Here is a list of all modules:

- **SL MQTT Client API**
  - SL MQTT Client Events
  - SL MQTT Oper Paramters
## SL MQTT Client API

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modules</td>
<td>Classes</td>
<td>Functions</td>
<td></td>
</tr>
</tbody>
</table>

**sl_mqtt_client**
# Modules

<table>
<thead>
<tr>
<th>SL MQTT Client Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>SL MQTT Oper Paramters</td>
</tr>
</tbody>
</table>
Classes

<table>
<thead>
<tr>
<th>struct</th>
<th>SIMqttClientCbs_t</th>
</tr>
</thead>
<tbody>
<tr>
<td>struct</td>
<td>SIMqttWill_t</td>
</tr>
<tr>
<td>struct</td>
<td>SIMqttServer_t</td>
</tr>
<tr>
<td>struct</td>
<td>SIMqttClientLibCfg_t</td>
</tr>
<tr>
<td>struct</td>
<td>SIMqttClientCtxCfg_t</td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>i32</th>
<th>sl_ExtLib_MqttClientInit (const SlMqttClientLibCfg_t *cfg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientExit ()</td>
</tr>
<tr>
<td>void*</td>
<td>sl_ExtLib_MqttClientCtxCreate (const SlMqttClientCtxCfg_t *ctx_cfg, const SlMqttClientCbs_t *msg_cbs, void *app_hndl)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientCtxDelete (void *cli_ctx)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientSet (void *cli_ctx, _i32 param, const void *value, _u32 len)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientGet (void *cli_ctx, _i32 param, void *value, _u32 len)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientConnect (void *cli_ctx, bool clean, _u16 keep_alive_time)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientDisconnect (void *cli_ctx)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientSub (void *cli_ctx, char *const *topics, _u8 *qos, _i32 count)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientUnsub (void *cli_ctx, char *const *topics, _i32 count)</td>
</tr>
<tr>
<td>i32</td>
<td>sl_ExtLib_MqttClientSend (void *cli_ctx, const char *topic, const void *data, _i32 len, char qos, bool retain)</td>
</tr>
</tbody>
</table>
CONNECT to the server. This routine establishes a connection with the server for MQTT transactions. The caller should specify a time-period with-in which the implementation should send a message to the server to keep-alive the connection.

**Parameters:**

- **[in]** `cli_ctx` refers to the handle to the client context
- **[in]** `clean` assert to make a clean start and purge the previous session
- **[in]** `keep_alive_time` the maximum time within which client should send a message to server. The unit of the interval is in seconds.

**Returns:**

- on success, variable header of CONNACK message in network byte order. Lowest Byte[Byte0] contains CONNACK Return Code. Byte1 Contains Session Present Bit. on failure returns(-1)

Create a new client context to connect to a server. A context has to be created prior to invoking the client services.
Parameters:

[in] `ctx_cfg` refers to client context configuration parameters

[in] `msg_cbs` refers to callbacks into application

[in] `app` refers to the application callback to be returned on callback

```c
_i32 sl_ExtLib_MqttClientCtxDelete (void * cli_ctx)
```

Deletes the specified client context.

```
\param[in] cli_ctx refers to client context to be deleted
\return Success (0) or Failure (< 0)
```

```c
_i32 sl_ExtLib_MqttClientDisconnect (void * cli_ctx)
```

DISCONNECT from the server. The caller must use this service to close the connection with the server.

**Parameters:**

[in] `cli_ctx` refers to the handle to the client context

**Returns:**

Success (0) or Failure (< 0)

```c
_i32 sl_ExtLib_MqttClientExit ( )
```

Exit the SL MQTT Implementation.

```
\return Success (0) or Failure (-1)
```

```c
_i32 sl_ExtLib_MqttClientGet (void * cli_ctx,
```
Initialize the SL MQTT Implementation. A caller must initialize the MQTT implementation prior to using its services.

Parameters:
[in] `cfg` refers to client lib configuration parameters

Returns:
Success (0) or Failure (-1)

PUBLISH a named message to the server. 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 server. This is applicable only for non-QoS0 messages.

In case, the app has chosen not to await for the ACK from the server, the SL MQTT implementation will notify the app about the subscription through the callback routine.
**Parameters:**

- [in] `cli_ctx` refers to the handle to the client context
- [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` assert if server should retain the message
- [in] `flags` Command flag. Refer to `sl_mqtt_cl_cmdflags`

**Returns:**

Success(transaction Message ID) or Failure(< 0)

```c
_i32 sl_ExtLib_MqttClientSet ( void * cli_ctx,
                                _i32 param,
                                const void * value,
                                _u32 len
                          )
```

Set parameters in SL MQTT implementation. The caller must configure these parameters prior to invoking any MQTT transaction.

**Note:**

The implementation does not copy the contents referred. Therefore, the caller must ensure that contents are persistent in the memory.

**Parameters:**

- [in] `cli_ctx` refers to the handle to the client context
- [in] `param` identifies parameter to set. Refer to **SL MQTT Oper Paramters**
- [in] `value` refers to the place-holder of value to be set
- [in] `len` length of the value of the parameter
Returns:
Success (0) or Failure (-1)

_i32 sl_ExtLib_MqttClientSub ( void * cli_ctx, char *const * topics, _u8 * qos, _i32 count )

SUBSCRIBE a set of topics. To receive data about a set of topics from the server, the app through this routine must subscribe to those topic names with the server. The caller can indicate whether the routine should block until a time, the message has been acknowledged by the server.

In case, the app has chosen not to await for the ACK from the server, the SL MQTT implementation will notify the app about the subscription through the callback routine.

Parameters:

[**in**] cli_ctx refers to the handle to the client context

[**in**] topics set of topic names to subscribe. It is an array of pointers to NUL terminated strings.

[**in,out**] qos array of qos values for each topic in the same order of the topic array. If configured to await for SUB-ACK from server, the array will contain qos responses for topics from the server.

[**in**] count number of such topics

Returns:
Success(transaction Message ID) or Failure(< 0)
_i32 sl_ExtLib_MqttClientUnsub ( void * cli_ctx,
                         char *const * topics,
                         _i32 count )

UNSUBSCRIBE a set of topics. The app should use this service to stop receiving data for the named topics from the server. The caller can indicate whether the routine should block until a time, the message has been acknowledged by the server.

In case, the app has chosen not to await for the ACK from the server, the SL MQTT implementation will notify the app about the subscription through the callback routine.

**Parameters:**
- [in] **cli_ctx** refers to the handle to the client context
- [in] **topics** set of topics to be unsubscribed. It is an array of pointers to NUL terminated strings.
- [in] **count** number of topics to be unsubscribed

**Returns:**
- Success(transaction Message ID) or Failure(< 0)

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by doxygen 1.8.0
### sl_mqtt_client

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
<th>Defines</th>
</tr>
</thead>
</table>

**SL MQTT Client Events**

**SL MQTT Client API**
Defines

#define SL_MQTT_CL_EVT_PUBACK 0x04
#define SL_MQTT_CL_EVT_PUBCOMP 0x07
#define SL_MQTT_CL_EVT_SUBACK 0x09
#define SL_MQTT_CL_EVT_UNSUBACK 0x0B
### Define Documentation

```c
#define SL_MQTT_CL_EVT_PUBACK  0x04
PUBACK has been received from the server
```

```c
#define SL_MQTT_CL_EVT_PUBCOMP  0x07
PUBCOMP has been received from the server
```

```c
#define SL_MQTT_CL_EVT_SUBACK  0x09
SUBACK has been received from the server
```

```c
#define SL_MQTT_CL_EVT_UNSUBACK  0x0B
UNSUBACK has been received from the server
```
# sl_mqtt_client

<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 Oper Parameters</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SL MQTT Client API</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Defines

<table>
<thead>
<tr>
<th>#define</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SL_MQTT_PARAM_CLIENT_ID</td>
<td>0x01</td>
</tr>
<tr>
<td>SL_MQTT_PARAM_USER_NAME</td>
<td>0x02</td>
</tr>
<tr>
<td>SL_MQTT_PARAM_PASS WORD</td>
<td>0x03</td>
</tr>
<tr>
<td>SL_MQTT_PARAM_TOPIC_QOS1</td>
<td>0x04</td>
</tr>
<tr>
<td>SL_MQTT_PARAM_WILL_PARAM</td>
<td>0x05</td>
</tr>
</tbody>
</table>
## Define Documentation

<table>
<thead>
<tr>
<th>Define</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#define SL_MQTT_PARAM_CLIENT_ID 0x01</code></td>
<td>Refers to Client ID</td>
</tr>
<tr>
<td><code>#define SL_MQTT_PARAM_PASS_WORD 0x03</code></td>
<td>Pass-word of client</td>
</tr>
<tr>
<td><code>#define SL_MQTT_PARAM_TOPIC_QOS1 0x04</code></td>
<td>Set a QoS1 SUB topic</td>
</tr>
<tr>
<td><code>#define SL_MQTT_PARAM_USER_NAME 0x02</code></td>
<td>User name of client</td>
</tr>
<tr>
<td><code>#define SL_MQTT_PARAM_WILL_PARAM 0x05</code></td>
<td>Set a WILL topic, WILL Message, WILL QOS, WILL Retain</td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [doxygen](http://doxygen.github.io) 1.8.0
**sl_mqtt_client**

### SIMqttClientCbs_t

**Struct Reference**

SL MQTT Client API

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

List of all members.
Public Attributes

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sl_ExtLib_MqttRecv</code></td>
<td>void(*app_hndl, const char *topstr, _i32 toplen, const void *payload, _i32 pay_len, bool dup, unsigned char qos, bool retain)</td>
</tr>
<tr>
<td><code>sl_ExtLib_MqttEvent</code></td>
<td>void(*app_hndl, _i32 evt, const void *buf, _u32 len)</td>
</tr>
<tr>
<td><code>sl_ExtLib_MqttDisconn</code></td>
<td>void(*app_hndl)</td>
</tr>
</tbody>
</table>
Detailed Description

Callbacks Routines The routines are invoked by SL Implementation onto Client 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

void(* SlMqttClientCbs_t::sl_ExtLib_MqttDisconn)(void *app_hndl)

Notifies the client app about the termination of MQTT connection. After servicing this callback, the client-app can destroy associated context if it no longer required

Parameters:
  [in] app_hndl  application handle returned

void(* SlMqttClientCbs_t::sl_ExtLib_MqttEvent)(void *app_hndl, _i32 evt, const void *buf, _u32 len)

Indication of event either from the server or implementation generated. These events are notified as part of the processing carried out by the internal recv task of the SL implementation. The application must populate the callback to receive events about the progress made by the SL Mqtt layer.

This handler is used by the SL Mqtt Layer to report acknowledgements from the server, in case, the application has chosen not to block the service invokes till the arrival of the corresponding ACK.

Parameters:
  [in] app_hndl  application handle returned
  [in] evt  identifier to the reported event. Refer to SL MQTT Client Events
  [in] buf  points to buffer
  [in] len  length of buffer

Note:
void(* SlMqttClientCbs_t::sl_ExtLib_MqttRecv)(void *app_hndl, const char *topstr, _i32 toplen, const void *payload, _i32 pay_len, bool dup, unsigned char qos, bool retain)

Callback routine to receive a PUBLISH from the server. The client app must provide this routine for the instances where it has subscribed to certain set of topics from the server. The callback is invoked in the context of the internal SL Receive Task.

**Parameters:**

- `[in]` **app_hndl**: application handle returned
- `[in]` **toplen**: length of the topic name published by the server.
- `[in]` **payload**: refers to payload published by the server.
- `[in]` **pay_len**: length of the payload.
- `[in]` **dup**: assert to indicate that it is re-send by the server
- `[in]` **qoS**: quality of service of the published message
- `[in]` **retain**: asserted to indicate that a retained message has been published

**Returns:**

- none.

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

- D:/Project/SimpleLink/mqtt/doxygen/client/sl_mqtt_client.h

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [doxygen](http://www.stackoverflow.com) 1.8.0
sl_mqtt_client

# include <sl_mqtt_client.h>

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>const char *</td>
<td><strong>will_topic</strong></td>
</tr>
<tr>
<td>const char *</td>
<td><strong>will_msg</strong></td>
</tr>
<tr>
<td>char</td>
<td><strong>will_qos</strong></td>
</tr>
<tr>
<td>bool</td>
<td><strong>retain</strong></td>
</tr>
</tbody>
</table>
## Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td><code>SlMqttWill_t::retain</code></td>
<td>Retain Flag</td>
</tr>
<tr>
<td>const char*</td>
<td><code>SlMqttWill_t::will_msg</code></td>
<td>Will message</td>
</tr>
<tr>
<td>char</td>
<td><code>SlMqttWill_t::will_qos</code></td>
<td>Will Qos</td>
</tr>
<tr>
<td>const char*</td>
<td><code>SlMqttWill_t::will_topic</code></td>
<td>Will Topic</td>
</tr>
</tbody>
</table>

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

- D:/Project/SimpleLink/mqtt/doxygen/client/sl_mqtt_client.h

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by doxygen 1.8.0
#include <sl_mqtt_client.h>

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_u32</code></td>
<td><code>netconn_flags</code></td>
</tr>
<tr>
<td>const char *</td>
<td><code>server_addr</code></td>
</tr>
<tr>
<td><code>_u16</code></td>
<td><code>port_number</code></td>
</tr>
<tr>
<td>char</td>
<td><code>method</code></td>
</tr>
<tr>
<td><code>_u32</code></td>
<td><code>cipher</code></td>
</tr>
<tr>
<td><code>_u32</code></td>
<td><code>n_files</code></td>
</tr>
<tr>
<td>char *const *</td>
<td><code>secure_files</code></td>
</tr>
</tbody>
</table>
Detailed Description

Secure Socket Parameters to open a secure connection
### Member Data Documentation

<table>
<thead>
<tr>
<th>Type</th>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_u32</td>
<td>SIImqttServer_t::cipher</td>
<td>Cipher to tcp secured socket</td>
</tr>
<tr>
<td>char</td>
<td>SIImqttServer_t::method</td>
<td>Method to tcp secured socket</td>
</tr>
<tr>
<td>_u32</td>
<td>SIImqttServer_t::n_files</td>
<td>Number of files for secure transfer</td>
</tr>
<tr>
<td>_u32</td>
<td>SIImqttServer_t::netconn_flags</td>
<td>Enumerate connection type</td>
</tr>
<tr>
<td>_u16</td>
<td>SIImqttServer_t::port_number</td>
<td>Port number of MQTT server</td>
</tr>
<tr>
<td>char* const</td>
<td>SIImqttServer_t::secure_files</td>
<td></td>
</tr>
<tr>
<td>const char*</td>
<td>SIImqttServer_t::server_addr</td>
<td>Server Address: URL or IP</td>
</tr>
</tbody>
</table>

The documentation for this struct was generated from the following file:
#include <sl_mqtt_client.h>

List of all members.
Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>_u16</td>
<td>loopback_port</td>
</tr>
<tr>
<td>_u32</td>
<td>rx_tsk_priority</td>
</tr>
<tr>
<td>_u32</td>
<td>resp_time</td>
</tr>
<tr>
<td>bool</td>
<td>aux_debug_en</td>
</tr>
</tbody>
</table>
| _i32(*)| dbg_print )|(const char *pcFormat,...)
Detailed Description

MQTT Lib structure which holds Initialization Data
Member Data Documentation

**bool** SlMqttClientLibCfg_t::aux_debug_en

Assert to indicate additional debug info

**_i32**(* SlMqttClientLibCfg_t::dbg_print)(const char *pcFormat,...)

Print debug information

**_u16** SlMqttClientLibCfg_t::loopback_port

< Loopback port is used to manage lib internal functioning in case of connections to multiple servers simultaneously is desired. Loopback port = 0, implies connection to only single server Loopback port != 0, implies connection to multiple servers

**_u32** SlMqttClientLibCfg_t::resp_time

Reasonable response time (seconds) from server

**_u32** SlMqttClientLibCfg_t::rx_tsk_priority

Priority of the receive task

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

- D:/Project/SimpleLink/mqtt/doxygen/client/sl_mqtt_client.h

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by
sl_mqtt_client

SIMqttClientCtxCfg_t

Struct Reference

SL MQTT Client API

#include <sl_mqtt_client.h>

List of all members.
## Public Attributes

<table>
<thead>
<tr>
<th>SIMqttServer_t</th>
<th>server_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>mqtt_mode31</td>
</tr>
<tr>
<td>bool</td>
<td>blocking_send</td>
</tr>
</tbody>
</table>
Detailed Description

MQTT client context configuration structure
Member Data Documentation

**bool** `SlMqttClientCtxCfg_t::blocking_send`

Select the mode of operation for send APIs (PUB, SUB, UNSUB). false - callback, true - blocking

**bool** `SlMqttClientCtxCfg_t::mqtt_mode31`

Operate LIB in MQTT 3.1 mode; default is 3.1.1. false - default(3.1.1) & true - 3.1)

**SlMqttServer_t SlMqttClientCtxCfg_t::server_info**

Server information

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

- D:/Project/SimpleLink/mqtt/doxygen/client/sl_mqtt_client.h

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [doxygen](http://www.stackoverflow.com) 1.8.0
**Class List**

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

<table>
<thead>
<tr>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>SlMqttClientCbs_t</td>
</tr>
<tr>
<td>SlMqttClientCtxCfg_t</td>
</tr>
<tr>
<td>SlMqttClientLibCfg_t</td>
</tr>
<tr>
<td>SlMqttServer_t</td>
</tr>
<tr>
<td>SlMqttWill_t</td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [Doxygen](http://www.doxygen.org) 1.8.0
## Class Index

<table>
<thead>
<tr>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMqttClientCtxCfg_t</td>
</tr>
<tr>
<td>SIMqttClientLibCfg_t</td>
</tr>
<tr>
<td>SIMqttServer_t</td>
</tr>
<tr>
<td>SIMqttWill_t</td>
</tr>
<tr>
<td>SIMqttClientCbs_t</td>
</tr>
</tbody>
</table>

 Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client 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 : `SlMqttClientLibCfg_t`
- blocking_send : `SlMqttClientCtxCfg_t`
- cipher : `SlMqttServer_t`
- dbg_print : `SlMqttClientLibCfg_t`
- loopback_port : `SlMqttClientLibCfg_t`
- method : `SlMqttServer_t`
- mqtt_mode31 : `SlMqttClientCtxCfg_t`
- n_files : `SlMqttServer_t`
- netconn_flags : `SlMqttServer_t`
- port_number : `SlMqttServer_t`
- resp_time : `SlMqttClientLibCfg_t`
- retain : `SlMqttWill_t`
- rx_tsk_priority : `SlMqttClientLibCfg_t`
- secure_files : `SlMqttServer_t`
- server_addr : `SlMqttServer_t`
- server_info : `SlMqttClientCtxCfg_t`
- sl_ExtLib_MqttDisconn : `SlMqttClientCbs_t`
- sl_ExtLib_MqttEvent : `SlMqttClientCbs_t`
- sl_ExtLib_MqttRecv : `SlMqttClientCbs_t`
- will_msg : `SlMqttWill_t`
- will_qos : `SlMqttWill_t`
- will_topic : `SlMqttWill_t`
sl_mqtt_client

- aux_debug_en : SIMqttClientLibCfg_t
- blocking_send : SIMqttClientCtxCfg_t
- cipher : SIMqttServer_t
- dbg_print : SIMqttClientLibCfg_t
- loopback_port : SIMqttClientLibCfg_t
- method : SIMqttServer_t
- mqtt_mode31 : SIMqttClientCtxCfg_t
- n_files : SIMqttServer_t
- netconn_flags : SIMqttServer_t
- port_number : SIMqttServer_t
- resp_time : SIMqttClientLibCfg_t
- retain : SIMqttWill_t
- rx_tsk_priority : SIMqttClientLibCfg_t
- secure_files : SIMqttServer_t
- server_addr : SIMqttServer_t
- server_info : SIMqttClientCtxCfg_t
- sl_ExtLib_MqttDisconn : SIMqttClientCbs_t
- sl_ExtLib_MqttEvent : SIMqttClientCbs_t
- sl_ExtLib_MqttRecv : SIMqttClientCbs_t
- will_msg : SIMqttWill_t
- will_qos : SIMqttWill_t
- will_topic : SIMqttWill_t
File List

Here is a list of all files with brief descriptions:

D:/Project/SimpleLink/mqtt/doxygen/client/sl_mqtt_client.h [code]

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by doxygen 1.8.0
sl_mqtt_client

D:/Project/SimpleLink/mqtt/doxygen/client/sl_mqtt_client.h

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>SI_MqttClientCbs_t</th>
</tr>
</thead>
<tbody>
<tr>
<td>struct</td>
<td>SI_MqttWill_t</td>
</tr>
<tr>
<td>struct</td>
<td>SI_MqttServer_t</td>
</tr>
<tr>
<td>struct</td>
<td>SI_MqttClientLibCfg_t</td>
</tr>
<tr>
<td>struct</td>
<td>SI_MqttClientCtxCfg_t</td>
</tr>
</tbody>
</table>
#define SL_MQTT_CL_EVT_PUBACK 0x04
#define SL_MQTT_CL_EVT_PUBCOMP 0x07
#define SL_MQTT_CL_EVT_SUBACK 0x09
#define SL_MQTT_CL_EVT_UNSUBACK 0x0B
#define SL_MQTT_NETCONN_IP6 0x04
#define SL_MQTT_NETCONN_URL 0x08
#define SL_MQTT_NETCONN_SEC 0x10
#define SL_MQTT_PARAM_CLIENT_ID 0x01
#define SL_MQTT_PARAM_USER_NAME 0x02
#define SL_MQTT_PARAM_PASS_WORD 0x03
#define SL_MQTT_PARAM_TOPIC_QOS1 0x04
#define SL_MQTT_PARAM_WILL_PARAM 0x05
### Functions

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientInit</td>
<td>(const SIMqttClientLibCfg_t *cfg)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientExit</td>
<td>()</td>
</tr>
<tr>
<td>void *</td>
<td>sl_ExtLib_MqttClientCtxCreate</td>
<td>(const SIMqttClientCtxCfg_t *ctx_cfg, const SIMqttClientCbs_t *msg_cbs, void *app_hdl)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientCtxDelete</td>
<td>(void *cli_ctx)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientSet</td>
<td>(void *cli_ctx, _i32 param, const void *value, _u32 len)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientGet</td>
<td>(void *cli_ctx, _i32 param, void *value, _u32 len)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientConnect</td>
<td>(void *cli_ctx, bool clean, _u16 keep_alive_time)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientDisconnect</td>
<td>(void *cli_ctx)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientSub</td>
<td>(void *cli_ctx, char *const *topics, _u8 *qos, _i32 count)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientUnsub</td>
<td>(void *cli_ctx, char *const *topics, _i32 count)</td>
</tr>
<tr>
<td>_i32</td>
<td>sl_ExtLib_MqttClientSend</td>
<td>(void *cli_ctx, const char *topic, const void *data, _i32 len, char qos, bool retain)</td>
</tr>
</tbody>
</table>
Define Documentation

#define SL_MQTT_NETCONN_IP6 0x04

Assert for IPv6 connection, otherwise IPv4

#define SL_MQTT_NETCONN_SEC 0x10

Connection to server must be secure (TLS)

#define SL_MQTT_NETCONN_URL 0x08

Server address is an URL and not IP address
Here is a list of all file members with links to the files they belong to:

- `sl_ExtLib_MqttClientConnect()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientCtxCreate()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientCtxDelete()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientDisconnect()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientExit()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientGet()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientInit()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientSend()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientSet()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientSub()` : [sl_mqtt_client.h](#)
- `sl_ExtLib_MqttClientUnsub()` : [sl_mqtt_client.h](#)
- `SL_MQTT_CL_EVT_PUBACK` : [sl_mqtt_client.h](#)
- `SL_MQTT_CL_EVT_PUBCOMP` : [sl_mqtt_client.h](#)
- `SL_MQTT_CL_EVT_SUBACK` : [sl_mqtt_client.h](#)
- `SL_MQTT_CL_EVT_UNSUBACK` : [sl_mqtt_client.h](#)
- `SL_MQTT_NETCONN_IP6` : [sl_mqtt_client.h](#)
- `SL_MQTT_NETCONN_SEC` : [sl_mqtt_client.h](#)
- `SL_MQTT_NETCONN_URL` : [sl_mqtt_client.h](#)
- `SL_MQTT_PARAM_CLIENT_ID` : [sl_mqtt_client.h](#)
- `SL_MQTT_PARAM_PASS_WORD` : [sl_mqtt_client.h](#)
- `SL_MQTT_PARAM_TOPIC_QOS1` : [sl_mqtt_client.h](#)
- `SL_MQTT_PARAM_USER_NAME` : [sl_mqtt_client.h](#)
- `SL_MQTT_PARAM_WILL_PARAM` : [sl_mqtt_client.h](#)
### sl_mqtt_client

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>File List</td>
<td>File Members</td>
<td>All</td>
<td>Functions</td>
</tr>
</tbody>
</table>

- `sl_ExtLib_MqttClientConnect()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientCtxCreate()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientCtxDelete()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientDisconnect()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientExit()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientGet()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientInit()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientSend()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientSet()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientSub()` : `sl_mqtt_client.h`
- `sl_ExtLib_MqttClientUnsub()` : `sl_mqtt_client.h`

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [doxygen](http://www.stackoverflow.com) 1.8.0
- SL_MQTT_CL_EVT_PUBACK : sl_mqtt_client.h
- SL_MQTT_CL_EVT_PUBCOMP : sl_mqtt_client.h
- SL_MQTT_CL_EVT_SUBACK : sl_mqtt_client.h
- SL_MQTT_CL_EVT_UNSUBACK : sl_mqtt_client.h
- SL_MQTT_NETCONN_IP6 : sl_mqtt_client.h
- SL_MQTT_NETCONN_SEC : sl_mqtt_client.h
- SL_MQTT_NETCONN_URL : sl_mqtt_client.h
- SL_MQTT_PARAM_CLIENT_ID : sl_mqtt_client.h
- SL_MQTT_PARAM_PASS_WORD : sl_mqtt_client.h
- SL_MQTT_PARAM_TOPIC_QOS1 : sl_mqtt_client.h
- SL_MQTT_PARAM_USER_NAME : sl_mqtt_client.h
- SL_MQTT_PARAM_WILL_PARAM : sl_mqtt_client.h

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by doxygen 1.8.0
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
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_H__
#define __SL_MQTT_H__

#ifdef __cplusplus
extern "C"
{

#endif

#ifndef __SL_MQTT_H__
#define __SL_MQTT_H__

#endif

#ifdef __cplusplus
}
#endif

extern "C"
{
/* End Client events */

/* Define server structure which holds server address and port number. These values are set by the sl_MqttSet API and retrieved by sl_MqttGet API*/

typedef struct {
    void (*sl_ExtLib_MqttRecv)(void *app_hndl, const char *topstr, _i32 toplen, const void *payload, _i32 pay_len, bool dup, unsigned char qos, bool retain);
    void (*sl_ExtLib_MqttEvent)(void *app_hndl, _i32 evt, const void *buf, _u32 len);
    void (*sl_ExtLib_MqttDisconn)(void *app_hndl);
} SlMqttClientCbs_t;
const char *will_topic;
const char *will_msg;
char will_qos;
bool retain;

} SlMqttWill_t;

typedef struct {

#define SL_MQTT_NETCONN_IP6 0x04
#define SL_MQTT_NETCONN_URL 0x08
#define SL_MQTT_NETCONN_SEC 0x10

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

} SlMqttServer_t;

typedef struct {

_u16 loopback_port;
_u32 rx_tsk_priority;
_u32 resp_time;
bool aux_debug_en;

_i32 (*dbg_print)(const char
typedef struct
{
    SlMqttServer_t server_info;
    bool mqtt_mode31;
    bool blocking_send;
} SlMqttClientCtxCfg_t;

_i32 sl_ExtLib_MqttClientInit(const SlMqttClientLibCfg_t *cfg);

_i32 sl_ExtLib_MqttClientExit();

void *sl_ExtLib_MqttClientCtxCreate(const SlMqttClientCtxCfg_t *ctx_cfg,
const SlMqttClientCbs_t *msg_cbs,
void *app_hndl);

_i32 sl_ExtLib_MqttClientCtxDelete(void *cli_ctx);

#define SL_MQTT_PARAM_CLIENT_ID 0x01
#define SL_MQTT_PARAM_USER_NAME 0x02
#define SL_MQTT_PARAM_PASS_WORD 0x03
#define SL_MQTT_PARAM_TOPIC_QOS1 0x04
#define SL_MQTT_PARAM_WILL_PARAM 0x05

_i32 sl_ExtLib_MqttClientSet(void *cli_ctx, _i32 param, const void *value, _u32 len);
```c
/*
 * 
 */

_i32 sl_ExtLib_MqttClientGet(void *cli_ctx, _i32 param, void *value, _u32 len);

_i32 sl_ExtLib_MqttClientConnect(void *cli_ctx, bool clean, _u16 keep_alive_time);

_i32 sl_ExtLib_MqttClientDisconnect(void *cli_ctx);

_i32 sl_ExtLib_MqttClientSub(void *cli_ctx, char* const *topics, _u8 *qos, _i32 count);

_i32 sl_ExtLib_MqttClientUnsub(void *cli_ctx, char* const *topics, _i32 count);

_i32 sl_ExtLib_MqttClientSend(void *cli_ctx, const char *topic, const void *data, _i32 len, char qos, bool retain);

static inline _i32 sl_ExtLib_MqttClientPub(void *cli_ctx, const char *topic, const void *data, _i32 len, char qos, bool retain)
{ }
```
return sl_ExtLib_MqttClientS
end(cli_ctx, topic, data, len, qos, retain);

/* End Client API */

#ifdef __cplusplus
}
#endif

#endif // __SL_MQTT_H__

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by doxygen 1.8.0
### SIMqttClientCbs_t Member List

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

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sl_ExtLib_MqttDisconn</code></td>
<td>SIMqttClientCbs_t</td>
</tr>
<tr>
<td><code>sl_ExtLib_MqttEvent</code></td>
<td>SIMqttClientCbs_t</td>
</tr>
<tr>
<td><code>sl_ExtLib_MqttRecv</code></td>
<td>SIMqttClientCbs_t</td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [doxygen](https://www.stackexchange.com) 1.8.0
sl_mqtt_client

SIMqttWill_t Member List

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

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>retain</td>
<td>SIMqttWill_t</td>
</tr>
<tr>
<td>will_msg</td>
<td>SIMqttWill_t</td>
</tr>
<tr>
<td>will_qos</td>
<td>SIMqttWill_t</td>
</tr>
<tr>
<td>will_topic</td>
<td>SIMqttWill_t</td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by doxygen 1.8.0
**SIMqttServer_t Member List**

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

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

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [doxygen](http://www.stackoverflow.com/) 1.8.0
# SIMqttClientLibCfg_t Member List

This is the complete list of members for `SIMqttClientLibCfg_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>SIMqttClientLibCfg_t</code></td>
</tr>
<tr>
<td>dbg_print</td>
<td><code>SIMqttClientLibCfg_t</code></td>
</tr>
<tr>
<td>loopback_port</td>
<td><code>SIMqttClientLibCfg_t</code></td>
</tr>
<tr>
<td>resp_time</td>
<td><code>SIMqttClientLibCfg_t</code></td>
</tr>
<tr>
<td>rx_tsk_priority</td>
<td><code>SIMqttClientLibCfg_t</code></td>
</tr>
</tbody>
</table>

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [Doxygen](https://www.doxygen.org) 1.8.0
SIMqttClientCtxCfg_t Member List

This is the complete list of members for **SIMqttClientCtxCfg_t**, including all inherited members.

- blocking_send
- mqtt_mode31
- server_info

Generated on Thu Jan 15 2015 18:26:27 for sl_mqtt_client by [doxygen](http://www.doxygen.org) 1.8.0