

## EHN Group 12 Practical 1

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b><a href="#">Class Index</a></b>	<b>1</b>
1.1	<a href="#">Class List</a> . . . . .	1
<b>2</b>	<b><a href="#">File Index</a></b>	<b>3</b>
2.1	<a href="#">File List</a> . . . . .	3
<b>3</b>	<b><a href="#">Class Documentation</a></b>	<b>5</b>
3.1	<a href="#">client_args Struct Reference</a> . . . . .	5
3.1.1	<a href="#">Detailed Description</a> . . . . .	6
3.1.2	<a href="#">Member Data Documentation</a> . . . . .	6
3.1.2.1	<a href="#">abio</a> . . . . .	6
3.1.2.2	<a href="#">thread_number</a> . . . . .	6
3.2	<a href="#">server_args Struct Reference</a> . . . . .	6
3.2.1	<a href="#">Detailed Description</a> . . . . .	7
3.2.2	<a href="#">Member Data Documentation</a> . . . . .	7
3.2.2.1	<a href="#">abio</a> . . . . .	7
3.2.2.2	<a href="#">acpt</a> . . . . .	7

<b>4 File Documentation</b>	<b>9</b>
4.1 Client.c File Reference	9
4.1.1 Function Documentation	9
4.1.1.1 clear_buffer()	9
4.1.1.2 main()	10
4.2 Client.h File Reference	10
4.2.1 Macro Definition Documentation	11
4.2.1.1 MAX_REQ_LEN	11
4.2.2 Function Documentation	11
4.2.2.1 clear_buffer()	11
4.2.2.2 main()	12
4.3 Server.c File Reference	12
4.3.1 Function Documentation	13
4.3.1.1 double_size()	13
4.3.1.2 itoa()	13
4.3.1.3 main()	13
4.3.1.4 new_client_connection()	14
4.3.1.5 read_media()	14
4.3.1.6 server_thread()	14
4.3.1.7 write_page()	15
4.4 Server.h File Reference	15
4.4.1 Macro Definition Documentation	16
4.4.1.1 DEBUG	17
4.4.2 Function Documentation	17
4.4.2.1 double_size()	17
4.4.2.2 itoa()	17
4.4.2.3 main()	18
4.4.2.4 new_client_connection()	18
4.4.2.5 read_media()	19
4.4.2.6 server_thread()	19
4.4.2.7 write_page()	19
4.4.3 Variable Documentation	20
4.4.3.1 Medialtems	20
4.4.3.2 numMedialtems	20
4.4.3.3 SERVER_RUN	20
<b>Index</b>	<b>21</b>

# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">client_args</a>	This struct is passed as an argument to newly created client threads to allow multiple arguments to be passed . . . . .	5
<a href="#">server_args</a>	This struct is passed as an argument to the server thread to allow multiple arguments to be passed . . . . .	6



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<a href="#">Client.c</a>	.....	9
<a href="#">Client.h</a>	.....	10
<a href="#">Server.c</a>	.....	12
<a href="#">Server.h</a>	.....	15





## Chapter 3

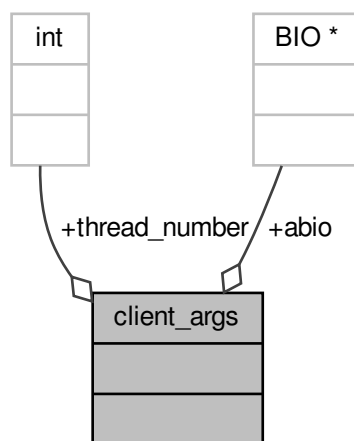
# Class Documentation

### 3.1 client\_args Struct Reference

This struct is passed as an argument to newly created client threads to allow multiple arguments to be passed.

```
#include <Server.h>
```

Collaboration diagram for client\_args:



#### Public Attributes

- `BIO * abio`  
*The SSL object pointer.*
- `int thread_number`  
*The current thread number.*

### 3.1.1 Detailed Description

This struct is passed as an argument to newly created client threads to allow multiple arguments to be passed.

### 3.1.2 Member Data Documentation

#### 3.1.2.1 abio

```
BIO* client_args::abio
```

The SSL object pointer.

#### 3.1.2.2 thread\_number

```
int client_args::thread_number
```

The current thread number.

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

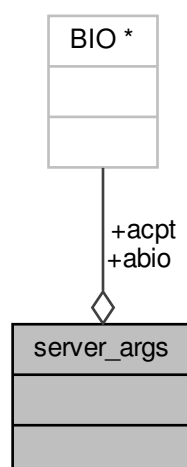
- [Server.h](#)

## 3.2 server\_args Struct Reference

This struct is passed as an argument to the server thread to allow multiple arguments to be passed.

```
#include <Server.h>
```

Collaboration diagram for server\_args:



## Public Attributes

- BIO \* [acpt](#)  
*The SSL reception buffer.*
- BIO \* [abio](#)  
*The SSL object pointer.*

### 3.2.1 Detailed Description

This struct is passed as an argument to the server thread to allow multiple arguments to be passed.

### 3.2.2 Member Data Documentation

#### 3.2.2.1 abio

```
BIO* server_args::abio
```

The SSL object pointer.

#### 3.2.2.2 acpt

```
BIO* server_args::acpt
```

The SSL reception buffer.

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

- [Server.h](#)



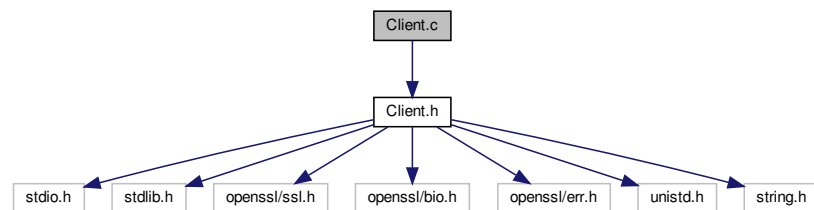
## Chapter 4

# File Documentation

### 4.1 Client.c File Reference

```
#include "Client.h"
```

Include dependency graph for Client.c:



### Functions

- int [main](#) (int argc, char \*argv[ ])
- void [clear\\_buffer](#) (char \*buffer, int length)

#### 4.1.1 Function Documentation

##### 4.1.1.1 clear\_buffer()

```
void clear_buffer (
    char * buffer,
    int length )
```

Clears a buffer up to a specified length.

**Parameters**

<i>buffer</i>	The buffer to be cleared.
<i>length</i>	The length up to which the buffer must be cleared.

**4.1.1.2 main()**

```
int main (
    int argc,
    char * argv[] )
```

Sets up the client SSL connection, connects to the server and then displays or downloads requested files from the server

**Parameters**

<i>argc</i>	The number of arguments passes to the function.
<i>argv</i>	The values of the passes arguments as c-strings.

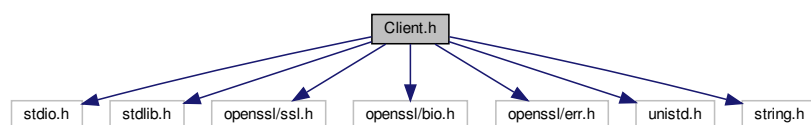
**Returns**

Successful or failed execution.

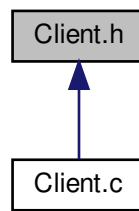
**4.2 Client.h File Reference**

```
#include <stdio.h>
#include <stdlib.h>
#include <openssl/ssl.h>
#include <openssl/bio.h>
#include <openssl/err.h>
#include <unistd.h>
#include <string.h>
```

Include dependency graph for Client.h:



This graph shows which files directly or indirectly include this file:



## Macros

- `#define MAX_REQ_LEN 255`  
*The maximum length of a request (characters).*

## Functions

- `int main (int argc, char *argv[ ])`
- `void clear_buffer (char *buffer, int length)`

### 4.2.1 Macro Definition Documentation

#### 4.2.1.1 MAX\_REQ\_LEN

```
#define MAX_REQ_LEN 255
```

The maximum length of a request (characters).

### 4.2.2 Function Documentation

#### 4.2.2.1 clear\_buffer()

```
void clear_buffer (  
    char * buffer,  
    int length )
```

Clears a buffer up to a specified length.

**Parameters**

<i>buffer</i>	The buffer to be cleared.
<i>length</i>	The length up to which the buffer must be cleared.

**4.2.2.2 main()**

```
int main (
    int argc,
    char * argv[] )
```

Sets up the client SSL connection, connects to the server and then displays or downloads requested files from the server

**Parameters**

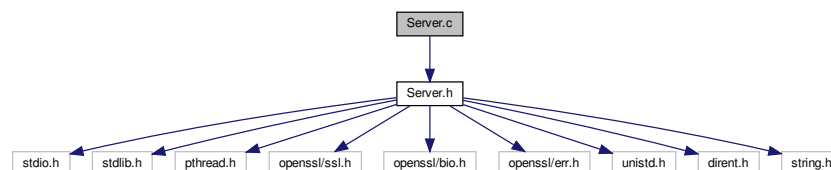
<i>argc</i>	The number of arguments passes to the function.
<i>argv</i>	The values of the passes arguments as c-strings.

**Returns**

Successful or failed execution.

**4.3 Server.c File Reference**

```
#include "Server.h"
Include dependency graph for Server.c:
```

**Functions**

- int [main](#) (int argc, char \*argv[])
- void \* [server\\_thread](#) (void \*ptr)
- void \* [new\\_client\\_connection](#) (void \*ptr)
- int [read\\_media](#) ()
- int [write\\_page](#) (BIO \*bio, const char \*page, const char \*filename)
- pthread\_t \* [double\\_size](#) (pthread\_t \*old\_clients, int current\_size)
- char \* [itoa](#) (char \*result, int number)



### 4.3.1 Function Documentation

#### 4.3.1.1 double\_size()

```
pthread_t* double_size (
    pthread_t * old_clients,
    int current_size )
```

When the current clients array is full, create a new one with double the size.

##### Parameters

<i>old_clients</i>	The previous array of clients.
<i>current_size</i>	The previous size of the clients array.

##### Returns

A pointer to the new clients array.

#### 4.3.1.2 itoa()

```
char* itoa (
    char * result,
    int number )
```

Convert between an integer and a c-string.

##### Parameters

<i>result</i>	The c-string to be used for the output.
<i>number</i>	The number to be converted

##### Returns

The same c-string used for the output.

#### 4.3.1.3 main()

```
int main (
    int argc,
    char * argv[] )
```

Sets up the client SSL connection, connects to the server and then displays or downloads requested files from the server

**Parameters**

<i>argc</i>	The number of arguments passes to the function.
<i>argv</i>	The values of the passes arguments as c-strings.

**Returns**

Successful or failed execution.

**4.3.1.4 new\_client\_connection()**

```
void* new_client_connection (
    void * ptr )
```

This function is created as a new thread for every client that makes a request to the server.

**Parameters**

<i>ptr</i>	The <a href="#">client_args</a> struct is passes as a void pointer.
------------	---

**Returns**

Successful or failed execution.

**4.3.1.5 read\_media()**

```
int read_media ( )
```

Read all the contents of the Media\_files folder for use later in GET requests.

**Returns**

Successful or failed execution.

**4.3.1.6 server\_thread()**

```
void* server_thread (
    void * ptr )
```

This function is created as a new thread and handles all client requests.

## Parameters

<i>ptr</i>	The <a href="#">server_args</a> struct is passes as a void pointer.
------------	---

## Returns

Successful or failed execution.

## 4.3.1.7 write\_page()

```
int write_page (
    BIO * bio,
    const char * page,
    const char * filename )
```

Write an arbitrary file to the client.

## Parameters

<i>bio</i>	A pointer to the client's SSL object.
<i>page</i>	The file to be written.
<i>filename</i>	The name of the file to be written.

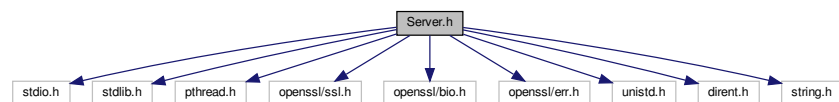
## Returns

Successful or failed execution.

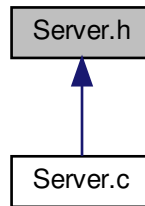
## 4.4 Server.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <openssl/ssl.h>
#include <openssl/bio.h>
#include <openssl/err.h>
#include <unistd.h>
#include <dirent.h>
#include <string.h>
```

Include dependency graph for Server.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [server\\_args](#)  
*This struct is passed as an argument to the server thread to allow multiple arguments to be passed.*
- struct [client\\_args](#)  
*This struct is passed as an argument to newly created client threads to allow multiple arguments to be passed.*

## Macros

- `#define` [DEBUG](#) 0  
*Enable (1) or disable (0) client thread debugging.*

## Functions

- int [main](#) (int argc, char \*argv[])
- void \* [server\\_thread](#) (void \*ptr)
- void \* [new\\_client\\_connection](#) (void \*ptr)
- pthread\_t \* [double\\_size](#) (pthread\_t \*old\_clients, int current\_size)
- int [write\\_page](#) (BIO \*bio, const char \*page, const char \*filename)
- int [read\\_media](#) ()
- char \* [itoa](#) (char \*result, int number)

## Variables

- char [MediaItems](#) [100][256]
- int [numMediaItems](#) = 0
- int [SERVER\\_RUN](#)

### 4.4.1 Macro Definition Documentation

#### 4.4.1.1 DEBUG

```
#define DEBUG 0
```

Enable (1) or disable (0) client thread debugging.

### 4.4.2 Function Documentation

#### 4.4.2.1 double\_size()

```
pthread_t* double_size (
    pthread_t * old_clients,
    int current_size )
```

When the current clients array is full, create a new one with double the size.

##### Parameters

<i>old_clients</i>	The previous array of clients.
<i>current_size</i>	The previous size of the clients array.

##### Returns

A pointer to the new clients array.

#### 4.4.2.2 itoa()

```
char* itoa (
    char * result,
    int number )
```

Convert between an integer and a c-string.

##### Parameters

<i>result</i>	The c-string to be used for the output.
<i>number</i>	The number to be converted

##### Returns

The same c-string used for the output.

#### 4.4.2.3 main()

```
int main (
    int argc,
    char * argv[] )
```

The main function sets up all the SSL functions, Certificates and starts the server.

##### Parameters

<i>argc</i>	The number of arguments passes to the function.
<i>argv</i>	The values of the passes arguments as c-strings.

##### Returns

Successful or failed execution.

Sets up the client SSL connection, connects to the server and then displays or downloads requested files from the server

##### Parameters

<i>argc</i>	The number of arguments passes to the function.
<i>argv</i>	The values of the passes arguments as c-strings.

##### Returns

Successful or failed execution.

#### 4.4.2.4 new\_client\_connection()

```
void* new_client_connection (
    void * ptr )
```

This function is created as a new thread for every client that makes a request to the server.

##### Parameters

<i>ptr</i>	The <a href="#">client_args</a> struct is passes as a void pointer.
------------	---

##### Returns

Successful or failed execution.

#### 4.4.2.5 read\_media()

```
int read_media ( )
```

Read all the contents of the Media\_files folder for use later in GET requests.

##### Returns

Successful or failed execution.

#### 4.4.2.6 server\_thread()

```
void* server_thread (
    void * ptr )
```

This function is created as a new thread and handles all client requests.

##### Parameters

<i>ptr</i>	The <a href="#">server_args</a> struct is passes as a void pointer.
------------	---

##### Returns

Successful or failed execution.

#### 4.4.2.7 write\_page()

```
int write_page (
    BIO * bio,
    const char * page,
    const char * filename )
```

Write an arbitrary file to the client.

##### Parameters

<i>bio</i>	A pointer to the client's SSL object.
<i>page</i>	The file to be written.
<i>filename</i>	The name of the file to be written.

##### Returns

Successful or failed execution.

### 4.4.3 Variable Documentation

#### 4.4.3.1 MediaItems

```
char MediaItems[100][256]
```

#### 4.4.3.2 numMediaItems

```
int numMediaItems = 0
```

#### 4.4.3.3 SERVER\_RUN

```
int SERVER_RUN
```



# Index

- abio
  - client\_args, [6](#)
  - server\_args, [7](#)
- acpt
  - server\_args, [7](#)
- clear\_buffer
  - Client.c, [9](#)
  - Client.h, [11](#)
- Client.c, [9](#)
  - clear\_buffer, [9](#)
  - main, [10](#)
- Client.h, [10](#)
  - clear\_buffer, [11](#)
  - MAX\_REQ\_LEN, [11](#)
  - main, [12](#)
- client\_args, [5](#)
  - abio, [6](#)
  - thread\_number, [6](#)
- DEBUG
  - Server.h, [16](#)
- double\_size
  - Server.c, [13](#)
  - Server.h, [17](#)
- itoa
  - Server.c, [13](#)
  - Server.h, [17](#)
- MAX\_REQ\_LEN
  - Client.h, [11](#)
- main
  - Client.c, [10](#)
  - Client.h, [12](#)
  - Server.c, [13](#)
  - Server.h, [17](#)
- MedialItems
  - Server.h, [20](#)
- new\_client\_connection
  - Server.c, [14](#)
  - Server.h, [18](#)
- numMedialItems
  - Server.h, [20](#)
- read\_media
  - Server.c, [14](#)
  - Server.h, [18](#)
- SERVER\_RUN
- Server.h, [20](#)
- Server.c, [12](#)
  - double\_size, [13](#)
  - itoa, [13](#)
  - main, [13](#)
  - new\_client\_connection, [14](#)
  - read\_media, [14](#)
  - server\_thread, [14](#)
  - write\_page, [15](#)
- Server.h, [15](#)
  - DEBUG, [16](#)
  - double\_size, [17](#)
  - itoa, [17](#)
  - main, [17](#)
  - MedialItems, [20](#)
  - new\_client\_connection, [18](#)
  - numMedialItems, [20](#)
  - read\_media, [18](#)
  - SERVER\_RUN, [20](#)
  - server\_thread, [19](#)
  - write\_page, [19](#)
- server\_args, [6](#)
  - abio, [7](#)
  - acpt, [7](#)
- server\_thread
  - Server.c, [14](#)
  - Server.h, [19](#)
- thread\_number
  - client\_args, [6](#)
- write\_page
  - Server.c, [15](#)
  - Server.h, [19](#)