Here is a list of all modules:

- **SL MQTT Server APIs**
  - **SL MQTT Server Events**
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
</table>

### SL MQTT Server APIs
Modules

| SL MQTT Server Events |
## Classes

<table>
<thead>
<tr>
<th>struct</th>
<th>SIMqttServerCbs_t</th>
</tr>
</thead>
<tbody>
<tr>
<td>struct</td>
<td>SIMqttServerParams_t</td>
</tr>
<tr>
<td>struct</td>
<td>SIMqttServerCfg_t</td>
</tr>
</tbody>
</table>
Functions

\[ \_i32 \text{sl\_ExtLib\_MqttServerInit} (_\text{const} \text{SIMqttServerCfg\_t} \*\text{cfg}, \_\text{const} \text{SIMqttServerCbs\_t} \*\text{cbs}) \]

\[ \_i32 \text{sl\_ExtLib\_MqttTopicEnroll} (_\text{const} \text{char} \*\text{topic}) \]

\[ \_i32 \text{sl\_ExtLib\_MqttTopicDisenroll} (_\text{const} \text{char} \*\text{topic}) \]

\[ \_i32 \text{sl\_ExtLib\_MqttServerSend} (_\text{const} \text{char} \*\text{topic}, _\text{const} \text{void} \*\text{data}, _i32 \text{len}, _u8 \text{qos}, \text{bool} \text{retain}, _u32 \text{flags}) \]
**Function Documentation**

```c
_i32 sl_ExtLib_MqttServerInit (_const SlMqttServerCfg_t * cfg,
                               _const SlMqttServerCbs_t * cbs)
```

Initialize the SL MQTT Server Implementation. A caller must initialize the MQTT Server implementation prior to using its services.

**Parameters:**

- `[in] cfg` refers to configuration parameters
- `[in] cbs` callbacks into server application

**Returns:**

Success (0) or Failure (-1)

```c
_i32 sl_ExtLib_MqttServerSend (_const char * topic,
                                _const void * data,
                                _i32 len,
                                _u8 qos,
                                bool retain,
                                _u32 flags)
```

PUBLISH a named message to a client. In addition to the PUBLISH specific parameters, the caller can indicate whether the routine should block until the time, the message has been acknowledged by the client. This is applicable only for non-QoS0 messages.

In case, the app has chosen not to await for the ACK from the client, the SL MQTT Server implementation will notify the app about the received ACK through the callback routine.
Note:

Only QoS0 and QoS1 messages are supported.

Parameters:

- [in] **topic** topic of the data to be published. It is NULL terminated.
- [in] **data** binary data to be published
- [in] **len** length of the data
- [in] **qos** QoS for the publish message
- [in] **retain** Retain bit in the PUBLISH message sent out
- [in] **flags** Reserved for future use.

Returns:

Success(0) or Failure(-1).

```c
_i32 sl_ExtLib_MqttTopicDisenroll ( _const char * topic )
```

DisEnroll a topic. App can cancel the previous enrollment of a topic and the SL layer will now stop forwarding data published for that topic to the server by the connected clients. This is analogous to an unsubscription by an MQTT client.

Parameters:

- [in] **topic** disenrollment topic. It is NULL terminated.

Returns:

Success(0) or Failure(-1).

```c
_i32 sl_ExtLib_MqttTopicEnroll ( _const char * topic )
```

Enroll a topic to receive data. App can enroll a topic of interest and the SL layer will forward to the app any data subsequently published for the enrolled topic to the server by any of the connected MQTT clients.
This is analogous or similar to a subscription to a topic by an MQTT client.

**Parameters:**

[in] **topic** enrollment topic for data to be received. It is NULL terminated.

**Returns:**

Success(0) or Failure (-1).
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
<th>Defines</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SL MQTT Server</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Events</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SL MQTT Server APIs**
# Defines

```
#define SL_MQTT_SRVR_EVT_PUBACK 0x11
#define SL_MQTT_SRVR_EVT_NOCONN 0x12
```
Define Documentation

#define SL_MQTT_SRVR_EVT_NOCONN  0x12
Server has lost network connection with a client

#define SL_MQTT_SRVR_EVT_PUBACK  0x11
PUBACK has been received from a client

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by doxygen 1.8.0
#include <sl_mqtt_server.h>

List of all members.
### Public Attributes

<table>
<thead>
<tr>
<th>Function</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sl_ExtLib_MqttConn</code></td>
<td><code>_u16(* clientId_len, _const char *username_str, _i32 username_len,</code></td>
</tr>
<tr>
<td></td>
<td><code>_const char *password_str, _i32 password_len, void **usr)</code></td>
</tr>
<tr>
<td><code>sl_ExtLib_MqttRecv</code></td>
<td><code>_const char *topstr, _i32 toplen, _const void *payload, _i32 pay_len,</code></td>
</tr>
<tr>
<td></td>
<td><code>bool dup, _u8 qos, bool retain)</code></td>
</tr>
<tr>
<td><code>sl_ExtLib_MqttDisconn</code></td>
<td><code>(void *usr, bool due2err)</code></td>
</tr>
<tr>
<td><code>sl_ExtLib_MqttEvent</code></td>
<td><code>(void *usr, _i32 evt, _const void *buf, _u32 len)</code></td>
</tr>
</tbody>
</table>
Detailed Description

Callbacks Routines The routines are invoked by SL MQTT Server Implementation onto Server application

Note:
The user applications implement the callbacks that are registered with the libraries. While using the MQTT library, invoking the core library APIs from a callback should be avoided and can lead to lockup scenarios. It is recommended to signal another task from the callback routines invoked from the library and invoke the core library API calls from that task.
Member Data Documentation

_connect _u16(* SlMqttServerCbs_t::sl_ExtLib_MqttConn)(const char *clientId_str,
i32 clientId_len,
const char *username_str,
i32 username_len,
const char *password_str,
i32 password_len,
void **usr)

Connect Request: Callback routine to indicate to server application that a CONNECT request has been received by the server.

\param[in] clientId_str clientId field in the CON NECT message received.
\param[in] clientId_len length of ClientId
\param[in] username_str Username field in teh CON NECT message received.
\param[in] username_len length of username
\param[in] password_str Password field in the CON NECT message received.
\param[in] password_len length of password
\param[out] usr placeholder to provision app's handle to this connection.
\return 0x0000 or 0x0100 for success; Otherwise 0 x02, 0x04 or 0x05 refer to spec for connack codes

void(* SlMqttServerCbs_t::sl_ExtLib_MqttDisconn)(void *usr, bool due2err)

DISCONNECT: Callback routine to indicate to the Server Application that a client has disconnected.

Parameters:

  [in] usr app's handle to this connection.
  [in] due2err set to 1, if connection has been closed, without server receiving a DISCONNECT messsage.

Returns:

  none.
void(* SlMqttServerCbs_t::sl_ExtLib_MqttEvent)(void *usr, _i32 evt)

Indication of event either from the server library or implementation generated. TBD - Reserved for future use.

Parameters:
- [in] usr app's handle to this connection.
- [in] evt identifier to the reported event. Refer to SL MQTT Server Events - TBD
- [in] buf points to buffer
- [in] len length of buffer

Returns:
none.

void(* SlMqttServerCbs_t::sl_ExtLib_MqttRecv)(_const char *topstr, _i32 toplen, _const void *payload, _i32 pay_len, bool dup, _u8 qos, bool retain)

Callback routine to receive a PUBLISH from a client. The server app must provide this routine for the instances where it receives PUBLISH messages from clients. The callback is invoked in the context of the internal SL MQTT server Receive Task.

Parameters:
- [in] topstr name of topic on which PUBLISH is received by the server. Not NULL terminated.
- [in] topllen length of the topic name
- [in] payload refers to payload published by the server.
- [in] pay_len length of the payload.
- [in] dup assert to indicate that it is a duplicate message sent by the client
- [in] qos quality of service of the received published message.
- [in] retain asserted to indicate that a retained message has been received
Returns:
none.

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

- D:/Project/SimpleLink/mqtt/doxygen/server/sl_mqtt_server.h

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by doxygen 1.8.0
# include `<sl_mqtt_server.h>`

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>u32</td>
<td>netconn_flags</td>
</tr>
<tr>
<td>const char *</td>
<td>server_addr</td>
</tr>
<tr>
<td>u16</td>
<td>port_number</td>
</tr>
<tr>
<td>u8</td>
<td>method</td>
</tr>
<tr>
<td>u32</td>
<td>cipher</td>
</tr>
<tr>
<td>u32</td>
<td>n_files</td>
</tr>
<tr>
<td>char * const *</td>
<td>secure_files</td>
</tr>
</tbody>
</table>
Detailed Description

Secure Socket Parameters to open a secure connection
Member Data Documentation

_u32 SIMqttServerParams_t::cipher
Cipher to tcp secured socket

_u8 SIMqttServerParams_t::method
Method to tcp secured socket

_u32 SIMqttServerParams_t::n_files
Number of files for secure transfer

_u32 SIMqttServerParams_t::netconn_flags
Enumerate connection type

_u16 SIMqttServerParams_t::port_number
Port number of MQTT server

char*_const* SIMqttServerParams_t::secure_files

_const char* SIMqttServerParams_t::server_addr
Server Address: URL or IP

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

**Class List**
**Class Index**
**Class Members**

### SIMqttServerCfg_t

**Struct Reference**

**SL MQTT Server APIs**

```c
#include <sl_mqtt_server.h>
```

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>SLMqttServerParams_t server_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>_u16 loopback_port</td>
</tr>
<tr>
<td>_u32 rx_tsk_priority</td>
</tr>
<tr>
<td>_u32 resp_time</td>
</tr>
<tr>
<td>bool aux_debug_en</td>
</tr>
<tr>
<td>_i32(* dbg_print )(_const char *pcFormat,...)</td>
</tr>
</tbody>
</table>
Detailed Description

MQTT Server Lib structure which holds Initialization Data
Member Data Documentation

**bool** `SlMqttServerCfg_t::aux_debug_en`  
Assert to indicate additional debug info

**_i32** (*`SlMqttServerCfg_t::dbg_print`)( _const char *pcFormat,...)  
Print debug information

**_u16** `SlMqttServerCfg_t::loopback_port`  
Loopback port to manage lib internal functioning - Mandatory

**_u32** `SlMqttServerCfg_t::resp_time`  
Reasonable response wait time (seconds) for server

**_u32** `SlMqttServerCfg_t::rx_tsk_priority`  
Priority of the receive task in server

`SIMqttServerParams_t` `SlMqttServerCfg_t::server_info`  
Server information

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

- D:/Project/SimpleLink/mqtt/doxygen/server/sl_mqtt_server.h
Here are the classes, structs, unions and interfaces with brief descriptions:

- SlMqttServerCbs_t
- SlMqttServerCfg_t
- SlMqttServerParams_t

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by doxygen 1.8.0
## Class Index

<table>
<thead>
<tr>
<th>Class List</th>
<th>Class Index</th>
<th>Class Members</th>
</tr>
</thead>
</table>

- **S**

- **S**

  - `SImqttServerCfg_t`
  - `SImqttServerParams_t`

- `SImqttServerCbs_t`

---

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by [doxygen](https://www.doxygen.org/) 1.8.0
Here is a list of all class members with links to the classes they belong to:

- aux_debug_en : `SlMqttServerCfg_t`
- cipher : `SlMqttServerParams_t`
- dbg_print : `SlMqttServerCfg_t`
- loopback_port : `SlMqttServerCfg_t`
- method : `SlMqttServerParams_t`
- n_files : `SlMqttServerParams_t`
- netconn_flags : `SlMqttServerParams_t`
- port_number : `SlMqttServerParams_t`
- resp_time : `SlMqttServerCfg_t`
- rx_tsk_priority : `SlMqttServerCfg_t`
- secure_files : `SlMqttServerParams_t`
- server_addr : `SlMqttServerParams_t`
- server_info : `SlMqttServerCfg_t`
- sl_ExtLib_MqttConn : `SlMqttServerCbs_t`
- sl_ExtLib_MqttDisconn : `SlMqttServerCbs_t`
- sl_ExtLib_MqttEvent : `SlMqttServerCbs_t`
- sl_ExtLib_MqttRecv : `SlMqttServerCbs_t`
sl_mqtt_server

- aux_debug_en : SIMqttServerCfg_t
- cipher : SIMqttServerParams_t
- dbg_print : SIMqttServerCfg_t
- loopback_port : SIMqttServerCfg_t
- method : SIMqttServerParams_t
- n_files : SIMqttServerParams_t
- netconn_flags : SIMqttServerParams_t
- port_number : SIMqttServerParams_t
- resp_time : SIMqttServerCfg_t
- rx_tsk_priority : SIMqttServerCfg_t
- secure_files : SIMqttServerParams_t
- server_addr : SIMqttServerParams_t
- server_info : SIMqttServerCfg_t
- sl_ExtLib_MqttConn : SIMqttServerCbs_t
- sl_ExtLib_MqttDisconn : SIMqttServerCbs_t
- sl_ExtLib_MqttEvent : SIMqttServerCbs_t
- sl_ExtLib_MqttRecv : SIMqttServerCbs_t

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by doxygen 1.8.0
## sl_mqtt_server

### File List

Here is a list of all files with brief descriptions:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D:/Project/SimpleLink/mqtt/doxygen/server/sl_mqtt_server.h</td>
<td></td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by [doxygen](http://www.doxygen.org) 1.8.0
D:/Project/SimpleLink/mqtt/doxygen/server/sl_mqtt_server

File Reference

#include <stdio.h> #include <string.h>
#include <stdbool.h>
#include "simplelink.h"

Go to the source code of this file.
## Classes

<table>
<thead>
<tr>
<th>struct</th>
<th>IMqttServerCbs_t</th>
</tr>
</thead>
<tbody>
<tr>
<td>struct</td>
<td>IMqttServerParams_t</td>
</tr>
<tr>
<td>struct</td>
<td>IMqttServerCfg_t</td>
</tr>
</tbody>
</table>
# Defines

<table>
<thead>
<tr>
<th>Define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SL_MQTT_SRVR_EVT_PUBACK</code></td>
<td>0x11</td>
</tr>
<tr>
<td><code>SL_MQTT_SRVR_EVT_NOCONN</code></td>
<td>0x12</td>
</tr>
<tr>
<td><code>SL_MQTT_SRVR_NETCONN_IP6</code></td>
<td>0x01</td>
</tr>
<tr>
<td><code>SL_MQTT_SRVR_NETCONN_URL</code></td>
<td>0x02</td>
</tr>
<tr>
<td><code>SL_MQTT_SRVR_NETCONN_SEC</code></td>
<td>0x04</td>
</tr>
</tbody>
</table>
## Functions

| _i32 | sl_ExtLib_MqttServerInit (_const SIMqttServerCfg_t *cfg, _const SIMqttServerCbs_t *cbs) |
| _i32 | sl_ExtLib_MqttTopicEnroll (_const char *topic) |
| _i32 | sl_ExtLib_MqttTopicDisenroll (_const char *topic) |
| _i32 | sl_ExtLib_MqttServerSend (_const char *topic, _const void *data, _i32 len, _u8 qos, bool retain, _u32 flags) |
Define Documentation

#define SL_MQTT_SRVR_NETCONN_IP6 0x01
Assert for IPv6 connection, otherwise IPv4

#define SL_MQTT_SRVR_NETCONN_SEC 0x04
Connection to server must be secure (TLS)

#define SL_MQTT_SRVR_NETCONN_URL 0x02
Server address is an URL and not IP address

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by doxygen 1.8.0
Here is a list of all file members with links to the files they belong to:

- `sl_ExtLib_MqttServerInit()`: [sl_mqtt_server.h](#)
- `sl_ExtLib_MqttServerSend()`: [sl_mqtt_server.h](#)
- `sl_ExtLib_MqttTopicDisenroll()`: [sl_mqtt_server.h](#)
- `sl_ExtLib_MqttTopicEnroll()`: [sl_mqtt_server.h](#)
- `SL_MQTT_SRVR_EVT_NOCONN`: [sl_mqtt_server.h](#)
- `SL_MQTT_SRVR_EVT_PUBACK`: [sl_mqtt_server.h](#)
- `SL_MQTT_SRVR_NETCONN_IP6`: [sl_mqtt_server.h](#)
- `SL_MQTT_SRVR_NETCONN_SEC`: [sl_mqtt_server.h](#)
- `SL_MQTT_SRVR_NETCONN_URL`: [sl_mqtt_server.h](#)

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by [doxygen](#) 1.8.0
### sl_mqtt_server

- `sl_Entity_MqttServerInit` : `sl_mqtt_server.h`
- `sl_Entity_MqttServerSend` : `sl_mqtt_server.h`
- `sl_Entity_MqttTopicDisease()` : `sl_mqtt_server.h`
- `sl_Entity_MqttTopicEnroll` : `sl_mqtt_server.h`

---

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by [doxygen](http://www.stackoverflow.com) 1.8.0
• SL_MQTT_SRVR_EVT_NOCONN : `sl_mqtt_server.h`
• SL_MQTT_SRVR_EVT_PUBACK : `sl_mqtt_server.h`
• SL_MQTT_SRVR_NETCONN_IP6 : `sl_mqtt_server.h`
• SL_MQTT_SRVR_NETCONN_SEC : `sl_mqtt_server.h`
• SL_MQTT_SRVR_NETCONN_URL : `sl_mqtt_server.h`
**sl_mqtt_server**

Go to the documentation of this file.

```
00001  /**************************************************************************
00002  Copyright (c) (2014) Texas Instruments Incorporated
00003  All rights reserved not granted herein.
00004
00005  Limited License.
00006
00007  Texas Instruments Incorporated grants a world-wide, royalty-free, non-exclusive
00008  license under copyrights and patents it now or hereafter owns or controls to make,
00009  have made, use, import, offer to sell and sell ("Utilize") this software subject
00010  to the terms herein. With respect to the foregoing patent license, such license
00011  is granted solely to the extent that any such patent is necessary to Utilize the
00012  software alone. The patent license shall not apply to any combinations which
00013  include this software, other than combinations with devices manufactured by or
00014  for TI (©TI Devices©). No hardware patent is licensed hereunder.
00015
00016  Redistributions must preserve existing copyright notices and reproduce this license
```
Redistribution and use in binary form, without modification, are permitted provided that the following conditions are met:

* No reverse engineering, decompilation, or disassembly of this software is permitted with respect to any software provided in binary form.

* Any redistribution and use are licensed by TI for use only with TI Devices.

* Nothing shall obligate TI to provide you with source code for the software licensed and provided to you in object code.

If software source code is provided to you, modification and redistribution of the source code are permitted provided that the following conditions are met:

* Any redistribution and use of the source code, including any resulting derivative works, are licensed by TI for use only with TI Devices.

* Any redistribution and use of any object code compiled from the source code and any resulting derivative works, are licensed by TI for use only with TI Devices.

Neither the name of Texas Instruments Incorporated nor the names of its suppliers may be used to endorse or promote products derived from this software without
specific prior written permission.

DISCLAIMER.

THIS SOFTWARE IS PROVIDED BY TI AND TI’S LICENSORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL TI AND TI’S LICENSORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

#include<stdio.h>
#include<string.h>
#include<stdbool.h>
#include"simplelink.h"

#ifndef__SL_MQTT_SRVR_H__
#define__SL_MQTT_SRVR_H__

#ifdef__cplusplus
extern"C"
{

#include <stdio.h>
#include <string.h>
#include <stdbool.h>
#include "simplelink.h"

#ifdef__cplusplus
extern "C"
{


typedef struct {
    _u16 (*sl_ExtLib_MqttConn)(
        _const char *clientId_str, _i32 clientId_len,
        _const char *username_str, _i32 username_len,
        _const char *password_str, _i32 password_len,
        void **usr);
    void (*sl_ExtLib_MqttRecv)(
        _const char *topstr, _i32 toplen,
        _const void *payload, _i32 pay_len,
        bool dup, _u8 qos,
        bool retain);
    // Note: Double check whether retain, dup, qos would be required for a server app (with no topic management)
    void (*sl_ExtLib_MqttDisconn)(void *usr, bool due2err);
    void (*sl_ExtLib_MqttEvent)(
}
```c
void *usr, _i32 evt, _const void *buf, _u32 len);

} SlMqttServerCbs_t;

typedef struct {
    
    #define SL_MQTT_SRVR_NETCONN _IP6 0x01
    #define SL_MQTT_SRVR_NETCONN _URL 0x02
    #define SL_MQTT_SRVR_NETCONN _SEC 0x04

    _u32    netconn_flags;
    _const char *server_addr;
    _u16    port_number;
    _u8     method;
    _u32    cipher;
    _u32    n_files;
    char * _const *secure_files; /*SL needs 4 files*/

} SlMqttServerParams_t;

} SlMqttServerParams_t;
```c
__i32 sl_ExtLib_MqttServerInit(_const SlMqttServerCfg_t *cfg,
  _const SlMqttServerCbs_t *cbs);

__i32 sl_ExtLib_MqttTopicEnroll(_const char *topic);

static inline __i32 sl_ExtLib_MqttTopicSub(_const char *topic)
{
  return sl_ExtLib_MqttTopicEnroll(topic);
}

__i32 sl_ExtLib_MqttTopicDisenroll(_const char *topic);

static inline __i32 sl_ExtLib_MqttTopicUnsub(_const char *topic)
{
  return sl_ExtLib_MqttTopicDisenroll(topic);
}

__i32 sl_ExtLib_MqttServerSend(_const char *topic, _const void *data, __i32 len,
  __u8 qos, bool retain, __u32 flags);

/* End SL MQTT Server API */
```
#endif // __SL_MQTT_SRVR_H__

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by doxygen 1.8.0
sl_mqtt_server

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>sl_ExtLib_MqttConn</td>
<td>SIMqttServerCbs_t</td>
</tr>
<tr>
<td>sl_ExtLib_MqttDisconn</td>
<td>SIMqttServerCbs_t</td>
</tr>
<tr>
<td>sl_ExtLib_MqttEvent</td>
<td>SIMqttServerCbs_t</td>
</tr>
<tr>
<td>sl_ExtLib_MqttRecv</td>
<td>SIMqttServerCbs_t</td>
</tr>
</tbody>
</table>

This is the complete list of members for `SIMqttServerCbs_t`, including all inherited members.
### SIMqttServerParams_t Member List

This is the complete list of members for `SIMqttServerParams_t`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>cipher</td>
<td><code>SIMqttServerParams_t</code></td>
</tr>
<tr>
<td>method</td>
<td><code>SIMqttServerParams_t</code></td>
</tr>
<tr>
<td>n_files</td>
<td><code>SIMqttServerParams_t</code></td>
</tr>
<tr>
<td>netconn_flags</td>
<td><code>SIMqttServerParams_t</code></td>
</tr>
<tr>
<td>port_number</td>
<td><code>SIMqttServerParams_t</code></td>
</tr>
<tr>
<td>secure_files</td>
<td><code>SIMqttServerParams_t</code></td>
</tr>
<tr>
<td>server_addr</td>
<td><code>SIMqttServerParams_t</code></td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:28:55 for `sl_mqtt_server` by [doxygen] 1.8.0
### SIMqttServerCfg_t Member List

This is the complete list of members for `SIMqttServerCfg_t`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>aux_debug_en</td>
<td><code>SIMqttServerCfg_t</code></td>
</tr>
<tr>
<td>dbg_print</td>
<td><code>SIMqttServerCfg_t</code></td>
</tr>
<tr>
<td>loopback_port</td>
<td><code>SIMqttServerCfg_t</code></td>
</tr>
<tr>
<td>resp_time</td>
<td><code>SIMqttServerCfg_t</code></td>
</tr>
<tr>
<td>rx_tsk_priority</td>
<td><code>SIMqttServerCfg_t</code></td>
</tr>
<tr>
<td>server_info</td>
<td><code>SIMqttServerCfg_t</code></td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:28:55 for sl_mqtt_server by [doxygen](http://www.stackoverflow.com) 1.8.0