

My Project

Generated by Doxygen 1.9.8

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 curling::Request Class Reference	5
3.1.1 Detailed Description	6
3.1.2 Member Enumeration Documentation	6
3.1.2.1 Method	6
3.1.3 Constructor & Destructor Documentation	6
3.1.3.1 Request()	6
3.1.3.2 ~Request()	6
3.1.4 Member Function Documentation	7
3.1.4.1 addArg()	7
3.1.4.2 addHeader()	7
3.1.4.3 setAuthToken()	7
3.1.4.4 setBody()	7
3.1.4.5 setConnectTimeout()	8
3.1.4.6 setMethod()	8
3.1.4.7 setProxy()	8
3.1.4.8 setTimeout()	8
3.1.4.9 setURL()	8
3.2 curling::Response Struct Reference	9
3.2.1 Detailed Description	9
3.2.2 Member Data Documentation	9
3.2.2.1 headers	9
4 File Documentation	11
4.1 curling.hpp	11
Index	13

Chapter 1

Class Index

1.1 Class List

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

curling::Request	
Handles HTTP requests using libcurl	5
curling::Response	
Represents an HTTP response	9

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

include/ curling.hpp	11
--	----

Chapter 3

Class Documentation

3.1 curling::Request Class Reference

Handles HTTP requests using libcurl.

```
#include <curling.hpp>
```

Public Types

- enum class [Method](#) { [GET](#) , [POST](#) , [PUT](#) , [DELETE](#) }
Enumerates the supported HTTP methods.

Public Member Functions

- [Request](#) ()
Constructor for the [Request](#) class.
- [~Request](#) ()
Destructor for the [Request](#) class.
- [Request](#) (const [Request](#) &)=delete
- [Request](#) & [operator=](#) (const [Request](#) &)=delete
- [Request](#) ([Request](#) &&)=delete
- [Request](#) & [operator=](#) ([Request](#) &&)=delete
- void [setMethod](#) ([Method](#) m)
Sets the HTTP method for the request.
- void [setURL](#) (const std::string &URL)
Sets the URL for the request.
- void [setProxy](#) (const std::string &URL)
Sets the URL for the request.
- void [addArg](#) (const std::string &key, const std::string &value)
Adds an argument to the query string of the request.
- void [addHeader](#) (const std::string &header)
Adds a header to the request.
- void [setBody](#) (const std::string &body)
Sets the body of the HTTP request, applicable for POST and PUT methods.
- [Response](#) [send](#) ()

Sends the HTTP request using libcurl.

- void **reset** ()
Resets the internal state for reuse of this [Request](#) instance.
- void **setTimeout** (long seconds)
Sets timeout of request.
- void **setConnectTimeout** (long seconds)
Sets timeout to connect.
- void **setAuthToken** (const std::string &token)
Sets the header "Authorization: Bearer <token>".

3.1.1 Detailed Description

Handles HTTP requests using libcurl.

This class provides functionality to perform HTTP operations such as GET, POST, PUT, and DELETE. It manages the setup and execution of these requests with libcurl.

3.1.2 Member Enumeration Documentation

3.1.2.1 Method

```
enum class curling::Request::Method [strong]
```

Enumerates the supported HTTP methods.

Enumerator

GET	Represents an HTTP GET request.
POST	Represents an HTTP POST request.
PUT	Represents an HTTP PUT request.
DELETE	Represents an HTTP DELETE request.

3.1.3 Constructor & Destructor Documentation

3.1.3.1 Request()

```
curling::Request::Request ( )
```

Constructor for the [Request](#) class.

Initializes a new instance of the [Request](#) class and increments the instances counter.

3.1.3.2 ~Request()

```
curling::Request::~~Request ( )
```

Destructor for the [Request](#) class.

Decrements the instances counter and performs any necessary cleanup.

3.1.4 Member Function Documentation

3.1.4.1 addArg()

```
void curling::Request::addArg (
    const std::string & key,
    const std::string & value )
```

Adds an argument to the query string of the request.

Parameters

<i>key</i>	The arg key
<i>value</i>	The arg value

3.1.4.2 addHeader()

```
void curling::Request::addHeader (
    const std::string & header )
```

Adds a header to the request.

Parameters

<i>header</i>	The header string to be added (e.g., "Content-Type: text/html").
---------------	--

3.1.4.3 setAuthToken()

```
void curling::Request::setAuthToken (
    const std::string & token )
```

Sets the header "Authorization: Bearer <token>".

This method sets the header with a given authorization token.

3.1.4.4 setBody()

```
void curling::Request::setBody (
    const std::string & body )
```

Sets the body of the HTTP request, applicable for POST and PUT methods.

Parameters

<i>body</i>	The body content as a string.
-------------	-------------------------------

3.1.4.5 setConnectTimeout()

```
void curling::Request::setConnectTimeout (
    long seconds )
```

Sets timeout to connect.

This method is to limit the amount of time per connection

3.1.4.6 setMethod()

```
void curling::Request::setMethod (
    Method m )
```

Sets the HTTP method for the request.

Parameters

<i>m</i>	The method to set (e.g., Method::GET , Method::POST).
----------	--

3.1.4.7 setProxy()

```
void curling::Request::setProxy (
    const std::string & URL )
```

Sets the URL for the request.

Parameters

<i>URL</i>	The URL of the Proxy, a string.
------------	---------------------------------

3.1.4.8 setTimeout()

```
void curling::Request::setTimeout (
    long seconds )
```

Sets timeout of request.

This method is to limit the amount of time per request

3.1.4.9 setURL()

```
void curling::Request::setURL (
    const std::string & URL )
```

Sets the URL for the request.

Parameters

<i>URL</i>	The URL string to be used in the HTTP request.
------------	--

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

- include/curling.hpp

3.2 curling::Response Struct Reference

Represents an HTTP response.

```
#include <curling.hpp>
```

Public Attributes

- long **httpCode**
The HTTP status code received in the response.
- std::string **body**
The body content of the HTTP response as a string.
- std::map< std::string, std::vector< std::string > > **headers**
A map to store HTTP headers from the response.

3.2.1 Detailed Description

Represents an HTTP response.

This structure holds details of an HTTP response, including the status code, body content, and headers.

3.2.2 Member Data Documentation

3.2.2.1 headers

```
std::map<std::string, std::vector<std::string> > curling::Response::headers
```

A map to store HTTP headers from the response.

Each header is stored with its name as the key and the corresponding value is stored into a vector, as there can be many headers with same key.

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

- include/curling.hpp

Chapter 4

File Documentation

4.1 curling.hpp

```
00001 #ifndef CURLING_HPP
00002 #define CURLING_HPP
00003
00004 #include <iostream>
00005 #include <string>
00006 #include <map>
00007 #include <vector>
00008 #include <sstream>
00009 #include <atomic>
00010 #include <mutex>
00011 #include <stdexcept>
00012 #include <algorithm>
00013 #include <cctype>
00014 #include <locale>
00015 #include <curl/curl.h>
00016
00017 namespace curling {
00018
00026 struct Response {
00027     long httpCode;
00028
00029     std::string body;
00030
00037     std::map<std::string, std::vector<std::string>> headers;
00038 };
00039
00040
00048 class Request {
00049 public:
00054     enum class Method {
00055         GET,
00056         POST,
00057         PUT,
00058         DELETE
00059     };
00060
00067     Request();
00068
00074     ~Request();
00075
00076     Request(const Request&) = delete;
00077     Request& operator=(const Request&) = delete;
00078     Request(Request&&) = delete;
00079     Request& operator=(Request&&) = delete;
00080
00086     void setMethod(Method m);
00087
00093     void setURL(const std::string& URL);
00094
00100     void setProxy(const std::string& URL);
00101
00109     void addArg(const std::string& key, const std::string& value);
00110
00116     void addHeader(const std::string& header);
00117
00123     void setBody(const std::string& body);
00124
00128     Response send();
```

```
00129
00133     void reset();
00134
00135
00141     void setTimeout(long seconds);
00142
00143
00149     void setConnectTimeout(long seconds);
00150
00156     void setAuthToken(const std::string& token);
00157
00158 private:
00159     Method method;
00160     CURL* curlHandle;
00161     struct curl_slist* list;
00162     std::string url, args, body, cookieFile, cookieJar;
00163
00179     static size_t WriteCallback(void* contents, size_t size, size_t nmem, void* userp);
00180
00195     static size_t HeaderCallback(char* buffer, size_t size, size_t nitems, void* userdata);
00196
00203     void clean();
00204
00211     void updateURL();
00212
00218     static void trim(std::string & s);
00219
00220 };
00221
00222 } // namespace curling
00223
00224 #endif // CURLING_HPP
```


Index

- ~Request
 - [curling::Request](#), 6
- addArg
 - [curling::Request](#), 7
- addHeader
 - [curling::Request](#), 7
- [curling::Request](#), 5
 - ~Request, 6
 - addArg, 7
 - addHeader, 7
 - DELETE, 6
 - GET, 6
 - Method, 6
 - POST, 6
 - PUT, 6
 - Request, 6
 - setAuthToken, 7
 - setBody, 7
 - setConnectTimeout, 7
 - setMethod, 8
 - setProxy, 8
 - setTimeout, 8
 - setURL, 8
- [curling::Response](#), 9
 - headers, 9
- DELETE
 - [curling::Request](#), 6
- GET
 - [curling::Request](#), 6
- headers
 - [curling::Response](#), 9
- [include/curling.hpp](#), 11
- Method
 - [curling::Request](#), 6
- POST
 - [curling::Request](#), 6
- PUT
 - [curling::Request](#), 6
- Request
 - [curling::Request](#), 6
- setAuthToken
 - [curling::Request](#), 7
- setBody
 - [curling::Request](#), 7
- setConnectTimeout
 - [curling::Request](#), 7
- setMethod
 - [curling::Request](#), 8
- setProxy
 - [curling::Request](#), 8
- setTimeout
 - [curling::Request](#), 8
- setURL
 - [curling::Request](#), 8