam\\_audio\\_config](#) (proto::audio\_config &config)  
*Get audio input configuration.*
- virtual bool [on\\_set\\_cam\\_audio\\_config](#) (const proto::audio\_config &config)  
*Set audio input/output config.*
- virtual bool [on\\_get\\_ptz\\_config](#) (proto::ptz\_config &config)  
*Get PTZ config.*
- virtual bool [on\\_cam\\_ptz](#) (proto::ptz\_command &command)

- PTZ command.*
- virtual bool [on\\_cam\\_ptz\\_preset](#) (proto::ptz\_preset &preset\_op)
- PTZ preset command.*
- virtual bool [on\\_get\\_osd\\_config](#) (proto::osd\_config &config)
- Get OSD config.*
- virtual bool [on\\_set\\_osd\\_config](#) (const proto::osd\_config &config)
- Set OSD config.*
- virtual bool [on\\_get\\_wifi\\_config](#) (proto::wifi\_config &config)
- Get WiFi config.*
- virtual bool [on\\_set\\_wifi\\_config](#) (const proto::wifi\_network &config)
- Set WiFi config.*
- virtual bool [on\\_get\\_motion\\_detection\\_config](#) (proto::motion\_detection\_config &config)
- Get motion detection configuration.*
- virtual bool [on\\_set\\_motion\\_detection\\_config](#) (const proto::motion\_detection\_config &config)
- Set motion detection config.*
- virtual bool [on\\_get\\_cam\\_events\\_config](#) (proto::events\_config &config)
- Get events configuration.*
- virtual bool [on\\_set\\_cam\\_events\\_config](#) (const proto::events\_config &config)
- Set motion detection config.*
- virtual bool [on\\_get\\_timezone](#) ( **std::string** &timezone)
- Get device timezone in IANA format.*
- virtual bool [on\\_set\\_timezone](#) ( **std::string** timezone)
- Set device timezone in IANA format.*
- virtual bool [on\\_get\\_memorycard\\_info](#) (proto::event\_object::memorycard\_info\_object &info)
- Get memory card information, If this callback returned false or if info status not equal to [proto::MCS\\_NORMAL](#), the recording will not be started, i.e.*
- virtual bool [on\\_cam\\_upgrade\\_firmware](#) (const **std::string** &firmware)
- Firmware upgrade.*
- virtual bool [on\\_audio\\_file\\_play](#) (const **std::string** audio\_file\_data, const **std::string** filename)
- Audio file play.*
- virtual bool [on\\_trigger\\_event](#) (proto::event\_object &event)

### 10.7.1 Detailed Description

VXG Cloud manager common callbacks class.

Definition at line 17 of file callback.h.

### 10.7.2 Member Typedef Documentation

#### 10.7.2.1 ptr

```
typedef std::unique_ptr<callback> vxg::cloud::agent::callback::ptr
```

**std::unique\_ptr** to callback

Definition at line 20 of file callback.h.

## 10.7.3 Member Function Documentation

### 10.7.3.1 on\_audio\_file\_play()

```
virtual bool vxg::cloud::agent::callback::on_audio_file_play (
    const std::string audio_file_data,
    const std::string filename ) [inline], [virtual]
```

Audio file play.

#### Parameters

in	<i>audio_file</i>	Audio file binary data.
in	<i>audio_file_format</i>	Audio file data format.

#### Returns

true if firware upgrade was successfull.

false if firware upgrade failed.

Definition at line 332 of file callback.h.

### 10.7.3.2 on\_bye()

```
virtual void vxg::cloud::agent::callback::on_bye (
    proto::command::bye_reason reason ) [pure virtual]
```

VXG Cloud Bye command callback.

#### Parameters

<i>reason</i>	bye reason
---------------	------------

### 10.7.3.3 on\_cam\_ptz()

```
virtual bool vxg::cloud::agent::callback::on_cam_ptz (
    proto::ptz_command & command ) [inline], [virtual]
```

PTZ command.



## Parameters

in	<i>command</i>	ptz command
----	----------------	-------------

## Returns

true success  
false PTZ command failure

Definition at line 162 of file callback.h.

### 10.7.3.4 on\_cam\_ptz\_preset()

```
virtual bool vxg::cloud::agent::callback::on_cam_ptz_preset (  
    proto::ptz_preset & preset_op ) [inline], [virtual]
```

PTZ preset command.

## Parameters

in, out	<i>preset_op</i>	ptz preset operation, if operation is <a href="#">proto::PA_CREATE</a> the callee should fill the token.
---------	------------------	--

## Returns

true PTZ preset operation success  
false PTZ preset operation failure

Definition at line 174 of file callback.h.

### 10.7.3.5 on\_cam\_upgrade\_firmware()

```
virtual bool vxg::cloud::agent::callback::on_cam_upgrade_firmware (  
    const std::string & firmware ) [inline], [virtual]
```

Firmware upgrade.

## Parameters

in	<i>firmware</i>	Firmware binary data.
----	-----------------	-----------------------

## Returns

true if firware upgrade was successfull.  
false if firware upgrade failed.

Definition at line 322 of file callback.h.

#### 10.7.3.6 on\_get\_cam\_audio\_config()

```
virtual bool vxg::cloud::agent::callback::on_get_cam_audio_config (
    proto::audio_config & config ) [inline], [virtual]
```

Get audio input configuration.

##### Parameters

out	<i>config</i>	audio input config
-----	---------------	--------------------

##### Returns

true get audio input configuration success  
false get audio input configuration failed

Definition at line 126 of file callback.h.

#### 10.7.3.7 on\_get\_cam\_events\_config()

```
virtual bool vxg::cloud::agent::callback::on_get_cam_events_config (
    proto::events_config & config ) [inline], [virtual]
```

Get events configuration.

##### Parameters

out	<i>config</i>	events config
-----	---------------	---------------

##### Returns

true if *config* is valid  
false if *config* is invalid

Definition at line 261 of file callback.h.

#### 10.7.3.8 on\_get\_cam\_video\_config()

```
virtual bool vxg::cloud::agent::callback::on_get_cam_video_config (
    proto::video_config & config ) [inline], [virtual]
```

Get video image config.

**Parameters**

out	<i>config</i>	video image config
-----	---------------	--------------------

**Returns**

true if get image config success  
false get image config failed

Definition at line 102 of file callback.h.

**10.7.3.9 on\_get\_log()**

```
virtual bool vxg::cloud::agent::callback::on_get_log (
    std::string & log_data ) [inline], [virtual]
```

Get logging data callback.

Cloud API provides the way to request log data using Cloud API

**Parameters**

<i>log_data</i>	log data
-----------------	----------

**Returns**

true on success  
false on failure

Definition at line 64 of file callback.h.

**10.7.3.10 on\_get\_memorycard\_info()**

```
virtual bool vxg::cloud::agent::callback::on_get_memorycard_info (
    proto::event_object::memorycard_info_object & info ) [inline], [virtual]
```

Get memory card information, If this callback returned false or if *info* status not equal to [proto::MCS\\_NORMAL](#), the recording will not be started, i.e.

no agent::media::stream::record\_start() will be called.

**Parameters**

out	<i>info</i>	memorycard info
-----	-------------	-----------------

**Returns**

true if `info` is valid  
false if `info` is not valid

Definition at line 312 of file `callback.h`.

**10.7.3.11 on\_get\_motion\_detection\_config()**

```
virtual bool vxg::cloud::agent::callback::on_get_motion_detection_config (  
    proto::motion_detection_config & config ) [inline], [virtual]
```

Get motion detection configuration.

**Parameters**

out	<i>config</i>	Motion detection config if return value is true
-----	---------------	---

**Returns**

true if `config` is valid  
false if failed to get motion detection config

Definition at line 235 of file `callback.h`.

**10.7.3.12 on\_get\_osd\_config()**

```
virtual bool vxg::cloud::agent::callback::on_get_osd_config (  
    proto::osd_config & config ) [inline], [virtual]
```

Get OSD config.

**Parameters**

out	<i>config</i>	OSD config
-----	---------------	------------

**Returns**

true OSD config get success, `config` is valid  
false OSD config get failure, `config` should not be used

Definition at line 186 of file `callback.h`.

### 10.7.3.13 on\_get\_ptz\_config()

```
virtual bool vxg::cloud::agent::callback::on_get_ptz_config (
    proto::ptz_config & config ) [inline], [virtual]
```

Get PTZ config.

#### Parameters

out	<i>config</i>	ptz config
-----	---------------	------------

#### Returns

true success

false Get PTZ config failed

Definition at line 150 of file callback.h.

### 10.7.3.14 on\_get\_timezone()

```
virtual bool vxg::cloud::agent::callback::on_get_timezone (
    std::string & timezone ) [inline], [virtual]
```

Get device timezone in IANA format.

#### Parameters

out	<i>timezone</i>	name in IANA format
-----	-----------------	---------------------

#### Returns

true if `timezone` is valid

false if `timezone` is not valid

Definition at line 285 of file callback.h.

### 10.7.3.15 on\_get\_wifi\_config()

```
virtual bool vxg::cloud::agent::callback::on_get_wifi_config (
    proto::wifi_config & config ) [inline], [virtual]
```

Get WiFi config.

## Parameters

out	<i>config</i>	WiFi config
-----	---------------	-------------

## Returns

true success  
false failed

Definition at line 210 of file callback.h.

## 10.7.3.16 on\_raw\_msg()

```
virtual bool vxg::cloud::agent::callback::on_raw_msg (
    std::string client_id,
    std::string & data ) [inline], [virtual]
```

raw message callback

## Parameters

in	<i>client_id</i>	unique id of the client, every raw messages session uses the same unique client_id
in, out	<i>data</i>	raw message payload from client, output value will be sent to the client if return value is true

## Returns

true raw message handled and reply in the output *data* argument should be sent to the client as reply  
false raw message handling failure, *data* output argument should not be sent to client

Definition at line 52 of file callback.h.

## 10.7.3.17 on\_registered()

```
virtual void vxg::cloud::agent::callback::on_registered (
    const std::string & sid ) [inline], [virtual]
```

Registration on the Cloud has passed callback.

## Parameters

<i>sid</i>	Cloud connection session id. Must be saved and provided via the <code>profile::global::instance().cm_register_sid</code> before the next <code>vxg::cloud::agent::manager::start()</code>
------------	---

Definition at line 36 of file callback.h.

### 10.7.3.18 on\_set\_cam\_audio\_config()

```
virtual bool vxg::cloud::agent::callback::on_set_cam_audio_config (
    const proto::audio_config & config ) [inline], [virtual]
```

Set audio input/output config.

#### Parameters

<i>config</i>	audio input/output config
---------------	---------------------------

#### Returns

true applied  
false failed to set config

Definition at line 138 of file callback.h.

### 10.7.3.19 on\_set\_cam\_events\_config()

```
virtual bool vxg::cloud::agent::callback::on_set_cam_events_config (
    const proto::events_config & config ) [inline], [virtual]
```

Set motion detection config.

#### Parameters

in	<i>config</i>	Motion detection config
----	---------------	-------------------------

#### Returns

true if *config* was successfully set  
false if failed to set *config*

Definition at line 273 of file callback.h.

### 10.7.3.20 on\_set\_cam\_video\_config()

```
virtual bool vxg::cloud::agent::callback::on_set_cam_video_config (
    const proto::video_config & config ) [inline], [virtual]
```

Set video input config.

**Parameters**

<i>config</i>	video input config
---------------	--------------------

**Returns**

true Video image input config was successfully set  
false Failed to set video input image config

Definition at line 114 of file callback.h.

**10.7.3.21 on\_set\_motion\_detection\_config()**

```
virtual bool vxg::cloud::agent::callback::on_set_motion_detection_config (  
    const proto::motion_detection_config & config ) [inline], [virtual]
```

Set motion detection config.

**Parameters**

in	<i>config</i>	motion detection config
----	---------------	-------------------------

**Returns**

true if *config* was successfully set  
false if failed to set *config*

Definition at line 248 of file callback.h.

**10.7.3.22 on\_set\_osd\_config()**

```
virtual bool vxg::cloud::agent::callback::on_set_osd_config (  
    const proto::osd_config & config ) [inline], [virtual]
```

Set OSD config.

**Parameters**

in	<i>config</i>	OSD config
----	---------------	------------

**Returns**

true OSD config was successfully set  
false failed to set OSD config



Definition at line 198 of file callback.h.

### 10.7.3.23 on\_set\_timezone()

```
virtual bool vxg::cloud::agent::callback::on_set_timezone (
    std::string timezone ) [inline], [virtual]
```

Set device timezone in IANA format.

#### Parameters

in	<i>timezone</i>	timezone in IANA format
----	-----------------	-------------------------

#### Returns

true if timezone was successfully set  
false if timezone was not set

Definition at line 297 of file callback.h.

### 10.7.3.24 on\_set\_wifi\_config()

```
virtual bool vxg::cloud::agent::callback::on_set_wifi_config (
    const proto::wifi_network & config ) [inline], [virtual]
```

Set WiFi config.

#### Parameters

in	<i>config</i>	WiFi configuration
----	---------------	--------------------

#### Returns

true if *config* is valid  
false if *config* is invalid

Definition at line 222 of file callback.h.

### 10.7.3.25 on\_start\_backward\_audio()

```
virtual bool vxg::cloud::agent::callback::on_start_backward_audio (
    std::string url ) [inline], [virtual]
```

Start backward audio stream.

**Parameters**

<i>url</i>	rtmp url for backward channel, device supports backward audio if <a href="#">on_get_cam_audio_config()</a> set proto::audio_config.caps spkr to true
------------	--

Implementation should start rtmp client by its own, final implementation is also responsible for the demuxing, decoding and rendering of the audio stream.

**Returns**

true on success  
false on failure

Definition at line 80 of file callback.h.

**10.7.3.26 on\_stop\_backward\_audio()**

```
virtual bool vxg::cloud::agent::callback::on_stop_backward_audio (  
    std::string url ) [inline], [virtual]
```

Stop backward audio.

**Parameters**

<i>url</i>	backward audio url which was used to start the backward channel
------------	---

Definition at line 91 of file callback.h.

**10.7.3.27 on\_trigger\_event()**

```
virtual bool vxg::cloud::agent::callback::on_trigger_event (  
    proto::event_object & event ) [inline], [virtual]
```

Definition at line 338 of file callback.h.

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

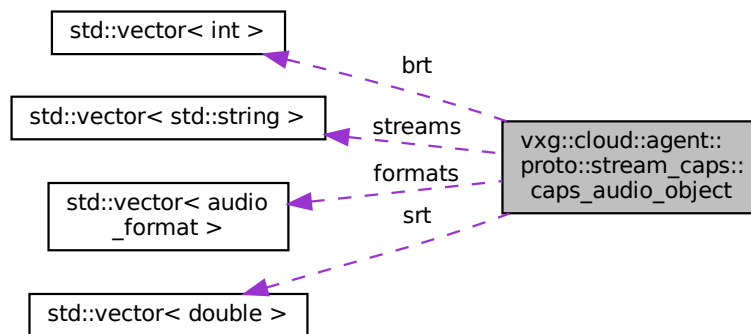
- [callback.h](#)

## 10.8 vxg::cloud::agent::proto::stream\_caps::caps\_audio\_object Struct Reference

Audio streams capabilities.

```
#include <agent-proto/objects/caps.h>
```

Collaboration diagram for vxg::cloud::agent::proto::stream\_caps::caps\_audio\_object:



### Data Fields

- **std::vector< std::string > streams**  
Mandatory: list of strings, audio ES that are covered by this capability config.
- **std::vector< audio\_format > formats**  
Mandatory: list of string, supported audio formats; currently only "AAC" and "G711U" is supported.
- **std::vector< int > brt**  
Mandatory: [min:int, max:int, step:int], range of bitrates, kbps.
- **std::vector< double > srt**  
Mandatory: list of float, supported samplersates.

### 10.8.1 Detailed Description

Audio streams capabilities.

Definition at line 247 of file caps.h.

### 10.8.2 Field Documentation

### 10.8.2.1 brt

```
std::vector<int> vxg::cloud::agent::proto::stream_caps::caps_audio_object::brt
```

Mandatory: [min:int, max:int, step:int], range of bitrates, kbps.

Definition at line 259 of file caps.h.

### 10.8.2.2 formats

```
std::vector<audio\_format> vxg::cloud::agent::proto::stream_caps::caps_audio_object::formats
```

Mandatory: list of string, supported audio formats; currently only "AAC" and "G711U" is supported.

Definition at line 255 of file caps.h.

### 10.8.2.3 srt

```
std::vector<double> vxg::cloud::agent::proto::stream_caps::caps_audio_object::srt
```

Mandatory: list of float, supported samplerates.

Definition at line 263 of file caps.h.

### 10.8.2.4 streams

```
std::vector< std::string > vxg::cloud::agent::proto::stream_caps::caps_audio_object::streams
```

Mandatory: list of strings, audio ES that are covered by this capability config.

Definition at line 250 of file caps.h.

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

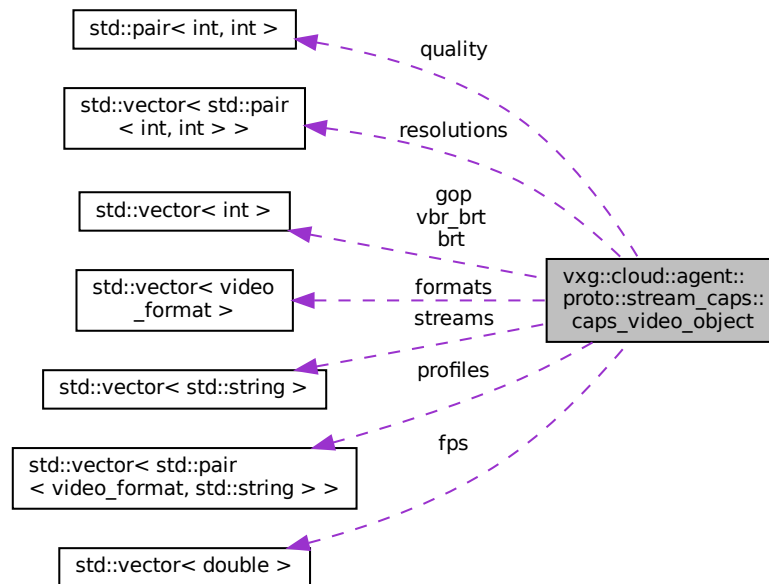
- [caps.h](#)

## 10.9 vxg::cloud::agent::proto::stream\_caps::caps\_video\_object Struct Reference

Video streams capabilities.

```
#include <agent-proto/objects/caps.h>
```

Collaboration diagram for vxg::cloud::agent::proto::stream\_caps::caps\_video\_object:



### Data Fields

- **std::vector< std::string > streams**  
Mandatory: list of strings, video ES that are covered by this capability config.
- **std::vector< video\_format > formats**  
Mandatory: list of string, supported video formats; currently only "H.264" is supported.
- **std::vector< std::pair< video\_format, std::string > > profiles**  
Optional: list of pairs [string (format), string (profile)], list of profiles for formats (when they have).
- **std::vector< std::pair< int, int > > resolutions**  
Mandatory: list of pairs [int (horz), int (vert)], - supported video resolutions.
- **std::vector< double > fps**  
Mandatory: list of float, supported framerates.
- bool **vbr**  
Mandatory: VBR is supported.
- **std::pair< int, int > quality**  
Optional: [min:int, max:int], range of quality for VBR.
- **std::vector< int > gop**  
Mandatory: gop: [min:int, max:int, step:int], range of gop sizes.
- **std::vector< int > brt**

- Mandatory: [min:int, max:int, step:int], range of bitrates, kbps.*
- **std::vector**< int > [vbr\\_brt](#)  
*Optional: [min:int, max:int, step:int], range of bitrates, kbps.*
  - bool [smoothing](#)  
*Optional: True when stream smoothing can be controlled.*

### 10.9.1 Detailed Description

Video streams capabilities.

Definition at line 177 of file caps.h.

### 10.9.2 Field Documentation

#### 10.9.2.1 brt

```
std::vector<int> vxg::cloud::agent::proto::stream_caps::caps_video_object::brt
```

Mandatory: [min:int, max:int, step:int], range of bitrates, kbps.

Definition at line 219 of file caps.h.

#### 10.9.2.2 formats

```
std::vector<video\_format> vxg::cloud::agent::proto::stream_caps::caps_video_object::formats
```

Mandatory: list of string, supported video formats; currently only "H.264" is supported.

Definition at line 185 of file caps.h.

#### 10.9.2.3 fps

```
std::vector<double> vxg::cloud::agent::proto::stream_caps::caps_video_object::fps
```

Mandatory: list of float, supported framerates.

Definition at line 203 of file caps.h.

### 10.9.2.4 gop

```
std::vector<int> vxg::cloud::agent::proto::stream_caps::caps_video_object::gop
```

Mandatory: gop: [min:int, max:int, step:int], range of gop sizes.

Definition at line 215 of file caps.h.

### 10.9.2.5 profiles

```
std::vector< std::pair<video_format, std::string> > vxg::cloud::agent::proto::stream_caps↵
::caps_video_object::profiles
```

Optional: list of pairs [string (format), string (profile)], list of profiles for formats (when they have).

Empty list means – color selection is not supported. “format” - one of listed in “formats” names. “profile”

- name of profile. Example: [“H.264”, “Baseline”], [“H.264”, “Main”], [“H.264”, “High”]]

Definition at line 194 of file caps.h.

### 10.9.2.6 quality

```
std::pair<int, int> vxg::cloud::agent::proto::stream_caps::caps_video_object::quality
```

Optional: [min:int, max:int], range of quality for VBR.

Definition at line 211 of file caps.h.

### 10.9.2.7 resolutions

```
std::vector< std::pair<int, int> > vxg::cloud::agent::proto::stream_caps::caps_video_↵
object::resolutions
```

Mandatory: list of pairs [int (horz), int (vert)], - supported video resolutions.

Definition at line 199 of file caps.h.

### 10.9.2.8 smoothing

```
bool vxg::cloud::agent::proto::stream_caps::caps_video_object::smoothing
```

Optional: True when stream smoothing can be controlled.

Definition at line 227 of file caps.h.

### 10.9.2.9 streams

```
std::vector< std::string> vxg::cloud::agent::proto::stream_caps::caps_video_object::streams
```

Mandatory: list of strings, video ES that are covered by this capability config.

Definition at line 180 of file caps.h.

### 10.9.2.10 vbr

```
bool vxg::cloud::agent::proto::stream_caps::caps_video_object::vbr
```

Mandatory: VBR is supported.

Definition at line 207 of file caps.h.

### 10.9.2.11 vbr\_brt

```
std::vector<int> vxg::cloud::agent::proto::stream_caps::caps_video_object::vbr_brt
```

Optional: [min:int, max:int, step:int], range of bitrates, kbps.

Definition at line 223 of file caps.h.

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

- [caps.h](#)

## 10.10 vxg::cloud::agent::proto::event\_caps Struct Reference

Events capabilities.

```
#include <agent-proto/objects/caps.h>
```



## Data Fields

- bool [stream](#)  
*stream: bool, event can generate stream start*
- bool [snapshot](#)  
*snapshot: bool, event is sent with snapshot*
- bool [periodic](#)  
*periodic: optional bool, the event is a periodic event (camera generates and processes it using specified time interval)*
- bool [trigger](#)  
*trigger: optional bool, the event can be triggered externally, using 6.7*
- bool [statefull](#)

### 10.10.1 Detailed Description

Events capabilities.

Definition at line 438 of file caps.h.

### 10.10.2 Field Documentation

#### 10.10.2.1 periodic

```
bool vxg::cloud::agent::proto::event_caps::periodic
```

periodic: optional bool, the event is a periodic event (camera generates and processes it using specified time interval)

Definition at line 447 of file caps.h.

#### 10.10.2.2 snapshot

```
bool vxg::cloud::agent::proto::event_caps::snapshot
```

snapshot: bool, event is sent with snapshot

Definition at line 443 of file caps.h.

#### 10.10.2.3 statefull

```
bool vxg::cloud::agent::proto::event_caps::statefull
```

Definition at line 469 of file caps.h.

### 10.10.2.4 stream

```
bool vxg::cloud::agent::proto::event_caps::stream
```

stream: bool, event can generate stream start

Definition at line 440 of file caps.h.

### 10.10.2.5 trigger

```
bool vxg::cloud::agent::proto::event_caps::trigger
```

trigger: optional bool, the event can be triggered externally, using 6.7

Definition at line 450 of file caps.h.

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

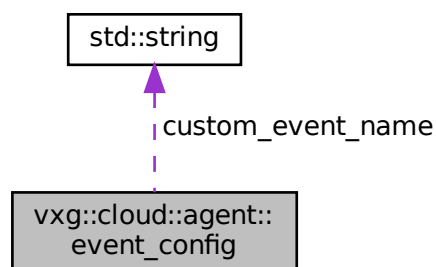
- [caps.h](#)

## 10.11 vxg::cloud::agent::event\_config Struct Reference

Event config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::event\_config:



### Public Member Functions

- bool [name\\_eq](#) (const [event\\_config](#) &r) const  
*Is-equal predicate based on event's name only.*
- bool [caps\\_eq](#) (const [event\\_config](#) &r) const  
*Is-equal predicate based on event's caps.*
- **std::string** [name](#) () const

## Data Fields

- event\_type [event](#)  
*event: string, event name, see 6.1 Events naming for details*
- std::string [custom\\_event\\_name](#)  
*Custom event name, used if event set to event\_type::ET\_CUSTOM.*
- bool [active](#)  
*active: bool, event is active; if not set, corresponding events will not be sent*
- bool [stream](#)  
*stream: bool, start stream when event happens*
- bool [snapshot](#)  
*snapshot: bool, generate snapshot when event happens*
- int [period](#)  
*period: optional int, an interval between periodic events, seconds*
- event\_caps [caps](#)  
*Event capabilities.*

### 10.11.1 Detailed Description

Event config.

Definition at line 897 of file config.h.

### 10.11.2 Member Function Documentation

#### 10.11.2.1 caps\_eq()

```
bool vxg::cloud::agent::event_config::caps_eq (
    const event\_config & r ) const [inline]
```

Is-equal predicate based on event's caps.

#### Parameters

<i>r</i>	
----------	--

#### Returns

true Compared configs have equal caps.

false Compared configs have non-equal caps.

Definition at line 937 of file config.h.

### 10.11.2.2 name()

```
std::string vxg::cloud::agent::event_config::name ( ) const [inline]
```

Definition at line 941 of file config.h.

### 10.11.2.3 name\_eq()

```
bool vxg::cloud::agent::event_config::name_eq (
    const event_config & r ) const [inline]
```

Is-equal predicate based on event's name only.

#### Parameters

<i>r</i>	
----------	--

#### Returns

- true Compared configs are for the event with equal names.
- false Compared configs are for events with non-equal names.

Definition at line 928 of file config.h.

## 10.11.3 Field Documentation

### 10.11.3.1 active

```
bool vxg::cloud::agent::event_config::active
```

active: bool, event is active; if not set, corresponding events will not be sent

Definition at line 906 of file config.h.

### 10.11.3.2 caps

```
event_caps vxg::cloud::agent::event_config::caps
```

Event capabilities.

Definition at line 921 of file config.h.

### 10.11.3.3 custom\_event\_name

```
std::string vxg::cloud::agent::event_config::custom_event_name
```

Custom event name, used if event set to event\_type::ET\_CUSTOM.

Definition at line 902 of file config.h.

### 10.11.3.4 event

```
event_type vxg::cloud::agent::event_config::event
```

event: string, event name, see 6.1 Events naming for details

Definition at line 899 of file config.h.

### 10.11.3.5 period

```
int vxg::cloud::agent::event_config::period
```

period: optional int, an interval between periodic events, seconds

Definition at line 915 of file config.h.

### 10.11.3.6 snapshot

```
bool vxg::cloud::agent::event_config::snapshot
```

snapshot: bool, generate snapshot when event happens

Definition at line 912 of file config.h.

### 10.11.3.7 stream

```
bool vxg::cloud::agent::event_config::stream
```

stream: bool, start stream when event happens

Definition at line 909 of file config.h.

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

- [config.h](#)

## 10.12 vxg::cloud::agent::manager::event\_state::event\_state\_caps Struct Reference

```
#include <agent/manager.h>
```

### Data Fields

- bool [stateful](#)
- bool [need\\_clip](#)
- bool [need\\_snapshot](#)

### 10.12.1 Detailed Description

Definition at line 71 of file manager.h.

### 10.12.2 Field Documentation

#### 10.12.2.1 need\_clip

```
bool vxg::cloud::agent::manager::event_state::event_state_caps::need_clip
```

Definition at line 73 of file manager.h.

#### 10.12.2.2 need\_snapshot

```
bool vxg::cloud::agent::manager::event_state::event_state_caps::need_snapshot
```

Definition at line 74 of file manager.h.

#### 10.12.2.3 stateful

```
bool vxg::cloud::agent::manager::event_state::event_state_caps::stateful
```

Definition at line 72 of file manager.h.

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

- [manager.h](#)

## 10.13 vxg::cloud::agent::event\_stream Class Reference

Event stream, abstract class for event generation.

```
#include <agent/event-stream.h>
```

### Public Types

- typedef **std::shared\_ptr**< [event\\_stream](#) > [ptr](#)  
*std::shared\_ptr to event\_stream*

### Public Member Functions

- [event\\_stream](#) ( **std::string** name)  
*Construct a new event stream object.*
- virtual [~event\\_stream](#) ()
- bool [notify](#) (proto::event\_object event)  
*Callback should be called to notify event.*
- virtual bool [start](#) ()=0  
*Start events generation, called by internal code when the events generation requested by the VXG Cloud.*
- virtual void [stop](#) ()=0  
*Stop events generation.*
- virtual bool [get\\_events](#) ( **std::vector**< proto::event\_config > &configs)=0  
*Get the events configs list This method should update config object and add all configurations for the events provided by this event stream.*
- virtual bool [set\\_events](#) (const **std::vector**< proto::event\_config > &config)=0  
*Set the events configuration.*
- virtual bool [trigger\\_event](#) (proto::event\_object &event)  
*Trigger event provided by event\_stream If get\_events() returned event config with proto::event\_config.caps.trigger == true and this event was triggered via the Cloud API this method will be called.*
- virtual bool [set\\_trigger\\_recording](#) (bool enabled, int pre, int post)=0  
*Turn on/off the event\_stream triggered recording and pre/post recording time.*
- virtual bool [init](#) ()=0
- virtual void [finit](#) ()=0

#### 10.13.1 Detailed Description

Event stream, abstract class for event generation.

Definition at line 14 of file event-stream.h.

#### 10.13.2 Member Typedef Documentation

### 10.13.2.1 ptr

```
typedef std::shared_ptr<event_stream> vxg::cloud::agent::event_stream::ptr
```

**std::shared\_ptr** to [event\\_stream](#)

Definition at line 25 of file event-stream.h.

## 10.13.3 Constructor & Destructor Documentation

### 10.13.3.1 event\_stream()

```
vxg::cloud::agent::event_stream::event_stream (
    std::string name ) [inline]
```

Construct a new event stream object.

#### Parameters

in	<i>name</i>	Event stream name, unique name for event stream
----	-------------	---

Definition at line 31 of file event-stream.h.

### 10.13.3.2 ~event\_stream()

```
virtual vxg::cloud::agent::event_stream::~~event_stream ( ) [inline], [virtual]
```

Definition at line 33 of file event-stream.h.

## 10.13.4 Member Function Documentation

### 10.13.4.1 finit()

```
virtual void vxg::cloud::agent::event_stream::finit ( ) [pure virtual]
```

### 10.13.4.2 get\_events()

```
virtual bool vxg::cloud::agent::event_stream::get_events (
    std::vector< proto::event_config > & configs ) [pure virtual]
```

Get the events configs list This method should update `config` object and add all configurations for the events provided by this event stream.

`config` may already include event configs reported by this `get_event()`, hence the implementation should consider this and do not include its event configs more than one time.



## Parameters

out	<i>configs</i>	Events configurations.
-----	----------------	------------------------

## Returns

true *configs* is valid.  
false *configs* is invalid, should not be applied.

## Note

This method MUST always return the configs with the same caps, otherwise the new config will not be applied by the library.

**10.13.4.3 init()**

```
virtual bool vxg::cloud::agent::event_stream::init ( ) [pure virtual]
```

**10.13.4.4 notify()**

```
bool vxg::cloud::agent::event_stream::notify (
    proto::event_object event ) [inline]
```

Callback should be called to notify event.

## Parameters

in	<i>event</i>	Event object
----	--------------	--------------

## Returns

true Event successfully notified  
false Notification failed

Definition at line 46 of file event-stream.h.

**10.13.4.5 set\_events()**

```
virtual bool vxg::cloud::agent::event_stream::set_events (
    const std::vector< proto::event_config > & config ) [pure virtual]
```

Set the events configuration.

**Parameters**

<i>config</i>	Events configurations list which includes all events reported by the system and other event streams, implementation should find own event configurations and apply them.
---------------	--

**Returns**

`true` `config` applied.  
`false` `config` not applied.

**10.13.4.6 set\_trigger\_recording()**

```
virtual bool vxg::cloud::agent::event_stream::set_trigger_recording (
    bool enabled,
    int pre,
    int post ) [pure virtual]
```

Turn on/off the [event\\_stream](#) triggered recording and pre/post recording time.

Triggered recording means that event generated by this [event\\_stream](#) should start recording. Final recorded file should have duration of pre time + duration of the even + post time.

**Note**

Trigger driven recording can be used if platform supports such type of recording, implementation of such type of recording should include specific [agent::media::stream](#) records exporting mechanism which handles two consecutive events pre/post time intersections.

**Parameters**

in	<i>enabled</i>	true if event stream should trigger the recording. Implementation may ignore this if not trigger driven record method is used.
in	<i>pre</i>	Pre recording time in milliseconds.
in	<i>post</i>	Post recording time in milliseconds.

**Returns**

`true`  
`false`

**10.13.4.7 start()**

```
virtual bool vxg::cloud::agent::event_stream::start ( ) [pure virtual]
```

Start events generation, called by internal code when the events generation requested by the VXG Cloud.

Event stream MUST immediately notify states of all stateful events after the [start\(\)](#) was invoked.

**Returns**

true Events generation started  
false Failed to start events generation

**10.13.4.8 stop()**

```
virtual void vxg::cloud::agent::event_stream::stop ( ) [pure virtual]
```

Stop events generation.

**10.13.4.9 trigger\_event()**

```
virtual bool vxg::cloud::agent::event_stream::trigger_event (
    proto::event_object & event ) [inline], [virtual]
```

Trigger event provided by [event\\_stream](#) If [get\\_events\(\)](#) returned event config with `proto::event_config.caps.trigger == true` and this event was triggered via the Cloud API this method will be called.

The logic of this method should be the same as for [vxg::cloud::agent::callback::on\\_trigger\\_event\(\)](#).

**See also**

[vxg::cloud::agent::callback::on\\_trigger\\_event\(\)](#)

**Parameters**

<i>event</i>	
--------------	--

**Returns**

true  
false

Definition at line 103 of file `event-stream.h`.

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

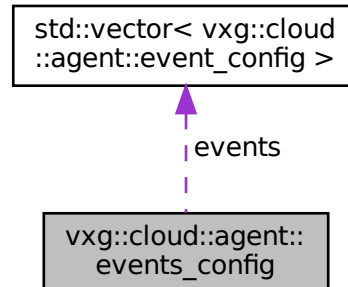
- [event-stream.h](#)

**10.14 vxg::cloud::agent::events\_config Struct Reference**

Events config, list of [event\\_config](#) objects.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::events\_config:



## Public Member Functions

- bool [get\\_event\\_config](#) (const event\_object &event, [event\\_config](#) &result)  
*Finds event which corresponds to [event\\_config](#) arg in the [events\\_config](#) structure.*

## Data Fields

- bool [enabled](#)  
*enabled: bool, indicates global events and event-driven streaming enabling flag*
- **std::vector**< [event\\_config](#) > [events](#)  
*events: list of [event\\_config](#) struct*

### 10.14.1 Detailed Description

Events config, list of [event\\_config](#) objects.

Definition at line 986 of file config.h.

### 10.14.2 Member Function Documentation

#### 10.14.2.1 get\_event\_config()

```
bool vxg::cloud::agent::events_config::get_event_config (
    const event_object & event,
    event_config & result ) [inline]
```

Finds event which corresponds to [event\\_config](#) arg in the [events\\_config](#) structure.

## Parameters

in	<i>event</i>	- event_object, event_object.event used to find the <a href="#">event_config</a>
out	<i>result</i>	- if <a href="#">event_config</a> found it will be stored here

## Returns

true event found  
false event not found

Definition at line 1003 of file config.h.

### 10.14.3 Field Documentation

#### 10.14.3.1 enabled

```
bool vxg::cloud::agent::events_config::enabled
```

enabled: bool, indicates global events and event-driven streaming enabling flag

Definition at line 989 of file config.h.

#### 10.14.3.2 events

```
std::vector<event_config> vxg::cloud::agent::events_config::events
```

events: list of [event\\_config](#) struct

Definition at line 992 of file config.h.

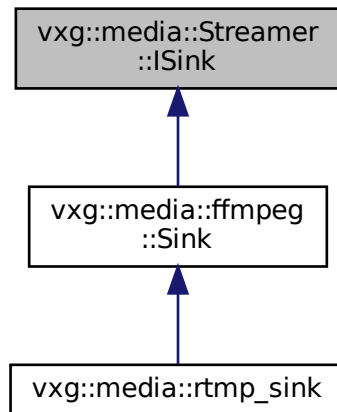
The documentation for this struct was generated from the following file:

- [config.h](#)

## 10.15 vxg::media::Streamer::ISink Class Reference

```
#include <streamer/base_streamer.h>
```

Inheritance diagram for vxg::media::Streamer::ISink:



### Public Types

- typedef `std::shared_ptr< ISink > ptr`  
*std::shared\_ptr alias*
- typedef `std::unique_ptr< ISink > PtrU`  
*std::unique\_ptr alias*

### Public Member Functions

- `ISink (uint8_t prio=SINK_THREAD_PRIO)`  
*Construct a new ISink object.*
- virtual `~ISink ()`
- virtual bool `init ( std::string url="" )=0`  
*Init sink.*
- virtual bool `finit ()=0`  
*Deinit sink.*
- virtual bool `process ( std::shared_ptr< MediaFrame > frame)=0`  
*Process next media frame.*
- virtual bool `droppable ()=0`  
*If sink of with dropping its media frames.*
- virtual bool `negotiate ( std::vector< Streamer::StreamInfo > info)`  
*Negotiation callback, this method called with collected from the ISource::negotiate media stream description.*
- virtual void `error (StreamError error)=0`  
*Media processing error callback, called when ISink::process returned false or linked source's ISource::pullFrame returned false, or when ISource::error was called.*

- virtual `std::string name` ()=0  
*Sink name.*
- virtual `cloud::duration duration` ()  
*Processed stream duration.*
- void `set_eos_cb` ( `std::function`< void(`cloud::duration`)> eos\_cb)
- void `set_eos` (bool eos)

### 10.15.1 Detailed Description

Definition at line 492 of file `base_streamer.h`.

### 10.15.2 Member Typedef Documentation

#### 10.15.2.1 ptr

```
typedef std::shared_ptr<ISink> vxg::media::Streamer::ISink::ptr
```

**std::shared\_ptr** alias

Definition at line 497 of file `base_streamer.h`.

#### 10.15.2.2 PtrU

```
typedef std::unique_ptr<ISink> vxg::media::Streamer::ISink::PtrU
```

**std::unique\_ptr** alias

Definition at line 499 of file `base_streamer.h`.

### 10.15.3 Constructor & Destructor Documentation

#### 10.15.3.1 ISink()

```
vxg::media::Streamer::ISink::ISink (
    uint8_t prio = SINK_THREAD_PRIO ) [inline]
```

Construct a new `ISink` object.

**Parameters**

<i>prio</i>	internall thread priority, used on RTOS.
-------------	--

Definition at line 504 of file base\_streamer.h.

**10.15.3.2 ~ISink()**

```
virtual vxg::media::Streamer::ISink::~~ISink ( ) [inline], [virtual]
```

Definition at line 510 of file base\_streamer.h.

**10.15.4 Member Function Documentation****10.15.4.1 droppable()**

```
virtual bool vxg::media::Streamer::ISink::droppable ( ) [pure virtual]
```

If sink of with dropping its media frames.

**Returns**

true Internal media thread allowed to drop frames if internal media queue is full.  
false No media frames dropping allowed.

Implemented in [vxg::media::rtmp\\_sink](#), and [vxg::media::ffmpeg::Sink](#).

**10.15.4.2 duration()**

```
virtual cloud::duration vxg::media::Streamer::ISink::duration ( ) [inline], [virtual]
```

Processed stream duration.

**Returns**

duration

Reimplemented in [vxg::media::ffmpeg::Sink](#).

Definition at line 605 of file base\_streamer.h.

**10.15.4.3 error()**

```
virtual void vxg::media::Streamer::ISink::error (
    StreamError error ) [pure virtual]
```

Media processing error callback, called when [ISink::process](#) returned false or linked source's [ISource::pullFrame](#) returned false, or when [ISource::error](#) was called.



## Parameters

<i>error</i>	Error type.
--------------	-------------

Implemented in [vxg::media::rtmp\\_sink](#), and [vxg::media::ffmpeg::Sink](#).

#### 10.15.4.4 finit()

```
virtual bool vxg::media::Streamer::ISink::finit ( ) [pure virtual]
```

Deinit sink.

## Returns

true finit success.  
false finit failed.

Implemented in [vxg::media::ffmpeg::Sink](#).

#### 10.15.4.5 init()

```
virtual bool vxg::media::Streamer::ISink::init (
    std::string url = "" ) [pure virtual]
```

Init sink.

## Parameters

<i>in</i>	<i>url</i>	Url if needed.
-----------	------------	----------------

## Returns

true init success.  
false init failed.

Implemented in [vxg::media::ffmpeg::Sink](#), and [vxg::media::rtmp\\_sink](#).

#### 10.15.4.6 name()

```
virtual std::string vxg::media::Streamer::ISink::name ( ) [pure virtual]
```

Sink name.

## Returns

**std::string**

Implemented in [vvg::media::rtmp\\_sink](#), and [vvg::media::ffmpeg::Sink](#).

**10.15.4.7 negotiate()**

```
virtual bool vvg::media::Streamer::ISink::negotiate (
    std::vector< Streamer::StreamInfo > info ) [inline], [virtual]
```

Negotiation callback, this method called with collected from the [ISource::negotiate](#) media stream description.

## Parameters

<i>info</i>	List of elementary streams descriptions.
-------------	--

## Returns

true If streams descriptions accepted.  
false Streams not accepted, will cause media thread stopping.

Reimplemented in [vvg::media::ffmpeg::Sink](#), and [vvg::media::rtmp\\_sink](#).

Definition at line 557 of file base\_streamer.h.

**10.15.4.8 process()**

```
virtual bool vvg::media::Streamer::ISink::process (
    std::shared_ptr< MediaFrame > frame ) [pure virtual]
```

Process next media frame.

Internal function called by media thread, the last function of media frame travel. Final class process frame in this function: sends to server, writes on disk etc.

## Parameters

in	<i>frame</i>	Media frame.
----	--------------	--------------

## Returns

true Media frame successfully processed.  
false Media frame processing failed.

#### 10.15.4.9 set\_eos()

```
void vxg::media::Streamer::ISink::set_eos (
    bool eos ) [inline]
```

Definition at line 668 of file base\_streamer.h.

#### 10.15.4.10 set\_eos\_cb()

```
void vxg::media::Streamer::ISink::set_eos_cb (
    std::function< void(cloud::duration)> eos_cb ) [inline]
```

Definition at line 664 of file base\_streamer.h.

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

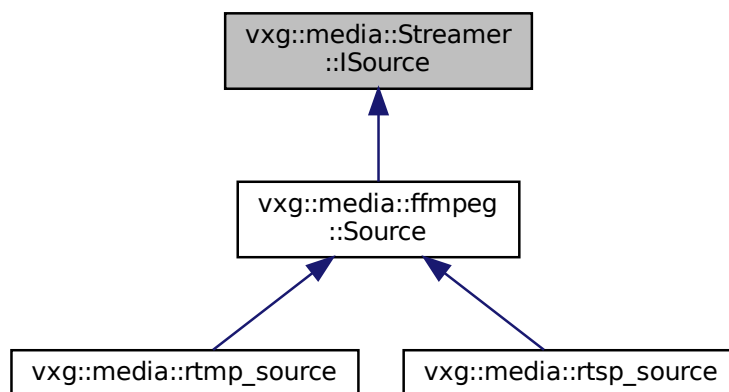
- [base\\_streamer.h](#)

## 10.16 vxg::media::Streamer::ISource Class Reference

[ISource](#) interface class.

```
#include <streamer/base_streamer.h>
```

Inheritance diagram for vxg::media::Streamer::ISource:



### Public Types

- enum [Mode](#) { [PULL](#), [PUSH](#) }  
*Source operation mode.*
- typedef `std::shared_ptr< ISource >` [ptr](#)

## Public Member Functions

- `ISource` (`uint8_t _prio=SRC_THREAD_PRIO`, `Mode _mode=PULL`, `bool drop=true`)  
*Construct a new `ISource` object.*
- virtual `bool init ( std::string url="" )=0`  
*Init source.*
- virtual `void finit ()=0`  
*Finit souce.*
- virtual `void error (StreamError stream_error)`  
*Error notification.*
- virtual `std::vector< Streamer::StreamInfo > negotiate ()=0`  
*Negotiation callback.*
- virtual `std::shared_ptr< MediaFrame > pullFrame ()=0`  
*Main method of the Mode::PULL mode data producing.*
- virtual `std::string name ()=0`  
*Source class name.*
- `void pushFrame ( std::shared_ptr< MediaFrame > frame)`  
*Implementation should call this method to provide media frames in the Mode::PUSH source operation mode.*

## Protected Attributes

- `Mode mode_`

### 10.16.1 Detailed Description

`ISource` interface class.

Definition at line 691 of file `base_streamer.h`.

### 10.16.2 Member Typedef Documentation

#### 10.16.2.1 ptr

```
typedef std::shared_ptr<ISource> vxg::media::Streamer::ISource::ptr
```

Definition at line 696 of file `base_streamer.h`.

### 10.16.3 Member Enumeration Documentation

#### 10.16.3.1 Mode

```
enum vxg::media::Streamer::ISource::Mode
```

Source operation mode.

## Enumerator

PULL	Pull mode. The <a href="#">ISource::pullFrame()</a> will be called from the separate thread. User should implement it and return <code>std::shared_ptr&lt;MediaFrame&gt;</code> .
PUSH	Push mode. Inherited class should feed media data on its own by calling the <a href="#">ISource::pushFrame()</a> method with <a href="#">MediaFrame</a> object passed as argument.

Definition at line 698 of file `base_streamer.h`.

## 10.16.4 Constructor & Destructor Documentation

### 10.16.4.1 ISource()

```
vxg::media::Streamer::ISource::ISource (
    uint8_t _prio = SRC_THREAD_PRIO,
    Mode _mode = PULL,
    bool drop = true ) [inline]
```

Construct a new [ISource](#) object.

## Parameters

in	<i>_prio</i>	Push thread priority. Used if <i>_mode</i> is <code>Mode::PUSH</code> .
in	<i>_mode</i>	Source operating mode.
in	<i>drop</i>	If true the media frames may be dropped if queue is full.

Definition at line 714 of file `base_streamer.h`.

## 10.16.5 Member Function Documentation

### 10.16.5.1 error()

```
virtual void vxg::media::Streamer::ISource::error (
    StreamError stream_error ) [inline], [virtual]
```

Error notification.

Calling this method will inform media thread and all sinks about error happened in the source.

## Parameters

in	<i>stream_error</i>	
----	---------------------	--

Definition at line 749 of file base\_streamer.h.

#### 10.16.5.2 finit()

```
virtual void vxg::media::Streamer::ISource::finit ( ) [pure virtual]
```

Finit souce.

Implemented in [vxg::media::ffmpeg::Source](#).

#### 10.16.5.3 init()

```
virtual bool vxg::media::Streamer::ISource::init (
    std::string url = "" ) [pure virtual]
```

Init source.

##### Parameters

<i>url</i>	Url if needed.
------------	----------------

##### Returns

true Init success.

false Init failed.

Implemented in [vxg::media::ffmpeg::Source](#), [vxg::media::rtsp\\_source](#), and [vxg::media::rtmp\\_source](#).

#### 10.16.5.4 name()

```
virtual std::string vxg::media::Streamer::ISource::name ( ) [pure virtual]
```

Source class name.

##### Returns

**std::string**

Implemented in [vxg::media::rtsp\\_source](#), and [vxg::media::ffmpeg::Source](#).

### 10.16.5.5 negotiate()

```
virtual std::vector<Streamer::StreamInfo> vxg::media::Streamer::ISource::negotiate ( ) [pure virtual]
```

Negotiation callback.

Called by internals. Class implementation should return the list of the streams info source will be producing for the sinks, this list will be then passed to the [ISink::negotiate](#) method.

#### Returns

std::vector<Streamer::StreamInfo>

Implemented in [vxg::media::ffmpeg::Source](#).

### 10.16.5.6 pullFrame()

```
virtual std::shared_ptr<MediaFrame> vxg::media::Streamer::ISource::pullFrame ( ) [pure virtual]
```

Main method of the Mode::PULL mode data producing.

Called by internals if the source operation mode is Mode::PULL. Implementation should return media frame object with correctly filled fields.

#### Returns

std::shared\_ptr<MediaFrame>

Implemented in [vxg::media::ffmpeg::Source](#).

### 10.16.5.7 pushFrame()

```
void vxg::media::Streamer::ISource::pushFrame (
    std::shared_ptr< MediaFrame > frame ) [inline]
```

Implementation should call this method to provide media frames in the Mode::PUSH source operation mode.

#### Parameters

<i>frame</i>	smart pointer to <a href="#">MediaFrame</a> .
--------------	---

Definition at line 849 of file base\_streamer.h.

## 10.16.6 Field Documentation

### 10.16.6.1 mode\_

`Mode vxg::media::Streamer::ISource::mode_` [protected]

Definition at line 986 of file `base_streamer.h`.

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

- [base\\_streamer.h](#)

## 10.17 vxg::logger Class Reference

Logger class, current implementation based on spdlog.

```
#include <utils/logging.h>
```

### Data Structures

- struct [options](#)

### Public Types

- enum [loglevel](#) {  
    [lvl\\_crit](#), [lvl\\_off](#), [lvl\\_error](#), [lvl\\_warn](#),  
    [lvl\\_info](#), [lvl\\_debug](#), [lvl\\_trace](#) }
- typedef `std::shared_ptr< spdlog::logger >` [logger\\_ptr](#)

### Static Public Member Functions

- static `std::shared_ptr< spdlog::logger >` [instance](#) ( `std::string` name)  
*Get pointer to the instance of the named spdlog::logger object.*
- static void [reset](#) (int argc, char \*\*argv, [loglevel](#) l, `std::string` syslog\_ident="VXGCloudAgentDefault", `std::string` crash\_logfile\_path="", `std::string` logfile\_path="", `size_t` logfile\_max\_size=(1024 \*1024), `size_t` logfile\_max\_files=3)  
*Reset default logger parameters.*
- static void [reset](#) (const [options](#) &opts)
- static void [set\\_level](#) ([logger\\_ptr](#) log\_ptr, [loglevel](#) lvl)  
*Change the logger object loglevel.*
- template<typename FormatString , typename... Args>  
    static void [info](#) (const FormatString &fmt, const Args &... args)  
*Static info log.*
- template<typename FormatString , typename... Args>  
    static void [error](#) (const FormatString &fmt, const Args &... args)



- template<typename FormatString , typename... Args>  
static void [warn](#) (const FormatString &fmt, const Args &... args)
- template<typename FormatString , typename... Args>  
static void [debug](#) (const FormatString &fmt, const Args &... args)
- template<typename FormatString , typename... Args>  
static void [trace](#) (const FormatString &fmt, const Args &... args)
- template<typename T >  
static void [trace](#) (const T &msg)
- template<typename T >  
static void [debug](#) (const T &msg)
- template<typename T >  
static void [info](#) (const T &msg)
- template<typename T >  
static void [warn](#) (const T &msg)
- template<typename T >  
static void [error](#) (const T &msg)

### 10.17.1 Detailed Description

Logger class, current implementation based on spdlog.

Definition at line 22 of file logging.h.

### 10.17.2 Member Typedef Documentation

#### 10.17.2.1 logger\_ptr

```
typedef std::shared\_ptr<spdlog::logger> vxg::logger::logger\_ptr
```

Definition at line 24 of file logging.h.

### 10.17.3 Member Enumeration Documentation

#### 10.17.3.1 loglevel

```
enum vxg::logger::loglevel
```

Enumerator

<a href="#">lvl_crit</a>	
<a href="#">lvl_off</a>	
<a href="#">lvl_error</a>	
<a href="#">lvl_warn</a>	
<a href="#">lvl_info</a>	
<a href="#">lvl_debug</a>	
<a href="#">lvl_trace</a>	

Definition at line 25 of file logging.h.

## 10.17.4 Member Function Documentation

### 10.17.4.1 debug() [1/2]

```
template<typename FormatString , typename... Args>
static void vxg::logger::debug (
    const FormatString & fmt,
    const Args &... args ) [inline], [static]
```

Definition at line 282 of file logging.h.

### 10.17.4.2 debug() [2/2]

```
template<typename T >
static void vxg::logger::debug (
    const T & msg ) [inline], [static]
```

Definition at line 295 of file logging.h.

### 10.17.4.3 error() [1/2]

```
template<typename FormatString , typename... Args>
static void vxg::logger::error (
    const FormatString & fmt,
    const Args &... args ) [inline], [static]
```

Definition at line 274 of file logging.h.

### 10.17.4.4 error() [2/2]

```
template<typename T >
static void vxg::logger::error (
    const T & msg ) [inline], [static]
```

Definition at line 310 of file logging.h.

### 10.17.4.5 info() [1/2]

```
template<typename FormatString , typename... Args>
static void vxg::logger::info (
    const FormatString & fmt,
    const Args &... args ) [inline], [static]
```

Static info log.

## Template Parameters

<i>FormatString</i>	
<i>Args</i>	

## Parameters

<i>fmt</i>	
<i>args</i>	

Definition at line 270 of file logging.h.

## 10.17.4.6 info() [2/2]

```
template<typename T >
static void vxg::logger::info (
    const T & msg ) [inline], [static]
```

Definition at line 300 of file logging.h.

## 10.17.4.7 instance()

```
static std::shared_ptr<spdlog::logger> vxg::logger::instance (
    std::string name ) [inline], [static]
```

Get pointer to the instance of the named spdlog::logger object.

On the very first call creates default logger named 'default'. Constructs new logger if logger with such name was never requested

## Parameters

in	<i>name</i>	Logger name. If logger with such name was already created, then it will be reused, otherwise a new one will be constructed.
----	-------------	---

## Returns

`std::shared_ptr<spdlog::logger>`

Definition at line 192 of file logging.h.

**10.17.4.8 reset()** [1/2]

```
static void vxg::logger::reset (
    const options & opts ) [inline], [static]
```

Definition at line 239 of file logging.h.

**10.17.4.9 reset()** [2/2]

```
static void vxg::logger::reset (
    int argc,
    char ** argv,
    loglevel l,
    std::string syslog_ident = "VXGCloudAgentDefault",
    std::string crash_logfile_path = "",
    std::string logfile_path = "",
    size_t logfile_max_size = (1024 * 1024),
    size_t logfile_max_files = 3 ) [inline], [static]
```

Reset default logger parameters.

Used to change all loggers parameters such as syslog/file sinks usage. Should be called before very first [logger::instance\(\)](#) call to take effect. If wasn't called the default console logging sink only will be used for all loggers.

**Deprecated** Use [reset\(const options& opts\)](#)

**Parameters**

<i>argc</i>	Process argc
<i>argv</i>	Process argv
<i>l</i>	default loglevel, all loggers will be created with this loglevel, can be overridden with SPDLOG_LEVEL env variable
<i>syslog_ident</i>	Syslog identification string, if empty syslog logging will be disabled.
<i>logfile_path</i>	Rotating plain log file path, if empty no plain log file will be used.
<i>logfile_max_size</i>	Max log file size before invoking logrotate.
<i>logfile_max_files</i>	Max number if rotating logfiles.

Definition at line 220 of file logging.h.

**10.17.4.10 set\_level()**

```
static void vxg::logger::set_level (
    logger_ptr log_ptr,
    loglevel lvl ) [inline], [static]
```

Change the logger object loglevel.

## Parameters

<i>log_ptr</i>	Logger object pointer.
<i>lvl</i>	New loglevel.

Definition at line 259 of file logging.h.

**10.17.4.11 trace()** [1/2]

```
template<typename FormatString , typename... Args>
static void vxg::logger::trace (
    const FormatString & fmt,
    const Args &... args ) [inline], [static]
```

Definition at line 286 of file logging.h.

**10.17.4.12 trace()** [2/2]

```
template<typename T >
static void vxg::logger::trace (
    const T & msg ) [inline], [static]
```

Definition at line 290 of file logging.h.

**10.17.4.13 warn()** [1/2]

```
template<typename FormatString , typename... Args>
static void vxg::logger::warn (
    const FormatString & fmt,
    const Args &... args ) [inline], [static]
```

Definition at line 278 of file logging.h.

**10.17.4.14 warn()** [2/2]

```
template<typename T >
static void vxg::logger::warn (
    const T & msg ) [inline], [static]
```

Definition at line 305 of file logging.h.

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

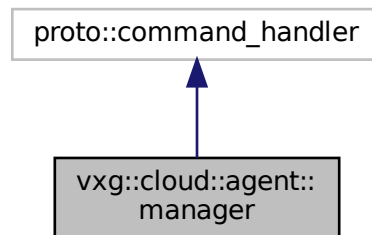
- [logging.h](#)

## 10.18 vxg::cloud::agent::manager Class Reference

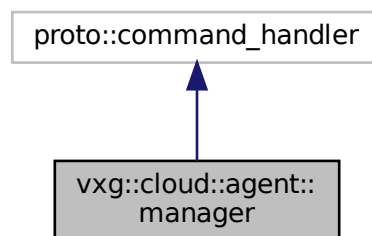
VXG Cloud agent manager class.

```
#include <agent/manager.h>
```

Inheritance diagram for vxg::cloud::agent::manager:



Collaboration diagram for vxg::cloud::agent::manager:



### Public Types

- typedef `std::shared_ptr< manager > ptr`  
*shared\_ptr to manager object*

### Public Member Functions

- bool `start ()`  
*Start internal workflow, this is the main function which starts all internal threads and connections.*
- void `stop ()`  
*Stop manager, disconnect from the VXG Cloud.*

## Static Public Member Functions

- static `manager::ptr create (callback::ptr callback, proto::access_token::ptr access_token, std::vector< agent::media::stream::ptr > media_streams, std::vector< event_stream::ptr > event_streams= std::vector< event_stream::ptr >(0))`

Create manager object.

## Protected Member Functions

- bool `notify_event (proto::event_object event)`
  - bool `_update_storage_status ()`
  - void `_stop_all_streams ()`
- Streams helpers.
- void `_stop_stream (agent::media::stream::ptr s)`
  - void `_stop_all_event_streams ()`
  - void `_schedule_periodic_events (proto::events_config &events_conf)`
  - void `_schedule_periodic_event (const proto::event_config &event_conf)`
  - void `_cancel_periodic_event (const proto::event_config &event_conf)`
  - void `_cancel_periodic_events (const proto::events_config &events_conf)`
  - void `_append_internal_custom_events (proto::events_config &config)`
  - void `__trigger_periodic_event (const proto::event_config &event_conf)`
  - void `_init_events_states (const proto::events_config &config)`
  - void `_load_events_configs (proto::events_config &config)`
  - bool `_update_events_configs (const std::vector< proto::event_config > &new_configs, std::vector< proto::event_config > &dest_configs)`
  - bool `_update_event_stream_configs (const std::string &stream_name, const std::vector< proto::event_config > &new_configs)`
  - `event_stream::ptr lookup_event_stream_by_event (const proto::event_object &event)`
  - `event_stream::ptr lookup_event_stream (const std::string &name)`
  - bool `handle_stream_event (proto::event_object &event)`
  - bool `_handle_stream_stateful_event (proto::event_object &event, event_state::stream_delivery_mode delivery_mode)`
  - bool `_handle_stream_stateless_event (proto::event_object &event, event_state::stream_delivery_mode delivery_mode)`
  - bool `handle_event_snapshot (proto::event_object &event)`
  - bool `handle_event_meta_file (proto::event_object &event)`
  - bool `__notify_record_event ( std::string stream_id, bool on)`
  - `event_state::stream_delivery_mode _current_delivery_mode ()`
  - bool `_schedule_direct_upload (proto::get_direct_upload_url get_upload_url)`
  - bool `_cancel_direct_uploads_by_ticket ( std::string ticket)`
  - bool `_request_direct_upload_video (proto::get_direct_upload_url direct_upload)`
  - bool `_request_direct_upload_snapshot (proto::get_direct_upload_url direct_upload)`
  - void `_update_direct_upload_queue_latency ()`
  - bool `direct_upload_sync_cb ()`
  - `agent::media::stream::ptr lookup_stream ( std::string name)`
  - virtual bool `on_get_stream_config (proto::stream_config &config)`
  - virtual bool `on_set_stream_config (const proto::stream_config &config)`
  - virtual bool `on_get_motion_detection_config (proto::motion_detection_config &config)`
  - virtual bool `on_set_motion_detection_config (const proto::motion_detection_config &config)`
  - virtual bool `on_get_cam_video_config (proto::video_config &config)`
  - virtual bool `on_set_cam_video_config (const proto::video_config &config)`
  - virtual bool `on_get_cam_events_config (proto::events_config &config)`
  - virtual bool `on_set_cam_events_config (const proto::events_config &config)`
  - virtual bool `on_get_cam_audio_config (proto::audio_config &config)`

- virtual bool [on\\_set\\_cam\\_audio\\_config](#) (const proto::audio\_config &config)
- virtual bool [on\\_get\\_ptz\\_config](#) (proto::ptz\_config &config)
- virtual bool [on\\_cam\\_ptz](#) (proto::ptz\_command command)
- virtual bool [on\\_cam\\_ptz\\_preset](#) (proto::ptz\_preset &preset\_op)
- virtual bool [on\\_get\\_osd\\_config](#) (proto::osd\_config &config)
- virtual bool [on\\_set\\_osd\\_config](#) (const proto::osd\_config &config)
- virtual bool [on\\_get\\_wifi\\_config](#) (proto::wifi\_config &config)
- virtual bool [on\\_set\\_wifi\\_config](#) (const [proto::wifi\\_network](#) &config)
- virtual bool [on\\_stream\\_start](#) (const **std::string** &streamId, int publishSessionID, proto::stream\_reason reason)
- virtual bool [on\\_stream\\_stop](#) (const **std::string** &streamId, proto::stream\_reason reason)
- virtual bool [on\\_get\\_stream\\_caps](#) (proto::stream\_caps &caps)
- virtual bool [on\\_get\\_supported\\_streams](#) (proto::supported\_streams\_config &supportedStreamsConfig)
- virtual bool [on\\_cam\\_upgrade\\_firmware](#) ( **std::string** url)
- virtual bool [on\\_raw\\_message](#) ( **std::string** client\_id, **std::string** &data)
- virtual bool [on\\_set\\_stream\\_by\\_event](#) (proto::stream\_by\_event\_config conf)
- virtual bool [on\\_get\\_stream\\_by\\_event](#) (proto::stream\_by\_event\_config &conf)
- virtual bool [on\\_update\\_preview](#) ( **std::string** url)
- virtual bool [on\\_direct\\_upload\\_url](#) (const proto::command::direct\_upload\_url\_base &direct\_upload, int eventId, int ref\_id)
- virtual bool [on\\_get\\_log](#) ()
- virtual void [on\\_prepared](#) ()
- virtual void [on\\_closed](#) (int error, proto::command::bye\_reason reason)
- virtual bool [on\\_get\\_timezone](#) ( **std::string** &timezone)
- virtual bool [on\\_set\\_timezone](#) ( **std::string** timezone)
- void [on\\_set\\_periodic\\_events](#) (const char \*name, int period, bool active)
- virtual bool [on\\_audio\\_file\\_play](#) ( **std::string** url)
- virtual bool [on\\_start\\_backward](#) ( **std::string** &url)
- virtual bool [on\\_stop\\_backward](#) ( **std::string** &url)
- virtual bool [on\\_get\\_cam\\_memorycard\\_timeline](#) (proto::command::cam\_memorycard\_timeline &timeline)
- virtual bool [on\\_cam\\_memorycard\\_synchronize](#) (proto::command::cam\_memorycard\_synchronize\_status &synchronize\_status, vxg::cloud::time start, vxg::cloud::time end)
- virtual bool [on\\_cam\\_memorycard\\_synchronize\\_cancel](#) (const **std::string** &request\_id)
- virtual bool [on\\_cam\\_memorycard\\_recording](#) (const **std::string** &stream\_id, bool enabled)
- virtual bool [on\\_trigger\\_event](#) ( **std::string** event, [json](#) meta, [cloud::time](#) time)
- virtual bool [on\\_set\\_log\\_enable](#) (bool bEnable)
- virtual bool [on\\_set\\_activity](#) (bool bEnable)
- virtual void [on\\_registered](#) (const **std::string** &sid)

### 10.18.1 Detailed Description

VXG Cloud agent manager class.

Definition at line 28 of file manager.h.

### 10.18.2 Member Typedef Documentation



### 10.18.2.1 ptr

```
typedef std::shared_ptr<manager> vxg::cloud::agent::manager::ptr
```

shared\_ptr to manager object

Definition at line 478 of file manager.h.

## 10.18.3 Member Function Documentation

### 10.18.3.1 \_\_notify\_record\_event()

```
bool vxg::cloud::agent::manager::__notify_record_event (
    std::string stream_id,
    bool on ) [protected]
```

### 10.18.3.2 \_\_trigger\_periodic\_event()

```
void vxg::cloud::agent::manager::__trigger_periodic_event (
    const proto::event_config & event_conf ) [protected]
```

### 10.18.3.3 \_append\_internal\_custom\_events()

```
void vxg::cloud::agent::manager::_append_internal_custom_events (
    proto::events_config & config ) [protected]
```

### 10.18.3.4 \_cancel\_direct\_uploads\_by\_ticket()

```
bool vxg::cloud::agent::manager::_cancel_direct_uploads_by_ticket (
    std::string ticket ) [protected]
```

### 10.18.3.5 \_cancel\_periodic\_event()

```
void vxg::cloud::agent::manager::_cancel_periodic_event (
    const proto::event_config & event_conf ) [protected]
```

### 10.18.3.6 \_cancel\_periodic\_events()

```
void vxg::cloud::agent::manager::_cancel_periodic_events (
    const proto::events_config & events_conf ) [protected]
```

### 10.18.3.7 \_current\_delivery\_mode()

```
event_state::stream_delivery_mode vxg::cloud::agent::manager::_current_delivery_mode ( ) [protected]
```

### 10.18.3.8 \_handle\_stream\_stateful\_event()

```
bool vxg::cloud::agent::manager::_handle_stream_stateful_event (
    proto::event_object & event,
    event_state::stream_delivery_mode delivery_mode ) [protected]
```

### 10.18.3.9 \_handle\_stream\_stateless\_event()

```
bool vxg::cloud::agent::manager::_handle_stream_stateless_event (
    proto::event_object & event,
    event_state::stream_delivery_mode delivery_mode ) [protected]
```

### 10.18.3.10 \_init\_events\_states()

```
void vxg::cloud::agent::manager::_init_events_states (
    const proto::events_config & config ) [protected]
```

### 10.18.3.11 \_load\_events\_configs()

```
void vxg::cloud::agent::manager::_load_events_configs (
    proto::events_config & config ) [protected]
```

### 10.18.3.12 \_lookup\_event\_stream()

```
event_stream::ptr vxg::cloud::agent::manager::_lookup_event_stream (
    const std::string & name ) [protected]
```

#### 10.18.3.13 \_lookup\_event\_stream\_by\_event()

```
event_stream::ptr vxg::cloud::agent::manager::_lookup_event_stream_by_event (
    const proto::event_object & event ) [protected]
```

#### 10.18.3.14 \_request\_direct\_upload\_snapshot()

```
bool vxg::cloud::agent::manager::_request_direct_upload_snapshot (
    proto::get_direct_upload_url direct_upload ) [protected]
```

#### 10.18.3.15 \_request\_direct\_upload\_video()

```
bool vxg::cloud::agent::manager::_request_direct_upload_video (
    proto::get_direct_upload_url direct_upload ) [protected]
```

#### 10.18.3.16 \_schedule\_direct\_upload()

```
bool vxg::cloud::agent::manager::_schedule_direct_upload (
    proto::get_direct_upload_url get_upload_url ) [protected]
```

#### 10.18.3.17 \_schedule\_periodic\_event()

```
void vxg::cloud::agent::manager::_schedule_periodic_event (
    const proto::event_config & event_conf ) [protected]
```

#### 10.18.3.18 \_schedule\_periodic\_events()

```
void vxg::cloud::agent::manager::_schedule_periodic_events (
    proto::events_config & events_conf ) [protected]
```

#### 10.18.3.19 \_stop\_all\_event\_streams()

```
void vxg::cloud::agent::manager::_stop_all_event_streams ( ) [protected]
```

**10.18.3.20 \_stop\_all\_streams()**

```
void vxg::cloud::agent::manager::_stop_all_streams ( ) [protected]
```

Streams helpers.

**10.18.3.21 \_stop\_stream()**

```
void vxg::cloud::agent::manager::_stop_stream (
    agent::media::stream::ptr s ) [protected]
```

**10.18.3.22 \_update\_direct\_upload\_queue\_latency()**

```
void vxg::cloud::agent::manager::_update_direct_upload_queue_latency ( ) [protected]
```

**10.18.3.23 \_update\_event\_stream\_configs()**

```
bool vxg::cloud::agent::manager::_update_event_stream_configs (
    const std::string & stream_name,
    const std::vector< proto::event_config > & new_configs ) [protected]
```

**10.18.3.24 \_update\_events\_configs()**

```
bool vxg::cloud::agent::manager::_update_events_configs (
    const std::vector< proto::event_config > & new_configs,
    std::vector< proto::event_config > & dest_configs ) [protected]
```

**10.18.3.25 \_update\_storage\_status()**

```
bool vxg::cloud::agent::manager::_update_storage_status ( ) [protected]
```

**10.18.3.26 create()**

```
static manager::ptr vxg::cloud::agent::manager::create (
    callback::ptr callback,
    proto::access_token::ptr access_token,
    std::vector< agent::media::stream::ptr > media_streams,
    std::vector< event_stream::ptr > event_streams = std::vector< event_stream::ptr > (0)
) [static]
```

Create manager object.

## Parameters

in	<i>callback</i>	cm::callback object, should not be null
in	<i>access_token</i>	VXG Cloud access token
in	<i>media_streams</i>	List of <b>std::shared_ptr</b> to base_stream derived objects. Should have at least one element. base_stream is abstract class so you need to declare you own class derived from the base_stream or use one of the provided classes (rtsp_stream,...), basically each stream is for example one rtsp stream provided by the device. Each media stream device has should be represented as a separate base_stream derived object, currently only two streams per device are supported by the VXG Cloud.
in	<i>event_streams</i>	List of <b>event_stream::ptr</b> , can be empty. <b>event_stream</b> is abstract class so final implementation should use own class derived from the <b>event_stream</b> .

## Returns

**manager::ptr**

## 10.18.3.27 direct\_upload\_sync\_cb()

```
bool vxg::cloud::agent::manager::direct_upload_sync_cb ( ) [protected]
```

## 10.18.3.28 handle\_event\_meta\_file()

```
bool vxg::cloud::agent::manager::handle_event_meta_file (
    proto::event_object & event ) [protected]
```

## 10.18.3.29 handle\_event\_snapshot()

```
bool vxg::cloud::agent::manager::handle_event_snapshot (
    proto::event_object & event ) [protected]
```

## 10.18.3.30 handle\_stream\_event()

```
bool vxg::cloud::agent::manager::handle_stream_event (
    proto::event_object & event ) [protected]
```

#### 10.18.3.31 lookup\_stream()

```
agent::media::stream::ptr vxg::cloud::agent::manager::lookup_stream (
    std::string name ) [protected]
```

#### 10.18.3.32 notify\_event()

```
bool vxg::cloud::agent::manager::notify_event (
    proto::event_object event ) [protected]
```

#### 10.18.3.33 on\_audio\_file\_play()

```
virtual bool vxg::cloud::agent::manager::on_audio_file_play (
    std::string url ) [protected], [virtual]
```

#### 10.18.3.34 on\_cam\_memorycard\_recording()

```
virtual bool vxg::cloud::agent::manager::on_cam_memorycard_recording (
    const std::string & stream_id,
    bool enabled ) [protected], [virtual]
```

#### 10.18.3.35 on\_cam\_memorycard\_synchronize()

```
virtual bool vxg::cloud::agent::manager::on_cam_memorycard_synchronize (
    proto::command::cam_memorycard_synchronize_status & synchronize_status,
    vxg::cloud::time start,
    vxg::cloud::time end ) [protected], [virtual]
```

#### 10.18.3.36 on\_cam\_memorycard\_synchronize\_cancel()

```
virtual bool vxg::cloud::agent::manager::on_cam_memorycard_synchronize_cancel (
    const std::string & request_id ) [protected], [virtual]
```

### 10.18.3.37 on\_cam\_ptz()

```
virtual bool vxg::cloud::agent::manager::on_cam_ptz (
    proto::ptz_command command ) [protected], [virtual]
```

### 10.18.3.38 on\_cam\_ptz\_preset()

```
virtual bool vxg::cloud::agent::manager::on_cam_ptz_preset (
    proto::ptz_preset & preset_op ) [protected], [virtual]
```

### 10.18.3.39 on\_cam\_upgrade\_firmware()

```
virtual bool vxg::cloud::agent::manager::on_cam_upgrade_firmware (
    std::string url ) [protected], [virtual]
```

### 10.18.3.40 on\_closed()

```
virtual void vxg::cloud::agent::manager::on_closed (
    int error,
    proto::command::bye_reason reason ) [protected], [virtual]
```

### 10.18.3.41 on\_direct\_upload\_url()

```
virtual bool vxg::cloud::agent::manager::on_direct_upload_url (
    const proto::command::direct_upload_url_base & direct_upload,
    int event_id,
    int ref_id ) [protected], [virtual]
```

### 10.18.3.42 on\_get\_cam\_audio\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_cam_audio_config (
    proto::audio_config & config ) [protected], [virtual]
```

#### 10.18.3.43 on\_get\_cam\_events\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_cam_events_config (
    proto::events_config & config ) [protected], [virtual]
```

#### 10.18.3.44 on\_get\_cam\_memorycard\_timeline()

```
virtual bool vxg::cloud::agent::manager::on_get_cam_memorycard_timeline (
    proto::command::cam_memorycard_timeline & timeline ) [protected], [virtual]
```

#### 10.18.3.45 on\_get\_cam\_video\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_cam_video_config (
    proto::video_config & config ) [protected], [virtual]
```

#### 10.18.3.46 on\_get\_log()

```
virtual bool vxg::cloud::agent::manager::on_get_log ( ) [protected], [virtual]
```

#### 10.18.3.47 on\_get\_motion\_detection\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_motion_detection_config (
    proto::motion_detection_config & config ) [protected], [virtual]
```

#### 10.18.3.48 on\_get\_osd\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_osd_config (
    proto::osd_config & config ) [protected], [virtual]
```

#### 10.18.3.49 on\_get\_ptz\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_ptz_config (
    proto::ptz_config & config ) [protected], [virtual]
```



#### 10.18.3.50 on\_get\_stream\_by\_event()

```
virtual bool vxg::cloud::agent::manager::on_get_stream_by_event (
    proto::stream_by_event_config & conf ) [protected], [virtual]
```

#### 10.18.3.51 on\_get\_stream\_caps()

```
virtual bool vxg::cloud::agent::manager::on_get_stream_caps (
    proto::stream_caps & caps ) [protected], [virtual]
```

#### 10.18.3.52 on\_get\_stream\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_stream_config (
    proto::stream_config & config ) [protected], [virtual]
```

#### 10.18.3.53 on\_get\_supported\_streams()

```
virtual bool vxg::cloud::agent::manager::on_get_supported_streams (
    proto::supported_streams_config & supportedStreamsConfig ) [protected], [virtual]
```

#### 10.18.3.54 on\_get\_timezone()

```
virtual bool vxg::cloud::agent::manager::on_get_timezone (
    std::string & timezone ) [protected], [virtual]
```

#### 10.18.3.55 on\_get\_wifi\_config()

```
virtual bool vxg::cloud::agent::manager::on_get_wifi_config (
    proto::wifi_config & config ) [protected], [virtual]
```

#### 10.18.3.56 on\_prepared()

```
virtual void vxg::cloud::agent::manager::on_prepared ( ) [protected], [virtual]
```

#### 10.18.3.57 on\_raw\_message()

```
virtual bool vxg::cloud::agent::manager::on_raw_message (
    std::string client_id,
    std::string & data ) [protected], [virtual]
```

#### 10.18.3.58 on\_registered()

```
virtual void vxg::cloud::agent::manager::on_registered (
    const std::string & sid ) [protected], [virtual]
```

#### 10.18.3.59 on\_set\_activity()

```
virtual bool vxg::cloud::agent::manager::on_set_activity (
    bool bEnable ) [protected], [virtual]
```

#### 10.18.3.60 on\_set\_cam\_audio\_config()

```
virtual bool vxg::cloud::agent::manager::on_set_cam_audio_config (
    const proto::audio_config & config ) [protected], [virtual]
```

#### 10.18.3.61 on\_set\_cam\_events\_config()

```
virtual bool vxg::cloud::agent::manager::on_set_cam_events_config (
    const proto::events_config & config ) [protected], [virtual]
```

#### 10.18.3.62 on\_set\_cam\_video\_config()

```
virtual bool vxg::cloud::agent::manager::on_set_cam_video_config (
    const proto::video_config & config ) [protected], [virtual]
```

#### 10.18.3.63 on\_set\_log\_enable()

```
virtual bool vxg::cloud::agent::manager::on_set_log_enable (
    bool bEnable ) [protected], [virtual]
```

#### 10.18.3.64 on\_set\_motion\_detection\_config()

```
virtual bool vxg::cloud::agent::manager::on_set_motion_detection_config (
    const proto::motion_detection_config & config ) [protected], [virtual]
```

#### 10.18.3.65 on\_set\_osd\_config()

```
virtual bool vxg::cloud::agent::manager::on_set_osd_config (
    const proto::osd_config & config ) [protected], [virtual]
```

#### 10.18.3.66 on\_set\_periodic\_events()

```
void vxg::cloud::agent::manager::on_set_periodic_events (
    const char * name,
    int period,
    bool active ) [protected]
```

#### 10.18.3.67 on\_set\_stream\_by\_event()

```
virtual bool vxg::cloud::agent::manager::on_set_stream_by_event (
    proto::stream_by_event_config conf ) [protected], [virtual]
```

#### 10.18.3.68 on\_set\_stream\_config()

```
virtual bool vxg::cloud::agent::manager::on_set_stream_config (
    const proto::stream_config & config ) [protected], [virtual]
```

#### 10.18.3.69 on\_set\_timezone()

```
virtual bool vxg::cloud::agent::manager::on_set_timezone (
    std::string timezone ) [protected], [virtual]
```

#### 10.18.3.70 on\_set\_wifi\_config()

```
virtual bool vxg::cloud::agent::manager::on_set_wifi_config (
    const proto::wifi_network & config ) [protected], [virtual]
```

#### 10.18.3.71 on\_start\_backward()

```
virtual bool vxg::cloud::agent::manager::on_start_backward (
    std::string & url ) [protected], [virtual]
```

#### 10.18.3.72 on\_stop\_backward()

```
virtual bool vxg::cloud::agent::manager::on_stop_backward (
    std::string & url ) [protected], [virtual]
```

#### 10.18.3.73 on\_stream\_start()

```
virtual bool vxg::cloud::agent::manager::on_stream_start (
    const std::string & streamId,
    int publishSessionID,
    proto::stream_reason reason ) [protected], [virtual]
```

#### 10.18.3.74 on\_stream\_stop()

```
virtual bool vxg::cloud::agent::manager::on_stream_stop (
    const std::string & streamId,
    proto::stream_reason reason ) [protected], [virtual]
```

#### 10.18.3.75 on\_trigger\_event()

```
virtual bool vxg::cloud::agent::manager::on_trigger_event (
    std::string event,
    json meta,
    cloud::time time ) [protected], [virtual]
```

#### 10.18.3.76 on\_update\_preview()

```
virtual bool vxg::cloud::agent::manager::on_update_preview (
    std::string url ) [protected], [virtual]
```

### 10.18.3.77 start()

```
bool vxg::cloud::agent::manager::start ( )
```

Start internal workflow, this is the main function which starts all internal threads and connections.

#### Returns

true started  
false start failed

### 10.18.3.78 stop()

```
void vxg::cloud::agent::manager::stop ( )
```

Stop manager, disconnect from the VXG Cloud.

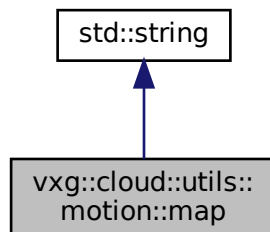
The documentation for this class was generated from the following file:

- [manager.h](#)

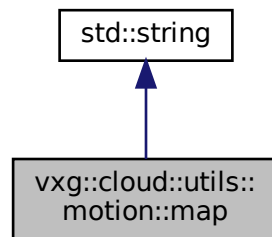
## 10.19 vxg::cloud::utils::motion::map Struct Reference

```
#include <utils/utils.h>
```

Inheritance diagram for vxg::cloud::utils::motion::map:



Collaboration diagram for vxg::cloud::utils::motion::map:



## Public Member Functions

- `map()`
- `map(const map &motionMap)`
- `map & operator= (const std::string &motionMap)`

## Static Public Member Functions

- static `std::string pack` (const `std::string` &unpackedGrid)
- static `std::string unpack` (const `std::string` &packedMap, `size_t` outputLen)

### 10.19.1 Detailed Description

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

### 10.19.2 Constructor & Destructor Documentation

#### 10.19.2.1 map() [1/2]

```
vxg::cloud::utils::motion::map::map ( ) [inline], [explicit]
```

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

#### 10.19.2.2 map() [2/2]

```
vxg::cloud::utils::motion::map::map (
    const map & motionMap ) [inline]
```

Definition at line 129 of file `utils.h`.

### 10.19.3 Member Function Documentation

#### 10.19.3.1 operator=()

```
map& vxg::cloud::utils::motion::map::operator= (
    const std::string & motionMap ) [inline]
```

Definition at line 131 of file utils.h.

#### 10.19.3.2 pack()

```
static std::string vxg::cloud::utils::motion::map::pack (
    const std::string & unpackedGrid ) [static]
```

#### 10.19.3.3 unpack()

```
static std::string vxg::cloud::utils::motion::map::unpack (
    const std::string & packedMap,
    size_t outputLen ) [static]
```

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

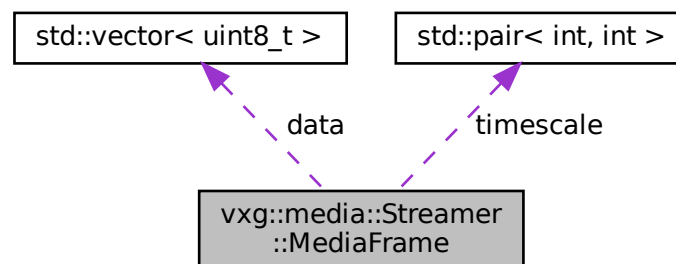
- [utils.h](#)

## 10.20 vxg::media::Streamer::MediaFrame Struct Reference

Media frame container.

```
#include <streamer/base_streamer.h>
```

Collaboration diagram for vxg::media::Streamer::MediaFrame:



## Public Member Functions

- bool `operator<` (const `MediaFrame` &rv)  
*Two frames comparison using timestamps.*

## Data Fields

- `std::vector< uint8_t >` `data`  
*Media frame data.*
- `size_t` `len`  
*Media frame data length.*
- `int64_t` `pts`  
*Media frame timestamp in timescale that corresponds to timescale.*
- `int64_t` `dts`  
*Media frame decoding timestamp in timescale that corresponds to timescale.*
- `int64_t` `duration`  
*Media frame duration if needed.*
- bool `is_key`  
*Is key frame flag.*
- `MediaType` `type`  
*Media frame type.*
- `std::pair< int, int >` `timescale`  
*Timescale of pts and duration. ex. : 1/90000, 1/1000 etc.*
- `int64_t` `time_realtime`  
*Real time if available from source, for ex.*

## Static Public Attributes

- static constexpr `int64_t` `NOPTS`

### 10.20.1 Detailed Description

Media frame container.

Definition at line 403 of file `base_streamer.h`.

### 10.20.2 Member Function Documentation

#### 10.20.2.1 `operator<()`

```
bool vxg::media::Streamer::MediaFrame::operator< (
    const MediaFrame & rv ) [inline]
```

Two frames comparison using timestamps.



#### Parameters

<i>rv</i>	Right value
-----------	-------------

#### Returns

true

false

Definition at line 421 of file base\_streamer.h.

### 10.20.3 Field Documentation

#### 10.20.3.1 data

```
std::vector<uint8_t> vxg::media::Streamer::MediaFrame::data
```

Media frame data.

Definition at line 426 of file base\_streamer.h.

#### 10.20.3.2 dts

```
int64_t vxg::media::Streamer::MediaFrame::dts
```

Media frame decoding timestamp in timescale that corresponds to timescale.

Definition at line 433 of file base\_streamer.h.

#### 10.20.3.3 duration

```
int64_t vxg::media::Streamer::MediaFrame::duration
```

Media frame duration if needed.

Definition at line 435 of file base\_streamer.h.

#### 10.20.3.4 is\_key

```
bool vxg::media::Streamer::MediaFrame::is_key
```

Is key frame flag.

Definition at line 437 of file base\_streamer.h.

#### 10.20.3.5 len

```
size_t vxg::media::Streamer::MediaFrame::len
```

Media frame data length.

Definition at line 428 of file base\_streamer.h.

#### 10.20.3.6 NOPTS

```
constexpr int64_t vxg::media::Streamer::MediaFrame::NOPTS [static], [constexpr]
```

Definition at line 423 of file base\_streamer.h.

#### 10.20.3.7 pts

```
int64_t vxg::media::Streamer::MediaFrame::pts
```

Media frame timestamp in timescale that corresponds to timescale.

Definition at line 430 of file base\_streamer.h.

#### 10.20.3.8 time\_realtime

```
int64_t vxg::media::Streamer::MediaFrame::time_realtime
```

Real time if available from source, for ex.

pts based on NTP time from RTCP SR

Definition at line 444 of file base\_streamer.h.

### 10.20.3.9 timescale

`std::pair<int, int> vxg::media::Streamer::MediaFrame::timescale`

Timescale of pts and duration. ex. : 1/90000, 1/1000 etc.

Definition at line 441 of file base\_streamer.h.

### 10.20.3.10 type

`MediaType vxg::media::Streamer::MediaFrame::type`

Media frame type.

Definition at line 439 of file base\_streamer.h.

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

- [base\\_streamer.h](#)

## 10.21 vxg::cloud::agent::proto::motion\_detection\_caps Struct Reference

Motion detection capabilities camera capabilities that limit possible motion detection configuration.

```
#include <agent-proto/objects/caps.h>
```

### Data Fields

- `size_t max_regions`  
*Mandatory: supported number of motion regions.*
- `motion_sensitivity sensitivity`  
*Mandatory: ("region", "frame"), default "region"; indicates if sensitivity can be set for region or for whole frame only.*
- `motion_region_shape region_shape`  
*Mandatory: ("rect", "any"), default "any"; specifies limitation of region shape.*

### 10.21.1 Detailed Description

Motion detection capabilities camera capabilities that limit possible motion detection configuration.

Definition at line 336 of file caps.h.

### 10.21.2 Field Documentation

### 10.21.2.1 max\_regions

```
size_t vxg::cloud::agent::proto::motion_detection_caps::max_regions
```

Mandatory: supported number of motion regions.

Definition at line 339 of file caps.h.

### 10.21.2.2 region\_shape

```
motion_region_shape vxg::cloud::agent::proto::motion_detection_caps::region_shape
```

Mandatory: ("rect", "any"), default "any"; specifies limitation of region shape.

Definition at line 348 of file caps.h.

### 10.21.2.3 sensitivity

```
motion_sensitivity vxg::cloud::agent::proto::motion_detection_caps::sensitivity
```

Mandatory: ("region", "frame"), default "region"; indicates if sensitivity can be set for region or for whole frame only.

Definition at line 344 of file caps.h.

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

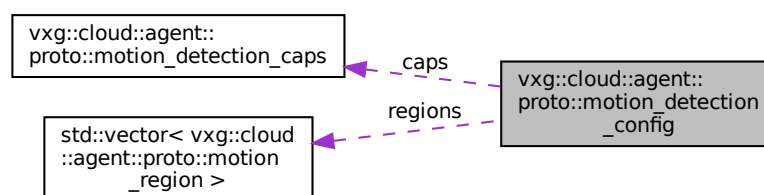
- [caps.h](#)

## 10.22 vxg::cloud::agent::proto::motion\_detection\_config Struct Reference

Motion detection config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::motion\_detection\_config:



## Data Fields

- int [columns](#)  
*Mandatory.*
- int [rows](#)  
*Mandatory.*
- [motion\\_detection\\_caps](#) caps  
*Mandatory for CM => SRV (reply to 'get\_motion\_detection') camera capabilities that limit possible motion detection configuration.*
- **std::vector**< [motion\\_region](#) > [regions](#)  
*Mandatory List of motion regions.*

### 10.22.1 Detailed Description

Motion detection config.

Definition at line 280 of file config.h.

### 10.22.2 Field Documentation

#### 10.22.2.1 caps

[motion\\_detection\\_caps](#) vxg::cloud::agent::proto::motion\_detection\_config::caps

Mandatory for CM => SRV (reply to 'get\_motion\_detection') camera capabilities that limit possible motion detection configuration.

Definition at line 289 of file config.h.

#### 10.22.2.2 columns

int vxg::cloud::agent::proto::motion\_detection\_config::columns

Mandatory.

Definition at line 283 of file config.h.

#### 10.22.2.3 regions

**std::vector**<[motion\\_region](#)> vxg::cloud::agent::proto::motion\_detection\_config::regions

Mandatory List of motion regions.

Definition at line 292 of file config.h.

#### 10.22.2.4 rows

```
int vxg::cloud::agent::proto::motion_detection_config::rows
```

Mandatory.

Definition at line 286 of file config.h.

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

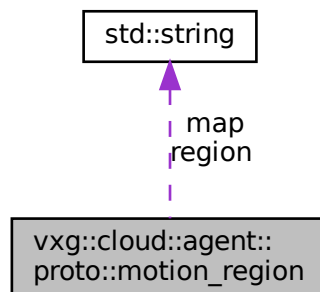
- [config.h](#)

### 10.23 vxg::cloud::agent::proto::motion\_region Struct Reference

Motion detection related structs.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::motion\_region:



#### Data Fields

- **std::string** [region](#)  
Mandatory: name of region if supported by camera.
- **std::string** [map](#)  
Mandatory: String is packed with Apple Packbit algorithm and after that encoded with Base64.
- **size\_t** [sensitivity](#)  
Mandatory: range 0-100; 0 - minimal sensitivity.
- **bool** [enabled](#)  
Mandatory: indicates that motion detection is enabled for the region.

### 10.23.1 Detailed Description

Motion detection related structs.

Motion region

Definition at line 243 of file config.h.

### 10.23.2 Field Documentation

#### 10.23.2.1 enabled

```
bool vxg::cloud::agent::proto::motion_region::enabled
```

Mandatory: indicates that motion detection is enabled for the region.

Definition at line 265 of file config.h.

#### 10.23.2.2 map

```
std::string vxg::cloud::agent::proto::motion_region::map
```

Mandatory: String is packed with Apple Packbit algorithm and after that encoded with Base64.

Bitstring where “1” denotes an active cell and a “0” an inactive cell. The first cell is in the upper left corner. Then the cell order goes first from left to right and then from up to down. If the number of cells is not a multiple of 8 the last byte is padded with zeros.

Definition at line 255 of file config.h.

#### 10.23.2.3 region

```
std::string vxg::cloud::agent::proto::motion_region::region
```

Mandatory: name of region if supported by camera.

Definition at line 246 of file config.h.

### 10.23.2.4 sensitivity

```
size_t vxg::cloud::agent::proto::motion_region::sensitivity
```

Mandatory: range 0-100; 0 - minimal sensitivity.

If sensitivity is supported only for whole frame, the same value should be used for all regions.

Definition at line 261 of file config.h.

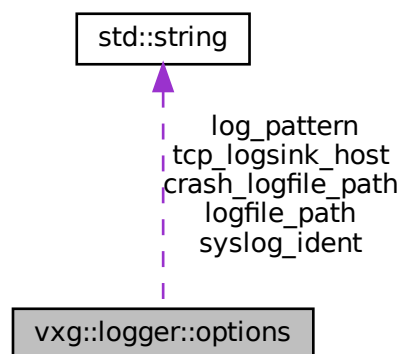
The documentation for this struct was generated from the following file:

- [config.h](#)

## 10.24 vxg::logger::options Struct Reference

```
#include <utils/logging.h>
```

Collaboration diagram for vxg::logger::options:



### Data Fields

- **std::string** [log\\_pattern](#)
- **std::string** [logfile\\_path](#)
- **size\_t** [logfile\\_max\\_size](#)
- **size\_t** [logfile\\_max\\_files](#)
- **std::string** [crash\\_logfile\\_path](#)
- **std::string** [syslog\\_ident](#)
- [loglevel](#) [default\\_loglevel](#)
- **bool** [tcp\\_logsink\\_enabled](#)
- **std::string** [tcp\\_logsink\\_host](#)
- **uint16\_t** [tcp\\_logsink\\_port](#)



## 10.24.1 Detailed Description

Definition at line 35 of file logging.h.

## 10.24.2 Field Documentation

### 10.24.2.1 crash\_logfile\_path

```
std::string vxg::logger::options::crash_logfile_path
```

Definition at line 41 of file logging.h.

### 10.24.2.2 default\_loglevel

```
loglevel vxg::logger::options::default_loglevel
```

Definition at line 43 of file logging.h.

### 10.24.2.3 log\_pattern

```
std::string vxg::logger::options::log_pattern
```

Definition at line 36 of file logging.h.

### 10.24.2.4 logfile\_max\_files

```
size_t vxg::logger::options::logfile_max_files
```

Definition at line 40 of file logging.h.

### 10.24.2.5 logfile\_max\_size

```
size_t vxg::logger::options::logfile_max_size
```

Definition at line 39 of file logging.h.

#### 10.24.2.6 logfile\_path

```
std::string vxg::logger::options::logfile_path
```

Definition at line 38 of file logging.h.

#### 10.24.2.7 syslog\_ident

```
std::string vxg::logger::options::syslog_ident
```

Definition at line 42 of file logging.h.

#### 10.24.2.8 tcp\_logsink\_enabled

```
bool vxg::logger::options::tcp_logsink_enabled
```

Definition at line 44 of file logging.h.

#### 10.24.2.9 tcp\_logsink\_host

```
std::string vxg::logger::options::tcp_logsink_host
```

Definition at line 45 of file logging.h.

#### 10.24.2.10 tcp\_logsink\_port

```
uint16_t vxg::logger::options::tcp_logsink_port
```

Definition at line 46 of file logging.h.

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

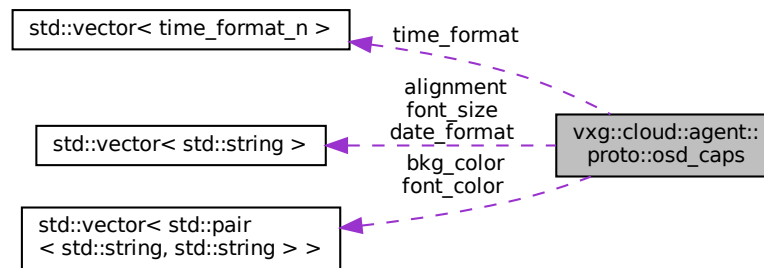
- [logging.h](#)

## 10.25 vxg::cloud::agent::proto::osd\_caps Struct Reference

OSD capabilities.

```
#include <agent-proto/objects/caps.h>
```

Collaboration diagram for vxg::cloud::agent::proto::osd\_caps:



### Data Fields

- bool [system\\_id](#)  
*system\_id*: bool, True when OSD supports separate system\_id enabling/disabling
- bool [system\\_id\\_text](#)  
*system\_id\_text*: bool, True when OSD supports separate system\_id customization
- bool [time](#)  
*time*: bool, True when OSD supports separate time enabling/disabling
- **std::vector**< [time\\_format\\_n](#) > [time\\_format](#)  
*time\_format*: list of string, supported time formats.
- bool [date](#)  
*date*: bool, True when OSD supports separate date enabling/disabling
- **std::vector**< **std::string** > [date\\_format](#)  
*date\_format*: list of string, supported date formats.
- **std::vector**< **std::string** > [font\\_size](#)  
*font\_size*: list of string, describes supported font sizes.
- **std::vector**< **std::pair**< **std::string**, **std::string** > > [font\\_color](#)  
*font\_color*: list of pairs [string (name), optional string (value)], predefined set of possible font colors.
- **std::vector**< **std::pair**< **std::string**, **std::string** > > [bkg\\_color](#)  
*bkg\_color*: list of pairs [string (name), optional string (value)], predefined set of possible background colors.
- bool [bkg\\_transp](#)  
*bkg\_transp*: bool, True when OSD supports background transparency
- **std::vector**< **std::string** > [alignment](#)  
*alignment*: list of strings, supported OSD positions.

### 10.25.1 Detailed Description

OSD capabilities.

Definition at line 615 of file caps.h.

## 10.25.2 Field Documentation

### 10.25.2.1 alignment

```
std::vector< std::string> vxg::cloud::agent::proto::osd_caps::alignment
```

alignment: list of strings, supported OSD positions.

Empty list means – position can't be changed. Example: ["UpperLeft", "UpperRight", "LowerLeft", "LowerRight"]

Definition at line 654 of file caps.h.

### 10.25.2.2 bkg\_color

```
std::vector< std::pair< std::string, std::string> > vxg::cloud::agent::proto::osd_caps↵  
::bkg_color
```

bkg\_color: list of pairs [string (name), optional string (value)], predefined set of possible background colors.

Empty list means – color selection is not supported. Optioanal value is a RGB color code in HEX. Example: [{"↵  
Black", "000000"}]

Definition at line 648 of file caps.h.

### 10.25.2.3 bkg\_transp

```
bool vxg::cloud::agent::proto::osd_caps::bkg_transp
```

bkg\_transp: bool, True when OSD supports background transparency

Definition at line 650 of file caps.h.

### 10.25.2.4 date

```
bool vxg::cloud::agent::proto::osd_caps::date
```

date: bool, True when OSD supports separate date enabling/disabling

Definition at line 629 of file caps.h.

### 10.25.2.5 date\_format

```
std::vector< std::string> vxg::cloud::agent::proto::osd_caps::date_format
```

date\_format: list of string, supported date formats.

Empty list means – date format selection is not supported. Example: ["YYYY-MM-DD", "MM-DD-YYYY", "DD-MM-YYYY", "YYYY/MM/DD", "MM/DD/YYYY2", "DD/MM/YYYY"]

Definition at line 633 of file caps.h.

### 10.25.2.6 font\_color

```
std::vector< std::pair< std::string, std::string> > vxg::cloud::agent::proto::osd_caps←  
::font_color
```

font\_color: list of pairs [string (name), optional string (value)], predefined set of possible font colors.

Empty list means – color selection is not supported. Optioanal value is a RGB color code in HEX. Example: [{"↵  
Orange", "FF9C00"}]

Definition at line 642 of file caps.h.

### 10.25.2.7 font\_size

```
std::vector< std::string> vxg::cloud::agent::proto::osd_caps::font_size
```

font\_size: list of string, describes supported font sizes.

Empty list means – font size format selection is not supported. Examples: ["16", "32", "48", "64", "auto"] or ["Small", "Normal", "Big"]

Definition at line 637 of file caps.h.

### 10.25.2.8 system\_id

```
bool vxg::cloud::agent::proto::osd_caps::system_id
```

system\_id: bool, True when OSD supports separate system\_id enabling/disabling

Definition at line 618 of file caps.h.

### 10.25.2.9 system\_id\_text

```
bool vxg::cloud::agent::proto::osd_caps::system_id_text
```

system\_id\_text: bool, True when OSD supports separate system\_id customization

Definition at line 621 of file caps.h.

### 10.25.2.10 time

```
bool vxg::cloud::agent::proto::osd_caps::time
```

time: bool, True when OSD supports separate time enabling/disabling

Definition at line 623 of file caps.h.

### 10.25.2.11 time\_format

```
std::vector<time_format_n> vxg::cloud::agent::proto::osd_caps::time_format
```

time\_format: list of string, supported time formats.

Empty list means – time format selection is not supported. Example: ["12h", "24h"]

Definition at line 627 of file caps.h.

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

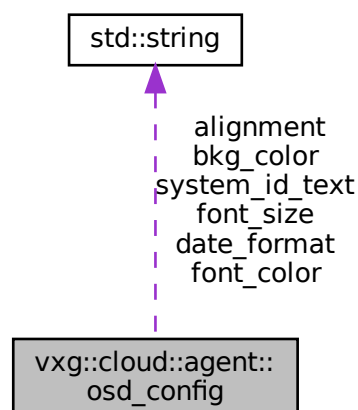
- [caps.h](#)

## 10.26 vxg::cloud::agent::osd\_config Struct Reference

OSD config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::osd\_config:



## Data Fields

- bool [system\\_id](#)  
*system\_id: optional bool, enable/disable static part of OSD*
- **std::string** [system\\_id\\_text](#)  
*system\_id\_text: optional string, a static content of OSD*
- bool [time](#)  
*time: optional bool, enable/disable time part of OSD*
- time\_format\_n [time\\_format](#)  
*time\_format: optional string, one of predefined values from the time\_format\_n, should be included in caps.*
- bool [date](#)  
*date: optional bool, enable/disable date part of OSD*
- **std::string** [date\\_format](#)  
*date\_format: optional string, one of predefined values from caps*
- **std::string** [font\\_size](#)  
*font\_size: optional string, one of predefined font sizes from caps*
- **std::string** [font\\_color](#)  
*font\_color: optional string, name of one of predefined font colors from caps*
- **std::string** [bkg\\_color](#)  
*bkg\_color: optional string, name of one of predefined background colors from caps*
- bool [bkg\\_transp](#)  
*bkg\_transp: optional bool, enable/disable OSD background transparency*
- **std::string** [alignment](#)  
*alignment: optional string, one of predefined positions from caps*
- osd\_caps [caps](#)  
*OSD capabilities of the device.*

### 10.26.1 Detailed Description

OSD config.

On Screen Display configuration object.

Definition at line 1137 of file config.h.

### 10.26.2 Field Documentation

#### 10.26.2.1 alignment

```
std::string vxg::cloud::agent::osd_config::alignment
```

alignment: optional string, one of predefined positions from caps

Definition at line 1168 of file config.h.

### 10.26.2.2 bkg\_color

```
std::string vxg::cloud::agent::osd_config::bkg_color
```

bkg\_color: optional string, name of one of predefined background colors from caps

Definition at line 1164 of file config.h.

### 10.26.2.3 bkg\_transp

```
bool vxg::cloud::agent::osd_config::bkg_transp
```

bkg\_transp: optional bool, enable/disable OSD background transparency

Definition at line 1166 of file config.h.

### 10.26.2.4 caps

```
osd_caps vxg::cloud::agent::osd_config::caps
```

OSD capabilities of the device.

Definition at line 1171 of file config.h.

### 10.26.2.5 date

```
bool vxg::cloud::agent::osd_config::date
```

date: optional bool, enable/disable date part of OSD

Definition at line 1152 of file config.h.

### 10.26.2.6 date\_format

```
std::string vxg::cloud::agent::osd_config::date_format
```

date\_format: optional string, one of predefined values from caps

Definition at line 1155 of file config.h.



### 10.26.2.7 font\_color

**std::string** vxg::cloud::agent::osd\_config::font\_color

font\_color: optional string, name of one of predefined font colors from caps

Definition at line 1161 of file config.h.

### 10.26.2.8 font\_size

**std::string** vxg::cloud::agent::osd\_config::font\_size

font\_size: optional string, one of predefined font sizes from caps

Definition at line 1158 of file config.h.

### 10.26.2.9 system\_id

**bool** vxg::cloud::agent::osd\_config::system\_id

system\_id: optional bool, enable/disable static part of OSD

Definition at line 1140 of file config.h.

### 10.26.2.10 system\_id\_text

**std::string** vxg::cloud::agent::osd\_config::system\_id\_text

system\_id\_text: optional string, a static content of OSD

Definition at line 1143 of file config.h.

### 10.26.2.11 time

**bool** vxg::cloud::agent::osd\_config::time

time: optional bool, enable/disable time part of OSD

Definition at line 1146 of file config.h.

### 10.26.2.12 time\_format

time\_format\_n vxg::cloud::agent::osd\_config::time\_format

time\_format: optional string, one of predefined values from the time\_format\_n, should be included in caps.

Definition at line 1149 of file config.h.

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

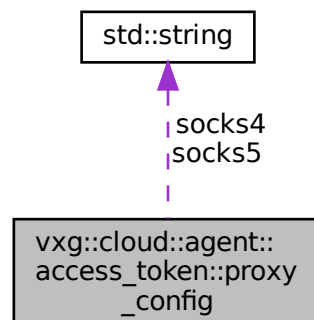
- [config.h](#)

## 10.27 vxg::cloud::agent::access\_token::proxy\_config Struct Reference

Socks proxy settings.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::access\_token::proxy\_config:



### Data Fields

- **std::string** [socks4](#)  
*SOCKS4 proxy uri.*
- **std::string** [socks5](#)  
*SOCKS5 proxy uri, ex. socks5://user:pwd@host:port.*

### 10.27.1 Detailed Description

Socks proxy settings.

Definition at line 1197 of file config.h.

## 10.27.2 Field Documentation

### 10.27.2.1 socks4

**std::string** vxg::cloud::agent::access\_token::proxy\_config::socks4

SOCKS4 proxy uri.

Definition at line 1199 of file config.h.

### 10.27.2.2 socks5

**std::string** vxg::cloud::agent::access\_token::proxy\_config::socks5

SOCKS5 proxy uri, ex. socks5://user:pwd@host:port.

Definition at line 1201 of file config.h.

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

- [config.h](#)

## 10.28 vxg::cloud::agent::ptz\_command Struct Reference

PTZ command.

```
#include <agent-proto/objects/config.h>
```

### Data Fields

- ptz\_action [action](#)  
*action: string, Camera informs server about list of supported actions with 3.30 cam\_ptz\_conf (CM) command*
- int [tm](#)  
*tm: optional int, operation time that allows to make PTZ with specified steps, msec*

### 10.28.1 Detailed Description

PTZ command.

Definition at line 1115 of file config.h.

## 10.28.2 Field Documentation

### 10.28.2.1 action

```
ptz_action vxg::cloud::agent::ptz_command::action
```

action: string, Camera informs server about list of supported actions with 3.30 cam\_ptz\_conf (CM) command

Definition at line 1119 of file config.h.

### 10.28.2.2 tm

```
int vxg::cloud::agent::ptz_command::tm
```

tm: optional int, operation time that allows to make PTZ with specified steps, msec

Definition at line 1123 of file config.h.

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

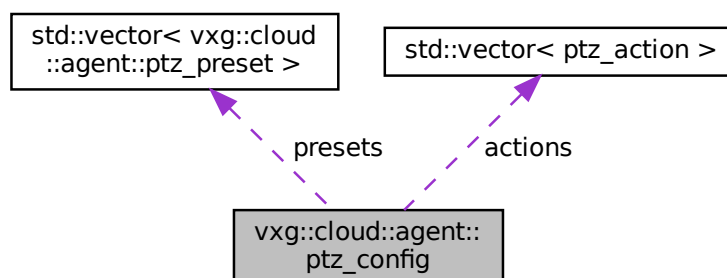
- [config.h](#)

## 10.29 vxg::cloud::agent::ptz\_config Struct Reference

PTZ config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::ptz\_config:



## Data Fields

- **std::vector**< ptz\_action > [actions](#)  
*actions: list of strings, list of supported PTZ actions.*
- int [maximum\\_number\\_of\\_presets](#)  
*maximum\_number\_of\_presets: optional int, max number of supported presets when camera supports.*
- **std::vector**< [ptz\\_preset](#) > [presets](#)  
*presets: optional list of structures [ptz\\_preset](#)*

### 10.29.1 Detailed Description

PTZ config.

Definition at line 1090 of file config.h.

### 10.29.2 Field Documentation

#### 10.29.2.1 actions

```
std::vector<ptz_action> vxg::cloud::agent::ptz_config::actions
```

actions: list of strings, list of supported PTZ actions.

Possible values: "left", "right", "top", "bottom", "zoom\_in", "zoom\_out", "stop". Server sends commands via 3.5 cam\_ptz (SRV)

Definition at line 1094 of file config.h.

#### 10.29.2.2 maximum\_number\_of\_presets

```
int vxg::cloud::agent::ptz_config::maximum_number_of_presets
```

maximum\_number\_of\_presets: optional int, max number of supported presets when camera supports.

Zero value, the missed parameter or missed or empty presets list are interpreted by server as "camera doesn't support PTZ"

Definition at line 1100 of file config.h.

### 10.29.2.3 presets

```
std::vector<ptz_preset> vxg::cloud::agent::ptz_config::presets
```

presets: optional list of structures [ptz\\_preset](#)

Definition at line 1103 of file config.h.

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

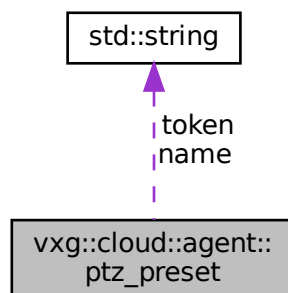
- [config.h](#)

## 10.30 vxg::cloud::agent::ptz\_preset Struct Reference

PTZ preset.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::ptz\_preset:



### Data Fields

- **std::string** [token](#)  
*token: string, an unique token of preset what is used for all operations with preset*
- **std::string** [name](#)  
*name: string, user friendly name of preset*
- **ptz\_preset\_action** [action](#)  
*actions: list of strings, required preset action.*

### 10.30.1 Detailed Description

PTZ preset.

Definition at line 1072 of file config.h.

## 10.30.2 Field Documentation

### 10.30.2.1 action

```
ptz_preset_action vxg::cloud::agent::ptz_preset::action
```

actions: list of strings, required preset action.

Possible values: "create", "delete", "goto", "update"

Definition at line 1081 of file config.h.

### 10.30.2.2 name

```
std::string vxg::cloud::agent::ptz_preset::name
```

name: string, user friendly name of preset

Definition at line 1077 of file config.h.

### 10.30.2.3 token

```
std::string vxg::cloud::agent::ptz_preset::token
```

token: string, an unique token of preset what is used for all operations with preset

Definition at line 1075 of file config.h.

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

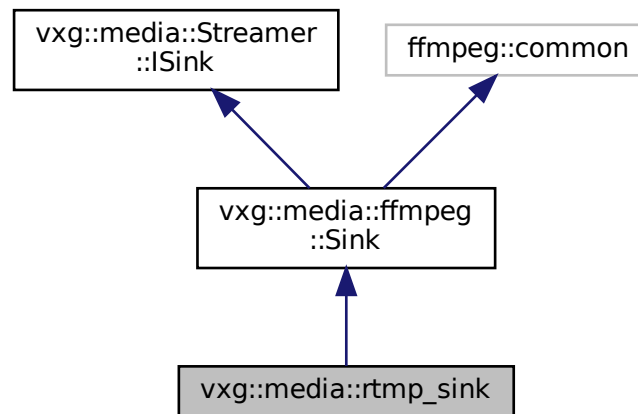
- [config.h](#)

## 10.31 vxg::media::rtmp\_sink Class Reference

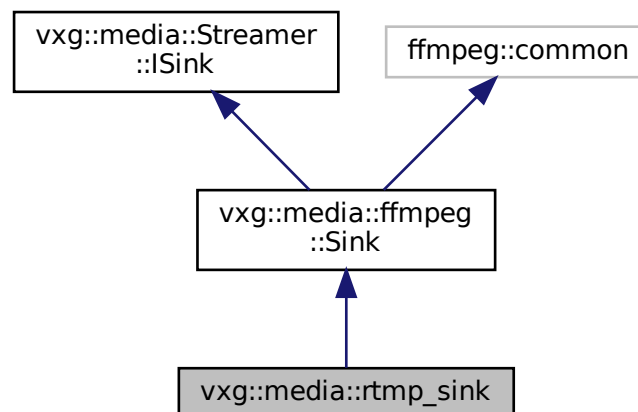
RTMP sink class.

```
#include <streamer/rtmp_sink.h>
```

Inheritance diagram for vxg::media::rtmp\_sink:



Collaboration diagram for vxg::media::rtmp\_sink:





## Public Member Functions

- `rtmp_sink` ( `std::function< void(vxg::media::Streamer::StreamError)>` cb)  
*Construct a new rtmp sink object.*
- virtual bool `init` ( `std::string` url) override  
*Overriden `vxg::media::ffmpeg::Sink::init( std::string, std::string)` "init" method with hidden output ffmpeg format.*
- virtual void `error` ( `Streamer::StreamError` stream\_error) override  
*Media processing error callback, called when `ISink::process` returned false or linked source's `ISource::pullFrame` returned false, or when `ISource::error` was called.*
- virtual `std::string` `name` () override  
*Sink name.*
- virtual bool `droppable` () override  
*If sink of with dropping its media frames.*
- bool `negotiate` ( `std::vector< Streamer::StreamInfo >` streams\_info)  
*Override `negotiate()` for removing all data streams.*

## Additional Inherited Members

### 10.31.1 Detailed Description

RTMP sink class.

Definition at line 13 of file `rtmp_sink.h`.

### 10.31.2 Constructor & Destructor Documentation

#### 10.31.2.1 rtmp\_sink()

```
vxg::media::rtmp_sink::rtmp_sink (
    std::function< void(vxg::media::Streamer::StreamError)> cb ) [inline]
```

Construct a new rtmp sink object.

#### Parameters

<code>in</code>	<code>cb</code>	error callback
-----------------	-----------------	----------------

Definition at line 20 of file `rtmp_sink.h`.

### 10.31.3 Member Function Documentation

### 10.31.3.1 droppable()

```
virtual bool vxg::media::rtmp_sink::droppable ( ) [inline], [override], [virtual]
```

If sink of with dropping its media frames.

#### Returns

true Internal media thread allowed to drop frames if internal media queue is full.  
false No media frames dropping allowed.

Reimplemented from [vxg::media::ffmpeg::Sink](#).

Definition at line 47 of file rtmp\_sink.h.

### 10.31.3.2 error()

```
virtual void vxg::media::rtmp_sink::error (
    Streamer::StreamError error ) [inline], [override], [virtual]
```

Media processing error callback, called when ISink::process returned false or linked source's ISource::pullFrame returned false, or when ISource::error was called.

#### Parameters

<i>error</i>	Error type.
--------------	-------------

Reimplemented from [vxg::media::ffmpeg::Sink](#).

Definition at line 33 of file rtmp\_sink.h.

### 10.31.3.3 init()

```
virtual bool vxg::media::rtmp_sink::init (
    std::string url ) [inline], [override], [virtual]
```

Overriden [vxg::media::ffmpeg::Sink::init\( std::string, std::string\)](#) "init" method with hidden output ffmpeg format.

#### Parameters

<i>url</i>	RTMP url
------------	----------

#### Returns

true On success  
false On failure

Reimplemented from [vxg::media::ffmpeg::Sink](#).

Definition at line 29 of file rtmp\_sink.h.

#### 10.31.3.4 name()

```
virtual std::string vxg::media::rtmp_sink::name ( ) [inline], [override], [virtual]
```

Sink name.

Returns

**std::string**

Reimplemented from [vxg::media::ffmpeg::Sink](#).

Definition at line 45 of file rtmp\_sink.h.

#### 10.31.3.5 negotiate()

```
bool vxg::media::rtmp_sink::negotiate (
    std::vector< Streamer::StreamInfo > streams_info ) [inline], [virtual]
```

Override [negotiate\(\)](#) for removing all data streams.

This is required for preventing buffering inside the ffmpeg muxer, ffmpeg waits for at least one packet for each stream or 10 seconds by default before output next chunk, this leads to 10 seconds delay if data track was added to output muxing context but no actual data packets were received hence sparse streams like onvif metadata may significantly increase delay.

Parameters

in	<i>streams_info</i>	- list of streams descriptions.
----	---------------------	---------------------------------

Returns

true

false

Reimplemented from [vxg::media::ffmpeg::Sink](#).

Definition at line 60 of file rtmp\_sink.h.

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

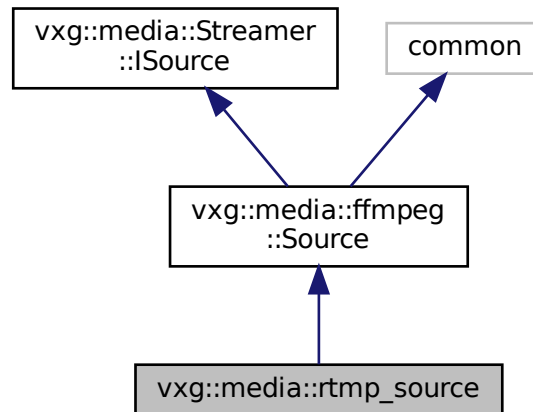
- [rtmp\\_sink.h](#)

## 10.32 vxg::media::rtmp\_source Class Reference

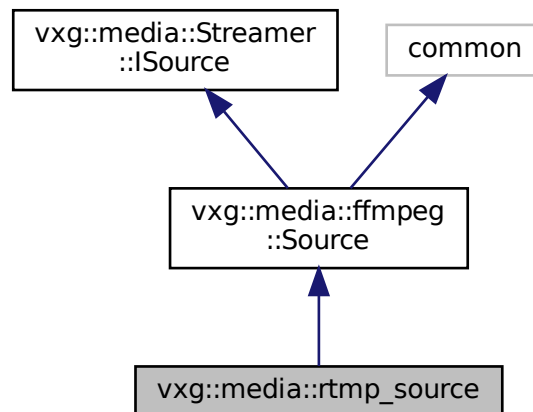
RTMP source class.

```
#include <streamer/rtmp_source.h>
```

Inheritance diagram for vxg::media::rtmp\_source:



Collaboration diagram for vxg::media::rtmp\_source:



### Public Member Functions

- virtual bool `init` ( `std::string` url)  
*Init source with url.*

## Additional Inherited Members

### 10.32.1 Detailed Description

RTMP source class.

Definition at line 13 of file rtmp\_source.h.

### 10.32.2 Member Function Documentation

#### 10.32.2.1 init()

```
virtual bool vxg::media::rtmp_source::init (
    std::string url ) [inline], [virtual]
```

Init source with url.

##### Parameters

in	url	RTMP url
----	-----	----------

##### Returns

true Success

false Failed

Reimplemented from [vxg::media::ffmpeg::Source](#).

Definition at line 24 of file rtmp\_source.h.

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

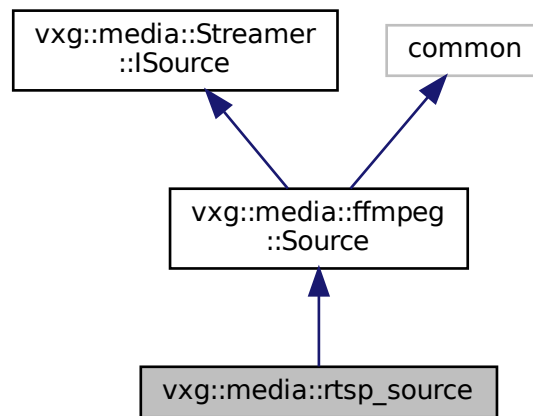
- [rtmp\\_source.h](#)

## 10.33 vxg::media::rtsp\_source Class Reference

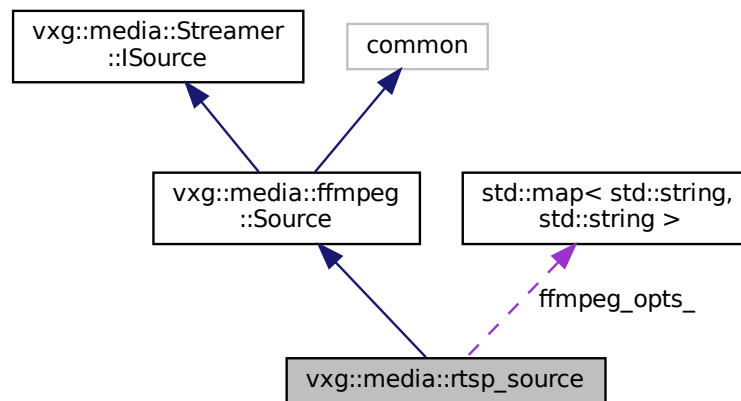
RTSP source class.

```
#include <streamer/rtsp_source.h>
```

Inheritance diagram for vxg::media::rtsp\_source:



Collaboration diagram for vxg::media::rtsp\_source:



## Public Member Functions

- `rtsp_source` (bool rtp\_over\_tcp=true, `std::vector< Streamer::MediaType >` media\_types= `std::vector< Streamer::MediaType >(0)`)  
Construct a new rtsp source object.
- `rtsp_source` ( `std::string` rtp\_transport="tcp", `std::vector< Streamer::MediaType >` media\_types= `std::vector< Streamer::MediaType >(0)`, `std::map< std::string, std::string >` ffmpeg\_opts={}, `std::chrono::seconds` timeout= `std::chrono::seconds(0)`)  
Construct a new rtsp source object.

- virtual bool [init](#) ( `std::string` url)  
*Overloaded init method.*
- virtual `std::string` [name](#) () override  
*Source class name.*

## Protected Attributes

- `std::map< std::string, std::string >` [ffmpeg\\_opts\\_](#)

## Additional Inherited Members

### 10.33.1 Detailed Description

RTSP source class.

Definition at line 13 of file `rtsp_source.h`.

### 10.33.2 Constructor & Destructor Documentation

#### 10.33.2.1 `rtsp_source()` [1/2]

```
vxg::media::rtsp_source::rtsp_source (
    bool rtp_over_tcp = true,
    std::vector< Streamer::MediaType > media_types = std::vector<Streamer::MediaType> (0)
) [inline]
```

Construct a new rtsp source object.

#### Parameters

in	<code>rtp_over_tcp</code>	Flag indicates if user wants RTP over TCP
in	<code>media_types</code>	List of media types to ask from RTSP server, can be used to filter out unnecessary tracks. If empty all types will be requested.

Definition at line 31 of file `rtsp_source.h`.

#### 10.33.2.2 `rtsp_source()` [2/2]

```
vxg::media::rtsp_source::rtsp_source (
    std::string rtp_transport = "tcp",
    std::vector< Streamer::MediaType > media_types = std::vector<Streamer::MediaType> (0),
```

```
std::map< std::string, std::string > ffmpeg_opts = {},  
std::chrono::seconds timeout = std::chrono::seconds(0) ) [inline]
```

Construct a new rtsp source object.



## Parameters

in	<i>rtp_transport</i>	RTP transport passed directly to ffmpeg.
in	<i>media_types</i>	List of media types to ask from RTSP server, can be used to filter out unnecessary tracks. If empty all types will be requested.
in	<i>ffmpeg_opts</i>	Map of ffmpeg options key values pairs.
in	<i>timeout</i>	RTSP client io timeout. Doesn't mean the connection will be closed after this timeout but specifies the amount of time ffmpeg spends in io loop spinning, infinite timeout causes spinning forever if connection wasn't closed but no data was received.

Definition at line 51 of file rtsp\_source.h.

### 10.33.3 Member Function Documentation

#### 10.33.3.1 init()

```
virtual bool vxg::media::rtsp_source::init (
    std::string url ) [inline], [virtual]
```

Overloaded init method.

## Parameters

in	<i>url</i>	RTSP URL link
----	------------	---------------

## Returns

true  
false

Reimplemented from [vxg::media::ffmpeg::Source](#).

Definition at line 69 of file rtsp\_source.h.

#### 10.33.3.2 name()

```
virtual std::string vxg::media::rtsp_source::name ( ) [inline], [override], [virtual]
```

Source class name.

## Returns

**std::string**

Reimplemented from [vxg::media::ffmpeg::Source](#).

Definition at line 164 of file rtsp\_source.h.

### 10.33.4 Field Documentation

#### 10.33.4.1 ffmpeg\_opts\_

```
std::map< std::string, std::string> vxg::media::rtsp_source::ffmpeg_opts_ [protected]
```

Definition at line 22 of file rtsp\_source.h.

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

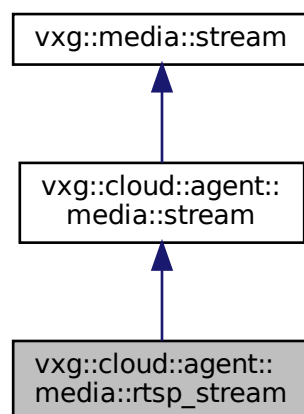
- [rtsp\\_source.h](#)

## 10.34 vxg::cloud::agent::media::rtsp\_stream Class Reference

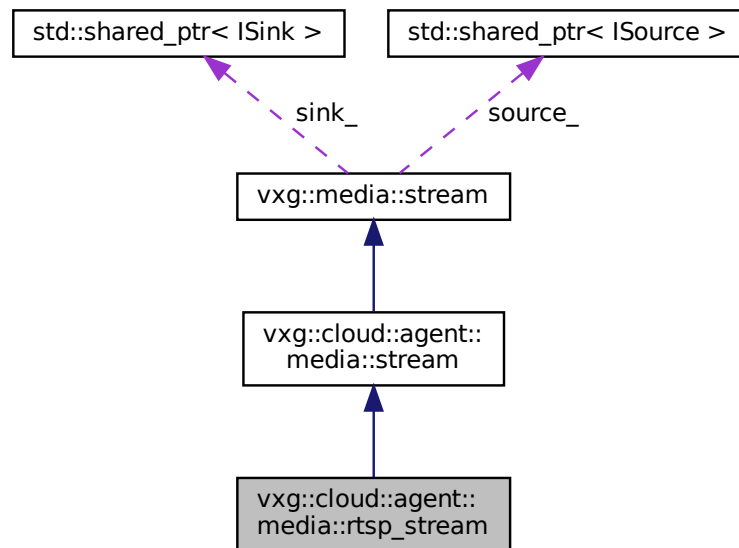
Implementation of the [media::stream](#) with RTSP source and NIY stubs.

```
#include <agent/rtsp-stream.h>
```

Inheritance diagram for vxg::cloud::agent::media::rtsp\_stream:



Collaboration diagram for vxg::cloud::agent::media::rtsp\_stream:



## Public Types

- typedef **std::shared\_ptr**< [rtsp\\_stream](#) > ptr

## Public Member Functions

- [rtsp\\_stream](#) ( **std::string** source\_url, **std::string** name, bool rtp\_transport\_tcp=true, bool recorder\_needs←\_source=false)  
*Construct a new rtsp stream object.*
- virtual [~rtsp\\_stream](#) ()
- virtual bool [start](#) ( **std::string** not\_used="" )
- bool [get\\_supported\\_stream](#) (proto::supported\_stream\_config &config)
- virtual bool [get\\_stream\\_caps](#) (proto::stream\_caps &caps) override  
*Get the media stream caps.*
- virtual bool [get\\_stream\\_config](#) (proto::stream\_config &streamConfig)  
*Get the stream config.*
- virtual bool [set\\_stream\\_config](#) (const [proto::stream\\_config](#) &streamConfig)  
*Set the streams config.*
- virtual bool [get\\_snapshot](#) (proto::event\_object::snapshot\_info\_object &snapshot)
- virtual **std::vector**< [proto::video\\_clip\\_info](#) > [record\\_get\\_list](#) ([cloud::time](#) begin, [cloud::time](#) end, bool align)  
*Get list of the recorded clips for specific time period.*
- virtual [proto::video\\_clip\\_info](#) [record\\_export](#) ([cloud::time](#) begin, [cloud::time](#) end)  
*Export recorded clip for specified time.*
- virtual bool [start\\_record](#) ()  
*Start recording of this media stream.*
- virtual bool [stop\\_record](#) ()  
*Stop recording of this stream.*

## Additional Inherited Members

### 10.34.1 Detailed Description

Implementation of the [media::stream](#) with RTSP source and NIY stubs.

Definition at line 18 of file rtsp-stream.h.

### 10.34.2 Member Typedef Documentation

#### 10.34.2.1 ptr

```
typedef std::shared_ptr<rtsp\_stream> vxg::cloud::agent::media::rtsp\_stream::ptr
```

Definition at line 34 of file rtsp-stream.h.

### 10.34.3 Constructor & Destructor Documentation

#### 10.34.3.1 rtsp\_stream()

```
vxg::cloud::agent::media::rtsp\_stream::rtsp\_stream (  
    std::string source_url,  
    std::string name,  
    bool rtp_transport_tcp = true,  
    bool recorder_needs_source = false ) [inline]
```

Construct a new rtsp stream object.

#### Parameters

<i>source_url</i>	RTSP url
<i>name</i>	Unique stream name
<i>rtp_transport_tcp</i>	true - RTP over TCP; false - RTP over UDP
<i>recorder_needs_source</i>	Indicates if stream needs source start before calling <a href="#">start_record()</a> virtual method.

Definition at line 43 of file rtsp-stream.h.

#### 10.34.3.2 ~rtsp\_stream()

```
virtual vxg::cloud::agent::media::rtsp\_stream::~~rtsp\_stream ( ) [inline], [virtual]
```

Definition at line 60 of file rtsp-stream.h.

## 10.34.4 Member Function Documentation

### 10.34.4.1 get\_snapshot()

```
virtual bool vxg::cloud::agent::media::rtsp_stream::get_snapshot (
    proto::event_object::snapshot_info_object & snapshot ) [inline], [virtual]
```

Definition at line 95 of file rtsp-stream.h.

### 10.34.4.2 get\_stream\_caps()

```
virtual bool vxg::cloud::agent::media::rtsp_stream::get_stream_caps (
    proto::stream_caps & caps ) [inline], [override], [virtual]
```

Get the media stream caps.

video/audio elementary streams caps request passes caps with names of the elementary streams for which caps are required to be filled inside this method

#### Parameters

out	caps	
-----	------	--

#### Returns

true if caps valid  
false if caps is invalid

Implements [vxg::cloud::agent::media::stream](#).

Definition at line 77 of file rtsp-stream.h.

### 10.34.4.3 get\_stream\_config()

```
virtual bool vxg::cloud::agent::media::rtsp_stream::get_stream_config (
    proto::stream_config & config ) [inline], [virtual]
```

Get the stream config.

#### Parameters

in, out	config	input config contains list of streams for which configuration should be returned
---------	--------	--

**Returns**

true if `config` is valid  
 false if `config` is invalid

Implements [vxd::cloud::agent::media::stream](#).

Definition at line 83 of file `rtsp-stream.h`.

**10.34.4.4 get\_supported\_stream()**

```
bool vxd::cloud::agent::media::rtsp_stream::get_supported_stream (
    proto::supported_stream_config & config ) [inline]
```

Definition at line 66 of file `rtsp-stream.h`.

**10.34.4.5 record\_export()**

```
virtual proto::video_clip_info vxd::cloud::agent::media::rtsp_stream::record_export (
    cloud::time begin,
    cloud::time end ) [inline], [virtual]
```

Export recorded clip for specified time.

**Parameters**

<i>begin</i>	
<i>end</i>	

**Returns**

[proto::video\\_clip\\_info](#)

Implements [vxd::cloud::agent::media::stream](#).

Definition at line 110 of file `rtsp-stream.h`.

**10.34.4.6 record\_get\_list()**

```
virtual std::vector<proto::video_clip_info> vxd::cloud::agent::media::rtsp_stream::record_↵
get_list (
    cloud::time begin,
    cloud::time end,
    bool align ) [inline], [virtual]
```

Get list of the recorded clips for specific time period.

## Parameters

in	<i>begin</i>	beginning of the time period
in	<i>end</i>	ending of the time period
in	<i>align</i>	Align returned records to key frames and begin/end. If true the implementation should align returned records to not include data with timestamps less than <i>begin</i> and greater than <i>end</i> . Also any returned record MUST start with key frame and the last frame of any not last record in the list MUST be the frame prior to key frame - first frame of the next record.
in	<i>limit</i>	Max records number that may be returned. Value 0 means no limitation.

## Returns

std::vector<proto::video\_clip\_info>

Implements [vxg::cloud::agent::media::stream](#).

Definition at line 103 of file rtsp-stream.h.

## 10.34.4.7 set\_stream\_config()

```
virtual bool vxg::cloud::agent::media::rtsp_stream::set_stream_config (
    const proto::stream_config & config ) [inline], [virtual]
```

Set the streams config.

## Parameters

in	<i>config</i>	input config contains list of streams for which configuration should be set
----	---------------	---

## Returns

true if config successfully set

false if config failed to set

Implements [vxg::cloud::agent::media::stream](#).

Definition at line 89 of file rtsp-stream.h.

## 10.34.4.8 start()

```
virtual bool vxg::cloud::agent::media::rtsp_stream::start (
    std::string not_used = "" ) [inline], [virtual]
```

Reimplemented from [vxg::media::stream](#).

Definition at line 62 of file rtsp-stream.h.

#### 10.34.4.9 start\_record()

```
virtual bool vxg::cloud::agent::media::rtsp_stream::start_record ( ) [inline], [virtual]
```

Start recording of this media stream.

Called only if memory card is presented and can be used.

##### Returns

true if recording started  
false if recording start failed

##### See also

[agent::event\\_stream::on\\_get\\_memorycard\\_info](#)

Implements [vxg::cloud::agent::media::stream](#).

Definition at line 118 of file rtsp-stream.h.

#### 10.34.4.10 stop\_record()

```
virtual bool vxg::cloud::agent::media::rtsp_stream::stop_record ( ) [inline], [virtual]
```

Stop recording of this stream.

##### Returns

true Stopped  
false Failed to stop

Implements [vxg::cloud::agent::media::stream](#).

Definition at line 124 of file rtsp-stream.h.

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

- [rtsp-stream.h](#)

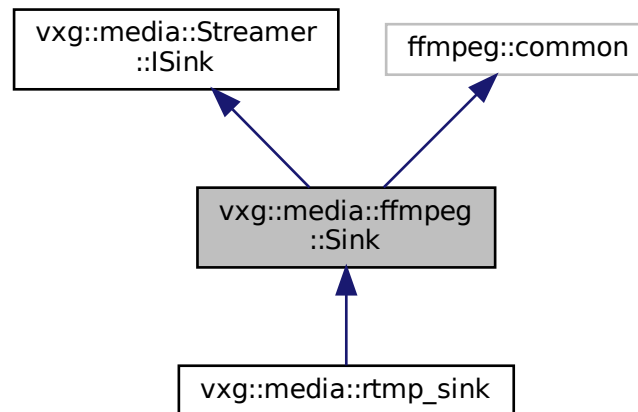


## 10.35 vxg::media::ffmpeg::Sink Class Reference

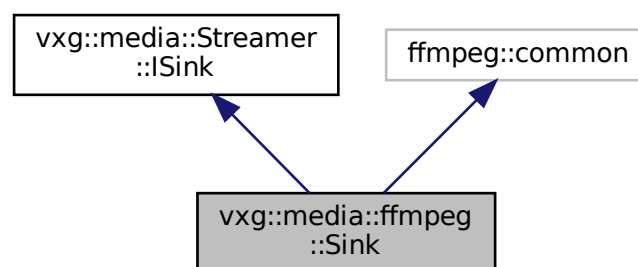
Base ffmpeg sink class.

```
#include <streamer/ffmpeg_sink.h>
```

Inheritance diagram for vxg::media::ffmpeg::Sink:



Collaboration diagram for vxg::media::ffmpeg::Sink:



### Public Member Functions

- [Sink](#) ()
- virtual [~Sink](#) ()
- bool [init](#) ( [std::string](#) url, [std::string](#) fmt, [std::shared\\_ptr](#)< [std::vector](#)< [uint8\\_t](#) >> data\_buffer=nullptr)  
*Sink* init.

- virtual bool `init` ( `std::string` url="" )  
*Init sink.*
- virtual bool `finit` ()  
*Deinit sink.*
- virtual void `stop` ()
- virtual void `error` ( `Streamer::StreamError` stream\_error )  
*Media processing error callback, called when ISink::process returned false or linked source's ISource::pullFrame returned false, or when ISource::error was called.*
- virtual `std::string` `name` ()  
*Sink name.*
- virtual bool `droppable` ()  
*If sink of with dropping its media frames.*
- virtual bool `negotiate` ( `std::vector`< `Streamer::StreamInfo` > )  
*Negotiation callback, this method called with collected from the ISource::negotiate media stream description.*
- virtual `cloud::duration` `duration` ()  
*Processed stream duration.*

## Additional Inherited Members

### 10.35.1 Detailed Description

Base ffmpeg sink class.

Definition at line 12 of file `ffmpeg_sink.h`.

### 10.35.2 Constructor & Destructor Documentation

#### 10.35.2.1 Sink()

```
vsg::media::ffmpeg::Sink::Sink ( )
```

#### 10.35.2.2 ~Sink()

```
virtual vsg::media::ffmpeg::Sink::~Sink ( ) [virtual]
```

### 10.35.3 Member Function Documentation

### 10.35.3.1 droppable()

```
virtual bool vxg::media::ffmpeg::Sink::droppable ( ) [inline], [virtual]
```

If sink of with dropping its media frames.

#### Returns

true Internal media thread allowed to drop frames if internal media queue is full.  
false No media frames dropping allowed.

Implements [vxg::media::Streamer::ISink](#).

Reimplemented in [vxg::media::rtmp\\_sink](#).

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

### 10.35.3.2 duration()

```
virtual cloud::duration vxg::media::ffmpeg::Sink::duration ( ) [inline], [virtual]
```

Processed stream duration.

#### Returns

duration

Reimplemented from [vxg::media::Streamer::ISink](#).

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

### 10.35.3.3 error()

```
virtual void vxg::media::ffmpeg::Sink::error (
    Streamer::StreamError error ) [inline], [virtual]
```

Media processing error callback, called when `ISink::process` returned false or linked source's `ISource::pullFrame` returned false, or when `ISource::error` was called.

#### Parameters

<i>error</i>	Error type.
--------------	-------------

Implements [vxg::media::Streamer::ISink](#).

Reimplemented in [vxg::media::rtmp\\_sink](#).

Definition at line 33 of file ffmpeg\_sink.h.

#### 10.35.3.4 finit()

```
virtual bool vxg::media::ffmpeg::Sink::finit ( ) [virtual]
```

Deinit sink.

##### Returns

true finit success.  
false finit failed.

Implements [vxg::media::Streamer::ISink](#).

#### 10.35.3.5 init() [1/2]

```
bool vxg::media::ffmpeg::Sink::init (
    std::string url,
    std::string fmt,
    std::shared_ptr< std::vector< uint8_t >> data_buffer = nullptr )
```

[Sink](#) init.

##### Parameters

<i>url</i>	Output url
<i>fmt</i>	Output format
<i>data_buffer</i>	Output buffer for output to memory, if specified and not nullptr the <code>url</code> will be ignored.

##### Returns

true On success  
false On failure

#### 10.35.3.6 init() [2/2]

```
virtual bool vxg::media::ffmpeg::Sink::init (
    std::string url = "" ) [virtual]
```

Init sink.

## Parameters

<i>in</i>	<i>url</i>	Url if needed.
-----------	------------	----------------

## Returns

true init success.

false init failed.

Implements [vxg::media::Streamer::ISink](#).

Reimplemented in [vxg::media::rtmp\\_sink](#).

### 10.35.3.7 name()

```
virtual std::string vxg::media::ffmpeg::Sink::name ( ) [inline], [virtual]
```

[Sink](#) name.

## Returns

**std::string**

Implements [vxg::media::Streamer::ISink](#).

Reimplemented in [vxg::media::rtmp\\_sink](#).

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

### 10.35.3.8 negotiate()

```
virtual bool vxg::media::ffmpeg::Sink::negotiate (
    std::vector< Streamer::StreamInfo > info ) [virtual]
```

Negotiation callback, this method called with collected from the `ISource::negotiate` media stream description.

## Parameters

<i>info</i>	List of elementary streams descriptions.
-------------	--

## Returns

true If streams descriptions accepted.

false Streams not accepted, will cause media thread stopping.

Reimplemented from [vxg::media::Streamer::ISink](#).

Reimplemented in [vxg::media::rtmp\\_sink](#).

### 10.35.3.9 stop()

```
virtual void vxg::media::ffmpeg::Sink::stop ( ) [virtual]
```

Reimplemented from [vxg::media::Streamer::ISink](#).

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

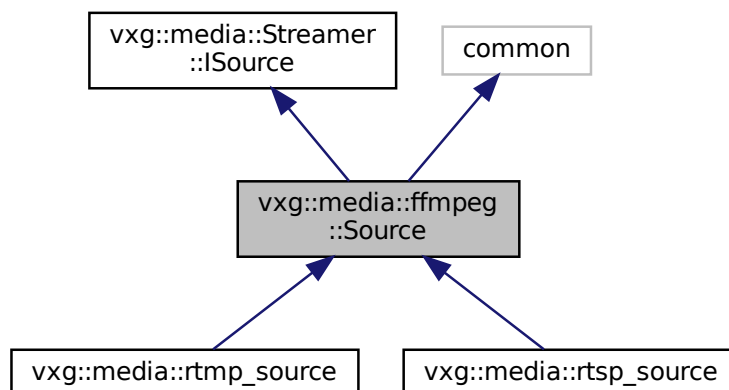
- [ffmpeg\\_sink.h](#)

## 10.36 vxg::media::ffmpeg::Source Class Reference

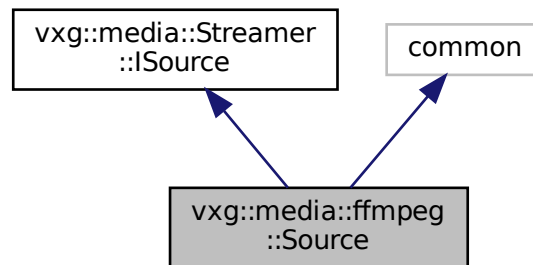
Base ffmpeg source class.

```
#include <streamer/ffmpeg_source.h>
```

Inheritance diagram for vxg::media::ffmpeg::Source:



Collaboration diagram for vxg::media::ffmpeg::Source:



## Public Member Functions

- [Source](#) ()
- virtual [~Source](#) ()
- bool [init](#) ( [std::string](#) url, AVDictionary \*opts, [std::string](#) fmt="")  
*Init ffmpeg source with specific ffmpeg options.*
- bool [init](#) ( [std::shared\\_ptr](#)< [std::vector](#)< uint8\_t >> input\_buffer, AVDictionary \*opts, [std::string](#) fmt)  
*Init ffmpeg memory source with specific ffmpeg options.*
- virtual bool [init](#) ( [std::string](#) url="")  
*Init source.*
- virtual void [finit](#) ()  
*Finit souce.*
- virtual [std::shared\\_ptr](#)< [Streamer::MediaFrame](#) > [pullFrame](#) ()  
*Main method of the Mode::PULL mode data producing.*
- virtual [std::string](#) [name](#) ()  
*Source class name.*
- virtual [std::vector](#)< [Streamer::StreamInfo](#) > [negotiate](#) ()  
*Negotiation callback.*
- virtual void [stop](#) ()

## Additional Inherited Members

### 10.36.1 Detailed Description

Base ffmpeg source class.

Definition at line 10 of file `ffmpeg_source.h`.

### 10.36.2 Constructor & Destructor Documentation

### 10.36.2.1 Source()

```
vxg::media::ffmpeg::Source::Source ( )
```

Definition at line 9 of file ffmpeg\_source.cc.

### 10.36.2.2 ~Source()

```
vxg::media::ffmpeg::Source::~~Source ( ) [virtual]
```

Definition at line 12 of file ffmpeg\_source.cc.

## 10.36.3 Member Function Documentation

### 10.36.3.1 finit()

```
void vxg::media::ffmpeg::Source::finit ( ) [virtual]
```

Finit souce.

Implements [vxg::media::Streamer::ISource](#).

Definition at line 30 of file ffmpeg\_source.cc.

### 10.36.3.2 init() [1/3]

```
bool vxg::media::ffmpeg::Source::init (
    std::shared_ptr< std::vector< uint8_t >> input_buffer,
    AVDictionary * opts,
    std::string fmt )
```

Init ffmpeg memory source with specific ffmpeg options.

#### Parameters

in	<i>input_buffer</i>	Input memory buffer containing whole media.
in	<i>opts</i>	ffmpeg options
in	<i>fmt</i>	ffmpeg input format to prevent auto-detection. ex.: "flv", "mp4", "http" etc.



**Returns**

true  
false

Definition at line 20 of file ffmpeg\_source.cc.

**10.36.3.3 init() [2/3]**

```
bool vxg::media::ffmpeg::Source::init (
    std::string url,
    AVDictionary * opts,
    std::string fmt = "" )
```

Init ffmpeg source with specific ffmpeg options.

**Parameters**

in	<i>url</i>	Url
in	<i>opts</i>	ffmpeg options
in	<i>fmt</i>	ffmpeg input format to prevent auto-detection. ex.: "flv", "rtsp", "http" etc.

**Returns**

true  
false

Definition at line 14 of file ffmpeg\_source.cc.

**10.36.3.4 init() [3/3]**

```
bool vxg::media::ffmpeg::Source::init (
    std::string url = "" ) [virtual]
```

Init source.

**Parameters**

<i>url</i>	Url if needed.
------------	----------------

**Returns**

true Init success.  
false Init failed.

Implements [vxg::media::Streamer::ISource](#).

Reimplemented in [vxg::media::rtsp\\_source](#), and [vxg::media::rtmp\\_source](#).

Definition at line 26 of file `ffmpeg_source.cc`.

#### 10.36.3.5 name()

```
virtual std::string vxg::media::ffmpeg::Source::name ( ) [inline], [virtual]
```

[Source](#) class name.

Returns

**std::string**

Implements [vxg::media::Streamer::ISource](#).

Reimplemented in [vxg::media::rtsp\\_source](#).

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

#### 10.36.3.6 negotiate()

```
std::vector< Streamer::StreamInfo > vxg::media::ffmpeg::Source::negotiate ( ) [virtual]
```

Negotiation callback.

Called by internals. Class implementation should return the list of the streams info source will be producing for the sinks, this list will be then passed to the `ISink::negotiate` method.

Returns

**std::vector**<[Streamer::StreamInfo](#)>

Implements [vxg::media::Streamer::ISource](#).

Definition at line 34 of file `ffmpeg_source.cc`.

#### 10.36.3.7 pullFrame()

```
std::shared_ptr< Streamer::MediaFrame > vxg::media::ffmpeg::Source::pullFrame ( ) [virtual]
```

Main method of the `Mode::PULL` mode data producing.

Called by internals if the source operation mode is `Mode::PULL`. Implementation should return media frame object with correctly filled fields.

Returns

**std::shared\_ptr**<[MediaFrame](#)>

Implements [vxg::media::Streamer::ISource](#).

Definition at line 93 of file `ffmpeg_source.cc`.

### 10.36.3.8 stop()

```
void vxg::media::ffmpeg::Source::stop ( ) [virtual]
```

Reimplemented from [vxg::media::Streamer::ISource](#).

Definition at line 186 of file `ffmpeg_source.cc`.

The documentation for this class was generated from the following files:

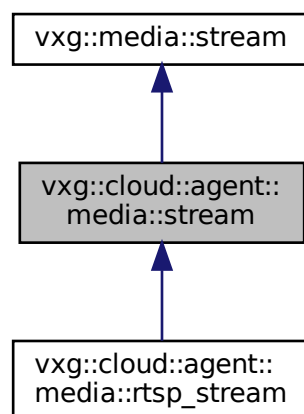
- [ffmpeg\\_source.h](#)
- [ffmpeg\\_source.cc](#)

## 10.37 vxg::cloud::agent::media::stream Class Reference

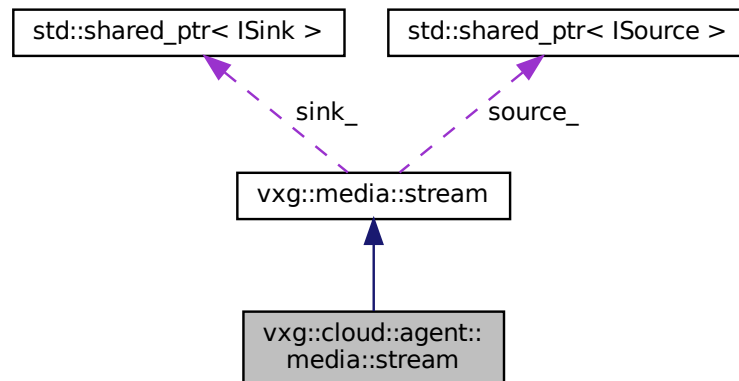
Cloud agent media stream abstract class.

```
#include <agent/stream.h>
```

Inheritance diagram for vxg::cloud::agent::media::stream:



Collaboration diagram for vxg::cloud::agent::media::stream:



## Public Types

- typedef **std::shared\_ptr**< [stream](#) > [ptr](#)  
*std::shared\_ptr to the base\_stream*

## Public Member Functions

- [stream](#) ( **std::string** name, [vxg::media::Streamer::ISource::ptr](#) source, **std::function**< void([vxg::media::Streamer::StreamError](#) sink\_error\_cb, bool recorder\_needs\_source=false) >  
*Construct a new agent media stream object.*
- virtual [~stream](#) ()
- virtual bool [get\\_stream\\_caps](#) ([cloud::agent::proto::stream\\_caps](#) &caps)=0  
*Get the media stream caps.*
- virtual bool [get\\_supported\\_stream](#) ([cloud::agent::proto::supported\\_stream\\_config](#) &supported\_stream)=0  
*Get the supported stream description.*
- virtual bool [get\\_stream\\_config](#) ([cloud::agent::proto::stream\\_config](#) &config)=0  
*Get the stream config.*
- virtual bool [set\\_stream\\_config](#) (const [cloud::agent::proto::stream\\_config](#) &config)=0  
*Set the streams config.*
- virtual bool [get\\_snapshot](#) ([cloud::agent::proto::event\\_object::snapshot\\_info\\_object](#) &snapshot)=0  
*Get the snapshot image of this media stream.*
- virtual bool [record\\_needs\\_source](#) ()  
*Should returns true if [agent::manager](#) should start stream source before calling [start\\_record\(\)](#)*
- virtual bool [start\\_record](#) ()=0  
*Start recording of this media stream.*
- virtual bool [stop\\_record](#) ()=0  
*Stop recording of this stream.*
- virtual **std::vector**< [cloud::agent::proto::video\\_clip\\_info](#) > [record\\_get\\_list](#) ([cloud::time](#) begin, [cloud::time](#) end, bool align=true)=0  
*Get list of the recorded clips for specific time period.*
- virtual [cloud::agent::proto::video\\_clip\\_info](#) [record\\_export](#) ([cloud::time](#) begin, [cloud::time](#) end)=0  
*Export recorded clip for specified time.*

## Additional Inherited Members

### 10.37.1 Detailed Description

Cloud agent media stream abstract class.

[vxg::media::stream](#) derived class with VXG Cloud proto callbacks

Definition at line 21 of file agent/stream.h.

### 10.37.2 Member Typedef Documentation

#### 10.37.2.1 ptr

```
typedef std::shared_ptr<stream> vxg::cloud::agent::media::stream::ptr
```

**std::shared\_ptr** to the base\_stream

Definition at line 29 of file agent/stream.h.

### 10.37.3 Constructor & Destructor Documentation

#### 10.37.3.1 stream()

```
vxg::cloud::agent::media::stream::stream (  
    std::string name,  
    vxg::media::Streamer::ISource::ptr source,  
    std::function< void(vxg::media::Streamer::StreamError)> sink_error_cb,  
    bool recorder_needs_source = false ) [inline]
```

Construct a new agent media stream object.

#### Parameters

in	<i>name</i>	Unique stream name which will be used by the VXG Cloud API
in	<i>source</i>	Source object pointer
in	<i>sink_error_cb</i>	Callback which will be called on sink error
in	<i>recorder_needs_source</i>	Indicates if stream needs source start before calling <a href="#">start_record()</a> virtual method.

Definition at line 39 of file agent/stream.h.

### 10.37.3.2 `~stream()`

```
virtual vxg::cloud::agent::media::stream::~~stream ( ) [inline], [virtual]
```

Reimplemented from [vxg::media::stream](#).

Definition at line 48 of file agent/stream.h.

## 10.37.4 Member Function Documentation

### 10.37.4.1 `get_snapshot()`

```
virtual bool vxg::cloud::agent::media::stream::get_snapshot (
    cloud::agent::proto::event_object::snapshot_info_object & snapshot ) [pure virtual]
```

Get the snapshot image of this media stream.

#### Parameters

out	<i>snapshot</i>	snapshot object
-----	-----------------	-----------------

#### Returns

true if snapshot is valid  
false if snapshot is invalid

### 10.37.4.2 `get_stream_caps()`

```
virtual bool vxg::cloud::agent::media::stream::get_stream_caps (
    cloud::agent::proto::stream_caps & caps ) [pure virtual]
```

Get the media stream caps.

video/audio elementary streams caps request passes caps with names of the elementary streams for which caps are required to be filled inside this method

#### Parameters

out	<i>caps</i>	
-----	-------------	--

#### Returns

true if caps valid  
false if caps is invalid

Implemented in [vxg::cloud::agent::media::rtsp\\_stream](#).

#### 10.37.4.3 get\_stream\_config()

```
virtual bool vxg::cloud::agent::media::stream::get_stream_config (
    cloud::agent::proto::stream_config & config ) [pure virtual]
```

Get the stream config.

##### Parameters

<i>in, out</i>	<i>config</i>	input config contains list of streams for which configuration should be returned
----------------	---------------	--

##### Returns

true if config is valid  
false if config is invalid

Implemented in [vxg::cloud::agent::media::rtsp\\_stream](#).

#### 10.37.4.4 get\_supported\_stream()

```
virtual bool vxg::cloud::agent::media::stream::get_supported_stream (
    cloud::agent::proto::supported_stream_config & supported_stream ) [pure virtual]
```

Get the supported stream description.

##### Parameters

<i>out</i>	<i>supported_stream</i>	Stream supported by device
------------	-------------------------	----------------------------

##### Returns

true if supported\_stream is valid  
false if supported\_stream is not valid

#### 10.37.4.5 record\_export()

```
virtual cloud::agent::proto::video_clip_info vxg::cloud::agent::media::stream::record_export (
    cloud::time begin,
    cloud::time end ) [pure virtual]
```

Export recorded clip for specified time.

## Parameters

<i>begin</i>	
<i>end</i>	

## Returns

[proto::video\\_clip\\_info](#)

Implemented in [vxg::cloud::agent::media::rtsp\\_stream](#).

**10.37.4.6 record\_get\_list()**

```
virtual std::vector<cloud::agent::proto::video_clip_info> vxg::cloud::agent::media::stream<->
::record_get_list (
    cloud::time begin,
    cloud::time end,
    bool align = true ) [pure virtual]
```

Get list of the recorded clips for specific time period.

## Parameters

in	<i>begin</i>	beginning of the time period
in	<i>end</i>	ending of the time period
in	<i>align</i>	Align returned records to key frames and begin/end. If true the implementation should align returned records to not include data with timestamps less than <i>begin</i> and greater than <i>end</i> . Also any returned record MUST start with key frame and the last frame of any not last record in the list MUST be the frame prior to key frame - first frame of the next record.
in	<i>limit</i>	Max records number that may be returned. Value 0 means no limitation.

## Returns

std::vector<proto::video\_clip\_info>

Implemented in [vxg::cloud::agent::media::rtsp\\_stream](#).

**10.37.4.7 record\_needs\_source()**

```
virtual bool vxg::cloud::agent::media::stream::record_needs_source ( ) [inline], [virtual]
```

Should returns true if [agent::manager](#) should start stream source before calling [start\\_record\(\)](#)

## Returns

true [agent::manager](#) should start stream source  
false [agent::manager](#) may not start stream source

Definition at line 104 of file agent/stream.h.



#### 10.37.4.8 set\_stream\_config()

```
virtual bool vxg::cloud::agent::media::stream::set_stream_config (
    const vxg::cloud::agent::proto::stream_config & config ) [pure virtual]
```

Set the streams config.

##### Parameters

in	<i>config</i>	input config contains list of streams for which configuration should be set
----	---------------	---

##### Returns

true if config successfully set  
false if config failed to set

Implemented in [vxg::cloud::agent::media::rtsp\\_stream](#).

#### 10.37.4.9 start\_record()

```
virtual bool vxg::cloud::agent::media::stream::start_record ( ) [pure virtual]
```

Start recording of this media stream.

Called only if memory card is presented and can be used.

##### Returns

true if recording started  
false if recording start failed

##### See also

[agent::event\\_stream::on\\_get\\_memorycard\\_info](#)

Implemented in [vxg::cloud::agent::media::rtsp\\_stream](#).

#### 10.37.4.10 stop\_record()

```
virtual bool vxg::cloud::agent::media::stream::stop_record ( ) [pure virtual]
```

Stop recording of this stream.

##### Returns

true Stopped  
false Failed to stop

Implemented in [vxg::cloud::agent::media::rtsp\\_stream](#).

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

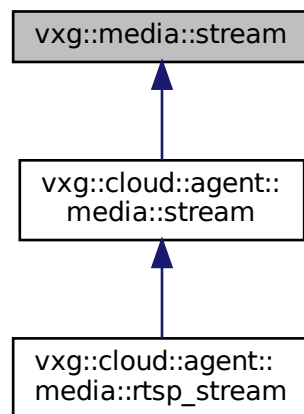
- [agent/stream.h](#)

## 10.38 vxg::media::stream Class Reference

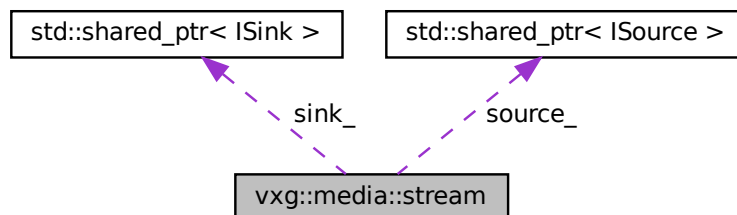
base media stream abstract class

```
#include <streamer/stream.h>
```

Inheritance diagram for vxg::media::stream:



Collaboration diagram for vxg::media::stream:



### Public Types

- typedef `std::shared_ptr< stream >` `ptr`  
*std::shared\_ptr to the base\_stream*

## Public Member Functions

- [stream](#) ( **std::string** name, [Streamer::ISource::ptr](#) source, [Streamer::ISink::ptr](#) sink)  
*Construct a new base stream object.*
- virtual [~stream](#) ()
- virtual bool [init\\_source](#) ( **std::string** url)  
*Initialize the source.*
- virtual void [finit\\_source](#) ()  
*Deinitialize source.*
- virtual bool [init\\_sink](#) ( **std::string** uri)  
*Init media sink.*
- virtual void [finit\\_sink](#) ()  
*Deinitialize sink.*

## Protected Attributes

- [Streamer::ISource::ptr](#) source\_  
*media source*
- [Streamer::ISink::ptr](#) sink\_  
*media sink*

### 10.38.1 Detailed Description

base media stream abstract class

Media stream is the class representing media stream retranslation from the media source derived from the [Streamer::ISource](#) to the media sink derived from the [Streamer::ISink](#). For instance, media stream could be a pair of RTSP source and RTMP sink, i.e. such media stream will be a retranslator of the RTSP stream to the RTMP

Definition at line 22 of file streamer/stream.h.

### 10.38.2 Member Typedef Documentation

#### 10.38.2.1 ptr

```
typedef std::shared_ptr<stream> vxg::media::stream::ptr
```

**std::shared\_ptr** to the base\_stream

Definition at line 27 of file streamer/stream.h.

### 10.38.3 Constructor & Destructor Documentation

#### 10.38.3.1 stream()

```
vxg::media::stream::stream (
    std::string name,
    Streamer::ISource::ptr source,
    Streamer::ISink::ptr sink ) [inline]
```

Construct a new base stream object.

**Parameters**

<i>name</i>	Unique stream name which will be used by the VXG Cloud API
<i>source</i>	Source object pointer
<i>sink</i>	Sink object pointer

Definition at line 34 of file streamer/stream.h.

**10.38.3.2 ~stream()**

```
virtual vxg::media::stream::~~stream ( ) [inline], [virtual]
```

Reimplemented in [vxg::cloud::agent::media::stream](#).

Definition at line 44 of file streamer/stream.h.

**10.38.4 Member Function Documentation****10.38.4.1 finit\_sink()**

```
virtual void vxg::media::stream::finit_sink ( ) [inline], [virtual]
```

Deinitialize sink.

Derived class deinitialize and deallocates base\_stream::sink\_

Definition at line 93 of file streamer/stream.h.

**10.38.4.2 finit\_source()**

```
virtual void vxg::media::stream::finit_source ( ) [inline], [virtual]
```

Deinitialize source.

Definition at line 66 of file streamer/stream.h.

**10.38.4.3 init\_sink()**

```
virtual bool vxg::media::stream::init_sink (
    std::string uri ) [inline], [virtual]
```

Init media sink.

Derived class should allocate and initialize base\_stream::sink\_ with RTMP sink publishing media stream to the RTMP server pointed by the uri

## Parameters

<i>in</i>	<i>uri</i>	sink stream url if needed
-----------	------------	---------------------------

## Returns

true Sink started  
false Sink start failed

Definition at line 80 of file streamer/stream.h.

## 10.38.4.4 init\_source()

```
virtual bool vxg::media::stream::init_source (
    std::string url ) [inline], [virtual]
```

Initialize the source.

Called by the internal code, derived class should allocate and set base\_stream::source\_ with [Streamer::ISink](#) derived object pointer.

## Parameters

<i>url</i>	source url
------------	------------

## Returns

true if successfully initialized source  
false if source initialization failed

Definition at line 56 of file streamer/stream.h.

## 10.38.5 Field Documentation

## 10.38.5.1 sink\_

```
Streamer::ISink::ptr vxg::media::stream::sink_ [protected]
```

media sink

Definition at line 201 of file streamer/stream.h.

### 10.38.5.2 source\_

```
Streamer::ISource::ptr vxg::media::stream::source_ [protected]
```

media source

Definition at line 199 of file streamer/stream.h.

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

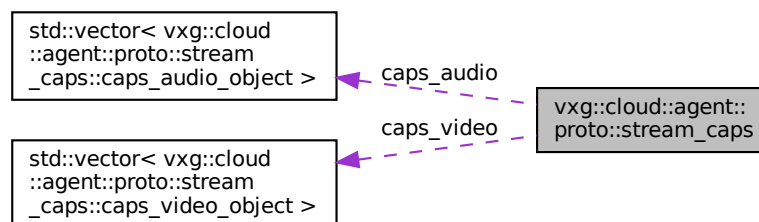
- [streamer/stream.h](#)

## 10.39 vxg::cloud::agent::proto::stream\_caps Struct Reference

Media stream capabilities.

```
#include <agent-proto/objects/caps.h>
```

Collaboration diagram for vxg::cloud::agent::proto::stream\_caps:



## Data Structures

- struct [caps\\_audio\\_object](#)  
*Audio streams capabilities.*
- struct [caps\\_video\\_object](#)  
*Video streams capabilities.*

## Data Fields

- **std::vector< [caps\\_video\\_object](#) > caps\_video**  
*List of video streams capabilities.*
- **std::vector< [caps\\_audio\\_object](#) > caps\_audio**  
*List of audio streams capabilities.*

### 10.39.1 Detailed Description

Media stream capabilities.

Definition at line 175 of file caps.h.

### 10.39.2 Field Documentation

#### 10.39.2.1 caps\_audio

```
std::vector<caps_audio_object> vxg::cloud::agent::proto::stream_caps::caps_audio
```

List of audio streams capabilities.

Definition at line 276 of file caps.h.

#### 10.39.2.2 caps\_video

```
std::vector<caps_video_object> vxg::cloud::agent::proto::stream_caps::caps_video
```

List of video streams capabilities.

Definition at line 274 of file caps.h.

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

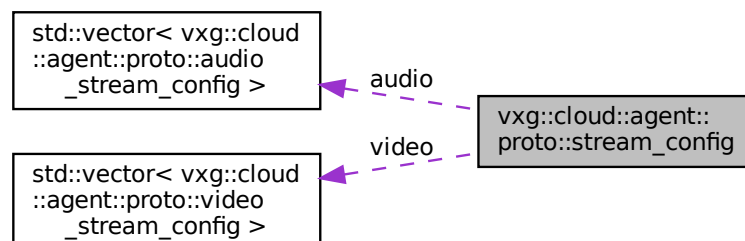
- [caps.h](#)

## 10.40 vxg::cloud::agent::proto::stream\_config Struct Reference

Media stream config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::stream\_config:



## Data Fields

- `std::vector< video\_stream\_config > video`  
*List of video media stream configs.*
- `std::vector< audio\_stream\_config > audio`  
*List of audio media stream configs.*

### 10.40.1 Detailed Description

Media stream config.

Definition at line 222 of file config.h.

### 10.40.2 Field Documentation

#### 10.40.2.1 audio

```
std::vector<audio\_stream\_config> vxg::cloud::agent::proto::stream_config::audio
```

List of audio media stream configs.

Definition at line 226 of file config.h.

#### 10.40.2.2 video

```
std::vector<video\_stream\_config> vxg::cloud::agent::proto::stream_config::video
```

List of video media stream configs.

Definition at line 224 of file config.h.

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

- [config.h](#)

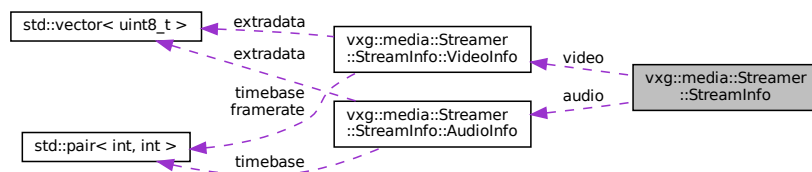


## 10.41 vxg::media::Streamer::StreamInfo Struct Reference

Stream info description.

```
#include <streamer/base_streamer.h>
```

Collaboration diagram for vxg::media::Streamer::StreamInfo:



### Data Structures

- struct [AudioInfo](#)  
*Audio stream info.*
- struct [VideoInfo](#)  
*Video stream info.*

### Public Types

- enum [StreamType](#) {  
ST\_UNKNOWN, ST\_VIDEO, ST\_AUDIO, ST\_DATA,  
ST\_ANY }  
*Stream type.*
- enum [VideoCodec](#) { VC\_UNKNOWN, VC\_H264 }  
*Video codec type.*
- enum [AudioCodec](#) {  
AC\_UNKNOWN, AC\_AAC, AC\_G711\_U, AC\_G711\_A,  
AC\_LPCM, AC\_G726, AC\_OPUS }  
*Audio codec.*
- enum [DataCodec](#) { DC\_UNKNOWN, DC\_ONVIF }  
*Data codec.*

### Data Fields

- [StreamType](#) type  
*Stream type.*
- [VideoInfo](#) video  
*Video stream info. Should be filled if stream type is ST\_VIDEO.*
- [AudioInfo](#) audio  
*Audio stream info. Should be filled if stream type is ST\_AUDIO.*

### 10.41.1 Detailed Description

Stream info description.

Definition at line 296 of file base\_streamer.h.

### 10.41.2 Member Enumeration Documentation

#### 10.41.2.1 AudioCodec

```
enum vxg::media::Streamer::StreamInfo::AudioCodec
```

Audio codec.

Enumerator

AC_UNKNOWN	
AC_AAC	
AC_G711_U	
AC_G711_A	
AC_LPCM	
AC_G726	
AC_OPUS	

Definition at line 336 of file base\_streamer.h.

#### 10.41.2.2 DataCodec

```
enum vxg::media::Streamer::StreamInfo::DataCodec
```

Data codec.

Enumerator

DC_UNKNOWN	
DC_ONVIF	

Definition at line 369 of file base\_streamer.h.

#### 10.41.2.3 StreamType

```
enum vxg::media::Streamer::StreamInfo::StreamType
```

Stream type.

#### Enumerator

ST_UNKNOWN	
ST_VIDEO	
ST_AUDIO	
ST_DATA	
ST_ANY	

Definition at line 298 of file base\_streamer.h.

### 10.41.2.4 VideoCodec

```
enum vxg::media::Streamer::StreamInfo::VideoCodec
```

Video codec type.

#### Enumerator

VC_UNKNOWN	
VC_H264	

Definition at line 301 of file base\_streamer.h.

## 10.41.3 Field Documentation

### 10.41.3.1 audio

```
AudioInfo vxg::media::Streamer::StreamInfo::audio
```

Audio stream info. Should be filled if stream type is ST\_AUDIO.

Definition at line 384 of file base\_streamer.h.

### 10.41.3.2 type

```
StreamType vxg::media::Streamer::StreamInfo::type
```

Stream type.

Definition at line 380 of file base\_streamer.h.

### 10.41.3.3 video

`VideoInfo vxg::media::Streamer::StreamInfo::video`

Video stream info. Should be filled if stream type is ST\_VIDEO.

Definition at line 382 of file `base_streamer.h`.

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

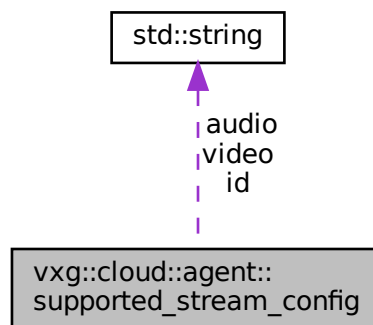
- [base\\_streamer.h](#)

## 10.42 vxg::cloud::agent::supported\_stream\_config Struct Reference

Supported stream config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for `vxg::cloud::agent::supported_stream_config`:



### Data Fields

- `std::string id`  
*id: string, name of media stream, unique for the camera*
- `std::string video`  
*video: optional string, video ES that is sent in this media stream*
- `std::string audio`  
*audio: optional string, audio ES that is sent in this media stream*

### 10.42.1 Detailed Description

Supported stream config.

Definition at line 1270 of file `config.h`.

## 10.42.2 Field Documentation

### 10.42.2.1 audio

**std::string** vxg::cloud::agent::supported\_stream\_config::audio

audio: optional string, audio ES that is sent in this media stream

Definition at line 1276 of file config.h.

### 10.42.2.2 id

**std::string** vxg::cloud::agent::supported\_stream\_config::id

id: string, name of media stream, unique for the camera

Definition at line 1272 of file config.h.

### 10.42.2.3 video

**std::string** vxg::cloud::agent::supported\_stream\_config::video

video: optional string, video ES that is sent in this media stream

Definition at line 1274 of file config.h.

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

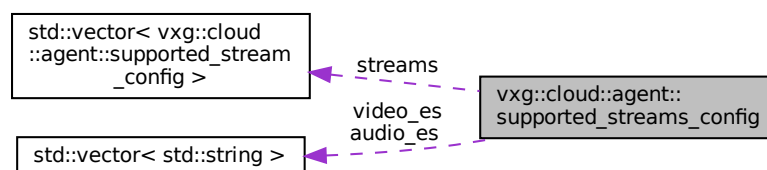
- [config.h](#)

## 10.43 vxg::cloud::agent::supported\_streams\_config Struct Reference

Supported streams config, list of [supported\\_stream\\_config](#).

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::supported\_streams\_config:



## Data Fields

- **std::vector**< [supported\\_stream\\_config](#) > **streams**  
*streams: list of [supported\\_stream\\_config](#) struct, camera media streams*
- **std::vector**< **std::string** > **video\_es**  
*list of string, camera video ES*
- **std::vector**< **std::string** > **audio\_es**  
*list of string, camera audio ES*

### 10.43.1 Detailed Description

Supported streams config, list of [supported\\_stream\\_config](#).

Definition at line 1286 of file config.h.

### 10.43.2 Field Documentation

#### 10.43.2.1 audio\_es

```
std::vector< std::string> vxg::cloud::agent::supported_streams_config::audio_es
```

list of string, camera audio ES

Definition at line 1292 of file config.h.

#### 10.43.2.2 streams

```
std::vector<supported\_stream\_config> vxg::cloud::agent::supported_streams_config::streams
```

streams: list of [supported\\_stream\\_config](#) struct, camera media streams

Definition at line 1288 of file config.h.

#### 10.43.2.3 video\_es

```
std::vector< std::string> vxg::cloud::agent::supported_streams_config::video_es
```

list of string, camera video ES

Definition at line 1290 of file config.h.

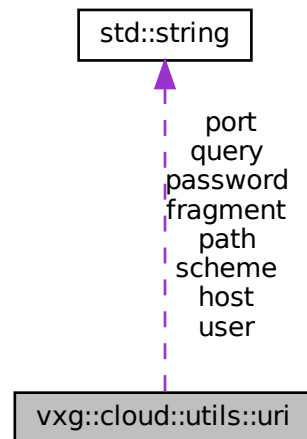
The documentation for this struct was generated from the following file:

- [config.h](#)

## 10.44 vxg::cloud::utils::uri Struct Reference

```
#include <utils/utils.h>
```

Collaboration diagram for vxg::cloud::utils::uri:



### Static Public Member Functions

- static bool [parse](#) (const **std::string** &in\_uri, [uri](#) &result)

### Data Fields

- **std::string** [scheme](#)
- **std::string** [user](#)
- **std::string** [password](#)
- **std::string** [host](#)
- **std::string** [port](#)
- **std::string** [path](#)
- **std::string** [query](#)
- **std::string** [fragment](#)

### 10.44.1 Detailed Description

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

### 10.44.2 Member Function Documentation

#### 10.44.2.1 parse()

```
static bool vxg::cloud::utils::uri::parse (  
    const std::string & in_uri,  
    uri & result ) [inline], [static]
```

Definition at line 76 of file utils.h.

### 10.44.3 Field Documentation

#### 10.44.3.1 fragment

```
std::string vxg::cloud::utils::uri::fragment
```

Definition at line 74 of file utils.h.

#### 10.44.3.2 host

```
std::string vxg::cloud::utils::uri::host
```

Definition at line 70 of file utils.h.

#### 10.44.3.3 password

```
std::string vxg::cloud::utils::uri::password
```

Definition at line 69 of file utils.h.

#### 10.44.3.4 path

```
std::string vxg::cloud::utils::uri::path
```

Definition at line 72 of file utils.h.



#### 10.44.3.5 port

```
std::string vxg::cloud::utils::uri::port
```

Definition at line 71 of file utils.h.

#### 10.44.3.6 query

```
std::string vxg::cloud::utils::uri::query
```

Definition at line 73 of file utils.h.

#### 10.44.3.7 scheme

```
std::string vxg::cloud::utils::uri::scheme
```

Definition at line 67 of file utils.h.

#### 10.44.3.8 user

```
std::string vxg::cloud::utils::uri::user
```

Definition at line 68 of file utils.h.

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

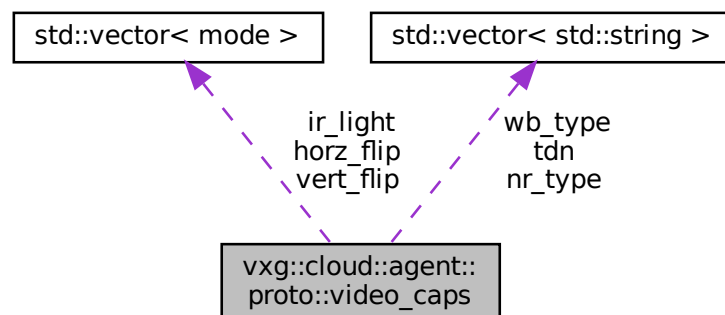
- [utils.h](#)

## 10.45 vxg::cloud::agent::proto::video\_caps Struct Reference

Video image capabilities.

```
#include <agent-proto/objects/caps.h>
```

Collaboration diagram for vxg::cloud::agent::proto::video\_caps:



## Data Fields

- **std::vector< mode > vert\_flip**  
*vert\_flip: list of string, supported vertical flip modes, possible values ["off", "on", "auto"]*
- **std::vector< mode > horz\_flip**  
*horz\_flip: list of string, supported horizontal flip modes, possible values ["off", "on", "auto"]*
- **std::vector< std::string > tdn**  
*tdn: list of string, supported TDM modes, possible values ["day", "night", "auto"]*
- **std::vector< mode > ir\_light**  
*ir\_light: list of string, supported IR light modes, possible values ["off", "on", "auto"]*
- **bool brightness**  
*brightness: bool, True when camera supports brightness control*
- **bool contrast**  
*contrast: bool, True when camera supports contrast control*
- **bool saturation**  
*saturation: bool, True when camera supports saturation control*
- **bool sharpness**  
*sharpness: bool, True when camera supports sharpness control*
- **std::vector< std::string > nr\_type**  
*nr\_type: list of string, supported noise reduce types.*
- **bool nr\_level**  
*nr\_level: bool, True when noise reduce filter assumes control of NR level*
- **std::vector< std::string > wb\_type**  
*wb\_type: list of string, supported white balance types.*
- **bool pwr\_frequency**  
*pwr\_frequency: bool, True camera supports compensation of images flickering due to flashing of lamps in indoor environment*

### 10.45.1 Detailed Description

Video image capabilities.

Definition at line 366 of file caps.h.

### 10.45.2 Field Documentation

#### 10.45.2.1 brightness

```
bool vxg::cloud::agent::proto::video_caps::brightness
```

brightness: bool, True when camera supports brightness control

Definition at line 384 of file caps.h.

### 10.45.2.2 contrast

`bool vxg::cloud::agent::proto::video_caps::contrast`

contrast: bool, True when camera supports contrast control

Definition at line 387 of file caps.h.

### 10.45.2.3 horz\_flip

`std::vector<mode> vxg::cloud::agent::proto::video_caps::horz_flip`

horz\_flip: list of string, supported horizontal flip modes, possible values ["off", "on", "auto"]

Definition at line 373 of file caps.h.

### 10.45.2.4 ir\_light

`std::vector<mode> vxg::cloud::agent::proto::video_caps::ir_light`

ir\_light: list of string, supported IR light modes, possible values ["off", "on", "auto"]

Definition at line 381 of file caps.h.

### 10.45.2.5 nr\_level

`bool vxg::cloud::agent::proto::video_caps::nr_level`

nr\_level: bool, True when noise reduce filter assumes control of NR level

Definition at line 402 of file caps.h.

### 10.45.2.6 nr\_type

`std::vector< std::string> vxg::cloud::agent::proto::video_caps::nr_type`

nr\_type: list of string, supported noise reduce types.

Empty list when camera doesn't support it. Example: ["off", "normal", "expert"]

Definition at line 398 of file caps.h.

#### 10.45.2.7 pwr\_frequency

```
bool vxg::cloud::agent::proto::video_caps::pwr_frequency
```

pwr\_frequency: bool, True camera supports compensation of images flickering due to flashing of lamps in indoor environment

Definition at line 411 of file caps.h.

#### 10.45.2.8 saturation

```
bool vxg::cloud::agent::proto::video_caps::saturation
```

saturation: bool, True when camera supports saturation control

Definition at line 390 of file caps.h.

#### 10.45.2.9 sharpness

```
bool vxg::cloud::agent::proto::video_caps::sharpness
```

sharpness: bool, True when camera supports sharpness control

Definition at line 393 of file caps.h.

#### 10.45.2.10 tdn

```
std::vector< std::string> vxg::cloud::agent::proto::video_caps::tdn
```

tdn: list of string, supported TDM modes, possible values ["day", "night", "auto"]

Definition at line 377 of file caps.h.

#### 10.45.2.11 vert\_flip

```
std::vector<mode> vxg::cloud::agent::proto::video_caps::vert_flip
```

vert\_flip: list of string, supported vertical flip modes, possible values ["off", "on", "auto"]

Definition at line 369 of file caps.h.

## 10.45.2.12 wb\_type

```
std::vector< std::string> vxg::cloud::agent::proto::video_caps::wb_type
```

wb\_type: list of string, supported white balance types.

Empty list when camera doesn't support it. Example: ["auto", "3200K (Indor)", "4200K (Fluo)", "5600K (Outdoor)"]

Definition at line 407 of file caps.h.

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

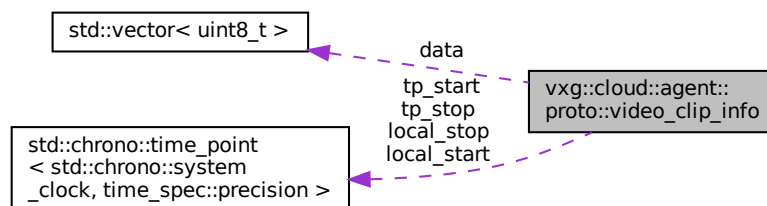
- [caps.h](#)

## 10.46 vxg::cloud::agent::proto::video\_clip\_info Struct Reference

Video recoding(mp4 file) clip description,.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::video\_clip\_info:



## Data Fields

- [cloud::time tp\\_start](#)  
*Clip start time UTC.*
- [cloud::time tp\\_stop](#)  
*Clip stop time UTC.*
- [cloud::time local\\_start](#)  
*Clip start time local.*
- [cloud::time local\\_stop](#)  
*Clip stop time local.*
- int [video\\_width](#)  
*Video clip picture width.*
- int [video\\_height](#)  
*Video clip picture height.*
- **std::vector< uint8\_t >** [data](#)  
*Video data buffer, we use move semantics internally so no data copying will be invoked.*

### 10.46.1 Detailed Description

Video recoding(mp4 file) clip description,.

Definition at line 452 of file config.h.

### 10.46.2 Field Documentation

#### 10.46.2.1 data

```
std::vector<uint8_t> vxg::cloud::agent::proto::video_clip_info::data
```

Video data buffer, we use move semantics internally so no data copying will be invoked.

Definition at line 478 of file config.h.

#### 10.46.2.2 local\_start

```
cloud::time vxg::cloud::agent::proto::video_clip_info::local_start
```

Clip start time local.

Definition at line 466 of file config.h.

#### 10.46.2.3 local\_stop

```
cloud::time vxg::cloud::agent::proto::video_clip_info::local_stop
```

Clip stop time local.

Definition at line 469 of file config.h.

#### 10.46.2.4 tp\_start

```
cloud::time vxg::cloud::agent::proto::video_clip_info::tp_start
```

Clip start time UTC.

Definition at line 461 of file config.h.

### 10.46.2.5 tp\_stop

```
cloud::time vxg::cloud::agent::proto::video_clip_info::tp_stop
```

Clip stop time UTC.

Definition at line 463 of file config.h.

### 10.46.2.6 video\_height

```
int vxg::cloud::agent::proto::video_clip_info::video_height
```

Video clip picture height.

Definition at line 474 of file config.h.

### 10.46.2.7 video\_width

```
int vxg::cloud::agent::proto::video_clip_info::video_width
```

Video clip picture width.

Definition at line 472 of file config.h.

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

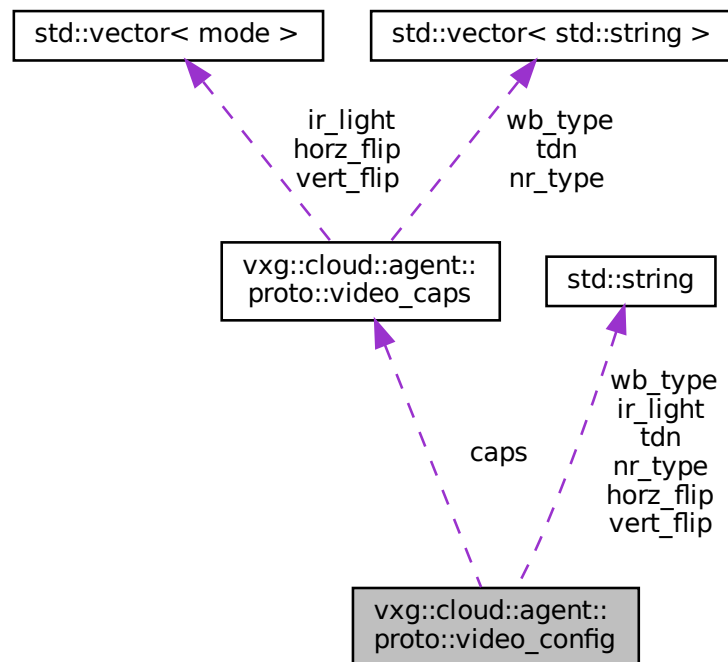
- [config.h](#)

## 10.47 vxg::cloud::agent::proto::video\_config Struct Reference

Video image config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::video\_config:



## Data Fields

- **std::string** [vert\\_flip](#)  
*vert\_flip: optional string, vertical image flip mode: ["off", "on", "auto"]*
- **std::string** [horz\\_flip](#)  
*horz\_flip: optional string, horizontal image flip mode: ["off", "on", "auto"]*
- **std::string** [tdn](#)  
*tdn: optional string, possible values ["day", "night", "auto"]*
- **std::string** [ir\\_light](#)  
*ir\_light: optional string, IR light for night conditions ["off", "on", "auto"]*
- **int** [brightness](#)  
*brightness: optional int, a brightness value from range 0-100 (%)*
- **int** [contrast](#)  
*contrast: optional int, a contrast value from range 0-100 (%)*
- **int** [saturation](#)  
*saturation: optional int, a saturation value from range 0-100 (%)*
- **int** [sharpness](#)  
*sharpness: optional int, a sharpness value from range 0-100 (%)*
- **std::string** [nr\\_type](#)  
*nr\_type: optional string, one of predefined noise reduce types from caps*
- **int** [nr\\_level](#)  
*nr\_level: optional int, level of noise reduce when filter requires it 0-100 (%)*
- **std::string** [wb\\_type](#)



- wb\_type: optional string, one of predefined white balance types from caps*
- int [pwr\\_frequency](#)  
*pwr\_frequency: optional int, power line frequency [50, 60] (Hz)*
- [video\\_caps caps](#)  
*caps*

### 10.47.1 Detailed Description

Video image config.

Definition at line 309 of file config.h.

### 10.47.2 Field Documentation

#### 10.47.2.1 brightness

```
int vxg::cloud::agent::proto::video_config::brightness
```

brightness: optional int, a brightness value from range 0-100 (%)

Definition at line 326 of file config.h.

#### 10.47.2.2 caps

```
video\_caps vxg::cloud::agent::proto::video_config::caps
```

caps

Definition at line 352 of file config.h.

#### 10.47.2.3 contrast

```
int vxg::cloud::agent::proto::video_config::contrast
```

contrast: optional int, a contrast value from range 0-100 (%)

Definition at line 329 of file config.h.

#### 10.47.2.4 horz\_flip

```
std::string vxg::cloud::agent::proto::video_config::horz_flip
```

horz\_flip: optional string, horizontal image flip mode: ["off", "on", "auto"]

Definition at line 316 of file config.h.

#### 10.47.2.5 ir\_light

```
std::string vxg::cloud::agent::proto::video_config::ir_light
```

ir\_light: optional string, IR light for night conditions ["off", "on", "auto"]

Definition at line 323 of file config.h.

#### 10.47.2.6 nr\_level

```
int vxg::cloud::agent::proto::video_config::nr_level
```

nr\_level: optional int, level of noise reduce when filter requires it 0-100 (%)

Definition at line 342 of file config.h.

#### 10.47.2.7 nr\_type

```
std::string vxg::cloud::agent::proto::video_config::nr_type
```

nr\_type: optional string, one of predefined noise reduce types from caps

Definition at line 338 of file config.h.

#### 10.47.2.8 pwr\_frequency

```
int vxg::cloud::agent::proto::video_config::pwr_frequency
```

pwr\_frequency: optional int, power line frequency [50, 60] (Hz)

Definition at line 349 of file config.h.

### 10.47.2.9 saturation

```
int vxg::cloud::agent::proto::video_config::saturation
```

saturation: optional int, a saturation value from range 0-100 (%)

Definition at line 332 of file config.h.

### 10.47.2.10 sharpness

```
int vxg::cloud::agent::proto::video_config::sharpness
```

sharpness: optional int, a sharpness value from range 0-100 (%)

Definition at line 335 of file config.h.

### 10.47.2.11 tdn

```
std::string vxg::cloud::agent::proto::video_config::tdn
```

tdn: optional string, possible values ["day", "night", "auto"]

Definition at line 319 of file config.h.

### 10.47.2.12 vert\_flip

```
std::string vxg::cloud::agent::proto::video_config::vert_flip
```

vert\_flip: optional string, vertical image flip mode: ["off", "on", "auto"]

Definition at line 312 of file config.h.

### 10.47.2.13 wb\_type

```
std::string vxg::cloud::agent::proto::video_config::wb_type
```

wb\_type: optional string, one of predefined white balance types from caps

Definition at line 346 of file config.h.

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

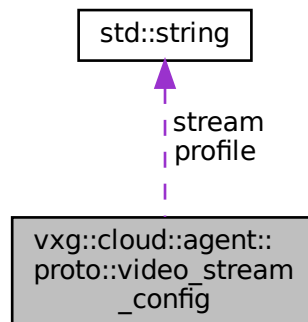
- [config.h](#)

## 10.48 vxg::cloud::agent::proto::video\_stream\_config Struct Reference

Video stream config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::video\_stream\_config:



### Data Fields

- **std::string** `stream`  
Mandatory: video ES to use.
- **video\_format** `format`  
Mandatory: video encoding format.
- **std::string** `profile`  
Optional: profile that specifies format, when format assumes it.
- **int** `horz`  
Mandatory: int (horz) - video resolution width x height.
- **int** `vert`  
Mandatory: int (vert) - video resolution width x height.
- **double** `fps`  
Mandatory: framerate.
- **bool** `vbr`  
Mandatory: prefer VBR; if false or not set CBR should be used.
- **int** `gop`  
Mandatory: gop size (I-Frame interval);.
- **int** `brt`  
Optional: bitrate, kbps.
- **int** `vbr_brt`  
Optional: bitrate for VBR, kbps.
- **int** `quality`  
Optional: int [-4..4], quality profile hint for encoder, where 0 means normal.
- **int** `smoothing`  
Optional: a smoothing value from range 0-100 (%)

## 10.48.1 Detailed Description

Video stream config.

Definition at line 86 of file config.h.

## 10.48.2 Field Documentation

### 10.48.2.1 brt

```
int vxg::cloud::agent::proto::video_stream_config::brt
```

Optional: bitrate, kbps.

Definition at line 120 of file config.h.

### 10.48.2.2 format

```
video_format vxg::cloud::agent::proto::video_stream_config::format
```

Mandatory: video encoding format.

Definition at line 93 of file config.h.

### 10.48.2.3 fps

```
double vxg::cloud::agent::proto::video_stream_config::fps
```

Mandatory: framerate.

Definition at line 108 of file config.h.

### 10.48.2.4 gop

```
int vxg::cloud::agent::proto::video_stream_config::gop
```

Mandatory: gop size (I-Frame interval);.

Definition at line 116 of file config.h.

#### 10.48.2.5 horz

```
int vxg::cloud::agent::proto::video_stream_config::horz
```

Mandatory: int (horz) - video resolution width x height.

Definition at line 101 of file config.h.

#### 10.48.2.6 profile

```
std::string vxg::cloud::agent::proto::video_stream_config::profile
```

Optional: profile that specifies format, when format assumes it.

Definition at line 97 of file config.h.

#### 10.48.2.7 quality

```
int vxg::cloud::agent::proto::video_stream_config::quality
```

Optional: int [-4..4], quality profile hint for encoder, where 0 means normal.

Definition at line 128 of file config.h.

#### 10.48.2.8 smoothing

```
int vxg::cloud::agent::proto::video_stream_config::smoothing
```

Optional: a smoothing value from range 0-100 (%)

Definition at line 132 of file config.h.

#### 10.48.2.9 stream

```
std::string vxg::cloud::agent::proto::video_stream_config::stream
```

Mandatory: video ES to use.

Definition at line 89 of file config.h.

**10.48.2.10 vbr**

```
bool vxg::cloud::agent::proto::video_stream_config::vbr
```

Mandatory: prefer VBR; if false or not set CBR should be used.

Definition at line 112 of file config.h.

**10.48.2.11 vbr\_brt**

```
int vxg::cloud::agent::proto::video_stream_config::vbr_brt
```

Optional: bitrate for VBR, kbps.

Definition at line 124 of file config.h.

**10.48.2.12 vert**

```
int vxg::cloud::agent::proto::video_stream_config::vert
```

Mandatory: int (vert) - video resolution width x height.

Definition at line 104 of file config.h.

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

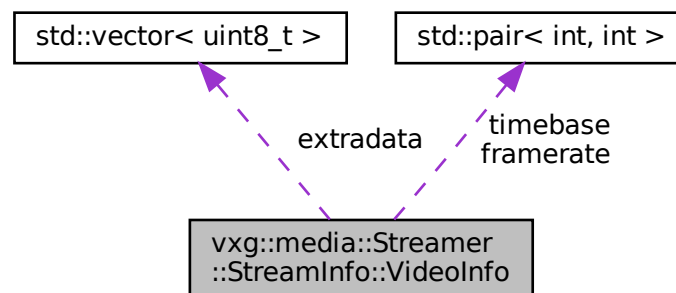
- [config.h](#)

**10.49 vxg::media::Streamer::StreamInfo::VideoInfo Struct Reference**

Video stream info.

```
#include <streamer/base_streamer.h>
```

Collaboration diagram for vxg::media::Streamer::StreamInfo::VideoInfo:



## Data Fields

- [VideoCodec codec](#)  
*Video codec type.*
- int [width](#)  
*Video width if needed.*
- int [height](#)  
*Video height if needed.*
- **std::pair**< int, int > [framerate](#)  
*Video framerate if needed.*
- int [bitrate](#)  
*Video bitrate if needed.*
- **std::pair**< int, int > [timebase](#)  
*Timescale of the timestamps, source fills it with timescale of timestamps source receives, [MediaFrame::pts](#) should use this timescale.*
- **std::vector**< uint8\_t > [extradata](#)  
*Can be AVC1 extradata or SPS/PPS, source fills it and sink should know and understand this format.*

### 10.49.1 Detailed Description

Video stream info.

This structure as well as [ISink::negotiate](#) method aimed to inform sink about streams source provides, if sink don't care the values of this structure may be ignored.

Definition at line 310 of file `base_streamer.h`.

### 10.49.2 Field Documentation

#### 10.49.2.1 bitrate

```
int vxg::media::Streamer::StreamInfo::VideoInfo::bitrate
```

Video bitrate if needed.

Definition at line 320 of file `base_streamer.h`.

#### 10.49.2.2 codec

```
VideoCodec vxg::media::Streamer::StreamInfo::VideoInfo::codec
```

Video codec type.

Definition at line 312 of file `base_streamer.h`.



### 10.49.2.3 extradata

```
std::vector<uint8_t> vxg::media::Streamer::StreamInfo::VideoInfo::extradata
```

Can be AVC1 extradata or SPS/PPS, source fills it and sink should know and understand this format.

Definition at line 327 of file base\_streamer.h.

### 10.49.2.4 framerate

```
std::pair<int, int> vxg::media::Streamer::StreamInfo::VideoInfo::framerate
```

Video framerate if needed.

Definition at line 318 of file base\_streamer.h.

### 10.49.2.5 height

```
int vxg::media::Streamer::StreamInfo::VideoInfo::height
```

Video height if needed.

Definition at line 316 of file base\_streamer.h.

### 10.49.2.6 timebase

```
std::pair<int, int> vxg::media::Streamer::StreamInfo::VideoInfo::timebase
```

Timescale of the timestamps, source fills it with timescale of timestamps source receives, [MediaFrame::pts](#) should use this timescale.

Definition at line 324 of file base\_streamer.h.

### 10.49.2.7 width

```
int vxg::media::Streamer::StreamInfo::VideoInfo::width
```

Video width if needed.

Definition at line 314 of file base\_streamer.h.

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

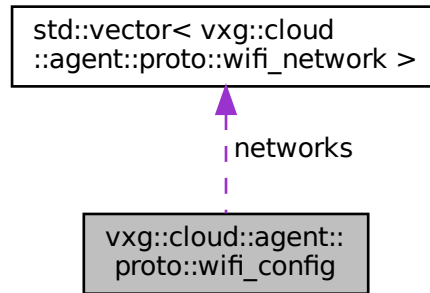
- [base\\_streamer.h](#)

## 10.50 vxg::cloud::agent::proto::wifi\_config Struct Reference

WiFi config.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::wifi\_config:



### Data Fields

- **std::vector< [wifi\\_network](#) > networks**  
List of [wifi\\_network](#) objects.

### 10.50.1 Detailed Description

WiFi config.

Definition at line 584 of file config.h.

### 10.50.2 Field Documentation

#### 10.50.2.1 networks

```
std::vector<wifi\_network> vxg::cloud::agent::proto::wifi_config::networks
```

List of [wifi\\_network](#) objects.

Definition at line 586 of file config.h.

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

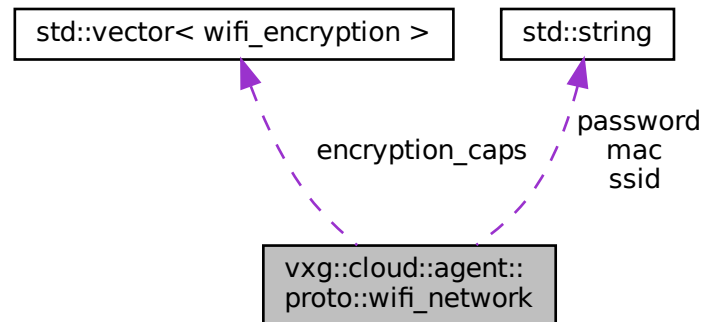
- [config.h](#)

## 10.51 vxg::cloud::agent::proto::wifi\_network Struct Reference

WiFi network object.

```
#include <agent-proto/objects/config.h>
```

Collaboration diagram for vxg::cloud::agent::proto::wifi\_network:



### Data Fields

- **std::string** [ssid](#)  
*ssid: string, network SSID*
- **int** [signal](#)  
*signal: int, signal strength, dB*
- **std::string** [mac](#)  
*mac: string, AP MAC address*
- **std::vector< [wifi\\_encryption](#) >** [encryption\\_caps](#)  
*encryption\_caps: list of string, supported encryption types,*
- [wifi\\_encryption](#) [encryption](#)  
*encryption: string, current encryption type, see encryption\_caps for possible values*
- **std::string** [password](#)  
*password: string, network password*

### 10.51.1 Detailed Description

WiFi network object.

Definition at line 555 of file config.h.

### 10.51.2 Field Documentation

### 10.51.2.1 encryption

`wifi_encryption` `vsg::cloud::agent::proto::wifi_network::encryption`

encryption: string, current encryption type, see `encryption_caps` for possible values

Definition at line 566 of file `config.h`.

### 10.51.2.2 encryption\_caps

`std::vector<wifi_encryption>` `vsg::cloud::agent::proto::wifi_network::encryption_caps`

encryption\_caps: list of string, supported encryption types,

Definition at line 563 of file `config.h`.

### 10.51.2.3 mac

`std::string` `vsg::cloud::agent::proto::wifi_network::mac`

mac: string, AP MAC address

Definition at line 561 of file `config.h`.

### 10.51.2.4 password

`std::string` `vsg::cloud::agent::proto::wifi_network::password`

password: string, network password

Definition at line 568 of file `config.h`.

### 10.51.2.5 signal

`int` `vsg::cloud::agent::proto::wifi_network::signal`

signal: int, signal strength, dB

Definition at line 559 of file `config.h`.

### 10.51.2.6 ssid

`std::string` `vsg::cloud::agent::proto::wifi_network::ssid`

ssid: string, network SSID

Definition at line 557 of file `config.h`.

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

- [config.h](#)

## Chapter 11

# File Documentation

### 11.1 app-dev.md File Reference

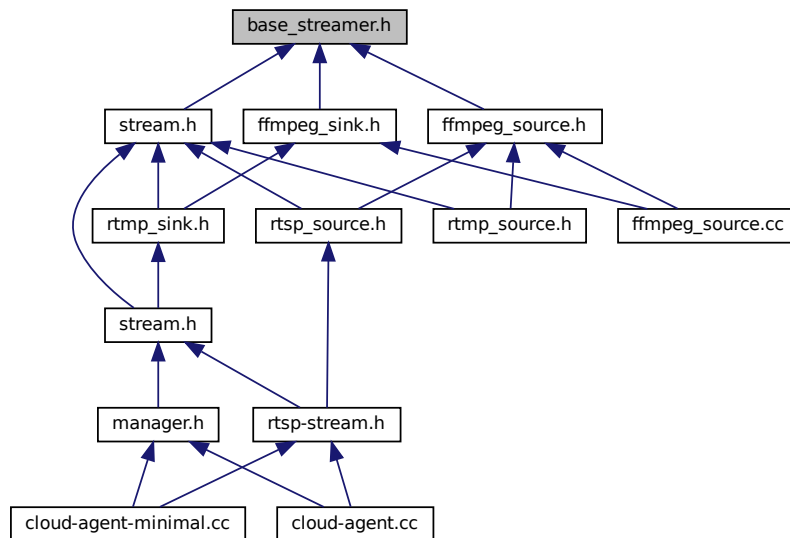
### 11.2 arm-example.txt File Reference

### 11.3 base\_streamer.h File Reference

```
#include <cstdlib>
#include <future>
#include <map>
#include <queue>
#include <string>
#include <pthread.h>
#include <streamer/stats.h>
#include <utils/logging.h>
#include <utils/utils.h>
Include dependency graph for base_streamer.h:
```



This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [vsg::media::Streamer::StreamInfo](#)  
*Stream info description.*
- struct [vsg::media::Streamer::StreamInfo::VideoInfo](#)  
*Video stream info.*
- struct [vsg::media::Streamer::StreamInfo::AudioInfo](#)  
*Audio stream info.*
- struct [vsg::media::Streamer::MediaFrame](#)  
*Media frame container.*
- class [vsg::media::Streamer::ISink](#)
- class [vsg::media::Streamer::ISource](#)  
*ISource interface class.*

## Namespaces

- [vsg](#)
- [vsg::media](#)
- [vsg::media::Streamer](#)

## Macros

- `#define \_\_BASE\_STREAMER\_H`

## Enumerations

- enum `vsg::media::Streamer::DropDirection` { `vsg::media::Streamer::DROP_FRONT`, `vsg::media::Streamer::DROP_BACK` }
- enum `vsg::media::Streamer::StreamError` { `vsg::media::Streamer::E_NONE`, `vsg::media::Streamer::E_FATAL`, `vsg::media::Streamer::E_EOS` }  
*Stream error.*
- enum `vsg::media::Streamer::MediaType` { `vsg::media::Streamer::UNKNOWN`, `vsg::media::Streamer::VIDEO`, `vsg::media::Streamer::VIDEO_AVC_SPS`, `vsg::media::Streamer::VIDEO_AVC_PPS`, `vsg::media::Streamer::VIDEO_SEQ_HDR`, `vsg::media::Streamer::AUDIO`, `vsg::media::Streamer::AUDIO_SEQ_HDR`, `vsg::media::Streamer::FLV`, `vsg::media::Streamer::DATA`, `vsg::media::Streamer::MAX` }  
*Media frame type.*

## Variables

- constexpr int `vsg::media::Streamer::SINK_THREAD_PRIO`
- constexpr int `vsg::media::Streamer::SRC_THREAD_PRIO`

### 11.3.1 Macro Definition Documentation

#### 11.3.1.1 \_\_BASE\_STREAMER\_H

```
#define __BASE_STREAMER_H
```

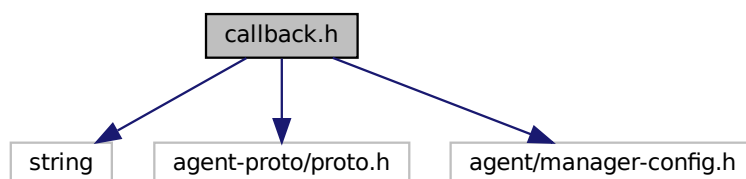
Definition at line 3 of file `base_streamer.h`.

## 11.4 build-system.md File Reference

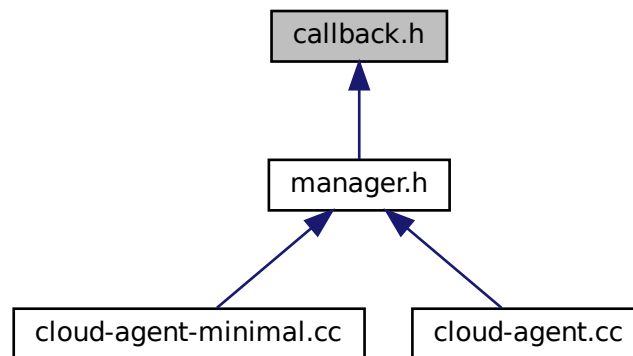
## 11.5 callback.h File Reference

```
#include <string>
#include <agent-proto/proto.h>
#include <agent/manager-config.h>
```

Include dependency graph for `callback.h`:



This graph shows which files directly or indirectly include this file:



## Data Structures

- class `vxg::cloud::agent::callback`  
*VXG Cloud manager common callbacks class.*

## Namespaces

- `vxg`
- `vxg::cloud`
- `vxg::cloud::agent`  
*VXG Cloud Agent namespace.*

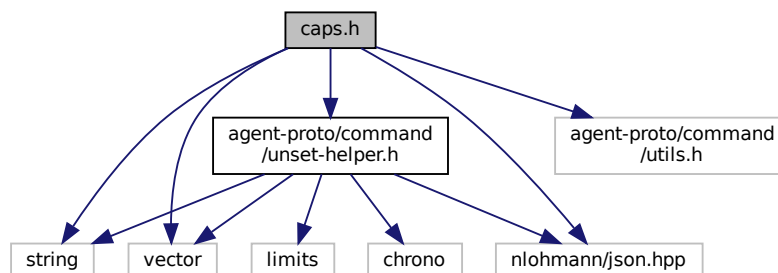
## 11.6 caps.h File Reference

```

#include <string>
#include <vector>
#include <nlohmann/json.hpp>
#include <agent-proto/command/unset-helper.h>
#include <agent-proto/command/utls.h>

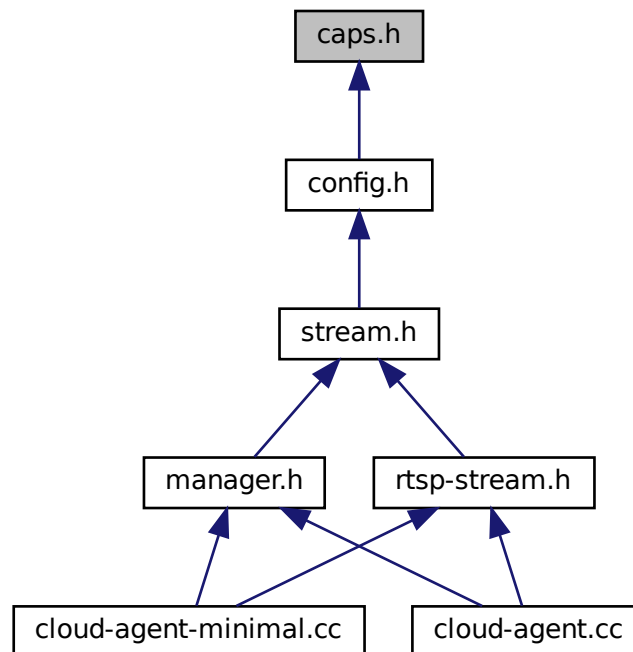
```

Include dependency graph for caps.h:





This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [vxg::cloud::agent::proto::stream\\_caps](#)  
*Media stream capabilities.*
- struct [vxg::cloud::agent::proto::stream\\_caps::caps\\_video\\_object](#)  
*Video streams capabilities.*
- struct [vxg::cloud::agent::proto::stream\\_caps::caps\\_audio\\_object](#)  
*Audio streams capabilities.*
- struct [vxg::cloud::agent::proto::motion\\_detection\\_caps](#)  
*Motion detection capabilities camera capabilities that limit possible motion detection configuration.*
- struct [vxg::cloud::agent::proto::video\\_caps](#)  
*Video image capabilities.*
- struct [vxg::cloud::agent::proto::event\\_caps](#)  
*Events capabilities.*
- struct [vxg::cloud::agent::proto::audio\\_caps](#)  
*Audio capabilities.*
- struct [vxg::cloud::agent::proto::osd\\_caps](#)  
*OSD capabilities.*

## Namespaces

- [vxg](#)
- [vxg::cloud](#)
- [vxg::cloud::agent](#)  
*VXG Cloud Agent namespace.*
- [vxg::cloud::agent::proto](#)

## Macros

- `#define ignore_exception(...)`

## Typedefs

- using `json` = `nlohmann::json`

## Enumerations

- enum `vxg::cloud::agent::proto::mode` { `vxg::cloud::agent::proto::M_OFF`, `vxg::cloud::agent::proto::M_ON`, `vxg::cloud::agent::proto::M_AUTO`, `vxg::cloud::agent::proto::M_INVALID` }  
*Mode on/off.*
  - enum `vxg::cloud::agent::proto::video_format` { `vxg::cloud::agent::proto::VF_H264`, `vxg::cloud::agent::proto::VF_H265`, `vxg::cloud::agent::proto::VF_MJPEG`, `vxg::cloud::agent::proto::VF_INVALID` }  
*Video codec format.*
  - enum `vxg::cloud::agent::proto::audio_format` { `vxg::cloud::agent::proto::AF_G711A`, `vxg::cloud::agent::proto::AF_G711U`, `vxg::cloud::agent::proto::AF_RAW`, `vxg::cloud::agent::proto::AF_ADPCM`, `vxg::cloud::agent::proto::AF_MP3`, `vxg::cloud::agent::proto::AF_NELLY8`, `vxg::cloud::agent::proto::AF_NELLY16`, `vxg::cloud::agent::proto::AF_NELLY`, `vxg::cloud::agent::proto::AF_OPUS`, `vxg::cloud::agent::proto::AF_AAC`, `vxg::cloud::agent::proto::AF_SPEEX`, `vxg::cloud::agent::proto::AF_INVALID` }  
*Audio codec format.*
  - enum `vxg::cloud::agent::proto::audio_file_format` { `vxg::cloud::agent::proto::AFF_AU_G711U`, `vxg::cloud::agent::proto::AFF_MP3`, `vxg::cloud::agent::proto::AFF_WAV_PCM`, `vxg::cloud::agent::proto::AFF_INVALID` }  
*Audio file format.*
  - enum `vxg::cloud::agent::proto::motion_sensitivity` { `vxg::cloud::agent::proto::MS_REGION`, `vxg::cloud::agent::proto::MS_FRAME`, `vxg::cloud::agent::proto::MS_INVALID` }  
*Motion sensitivity.*
  - enum `vxg::cloud::agent::proto::motion_region_shape` { `vxg::cloud::agent::proto::MR_RECTANGLE`, `vxg::cloud::agent::proto::MR_ANY`, `vxg::cloud::agent::proto::MR_INVALID` }  
*Motion region shape.*
  - enum `vxg::cloud::agent::proto::ptz_action` { `vxg::cloud::agent::proto::A_LEFT`, `vxg::cloud::agent::proto::A_RIGHT`, `vxg::cloud::agent::proto::A_TOP`, `vxg::cloud::agent::proto::A_BOTTOM`, `vxg::cloud::agent::proto::A_ZOOM_IN`, `vxg::cloud::agent::proto::A_ZOOM_OUT`, `vxg::cloud::agent::proto::A_STOP`, `vxg::cloud::agent::proto::A_INVALID` }  
*PTZ actions.*
  - enum `vxg::cloud::agent::proto::ptz_preset_action` { `vxg::cloud::agent::proto::PA_CREATE`, `vxg::cloud::agent::proto::PA_DELETE`, `vxg::cloud::agent::proto::PA_GOTO`, `vxg::cloud::agent::proto::PA_UPDATE`, `vxg::cloud::agent::proto::PA_INVALID` }  
*PTZ preset action.*
  - enum `vxg::cloud::agent::proto::time_format_n` { `vxg::cloud::agent::proto::TF_12H`, `vxg::cloud::agent::proto::TF_24H`, `vxg::cloud::agent::proto::TF_INVALID` }
- 3.34 `get_osd_conf` (SRV) 3.35 `osd_conf` (CM) 3.36 `set_osd_conf` (SRV)

### 11.6.1 Macro Definition Documentation

### 11.6.1.1 ignore\_exception

```
#define ignore_exception(
    ... )
```

Definition at line 20 of file caps.h.

## 11.6.2 Typedef Documentation

### 11.6.2.1 json

```
using json = nlohmann::json
```

Definition at line 12 of file caps.h.

## 11.7 cloud-agent-minimal.cc File Reference

```
#include <signal.h>
#include <args.hxx>
#include <agent/manager.h>
#include <agent/rtsp-stream.h>
#include <utils/logging.h>
#include <utils/properties.h>
Include dependency graph for cloud-agent-minimal.cc:
```



## Functions

- static void [signal\\_handler](#) (int sig)
- bool [parse\\_args](#) (int argc, char \*\*argv)
- int [main](#) (int argc, char \*\*argv)

## Variables

- static bool [quit](#)  
[Includes and namespaces]
- static vxg::properties [props](#)
- **std::string** [vxg\\_cloud\\_token](#)  
[Minimal callback class implementation]
- **std::string** [rtsp\\_url](#)

## 11.7.1 Function Documentation

### 11.7.1.1 `main()`

```
int main (  
    int argc,  
    char ** argv )
```

[Create and start agent object]

[Create and start agent object]

[Stop agent]

[Stop agent]

Definition at line 85 of file cloud-agent-minimal.cc.

### 11.7.1.2 `parse_args()`

```
bool parse_args (  
    int argc,  
    char ** argv )
```

Definition at line 46 of file cloud-agent-minimal.cc.

### 11.7.1.3 `signal_handler()`

```
static void signal_handler (  
    int sig ) [static]
```

Definition at line 18 of file cloud-agent-minimal.cc.

## 11.7.2 Variable Documentation

### 11.7.2.1 `props`

```
vxg::properties props [static]
```

Definition at line 16 of file cloud-agent-minimal.cc.

### 11.7.2.2 quit

```
bool quit [static]
```

[Includes and namespaces]

Definition at line 15 of file cloud-agent-minimal.cc.

### 11.7.2.3 rtsp\_url

```
std::string rtsp_url
```

Definition at line 44 of file cloud-agent-minimal.cc.

### 11.7.2.4 vxg\_cloud\_token

```
std::string vxg_cloud_token
```

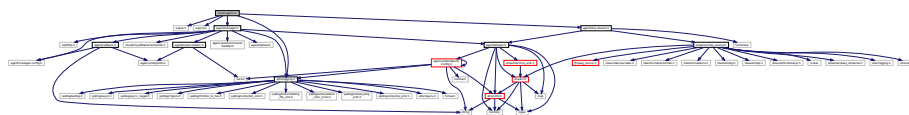
[Minimal callback class implementation]

Definition at line 43 of file cloud-agent-minimal.cc.

## 11.8 cloud-agent.cc File Reference

```
#include <signal.h>
#include <args.hxx>
#include <agent/manager.h>
#include <agent/rtsp-stream.h>
#include <utils/logging.h>
```

Include dependency graph for cloud-agent.cc:



## Functions

- static void [signal\\_handler](#) (int sig)
- bool [parse\\_args](#) (int argc, char \*\*argv)
- int [main](#) (int argc, char \*\*argv)

## Variables

- static bool `quit`  
*[Includes and namespaces]*
- `std::string` `vxg_cloud_token`  
*[Event stream callback class implementation]*
- `std::string` `rtsp_url`

## 11.8.1 Function Documentation

### 11.8.1.1 `main()`

```
int main (  
    int argc,  
    char ** argv )
```

[Create and start agent object]

[Create and start agent object]

[Stop agent]

[Stop agent]

Definition at line 349 of file cloud-agent.cc.

### 11.8.1.2 `parse_args()`

```
bool parse_args (  
    int argc,  
    char ** argv )
```

Definition at line 317 of file cloud-agent.cc.

### 11.8.1.3 `signal_handler()`

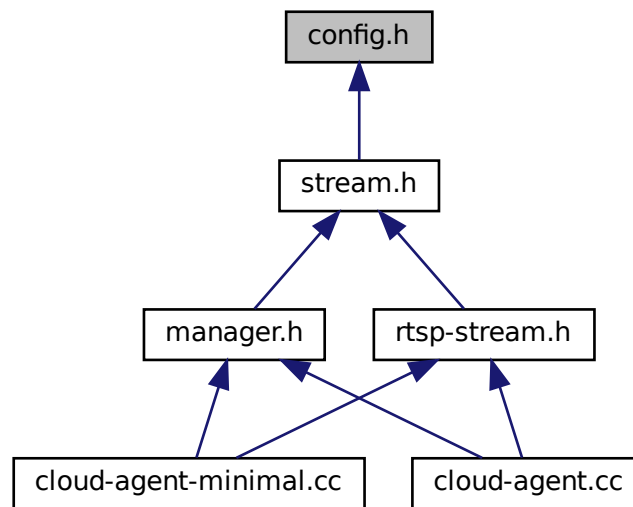
```
static void signal_handler (  
    int sig ) [static]
```

Definition at line 17 of file cloud-agent.cc.

## 11.8.2 Variable Documentation



This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [vxg::cloud::agent::proto::video\\_stream\\_config](#)  
*Video stream config.*
- struct [vxg::cloud::agent::proto::audio\\_stream\\_config](#)  
*Audio media stream config.*
- struct [vxg::cloud::agent::proto::stream\\_config](#)  
*Media stream config.*
- struct [vxg::cloud::agent::proto::motion\\_region](#)  
*Motion detection related structs.*
- struct [vxg::cloud::agent::proto::motion\\_detection\\_config](#)  
*Motion detection config.*
- struct [vxg::cloud::agent::proto::video\\_config](#)  
*Video image config.*
- struct [vxg::cloud::agent::proto::video\\_clip\\_info](#)  
*Video recoding(mp4 file) clip description,.*
- struct [vxg::cloud::agent::proto::wifi\\_network](#)  
*WiFi network object.*
- struct [vxg::cloud::agent::proto::wifi\\_config](#)  
*WiFi config.*
- struct [vxg::cloud::agent::event\\_config](#)  
*Event config.*
- struct [vxg::cloud::agent::events\\_config](#)  
*Events config, list of [event\\_config](#) objects.*
- struct [vxg::cloud::agent::audio\\_config](#)  
*Audio config.*
- struct [vxg::cloud::agent::ptz\\_preset](#)



- PTZ preset.*
- struct [vxd::cloud::agent::ptz\\_config](#)
- PTZ config.*
- struct [vxd::cloud::agent::ptz\\_command](#)
- PTZ command.*
- struct [vxd::cloud::agent::osd\\_config](#)
- OSD config.*
- struct [vxd::cloud::agent::access\\_token](#)
- VXD Cloud access token.*
- struct [vxd::cloud::agent::access\\_token::proxy\\_config](#)
- Socks proxy settings.*
- struct [vxd::cloud::agent::supported\\_stream\\_config](#)
- Supported stream config.*
- struct [vxd::cloud::agent::supported\\_streams\\_config](#)
- Supported streams config, list of [supported\\_stream\\_config](#).*

## Namespaces

- [vxd](#)
- [vxd::cloud](#)
- [vxd::cloud::time\\_spec](#)
- time point*
- [nlohmann](#)
- [vxd::cloud::agent](#)
- VXD Cloud Agent namespace.*
- [vxd::cloud::agent::proto](#)

## Macros

- `#define \_\_CONFIG\_H\_\_`

## Typedefs

- using [vxd::cloud::time\\_spec::precision](#) = `std::chrono::nanoseconds`
- template<typename T >  
using [vxd::cloud::time\\_spec::duration](#) = typename `std::conditional< std::is_same< T, precision >::value, precision, std::chrono::duration< T > >::type`
- using [vxd::cloud::time](#) = `std::chrono::time_point< std::chrono::system_clock, time_spec::precision >`
- using [vxd::cloud::duration](#) = `time_spec::duration< time_spec::precision >`
- typedef `wifi_config` [vxd::cloud::agent::proto::wifi\\_list](#)
- wifi\_config*

## Enumerations

- enum `vxg::cloud::agent::proto::event_status` { `vxg::cloud::agent::proto::ES_OK`, `vxg::cloud::agent::proto::ES_ERROR`, `vxg::cloud::agent::proto::ES_INVALID` }  
*Event status.*
- enum `vxg::cloud::agent::proto::event_type` { `vxg::cloud::agent::proto::ET_MOTION`, `vxg::cloud::agent::proto::ET_SOUND`, `vxg::cloud::agent::proto::ET_NET`, `vxg::cloud::agent::proto::ET_RECORD`, `vxg::cloud::agent::proto::ET_MEMORYCARD`, `vxg::cloud::agent::proto::ET_WIFI`, `vxg::cloud::agent::proto::ET_CUSTOM`, `vxg::cloud::agent::proto::ET_INVALID` }  
*Types of events.*
- enum `vxg::cloud::agent::proto::memorycard_status` { `vxg::cloud::agent::proto::MCS_NONE`, `vxg::cloud::agent::proto::MCS_NORMAL`, `vxg::cloud::agent::proto::MCS_NEED_FORMAT`, `vxg::cloud::agent::proto::MCS_FORMATTING`, `vxg::cloud::agent::proto::MCS_INITIALIZATION`, `vxg::cloud::agent::proto::MCS_INVALID` }  
*Memory card status.*
- enum `vxg::cloud::agent::proto::wifi_encryption` { `vxg::cloud::agent::proto::WFE_OPEN`, `vxg::cloud::agent::proto::WFE_WEP`, `vxg::cloud::agent::proto::WFE_WPA`, `vxg::cloud::agent::proto::WFE_WPA2`, `vxg::cloud::agent::proto::WFE_WPA_ENTERPRISE`, `vxg::cloud::agent::proto::WFE_WPA2_ENTERPRISE`, `vxg::cloud::agent::proto::WFE_INVALID` }  
*WiFi encryption type.*
- enum `vxg::cloud::agent::proto::wifi_network_state` { `vxg::cloud::agent::proto::WNS_UNKNOWN`, `vxg::cloud::agent::proto::WNS_INITIALIZE_0`, `vxg::cloud::agent::proto::WNS_INITIALIZE_1`, `vxg::cloud::agent::proto::WNS_TRY_CONNECT`, `vxg::cloud::agent::proto::WNS_RECEIVING_IP`, `vxg::cloud::agent::proto::WNS_CONNECTED`, `vxg::cloud::agent::proto::WNS_DISCONNECTED` }  
*WiFi connection state.*

## Functions

- `std::string vxg::cloud::agent::proto::name ()`

### 11.10.1 Detailed Description

VXG Cloud CM protocol objects

### 11.10.2 Macro Definition Documentation

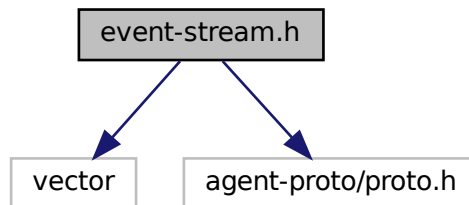
#### 11.10.2.1 \_\_CONFIG\_H\_\_

```
#define __CONFIG_H__
```

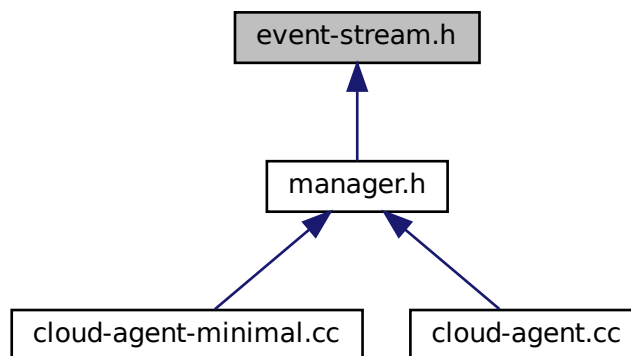
Definition at line 4 of file config.h.

## 11.11 event-stream.h File Reference

```
#include <vector>
#include <agent-proto/proto.h>
Include dependency graph for event-stream.h:
```



This graph shows which files directly or indirectly include this file:



### Data Structures

- class [vxg::cloud::agent::event\\_stream](#)  
*Event stream, abstract class for event generation.*

### Namespaces

- [vxg](#)
- [vxg::cloud](#)
- [vxg::cloud::agent](#)  
*VXG Cloud Agent namespace.*

## 11.12 ffmpeg\_sink.h File Reference

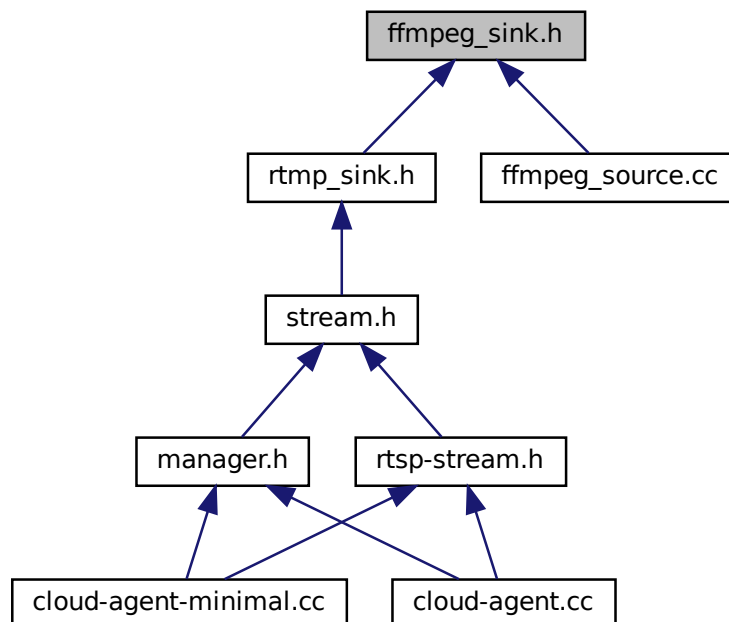
```
#include "base_streamer.h"
```

```
#include "ffmpeg_common.h"
```

Include dependency graph for ffmpeg\_sink.h:



This graph shows which files directly or indirectly include this file:



### Data Structures

- class [vxg::media::ffmpeg::Sink](#)  
*Base ffmpeg sink class.*

### Namespaces

- [vxg](#)
- [vxg::media](#)
- [vxg::media::ffmpeg](#)

### 11.13 ffmpeg\_source.cc File Reference

```
#include <streamer/ffmpeg_sink.h>
#include <streamer/ffmpeg_source.h>
#include <iomanip>
#include <iostream>
```

## Data Structures

- class `vvg::media::ffmpeg::Source`  
*Base ffmpeg source class.*

## Namespaces

- `vvg`
- `vvg::media`
- `vvg::media::ffmpeg`

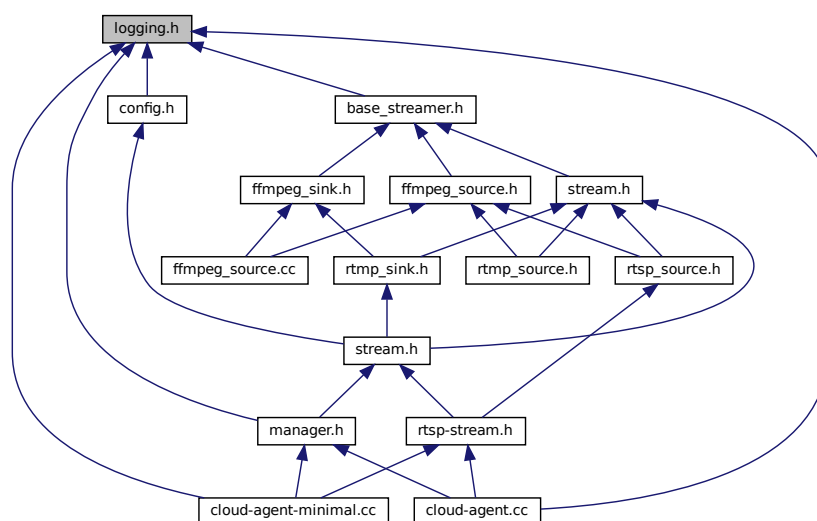
## 11.15 logging.h File Reference

```
#include <spdlog/spdlog.h>
#include <spdlog/async.h>
#include <spdlog/async_logger.h>
#include <spdlog/cfg/env.h>
#include <spdlog/fmt/bin_to_hex.h>
#include <spdlog/sinks/dist_sink.h>
#include <spdlog/sinks/rotating_file_sink.h>
#include <spdlog/sinks/stdout_color_sinks.h>
#include <spdlog/sinks/syslog_sink.h>
#include <spdlog/sinks/tcp_sink.h>
#include <utils/loguru.h>
#include <fstream>
```

Include dependency graph for logging.h:



This graph shows which files directly or indirectly include this file:



## Data Structures

- class `vsg::logger`  
*Logger class, current implementation based on spdlog.*
- struct `vsg::logger::options`

## Namespaces

- `vsg`

## 11.16 mainpage.md File Reference

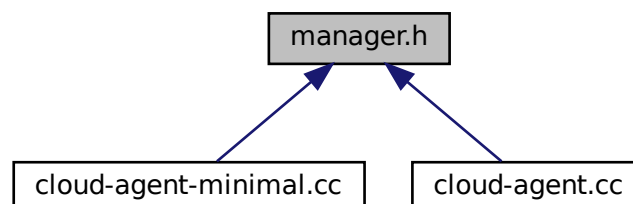
## 11.17 manager.h File Reference

```
#include <agent-proto/command-handler.h>
#include <agent/callback.h>
#include <agent/event-stream.h>
#include <agent/manager-config.h>
#include <cloud/CloudShareConnection.h>
#include <agent/stream.h>
#include <agent/upload.h>
#include <net/http.h>
#include <utils/logging.h>
```

Include dependency graph for manager.h:



This graph shows which files directly or indirectly include this file:









## Data Structures

- class [vsg::media::rtmp\\_source](#)  
*RTMP source class.*

## Namespaces

- [vsg](#)
- [vsg::media](#)

### 11.20.1 Detailed Description

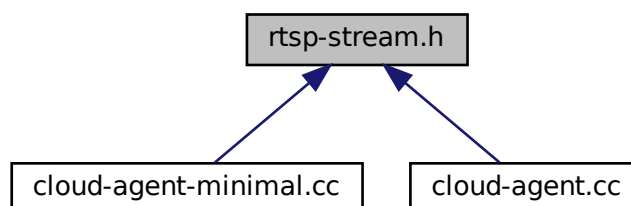
RTMP source

## 11.21 rtsp-stream.h File Reference

```
#include <functional>
#include <agent/stream.h>
#include <streamer/rtsp_source.h>
Include dependency graph for rtsp-stream.h:
```



This graph shows which files directly or indirectly include this file:



## Data Structures

- class [vsg::cloud::agent::media::rtsp\\_stream](#)  
*Implementation of the [media::stream](#) with RTSP source and NIY stubs.*

## Namespaces

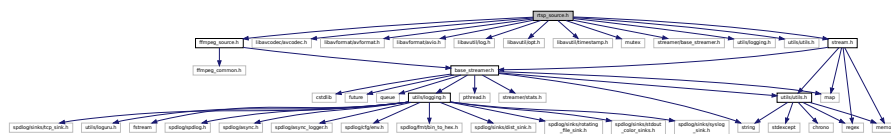
- [vsg](#)
- [vsg::cloud](#)
- [vsg::cloud::agent](#)  
VXG Cloud Agent namespace.
- [vsg::cloud::agent::media](#)

## 11.22 rtsp\_source.h File Reference

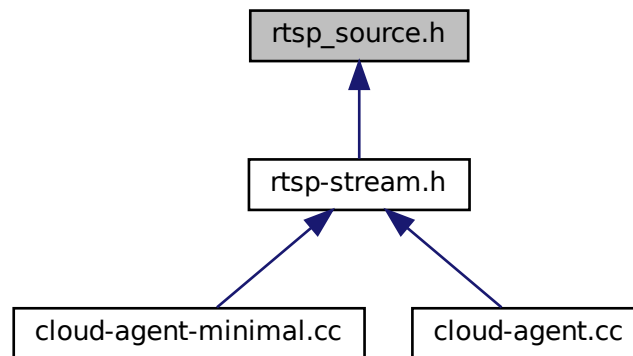
```
#include "ffmpeg_source.h"
```

```
#include "stream.h"
```

Include dependency graph for rtsp\_source.h:



This graph shows which files directly or indirectly include this file:



## Data Structures

- class [vsg::media::rtsp\\_source](#)  
RTSP source class.

## Namespaces

- [vsg](#)
- [vsg::media](#)

### 11.22.1 Detailed Description

RTSP source

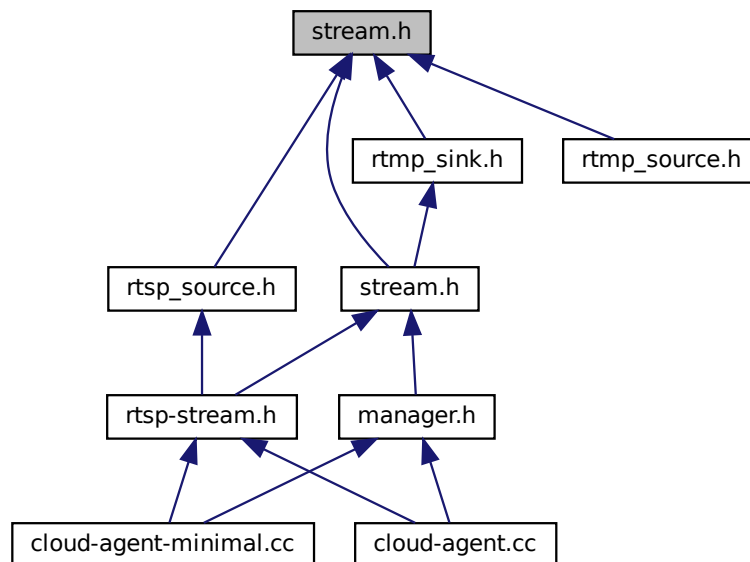
### 11.23 stream.h File Reference

```
#include <map>
#include <memory>
#include <regex>
#include <streamer/base_streamer.h>
#include <utils/utils.h>
```

Include dependency graph for streamer/stream.h:



This graph shows which files directly or indirectly include this file:



### Data Structures

- class [vsg::media::stream](#)  
*base media stream abstract class*

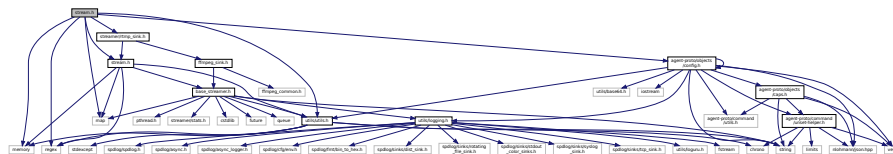
## Namespaces

- [vsg](#)
- [vsg::media](#)

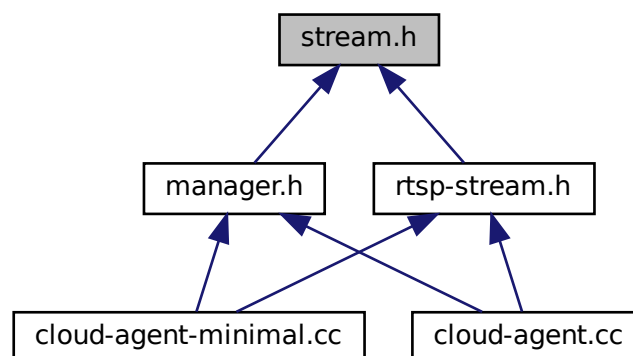
## 11.24 stream.h File Reference

```
#include <map>
#include <memory>
#include <regex>
#include <agent-proto/objects/config.h>
#include <streamer/rtpm_sink.h>
#include <streamer/stream.h>
#include <utils/utils.h>
```

Include dependency graph for agent/stream.h:



This graph shows which files directly or indirectly include this file:



## Data Structures

- class [vsg::cloud::agent::media::stream](#)  
*Cloud agent media stream abstract class.*

## Namespaces

- [vsg](#)
- [vsg::cloud](#)
- [vsg::cloud::agent](#)

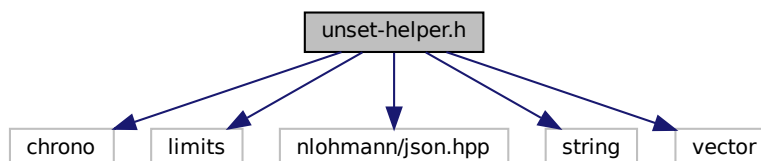
*VXG Cloud Agent namespace.*

- [vsg::cloud::agent::media](#)

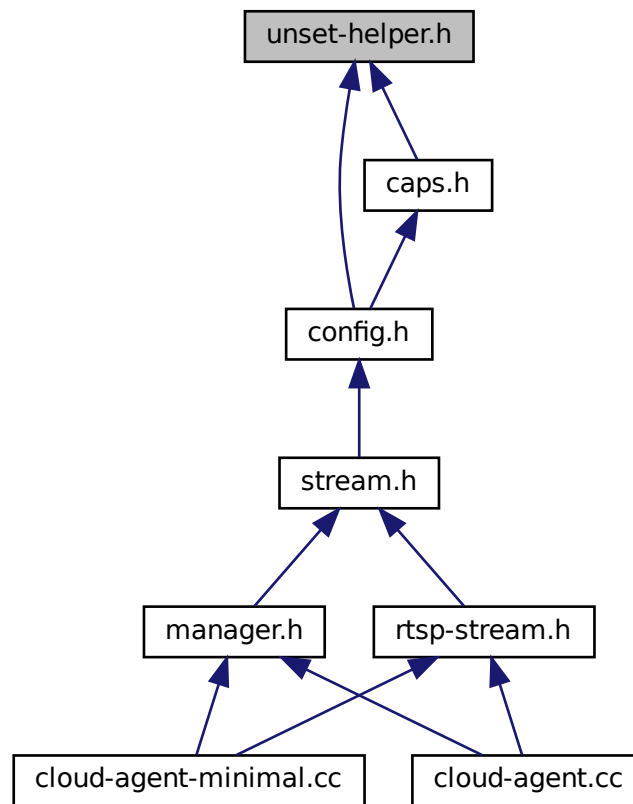
## 11.25 unset-helper.h File Reference

```
#include <chrono>
#include <limits>
#include <nlohmann/json.hpp>
#include <string>
#include <vector>
```

Include dependency graph for unset-helper.h:



This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [alter\\_bool](#)

*alternative bool class Standard bool type has two states, this class adds 3rd state - undefined.*

## Namespaces

- [vsg](#)
- [vsg::cloud](#)
- [vsg::cloud::time\\_spec](#)

*time point*

## Functions

- **std::string** [unset\\_value\\_for\\_impl](#) ( **std::string** \*)
- int [unset\\_value\\_for\\_impl](#) (int \*)  
*Returns value of int type that can be treated as unset.*
- double [unset\\_value\\_for\\_impl](#) (double \*)

- `uint64_t unset_value_for_impl (uint64_t *)`
- `int64_t unset_value_for_impl (int64_t *)`
- `vxg::cloud::time unset_value_for_impl (vxg::cloud::time *)`
- `vxg::cloud::duration unset_value_for_impl (vxg::cloud::duration *)`
- `nlohmann::json unset_value_for_impl (nlohmann::json *)`
- `template<typename T >`  
`T unset_value_for ()`  
*Template function which returns object value treated as 'unset' or uninitialized.*
- `template<typename T >`  
`std::vector< T > unset_value_for_impl ( std::vector< T > *)`
- `template<typename T >`  
`T unset_value_for_impl (T *)`
- `template<typename T >`  
`bool __is_unset (T)`  
*Used for objects constructed from json, helps to check if original json object has specific field.*
- `template<> bool __is_unset< int > (int t)`  
*Predicate function checks if int value was not initialized.*
- `template<> bool __is_unset< std::string > ( std::string t)`
- `template<> bool __is_unset< double > (double t)`
- `template<> bool __is_unset< vxg::cloud::time > (vxg::cloud::time t)`
- `template<> bool __is_unset< vxg::cloud::duration > (vxg::cloud::duration t)`
- `template<> bool __is_unset< nlohmann::json > (nlohmann::json t)`
- `template<> bool __is_unset< std::nullptr_t > ( std::nullptr_t t)`
- `template<typename T >`  
`bool __is_unset (nlohmann::json t)`
- `template<> bool __is_unset< alter_bool > (alter_bool t)`

## Variables

- `const std::string UnsetString`
- `const vxg::cloud::time UnsetTime`
- `const vxg::cloud::duration UnsetDuration`
- `const int UnsetInt`
- `const double UnsetFloat`
- `const double UnsetDouble`
- `const uint64_t UnsetUInt64`
- `const int64_t UnsetInt64`

## 11.25.1 Function Documentation

### 11.25.1.1 \_\_is\_unset() [1/2]

```
template<typename T >
bool __is_unset (
    nlohmann::json t ) [inline]
```

Definition at line 155 of file unset-helper.h.



**11.25.1.2 \_\_is\_unset()** [2/2]

```
template<typename T >
bool __is_unset (
    T ) [inline]
```

Used for objects constructed from json, helps to check if original json object has specific field.

You need to declare template specification for new types.

See also

[\\_\\_is\\_unset<int>\(int t\)](#)

**Template Parameters**

<i>T</i>	object of type
----------	----------------

**Returns**

true If object's field was actually set during construction, i.e. original json has such field in it's body.

false If object's field wasn't set, original json has no such field. It's also possible that json has such field but its value is set to value treated as unset value.

See also

[\\_\\_is\\_unset<>\(\)](#)

Definition at line 104 of file unset-helper.h.

**11.25.1.3 \_\_is\_unset< alter\_bool >()**

```
template<>
bool __is_unset< alter_bool > (
    alter_bool t ) [inline]
```

Definition at line 219 of file unset-helper.h.

**11.25.1.4 \_\_is\_unset< double >()**

```
template<>
bool __is_unset< double > (
    double t ) [inline]
```

Definition at line 126 of file unset-helper.h.

**11.25.1.5 \_\_is\_unset< int >()**

```
template<>
bool __is_unset< int > (
    int t ) [inline]
```

Predicate function checks if int value was not initialized.

### Template Parameters

<i>int</i>	
------------	--

### Parameters

<i>t</i>	
----------	--

### Returns

true value is uninitialized.

false value is initialized.

### See also

[unset\\_value\\_for<int>\(\)](#)

Definition at line 116 of file unset-helper.h.

#### 11.25.1.6 `__is_unset< nlohmann::json >()`

```
template<>
bool __is_unset< nlohmann::json > (
    nlohmann::json t ) [inline]
```

Definition at line 141 of file unset-helper.h.

#### 11.25.1.7 `__is_unset< std::nullptr_t >()`

```
template<>
bool __is_unset< std::nullptr_t > (
    std::nullptr_t t ) [inline]
```

Definition at line 150 of file unset-helper.h.

#### 11.25.1.8 `__is_unset< std::string >()`

```
template<>
bool __is_unset< std::string > (
    std::string t ) [inline]
```

Definition at line 121 of file unset-helper.h.

**11.25.1.9 `__is_unset< vxg::cloud::duration >()`**

```
template<>
bool __is_unset< vxg::cloud::duration > (
    vxg::cloud::duration t ) [inline]
```

Definition at line 136 of file unset-helper.h.

**11.25.1.10 `__is_unset< vxg::cloud::time >()`**

```
template<>
bool __is_unset< vxg::cloud::time > (
    vxg::cloud::time t ) [inline]
```

Definition at line 131 of file unset-helper.h.

**11.25.1.11 `unset_value_for()`**

```
template<typename T >
T unset_value_for ( )
```

Template function which returns object value treated as 'unset' or uninitialized.

**Template Parameters**

<i>T</i>	
----------	--

**Returns**

T Value equals to conditionally 'unset'.

Definition at line 73 of file unset-helper.h.

**11.25.1.12 `unset_value_for_impl()` [1/10]**

```
double unset_value_for_impl (
    double * ) [inline]
```

Definition at line 39 of file unset-helper.h.

**11.25.1.13 unset\_value\_for\_impl()** [2/10]

```
int unset_value_for_impl (  
    int * ) [inline]
```

Returns value of int type that can be treated as unset.

**Returns**

int

Definition at line 35 of file unset-helper.h.

**11.25.1.14 unset\_value\_for\_impl()** [3/10]

```
int64_t unset_value_for_impl (  
    int64_t * ) [inline]
```

Definition at line 47 of file unset-helper.h.

**11.25.1.15 unset\_value\_for\_impl()** [4/10]

```
nlohmann::json unset_value_for_impl (  
    nlohmann::json * ) [inline]
```

Definition at line 62 of file unset-helper.h.

**11.25.1.16 unset\_value\_for\_impl()** [5/10]

```
std::string unset_value_for_impl (  
    std::string * ) [inline]
```

Definition at line 27 of file unset-helper.h.

**11.25.1.17 unset\_value\_for\_impl()** [6/10]

```
template<typename T >  
std::vector<T> unset_value_for_impl (  
    std::vector< T > * ) [inline]
```

Definition at line 78 of file unset-helper.h.

**11.25.1.18 unset\_value\_for\_impl() [7/10]**

```
template<typename T >
T unset_value_for_impl (
    T * )
```

Definition at line 85 of file unset-helper.h.

**11.25.1.19 unset\_value\_for\_impl() [8/10]**

```
uint64_t unset_value_for_impl (
    uint64_t * ) [inline]
```

Definition at line 43 of file unset-helper.h.

**11.25.1.20 unset\_value\_for\_impl() [9/10]**

```
vsg::cloud::duration unset_value_for_impl (
    vsg::cloud::duration * ) [inline]
```

Definition at line 57 of file unset-helper.h.

**11.25.1.21 unset\_value\_for\_impl() [10/10]**

```
vsg::cloud::time unset_value_for_impl (
    vsg::cloud::time * ) [inline]
```

Definition at line 51 of file unset-helper.h.

**11.25.2 Variable Documentation****11.25.2.1 UnsetDouble**

```
const double UnsetDouble
```

Definition at line 229 of file unset-helper.h.

### 11.25.2.2 UnsetDuration

```
const vxg::cloud::duration UnsetDuration
```

Definition at line 225 of file unset-helper.h.

### 11.25.2.3 UnsetFloat

```
const double UnsetFloat
```

Definition at line 228 of file unset-helper.h.

### 11.25.2.4 UnsetInt

```
const int UnsetInt
```

Definition at line 227 of file unset-helper.h.

### 11.25.2.5 UnsetInt64

```
const int64_t UnsetInt64
```

Definition at line 231 of file unset-helper.h.

### 11.25.2.6 UnsetString

```
const std::string UnsetString
```

Definition at line 223 of file unset-helper.h.

### 11.25.2.7 UnsetTime

```
const vxg::cloud::time UnsetTime
```

Definition at line 224 of file unset-helper.h.

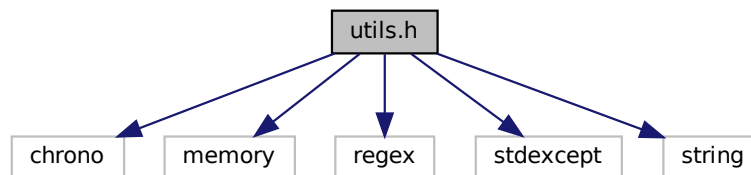
### 11.25.2.8 UnsetUInt64

```
const uint64_t UnsetUInt64
```

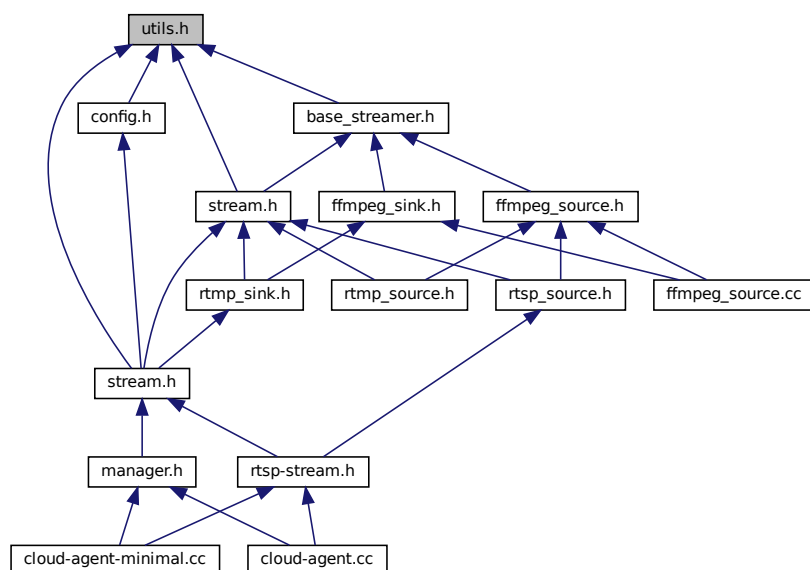
Definition at line 230 of file unset-helper.h.

## 11.26 utils.h File Reference

```
#include <chrono>
#include <memory>
#include <regex>
#include <stdexcept>
#include <string>
Include dependency graph for utils.h:
```



This graph shows which files directly or indirectly include this file:



## Data Structures

- struct [vsg::cloud::utils::uri](#)
- struct [vsg::cloud::utils::motion::map](#)

## Namespaces

- [vsg](#)
- [vsg::cloud](#)
- [vsg::cloud::time\\_spec](#)
- *time point*
- [vsg::cloud::utils](#)
- [vsg::cloud::utils::time](#)
- [vsg::cloud::utils::motion](#)
- [vsg::cloud::utils::gcc\\_abi](#)
- [std](#)

## Functions

- void [vsg::cloud::utils::set\\_thread\\_name](#) ( **std::string** name)
- cloud::time [vsg::cloud::utils::time::now](#) ()
- **std::string** [vsg::cloud::utils::time::time\\_to\\_ISO8601](#) ( **std::time\_t**)
- **std::string** [vsg::cloud::utils::time::time\\_to\\_ISO8601\\_packed](#) ( **std::time\_t**)
- **std::string** [vsg::cloud::utils::time::now\\_ISO8601.UTC](#) ()
- **std::string** [vsg::cloud::utils::time::now\\_ISO8601.UTC\\_packed](#) ()
- **std::time\_t** [vsg::cloud::utils::time::now\\_time.UTC](#) ()
- **std::time\_t** [vsg::cloud::utils::time::ISO8601\\_to\\_time](#) (const **std::string** &input)
- **std::string** [vsg::cloud::utils::time::to\\_iso\\_8601](#) (cloud::time t)
- **std::string** [vsg::cloud::utils::time::to\\_iso](#) (cloud::time t)
- **std::string** [vsg::cloud::utils::time::to\\_iso2](#) (cloud::time t)
- **std::string** [vsg::cloud::utils::time::to\\_iso\\_packed](#) (cloud::time t)
- **std::string** [vsg::cloud::utils::time::to\\_iso\\_local](#) (cloud::time t)
- cloud::time [vsg::cloud::utils::time::from\\_double](#) (double t)
- double [vsg::cloud::utils::time::to\\_double](#) (cloud::time t)
- cloud::time [vsg::cloud::utils::time::from\\_iso](#) ( **std::string** st)
- cloud::time [vsg::cloud::utils::time::from\\_iso2](#) ( **std::string** st)
- cloud::time [vsg::cloud::utils::time::from\\_iso\\_packed](#) ( **std::string** st)
- bool [vsg::cloud::utils::time::iso\\_time\\_valid](#) (const **std::string** &s)
- cloud::time [vsg::cloud::utils::time::null](#) ()
- cloud::time [vsg::cloud::utils::time::max](#) ()
- bool [vsg::cloud::utils::time::is\\_iso\\_packed](#) (const **std::string** &s)
- bool [vsg::cloud::utils::time::is\\_iso](#) (const **std::string** &s)
- template<typename... Args>  
**std::string** [vsg::cloud::utils::string\\_format](#) (const **std::string** &format, Args... args)
- **std::string** [vsg::cloud::utils::string\\_trim](#) (const **std::string** &name, **std::regex** regx)
- **std::string** [vsg::cloud::utils::string\\_trim](#) (const **std::string** &name)
- **std::vector**< **std::string** > [vsg::cloud::utils::string\\_split](#) (const **std::string** &s, char delimiter)
- bool [vsg::cloud::utils::string\\_startswith](#) ( **std::string** const &fullString, **std::string** const &start)
- bool [vsg::cloud::utils::string\\_endswith](#) ( **std::string** const &fullString, **std::string** const &ending)
- bool [vsg::cloud::utils::string\\_replace](#) ( **std::string** &str, const **std::string** &from, const **std::string** &to)
- **std::string** [vsg::cloud::utils::string\\_urlencode](#) (const **std::string** &value)
- **std::string** [vsg::cloud::utils::string\\_urldecode](#) (const **std::string** &text)
- **std::string** [vsg::cloud::utils::string\\_tolower](#) (const **std::string** &s)



- **std::string** [vsg::cloud::utils::string\\_toupper](#) (const **std::string** &s)
- bool [vsg::cloud::utils::string\\_contains](#) ( **std::string** s, char c)
- **std::string** [vsg::cloud::utils::dirname](#) (const **std::string** &filepath)
- **std::string** [vsg::cloud::utils::gcc\\_abi::demangle](#) ( **std::string** name)
- template<typename T , typename... CONSTRUCTOR\_ARGS>  
    **std::unique\_ptr**< T > [std::make\\_unique](#) (CONSTRUCTOR\_ARGS &&... constructor\_args)



# Index

- `__BASE_STREAMER_H`
  - `base_streamer.h`, 239
- `__CONFIG_H`
  - `config.h`, 250
- `__is_unset`
  - `unset-helper.h`, 264
- `__is_unset< alter_bool >`
  - `unset-helper.h`, 265
- `__is_unset< double >`
  - `unset-helper.h`, 265
- `__is_unset< int >`
  - `unset-helper.h`, 265
- `__is_unset< nlohmann::json >`
  - `unset-helper.h`, 266
- `__is_unset< std::nullptr_t >`
  - `unset-helper.h`, 266
- `__is_unset< std::string >`
  - `unset-helper.h`, 266
- `__is_unset< vxg::cloud::duration >`
  - `unset-helper.h`, 266
- `__is_unset< vxg::cloud::time >`
  - `unset-helper.h`, 267
- `__notify_record_event`
  - `vxg::cloud::agent::manager`, 129
- `__trigger_periodic_event`
  - `vxg::cloud::agent::manager`, 129
- `_append_internal_custom_events`
  - `vxg::cloud::agent::manager`, 129
- `_cancel_direct_uploads_by_ticket`
  - `vxg::cloud::agent::manager`, 129
- `_cancel_periodic_event`
  - `vxg::cloud::agent::manager`, 129
- `_cancel_periodic_events`
  - `vxg::cloud::agent::manager`, 129
- `_current_delivery_mode`
  - `vxg::cloud::agent::manager`, 130
- `_handle_stream_stateful_event`
  - `vxg::cloud::agent::manager`, 130
- `_handle_stream_stateless_event`
  - `vxg::cloud::agent::manager`, 130
- `_init_events_states`
  - `vxg::cloud::agent::manager`, 130
- `_load_events_configs`
  - `vxg::cloud::agent::manager`, 130
- `_lookup_event_stream`
  - `vxg::cloud::agent::manager`, 130
- `_lookup_event_stream_by_event`
  - `vxg::cloud::agent::manager`, 130
- `_request_direct_upload_snapshot`
  - `vxg::cloud::agent::manager`, 131
- `_request_direct_upload_video`
  - `vxg::cloud::agent::manager`, 131
- `_schedule_direct_upload`
  - `vxg::cloud::agent::manager`, 131
- `_schedule_periodic_event`
  - `vxg::cloud::agent::manager`, 131
- `_schedule_periodic_events`
  - `vxg::cloud::agent::manager`, 131
- `_stop_all_event_streams`
  - `vxg::cloud::agent::manager`, 131
- `_stop_all_streams`
  - `vxg::cloud::agent::manager`, 131
- `_stop_stream`
  - `vxg::cloud::agent::manager`, 132
- `_update_direct_upload_queue_latency`
  - `vxg::cloud::agent::manager`, 132
- `_update_event_stream_configs`
  - `vxg::cloud::agent::manager`, 132
- `_update_events_configs`
  - `vxg::cloud::agent::manager`, 132
- `_update_storage_status`
  - `vxg::cloud::agent::manager`, 132
- `~ISink`
  - `vxg::media::Streamer::ISink`, 112
- `~Sink`
  - `vxg::media::ffmpeg::Sink`, 186
- `~Source`
  - `vxg::media::ffmpeg::Source`, 192
- `~event_stream`
  - `vxg::cloud::agent::event_stream`, 104
- `~rtsp_stream`
  - `vxg::cloud::agent::media::rtsp_stream`, 180
- `~stream`
  - `vxg::cloud::agent::media::stream`, 197
  - `vxg::media::stream`, 204
- `A_BOTTOM`
  - `vxg::cloud::agent::proto`, 51
- `A_INVALID`
  - `vxg::cloud::agent::proto`, 51
- `A_LEFT`
  - `vxg::cloud::agent::proto`, 51
- `A_RIGHT`
  - `vxg::cloud::agent::proto`, 51
- `A_STOP`
  - `vxg::cloud::agent::proto`, 51
- `A_TOP`
  - `vxg::cloud::agent::proto`, 51
- `A_ZOOM_IN`

- vxg::cloud::agent::proto, 51
- A\_ZOOM\_OUT
  - vxg::cloud::agent::proto, 51
- AC\_AAC
  - vxg::media::Streamer::StreamInfo, 210
- AC\_G711\_A
  - vxg::media::Streamer::StreamInfo, 210
- AC\_G711\_U
  - vxg::media::Streamer::StreamInfo, 210
- AC\_G726
  - vxg::media::Streamer::StreamInfo, 210
- AC\_LPCM
  - vxg::media::Streamer::StreamInfo, 210
- AC\_OPUS
  - vxg::media::Streamer::StreamInfo, 210
- AC\_UNKNOWN
  - vxg::media::Streamer::StreamInfo, 210
- action
  - vxg::cloud::agent::ptz\_command, 164
  - vxg::cloud::agent::ptz\_preset, 167
- actions
  - vxg::cloud::agent::ptz\_config, 165
- active
  - vxg::cloud::agent::event\_config, 100
- AF\_AAC
  - vxg::cloud::agent::proto, 49
- AF\_ADPCM
  - vxg::cloud::agent::proto, 49
- AF\_G711A
  - vxg::cloud::agent::proto, 49
- AF\_G711U
  - vxg::cloud::agent::proto, 49
- AF\_INVALID
  - vxg::cloud::agent::proto, 49
- AF\_MP3
  - vxg::cloud::agent::proto, 49
- AF\_NELLY
  - vxg::cloud::agent::proto, 49
- AF\_NELLY16
  - vxg::cloud::agent::proto, 49
- AF\_NELLY8
  - vxg::cloud::agent::proto, 49
- AF\_OPUS
  - vxg::cloud::agent::proto, 49
- AF\_RAW
  - vxg::cloud::agent::proto, 49
- AF\_SPEEX
  - vxg::cloud::agent::proto, 49
- AFF\_AU\_G711U
  - vxg::cloud::agent::proto, 48
- AFF\_INVALID
  - vxg::cloud::agent::proto, 48
- AFF\_MP3
  - vxg::cloud::agent::proto, 48
- AFF\_WAV\_PCM
  - vxg::cloud::agent::proto, 48
- alignment
  - vxg::cloud::agent::osd\_config, 159
- vxg::cloud::agent::proto::osd\_caps, 156
- alter\_bool, 66
  - alter\_bool, 68
  - B\_FALSE, 67
  - B\_INVALID, 67
  - B\_TRUE, 67
  - from\_json, 68
  - n\_alter\_bool, 67
  - operator bool, 68
  - operator=, 68
  - to\_json, 69
  - val, 69
- api\_uri
  - vxg::cloud::agent::access\_token, 66
- app-dev.md, 237
- arm-example.txt, 237
- AUDIO
  - vxg::media::Streamer, 63
- audio
  - vxg::cloud::agent::proto::stream\_config, 208
  - vxg::cloud::agent::supported\_stream\_config, 213
  - vxg::media::Streamer::StreamInfo, 211
- audio\_es
  - vxg::cloud::agent::supported\_streams\_config, 214
- audio\_file\_format
  - vxg::cloud::agent::proto, 48
- audio\_file\_formats
  - vxg::cloud::agent::proto::audio\_caps, 70
- audio\_format
  - vxg::cloud::agent::proto, 48
- AUDIO\_SEQ\_HDR
  - vxg::media::Streamer, 63
- AudioCodec
  - vxg::media::Streamer::StreamInfo, 210
- B\_FALSE
  - alter\_bool, 67
- B\_INVALID
  - alter\_bool, 67
- B\_TRUE
  - alter\_bool, 67
- backward
  - vxg::cloud::agent::proto::audio\_caps, 70
- backward\_formats
  - vxg::cloud::agent::proto::audio\_caps, 70
- base\_streamer.h, 237
  - \_\_BASE\_STREAMER\_H, 239
- bitrate
  - vxg::media::Streamer::StreamInfo::AudioInfo, 76
  - vxg::media::Streamer::StreamInfo::VideoInfo, 232
- bkg\_color
  - vxg::cloud::agent::osd\_config, 159
  - vxg::cloud::agent::proto::osd\_caps, 156
- bkg\_transp
  - vxg::cloud::agent::osd\_config, 160
  - vxg::cloud::agent::proto::osd\_caps, 156
- brightness
  - vxg::cloud::agent::proto::video\_caps, 218
  - vxg::cloud::agent::proto::video\_config, 225

- brt
  - vxg::cloud::agent::proto::audio\_stream\_config, 75
  - vxg::cloud::agent::proto::stream\_caps::caps\_audio\_object, 91
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 94
  - vxg::cloud::agent::proto::video\_stream\_config, 229
- build-system.md, 239
- callback.h, 239
- caps
  - vxg::cloud::agent::audio\_config, 72
  - vxg::cloud::agent::event\_config, 100
  - vxg::cloud::agent::osd\_config, 160
  - vxg::cloud::agent::proto::motion\_detection\_config, 149
  - vxg::cloud::agent::proto::video\_config, 225
- caps.h, 240
  - ignore\_exception, 242
  - json, 243
- caps\_audio
  - vxg::cloud::agent::proto::stream\_caps, 207
- caps\_eq
  - vxg::cloud::agent::event\_config, 99
- caps\_video
  - vxg::cloud::agent::proto::stream\_caps, 207
- channels
  - vxg::media::Streamer::StreamInfo::AudioInfo, 77
- cloud-agent-minimal.cc, 243
  - main, 244
  - parse\_args, 244
  - props, 244
  - quit, 244
  - rtsp\_url, 245
  - signal\_handler, 244
  - vxg\_cloud\_token, 245
- cloud-agent.cc, 245
  - main, 246
  - parse\_args, 246
  - quit, 246
  - rtsp\_url, 247
  - signal\_handler, 246
  - vxg\_cloud\_token, 247
- codec
  - vxg::media::Streamer::StreamInfo::AudioInfo, 77
  - vxg::media::Streamer::StreamInfo::VideoInfo, 232
- columns
  - vxg::cloud::agent::proto::motion\_detection\_config, 149
- compile.md, 247
- config.h, 247
  - \_\_CONFIG\_H\_, 250
- contrast
  - vxg::cloud::agent::proto::video\_caps, 218
  - vxg::cloud::agent::proto::video\_config, 225
- crash\_logfile\_path
  - vxg::logger::options, 153
- create
  - vxg::cloud::agent::manager, 132
- custom\_event\_name
  - vxg::cloud::agent::event\_config, 100
- DATA
  - vxg::media::Streamer, 63
- data
  - vxg::cloud::agent::proto::video\_clip\_info, 222
  - vxg::media::Streamer::MediaFrame, 145
- DataCodec
  - vxg::media::Streamer::StreamInfo, 210
- date
  - vxg::cloud::agent::osd\_config, 160
  - vxg::cloud::agent::proto::osd\_caps, 156
- date\_format
  - vxg::cloud::agent::osd\_config, 160
  - vxg::cloud::agent::proto::osd\_caps, 156
- DC\_ONVIF
  - vxg::media::Streamer::StreamInfo, 210
- DC\_UNKNOWN
  - vxg::media::Streamer::StreamInfo, 210
- debug
  - vxg::logger, 122
- default\_loglevel
  - vxg::logger::options, 153
- demangle
  - vxg::cloud::utils::gcc\_abi, 57
- direct\_upload\_sync\_cb
  - vxg::cloud::agent::manager, 133
- dirname
  - vxg::cloud::utils, 55
- DROP\_BACK
  - vxg::media::Streamer, 63
- DROP\_FRONT
  - vxg::media::Streamer, 63
- DropDirection
  - vxg::media::Streamer, 63
- droppable
  - vxg::media::ffmpeg::Sink, 186
  - vxg::media::rtmp\_sink, 169
  - vxg::media::Streamer::ISink, 112
- dts
  - vxg::media::Streamer::MediaFrame, 145
- duration
  - vxg::cloud, 44
  - vxg::cloud::time\_spec, 54
  - vxg::media::ffmpeg::Sink, 187
  - vxg::media::Streamer::ISink, 112
  - vxg::media::Streamer::MediaFrame, 145
- E\_EOS
  - vxg::media::Streamer, 64
- E\_FATAL
  - vxg::media::Streamer, 64
- E\_NONE
  - vxg::media::Streamer, 64
- echo\_cancel
  - vxg::cloud::agent::audio\_config, 73
  - vxg::cloud::agent::proto::audio\_caps, 71
- enabled

- vxg::cloud::agent::events\_config, 109
  - vxg::cloud::agent::proto::motion\_region, 151
- encryption
  - vxg::cloud::agent::proto::wifi\_network, 235
- encryption\_caps
  - vxg::cloud::agent::proto::wifi\_network, 236
- error
  - vxg::logger, 122
  - vxg::media::ffmpeg::Sink, 187
  - vxg::media::rtmp\_sink, 170
  - vxg::media::Streamer::ISink, 112
  - vxg::media::Streamer::ISource, 117
- ES\_ERROR
  - vxg::cloud::agent::proto, 49
- ES\_INVALID
  - vxg::cloud::agent::proto, 49
- ES\_OK
  - vxg::cloud::agent::proto, 49
- ET\_CUSTOM
  - vxg::cloud::agent::proto, 49
- ET\_INVALID
  - vxg::cloud::agent::proto, 49
- ET\_MEMORYCARD
  - vxg::cloud::agent::proto, 49
- ET\_MOTION
  - vxg::cloud::agent::proto, 49
- ET\_NET
  - vxg::cloud::agent::proto, 49
- ET\_RECORD
  - vxg::cloud::agent::proto, 49
- ET\_SOUND
  - vxg::cloud::agent::proto, 49
- ET\_WIFI
  - vxg::cloud::agent::proto, 49
- event
  - vxg::cloud::agent::event\_config, 101
- event-stream.h, 251
- event\_status
  - vxg::cloud::agent::proto, 49
- event\_stream
  - vxg::cloud::agent::event\_stream, 104
- event\_type
  - vxg::cloud::agent::proto, 49
- events
  - vxg::cloud::agent::events\_config, 109
- extradata
  - vxg::media::Streamer::StreamInfo::AudioInfo, 77
  - vxg::media::Streamer::StreamInfo::VideoInfo, 232
- ffmpeg\_opts\_
  - vxg::media::rtsp\_source, 178
- ffmpeg\_sink.h, 252
- ffmpeg\_source.cc, 253
- ffmpeg\_source.h, 253
- finit
  - vxg::cloud::agent::event\_stream, 104
  - vxg::media::ffmpeg::Sink, 188
  - vxg::media::ffmpeg::Source, 192
  - vxg::media::Streamer::ISink, 113
  - vxg::media::Streamer::ISource, 118
- finit\_sink
  - vxg::media::stream, 204
- finit\_source
  - vxg::media::stream, 204
- FLV
  - vxg::media::Streamer, 63
- font\_color
  - vxg::cloud::agent::osd\_config, 160
  - vxg::cloud::agent::proto::osd\_caps, 157
- font\_size
  - vxg::cloud::agent::osd\_config, 161
  - vxg::cloud::agent::proto::osd\_caps, 157
- format
  - vxg::cloud::agent::proto::audio\_stream\_config, 75
  - vxg::cloud::agent::proto::video\_stream\_config, 229
- formats
  - vxg::cloud::agent::proto::stream\_caps::caps\_audio\_object, 92
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 94
- fps
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 94
  - vxg::cloud::agent::proto::video\_stream\_config, 229
- fragment
  - vxg::cloud::utils::uri, 216
- framerate
  - vxg::media::Streamer::StreamInfo::VideoInfo, 233
- from\_double
  - vxg::cloud::utils::time, 58
- from\_iso
  - vxg::cloud::utils::time, 58
- from\_iso2
  - vxg::cloud::utils::time, 59
- from\_iso\_packed
  - vxg::cloud::utils::time, 59
- from\_json
  - alter\_bool, 68
- get\_event\_config
  - vxg::cloud::agent::events\_config, 108
- get\_events
  - vxg::cloud::agent::event\_stream, 104
- get\_snapshot
  - vxg::cloud::agent::media::rtsp\_stream, 181
  - vxg::cloud::agent::media::stream, 198
- get\_stream\_caps
  - vxg::cloud::agent::media::rtsp\_stream, 181
  - vxg::cloud::agent::media::stream, 198
- get\_stream\_config
  - vxg::cloud::agent::media::rtsp\_stream, 181
  - vxg::cloud::agent::media::stream, 199
- get\_supported\_stream
  - vxg::cloud::agent::media::rtsp\_stream, 182
  - vxg::cloud::agent::media::stream, 199
- gop
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 94

- vxg::cloud::agent::proto::video\_stream\_config, 229
- handle\_event\_meta\_file
  - vxg::cloud::agent::manager, 133
- handle\_event\_snapshot
  - vxg::cloud::agent::manager, 133
- handle\_stream\_event
  - vxg::cloud::agent::manager, 133
- height
  - vxg::media::Streamer::StreamInfo::VideoInfo, 233
- horz
  - vxg::cloud::agent::proto::video\_stream\_config, 229
- horz\_flip
  - vxg::cloud::agent::proto::video\_caps, 219
  - vxg::cloud::agent::proto::video\_config, 225
- host
  - vxg::cloud::utils::uri, 216
- id
  - vxg::cloud::agent::supported\_stream\_config, 213
- ignore\_exception
  - caps.h, 242
- info
  - vxg::logger, 122, 123
- init
  - vxg::cloud::agent::event\_stream, 105
  - vxg::media::ffmpeg::Sink, 188
  - vxg::media::ffmpeg::Source, 192, 193
  - vxg::media::rtmp\_sink, 170
  - vxg::media::rtmp\_source, 173
  - vxg::media::rtsp\_source, 177
  - vxg::media::Streamer::ISink, 113
  - vxg::media::Streamer::ISource, 118
- init\_sink
  - vxg::media::stream, 204
- init\_source
  - vxg::media::stream, 205
- instance
  - vxg::logger, 123
- ir\_light
  - vxg::cloud::agent::proto::video\_caps, 219
  - vxg::cloud::agent::proto::video\_config, 226
- is\_iso
  - vxg::cloud::utils::time, 59
- is\_iso\_packed
  - vxg::cloud::utils::time, 59
- is\_key
  - vxg::media::Streamer::MediaFrame, 145
- ISink
  - vxg::media::Streamer::ISink, 111
- ISO8601\_to\_time
  - vxg::cloud::utils::time, 59
- iso\_time\_valid
  - vxg::cloud::utils::time, 59
- ISource
  - vxg::media::Streamer::ISource, 117
- json
  - caps.h, 243
- len
  - vxg::media::Streamer::MediaFrame, 146
- local\_start
  - vxg::cloud::agent::proto::video\_clip\_info, 222
- local\_stop
  - vxg::cloud::agent::proto::video\_clip\_info, 222
- log\_pattern
  - vxg::logger::options, 153
- logfile\_max\_files
  - vxg::logger::options, 153
- logfile\_max\_size
  - vxg::logger::options, 153
- logfile\_path
  - vxg::logger::options, 153
- logger\_ptr
  - vxg::logger, 121
- logging.h, 254
- loglevel
  - vxg::logger, 121
- lookup\_stream
  - vxg::cloud::agent::manager, 133
- lvl\_crit
  - vxg::logger, 121
- lvl\_debug
  - vxg::logger, 121
- lvl\_error
  - vxg::logger, 121
- lvl\_info
  - vxg::logger, 121
- lvl\_off
  - vxg::logger, 121
- lvl\_trace
  - vxg::logger, 121
- lvl\_warn
  - vxg::logger, 121
- M\_AUTO
  - vxg::cloud::agent::proto, 50
- M\_INVALID
  - vxg::cloud::agent::proto, 50
- M\_OFF
  - vxg::cloud::agent::proto, 50
- M\_ON
  - vxg::cloud::agent::proto, 50
- mac
  - vxg::cloud::agent::proto::wifi\_network, 236
- main
  - cloud-agent-minimal.cc, 244
  - cloud-agent.cc, 246
- mainpage.md, 255
- make\_unique
  - std, 43
- manager.h, 255
- map
  - vxg::cloud::agent::proto::motion\_region, 151
  - vxg::cloud::utils::motion::map, 142
- MAX
  - vxg::media::Streamer, 63
- max

- vxg::cloud::utils::time, 59
- max\_regions
  - vxg::cloud::agent::proto::motion\_detection\_caps, 147
- maximum\_number\_of\_presets
  - vxg::cloud::agent::ptz\_config, 165
- MCS\_FORMATTING
  - vxg::cloud::agent::proto, 50
- MCS\_INITIALIZATION
  - vxg::cloud::agent::proto, 50
- MCS\_INVALID
  - vxg::cloud::agent::proto, 50
- MCS\_NEED\_FORMAT
  - vxg::cloud::agent::proto, 50
- MCS\_NONE
  - vxg::cloud::agent::proto, 50
- MCS\_NORMAL
  - vxg::cloud::agent::proto, 50
- MediaType
  - vxg::media::Streamer, 63
- memorycard\_status
  - vxg::cloud::agent::proto, 50
- meson.build, 256
- mic
  - vxg::cloud::agent::proto::audio\_caps, 71
- mic\_gain
  - vxg::cloud::agent::audio\_config, 73
- mic\_mute
  - vxg::cloud::agent::audio\_config, 73
- Mode
  - vxg::media::Streamer::ISource, 116
- mode
  - vxg::cloud::agent::proto, 50
- mode\_
  - vxg::media::Streamer::ISource, 120
- motion\_region\_shape
  - vxg::cloud::agent::proto, 50
- motion\_sensitivity
  - vxg::cloud::agent::proto, 51
- MR\_ANY
  - vxg::cloud::agent::proto, 50
- MR\_INVALID
  - vxg::cloud::agent::proto, 50
- MR\_RECTANGLE
  - vxg::cloud::agent::proto, 50
- MS\_FRAME
  - vxg::cloud::agent::proto, 51
- MS\_INVALID
  - vxg::cloud::agent::proto, 51
- MS\_REGION
  - vxg::cloud::agent::proto, 51
- n\_alter\_bool
  - alter\_bool, 67
- name
  - vxg::cloud::agent::event\_config, 99
  - vxg::cloud::agent::proto, 53
  - vxg::cloud::agent::ptz\_preset, 167
  - vxg::media::ffmpeg::Sink, 189
  - vxg::media::ffmpeg::Source, 194
  - vxg::media::rtmp\_sink, 171
  - vxg::media::rtsp\_source, 177
  - vxg::media::Streamer::ISink, 113
  - vxg::media::Streamer::ISource, 118
- name\_eq
  - vxg::cloud::agent::event\_config, 100
- need\_clip
  - vxg::cloud::agent::manager::event\_state::event\_state\_caps, 102
- need\_snapshot
  - vxg::cloud::agent::manager::event\_state::event\_state\_caps, 102
- negotiate
  - vxg::media::ffmpeg::Sink, 189
  - vxg::media::ffmpeg::Source, 194
  - vxg::media::rtmp\_sink, 171
  - vxg::media::Streamer::ISink, 114
  - vxg::media::Streamer::ISource, 118
- networks
  - vxg::cloud::agent::proto::wifi\_config, 234
- nlohmann, 25
- NO\_PTS
  - vxg::media::Streamer::MediaFrame, 146
- notify
  - vxg::cloud::agent::event\_stream, 105
- notify\_event
  - vxg::cloud::agent::manager, 134
- now
  - vxg::cloud::utils::time, 60
- now\_ISO8601\_UTC
  - vxg::cloud::utils::time, 60
- now\_ISO8601\_UTC\_packed
  - vxg::cloud::utils::time, 60
- now\_time\_UTC
  - vxg::cloud::utils::time, 60
- nr\_level
  - vxg::cloud::agent::proto::video\_caps, 219
  - vxg::cloud::agent::proto::video\_config, 226
- nr\_type
  - vxg::cloud::agent::proto::video\_caps, 219
  - vxg::cloud::agent::proto::video\_config, 226
- null
  - vxg::cloud::utils::time, 60
- on\_audio\_file\_play
  - vxg::cloud::agent::callback, 80
  - vxg::cloud::agent::manager, 134
- on\_bye
  - vxg::cloud::agent::callback, 80
- on\_cam\_memorycard\_recording
  - vxg::cloud::agent::manager, 134
- on\_cam\_memorycard\_synchronize
  - vxg::cloud::agent::manager, 134
- on\_cam\_memorycard\_synchronize\_cancel
  - vxg::cloud::agent::manager, 134
- on\_cam\_ptz
  - vxg::cloud::agent::callback, 80
  - vxg::cloud::agent::manager, 134



- on\_cam\_ptz\_preset
  - vxg::cloud::agent::callback, [81](#)
  - vxg::cloud::agent::manager, [135](#)
- on\_cam\_upgrade\_firmware
  - vxg::cloud::agent::callback, [81](#)
  - vxg::cloud::agent::manager, [135](#)
- on\_closed
  - vxg::cloud::agent::manager, [135](#)
- on\_direct\_upload\_url
  - vxg::cloud::agent::manager, [135](#)
- on\_get\_cam\_audio\_config
  - vxg::cloud::agent::callback, [82](#)
  - vxg::cloud::agent::manager, [135](#)
- on\_get\_cam\_events\_config
  - vxg::cloud::agent::callback, [82](#)
  - vxg::cloud::agent::manager, [135](#)
- on\_get\_cam\_memorycard\_timeline
  - vxg::cloud::agent::manager, [136](#)
- on\_get\_cam\_video\_config
  - vxg::cloud::agent::callback, [82](#)
  - vxg::cloud::agent::manager, [136](#)
- on\_get\_log
  - vxg::cloud::agent::callback, [83](#)
  - vxg::cloud::agent::manager, [136](#)
- on\_get\_memorycard\_info
  - vxg::cloud::agent::callback, [83](#)
- on\_get\_motion\_detection\_config
  - vxg::cloud::agent::callback, [84](#)
  - vxg::cloud::agent::manager, [136](#)
- on\_get\_osd\_config
  - vxg::cloud::agent::callback, [84](#)
  - vxg::cloud::agent::manager, [136](#)
- on\_get\_ptz\_config
  - vxg::cloud::agent::callback, [84](#)
  - vxg::cloud::agent::manager, [136](#)
- on\_get\_stream\_by\_event
  - vxg::cloud::agent::manager, [136](#)
- on\_get\_stream\_caps
  - vxg::cloud::agent::manager, [137](#)
- on\_get\_stream\_config
  - vxg::cloud::agent::manager, [137](#)
- on\_get\_supported\_streams
  - vxg::cloud::agent::manager, [137](#)
- on\_get\_timezone
  - vxg::cloud::agent::callback, [85](#)
  - vxg::cloud::agent::manager, [137](#)
- on\_get\_wifi\_config
  - vxg::cloud::agent::callback, [85](#)
  - vxg::cloud::agent::manager, [137](#)
- on\_prepared
  - vxg::cloud::agent::manager, [137](#)
- on\_raw\_message
  - vxg::cloud::agent::manager, [137](#)
- on\_raw\_msg
  - vxg::cloud::agent::callback, [86](#)
- on\_registered
  - vxg::cloud::agent::callback, [86](#)
  - vxg::cloud::agent::manager, [138](#)
- on\_set\_activity
  - vxg::cloud::agent::manager, [138](#)
- on\_set\_cam\_audio\_config
  - vxg::cloud::agent::callback, [87](#)
  - vxg::cloud::agent::manager, [138](#)
- on\_set\_cam\_events\_config
  - vxg::cloud::agent::callback, [87](#)
  - vxg::cloud::agent::manager, [138](#)
- on\_set\_cam\_video\_config
  - vxg::cloud::agent::callback, [87](#)
  - vxg::cloud::agent::manager, [138](#)
- on\_set\_log\_enable
  - vxg::cloud::agent::manager, [138](#)
- on\_set\_motion\_detection\_config
  - vxg::cloud::agent::callback, [88](#)
  - vxg::cloud::agent::manager, [138](#)
- on\_set\_osd\_config
  - vxg::cloud::agent::callback, [88](#)
  - vxg::cloud::agent::manager, [139](#)
- on\_set\_periodic\_events
  - vxg::cloud::agent::manager, [139](#)
- on\_set\_stream\_by\_event
  - vxg::cloud::agent::manager, [139](#)
- on\_set\_stream\_config
  - vxg::cloud::agent::manager, [139](#)
- on\_set\_timezone
  - vxg::cloud::agent::callback, [89](#)
  - vxg::cloud::agent::manager, [139](#)
- on\_set\_wifi\_config
  - vxg::cloud::agent::callback, [89](#)
  - vxg::cloud::agent::manager, [139](#)
- on\_start\_backward
  - vxg::cloud::agent::manager, [139](#)
- on\_start\_backward\_audio
  - vxg::cloud::agent::callback, [89](#)
- on\_stop\_backward
  - vxg::cloud::agent::manager, [140](#)
- on\_stop\_backward\_audio
  - vxg::cloud::agent::callback, [90](#)
- on\_stream\_start
  - vxg::cloud::agent::manager, [140](#)
- on\_stream\_stop
  - vxg::cloud::agent::manager, [140](#)
- on\_trigger\_event
  - vxg::cloud::agent::callback, [90](#)
  - vxg::cloud::agent::manager, [140](#)
- on\_update\_preview
  - vxg::cloud::agent::manager, [140](#)
- operator bool
  - alter\_bool, [68](#)
- operator<
  - vxg::media::Streamer::MediaFrame, [144](#)
- operator=
  - alter\_bool, [68](#)
  - vxg::cloud::utils::motion::map, [143](#)
- PA\_CREATE
  - vxg::cloud::agent::proto, [52](#)
- PA\_DELETE

- vxg::cloud::agent::proto, 52
- PA\_GOTO
  - vxg::cloud::agent::proto, 52
- PA\_INVALID
  - vxg::cloud::agent::proto, 52
- PA\_UPDATE
  - vxg::cloud::agent::proto, 52
- pack
  - vxg::cloud::agent::access\_token, 66
  - vxg::cloud::utils::motion::map, 143
- parse
  - vxg::cloud::agent::access\_token, 66
  - vxg::cloud::utils::uri, 215
- parse\_args
  - cloud-agent-minimal.cc, 244
  - cloud-agent.cc, 246
- password
  - vxg::cloud::agent::proto::wifi\_network, 236
  - vxg::cloud::utils::uri, 216
- path
  - vxg::cloud::utils::uri, 216
- period
  - vxg::cloud::agent::event\_config, 101
- periodic
  - vxg::cloud::agent::proto::event\_caps, 97
- port
  - vxg::cloud::utils::uri, 216
- precision
  - vxg::cloud::time\_spec, 54
- presets
  - vxg::cloud::agent::ptz\_config, 165
- process
  - vxg::media::Streamer::ISink, 114
- profile
  - vxg::cloud::agent::proto::video\_stream\_config, 230
- profiles
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 95
- props
  - cloud-agent-minimal.cc, 244
- ptr
  - vxg::cloud::agent::access\_token, 66
  - vxg::cloud::agent::callback, 79
  - vxg::cloud::agent::event\_stream, 103
  - vxg::cloud::agent::manager, 128
  - vxg::cloud::agent::media::rtsp\_stream, 180
  - vxg::cloud::agent::media::stream, 197
  - vxg::media::stream, 203
  - vxg::media::Streamer::ISink, 111
  - vxg::media::Streamer::ISource, 116
- PtrU
  - vxg::media::Streamer::ISink, 111
- pts
  - vxg::media::Streamer::MediaFrame, 146
- ptz\_action
  - vxg::cloud::agent::proto, 51
- ptz\_preset\_action
  - vxg::cloud::agent::proto, 51
- PULL
  - vxg::media::Streamer::ISource, 117
- pullFrame
  - vxg::media::ffmpeg::Source, 194
  - vxg::media::Streamer::ISource, 119
- PUSH
  - vxg::media::Streamer::ISource, 117
- pushFrame
  - vxg::media::Streamer::ISource, 119
- pwr\_frequency
  - vxg::cloud::agent::proto::video\_caps, 219
  - vxg::cloud::agent::proto::video\_config, 226
- quality
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 95
  - vxg::cloud::agent::proto::video\_stream\_config, 230
- query
  - vxg::cloud::utils::uri, 217
- quit
  - cloud-agent-minimal.cc, 244
  - cloud-agent.cc, 246
- record\_export
  - vxg::cloud::agent::media::rtsp\_stream, 182
  - vxg::cloud::agent::media::stream, 199
- record\_get\_list
  - vxg::cloud::agent::media::rtsp\_stream, 182
  - vxg::cloud::agent::media::stream, 200
- record\_needs\_source
  - vxg::cloud::agent::media::stream, 200
- region
  - vxg::cloud::agent::proto::motion\_region, 151
- region\_shape
  - vxg::cloud::agent::proto::motion\_detection\_caps, 148
- regions
  - vxg::cloud::agent::proto::motion\_detection\_config, 149
- reset
  - vxg::logger, 123, 124
- resolutions
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 95
- rows
  - vxg::cloud::agent::proto::motion\_detection\_config, 149
- rtmp\_sink
  - vxg::media::rtmp\_sink, 169
- rtmp\_sink.h, 256
- rtmp\_source.h, 257
- rtsp-stream.h, 258
- rtsp\_source
  - vxg::media::rtsp\_source, 175
- rtsp\_source.h, 259
- rtsp\_stream
  - vxg::cloud::agent::media::rtsp\_stream, 180
- rtsp\_url
  - cloud-agent-minimal.cc, 245

- cloud-agent.cc, [247](#)
- samplerate
  - vxg::media::Streamer::StreamInfo::AudioInfo, [77](#)
- saturation
  - vxg::cloud::agent::proto::video\_caps, [220](#)
  - vxg::cloud::agent::proto::video\_config, [226](#)
- scheme
  - vxg::cloud::utils::uri, [217](#)
- sensitivity
  - vxg::cloud::agent::proto::motion\_detection\_caps, [148](#)
  - vxg::cloud::agent::proto::motion\_region, [151](#)
- set\_eos
  - vxg::media::Streamer::ISink, [114](#)
- set\_eos\_cb
  - vxg::media::Streamer::ISink, [115](#)
- set\_events
  - vxg::cloud::agent::event\_stream, [105](#)
- set\_level
  - vxg::logger, [124](#)
- set\_stream\_config
  - vxg::cloud::agent::media::rtsp\_stream, [183](#)
  - vxg::cloud::agent::media::stream, [200](#)
- set\_thread\_name
  - vxg::cloud::utils, [55](#)
- set\_trigger\_recording
  - vxg::cloud::agent::event\_stream, [106](#)
- sharpness
  - vxg::cloud::agent::proto::video\_caps, [220](#)
  - vxg::cloud::agent::proto::video\_config, [227](#)
- signal
  - vxg::cloud::agent::proto::wifi\_network, [236](#)
- signal\_handler
  - cloud-agent-minimal.cc, [244](#)
  - cloud-agent.cc, [246](#)
- Sink
  - vxg::media::ffmpeg::Sink, [186](#)
- sink\_
  - vxg::media::stream, [205](#)
- SINK\_THREAD\_PRIO
  - vxg::media::Streamer, [64](#)
- smoothing
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, [95](#)
  - vxg::cloud::agent::proto::video\_stream\_config, [230](#)
- snapshot
  - vxg::cloud::agent::event\_config, [101](#)
  - vxg::cloud::agent::proto::event\_caps, [97](#)
- socks4
  - vxg::cloud::agent::access\_token::proxy\_config, [163](#)
- socks5
  - vxg::cloud::agent::access\_token::proxy\_config, [163](#)
- Source
  - vxg::media::ffmpeg::Source, [191](#)
- source\_
  - vxg::media::stream, [205](#)
- spkr
  - vxg::cloud::agent::proto::audio\_caps, [71](#)
- spkr\_mute
  - vxg::cloud::agent::audio\_config, [73](#)
- spkr\_vol
  - vxg::cloud::agent::audio\_config, [73](#)
- SRC\_THREAD\_PRIO
  - vxg::media::Streamer, [64](#)
- srt
  - vxg::cloud::agent::proto::audio\_stream\_config, [75](#)
  - vxg::cloud::agent::proto::stream\_caps::caps\_audio\_object, [92](#)
- ssid
  - vxg::cloud::agent::proto::wifi\_network, [236](#)
- ST\_ANY
  - vxg::media::Streamer::StreamInfo, [211](#)
- ST\_AUDIO
  - vxg::media::Streamer::StreamInfo, [211](#)
- ST\_DATA
  - vxg::media::Streamer::StreamInfo, [211](#)
- ST\_UNKNOWN
  - vxg::media::Streamer::StreamInfo, [211](#)
- ST\_VIDEO
  - vxg::media::Streamer::StreamInfo, [211](#)
- start
  - vxg::cloud::agent::event\_stream, [106](#)
  - vxg::cloud::agent::manager, [140](#)
  - vxg::cloud::agent::media::rtsp\_stream, [183](#)
- start\_record
  - vxg::cloud::agent::media::rtsp\_stream, [183](#)
  - vxg::cloud::agent::media::stream, [201](#)
- stateful
  - vxg::cloud::agent::manager::event\_state::event\_state\_caps, [102](#)
- statefull
  - vxg::cloud::agent::proto::event\_caps, [97](#)
- std, [25](#)
  - make\_unique, [43](#)
- stop
  - vxg::cloud::agent::event\_stream, [107](#)
  - vxg::cloud::agent::manager, [141](#)
  - vxg::media::ffmpeg::Sink, [190](#)
  - vxg::media::ffmpeg::Source, [194](#)
- stop\_record
  - vxg::cloud::agent::media::rtsp\_stream, [184](#)
  - vxg::cloud::agent::media::stream, [201](#)
- stream
  - vxg::cloud::agent::event\_config, [101](#)
  - vxg::cloud::agent::media::stream, [197](#)
  - vxg::cloud::agent::proto::audio\_stream\_config, [75](#)
  - vxg::cloud::agent::proto::event\_caps, [97](#)
  - vxg::cloud::agent::proto::video\_stream\_config, [230](#)
  - vxg::media::stream, [203](#)
- stream.h, [260](#), [261](#)
- StreamError
  - vxg::media::Streamer, [63](#)
- streams

- vxg::cloud::agent::proto::stream\_caps::caps\_audio\_object, vxg::cloud::agent::proto::osd\_caps, 158
  - 92
  - time\_format\_n
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, vxg::cloud::agent::proto, 52
  - 96
  - time\_realtime
  - vxg::cloud::agent::supported\_streams\_config, 214
  - vxg::media::Streamer::MediaFrame, 146
- StreamType
  - vxg::media::Streamer::StreamInfo, 210
- string\_contains
  - vxg::cloud::utils, 55
- string\_endswith
  - vxg::cloud::utils, 55
- string\_format
  - vxg::cloud::utils, 56
- string\_replace
  - vxg::cloud::utils, 56
- string\_split
  - vxg::cloud::utils, 56
- string\_startswith
  - vxg::cloud::utils, 56
- string\_tolower
  - vxg::cloud::utils, 56
- string\_toupper
  - vxg::cloud::utils, 56
- string\_trim
  - vxg::cloud::utils, 57
- string\_urldecode
  - vxg::cloud::utils, 57
- string\_urlencode
  - vxg::cloud::utils, 57
- syslog\_ident
  - vxg::logger::options, 154
- system\_id
  - vxg::cloud::agent::osd\_config, 161
  - vxg::cloud::agent::proto::osd\_caps, 157
- system\_id\_text
  - vxg::cloud::agent::osd\_config, 161
  - vxg::cloud::agent::proto::osd\_caps, 157
- tcp\_logsink\_enabled
  - vxg::logger::options, 154
- tcp\_logsink\_host
  - vxg::logger::options, 154
- tcp\_logsink\_port
  - vxg::logger::options, 154
- tdn
  - vxg::cloud::agent::proto::video\_caps, 220
  - vxg::cloud::agent::proto::video\_config, 227
- TF\_12H
  - vxg::cloud::agent::proto, 52
- TF\_24H
  - vxg::cloud::agent::proto, 52
- TF\_INVALID
  - vxg::cloud::agent::proto, 52
- time
  - vxg::cloud, 44
  - vxg::cloud::agent::osd\_config, 161
  - vxg::cloud::agent::proto::osd\_caps, 158
- time\_format
  - vxg::cloud::agent::osd\_config, 161
- time\_format\_n
- time\_realtime
- timebase
  - vxg::media::Streamer::StreamInfo::AudioInfo, 77
  - vxg::media::Streamer::StreamInfo::VideoInfo, 233
- timescale
  - vxg::media::Streamer::MediaFrame, 146
- tm
  - vxg::cloud::agent::ptz\_command, 164
- to\_double
  - vxg::cloud::utils::time, 61
- to\_iso
  - vxg::cloud::utils::time, 61
- to\_iso2
  - vxg::cloud::utils::time, 61
- to\_iso\_8601
  - vxg::cloud::utils::time, 61
- to\_iso\_local
  - vxg::cloud::utils::time, 61
- to\_iso\_packed
  - vxg::cloud::utils::time, 61
- to\_json
  - alter\_bool, 69
- token
  - vxg::cloud::agent::ptz\_preset, 167
- tp\_start
  - vxg::cloud::agent::proto::video\_clip\_info, 222
- tp\_stop
  - vxg::cloud::agent::proto::video\_clip\_info, 222
- trace
  - vxg::logger, 125
- trigger
  - vxg::cloud::agent::proto::event\_caps, 98
- trigger\_event
  - vxg::cloud::agent::event\_stream, 107
- type
  - vxg::media::Streamer::MediaFrame, 147
  - vxg::media::Streamer::StreamInfo, 211
- UNKNOWN
  - vxg::media::Streamer, 63
- unpack
  - vxg::cloud::utils::motion::map, 143
- unset-helper.h, 262
  - \_\_is\_unset, 264
  - \_\_is\_unset< alter\_bool >, 265
  - \_\_is\_unset< double >, 265
  - \_\_is\_unset< int >, 265
  - \_\_is\_unset< nlohmann::json >, 266
  - \_\_is\_unset< std::nullptr\_t >, 266
  - \_\_is\_unset< std::string >, 266
  - \_\_is\_unset< vxg::cloud::duration >, 266

- \_\_is\_unset< vxg::cloud::time >, 267
- unset\_value\_for, 267
- unset\_value\_for\_impl, 267–269
- UnsetDouble, 269
- UnsetDuration, 269
- UnsetFloat, 270
- UnsetInt, 270
- UnsetInt64, 270
- UnsetString, 270
- UnsetTime, 270
- UnsetUInt64, 270
- unset\_value\_for
  - unset-helper.h, 267
- unset\_value\_for\_impl
  - unset-helper.h, 267–269
- UnsetDouble
  - unset-helper.h, 269
- UnsetDuration
  - unset-helper.h, 269
- UnsetFloat
  - unset-helper.h, 270
- UnsetInt
  - unset-helper.h, 270
- UnsetInt64
  - unset-helper.h, 270
- UnsetString
  - unset-helper.h, 270
- UnsetTime
  - unset-helper.h, 270
- UnsetUInt64
  - unset-helper.h, 270
- user
  - vxg::cloud::utils::uri, 217
- utils.h, 271
- val
  - alter\_bool, 69
- vbr
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 96
  - vxg::cloud::agent::proto::video\_stream\_config, 230
- vbr\_brt
  - vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 96
  - vxg::cloud::agent::proto::video\_stream\_config, 231
- VC\_H264
  - vxg::media::Streamer::StreamInfo, 211
- VC\_UNKNOWN
  - vxg::media::Streamer::StreamInfo, 211
- version
  - vxg::cloud::agent, 45
- vert
  - vxg::cloud::agent::proto::video\_stream\_config, 231
- vert\_flip
  - vxg::cloud::agent::proto::video\_caps, 220
  - vxg::cloud::agent::proto::video\_config, 227
- VF\_H264
  - vxg::cloud::agent::proto, 52
- VF\_H265
  - vxg::cloud::agent::proto, 52
- VF\_INVALID
  - vxg::cloud::agent::proto, 52
- VF\_MJPEG
  - vxg::cloud::agent::proto, 52
- VIDEO
  - vxg::media::Streamer, 63
- video
  - vxg::cloud::agent::proto::stream\_config, 208
  - vxg::cloud::agent::supported\_stream\_config, 213
  - vxg::media::Streamer::StreamInfo, 211
- VIDEO\_AVC\_PPS
  - vxg::media::Streamer, 63
- VIDEO\_AVC\_SPS
  - vxg::media::Streamer, 63
- video\_es
  - vxg::cloud::agent::supported\_streams\_config, 214
- video\_format
  - vxg::cloud::agent::proto, 52
- video\_height
  - vxg::cloud::agent::proto::video\_clip\_info, 223
- VIDEO\_SEQ\_HDR
  - vxg::media::Streamer, 63
- video\_width
  - vxg::cloud::agent::proto::video\_clip\_info, 223
- VideoCodec
  - vxg::media::Streamer::StreamInfo, 211
- vxg, 43
- vxg::cloud, 44
  - duration, 44
  - time, 44
- vxg::cloud::agent, 44
  - version, 45
- vxg::cloud::agent::access\_token, 65
  - api\_uri, 66
  - pack, 66
  - parse, 66
  - ptr, 66
- vxg::cloud::agent::access\_token::proxy\_config, 162
  - socks4, 163
  - socks5, 163
- vxg::cloud::agent::audio\_config, 72
  - caps, 72
  - echo\_cancel, 73
  - mic\_gain, 73
  - mic\_mute, 73
  - spkr\_mute, 73
  - spkr\_vol, 73
- vxg::cloud::agent::callback, 78
  - on\_audio\_file\_play, 80
  - on\_bye, 80
  - on\_cam\_ptz, 80
  - on\_cam\_ptz\_preset, 81
  - on\_cam\_upgrade\_firmware, 81
  - on\_get\_cam\_audio\_config, 82
  - on\_get\_cam\_events\_config, 82
  - on\_get\_cam\_video\_config, 82
  - on\_get\_log, 83

- on\_get\_memorycard\_info, 83
- on\_get\_motion\_detection\_config, 84
- on\_get\_osd\_config, 84
- on\_get\_ptz\_config, 84
- on\_get\_timezone, 85
- on\_get\_wifi\_config, 85
- on\_raw\_msg, 86
- on\_registered, 86
- on\_set\_cam\_audio\_config, 87
- on\_set\_cam\_events\_config, 87
- on\_set\_cam\_video\_config, 87
- on\_set\_motion\_detection\_config, 88
- on\_set\_osd\_config, 88
- on\_set\_timezone, 89
- on\_set\_wifi\_config, 89
- on\_start\_backward\_audio, 89
- on\_stop\_backward\_audio, 90
- on\_trigger\_event, 90
- ptr, 79
- vxdm::cloud::agent::event\_config, 98
  - active, 100
  - caps, 100
  - caps\_eq, 99
  - custom\_event\_name, 100
  - event, 101
  - name, 99
  - name\_eq, 100
  - period, 101
  - snapshot, 101
  - stream, 101
- vxdm::cloud::agent::event\_stream, 103
  - ~event\_stream, 104
  - event\_stream, 104
  - finit, 104
  - get\_events, 104
  - init, 105
  - notify, 105
  - ptr, 103
  - set\_events, 105
  - set\_trigger\_recording, 106
  - start, 106
  - stop, 107
  - trigger\_event, 107
- vxdm::cloud::agent::events\_config, 107
  - enabled, 109
  - events, 109
  - get\_event\_config, 108
- vxdm::cloud::agent::manager, 126
  - \_\_notify\_record\_event, 129
  - \_\_trigger\_periodic\_event, 129
  - \_append\_internal\_custom\_events, 129
  - \_cancel\_direct\_uploads\_by\_ticket, 129
  - \_cancel\_periodic\_event, 129
  - \_cancel\_periodic\_events, 129
  - \_current\_delivery\_mode, 130
  - \_handle\_stream\_stateful\_event, 130
  - \_handle\_stream\_stateless\_event, 130
  - \_init\_events\_states, 130
  - \_load\_events\_configs, 130
  - \_lookup\_event\_stream, 130
  - \_lookup\_event\_stream\_by\_event, 130
  - \_request\_direct\_upload\_snapshot, 131
  - \_request\_direct\_upload\_video, 131
  - \_schedule\_direct\_upload, 131
  - \_schedule\_periodic\_event, 131
  - \_schedule\_periodic\_events, 131
  - \_stop\_all\_event\_streams, 131
  - \_stop\_all\_streams, 131
  - \_stop\_stream, 132
  - \_update\_direct\_upload\_queue\_latency, 132
  - \_update\_event\_stream\_configs, 132
  - \_update\_events\_configs, 132
  - \_update\_storage\_status, 132
  - create, 132
  - direct\_upload\_sync\_cb, 133
  - handle\_event\_meta\_file, 133
  - handle\_event\_snapshot, 133
  - handle\_stream\_event, 133
  - lookup\_stream, 133
  - notify\_event, 134
  - on\_audio\_file\_play, 134
  - on\_cam\_memorycard\_recording, 134
  - on\_cam\_memorycard\_synchronize, 134
  - on\_cam\_memorycard\_synchronize\_cancel, 134
  - on\_cam\_ptz, 134
  - on\_cam\_ptz\_preset, 135
  - on\_cam\_upgrade\_firmware, 135
  - on\_closed, 135
  - on\_direct\_upload\_url, 135
  - on\_get\_cam\_audio\_config, 135
  - on\_get\_cam\_events\_config, 135
  - on\_get\_cam\_memorycard\_timeline, 136
  - on\_get\_cam\_video\_config, 136
  - on\_get\_log, 136
  - on\_get\_motion\_detection\_config, 136
  - on\_get\_osd\_config, 136
  - on\_get\_ptz\_config, 136
  - on\_get\_stream\_by\_event, 136
  - on\_get\_stream\_caps, 137
  - on\_get\_stream\_config, 137
  - on\_get\_supported\_streams, 137
  - on\_get\_timezone, 137
  - on\_get\_wifi\_config, 137
  - on\_prepared, 137
  - on\_raw\_message, 137
  - on\_registered, 138
  - on\_set\_activity, 138
  - on\_set\_cam\_audio\_config, 138
  - on\_set\_cam\_events\_config, 138
  - on\_set\_cam\_video\_config, 138
  - on\_set\_log\_enable, 138
  - on\_set\_motion\_detection\_config, 138
  - on\_set\_osd\_config, 139
  - on\_set\_periodic\_events, 139
  - on\_set\_stream\_by\_event, 139
  - on\_set\_stream\_config, 139

- on\_set\_timezone, 139
- on\_set\_wifi\_config, 139
- on\_start\_backward, 139
- on\_stop\_backward, 140
- on\_stream\_start, 140
- on\_stream\_stop, 140
- on\_trigger\_event, 140
- on\_update\_preview, 140
- ptr, 128
- start, 140
- stop, 141
- vxg::cloud::agent::manager::event\_state::event\_state\_caps, 102
  - need\_clip, 102
  - need\_snapshot, 102
  - stateful, 102
- vxg::cloud::agent::media, 46
- vxg::cloud::agent::media::rtsp\_stream, 178
  - ~rtsp\_stream, 180
  - get\_snapshot, 181
  - get\_stream\_caps, 181
  - get\_stream\_config, 181
  - get\_supported\_stream, 182
  - ptr, 180
  - record\_export, 182
  - record\_get\_list, 182
  - rtsp\_stream, 180
  - set\_stream\_config, 183
  - start, 183
  - start\_record, 183
  - stop\_record, 184
- vxg::cloud::agent::media::stream, 195
  - ~stream, 197
  - get\_snapshot, 198
  - get\_stream\_caps, 198
  - get\_stream\_config, 199
  - get\_supported\_stream, 199
  - ptr, 197
  - record\_export, 199
  - record\_get\_list, 200
  - record\_needs\_source, 200
  - set\_stream\_config, 200
  - start\_record, 201
  - stop\_record, 201
  - stream, 197
- vxg::cloud::agent::osd\_config, 158
  - alignment, 159
  - bkg\_color, 159
  - bkg\_transp, 160
  - caps, 160
  - date, 160
  - date\_format, 160
  - font\_color, 160
  - font\_size, 161
  - system\_id, 161
  - system\_id\_text, 161
  - time, 161
  - time\_format, 161
  - vxg::cloud::agent::proto, 46
    - A\_BOTTOM, 51
    - A\_INVALID, 51
    - A\_LEFT, 51
    - A\_RIGHT, 51
    - A\_STOP, 51
    - A\_TOP, 51
    - A\_ZOOM\_IN, 51
    - A\_ZOOM\_OUT, 51
    - AF\_AAC, 49
    - AF\_ADPCM, 49
    - AF\_G711A, 49
    - AF\_G711U, 49
    - AF\_INVALID, 49
    - AF\_MP3, 49
    - AF\_NELLY, 49
    - AF\_NELLY16, 49
    - AF\_NELLY8, 49
    - AF\_OPUS, 49
    - AF\_RAW, 49
    - AF\_SPEEX, 49
    - AFF\_AU\_G711U, 48
    - AFF\_INVALID, 48
    - AFF\_MP3, 48
    - AFF\_WAV\_PCM, 48
    - audio\_file\_format, 48
    - audio\_format, 48
    - ES\_ERROR, 49
    - ES\_INVALID, 49
    - ES\_OK, 49
    - ET\_CUSTOM, 49
    - ET\_INVALID, 49
    - ET\_MEMORYCARD, 49
    - ET\_MOTION, 49
    - ET\_NET, 49
    - ET\_RECORD, 49
    - ET\_SOUND, 49
    - ET\_WIFI, 49
    - event\_status, 49
    - event\_type, 49
    - M\_AUTO, 50
    - M\_INVALID, 50
    - M\_OFF, 50
    - M\_ON, 50
    - MCS\_FORMATTING, 50
    - MCS\_INITIALIZATION, 50
    - MCS\_INVALID, 50
    - MCS\_NEED\_FORMAT, 50
    - MCS\_NONE, 50
    - MCS\_NORMAL, 50
    - memorycard\_status, 50
    - mode, 50
    - motion\_region\_shape, 50
    - motion\_sensitivity, 51
    - MR\_ANY, 50
    - MR\_INVALID, 50
    - MR\_RECTANGLE, 50
    - MS\_FRAME, 51



- MS\_INVALID, 51
- MS\_REGION, 51
- name, 53
- PA\_CREATE, 52
- PA\_DELETE, 52
- PA\_GOTO, 52
- PA\_INVALID, 52
- PA\_UPDATE, 52
- ptz\_action, 51
- ptz\_preset\_action, 51
- TF\_12H, 52
- TF\_24H, 52
- TF\_INVALID, 52
- time\_format\_n, 52
- VF\_H264, 52
- VF\_H265, 52
- VF\_INVALID, 52
- VF\_MJPEG, 52
- video\_format, 52
- WFE\_INVALID, 53
- WFE\_OPEN, 53
- WFE\_WEP, 53
- WFE\_WPA, 53
- WFE\_WPA2, 53
- WFE\_WPA2\_ENTERPRISE, 53
- WFE\_WPA\_ENTERPRISE, 53
- wifi\_encryption, 52
- wifi\_list, 48
- wifi\_network\_state, 53
- WNS\_CONNECTED, 53
- WNS\_INITIALIZE\_0, 53
- WNS\_INITIALIZE\_1, 53
- WNS\_INVALID, 53
- WNS\_RECEIVING\_IP, 53
- WNS\_TRY\_CONNECT, 53
- WNS\_UNKNOWN, 53
- vxg::cloud::agent::proto::audio\_caps, 69
  - audio\_file\_formats, 70
  - backward, 70
  - backward\_formats, 70
  - echo\_cancel, 71
  - mic, 71
  - spkr, 71
- vxg::cloud::agent::proto::audio\_stream\_config, 74
  - brt, 75
  - format, 75
  - srt, 75
  - stream, 75
- vxg::cloud::agent::proto::event\_caps, 96
  - periodic, 97
  - snapshot, 97
  - statefull, 97
  - stream, 97
  - trigger, 98
- vxg::cloud::agent::proto::motion\_detection\_caps, 147
  - max\_regions, 147
  - region\_shape, 148
  - sensitivity, 148
- vxg::cloud::agent::proto::motion\_detection\_config, 148
  - caps, 149
  - columns, 149
  - regions, 149
  - rows, 149
- vxg::cloud::agent::proto::motion\_region, 150
  - enabled, 151
  - map, 151
  - region, 151
  - sensitivity, 151
- vxg::cloud::agent::proto::osd\_caps, 155
  - alignment, 156
  - bkg\_color, 156
  - bkg\_transp, 156
  - date, 156
  - date\_format, 156
  - font\_color, 157
  - font\_size, 157
  - system\_id, 157
  - system\_id\_text, 157
  - time, 158
  - time\_format, 158
- vxg::cloud::agent::proto::stream\_caps, 206
  - caps\_audio, 207
  - caps\_video, 207
- vxg::cloud::agent::proto::stream\_caps::caps\_audio\_object, 91
  - brt, 91
  - formats, 92
  - srt, 92
  - streams, 92
- vxg::cloud::agent::proto::stream\_caps::caps\_video\_object, 93
  - brt, 94
  - formats, 94
  - fps, 94
  - gop, 94
  - profiles, 95
  - quality, 95
  - resolutions, 95
  - smoothing, 95
  - streams, 96
  - vbr, 96
  - vbr\_brt, 96
- vxg::cloud::agent::proto::stream\_config, 207
  - audio, 208
  - video, 208
- vxg::cloud::agent::proto::video\_caps, 217
  - brightness, 218
  - contrast, 218
  - horz\_flip, 219
  - ir\_light, 219
  - nr\_level, 219
  - nr\_type, 219
  - pwr\_frequency, 219
  - saturation, 220
  - sharpness, 220
  - tdn, 220



- vert\_flip, [220](#)
  - wb\_type, [220](#)
- vxx::cloud::agent::proto::video\_clip\_info, [221](#)
  - data, [222](#)
  - local\_start, [222](#)
  - local\_stop, [222](#)
  - tp\_start, [222](#)
  - tp\_stop, [222](#)
  - video\_height, [223](#)
  - video\_width, [223](#)
- vxx::cloud::agent::proto::video\_config, [223](#)
  - brightness, [225](#)
  - caps, [225](#)
  - contrast, [225](#)
  - horz\_flip, [225](#)
  - ir\_light, [226](#)
  - nr\_level, [226](#)
  - nr\_type, [226](#)
  - pwr\_frequency, [226](#)
  - saturation, [226](#)
  - sharpness, [227](#)
  - tdn, [227](#)
  - vert\_flip, [227](#)
  - wb\_type, [227](#)
- vxx::cloud::agent::proto::video\_stream\_config, [228](#)
  - brt, [229](#)
  - format, [229](#)
  - fps, [229](#)
  - gop, [229](#)
  - horz, [229](#)
  - profile, [230](#)
  - quality, [230](#)
  - smoothing, [230](#)
  - stream, [230](#)
  - vbr, [230](#)
  - vbr\_brt, [231](#)
  - vert, [231](#)
- vxx::cloud::agent::proto::wifi\_config, [234](#)
  - networks, [234](#)
- vxx::cloud::agent::proto::wifi\_network, [235](#)
  - encryption, [235](#)
  - encryption\_caps, [236](#)
  - mac, [236](#)
  - password, [236](#)
  - signal, [236](#)
  - ssid, [236](#)
- vxx::cloud::agent::ptz\_command, [163](#)
  - action, [164](#)
  - tm, [164](#)
- vxx::cloud::agent::ptz\_config, [164](#)
  - actions, [165](#)
  - maximum\_number\_of\_presets, [165](#)
  - presets, [165](#)
- vxx::cloud::agent::ptz\_preset, [166](#)
  - action, [167](#)
  - name, [167](#)
  - token, [167](#)
- vxx::cloud::agent::supported\_stream\_config, [212](#)
  - audio, [213](#)
  - id, [213](#)
  - video, [213](#)
- vxx::cloud::agent::supported\_streams\_config, [213](#)
  - audio\_es, [214](#)
  - streams, [214](#)
  - video\_es, [214](#)
- vxx::cloud::time\_spec, [53](#)
  - duration, [54](#)
  - precision, [54](#)
- vxx::cloud::utils, [54](#)
  - dirname, [55](#)
  - set\_thread\_name, [55](#)
  - string\_contains, [55](#)
  - string\_endswith, [55](#)
  - string\_format, [56](#)
  - string\_replace, [56](#)
  - string\_split, [56](#)
  - string\_startswith, [56](#)
  - string\_tolower, [56](#)
  - string\_toupper, [56](#)
  - string\_trim, [57](#)
  - string\_urldecode, [57](#)
  - string\_urlencode, [57](#)
- vxx::cloud::utils::gcc\_abi, [57](#)
  - demangle, [57](#)
- vxx::cloud::utils::motion, [58](#)
- vxx::cloud::utils::motion::map, [141](#)
  - map, [142](#)
  - operator=, [143](#)
  - pack, [143](#)
  - unpack, [143](#)
- vxx::cloud::utils::time, [58](#)
  - from\_double, [58](#)
  - from\_iso, [58](#)
  - from\_iso2, [59](#)
  - from\_iso\_packed, [59](#)
  - is\_iso, [59](#)
  - is\_iso\_packed, [59](#)
  - ISO8601\_to\_time, [59](#)
  - iso\_time\_valid, [59](#)
  - max, [59](#)
  - now, [60](#)
  - now\_ISO8601\_UTC, [60](#)
  - now\_ISO8601\_UTC\_packed, [60](#)
  - now\_time\_UTC, [60](#)
  - null, [60](#)
  - time\_to\_ISO8601, [60](#)
  - time\_to\_ISO8601\_packed, [60](#)
  - to\_double, [61](#)
  - to\_iso, [61](#)
  - to\_iso2, [61](#)
  - to\_iso\_8601, [61](#)
  - to\_iso\_local, [61](#)
  - to\_iso\_packed, [61](#)
- vxx::cloud::utils::uri, [215](#)
  - fragment, [216](#)
  - host, [216](#)

- parse, 215
- password, 216
- path, 216
- port, 216
- query, 217
- scheme, 217
- user, 217
- vxg::logger, 120
  - debug, 122
  - error, 122
  - info, 122, 123
  - instance, 123
  - logger\_ptr, 121
  - loglevel, 121
  - lvl\_crit, 121
  - lvl\_debug, 121
  - lvl\_error, 121
  - lvl\_info, 121
  - lvl\_off, 121
  - lvl\_trace, 121
  - lvl\_warn, 121
  - reset, 123, 124
  - set\_level, 124
  - trace, 125
  - warn, 125
- vxg::logger::options, 152
  - crash\_logfile\_path, 153
  - default\_loglevel, 153
  - log\_pattern, 153
  - logfile\_max\_files, 153
  - logfile\_max\_size, 153
  - logfile\_path, 153
  - syslog\_ident, 154
  - tcp\_logsink\_enabled, 154
  - tcp\_logsink\_host, 154
  - tcp\_logsink\_port, 154
- vxg::media, 62
- vxg::media::ffmpeg, 62
- vxg::media::ffmpeg::Sink, 185
  - ~Sink, 186
  - droppable, 186
  - duration, 187
  - error, 187
  - finit, 188
  - init, 188
  - name, 189
  - negotiate, 189
  - Sink, 186
  - stop, 190
- vxg::media::ffmpeg::Source, 190
  - ~Source, 192
  - finit, 192
  - init, 192, 193
  - name, 194
  - negotiate, 194
  - pullFrame, 194
  - Source, 191
  - stop, 194
- vxg::media::rtmp\_sink, 168
  - droppable, 169
  - error, 170
  - init, 170
  - name, 171
  - negotiate, 171
  - rtmp\_sink, 169
- vxg::media::rtmp\_source, 172
  - init, 173
- vxg::media::rtsp\_source, 173
  - ffmpeg\_opts\_, 178
  - init, 177
  - name, 177
  - rtsp\_source, 175
- vxg::media::stream, 202
  - ~stream, 204
  - finit\_sink, 204
  - finit\_source, 204
  - init\_sink, 204
  - init\_source, 205
  - ptr, 203
  - sink\_, 205
  - source\_, 205
  - stream, 203
- vxg::media::Streamer, 62
  - AUDIO, 63
  - AUDIO\_SEQ\_HDR, 63
  - DATA, 63
  - DROP\_BACK, 63
  - DROP\_FRONT, 63
  - DropDirection, 63
  - E\_EOS, 64
  - E\_FATAL, 64
  - E\_NONE, 64
  - FLV, 63
  - MAX, 63
  - MediaType, 63
  - SINK\_THREAD\_PRIO, 64
  - SRC\_THREAD\_PRIO, 64
  - StreamError, 63
  - UNKNOWN, 63
  - VIDEO, 63
  - VIDEO\_AVC\_PPS, 63
  - VIDEO\_AVC\_SPS, 63
  - VIDEO\_SEQ\_HDR, 63
- vxg::media::Streamer::ISink, 110
  - ~ISink, 112
  - droppable, 112
  - duration, 112
  - error, 112
  - finit, 113
  - init, 113
  - ISink, 111
  - name, 113
  - negotiate, 114
  - process, 114
  - ptr, 111
  - PtrU, 111

- set\_eos, 114
- set\_eos\_cb, 115
- vngx::media::Streamer::ISource, 115
  - error, 117
  - finit, 118
  - init, 118
  - ISource, 117
  - Mode, 116
  - mode\_, 120
  - name, 118
  - negotiate, 118
  - ptr, 116
  - PULL, 117
  - pullFrame, 119
  - PUSH, 117
  - pushFrame, 119
- vngx::media::Streamer::MediaFrame, 143
  - data, 145
  - dts, 145
  - duration, 145
  - is\_key, 145
  - len, 146
  - NO\_PTS, 146
  - operator<, 144
  - pts, 146
  - time\_realtime, 146
  - timescale, 146
  - type, 147
- vngx::media::Streamer::StreamInfo, 209
  - AC\_AAC, 210
  - AC\_G711\_A, 210
  - AC\_G711\_U, 210
  - AC\_G726, 210
  - AC\_LPCM, 210
  - AC\_OPUS, 210
  - AC\_UNKNOWN, 210
  - audio, 211
  - AudioCodec, 210
  - DataCodec, 210
  - DC\_ONVIF, 210
  - DC\_UNKNOWN, 210
  - ST\_ANY, 211
  - ST\_AUDIO, 211
  - ST\_DATA, 211
  - ST\_UNKNOWN, 211
  - ST\_VIDEO, 211
  - StreamType, 210
  - type, 211
  - VC\_H264, 211
  - VC\_UNKNOWN, 211
  - video, 211
  - VideoCodec, 211
- vngx::media::Streamer::StreamInfo::AudioInfo, 76
  - bitrate, 76
  - channels, 77
  - codec, 77
  - extradata, 77
  - samplerate, 77
  - timebase, 77
- vngx::media::Streamer::StreamInfo::VideoInfo, 231
  - bitrate, 232
  - codec, 232
  - extradata, 232
  - framerate, 233
  - height, 233
  - timebase, 233
  - width, 233
- vngx\_cloud\_token
  - cloud-agent-minimal.cc, 245
  - cloud-agent.cc, 247
- warn
  - vngx::logger, 125
- wb\_type
  - vngx::cloud::agent::proto::video\_caps, 220
  - vngx::cloud::agent::proto::video\_config, 227
- WFE\_INVALID
  - vngx::cloud::agent::proto, 53
- WFE\_OPEN
  - vngx::cloud::agent::proto, 53
- WFE\_WEP
  - vngx::cloud::agent::proto, 53
- WFE\_WPA
  - vngx::cloud::agent::proto, 53
- WFE\_WPA2
  - vngx::cloud::agent::proto, 53
- WFE\_WPA2\_ENTERPRISE
  - vngx::cloud::agent::proto, 53
- WFE\_WPA\_ENTERPRISE
  - vngx::cloud::agent::proto, 53
- width
  - vngx::media::Streamer::StreamInfo::VideoInfo, 233
- wifi\_encryption
  - vngx::cloud::agent::proto, 52
- wifi\_list
  - vngx::cloud::agent::proto, 48
- wifi\_network\_state
  - vngx::cloud::agent::proto, 53
- WNS\_CONNECTED
  - vngx::cloud::agent::proto, 53
- WNS\_INITIALIZE\_0
  - vngx::cloud::agent::proto, 53
- WNS\_INITIALIZE\_1
  - vngx::cloud::agent::proto, 53
- WNS\_INVALID
  - vngx::cloud::agent::proto, 53
- WNS\_RECEIVING\_IP
  - vngx::cloud::agent::proto, 53
- WNS\_TRY\_CONNECT
  - vngx::cloud::agent::proto, 53
- WNS\_UNKNOWN
  - vngx::cloud::agent::proto, 53