HTTP is a stateless application-level protocol used for data transfer. This section provides information about the HTTP Client APIs and data structures that can be used by user applications to connect to HTTP server and transfer data.

# Features and Limitations

Following is the list of salient features of this HTTP/S implementation:

1. HTTP 1.1 version. Both HTTP and HTTPS
2. Both the server IP address and the host name (Domain Name)
3. HTTP GET and POST methods
4. Setting of headers by the application. Only the Host header is implicitly set
5. Supports receiving the response with Transfer encoding Chunked
6. Supports chain of CA certificates or the bundle of CA certificates (as configured in browser)

Following are the limitations:

1. CA certificate must be in the PEM format
2. HTTP redirection is not supported

# Header file/s

Components/http/inc/http\_client.h.

# Data Structure Definitions

## http\_client\_config\_t

This data structure is used to pass the parameters while opening a HTTP connection with the remote server using http\_client\_open API.

|  |  |
| --- | --- |
| ***hostname*** | Host name or the IP address of the server. Example, “google.com” or “192.168.1.1” |
| ***port*** | Server port |
| ***secured*** | 0 – HTTP  1- HTTPS without server verification  2 – HTTPS with server certificate validation |
| ***ssl\_wrap\_cfg\_t ssl\_cfg*** | SSL configuration for secured connection |
| ***time\_out*** | Connect timeout in seconds |

Table : http\_client\_config\_t - parameters

## http\_client\_resp\_info\_t

This data structure is used to pass information about the data received from the server when HTTP GET is done using the http\_client\_get API.

|  |  |
| --- | --- |
| ***status\_code*** | HTTP response status code |
| ***resp\_hdrs*** | Response headers. Array of strings |
| ***resp\_body*** | Response body len |
| ***resp\_len*** | Resp len, currently available in the resp\_body |
| ***resp\_total\_len*** | Total length of the response body. If 0, no total length available beforehand as the body maybe sent using chunked or multipart encoding |
| ***more\_data*** | More data will be followed. The callback will be called again |

Table : http\_client\_resp\_info\_t - parameters

# API Reference

## http\_client\_open

### Overview

This API connects to the remote HTTP server. The configuration needed for the connection is passed using http\_client\_config\_t.

### Definition

|  |
| --- |
| http\_client\_handle\_t  http\_client\_open(http\_client\_config\_t \*cfg) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *cfg* | Pointer to the data structure http\_client\_config\_t |

Table : http\_client\_open – parameters

### Return

Success: Pointer to HTTP client connection handle

Error: NULL

## http\_client\_get

### Overview

This function is used for performing HTTP GET. The HTTP response is provided through the call back. The call back is called multiple times until the whole response is received.

### Definition

|  |
| --- |
| int  http\_client\_get(http\_client\_handle\_t handle, char \*uri, http\_client\_resp\_cb cb, void \*cb\_ctx, int time\_out) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *handle* | Handle returned by http\_client\_open() |
| *uri* | HTTP URI to GET |
| *cb* | User application callback to be called when response is received from the server |

Table : http\_client\_get - parameters

### Return

Success: 0

Error: -1

## http\_client\_post

### Overview

This function is used to perform HTTP POST. Using this data can be sent to the HTTP server. The response is provided using the call back. Setting content length header is a must using http\_client\_set\_req\_hdr before calling this API.

### Definition

|  |
| --- |
| int  http\_client\_post(http\_client\_handle\_t handle, char \*uri,  char \*buff, int buff\_len,  http\_client\_resp\_cb cb, void \*cb\_ctx,  int time\_out) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *handle* | Handle returned by http\_client\_open() |
| *uri* | HTTP URI to POST |
| *buff* | Buffer having data to be sent to server |
| *buff\_len* | Length of the data present in the buff. This is the length of the data to be POSTed |
| *cb* | User application callback to be called when response is received from the server |

Table : http\_client\_post - parameters

### Return

Success: 0

Error: -1

## http\_client\_set\_req\_hdr

### Overview

User application can set the header to be sent with GET/POST request using this API.

### Definition

|  |
| --- |
| int  http\_client\_set\_req\_hdr(http\_client\_handle\_t handle,  const char \*hdrname, const char \*hdrval) |

### Parameters

|  |  |
| --- | --- |
| **Parameters** | **Description** |
| *handle* | HTTP connection handle |
| *hdrname* | Name part of the header. For example: “content length” |
| *hdrval* | Value part of the header. For example: “1024” |

Table : http\_client\_set\_req\_hdr - parameters

### Return

Success: 0

Error: -1

## http\_client\_close

### Overview

This API is used for closing the connection.

### Definition

|  |
| --- |
| int  http\_client\_close(http\_client\_handle\_t handle) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *handle* | HTTP connection handle |

Table : http\_client\_close - parameters

### Return

Success: 0

Error: -1

# Application Example

For the example code, refer: *examples/http\_client application*.