SFML

* [Main Page](http://docs.google.com/index.htm)
* [Modules](http://docs.google.com/modules.htm)
* [Classes](http://docs.google.com/annotated.htm)
* [Files](http://docs.google.com/files.htm)
* [Class List](http://docs.google.com/annotated.htm)
* [Class Index](http://docs.google.com/classes.htm)
* [Class Hierarchy](http://docs.google.com/hierarchy.htm)
* [Class Members](http://docs.google.com/functions.htm)
* **sf**
* [Http](http://docs.google.com/classsf_1_1Http.htm)

[Classes](#_gjdgxs) | [Public Member Functions](#_30j0zll) | [List of all members](http://docs.google.com/classsf_1_1Http-members.htm)

sf::Http Class Reference

[Network module](http://docs.google.com/group__network.htm)

A HTTP client. [More...](http://docs.google.com/classsf_1_1Http.htm#details)

#include <[Http.hpp](http://docs.google.com/Http_8hpp_source.htm)>

Inheritance diagram for sf::Http:



| Classes | |
| --- | --- |
| class | [Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm) |
|  | Define a HTTP request. [More...](http://docs.google.com/classsf_1_1Http_1_1Request.htm#details) |
|  | |
| class | [Response](http://docs.google.com/classsf_1_1Http_1_1Response.htm) |
|  | Define a HTTP response. [More...](http://docs.google.com/classsf_1_1Http_1_1Response.htm#details) |
|  | |

| Public Member Functions | |
| --- | --- |
|  | [Http](http://docs.google.com/classsf_1_1Http.htm#abe2360194f99bdde402c9f97a85cf067) () |
|  | Default constructor. |
|  | |
|  | [Http](http://docs.google.com/classsf_1_1Http.htm#a79efd844a735f083fcce0edbf1092385) (const std::string &host, unsigned short port=0) |
|  | Construct the HTTP client with the target host. |
|  | |
| void | [setHost](http://docs.google.com/classsf_1_1Http.htm#a55121d543b61c41cf20b885a97b04e65) (const std::string &host, unsigned short port=0) |
|  | Set the target host. |
|  | |
| [Response](http://docs.google.com/classsf_1_1Http_1_1Response.htm) | [sendRequest](http://docs.google.com/classsf_1_1Http.htm#aaf09ebfb5e00dcc82e0d494d5c6a9e2a) (const [Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm) &request, [Time](http://docs.google.com/classsf_1_1Time.htm) timeout=[Time::Zero](http://docs.google.com/classsf_1_1Time.htm#a8db127b632fa8da21550e7282af11fa0)) |
|  | Send a HTTP request and return the server's response. |
|  | |

## Detailed Description

A HTTP client.

[sf::Http](http://docs.google.com/classsf_1_1Http.htm) is a very simple HTTP client that allows you to communicate with a web server.

You can retrieve web pages, send data to an interactive resource, download a remote file, etc.

The HTTP client is split into 3 classes:

* [sf::Http::Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm)
* [sf::Http::Response](http://docs.google.com/classsf_1_1Http_1_1Response.htm)
* [sf::Http](http://docs.google.com/classsf_1_1Http.htm)

[sf::Http::Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm) builds the request that will be sent to the server. A request is made of:

* a method (what you want to do)
* a target URI (usually the name of the web page or file)
* one or more header fields (options that you can pass to the server)
* an optional body (for POST requests)

[sf::Http::Response](http://docs.google.com/classsf_1_1Http_1_1Response.htm) parse the response from the web server and provides getters to read them. The response contains:

* a status code
* header fields (that may be answers to the ones that you requested)
* a body, which contains the contents of the requested resource

[sf::Http](http://docs.google.com/classsf_1_1Http.htm) provides a simple function, SendRequest, to send a [sf::Http::Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm) and return the corresponding [sf::Http::Response](http://docs.google.com/classsf_1_1Http_1_1Response.htm) from the server.

Usage example:

// Create a new HTTP client

[sf::Http](http://docs.google.com/classsf_1_1Http.htm) http;

// We'll work on http://www.sfml-dev.org

http.[setHost](http://docs.google.com/classsf_1_1Http.htm#a55121d543b61c41cf20b885a97b04e65)("http://www.sfml-dev.org");

// Prepare a request to get the 'features.php' page

[sf::Http::Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm) request("features.php");

// Send the request

[sf::Http::Response](http://docs.google.com/classsf_1_1Http_1_1Response.htm) response = http.[sendRequest](http://docs.google.com/classsf_1_1Http.htm#aaf09ebfb5e00dcc82e0d494d5c6a9e2a)(request);

// Check the status code and display the result

[sf::Http::Response::Status](http://docs.google.com/classsf_1_1Http_1_1Response.htm#a663e071978e30fbbeb20ed045be874d8) status = response.[getStatus](http://docs.google.com/classsf_1_1Http_1_1Response.htm#a542e9856b1dd260a83940eb982b7f19a)();

if (status == [sf::Http::Response::Ok](http://docs.google.com/classsf_1_1Http_1_1Response.htm#a663e071978e30fbbeb20ed045be874d8a0158f932254d3f09647dd1f64bd43832))

{

std::cout << response.[getBody](http://docs.google.com/classsf_1_1Http_1_1Response.htm#a6b74ef73051a16ebb20041495c758e22)() << std::endl;

}

else

{

std::cout << "Error " << status << std::endl;

}

Definition at line [46](http://docs.google.com/Http_8hpp_source.htm#l00046) of file [Http.hpp](http://docs.google.com/Http_8hpp_source.htm).

## Constructor & Destructor Documentation

| sf::Http::Http | ( |  | ) |  |
| --- | --- | --- | --- | --- |

Default constructor.

| sf::Http::Http | ( | const std::string & | *host*, |
| --- | --- | --- | --- |
|  |  | unsigned short | *port* = 0 |
|  | ) |  |  |

Construct the HTTP client with the target host.

This is equivalent to calling setHost(host, port). The port has a default value of 0, which means that the HTTP client will use the right port according to the protocol used (80 for HTTP, 443 for HTTPS). You should leave it like this unless you really need a port other than the standard one, or use an unknown protocol.

Parameters

| host | Web server to connect to |
| --- | --- |
| port | Port to use for connection |

## Member Function Documentation

| [Response](http://docs.google.com/classsf_1_1Http_1_1Response.htm) sf::Http::sendRequest | ( | const [Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm) & | *request*, |
| --- | --- | --- | --- |
|  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *timeout* = [Time::Zero](http://docs.google.com/classsf_1_1Time.htm#a8db127b632fa8da21550e7282af11fa0) |
|  | ) |  |  |

Send a HTTP request and return the server's response.

You must have a valid host before sending a request (see setHost). Any missing mandatory header field in the request will be added with an appropriate value. Warning: this function waits for the server's response and may not return instantly; use a thread if you don't want to block your application, or use a timeout to limit the time to wait. A value of [Time::Zero](http://docs.google.com/classsf_1_1Time.htm#a8db127b632fa8da21550e7282af11fa0) means that the client will use the system defaut timeout (which is usually pretty long).

Parameters

| request | [Request](http://docs.google.com/classsf_1_1Http_1_1Request.htm) to send |
| --- | --- |
| timeout | Maximum time to wait |

ReturnsServer's response

| void sf::Http::setHost | ( | const std::string & | *host*, |
| --- | --- | --- | --- |
|  |  | unsigned short | *port* = 0 |
|  | ) |  |  |

Set the target host.

This function just stores the host address and port, it doesn't actually connect to it until you send a request. The port has a default value of 0, which means that the HTTP client will use the right port according to the protocol used (80 for HTTP, 443 for HTTPS). You should leave it like this unless you really need a port other than the standard one, or use an unknown protocol.

Parameters

| host | Web server to connect to |
| --- | --- |
| port | Port to use for connection |

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

* [Http.hpp](http://docs.google.com/Http_8hpp_source.htm)

Copyright � Laurent Gomila  ::  Documentation generated by [doxygen](http://www.doxygen.org/)  ::