Here is a list of all modules:

<table>
<thead>
<tr>
<th>Modules</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Client</td>
<td>HTTP/1.1 Client interface</td>
</tr>
</tbody>
</table>

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTP Client

HTTP/1.1 Client interface. More...
### Classes

<table>
<thead>
<tr>
<th>Struct Type</th>
<th>Alias</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>struct</td>
<td><strong>HTTPCli_Field</strong></td>
<td>HTTPCli Request Header Field. <a href="#">More...</a></td>
</tr>
<tr>
<td>struct</td>
<td><strong>HTTPCli_StatusHandler</strong></td>
<td>HTTPCli Response status code handlers. <a href="#">More...</a></td>
</tr>
<tr>
<td>struct</td>
<td><strong>HTTPCli_ContentHandler</strong></td>
<td>HTTPCli Content Handler type. <a href="#">More...</a></td>
</tr>
<tr>
<td>struct</td>
<td><strong>HTTPCli_Struct</strong></td>
<td>HTTPCli instance type. <a href="#">More...</a></td>
</tr>
<tr>
<td>struct</td>
<td><strong>HTTPCli_Params</strong></td>
<td>HTTPCli instance parameters. <a href="#">More...</a></td>
</tr>
<tr>
<td>struct</td>
<td><strong>HTTPCli_SecureParams</strong></td>
<td>HTTPCli secure parameters for TLS. <a href="#">More...</a></td>
</tr>
</tbody>
</table>
## Macros

<table>
<thead>
<tr>
<th>Macro Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>HTTPCli_BUF_LEN</code></td>
<td>128</td>
</tr>
<tr>
<td><code>HTTPCli_CERT_NAME_LEN</code></td>
<td>16</td>
</tr>
<tr>
<td><code>HTTPCli_ESOCKETFAIL</code></td>
<td>(-101)</td>
</tr>
<tr>
<td>Socket create failed.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_ECONNECTFAIL</code></td>
<td>(-102)</td>
</tr>
<tr>
<td>Connect failed.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_ESENDFAIL</code></td>
<td>(-103)</td>
</tr>
<tr>
<td>Send failed.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_ERECVFAIL</code></td>
<td>(-104)</td>
</tr>
<tr>
<td>Recieve failed.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_ETLSFAIL</code></td>
<td>(-105)</td>
</tr>
<tr>
<td>TLS create failed.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_EHOSTNAME</code></td>
<td>(-106)</td>
</tr>
<tr>
<td>Cannot get IP address from host name.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_ESENDBUFSMALL</code></td>
<td>(-107)</td>
</tr>
<tr>
<td>Send buffer is not big enough.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_ERECVBUFSMALL</code></td>
<td>(-108)</td>
</tr>
<tr>
<td>Recieve buffer is not big enough.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_EASYNCMODE</code></td>
<td>(-109)</td>
</tr>
<tr>
<td>Cannot call getResponseStatus in async mode.</td>
<td></td>
</tr>
<tr>
<td><code>HTTPCli_ETHREADFAIL</code></td>
<td>(-110)</td>
</tr>
<tr>
<td>Thread create failed.</td>
<td></td>
</tr>
<tr>
<td>Definition</td>
<td>HTTPCliPROPERTY</td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
</tr>
<tr>
<td>HTTPCli_EPROXYTUNNELFAIL</td>
<td>(-111)</td>
</tr>
<tr>
<td>HTTPCli_ERESPONSEINVALID</td>
<td>(-112)</td>
</tr>
<tr>
<td>HTTPCli_ECONTENTLENLARGE</td>
<td>(-114)</td>
</tr>
<tr>
<td>HTTPCli_EREDIRECTURILONG</td>
<td>(-115)</td>
</tr>
<tr>
<td>HTTPCli_ECONTENTTYPETYPELONG</td>
<td>(-116)</td>
</tr>
<tr>
<td>HTTPCli_ENOCONTENTCALLBACK</td>
<td>(-117)</td>
</tr>
<tr>
<td>HTTPCli_ENOTCHUNKDATA</td>
<td>(-118)</td>
</tr>
<tr>
<td>HTTPCli_EINPROGRESS</td>
<td>(-119)</td>
</tr>
<tr>
<td>HTTPCli_EINTERNALBUFSIZEMALL</td>
<td>(-120)</td>
</tr>
<tr>
<td>HTTPCli_ESETNOTIFYFAIL</td>
<td>(-121)</td>
</tr>
<tr>
<td>HTTPCli_EURILENLONG</td>
<td>(-120)</td>
</tr>
<tr>
<td>HTTPCli_METHOD_GET</td>
<td>&quot;GET&quot;</td>
</tr>
<tr>
<td>Definition</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><code>#define HTTPCli_METHOD_POST &quot;POST&quot;</code></td>
<td>POST method</td>
</tr>
<tr>
<td><code>#define HTTPCli_METHOD_HEAD &quot;HEAD&quot;</code></td>
<td>HEAD method</td>
</tr>
<tr>
<td><code>#define HTTPCli_METHOD_OPTIONS &quot;OPTIONS&quot;</code></td>
<td>OPTIONS method</td>
</tr>
<tr>
<td><code>#define HTTPCli_METHOD_PUT &quot;PUT&quot;</code></td>
<td>PUT method</td>
</tr>
<tr>
<td><code>#define HTTPCli_METHOD_DELETE &quot;DELETE&quot;</code></td>
<td>DELETE method</td>
</tr>
<tr>
<td><code>#define HTTPCli_METHOD_CONNECT &quot;CONNECT&quot;</code></td>
<td>CONNECT method</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_ACCEPT &quot;Accept&quot;</code></td>
<td>Accept field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_ACCEPT_CHARSET &quot;Accept-Charset&quot;</code></td>
<td>Accept-Charset field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_ACCEPT_ENCODING &quot;Accept-Encoding&quot;</code></td>
<td>Accept-Encoding field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_ACCEPT_LANGUAGE &quot;Accept-Language&quot;</code></td>
<td>Accept-Language field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_ACCEPT_RANGES &quot;Accept-Ranges&quot;</code></td>
<td>Accept-Ranges field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_AGE &quot;Age&quot;</code></td>
<td>Age field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_ALLOW &quot;Allow&quot;</code></td>
<td>Allow field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_AUTHORIZATION &quot;Authorization&quot;</code></td>
<td>Authorization field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_CACHE_CONTROL &quot;Cache-Control&quot;</code></td>
<td>Cache-Control field name</td>
</tr>
<tr>
<td><code>#define HTTPCli_FIELD_NAME_CONNECTION &quot;Connection&quot;</code></td>
<td>Connection field name</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_CONTENT_ENCODING</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_CONTENT_LANGUAGE</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_CONTENT_LENGTH</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_CONTENT_LOCATION</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_CONTENT_MD5</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_CONTENT_RANGE</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_CONTENT_TYPE</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_COOKIE</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_DATE</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_ETAG</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_EXPECT</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_EXPIRES</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_FROM</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_HOST</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_IF_MATCH</td>
</tr>
<tr>
<td>Define</td>
<td>HTTPCli_FIELD_NAME_IF_MODIFIED_SINCE</td>
</tr>
</tbody>
</table>
#define HTTPCli_FIELD_NAME_IF_NONE_MATCH "If-None-Match"

#define HTTPCli_FIELD_NAME_IF_RANGE "If-Range"

#define HTTPCli_FIELD_NAME_IF_UNMODIFIED_SINCE "If-Unmodified-Since"

#define HTTPCli_FIELD_NAME_LAST_MODIFIED "Last-Modified"

#define HTTPCli_FIELD_NAME_LOCATION "Location"

#define HTTPCli_FIELD_NAME_MAX_FORWARDS "Max-Forwards"

#define HTTPCli_FIELD_NAME_ORIGIN "Origin"

#define HTTPCli_FIELD_NAME_PRAGMA "Pragma"

#define HTTPCli_FIELD_NAME_PROXY_AUTHENTICATE "Proxy-Authenticate"

#define HTTPCli_FIELD_NAME_PROXY_AUTHORIZATION "Proxy-Authorization"

#define HTTPCli_FIELD_NAME_RANGE "Range"

#define HTTPCli_FIELD_NAMEREFERER "Referer"

#define HTTPCli_FIELD_NAME_RETRY_AFTER "Retry-After"

#define HTTPCli_FIELD_NAME_SERVER "Server"

#define HTTPCli_FIELD_NAME_TE "TE"

#define HTTPCli_FIELD_NAME_TRAILER "Trailer"

#define HTTPCli_FIELD_NAME_TRANSFER_ENCODING "Transfer-Encoding"
```c
#define HTTPCli_FIELD_NAME_UPGRADE "Upgrade"
#define HTTPCli_FIELD_NAME_USER_AGENT "User-Agent"
#define HTTPCli_FIELD_NAME_VARY "Vary"
#define HTTPCli_FIELD_NAME_VIA "Via"
#define HTTPCli_FIELD_NAME_WWW_AUTHENTICATE "WWW-Authenticate"
#define HTTPCli_FIELD_NAME_WARNING "Warning"
#define HTTPCli_FIELD_NAME_X_FORWARDED_FOR "X-Forwarded-For"
#define HTTPCli_TYPE_TLS (0x02)
#define HTTPCli_TYPE_IPV6 (0x04)
#define HTTPCli_FIELD_ID_DUMMY (-11)
#define HTTPCli_FIELD_ID_END (-12)
```
### Typedefs

<table>
<thead>
<tr>
<th>Typedef</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>typedef struct HTTPCli_Field</code></td>
<td>HTTPCli Request Header Field.</td>
</tr>
<tr>
<td><code>typedef void(*HTTPCli_StatusCallback)(void *cli, int status)</code></td>
<td>HTTPCli callback function prototype for status handling. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>typedef struct HTTPCli_StatusHandler</code></td>
<td>HTTPCli Response status code handlers.</td>
</tr>
<tr>
<td><code>typedef int(*HTTPCli_ContentCallback)(void *cli, int status, char *body, int len, bool moreFlag)</code></td>
<td>HTTPCli callback function prototype for content handling. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>typedef struct HTTPCli_ContentHandler</code></td>
<td>HTTPCli Content Handler type.</td>
</tr>
<tr>
<td><code>typedef void(*HTTPCli_RouterCallback)(void *cli, int status, char *uri)</code></td>
<td>HTTPCli callback function prototype for redirection handling. <a href="#">More...</a></td>
</tr>
<tr>
<td><code>typedef void(*HTTPCli.Notify)(long skt, void *cli)</code></td>
<td></td>
</tr>
</tbody>
</table>

[More...](#)
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typedef struct <strong>HTTPCli_Struct</strong></td>
<td>HTTPCli instance type.</td>
</tr>
<tr>
<td>typedef struct <strong>HTTPCli_Params</strong></td>
<td>HTTPCli instance parameters.</td>
</tr>
<tr>
<td>typedef struct <strong>HTTPCli_SecureParams</strong></td>
<td>HTTPCli secure parameters for TLS.</td>
</tr>
<tr>
<td>typedef <strong>HTTPCli_Struct</strong> *</td>
<td>HTTPCli_Handle.</td>
</tr>
</tbody>
</table>
## Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>int HTTPCli_initSockAddr</strong> (struct sockaddr *addr, const char *uri, int flags)</td>
<td>Initialize the socket address structure for the given URI. Supported URI formats are: &quot;<a href="http://www.example.com:8000">http://www.example.com:8000</a>&quot;, &quot;<a href="https://www.example.com:8000">https://www.example.com:8000</a>&quot;, &quot;<a href="https://www.example.com">https://www.example.com</a>&quot;, &quot;www.example.com:8000&quot;, &quot;www.example.com&quot;. For cases where port is not provided, the default port number is set. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>void HTTPCli_construct</strong> (HTTPCli_Struct *cli)</td>
<td>Create a new instance object in the provided structure. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>HTTPCli_Handle HTTPCli_create</strong> ()</td>
<td>Allocate and initialize a new instance object and return its handle. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>int HTTPCli_connect</strong> (HTTPCli_Struct *cli, const struct sockaddr *addr, int flags, const HTTPCli_Params *params)</td>
<td>Open a connection to a HTTP server. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>void HTTPCli_delete</strong> (HTTPCli_Handle cli)</td>
<td>Destory the HTTP client instance and free the previously allocated instance object. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>void HTTPCli_destruct</strong> (HTTPCli_Struct *cli)</td>
<td>Destory the HTTP client instance. <a href="#">More...</a></td>
</tr>
<tr>
<td><strong>void HTTPCli_disconnect</strong> (HTTPCli_Struct *cli)</td>
<td>Disconnect from the HTTP server and destroy the HTTP client instance. <a href="#">More...</a></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>HTTPCli_setRequestFields</code></td>
<td>Set an array of header fields to be sent for every HTTP request. More...</td>
</tr>
<tr>
<td><code>HTTPCli_setResponseFields</code></td>
<td>Set the header fields to filter the response headers. More...</td>
</tr>
<tr>
<td><code>HTTPCli_sendRequest</code></td>
<td>Make a HTTP 1.1 request to the HTTP server. More...</td>
</tr>
<tr>
<td><code>HTTPCli_sendField</code></td>
<td>Send a header field to the HTTP server. More...</td>
</tr>
<tr>
<td><code>HTTPCli_sendRequestBody</code></td>
<td>Send the request message body to the HTTP server. More...</td>
</tr>
<tr>
<td><code>HTTPCli_getResponseStatus</code></td>
<td>Process the response header from the HTTP server and return status. More...</td>
</tr>
<tr>
<td><code>HTTPCli_getResponseField</code></td>
<td>Process the response header from the HTTP server and return field. More...</td>
</tr>
<tr>
<td><code>HTTPCli_readResponseBody</code></td>
<td>Read the parsed response body data from the HTTP server. More...</td>
</tr>
</tbody>
</table>
HTTP server. More...

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int HTTPCli_readRawResponseBody(HTTPCli_Handle cli, char *body, int len)</code></td>
<td></td>
</tr>
<tr>
<td>Read the raw response message body from the HTTP server. More...</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void HTTPCli_setSecureParams(HTTPCli_SecureParams *sparams)</code></td>
<td></td>
</tr>
<tr>
<td>Set the secure communication parameters. More...</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void HTTPCli_setProxy(const struct sockaddr *addr)</code></td>
<td></td>
</tr>
<tr>
<td>Set the proxy address. More...</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

HTTP/1.1 Client interface.

This module provides a HTTP client implementation of IETF standard for HTTP/1.1 - RFC 2616 and TLS support to interact with HTTP/1.1 servers.

Features

- Supports GET, POST, PUT, HEAD, OPTIONS, DELETE methods, request and response headers/bodies, and redirection response handling.
- Using this module, connections can be opened/closed to HTTP servers. Each connection is an instance which can be configured with a set of repeated requests/responses parameters so that they can be used across multiple transactions.
- For supporting small memory devices, the module has been specially designed with split request/response APIs which enable sending requests and responses in smaller chunks, and static memory based APIs to eliminate the need for dynamic memory allocations.
- Supports security features which include TLS and communication through proxy.
- Supports two types of programming model: synchronous and asynchronous modes. In the default synchronous mode, an application uses APIs to make request to the server and blocks till the server responds. With asynchronous mode, a thread to handle the response is created which invokes callbacks registered with the module.
- Supports TI-RTOS NDK, SimpleLink WiFi and Linux networking stacks. At IP level supports both IPv4 and IPv6.
- Supports conditional compilation of the module to include/exclude some of the features mentioned above.

Limitations
- HTTP client on SimpleLink WiFi supports IPv4 only.
- HTTP client on TI-RTOS NDK does not support host name resolution. So the API `HTTPCli_initSockAddr()` does not resolve host name. The API can be used only if an IP string is passed instead of the host name.

### GET Example

```c
#include <ti/net/http/httpcli.h>

HTTPCli_Struct cli;

// Construct a static instance
HTTPCli_construct(&cli);

// Connect to the HTTP Server
HTTPCli_connect(&cli, &addr, 0, NULL);

// Set request fields
HTTPCli_Field fields[2] = {
    { HTTPCli_FIELD_NAME_HOST, "www.example.com" },
    { NULL, NULL }
};

HTTPCli_setRequestFields(&cli, fields);

// Set response field filters
char *respFields[2] = {
    HTTPCli_FIELD_NAME_CONTENT_LENGTH,
    NULL
};

HTTPCli_setResponseFields(&cli, respFields);

// Make HTTP 1.1 GET request
//
// Send request to the server:
//
// GET /index.html HTTP/1.1
// Host: www.example.com
```
HTTPCli_sendRequest(&cli, HTTPCli_METHOD_GET, "/index.html", false);

// Get the processed response status
//
// HTTP/1.1 200 OK
//
status = HTTPCli_getResponseStatus(&cli);

// Check the HTTP return status and process remaining response
if (status == 200) {
  do {
    // Filter the response headers and get the set response field
    //
    // ...
    // Content-type: text/xml; charset=utf-8\r\n    // Content-length: 34
    // ...
    ret = HTTPCli_getResponseField(&cli, buf, sizeof(buf), &moreFlag);
    if (ret == 0) {
      len = (int)strtoul(buf, NULL, 0);
    }
  }
  //loop till the end of the response fields
  while (ret != HTTPCli_FIELD_ID_END);

  // Read message based on content length field value
  while (len > 0) {
    len -= HTTPCli_readRawResponseBody(&cli, buf, sizeof(buf));
  }
}
// process buf data and save
}
}

// Close connection
HTTPCli_disconnect(&cli);

// Destroy the instance
HTTPCli_destruct(&cli);

POST Example

#include <ti/net/http/httpcli.h>

HTTPCli_Struct cli;

// Construct a static instance
HTTPCli_construct(&cli);

// Connect to the HTTP Server
HTTPCli_connect(&cli, &addr, 0, NULL);

// Set request fields
HTTPCli_Field fields[2] = {
    { HTTPCli_FIELD_NAME_HOST, "www.example.com" },
    { NULL, NULL }
};

HTTPCli_setRequestFields(&cli, fields);

// Set response field filters
char *respFields[2] = {
    HTTPCli_FIELD_NAME_CONTENT_LENGTH,
    NULL
};

HTTPCli_setResponseFields(&cli, respFields);

// Make HTTP 1.1 POST request
//
// Send request to the server:
// POST /index.html HTTP/1.1
// Host: www.example.com
HTTPCli_sendRequest(&cli, HTTPCli_METHOD_POST, 
    "index.html", true);

// Send additional fields
//
// Content-Length: <length>
// <blank line>
HTTPCli_sendField(&cli,
    HTTPCli_FIELD_NAME_CONTENT_LENGTH, len, true);

// Send request body
//
// <data>
HTTPCli_sendRequestBody(&cli, data, strlen(data));

// Get the processed response status
//
// HTTP/1.1 200 OK
//
status = HTTPCli_getResponseStatus(&cli);

// Check the HTTP return status and process remaining response
if (status == 200) {
    do {
        // Filter the response headers and get the set response field
        //
        // ...
        // Content-type: text/xml; charset=utf-8
        // Content-length: 34
        // ...
        ret = HTTPCli_getResponseField(&cli, buf, sizeof(buf), &moreFlag);
// process data in buf if field is content length
// Zero is the index of Content length in respFields array
if (ret == 0) {
    len = (int)strtoul(buf, NULL, 0);
}

// loop till the end of the response fields
} while (ret != HTTPCli_FIELD_ID_END);

// Read message based on content length field value
while (len > 0) {
    len -= HTTPCli_readRawResponseBody(&cli, buf, sizeof(buf));
    // process buf data and save
}

// Close connection
HTTPCli_disconnect(&cli);

// Destroy the instance
HTTPCli_destruct(&cli);
Macro Definition Documentation

#define HTTPCli_BUF_LEN 128
typedef int(* HTTPCli_ContentCallback) (void *cli, int status, char *body, int len, bool moreFlag)

HTTPCli callback function prototype for content handling.

Parameters

[in] cli Instance of the HTTP connection
[in] status Response status code
[in] body Data from the response body
[in] len Length of response body buffer
[in] moreFlag Set if more response data is available

Returns

1 to continue or 0 to stop further processing.

typedef void(* HTTPCli_Notify) (long skt, void *cli)

HTTPCli callback function prototype for asynchronous notify
Supported for 6LoWPAN stack only.

Parameters

[in] skt socket handle
[in] cli Instance of the HTTP connection

typedef void(* HTTPCli_RedirectCallback) (void *cli, int status, char *uri)

HTTPCli callback function prototype for redirection handling.

Parameters
typedef void(* HTTPCli_StatusCallback) (void *cli, int status)

HTTPCli callback function prototype for status handling.

Parameters

- [in] cli Instance of the HTTP connection
- [in] status Response status code
Function Documentation

int HTTPCli_connect ( HTTPCli_Struct * cli, const struct sockaddr * addr, int flags, const HTTPCli_Params * params )

Open a connection to a HTTP server.

Parameters
[ in ] cli Instance of a HTTP client
[ in ] addr IP address of the server
[ in ] flags Sets the type of HTTP instance (ex: HTTPCli_TYPE_TLS). Multiple types can be OR'ed together.
[ in ] params Per-instance config params, or NULL for default values

Returns
0 on success or error code on failure.

void HTTPCli_construct ( HTTPCli_Struct * cli )

Create a new instance object in the provided structure.

Parameters
[ out ] cli Instance of a HTTP client

HTTPCli_Handle HTTPCli_create ( )
Allocate and initialize a new instance object and return its handle.

**Returns**
handle of the HTTP client instance on success or NULL on failure.

```c
void HTTPCli_delete ( HTTPCli_Handle cli )
```

Destroy the HTTP client instance and free the previously allocated instance object.

**Parameters**
- `[in] cli` Instance of the HTTP client

```c
void HTTPCli_destruct ( HTTPCli_Struct * cli )
```

Destroy the HTTP client instance.

**Parameters**
- `[in] cli` Instance of the HTTP client

```c
void HTTPCli_disconnect ( HTTPCli_Struct * cli )
```

Disconnect from the HTTP server and destroy the HTTP client instance.

**Parameters**
- `[in] cli` Instance of the HTTP client

```c
int HTTPCli_getResponseField ( HTTPCli_Handle cli, char * value, int len,
```
bool * moreFlag
)

Process the response header from the HTTP server and return field.

Filters the response headers based on the array of fields (see HTTPCli_setResponseFields()).

Repeatedly call this function till HTTPCli_FIELD_ID_END is returned.

Parameters

[in] cli Instance of a HTTP client
[out] value Field value string.
[in] len Length of field value string
[out] moreFlag Flag set if the field value could not be completely read into value. A subsequent call to this function will read the remaining field value into value and will return HTTPCli_FIELD_ID_DUMMY.

Returns

On Success, the index of the field set in the HTTPCli_setResponseFields() or HTTPCli_FIELD_ID_END or HTTPCli_FIELD_ID_DUMMY, or error code on failure.

int HTTPCli_getResponseStatus ( HTTPCli_Handle cli )

Process the response header from the HTTP server and return status.

Remarks

Do not call in asynchronous mode. This function will return HTTPCli_EASYNCMODE.

Parameters

[in] cli Instance of a HTTP client
Returns
The status code from the server (1xx, 2xx, 3xx, 4xx, 5xx) on success or error code on failure.

```
int HTTPCli_initSockAddr ( struct sockaddr * addr, const char * uri, int flags )
```

Initialize the socket address structure for the given URI. Supported URI formats are: "http://www.example.com:8000", "https://www.example.com:8000", "https://www.example.com", "www.example.com:8000", "www.example.com". For cases where port is not provided, the default port number is set.

Parameters
- [out] `addr` Handle to the sockaddr structure
- [in] `uri` A null terminated URI string

Returns
0 on success or error code on failure.

```
int HTTPCli_readRawResponseBody ( HTTPCli_Handle cli, char * body, int len )
```

Read the raw response message body from the HTTP server.

Make a call to this function only after the call to `HTTPCli_getResponseStatus()` and `HTTPCli_getResponseField()`.

Repeatedly call this function till entire response message is read.
Read the parsed response body data from the HTTP server.

This function parses the response body if the content type is chunked transfer encoding or if the content length field is returned by the HTTP server.

Make a call to this function only after the call to `HTTPCli_getResponseStatus()` and `HTTPCli_getResponseField()`.

```c
int HTTPCli_readResponseBody (HTTPCli_Handle cli,
                            char * body,
                            int len,
                            bool * moreFlag)
```

Parameters
- [in] cli Instance of a HTTP client
- [out] body Response body buffer
- [in] len Length of response body buffer

Returns
The number of characters read on success or error code on failure

```c
int HTTPCli_sendField (HTTPCli_Handle cli,
```
Send a header field to the HTTP server.

This is a complementary function to HTTPCli_sendRequest() when more header fields are to be sent to the server.

**Parameters**
- `[in] cli` Instance of a HTTP client
- `[in] name` HTTP 1.1 request header field (ex: HTTPCli_FIELD_NAME_HOST)
- `[in] value` HTTP 1.1 request header field value
- `[in] lastFlag` Set this flag when sending the last header field

**Returns**
- 0 on success or error code on failure

```c
int HTTPCli_sendRequest ( HTTPCli_Handle cli,
                          const char * method,
                          const char * requestURI,
                          bool moreFlag )
```

Make a HTTP 1.1 request to the HTTP server.

Sends a HTTP 1.1 request-line and the header fields from the user set array (see HTTPCli_setRequestFields()) to the server.

Additionally, more fields apart from the user set array of header fields can be sent to the server. To send more fields, set the `moreFlag` when calling this function and then call HTTPCli_sendField() with more fields.

**Parameters**
[in] cli Instance of a HTTP client
[in] method HTTP 1.1 method (ex: HTTPCli_METHOD_GET)
[in] requestURI the path on the server to open and any CGI parameters
[in] moreFlag Set this flag when more fields will be sent to the server

Returns
0 on success or error code on failure

int HTTPCli_sendRequestBody (HTTPCli_Handle cli, const char * body, int len)

Send the request message body to the HTTP server.
Make a call to this function after HTTPCli_sendRequest() (always) and HTTPCli_sendField() (if applicable).

Parameters
[in] cli Instance of a HTTP client
[in] body Request body buffer
[in] len Length of the request body buffer

Returns
0 on success or error code on failure

void HTTPCli_setProxy (const struct sockaddr * addr)

Set the proxy address.

Parameters
[in] addr IP address of the proxy server
**HTTPCli_setRequestFields**

```c
HTTPCli_setRequestFields (HTTPCli_Handle cli, const HTTPCli_Field * fields)
```

Set an array of header fields to be sent for every HTTP request.

**Parameters**
- `[in] cli` Instance of a HTTP client
- `[in] fields` An array of HTTP request header fields terminated by an object with NULL fields, or NULL to get previously set array.

**Remarks**
The array should be persistent for the lifetime of HTTP instance.

**Returns**
- previously set array

---

**char** **HTTPCli_setResponseFields**

```c
char** HTTPCli_setResponseFields (HTTPCli_Handle cli, const char * fields[])
```

Set the header fields to filter the response headers.

**Parameters**
- `[in] cli` Instance of a HTTP client
- `[in] fields` An array of HTTP response header field strings terminated by a NULL, or NULL to get previously set array.

**Remarks**
The array should be persistent for the lifetime of HTTP instance.

**Returns**
- previously set array
void HTTPCli_setSecureParams (HTTPCli_SecureParams * sparams)

Set the secure communication parameters.

Parameters

[in] sparams pointer to secure communication params struct
HTTP Client
Library
1.00.01.04

HTTPClient_Field Struct
Reference

HTTPClient Request Header Field. More...

#include <httpcli.h>
# Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>const char *</td>
<td>name</td>
</tr>
<tr>
<td>const char *</td>
<td>value</td>
</tr>
</tbody>
</table>
Detailed Description

HTTPCli Request Header Field.
### Member Data Documentation

**const char* HTTPCli_Field::name**

Field name, ex: HTTPCli_FIELD_NAME_ACCEPT

**const char* HTTPCli_Field::value**

Field value, ex: "text/plain"

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/httpcli.h

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTPCli StatusHandler Struct Reference

HTTP Client

HTTPCli Response status code handlers. More...

#include <httpcli.h>
## Public Attributes

<table>
<thead>
<tr>
<th>HTTPCli_StatusCallback</th>
<th>handle1xx</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTPCli_StatusCallback</td>
<td>handle2xx</td>
</tr>
<tr>
<td>HTTPCli_StatusCallback</td>
<td>handle4xx</td>
</tr>
</tbody>
</table>
Detailed Description

HTTPCli Response status code handlers.
### Member Data Documentation

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>HTTPCli_StatusCallback</code></td>
<td><strong>HTTPCli_StatusHandler::handle1xx</strong></td>
</tr>
<tr>
<td></td>
<td>1xx status code callback</td>
</tr>
<tr>
<td><code>HTTPCli_StatusCallback</code></td>
<td><strong>HTTPCli_StatusHandler::handle2xx</strong></td>
</tr>
<tr>
<td></td>
<td>2xx status code callback</td>
</tr>
<tr>
<td><code>HTTPCli_StatusCallback</code></td>
<td><strong>HTTPCli_StatusHandler::handle4xx</strong></td>
</tr>
<tr>
<td></td>
<td>4xx/5xx status code callback</td>
</tr>
</tbody>
</table>

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

- `D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/httpcli.h`
HTTP Cli Content Handler type.

More...

#include <httpcli.h>
Public Attributes

<table>
<thead>
<tr>
<th>char *</th>
<th>contentType</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTPCli_ContentCallback</td>
<td>handle</td>
</tr>
</tbody>
</table>
Detailed Description

HTTPTCli Content Handler type.
Member Data Documentation

char* HTTPCli_ContentHandler::contentType

ex: application/json

HTTPCli_ContentCallback HTTPCli_ContentHandler::handle

Callback for content Type

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/httpcli.h

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTP Client

HTTPCli instance type. More...

#include <httpcli.h>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>char **</td>
<td><code>respFields</code></td>
</tr>
<tr>
<td>unsigned int</td>
<td><code>state</code></td>
</tr>
<tr>
<td>unsigned long</td>
<td><code>clen</code></td>
</tr>
<tr>
<td><strong>Ssock_Struct</strong></td>
<td><code>ssock</code></td>
</tr>
<tr>
<td>HTTPCli_Field *</td>
<td><code>fields</code></td>
</tr>
<tr>
<td>char</td>
<td><code>buf [HTTPCli_BUF_LEN]</code></td>
</tr>
<tr>
<td>unsigned int</td>
<td><code>buflen</code></td>
</tr>
<tr>
<td>char *</td>
<td><code>bufptr</code></td>
</tr>
<tr>
<td>HTTPCli_StatusHandler *</td>
<td><code>shandle</code></td>
</tr>
<tr>
<td>HTTPCli_ContentHandler *</td>
<td><code>chandle</code></td>
</tr>
<tr>
<td>HTTPCli_RedirectCallback</td>
<td><code>rhandle</code></td>
</tr>
<tr>
<td>unsigned int</td>
<td><code>stackSize</code></td>
</tr>
<tr>
<td>unsigned int</td>
<td><code>priority</code></td>
</tr>
</tbody>
</table>
Detailed Description

HTTPCli instance type.

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/httpcli.h

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTPClient instance parameters. More...

#include <httpcli.h>
## Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>HTTPCli_StatusHandler</code>*</td>
<td><code>shandle</code></td>
</tr>
<tr>
<td><code>HTTPCli_ContentHandler</code>*</td>
<td><code>chandle</code></td>
</tr>
<tr>
<td><code>HTTPCli_RedirectCallback</code></td>
<td><code>rhandle</code></td>
</tr>
<tr>
<td>unsigned int</td>
<td><code>stackSize</code></td>
</tr>
<tr>
<td>unsigned int</td>
<td><code>priority</code></td>
</tr>
<tr>
<td>int</td>
<td><code>timeout</code></td>
</tr>
</tbody>
</table>
Detailed Description

HTTPCli instance parameters.
Member Data Documentation

**unsigned int HTTPCli_Params::priority**

Async thread priority. 0 for default

**unsigned int HTTPCli_Params::stackSize**

Async thread stack size. 0 for default

**int HTTPCli_Params::timeout**

Timeout value (in seconds) for socket. Set 0 for default value

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/httpcli.h

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTP Client secure parameters for TLS. More...

#include <httpcli.h>
## Public Attributes

| char s |
Detailed Description

HTTPCli secure parameters for TLS.

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/httpcli.h

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
Here are the classes, structs, unions and interfaces with brief descriptions:

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTPCli_ContentHandler</td>
<td>HTTPCli Content Handler type</td>
</tr>
<tr>
<td>HTTPCli_Field</td>
<td>HTTPCli Request Header Field</td>
</tr>
<tr>
<td>HTTPCli_Params</td>
<td>HTTPCli instance parameters</td>
</tr>
<tr>
<td>HTTPCli_SecureParams</td>
<td>HTTPCli secure parameters for TLS</td>
</tr>
<tr>
<td>HTTPCli_StatusHandler</td>
<td>HTTPCli Response status code handlers</td>
</tr>
<tr>
<td>HTTPCli_Struct</td>
<td>HTTPCli instance type</td>
</tr>
<tr>
<td>Ssock_SecureFxns</td>
<td></td>
</tr>
<tr>
<td>Ssock_Struct</td>
<td></td>
</tr>
<tr>
<td>StatusMap</td>
<td></td>
</tr>
<tr>
<td>URLFile_Object</td>
<td></td>
</tr>
<tr>
<td>URLHandler_Setup</td>
<td></td>
</tr>
<tr>
<td>Main Page</td>
<td>Modules</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Index</td>
</tr>
</tbody>
</table>

**Ssock_SecureFxns**

Struct Reference
## Public Attributes

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Attribute Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ssock_EncryptFxn</td>
<td>encrypt</td>
</tr>
<tr>
<td>Ssock_DecryptFxn</td>
<td>decrypt</td>
</tr>
<tr>
<td>Ssock_SendFxn</td>
<td>send</td>
</tr>
<tr>
<td>Ssock_RecvFxn</td>
<td>recv</td>
</tr>
<tr>
<td>Ssock_DeleteFxn</td>
<td>del</td>
</tr>
<tr>
<td>int32_t</td>
<td>extraBytes</td>
</tr>
</tbody>
</table>

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

- **D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/ssock.h**
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Index</td>
<td>Class Members</td>
<td></td>
</tr>
</tbody>
</table>

### Ssock_Struct Struct Reference

**HTTP Client Library**

1.00.01.04

Public Attributes | List of all members
Public Attributes

```
int s

Ssock_SecureFxns sec

void * ctx
```

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/ssock.h

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
### StatusMap Struct Reference

| Class List | Class Index | Class Members |
|------------|-------------|---------------|------------|

Public Attributes | List of all members
Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>status</td>
</tr>
<tr>
<td>const char *</td>
<td>message</td>
</tr>
</tbody>
</table>

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/httpsend.c

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
URLFile_Object Struct

Reference
Public Attributes

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>length</td>
</tr>
<tr>
<td>char</td>
<td>prefix []</td>
</tr>
</tbody>
</table>

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/urlfile.c
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th><strong>Classes</strong></th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Index</td>
<td>Class Members</td>
<td></td>
</tr>
</tbody>
</table>

**HTTP Client Library**
1.00.01.04

**URLHandler_Setup**

Struct Reference

Public Attributes | List of all members
### Public Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>void *params</td>
<td></td>
</tr>
<tr>
<td>URLHandler_CreateFxnf</td>
<td></td>
</tr>
<tr>
<td>URLHandler_DeleteFxnf</td>
<td></td>
</tr>
<tr>
<td>URLHandler_ProcessFxnf</td>
<td></td>
</tr>
<tr>
<td>URLHandler_ScanFieldFxnf</td>
<td></td>
</tr>
</tbody>
</table>

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

- D:/CC3xxx/CC3100/CC3100 SDK/CC3100 1.1.0/sdk/netapps/http/client/urlhandler.h

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
# Class Index

<table>
<thead>
<tr>
<th>H</th>
<th>S</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>H</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTTPCli_Params</td>
<td>HTTPCli_SecureParams</td>
<td>Ssock_Secure</td>
</tr>
<tr>
<td>HTTPCli_ContentHandler</td>
<td>HTTPCli_StatusHandler</td>
<td>Ssock_Struct</td>
</tr>
<tr>
<td>HTTPCli_Field</td>
<td>HTTPCli_Struct</td>
<td></td>
</tr>
<tr>
<td>Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by <a href="http://www.doxygen.org">doxygen</a> 1.8.9.1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Here is a list of all documented class members with links to the class documentation for each member:

- contentType : `HTTPCli_ContentHandler`
- handle : `HTTPCli_ContentHandler`
- handle1xx : `HTTPCli_StatusHandler`
- handle2xx : `HTTPCli_StatusHandler`
- handle4xx : `HTTPCli_StatusHandler`
- name : `HTTPCli_Field`
- priority : `HTTPCli_Params`
- stackSize : `HTTPCli_Params`
- timeout : `HTTPCli_Params`
- value : `HTTPCli_Field`
- contentType : HTTPCli_ContentHandler
- handle : HTTPCli_ContentHandler
- handle1xx : HTTPCli_StatusHandler
- handle2xx : HTTPCli_StatusHandler
- handle4xx : HTTPCli_StatusHandler
- name : HTTPCli_Field
- priority : HTTPCli_Params
- stackSize : HTTPCli_Params
- timeout : HTTPCli_Params
- value : HTTPCli_Field
Here is a list of all documented files with brief descriptions:

<table>
<thead>
<tr>
<th>Category</th>
<th>File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>cc3xxx</td>
<td>cc3100</td>
</tr>
<tr>
<td></td>
<td>cc3100-sdk</td>
</tr>
<tr>
<td></td>
<td>cc3100-1.1.0</td>
</tr>
<tr>
<td></td>
<td>sdk</td>
</tr>
<tr>
<td></td>
<td>netapps</td>
</tr>
<tr>
<td></td>
<td>http</td>
</tr>
<tr>
<td></td>
<td>client</td>
</tr>
<tr>
<td></td>
<td>common.h</td>
</tr>
<tr>
<td></td>
<td>httpcli.h</td>
</tr>
<tr>
<td></td>
<td>httpstd.h</td>
</tr>
<tr>
<td></td>
<td>logging.h</td>
</tr>
<tr>
<td></td>
<td>network.h</td>
</tr>
<tr>
<td></td>
<td>ssock.h</td>
</tr>
<tr>
<td></td>
<td>urlfile.h</td>
</tr>
<tr>
<td></td>
<td>urlhandler.h</td>
</tr>
</tbody>
</table>

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>D:</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

D: Directory Reference
| directory | CC3xxx |

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
CC3xxx Directory Reference
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>D: CC3xxx</td>
<td>CC3100</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CC3100 Directory Reference**
| directory       | CC3100 SDK |

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
CC3100 SDK Directory Reference
Directories

| directory | CC3100 1.1.0 |

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>D: CC3xxx</td>
<td>CC3100</td>
<td>CC3100 SDK</td>
<td>CC3100 1.1.0</td>
</tr>
</tbody>
</table>

**CC3100 1.1.0 Directory Reference**
## Directories

| directory | sdk |

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by [doxygen](https://www.doxygen.org/) 1.8.9.1
HTTP Client Library
1.00.01.04

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>D: CC3xxx</td>
<td>CC3100</td>
<td>CC3100 SDK</td>
<td>CC3100 1.1.0</td>
</tr>
</tbody>
</table>

sdk Directory Reference
Directories

directory  netapps

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>D: CC3xxx</td>
<td>CC3100</td>
<td>CC3100 SDK</td>
<td>CC3100 1.1.0</td>
</tr>
</tbody>
</table>

**netapps Directory Reference**
| directory | http |

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
http Directory Reference
HTTP Client Library
1.00.01.04

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>D: CC3xxx</td>
<td>CC3100</td>
<td>CC3100 SDK</td>
<td>CC3100 1.1.0</td>
</tr>
</tbody>
</table>

client Directory Reference
### Files

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>common.h</td>
<td></td>
</tr>
<tr>
<td>httpcli.c</td>
<td></td>
</tr>
<tr>
<td>httpcli.h</td>
<td>[code]</td>
</tr>
<tr>
<td>httpsend.c</td>
<td></td>
</tr>
<tr>
<td>httpsend.h</td>
<td>[code]</td>
</tr>
<tr>
<td>httpstd.h</td>
<td>[code]</td>
</tr>
<tr>
<td>httpstr.c</td>
<td></td>
</tr>
<tr>
<td>logging.h</td>
<td>[code]</td>
</tr>
<tr>
<td>network.h</td>
<td>[code]</td>
</tr>
<tr>
<td>ssnull.c</td>
<td></td>
</tr>
<tr>
<td>ssock.c</td>
<td></td>
</tr>
<tr>
<td>ssock.h</td>
<td>[code]</td>
</tr>
<tr>
<td>urlfile.c</td>
<td></td>
</tr>
<tr>
<td>urlfile.h</td>
<td>[code]</td>
</tr>
<tr>
<td>urlhandler.h</td>
<td>[code]</td>
</tr>
</tbody>
</table>
common.h

1  /*
2  * Copyright (c) 2014, Texas Instruments Incorporated
3  * All rights reserved.
4  *
5  * Redistribution and use in source and binary forms, with or without
6  * modification, are permitted provided that the following conditions
7  * are met:
8  *
9  * Redistributions of source code must retain the above copyright
10  * notice, this list of conditions and the following disclaimer.
11  *
12  * Redistributions in binary form must reproduce the above copyright
13  * notice, this list of conditions and the following disclaimer in the
14  * documentation and/or other materials provided with the distribution.
15  *
Neither the name of Texas Instruments Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 THE COPYRIGHT OWNER OR CONTRIBUTORS 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.

*/

#ifndef _COMMON_H_
#define _COMMON_H_

#ifdef __linux__
#include <pthread.h>
#endif

#endif
#include <pthread.h>
```c
#include <stdio.h>
#include <errno.h>
#include <string.h>
#include <stdlib.h>
#include <assert.h>
#include <stdarg.h>

#elif defined(__SL__) || defined(__NDK__) || defined(__SLP__) /*__linux__*/
    // #include <ti/sysbios/knl/Task.h>
    // #include <xdc/runtime/System.h>
    // #include <stdio.h>

#include <string.h>
#include <stdlib.h>
#include <assert.h>
#include <stdarg.h>
extern int vsnprintf(char *string, size_t _n, const char *format, va_list _ap);
#else /*__linux__*/
    #error Unsupported configuration specified
#endif /*__linux__*/

/*
 * xassert
 *
*/
static inline void xassert(int expr)
{
    //assert(expr);
    if(!expr) { while(1); }
}

/*
 * xfree
 *
*/
static inline void xfree(void *ptr)
{

free(ptr);

/**
 * ======== xmalloc ========
 */

static inline void *xmalloc(size_t size)
{
    return (malloc(size));
}

/**
 * ======== xvsnprintf ========
 */

static inline int xvsnprintf(char *s, size_t n, const char *fmt, va_list arg)
{
    #if defined(__SL__) || defined(__NDK__) || defined(__SLP__)
        //return (System_vsnprintf(s, n, fmt, arg));
    
    return (vsnprintf(s, n, fmt, arg));
    
    #else /* __SL__ || __NDK__ || __SLP__ */
        return (vsnprintf(s, n, fmt, arg));
    
    #endif /* __SL__ || __NDK__ */
}

#endif /* _COMMON_H_ */
/*
 * Copyright (c) 2014, Texas Instruments Incorporated
 * All rights reserved.
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 */
Neither the name of Texas Instruments Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 THE COPYRIGHT OWNER OR CONTRIBUTORS 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.

*/

 ifndef _HTTPCli_H_
define _HTTPCli_H_

 ifndef __cplusplus
 extern "C" {

 endif /* __cplusplus */
```
#include <stdbool.h>
#include <http/client/httpstd.h>
#include <http/client/network.h>
#include <http/client/ssock.h>

#ifndef HTTPCli_BUF_LEN
#define HTTPCli_BUF_LEN 128
#endif

#ifndef HTTPCli_CERT_NAME_LEN
#define HTTPCli_CERT_NAME_LEN 16
#endif

/* HTTP Client Error Codes */
#define HTTPCli_ESOCKETFAIL (-101)
#define HTTPCli_ECONNECTFAIL (-102)
#define HTTPCli_ESENDFAIL (-103)
#define HTTPCli_ERECVFAIL (-104)
#define HTTP Cli_ETLSFAIL (-105)
#define HTTPCli_EHOSTNAME (-106)
#define HTTPCli_ESENDBUFSMALL (-107)
#define HTTPCli_ERECVBUFSMALL (-108)
#define HTTPCli_EASYNCMODE (-109)
#define HTTPCli_ETHREADFAIL (-110)
#define HTTPCli_EPROXYTUNNELFAIL (-111)
```
#define HTTPCli_ERESPONSEINVALID (-112)

#define HTTPCli_ECONTENTLENENLARGE (-114)

#define HTTPCli_EREDIRECTURILONG (-115)

#define HTTPCli_ECONTENTTYPELONG (-116)

#define HTTPCli_ENOCONTENTCALLBACK (-117)

#define HTTPCli_ENOTCHUNKDATA (-118)

#define HTTPCli_EINPROGRESS (-119)

#define HTTPCli_EINTERNALBUFSMALL (-120)

#define HTTPCli_ESETNOTIFYFAIL (-121)

#define HTTPCli_EURILENLONG (-120)

/* HTTP methods */
#define HTTPCli_METHOD_GET "GET"
#define HTTPCli_METHOD_POST "POST"
#define HTTPCli_METHOD_HEAD "HEAD"
#define HTTPCli_METHOD_OPTIONS "OPTIONS"
#define HTTPCli_METHOD_PUT "PUT"
#define HTTPCli_METHOD_DELETE "DELETE"
#define HTTPCli_METHOD_CONNECT "CONNECT"

/* HTTP Request Field Name */
#define HTTPCli_FIELD_NAME_ACCEPT "Accept"
#define HTTPCli_FIELD_NAME_ACCEPT_CHARSET "Accept-Charset"
#define HTTPCli_FIELD_NAME_ACCEPT_ENCODING "Accept-Encoding"
#define HTTPCli_FIELD_NAME_ACCEPT_LANGUAGE "Accept-Language"
```c
#define HTTPCli_FIELD_NAME_ACCEPT_RANGES "Accept-Ranges"
#define HTTPCli_FIELD_NAME_AGE "Age"
#define HTTPCli_FIELD_NAME_ALLOW "Allow"
#define HTTPCli_FIELD_NAME_AUTHORIZATION "Authorization"
#define HTTPCli_FIELD_NAME_CACHE_CONTROL "Cache-Control"
#define HTTPCli_FIELD_NAME_CONNECTION "Connection"
#define HTTPCli_FIELD_NAME_CONTENT_ENCODING "Content-Encoding"
#define HTTPCli_FIELD_NAME_CONTENT_LANGUAGE "Content-Language"
#define HTTPCli_FIELD_NAME_CONTENT_LENGTH "Content-Length"
#define HTTPCli_FIELD_NAME_CONTENT_LOCATION "Content-Location"
#define HTTPCli_FIELD_NAME_CONTENT_MD5 "Content-MD5"
#define HTTPCli_FIELD_NAME_CONTENT_RANGE "Content-Range"
#define HTTPCli_FIELD_NAME_COOKIE "Cookie"
#define HTTPCli_FIELD_NAME_DATE "Date"
#define HTTPCli_FIELD_NAME_ETAG "ETag"
#define HTTPCli_FIELD_NAME_EXPECT "Expect"
#define HTTPCli_FIELD_NAME_EXPIRES "Expires"
#define HTTPCli_FIELD_NAME_FROM "From"
#define HTTPCli_FIELD_NAME_HOST "Host"
#define HTTPCli_FIELD_NAME_IF_MATCH "If-Match"
#define HTTPCli_FIELD_NAME_IF_MODIFIED_SINCE "If-Modified-Since"
#define HTTPCli_FIELD_NAME_IF_NONE_MATCH "If-None-Match"
```
"If-None-Match"

#define HTTPCli_FIELD_NAME_IF_RANGE "If-Range"

#define HTTPCli_FIELD_NAME_IF_UNMODIFIED_SINCE "If-Unmodified-Since"

#define HTTPCli_FIELD_NAME_LAST_MODIFIED "Last-Modified"

#define HTTPCli_FIELD_NAME_LOCATION "Location"

#define HTTPCli_FIELD_NAME_MAX_FORWARDS "Max-Forwards"

#define HTTPCli_FIELD_NAME_ORIGIN "Origin"

#define HTTPCli_FIELD_NAME_PRAGMA "Pragma"

#define HTTPCli_FIELD_NAME_PROXY_AUTHENTICATE "Proxy-Authenticate"

#define HTTPCli_FIELD_NAME_PROXY_AUTHORIZATION "Proxy-Authentication"

#define HTTPCli_FIELD_NAME_RANGE "Range"

#define HTTPCli_FIELD_NAME_REFERER "Referer"

#define HTTPCli_FIELD_NAME_RETRY_AFTER "Retry-After"

#define HTTPCli_FIELD_NAME_SERVER "Server"

#define HTTPCli_FIELD_NAME_TE "TE"

#define HTTPCli_FIELD_NAME_TRAILER "Trailer"

#define HTTPCli_FIELD_NAME_TRANSFER_ENCODING "Transfer-Encoding"

#define HTTPCli_FIELD_NAME_UPGRADE "Upgrade"

#define HTTPCli_FIELD_NAME_USER_AGENT "User-Agent"

#define HTTPCli_FIELD_NAME_VARY "Vary"

#define HTTPCli_FIELD_NAME_VIA "Via"
#define HTTPCli_FIELD_NAME_WWW_AUTHENTICATE "WWW-Authenticate"
#define HTTPCli_FIELD_NAME_WARNING "Warning"
#define HTTPCli_FIELD_NAME_X_FORWARDED_FOR "X-Forwarded-For"

/* HTTP client instance configuration */
#define HTTPCli_TYPE_TLS (0x02)
#define HTTPCli_TYPE_IPV6 (0x04)

/* HTTP client getResponseField() special return codes */
#define HTTPCli_FIELD_ID_DUMMY (-11)
#define HTTPCli_FIELD_ID_END (-12)

typedef struct HTTPCli_Field {
    const char *name;
    const char *value;
} HTTPCli_Field;

typedef void (*HTTPCli_StatusCallback)(void *cli, int status);

typedef struct HTTPCli_StatusHandler {
    HTTPCli_StatusCallback handle1xx;
    HTTPCli_StatusCallback handle2xx;
    HTTPCli_StatusCallback handle4xx;
} HTTPCli_StatusHandler;

typedef int (*HTTPCli_ContentCallback)(void *cli, int status, char *body, int len, bool moreFlag);

typedef struct HTTPCli_ContentHandler {
    char *contentType;
    HTTPCli_ContentCallback handle;
} HTTPCli_ContentHandler;
typedef void (*HTTPCli_RedirectCallback)(void *cli, int status, char *uri);

typedef void (*HTTPCli_Notify)(long skt, void *cli);

typedef struct HTTPCli_Struct {
    char **respFields;
    unsigned int state;
    unsigned long clen;
    Ssock_Struct ssock;
    HTTPCli_Field *fields;
    char buf[HTTPCli_BUF_LEN];
    unsigned int buflen;
    char *bufptr;
}

#define HTTPCli_LIBTYPE_MIN
HTTPCli_StatusHandler *shandle;
HTTPCli_ContentHandler *chandle;
HTTPCli_RedirectCallback rhandle;
#endif
#endif /* __linux__ */
#endif /* HTTPCli_LIBTYPE_MIN */
#endif HTTPCli_Struct;

typedef struct HTTPCli_Params {
    HTTPCli_StatusHandler *shandle;
    HTTPCli_ContentHandler *chandle;
    HTTPCli_RedirectCallback rhandle;
} HTTPCli_Params;

#define HTTPCli_LIBTYPE_MIN
HTTPCli_StatusHandler *shandle;
HTTPCli_ContentHandler *chandle;
HTTPCli_RedirectCallback rhandle;
#endif
#endif /* __linux__ */
#endif /* HTTPCli_LIBTYPE_MIN */
#endif HTTPCli_Params;
#ifndef __linux__
  unsigned int stackSize;
  unsigned int priority;
#endif /* __linux__ */
#endif /* HTTPCli_LIBTYPE_MIN */

#ifdef __SLP__
  HTTPCli_Notify rnotify;
  HTTPCli_Notify wnotify;
  HTTPCli_Notify enotify;
#endif /* __SLP__ */

int timeout;
}

HTTPCli_Params;

typedef struct HTTPCli_SecureParams {
  #ifdef __CYASSL__
  CYASSL_CTX *ctx; /* CYASSL context */
  #elif defined(__SL__) /* __CYASSL__ */
  SlSockSecureMethod method;
  SlSockSecureMask mask;
  char
cafile[HTTPCli_CERT_NAME_LEN];
  char
privkey[HTTPCli_CERT_NAME_LEN];
  char
cert[HTTPCli_CERT_NAME_LEN];
  char
dhkey[HTTPCli_CERT_NAME_LEN];
  #else /* __CYASSL__ */
  char s;
  #endif /* __CYASSL__ */
} HTTPCli_SecureParams;

typedef HTTPCli_Struct* HTTPCli_Handle;
extern int HTTPCli_initSockAddr(struct sockaddr *addr, const char *uri, int flags);

extern void HTTPCli_construct(HTTPCli_Struct *cli);

extern HTTPCli_Handle HTTPCli_create();

extern int HTTPCli_connect(HTTPCli_Struct *cli, const struct sockaddr *addr, int flags, const HTTPCli_Params *params);

extern void HTTPCli_delete(HTTPCli_Handle cli);

extern void HTTPCli_destruct(HTTPCli_Struct *cli);

extern void HTTPCli_disconnect(HTTPCli_Struct *cli);

extern HTTPCli_Field *HTTPCli_setRequestFields(HTTPCli_Handle cli, const HTTPCli_Field *fields);

extern char **HTTPCli_setResponseFields(HTTPCli_Handle cli, const char *fields[]);

extern int HTTPCli_sendRequest(HTTPCli_Handle cli, const char *method, const char *requestURI, bool
extern int HTTPCli_sendField(HTTPCli_Handle cli, const char *name, const char *value, bool lastFlag);

extern int HTTPCli_sendRequestBody(HTTPCli_Handle cli, const char *body, int len);

extern int HTTPCli_getResponseStatus(HTTPCli_Handle cli);

extern int HTTPCli_getResponseField(HTTPCli_Handle cli, char *value, int len, bool *moreFlag);

extern int HTTPCli_readResponseBody(HTTPCli_Handle cli, char *body, int len, bool *moreFlag);

extern int HTTP Cli_readRawResponseBody(HTTPCli_Handle cli, char *body, int len);

extern void HTTPCli_setSecureParams(HTTPCli_SecureParams *sparams);

extern void HTTPCli_setProxy(const struct sockaddr *addr);

#ifndef __cplusplus
}
#endif
#endif /* __cplusplus */
#endif /* __HTTPCli_H__ */
/*
 * Copyright (c) 2014, Texas Instruments Incorporated
 * All rights reserved.
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 */
* * Neither the name of Texas Instruments Incorporated nor the names of
its contributors may be used to endorse or promote products derived
from this software without specific prior written permission.

* *

* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 THE COPYRIGHT OWNER OR
* CONTRIBUTORS 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.

*/

#ifndef _HTTPSTD_H_
#define _HTTPSTD_H_

#ifdef __cplusplus
extern "C" {
#endif /* __cplusplus */

#define _HTTPSTD_H_

#ifdef __cplusplus
extern "C" {
#endif /* __cplusplus */
/* HTTP Status Codes */

#define HTTP_CONTINUE 100
#define HTTP_SWITCHING_PROTOCOLS 101
#define HTTP_OK 200
#define HTTP_CREATED 201
#define HTTP_ACCEPTED 202
#define HTTP_NON_AUTHORITATIVE_INFORMATION 203
#define HTTP_NO_CONTENT 204
#define HTTP_RESET_CONTENT 205
#define HTTP_PARTIAL_CONTENT 206
#define HTTP_MULTIPLE_CHOICES 300
#define HTTP_MOVED_PERMANENTLY 301
#define HTTP_FOUND 302
#define HTTP_SEE_OTHER 303
#define HTTP_NOT_MODIFIED 304
#define HTTP_USE_PROXY 305
#define HTTP_TEMPORARY_REDIRECT 307
#define HTTP_BAD_REQUEST 400
#define HTTP_UNAUTHORIZED 401
<table>
<thead>
<tr>
<th>Line</th>
<th>Definition</th>
<th>Status Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>58</td>
<td><code>#define HTTP_PAYMENT_REQUIRED</code></td>
<td>402</td>
</tr>
<tr>
<td>59</td>
<td><code>#define HTTP_FORBIDDEN</code></td>
<td>403</td>
</tr>
<tr>
<td>60</td>
<td><code>#define HTTP_NOT_FOUND</code></td>
<td>404</td>
</tr>
<tr>
<td>61</td>
<td><code>#define HTTP_METHOD_NOT_ALLOWED</code></td>
<td>405</td>
</tr>
<tr>
<td>62</td>
<td><code>#define HTTP_NOT_ACCEPTABLE</code></td>
<td>406</td>
</tr>
<tr>
<td>63</td>
<td><code>#define HTTP_PROXY_AUTHENTICATION_REQUIRED</code></td>
<td>407</td>
</tr>
<tr>
<td>64</td>
<td><code>#define HTTP_REQUEST_TIMEOUT</code></td>
<td>408</td>
</tr>
<tr>
<td>65</td>
<td><code>#define HTTP_CONFLICT</code></td>
<td>409</td>
</tr>
<tr>
<td>66</td>
<td><code>#define HTTP_GONE</code></td>
<td>410</td>
</tr>
<tr>
<td>67</td>
<td><code>#define HTTP_LENGTH_REQUIRED</code></td>
<td>411</td>
</tr>
<tr>
<td>68</td>
<td><code>#define HTTP_PRECONDITION_FAILED</code></td>
<td>412</td>
</tr>
<tr>
<td>69</td>
<td><code>#define HTTP_REQUEST_ENTITY_TOO_LARGE</code></td>
<td>413</td>
</tr>
<tr>
<td>70</td>
<td><code>#define HTTP_REQUEST_URI_TOO_LONG</code></td>
<td>414</td>
</tr>
<tr>
<td>71</td>
<td><code>#define HTTP_UNSUPPORTED_MEDIA_TYPE</code></td>
<td>415</td>
</tr>
<tr>
<td>72</td>
<td><code>#define HTTP_REQUESTED_RANGE_NOT_SATISFIAIBLE</code></td>
<td>416</td>
</tr>
<tr>
<td>73</td>
<td><code>#define HTTP_EXPECTATION_FAILED</code></td>
<td>417</td>
</tr>
<tr>
<td>74</td>
<td><code>#define HTTP_INTERNAL_SERVER_ERROR</code></td>
<td>500</td>
</tr>
<tr>
<td>75</td>
<td><code>#define HTTP_NOT_IMPLEMENTED</code></td>
<td>501</td>
</tr>
<tr>
<td>76</td>
<td><code>#define HTTP_BAD_GATEWAY</code></td>
<td></td>
</tr>
</tbody>
</table>
#define HTTP_SERVICE_UNAVAILABLE

#define HTTP_GATEWAY_TIMEOUT

#define HTTP_HTTP_VERSION_NOT_SUPPORTED

/* Standard ports */
#define HTTP_PORT 80
#define HTTPS_PORT 443

static inline int is1xx(int status)
{
    return ((status >= 100) && (status < 200));
}

static inline int is2xx(int status)
{
    return ((status >= 200) && (status < 300));
}

static inline int is3xx(int status)
{
    return ((status >= 300) && (status < 400));
}

static inline int is4xx(int status)
{
    return ((status >= 400) && (status < 500));
}

static inline int is5xx(int status)
{  
    return ((status >= 500) && (status < 600));
}
#endif /* __cplusplus */
#endif /* _HTTPSTD_H_ */

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
#ifndef _LOGGING_H_
#define _LOGGING_H_

/* This define must precede inclusion of any xdc header files */
#define Registry_CURDESC
    ti_net_http_HTTPSrv_desc

#include <xdc/std.h>
#include <xdc/runtime/Assert.h>
#include <xdc/runtime/Diags.h>
#include <xdc/runtime/Log.h>
#include <xdc/runtime/Registry.h>

extern Registry_Desc
    ti_net_http_HTTPSrv_desc;

#endif
network.h

/*
 * Copyright (c) 2014, Texas Instruments Incorporated
 * All rights reserved.
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * * Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * */
* * Neither the name of Texas Instruments Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

* * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 THE COPYRIGHT OWNER OR CONTRIBUTORS 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.

*/

#ifndef _TI_NET_NETWORK_H_
#define _TI_NET_NETWORK_H_

#ifdef __linux__
#include <sys/types.h>
#include <sys/socket.h>

#include <sys/types.h>
#include <sys/socket.h>

#ifdef __linux__
#include <sys/types.h>
#include <sys/socket.h>

#include <sys/time.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <netinet/in.h>

elif defined(__SL__) /* __linux__ */
#include <simplelink.h>

ifndef __ssize_t_defined
#define __ssize_t_defined
typedef long int ssize_t;
endif /* __ssize_t_defined */

elif defined(__SLP__)
#include <socket.h>

ifndef __ssize_t_defined
#define __ssize_t_defined
typedef long int ssize_t;
endif /* __ssize_t_defined */

elif defined(__NDK__) /* __linux__ */
#include <stdint.h>
#include <sys/socket.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define INET6_ADDRSTRLEN 46

else /* __linux__ */
#error No or unrecognized network configuration specified
#ifndef __linux__ */
#endif
#if defined(__CYASSL__) && (defined (__linux__) || defined (__NDK__))
#include <cyassl/ssl.h>
#endif /* __CYASSL__ */
#endif /* _TI_NET_NETWORK_H_ */
ssock.h

1 /*
2  * Copyright (c) 2014, Texas Instruments Incorporated
3  * All rights reserved.
4  *
5  * Redistribution and use in source and binary forms, with or without
6  * modification, are permitted provided that the following conditions
7  * are met:
8  *
9  * * Redistributions of source code must retain the above copyright
10  * notice, this list of conditions and the following disclaimer.
11  *
12  * * Redistributions in binary form must reproduce the above copyright
13  * notice, this list of conditions and the following disclaimer in the
14  * documentation and/or other materials provided with the distribution.
15  *
Neither the name of Texas Instruments Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 THE COPYRIGHT OWNER OR CONTRIBUTORS 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.

*/

#ifndef _SSOCK_H_
#define _SSOCK_H_

#include <stdint.h>
#include <http/client/network.h>

#define Ssock_TIMEOUT (-11)
typedef ssize_t (*Ssock_EncryptFxn)(const void * ctx, uint8_t * oct,
                                const uint8_t * ipt, size_t len);

typedef ssize_t (*Ssock_SendFxn)(const void * ctx, int s, const void * buf,
                                size_t len, int flags);

typedef ssize_t (*Ssock_DecryptFxn)(const void * ctx, uint8_t * opt,
                                     const uint8_t * ict, size_t len);

typedef ssize_t (*Ssock_RecvFxn)(void * ctx, int s, void * buf,
                                  size_t len, int flags);

typedef void (*Ssock_DeleteFxn)(void * ctx);

typedef struct Ssock_SecureFxns {
    Ssock_EncryptFxn encrypt;
    Ssock_DecryptFxn decrypt;
    Ssock_SendFxnn send;
    Ssock_RecvFxnn recv;
    Ssock_DeleteFxnn del;
    int32_t extraBytes;
} Ssock_SecureFxns;

typedef struct Ssock_Struct {
    int s;
    Ssock_SecureFxns sec;
    void * ctx;
} Ssock_Struct;

typedef struct Ssock_Struct * Ssock_Handle;

extern void Ssock_construct(Ssock_Struct *
ssockP, int s);

extern Ssock_Handle Ssock_create(int s);

extern void Ssock_delete(Ssock_Handle * ss);

extern void Ssock_destruct(Ssock_Struct * ssockP);

extern int Ssock_getSocket(Ssock_Handle ss);

extern ssize_t Ssock_recv(Ssock_Handle ss, void * buf, size_t len, int flags);

extern ssize_t Ssock_recvall(Ssock_Handle ssock, void * buf, size_t len,
                                  int flags);

extern ssize_t Ssock_send(Ssock_Handle ss, const void * buf, size_t len,
                          int flags);

extern int Ssock_startSecure(Ssock_Handle ss, Ssock_SecureFxns * sec,
                              void * ctx);

#endif /* _SSOCK_H */
urlfile.h

1  /*
2  * Copyright (c) 2014, Texas Instruments Incorporated
3  * All rights reserved.
4  *
5  * Redistribution and use in source and binary forms, with or without
6  * modification, are permitted provided that the following conditions
7  * are met:
8  *
9  * Redistributions of source code must retain the above copyright
10  * notice, this list of conditions and the following disclaimer.
11  *
12  * Redistributions in binary form must reproduce the above copyright
13  * notice, this list of conditions and the following disclaimer in the
14  * documentation and/or other materials provided with the distribution.
15  */
* * Neither the name of Texas Instruments Incorporated nor the names of
its contributors may be used to endorse or promote products derived
from this software without specific prior written permission.

* * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 THE COPYRIGHT OWNER OR
CONTRIBUTORS 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.

*/

#ifndef _URLFILE_H_
#define _URLFILE_H_

#include <ti/net/http/urlhandler.h>
#include <ti/net/http/ssock.h>

typedef struct URLFile_Object *
URLFile_Handle;

extern URLHandler_Handle URLFile_create(void * params);
extern void URLFile_delete(URLHandler_Handle * h);
extern int URLFile_process(URLHandler_Handle u, int method, const char * url,
const char * urlArgs, int contentLength, Ssock_Handle s);

#define
urlhandler.h

1 /*
2 * Copyright (c) 2014, Texas Instruments Incorporated
3 * All rights reserved.
4 *
5 * Redistribution and use in source and binary forms, with or without
6 * modification, are permitted provided that the following conditions
7 * are met:
8 *
9 *   * Redistributions of source code must retain the above copyright
10 *     notice, this list of conditions and the following disclaimer.
11 *
12 *   * Redistributions in binary form must reproduce the above copyright
13 *     notice, this list of conditions and the following disclaimer in the
14 *     documentation and/or other materials provided with the distribution.
15 */
Neither the name of Texas Instruments Incorporated nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "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 THE COPYRIGHT OWNER OR CONTRIBUTORS 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.

/*
#ifndef _URLHANDLER_H_
#define _URLHANDLER_H_

#include <stddef.h>

#include "ssock.h"

#include "stdafx.h"
#define URLHandler_GET 1
#define URLHandler_POST 2
#define URLHandler_PUT 3
#define URLHandler_DELETE 4

#define URLHandler_ENOTHANDLED 0
#define URLHandler_EHANDLED 1
#define URLHandler_EERRORHANDLED 2

typedef struct URLHandler_Object * URLHandler_Handle;

typedef URLHandler_Handle (*URLHandler_CreateFxn)(void * arg);
typedef int (*URLHandler_ProcessFxn)(URLHandler_Handle u, int method, const char * url, const char * urlArgs, int contentLength, Ssock_Handle s);
typedef void (*URLHandler_ScanFieldFxn)(URLHandler_Handle u, int method, const char * url, const char * field);
typedef void (*URLHandler_DeleteFxn)(URLHandler_Handle * u);

typedef struct URLHandler_Setup {
    void * params;
    URLHandler_CreateFxn create;
    URLHandler_DeleteFxn delete;
    URLHandler_ProcessFxn process;
    URLHandler_ScanFieldFxn scan;
} URLHandler_Setup;
HTTP Client Library
1.00.01.04

HTTPCli_Field Member List

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

- name HTTPCli_Field
- value HTTPCli_Field

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTP Client Library
1.00.01.04

HTTPCli_StatusHandler Member List

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

handle1xx HTTPCli_StatusHandler
handle2xx HTTPCli_StatusHandler
handle4xx HTTPCli_StatusHandler

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTP Client Library
1.00.01.04

HTTPCli_ContentHandler Member List

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

<table>
<thead>
<tr>
<th>contentType</th>
<th>HTTPCli_ContentHandler</th>
</tr>
</thead>
<tbody>
<tr>
<td>handle</td>
<td>HTTPCli_ContentHandler</td>
</tr>
</tbody>
</table>

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTP Client Library
1.00.01.04

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Index</td>
<td>Class Members</td>
<td></td>
</tr>
</tbody>
</table>

**HTTPCli_Struct Member List**

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

- **buf** (defined in **HTTPCli_Struct**)  
- ** buflen** (defined in **HTTPCli_Struct**)  
- ** bufptr** (defined in **HTTPCli_Struct**)  
- **chandle** (defined in **HTTPCli_Struct**)  
- **clen** (defined in **HTTPCli_Struct**)  
- **fields** (defined in **HTTPCli_Struct**)  
- **priority** (defined in **HTTPCli_Struct**)  
- **respFields** (defined in **HTTPCli_Struct**)  
- **rhandle** (defined in **HTTPCli_Struct**)  
- **shandle** (defined in **HTTPCli_Struct**)  
- **ssock** (defined in **HTTPCli_Struct**)  
- **stackSize** (defined in **HTTPCli_Struct**)  
- **state** (defined in **HTTPCli_Struct**)  

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by [doxygen](http://www.stackoverflow.com) 1.8.9.1
HTTP Client Library
1.00.01.04

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Index</td>
<td>Class Members</td>
<td></td>
</tr>
</tbody>
</table>

**HTTPCli_Params Member List**

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

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>chandle</td>
<td>(defined in <code>HTTPCli_Params</code>)</td>
<td><code>HTTPCli_Params</code></td>
</tr>
<tr>
<td>priority</td>
<td></td>
<td><code>HTTPCli_Params</code></td>
</tr>
<tr>
<td>rhandle</td>
<td>(defined in <code>HTTPCli_Params</code>)</td>
<td><code>HTTPCli_Params</code></td>
</tr>
<tr>
<td>shandle</td>
<td>(defined in <code>HTTPCli_Params</code>)</td>
<td><code>HTTPCli_Params</code></td>
</tr>
<tr>
<td>stackSize</td>
<td></td>
<td><code>HTTPCli_Params</code></td>
</tr>
<tr>
<td>timeout</td>
<td></td>
<td><code>HTTPCli_Params</code></td>
</tr>
</tbody>
</table>

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by doxygen 1.8.9.1
HTTPCli_SecureParams Member List

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

s (defined in HTTPCli_SecureParams) HTTPCli_SecureParams
Ssock_SecureFxns Member List

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Definition</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>decrypt</td>
<td>(defined in Ssock_SecureFxns)</td>
<td>Ssock_SecureFxns</td>
</tr>
<tr>
<td>del</td>
<td>(defined in Ssock_SecureFxns)</td>
<td>Ssock_SecureFxns</td>
</tr>
<tr>
<td>encrypt</td>
<td>(defined in Ssock_SecureFxns)</td>
<td>Ssock_SecureFxns</td>
</tr>
<tr>
<td>extraBytes</td>
<td>(defined in Ssock_SecureFxns)</td>
<td>Ssock_SecureFxns</td>
</tr>
<tr>
<td>recv</td>
<td>(defined in Ssock_SecureFxns)</td>
<td>Ssock_SecureFxns</td>
</tr>
<tr>
<td>send</td>
<td>(defined in Ssock_SecureFxns)</td>
<td>Ssock_SecureFxns</td>
</tr>
</tbody>
</table>

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by [doxygen](http://www.doxygen.org) 1.8.9.1
Ssock_Struct Member List

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

<table>
<thead>
<tr>
<th>Member</th>
<th>Definition</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctx</td>
<td>(defined in Ssock_Struct)</td>
<td>Ssock_Struct</td>
</tr>
<tr>
<td>s</td>
<td>(defined in Ssock_Struct)</td>
<td>Ssock_Struct</td>
</tr>
<tr>
<td>sec</td>
<td>(defined in Ssock_Struct)</td>
<td>Ssock_Struct</td>
</tr>
</tbody>
</table>

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by [doxygen](http://www.stackexchange.com) 1.8.9.1
This is the complete list of members for StatusMap, including all inherited members.

<table>
<thead>
<tr>
<th>member</th>
<th>defined in</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>StatusMap</td>
</tr>
<tr>
<td>status</td>
<td>StatusMap</td>
</tr>
</tbody>
</table>
URLFile_Object Member List

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

- `length` (defined in `URLFile_Object`)
- `prefix` (defined in `URLFile_Object`)

Generated on Tue Jan 13 2015 14:42:03 for HTTP Client Library by `doxygen` 1.8.9.1
This is the complete list of members for `URLHandler_Setup`, including all inherited members.

- **create** (defined in `URLHandler_Setup`)  
  `URLHandler_Setup`

- **delete** (defined in `URLHandler_Setup`)  
  `URLHandler_Setup`

- **params** (defined in `URLHandler_Setup`)  
  `URLHandler_Setup`

- **process** (defined in `URLHandler_Setup`)  
  `URLHandler_Setup`

- **scan** (defined in `URLHandler_Setup`)  
  `URLHandler_Setup`