Developer manual

（Onvif Client Library）

Happytimesoft Technology Co., LTD

Declaration

Since the product version upgrade or other reasons, this manual will subsequently be updated. Unless otherwise agreed, this manual only as a guide, this manual all statements, information, recommendations do not constitute any express or implied warranties。

Table of Contents

Chapter 1 Build

1.1 Windows Platform

Use VS2015 or later open OnvifClientLibrary.sln to build

1.2 Linux Platform

In the source code directory execute the following command to compile:

cd OnvifClientLibrary

Make (Compile the dynamic library)

Or

Make -f static.mk (Compile the static library)

Note : if you enable HTTPS function, please first compile and install OPENSSL library. The download link:

http://www.openssl.org/source/openssl-1.0.1s.tar.gz

1.3 Android Platform

cd OnvifClientLibrary

make -f android.mk (Need to install android build environment)

Or

use QtCreator open OnvifClientLibrary.pro to build (QT 5.3.0 or later for android needs to be installed)

Note : if you enable HTTPS function, please first cross compile OPENSSL library. The download link:

http://www.openssl.org/source/openssl-1.0.1s.tar.gz

1.4 MAC Platform

cd OnvifClientLibrary

make -f macos.mk

Note : if you enable HTTPS function, please first compile and install

OPENSSL library. The download link:

http://www.openssl.org/source/openssl-1.0.1s.tar.gz

1.5 IOS Platform

*use QtCreator open OnvifClientLibraryIOS.pro to build (*QT 5.3.0 or later for IOS needs to be installed*)*

Note : if you enable HTTPS function, please first cross compile OPENSSL library. The download link:

http://www.openssl.org/source/openssl-1.0.1s.tar.gz

1.6 Embedded Platform

Modify Makefile to specify cross-compiler, then make

Note : if you enable HTTPS function, please first cross compile OPENSSL library. The download link:

http://www.openssl.org/source/openssl-1.0.1s.tar.gz

Chapter 2 Data Structure

2.1 ONVIF_DEVICE

The ONVIF_DEVICE structure used to store the device configuration parameters.

```
typedef struct
{
    unsigned int    local_ip;           // local ip address to connect to server, network byte order

    DEVICE_BINFO    binfo;      // device basic information

    // request
    char            username[32]; // login user name, set by user
    char            password[32]; // login password, set by user

    BOOL            authFailed;     // when login auth failed, set by onvif stack

    ONVIF_Profile * curProfile;     // current profile pointer, the default pointer the first profile, user can set it

    /*********************************************************/

    ONVIF_VideoSource * video_src;                  // the list of video source
    ONVIF_AudioSource * audio_src;                  // the list of audio source
    ONVIF_Profile * profiles;                       // the list of profile
    ONVIF_VideoSourceConfiguration * video_src_cfg; // the list of video source configuration
    ONVIF_AudioSourceConfiguration * audio_src_cfg; // the list of audio source configuration
    ONVIF_VideoEncoderConfiguration * video_enc;    // the list of video encoder configuration
    ONVIF_AudioEncoderConfiguration * audio_enc;    // the list of audio encoder configuration
    ONVIF_PTZNode * ptznodes;                       // the list of ptz node
    ONVIF_PTZConfiguration * ptz_cfg;               // the list of ptz configuration

    /*********************************************************/
```

```
    ONVIF_EVENT     events;                         // event information
    onvif_DeviceInformation     DeviceInformation;  // device information
    onvif_Capabilities  Capabilities;               // device capabilities
} ONVIF_DEVICE;
```

Field	Description
local_ip	local ip address to connect to server, network byte order
binfo	device basic information
username	login user name, set by user
password	login password, set by user
authFailed	when login auth failed, set by onvif stack
curProfile	current profile pointer, the default pointer the first profile, user can set it
video_src	Video source list
audio_src	Audio source list
profiles	Media profiles list
video_src_cfg	Video source configurations list
audio_src_cfg	Audio source configurations list
video_enc	Video encoder list
audio_enc	Audio encoder list
ptznodes	PTZ nodes list
ptz_cfg	PTZ configurations list
events	Onvif event message
DeviceInformation	Device information
Capabilities	Device capabilities

2.2 Other Data structure

Other ONVIF standard data structure, please refer **onvif_cm.h** file

2.3 Request Data structure

ONVIF request data structure, please refer **onvif_req.h** file

2.4 Response Data structure

ONVIF response data structure, please refer **onvif_res.h** file

Chapter 3 API Interface

3.1 ONVIF Standard Interface

All onvif standard interfaces please refer onvif_cln.h file.

For example GetCapabilities, the interface prototype is as follow:

ONVIF_API BOOL **onvif_GetCapabilities**
(
ONVIF_DEVICE * p_dev,
GetCapabilities_REQ * p_req,
GetCapabilities_RES * p_res
);

p_dev : the requested device
p_req : the request parameters
p_res : the response parameters

Note : Return result in the structure with a pointer that the caller have responsibility to release

3.2 Simplified API interface

ONVIF_API BOOL **GetCapabilities**(ONVIF_DEVICE * p_dev);
Get device capability set, the result store to Capabilities field of p_dev.

ONVIF_API BOOL **GetServices**(ONVIF_DEVICE * p_dev);
Get device service capability set, the result store to Capabilities field of p_dev

ONVIF_API BOOL **GetDeviceInformation**(ONVIF_DEVICE * p_dev);
Get device information, the result store to DeviceInformation field of p_dev

ONVIF_API BOOL **GetProfiles**(ONVIF_DEVICE * p_dev);
Get device media profiles, the result store to profiles field of p_dev

ONVIF_API BOOL **GetStreamUris**(ONVIF_DEVICE * p_dev);
Get the device rtsp stream address of each media profile, the result store to stream_uri field of

ONVIF_PROFILE structure

ONVIF_API BOOL **GetVideoSourceConfigurations**(ONVIF_DEVICE * p_dev);
Get device video source configurations, the result store to video_src_cfg field of p_dev

ONVIF_API BOOL **GetAudioSourceConfigurations**(ONVIF_DEVICE * p_dev);
Get device audio source configurations, the result store to audio_src_cfg field of p_dev

ONVIF_API BOOL **GetVideoEncoderConfigurations**(ONVIF_DEVICE * p_dev);
Get device video encoder configurations, the result store to video_enc field of p_dev

ONVIF_API BOOL **GetAudioEncoderConfigurations**(ONVIF_DEVICE * p_dev);
Get device audio encoder configurations, the result store to audio_enc field of p_dev

ONVIF_API BOOL **GetNodes**(ONVIF_DEVICE * p_dev);
Get device PTZ nodes, the result store to ptznodes field of p_dev

ONVIF_API BOOL **GetConfigurations**(ONVIF_DEVICE * p_dev);
Get device PTZ configurations, the result store to ptz_cfg field of p_dev

ONVIF_API BOOL **GetVideoSources**(ONVIF_DEVICE * p_dev);
Get device video sources, the result store to video_src field of p_dev

ONVIF_API BOOL **GetAudioSources**(ONVIF_DEVICE * p_dev);
Get device audio sources, the result store to audio_src field of p_dev

ONVIF_API BOOL **GetImagingSettings**(ONVIF_DEVICE * p_dev);
Get device image settings, the result store to VideoSource.ImagingSettings field of VIDEO_SRC structure

ONVIF_API BOOL **Subscribe**(ONVIF_DEVICE * p_dev, int index);
Subscribe events, the Subscribers address is:
“http://localip:localport/subscription***index***”
The localport is 30100+***index***

ONVIF_API BOOL **Unsubscribe**(ONVIF_DEVICE * p_dev);
Unsubscribe events

ONVIF_API BOOL **GetSnapshot**
(
ONVIF_DEVICE * p_dev,
const char * profile_token,
unsigned char ** p_buf,
int * buflen
);
Get device snapshot.
profile_token, specify the media profile token
p_buf, the snapshot buffer, the caller have responsibility to release
buflen, the snapshot buffer length

ONVIF_API BOOL **FirmwareUpgrade**(ONVIF_DEVICE * p_dev, const char * filename);
Upgrade device firmware.
filename is the firmware file name.

3.3 Other API Interface

ONVIF_API void **onvif_SetAuthInfo**(ONVIF_DEVICE * p_dev, const char * user, const char * pass);
Set device authenticate information

ONVIF_API void **set_probe_cb**(onvif_probe_cb cb, void * pdata);
Set device probe callback, the callback prototype is as follow:
typedef void (* onvif_probe_cb)(DEVICE_BINFO * p_res, void * pdata);

ONVIF_API void **set_probe_interval**(int interval);
Set device probe interval, unit is second, default is 30 second.

ONVIF_API int **start_probe**(int interval);
Start device probe task.
interval, specify the device probe interval, unit is second

ONVIF_API void **stop_probe**();
Stop device probe task

ONVIF_API void **send_probe_req**();
Send device probe request message

ONVIF_API void **onvif_set_event_notify_cb**(onvif_event_notify_cb cb, void * pdata);

Set onvif event notify callback, the callback prototype is as follow:

typedef void (* onvif_event_notify_cb)(Notify_REQ * p_req, void * pdata);

ONVIF_API void **onvif_set_subscribe_disconnect_cb**(onvif_subscribe_disconnect_cb cb, void * pdata);

Set event subscribe disconnect notify callback, he callback prototype is as follow:

typedef void (* onvif_subscribe_disconnect_cb)(ONVIF_DEVICE * p_dev, void * pdata);

Other resource malloc and free API interface please refer onvif.h file

The utility API interface please refer onvif_utils.h file

Chapter 4 Example

Onvif client library usage examples refer to OnvifTest.cpp file.