ws

1.0

Generated by Doxygen 1.8.9.1

Mon Mar 2 2015 00:03:33

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

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)

 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_TY↩ PE_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

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*)

 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*)

 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 * shttp_alloc_req (int fd, char *request)
- void shttp_free_req (struct http_req *r)
- int shttp_get_path (struct http_req *r)
- int shttp_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* shttp_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 shttp_alloc_response_head (struct http_req * *r,* char * *data,* int *dlen*)

4.7.2.3 void shttp_free_req (struct http_req * *r*)

4.7.2.4 int shttp_get_path (struct http_req * *r*)

4.8 simple_http.h File Reference

Data Structures

- struct http_req

Functions

- struct http_req * shttp_alloc_req (int fd, char *request)
- void shttp_free_req (struct http_req *r)
- int shttp_get_path (struct http_req *r)
- int shttp_alloc_response_head (struct http_req *r, char *resp, int rlen)

4.8.1 Function Documentation

4.8.1.1 struct http_req* shttp_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 shttp_alloc_response_head (struct http_req * r, char * resp, int rlen)

4.8.1.3 void shttp_free_req (struct http_req * r)

4.8.1.4 int shttp_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

max_concurrency	The maximum number of concurrent threads
accept_fd	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

fd	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

max_concurrency	The maximum number of concurrent threads
accept_fd	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

fd	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

max_size	The max size of the thread pool
accept_fd	The accepted file descriptor of the listener

4.11.1.3 void* start_job (void * *args*)

The function that each worker job performs.

Parameters

args	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

max_size	The max size of the thread pool
accept_fd	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

thread_node	The node to be inserted
thread_list	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

thread_node	The node to be inserted
thread_list	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*)

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

Index