

WS
1.0

Generated by Doxygen 1.8.9.1

Mon Mar 2 2015 00:03:33

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	http_req Struct Reference	5
3.1.1	Detailed Description	5
3.1.2	Field Documentation	5
3.1.2.1	fd	5
3.1.2.2	path	5
3.1.2.3	req_len	5
3.1.2.4	request	5
3.1.2.5	resp_hd_len	5
3.1.2.6	resp_head	5
3.1.2.7	resp_len	6
3.1.2.8	response	6
3.2	job Struct Reference	6
3.2.1	Detailed Description	6
3.2.2	Field Documentation	6
3.2.2.1	args	6
3.2.2.2	function	6
3.2.2.3	next	6
3.2.2.4	prev	7
3.3	job_list Struct Reference	7
3.3.1	Detailed Description	7
3.3.2	Field Documentation	7
3.3.2.1	completed_jobs	7

3.3.2.2	cond	7
3.3.2.3	current_jobs	7
3.3.2.4	head	8
3.3.2.5	mutex	8
3.3.2.6	tail	8
3.4	pool_args Struct Reference	8
3.4.1	Detailed Description	8
3.4.2	Field Documentation	8
3.4.2.1	jobs	8
3.4.2.2	pool_max_size	8
3.5	thread_list Struct Reference	9
3.5.1	Detailed Description	9
3.5.2	Field Documentation	9
3.5.2.1	head	9
3.5.2.2	tail	9
3.6	thread_node Struct Reference	9
3.6.1	Detailed Description	10
3.6.2	Field Documentation	10
3.6.2.1	next	10
3.6.2.2	prev	10
3.6.2.3	thread	10
4	File Documentation	11
4.1	cas.h File Reference	11
4.2	content.c File Reference	11
4.2.1	Macro Definition Documentation	11
4.2.1.1	MAX_CONTENT_SZ	11
4.2.2	Function Documentation	11
4.2.2.1	content_get	12
4.2.2.2	error_resp	12
4.2.2.3	sanity_check	12
4.3	content.h File Reference	12
4.3.1	Function Documentation	12
4.3.1.1	content_get	12
4.4	main.c File Reference	12
4.4.1	Macro Definition Documentation	13
4.4.1.1	BUFFER_LENGTH	13

4.4.1.2	MAX_CONCURRENCY	13
4.4.1.3	MAX_DATA_SZ	13
4.4.2	Enumeration Type Documentation	13
4.4.2.1	server_type_t	13
4.4.3	Function Documentation	13
4.4.3.1	main	13
4.4.3.2	server_single_request	13
4.4.3.3	server_thread_per_req	13
4.4.3.4	server_thread_pool_bounded	13
4.5	server.c File Reference	13
4.5.1	Function Documentation	14
4.5.1.1	server_accept	14
4.5.1.2	server_create	14
4.6	server.h File Reference	14
4.6.1	Function Documentation	14
4.6.1.1	server_accept	14
4.6.1.2	server_create	14
4.7	simple_http.c File Reference	14
4.7.1	Macro Definition Documentation	15
4.7.1.1	MAX_DIGITS	15
4.7.2	Function Documentation	15
4.7.2.1	shttp_alloc_req	15
4.7.2.2	shttp_alloc_response_head	15
4.7.2.3	shttp_free_req	15
4.7.2.4	shttp_get_path	15
4.8	simple_http.h File Reference	15
4.8.1	Function Documentation	15
4.8.1.1	shttp_alloc_req	15
4.8.1.2	shttp_alloc_response_head	16
4.8.1.3	shttp_free_req	16
4.8.1.4	shttp_get_path	16
4.9	thread_per_request.c File Reference	16
4.9.1	Function Documentation	16
4.9.1.1	process_threads_per_request	16
4.9.1.2	thread_process	16
4.10	thread_per_request.h File Reference	17
4.10.1	Function Documentation	17

4.10.1.1	process_threads_per_request	17
4.11	thread_pool_request.c File Reference	17
4.11.1	Function Documentation	17
4.11.1.1	pool_thread_process	17
4.11.1.2	process_request_thread_pool	18
4.11.1.3	start_job	18
4.12	thread_pool_request.h File Reference	18
4.12.1	Function Documentation	18
4.12.1.1	process_request_thread_pool	18
4.13	threadlist.c File Reference	19
4.13.1	Function Documentation	19
4.13.1.1	insert_thread_list_head	19
4.13.1.2	insert_thread_list_tail	19
4.13.1.3	remove_from_thread_list	19
4.14	threadlist.h File Reference	19
4.14.1	Function Documentation	20
4.14.1.1	insert_thread_list_head	20
4.14.1.2	insert_thread_list_tail	20
4.14.1.3	remove_from_thread_list	20
4.15	threadpool.c File Reference	20
4.15.1	Function Documentation	21
4.15.1.1	insert_job_head	21
4.15.1.2	insert_job_tail	21
4.15.1.3	remove_job	21
4.16	threadpool.h File Reference	21
4.16.1	Function Documentation	21
4.16.1.1	insert_job_head	21
4.16.1.2	insert_job_tail	21
4.16.1.3	remove_job	21
4.17	util.c File Reference	22
4.17.1	Macro Definition Documentation	22
4.17.1.1	MAX_REQ_SZ	22
4.17.2	Function Documentation	22
4.17.2.1	client_process	22
4.17.2.2	newfd_create_req	22
4.17.2.3	respond_and_free_req	22
4.18	util.h File Reference	22

CONTENTS	vii
4.18.1 Function Documentation	23
4.18.1.1 client_process	23
Index	25

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

http_req	5
job	The job information	6
job_list	The list of jobs	7
pool_args	The arguments to the pool	8
thread_list	The list of threads	9
thread_node	The node holding the thread info	9

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

cas.h	11
content.c	11
content.h	12
main.c	12
server.c	13
server.h	14
simple_http.c	14
simple_http.h	15
thread_per_request.c	16
thread_per_request.h	17
thread_pool_request.c	17
thread_pool_request.h	18
threadlist.c	19
threadlist.h	19
threadpool.c	20
threadpool.h	21
util.c	22
util.h	22

Chapter 3

Data Structure Documentation

3.1 http_req Struct Reference

```
#include <simple_http.h>
```

Data Fields

- int `fd`
- char * `request`
- int `req_len`
- char * `path`
- char * `resp_head`
- char * `response`
- int `resp_hd_len`
- int `resp_len`

3.1.1 Detailed Description

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

3.1.2 Field Documentation

3.1.2.1 int http_req::fd

3.1.2.2 char* http_req::path

3.1.2.3 int http_req::req_len

3.1.2.4 char* http_req::request

3.1.2.5 int http_req::resp_hd_len

3.1.2.6 char* http_req::resp_head

3.1.2.7 `int http_req::resp_len`

3.1.2.8 `char * http_req::response`

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

- [simple_http.h](#)

3.2 job Struct Reference

The job information.

```
#include <threadpool.h>
```

Data Fields

- struct [job](#) * [prev](#)
The previous job in the list.
- struct [job](#) * [next](#)
The next job in the list.
- void (*)([function](#))(void *arg)
The function pointer for the job task.
- void * [args](#)
The arguments for the function pointer.

3.2.1 Detailed Description

The job information.

3.2.2 Field Documentation

3.2.2.1 `void* job::args`

The arguments for the function pointer.

3.2.2.2 `void*(* job::function)(void *arg)`

The function pointer for the job task.

3.2.2.3 `struct job* job::next`

The next job in the list.

3.2.2.4 struct job* job::prev

The previous job in the list.

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

- [threadpool.h](#)

3.3 job_list Struct Reference

The list of jobs.

```
#include <threadpool.h>
```

Data Fields

- pthread_mutex_t * [mutex](#)
The mutex lock for the jobs.
- pthread_cond_t * [cond](#)
The condition variable for the jobs.
- struct job * [head](#)
The head of the list.
- struct job * [tail](#)
The tail of the list.
- int [current_jobs](#)
The number of jobs currently being processed.
- int [completed_jobs](#)
The number of completed jobs.

3.3.1 Detailed Description

The list of jobs.

3.3.2 Field Documentation

3.3.2.1 int job_list::completed_jobs

The number of completed jobs.

3.3.2.2 pthread_cond_t* job_list::cond

The condition variable for the jobs.

3.3.2.3 int job_list::current_jobs

The number of jobs currently being processed.

3.3.2.4 struct job* job_list::head

The head of the list.

3.3.2.5 pthread_mutex_t* job_list::mutex

The mutex lock for the jobs.

3.3.2.6 struct job* job_list::tail

The tail of the list.

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

- [threadpool.h](#)

3.4 pool_args Struct Reference

The arguments to the pool.

```
#include <threadpool.h>
```

Data Fields

- int [pool_max_size](#)
The size of the pool.
- struct [job_list](#) * [jobs](#)
The list of jobs.

3.4.1 Detailed Description

The arguments to the pool.

3.4.2 Field Documentation

3.4.2.1 struct job_list* pool_args::jobs

The list of jobs.

3.4.2.2 int pool_args::pool_max_size

The size of the pool.

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

- [threadpool.h](#)

3.5 thread_list Struct Reference

The list of threads.

```
#include <threadlist.h>
```

Data Fields

- struct [thread_node](#) * [head](#)
The head of the list.
- struct [thread_node](#) * [tail](#)
The tail of the list.

3.5.1 Detailed Description

The list of threads.

3.5.2 Field Documentation

3.5.2.1 struct [thread_node](#)* [thread_list::head](#)

The head of the list.

3.5.2.2 struct [thread_node](#)* [thread_list::tail](#)

The tail of the list.

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

- [threadlist.h](#)

3.6 thread_node Struct Reference

The node holding the thread info.

```
#include <threadlist.h>
```

Data Fields

- struct [thread_node](#) * [prev](#)
The previous node.
- struct [thread_node](#) * [next](#)
The next node.
- pthread_t * [thread](#)
The thread.

3.6.1 Detailed Description

The node holding the thread info.

3.6.2 Field Documentation

3.6.2.1 `struct thread_node* thread_node::next`

The next node.

3.6.2.2 `struct thread_node* thread_node::prev`

The previous node.

3.6.2.3 `pthread_t* thread_node::thread`

The thread.

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

- [threadlist.h](#)

Chapter 4

File Documentation

4.1 cas.h File Reference

4.2 content.c File Reference

```
#include <unistd.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
```

Macros

- `#define MAX_CONTENT_SZ (1024*1024*10)`

Functions

- `char * error_resp (char *path, int *len)`
- `int sanity_check (char *path)`
- `char * content_get (char *path, int *content_len)`

4.2.1 Macro Definition Documentation

4.2.1.1 `#define MAX_CONTENT_SZ (1024*1024*10)`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.2.2 Function Documentation

4.2.2.1 `char* content_get (char * path, int * content_len)`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.2.2.2 `char* error_resp (char * path, int * len)`

4.2.2.3 `int sanity_check (char * path)`

4.3 `content.h` File Reference

Functions

- `char * content_get (char *path, int *content_len)`

4.3.1 Function Documentation

4.3.1.1 `char* content_get (char * path, int * content_len)`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.4 `main.c` File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <assert.h>
#include <sys/wait.h>
#include <pthread.h>
#include <util.h>
#include <server.h>
#include <thread_per_request.h>
#include <thread_pool_request.h>
#include <cas.h>
```

Macros

- `#define MAX_DATA_SZ 1024`
- `#define MAX_CONCURRENCY 4`
- `#define BUFFER_LENGTH 256`

Enumerations

- `enum server_type_t { SERVER_TYPE_ONE = 0, SERVER_TYPE_THREAD_PER_REQUEST, SERVER_TYPE_THREAD_POOL_BOUND }`

Functions

- void [server_single_request](#) (int accept_fd)
- void [server_thread_per_req](#) (int accept_fd)
- void [server_thread_pool_bounded](#) (int accept_fd)
- int [main](#) (int argc, char *argv[])

4.4.1 Macro Definition Documentation

4.4.1.1 `#define BUFFER_LENGTH 256`

4.4.1.2 `#define MAX_CONCURRENCY 4`

4.4.1.3 `#define MAX_DATA_SZ 1024`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.4.2 Enumeration Type Documentation

4.4.2.1 `enum server_type_t`

Enumerator

SERVER_TYPE_ONE
SERVER_TYPE_THREAD_PER_REQUEST
SERVER_TYPE_THREAD_POOL_BOUND

4.4.3 Function Documentation

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

4.4.3.2 `void server_single_request (int accept_fd)`

4.4.3.3 `void server_thread_per_req (int accept_fd)`

4.4.3.4 `void server_thread_pool_bounded (int accept_fd)`

4.5 server.c File Reference

```
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <errno.h>
#include <netinet/in.h>
#include <fcntl.h>
#include <arpa/inet.h>
#include <stdlib.h>
#include <unistd.h>
```

Functions

- int [server_create](#) (short int port)
- int [server_accept](#) (int fd)

4.5.1 Function Documentation

4.5.1.1 int [server_accept](#) (int *fd*)

4.5.1.2 int [server_create](#) (short int *port*)

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.6 server.h File Reference

Functions

- int [server_create](#) (short int port)
- int [server_accept](#) (int fd)

4.6.1 Function Documentation

4.6.1.1 int [server_accept](#) (int *fd*)

4.6.1.2 int [server_create](#) (short int *port*)

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.7 simple_http.c File Reference

```
#include <string.h>
#include <assert.h>
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <simple_http.h>
```

Macros

- #define [MAX_DIGITS](#) 128

Functions

- struct [http_req](#) * [shhttp_alloc_req](#) (int fd, char *request)
- void [shhttp_free_req](#) (struct [http_req](#) *r)
- int [shhttp_get_path](#) (struct [http_req](#) *r)
- int [shhttp_alloc_response_head](#) (struct [http_req](#) *r, char *data, int dlen)

4.7.1 Macro Definition Documentation

4.7.1.1 `#define MAX_DIGITS 128`

4.7.2 Function Documentation

4.7.2.1 `struct http_req* shhttp_alloc_req (int fd, char * request)`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.7.2.2 `int shhttp_alloc_response_head (struct http_req * r, char * data, int dlen)`

4.7.2.3 `void shhttp_free_req (struct http_req * r)`

4.7.2.4 `int shhttp_get_path (struct http_req * r)`

4.8 simple_http.h File Reference

Data Structures

- struct [http_req](#)

Functions

- struct [http_req](#) * [shhttp_alloc_req](#) (int fd, char *request)
- void [shhttp_free_req](#) (struct [http_req](#) *r)
- int [shhttp_get_path](#) (struct [http_req](#) *r)
- int [shhttp_alloc_response_head](#) (struct [http_req](#) *r, char *resp, int rlen)

4.8.1 Function Documentation

4.8.1.1 `struct http_req* shhttp_alloc_req (int fd, char * request)`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.8.1.2 `int shhttp_alloc_response_head (struct http_req * r, char * resp, int rlen)`

4.8.1.3 `void shhttp_free_req (struct http_req * r)`

4.8.1.4 `int shhttp_get_path (struct http_req * r)`

4.9 thread_per_request.c File Reference

```
#include <stddef.h>
#include <pthread.h>
#include <stdlib.h>
#include <server.h>
#include <util.h>
#include <assert.h>
#include <unistd.h>
#include <stdio.h>
#include "threadlist.h"
```

Functions

- `void * thread_process (void *fd)`
thread_process Function to execute on the threads; services the client's request
- `void process_threads_per_request (int max_concurrency, int accept_fd)`
process_threads_per_request Processes the request using a thread per request

4.9.1 Function Documentation

4.9.1.1 `void process_threads_per_request (int max_concurrency, int accept_fd)`

`process_threads_per_request` Processes the request using a thread per request

Parameters

<i>max_↔ concurrency</i>	The maximum number of concurrent threads
<i>accept_fd</i>	The accepted file descriptor

4.9.1.2 `void* thread_process (void * fd)`

`thread_process` Function to execute on the threads; services the client's request

Parameters

<i>fd</i>	The file descriptor of the opened request
-----------	---

Returns

0

4.10 thread_per_request.h File Reference

Functions

- void [process_threads_per_request](#) (int max_concurrency, int accept_fd)
process_threads_per_request Processes the request using a thread per request

4.10.1 Function Documentation

4.10.1.1 void [process_threads_per_request](#) (int max_concurrency, int accept_fd)

[process_threads_per_request](#) Processes the request using a thread per request

Parameters

<i>max_concurrency</i>	The maximum number of concurrent threads
<i>accept_fd</i>	The accepted file descriptor

4.11 thread_pool_request.c File Reference

```
#include <stdlib.h>
#include "threadlist.h"
#include "threadpool.h"
#include <server.h>
#include <util.h>
#include <stdio.h>
#include <assert.h>
#include <unistd.h>
#include <stddef.h>
```

Functions

- void * [start_job](#) (void *args)
The function that each worker job performs.
- void * [pool_thread_process](#) (void *fd)
thread_process Function to execute on the threads; services the client's request
- void [process_request_thread_pool](#) (int max_size, int accept_fd)
process_request_thread_pool Creates and runs a thread pool

4.11.1 Function Documentation

4.11.1.1 void* [pool_thread_process](#) (void * fd)

[thread_process](#) Function to execute on the threads; services the client's request

Parameters

<i>fd</i>	The file descriptor of the opened request
-----------	---

Returns

0

4.11.1.2 void process_request_thread_pool (int max_size, int accept_fd)

process_request_thread_pool Creates and runs a thread pool

Parameters

<i>max_size</i>	The max size of the thread pool
<i>accept_fd</i>	The accepted file descriptor of the listener

4.11.1.3 void* start_job (void * args)

The function that each worker job performs.

Parameters

<i>args</i>	The arguments for the pool
-------------	----------------------------

See also

[pool_args](#)

Returns

0 if no errors

4.12 thread_pool_request.h File Reference

Functions

- void [process_request_thread_pool](#) (int max_size, int accept_fd)
process_request_thread_pool Creates and runs a thread pool

4.12.1 Function Documentation

4.12.1.1 void process_request_thread_pool (int max_size, int accept_fd)

process_request_thread_pool Creates and runs a thread pool

Parameters

<i>max_size</i>	The max size of the thread pool
<i>accept_fd</i>	The accepted file descriptor of the listener

4.13 threadlist.c File Reference

```
#import "threadlist.h"
#include <stddef.h>
```

Functions

- void [insert_thread_list_head](#) (struct [thread_node](#) *[thread_node](#), struct [thread_list](#) *[thread_list](#))
Inserts the node to the head of the list.
- void [insert_thread_list_tail](#) (struct [thread_node](#) *[thread_node](#), struct [thread_list](#) *[thread_list](#))
Inserts the given node to the end of the list.
- void [remove_from_thread_list](#) (struct [thread_node](#) *[thread_node](#), struct [thread_list](#) *[thread_list](#))

4.13.1 Function Documentation

4.13.1.1 void [insert_thread_list_head](#) (struct [thread_node](#) * [thread_node](#), struct [thread_list](#) * [thread_list](#))

Inserts the node to the head of the list.

Parameters

thread_node	The node to be inserted
thread_list	The list of threads

4.13.1.2 void [insert_thread_list_tail](#) (struct [thread_node](#) * [thread_node](#), struct [thread_list](#) * [thread_list](#))

Inserts the given node to the end of the list.

Parameters

thread_node	The node to be inserted
thread_list	The list of threads

4.13.1.3 void [remove_from_thread_list](#) (struct [thread_node](#) * [thread_node](#), struct [thread_list](#) * [thread_list](#))

4.14 threadlist.h File Reference

```
#include <pthread.h>
```

Data Structures

- struct `thread_node`
The node holding the thread info.
- struct `thread_list`
The list of threads.

Functions

- void `insert_thread_list_head` (struct `thread_node` *, struct `thread_list` *)
Inserts the node to the head of the list.
- void `insert_thread_list_tail` (struct `thread_node` *, struct `thread_list` *)
Inserts the given node to the end of the list.
- void `remove_from_thread_list` (struct `thread_node` *, struct `thread_list` *)

4.14.1 Function Documentation

4.14.1.1 void `insert_thread_list_head` (struct `thread_node` * *thread_node*, struct `thread_list` * *thread_list*)

Inserts the node to the head of the list.

Parameters

<code>thread_node</code>	The node to be inserted
<code>thread_list</code>	The list of threads

4.14.1.2 void `insert_thread_list_tail` (struct `thread_node` * *thread_node*, struct `thread_list` * *thread_list*)

Inserts the given node to the end of the list.

Parameters

<code>thread_node</code>	The node to be inserted
<code>thread_list</code>	The list of threads

4.14.1.3 void `remove_from_thread_list` (struct `thread_node` *, struct `thread_list` *)

4.15 threadpool.c File Reference

```
#include "threadpool.h"
#include <stddef.h>
#include <stdlib.h>
#include <stdio.h>
#include <assert.h>
```

Functions

- void [insert_job_head](#) (struct [job_list](#) *[job_list](#), struct [job](#) *[job](#))
- void [insert_job_tail](#) (struct [job_list](#) *[job_list](#), struct [job](#) *[job](#))
- void [remove_job](#) (struct [job_list](#) *[job_list](#), struct [job](#) *[job](#))

4.15.1 Function Documentation

4.15.1.1 void [insert_job_head](#) (struct [job_list](#) * [job_list](#), struct [job](#) * [job](#))

4.15.1.2 void [insert_job_tail](#) (struct [job_list](#) * [job_list](#), struct [job](#) * [job](#))

4.15.1.3 void [remove_job](#) (struct [job_list](#) * [job_list](#), struct [job](#) * [job](#))

4.16 threadpool.h File Reference

```
#import <pthread.h>
```

Data Structures

- struct [job](#)
The job information.
- struct [job_list](#)
The list of jobs.
- struct [pool_args](#)
The arguments to the pool.

Functions

- void [insert_job_head](#) (struct [job_list](#) *, struct [job](#) *)
- void [insert_job_tail](#) (struct [job_list](#) *, struct [job](#) *)
- void [remove_job](#) (struct [job_list](#) *, struct [job](#) *)

4.16.1 Function Documentation

4.16.1.1 void [insert_job_head](#) (struct [job_list](#) * , struct [job](#) *)

4.16.1.2 void [insert_job_tail](#) (struct [job_list](#) * , struct [job](#) *)

4.16.1.3 void [remove_job](#) (struct [job_list](#) * , struct [job](#) *)

4.17 util.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <assert.h>
#include <server.h>
#include <simple_http.h>
#include <content.h>
```

Macros

- `#define MAX_REQ_SZ 1024`

Functions

- `struct http_req * newfd_create_req (int new_fd)`
- `void respond_and_free_req (struct http_req *r, char *response, int len)`
- `void client_process (int fd)`

4.17.1 Macro Definition Documentation

4.17.1.1 `#define MAX_REQ_SZ 1024`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.17.2 Function Documentation

4.17.2.1 `void client_process (int fd)`

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

4.17.2.2 `struct http_req* newfd_create_req (int new_fd)`

4.17.2.3 `void respond_and_free_req (struct http_req * r, char * response, int len)`

4.18 util.h File Reference

Functions

- `void client_process (int fd)`

4.18.1 Function Documentation

4.18.1.1 void client_process (int *fd*)

Redistribution of this file is permitted under the GNU General Public License v2.

Copyright 2012 by Gabriel Parmer. Author: Gabriel Parmer, gparmer@gwu.edu, 2012

Index

- args
 - job, [6](#)
- BUFFER_LENGTH
 - main.c, [13](#)
- cas.h, [11](#)
- client_process
 - util.c, [22](#)
 - util.h, [23](#)
- completed_jobs
 - job_list, [7](#)
- cond
 - job_list, [7](#)
- content.c, [11](#)
 - content_get, [11](#)
 - error_resp, [12](#)
 - MAX_CONTENT_SZ, [11](#)
 - sanity_check, [12](#)
- content.h, [12](#)
 - content_get, [12](#)
- content_get
 - content.c, [11](#)
 - content.h, [12](#)
- current_jobs
 - job_list, [7](#)
- error_resp
 - content.c, [12](#)
- fd
 - http_req, [5](#)
- function
 - job, [6](#)
- head
 - job_list, [7](#)
 - thread_list, [9](#)
- http_req, [5](#)
 - fd, [5](#)
 - path, [5](#)
 - req_len, [5](#)
 - request, [5](#)
 - resp_hd_len, [5](#)
 - resp_head, [5](#)
 - resp_len, [5](#)
 - response, [6](#)
- insert_job_head
 - threadpool.c, [21](#)
 - threadpool.h, [21](#)
- insert_job_tail
 - threadpool.c, [21](#)
 - threadpool.h, [21](#)
- insert_thread_list_head
 - threadlist.c, [19](#)
 - threadlist.h, [20](#)
- insert_thread_list_tail
 - threadlist.c, [19](#)
 - threadlist.h, [20](#)
- job, [6](#)
 - args, [6](#)
 - function, [6](#)
 - next, [6](#)
 - prev, [6](#)
- job_list, [7](#)
 - completed_jobs, [7](#)
 - cond, [7](#)
 - current_jobs, [7](#)
 - head, [7](#)
 - mutex, [8](#)
 - tail, [8](#)
- jobs
 - pool_args, [8](#)
- MAX_CONCURRENCY
 - main.c, [13](#)
- MAX_CONTENT_SZ
 - content.c, [11](#)
- MAX_DATA_SZ
 - main.c, [13](#)
- MAX_DIGITS
 - simple_http.c, [15](#)
- MAX_REQ_SZ
 - util.c, [22](#)
- main
 - main.c, [13](#)
- main.c, [12](#)
 - BUFFER_LENGTH, [13](#)
 - MAX_CONCURRENCY, [13](#)
 - MAX_DATA_SZ, [13](#)

- main, 13
- SERVER_TYPE_ONE, 13
- SERVER_TYPE_THREAD_PER_REQUEST, 13
- SERVER_TYPE_THREAD_POOL_BOUND, 13
- server_single_request, 13
- server_thread_per_req, 13
- server_thread_pool_bounded, 13
- server_type_t, 13
- mutex
 - job_list, 8
- newfd_create_req
 - util.c, 22
- next
 - job, 6
 - thread_node, 10
- path
 - http_req, 5
- pool_args, 8
 - jobs, 8
 - pool_max_size, 8
- pool_max_size
 - pool_args, 8
- pool_thread_process
 - thread_pool_request.c, 17
- prev
 - job, 6
 - thread_node, 10
- process_request_thread_pool
 - thread_pool_request.c, 18
 - thread_pool_request.h, 18
- process_threads_per_request
 - thread_per_request.c, 16
 - thread_per_request.h, 17
- remove_from_thread_list
 - threadlist.c, 19
 - threadlist.h, 20
- remove_job
 - threadpool.c, 21
 - threadpool.h, 21
- req_len
 - http_req, 5
- request
 - http_req, 5
- resp_hd_len
 - http_req, 5
- resp_head
 - http_req, 5
- resp_len
 - http_req, 5
- respond_and_free_req
 - util.c, 22
- response
 - http_req, 6
- SERVER_TYPE_ONE
 - main.c, 13
- SERVER_TYPE_THREAD_PER_REQUEST
 - main.c, 13
- SERVER_TYPE_THREAD_POOL_BOUND
 - main.c, 13
- sanity_check
 - content.c, 12
- server.c, 13
 - server_accept, 14
 - server_create, 14
- server.h, 14
 - server_accept, 14
 - server_create, 14
- server_accept
 - server.c, 14
 - server.h, 14
- server_create
 - server.c, 14
 - server.h, 14
- server_single_request
 - main.c, 13
- server_thread_per_req
 - main.c, 13
- server_thread_pool_bounded
 - main.c, 13
- server_type_t
 - main.c, 13
- shttp_alloc_req
 - simple_http.c, 15
 - simple_http.h, 15
- shttp_alloc_response_head
 - simple_http.c, 15
 - simple_http.h, 15
- shttp_free_req
 - simple_http.c, 15
 - simple_http.h, 16
- shttp_get_path
 - simple_http.c, 15
 - simple_http.h, 16
- simple_http.c, 14
 - MAX_DIGITS, 15
 - shttp_alloc_req, 15
 - shttp_alloc_response_head, 15
 - shttp_free_req, 15
 - shttp_get_path, 15
- simple_http.h, 15
 - shttp_alloc_req, 15
 - shttp_alloc_response_head, 15
 - shttp_free_req, 16
 - shttp_get_path, 16
- start_job

- thread_pool_request.c, [18](#)
- tail
 - job_list, [8](#)
 - thread_list, [9](#)
- thread
 - thread_node, [10](#)
- thread_list, [9](#)
 - head, [9](#)
 - tail, [9](#)
- thread_node, [9](#)
 - next, [10](#)
 - prev, [10](#)
 - thread, [10](#)
- thread_per_request.c, [16](#)
 - process_threads_per_request, [16](#)
 - thread_process, [16](#)
- thread_per_request.h, [17](#)
 - process_threads_per_request, [17](#)
- thread_pool_request.c, [17](#)
 - pool_thread_process, [17](#)
 - process_request_thread_pool, [18](#)
 - start_job, [18](#)
- thread_pool_request.h, [18](#)
 - process_request_thread_pool, [18](#)
- thread_process
 - thread_per_request.c, [16](#)
- threadlist.c, [19](#)
 - insert_thread_list_head, [19](#)
 - insert_thread_list_tail, [19](#)
 - remove_from_thread_list, [19](#)
- threadlist.h, [19](#)
 - insert_thread_list_head, [20](#)
 - insert_thread_list_tail, [20](#)
 - remove_from_thread_list, [20](#)
- threadpool.c, [20](#)
 - insert_job_head, [21](#)
 - insert_job_tail, [21](#)
 - remove_job, [21](#)
- threadpool.h, [21](#)
 - insert_job_head, [21](#)
 - insert_job_tail, [21](#)
 - remove_job, [21](#)
- util.c, [22](#)
 - client_process, [22](#)
 - MAX_REQ_SZ, [22](#)
 - newfd_create_req, [22](#)
 - respond_and_free_req, [22](#)
- util.h, [22](#)
 - client_process, [23](#)