

gds

Generated by Doxygen 1.8.1.2

Sat Nov 8 2014 15:28:03



# Contents

<b>1</b>	<b>Data Structure Index</b>	<b>1</b>
1.1	Data Structures . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Data Structure Documentation</b>	<b>5</b>
3.1	gdt_generic_datatype Struct Reference . . . . .	5
3.2	hms Struct Reference . . . . .	5
3.3	list Struct Reference . . . . .	6
3.4	list_node Struct Reference . . . . .	7
3.5	queue Struct Reference . . . . .	7
3.6	stack Struct Reference . . . . .	8
3.7	vector Struct Reference . . . . .	9
<b>4</b>	<b>File Documentation</b>	<b>11</b>
4.1	tests/test_logging.c File Reference . . . . .	11
4.1.1	Detailed Description . . . . .	12
4.1.2	Function Documentation . . . . .	12
4.1.2.1	tests_get_failures . . . . .	12
4.1.2.2	tests_get_successes . . . . .	12
4.1.2.3	tests_get_total_tests . . . . .	12
4.1.2.4	tests_log_test . . . . .	12
4.2	tests/test_logging.h File Reference . . . . .	12
4.2.1	Detailed Description . . . . .	13
4.2.2	Function Documentation . . . . .	14
4.2.2.1	tests_get_failures . . . . .	14
4.2.2.2	tests_get_successes . . . . .	14
4.2.2.3	tests_get_total_tests . . . . .	14
4.2.2.4	tests_log_test . . . . .	14



# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">gdt_generic_datatype</a>	5
<a href="#">hms</a>	5
<a href="#">list</a>	6
<a href="#">list_node</a>	7
<a href="#">queue</a>	7
<a href="#">stack</a>	8
<a href="#">vector</a>	9



## Chapter 2

# File Index

### 2.1 File List

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

include/private/ <b>gds_common.h</b>	??
include/private/ <b>gdt.h</b>	??
include/public/ <b>gds_public_types.h</b>	??
include/public/ <b>gds_util.h</b>	??
include/public/ <b>list.h</b>	??
include/public/ <b>queue.h</b>	??
include/public/ <b>stack.h</b>	??
include/public/ <b>vector.h</b>	??
tests/ <b>test_list.h</b>	??
tests/ <a href="#">test_logging.c</a>	
Implementation of unit test logging functionality	11
tests/ <a href="#">test_logging.h</a>	
Interface to unit test logging functionality	12
tests/ <b>test_queue.h</b>	??
tests/ <b>test_stack.h</b>	??
tests/ <b>test_vector.h</b>	??





## Chapter 3

# Data Structure Documentation

### 3.1 gdt\_generic\_datatype Struct Reference

#### Data Fields

- enum gds\_datatype **type**
- gds\_cfunc **compfunc**
- union {
  - char **c**
  - unsigned char **uc**
  - signed char **sc**
  - int **i**
  - unsigned int **ui**
  - long **l**
  - unsigned long **ul**
  - long long int **ll**
  - unsigned long long int **ull**
  - size\_t **st**
  - double **d**
  - char \* **pc**
  - void \* **p**
- } **data**

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

- include/private/gdt.h

### 3.2 hms Struct Reference

#### Data Fields

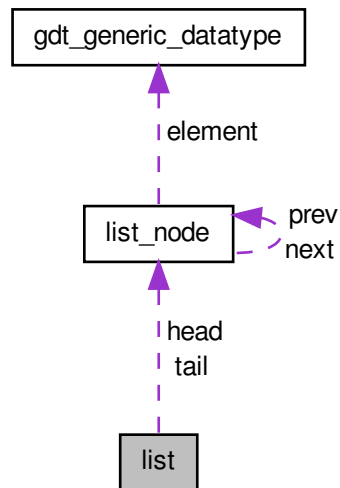
- int **hour**
- int **minute**
- int **second**

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

- tests/test\_list.c
- tests/test\_vector.c

### 3.3 list Struct Reference

Collaboration diagram for list:



#### Data Fields

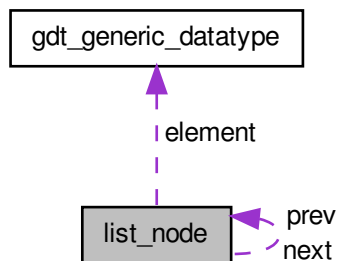
- `size_t` **length**
- `enum gds_datatype` **type**
- `gds_cfunc` **compfunc**
- `struct list_node *` **head**
- `struct list_node *` **tail**
- `bool` **free\_on\_destroy**
- `bool` **exit\_on\_error**

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

- `src/list.c`

## 3.4 list\_node Struct Reference

Collaboration diagram for list\_node:



### Data Fields

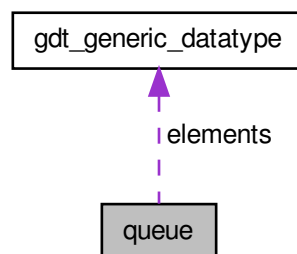
- struct [gdt\\_generic\\_datatype](#) **element**
- struct [list\\_node](#) \* **prev**
- struct [list\\_node](#) \* **next**

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

- src/list.c

## 3.5 queue Struct Reference

Collaboration diagram for queue:



### Data Fields

- **size\_t front**

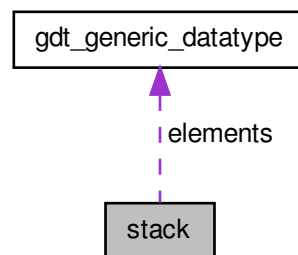
- `size_t` **back**
- `size_t` **capacity**
- `size_t` **size**
- `enum gds_datatype` **type**
- `struct gdt_generic_datatype *` **elements**
- `bool` **resizable**
- `bool` **free\_on\_destroy**
- `bool` **exit\_on\_error**

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

- `src/queue.c`

## 3.6 stack Struct Reference

Collaboration diagram for stack:



### Data Fields

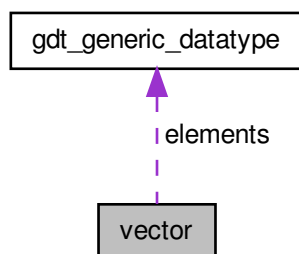
- `size_t` **top**
- `size_t` **capacity**
- `enum gds_datatype` **type**
- `struct gdt_generic_datatype *` **elements**
- `bool` **resizable**
- `bool` **free\_on\_destroy**
- `bool` **exit\_on\_error**

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

- `src/stack.c`

## 3.7 vector Struct Reference

Collaboration diagram for vector:



### Data Fields

- `size_t` **length**
- `size_t` **capacity**
- `enum gds_datatype` **type**
- `struct gdt_generic_datatype *` **elements**
- `int(* compfunc )(const void *, const void *)`
- `bool` **free\_on\_destroy**
- `bool` **exit\_on\_error**

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

- `src/vector.c`



## Chapter 4

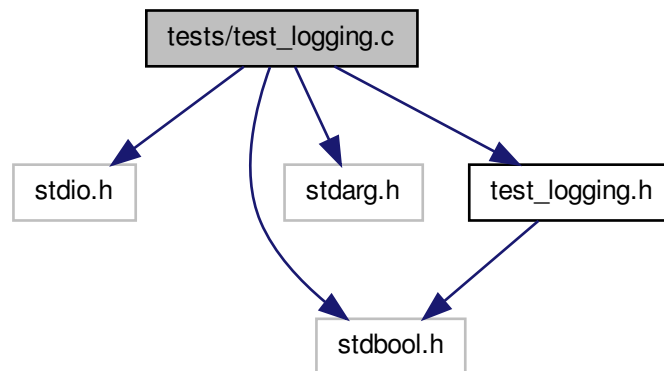
# File Documentation

### 4.1 tests/test\_logging.c File Reference

Implementation of unit test logging functionality.

```
#include <stdio.h>
#include <stdbool.h>
#include <stdarg.h>
#include "test_logging.h"
```

Include dependency graph for test\_logging.c:



### Functions

- void `tests_log_test` (const bool success, const char \*fmt,...)  
*Logs the result of a unit test.*
- int `tests_get_total_tests` (void)  
*Returns the total number of tests run.*
- int `tests_get_successes` (void)  
*Returns the total number of successful tests.*
- int `tests_get_failures` (void)  
*Returns the total number of failed tests.*

### 4.1.1 Detailed Description

Implementation of unit test logging functionality.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

### 4.1.2 Function Documentation

#### 4.1.2.1 `int tests_get_failures ( void )`

Returns the total number of failed tests.

#### Returns

The total number of failed tests.

#### 4.1.2.2 `int tests_get_successes ( void )`

Returns the total number of successful tests.

#### Returns

The total number of successful tests.

#### 4.1.2.3 `int tests_get_total_tests ( void )`

Returns the total number of tests run.

#### Returns

The total number of tests run.

#### 4.1.2.4 `void tests_log_test ( const bool success, const char * fmt, ... )`

Logs the result of a unit test.

#### Parameters

<i>success</i>	true if the test succeeded, false otherwise.
<i>fmt</i>	Format string for failure message.
...	Arguements to format string.

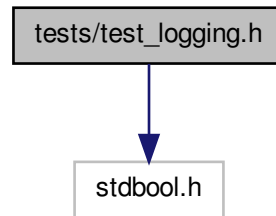
## 4.2 tests/test\_logging.h File Reference

Interface to unit test logging functionality.

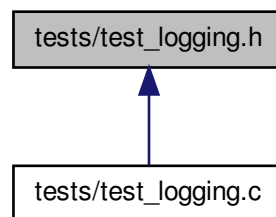


```
#include <stdbool.h>
```

Include dependency graph for test\_logging.h:



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



## Functions

- void [tests\\_log\\_test](#) (const bool success, const char \*fmt,...)  
*Logs the result of a unit test.*
- int [tests\\_get\\_total\\_tests](#) (void)  
*Returns the total number of tests run.*
- int [tests\\_get\\_successes](#) (void)  
*Returns the total number of successful tests.*
- int [tests\\_get\\_failures](#) (void)  
*Returns the total number of failed tests.*

### 4.2.1 Detailed Description

Interface to unit test logging functionality.

#### Author

Paul Griffiths

## Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 4.2.2 Function Documentation

### 4.2.2.1 `int tests_get_failures ( void )`

Returns the total number of failed tests.

#### Returns

The total number of failed tests.

### 4.2.2.2 `int tests_get_successes ( void )`

Returns the total number of successful tests.

#### Returns

The total number of successful tests.

### 4.2.2.3 `int tests_get_total_tests ( void )`

Returns the total number of tests run.

#### Returns

The total number of tests run.

### 4.2.2.4 `void tests_log_test ( const bool success, const char * fmt, ... )`

Logs the result of a unit test.

#### Parameters

<i>success</i>	<code>true</code> if the test succeeded, <code>false</code> otherwise.
<i>fmt</i>	Format string for failure message.
<i>...</i>	Arguements to format string.

# Index

gdt\_generic\_datatype, [5](#)

hms, [5](#)

list, [6](#)

list\_node, [7](#)

queue, [7](#)

stack, [8](#)

test\_logging.c

tests\_get\_failures, [12](#)

tests\_get\_successes, [12](#)

tests\_get\_total\_tests, [12](#)

tests\_log\_test, [12](#)

test\_logging.h

tests\_get\_failures, [14](#)

tests\_get\_successes, [14](#)

tests\_get\_total\_tests, [14](#)

tests\_log\_test, [14](#)

tests/test\_logging.c, [11](#)

tests/test\_logging.h, [12](#)

tests\_get\_failures

test\_logging.c, [12](#)

test\_logging.h, [14](#)

tests\_get\_successes

test\_logging.c, [12](#)

test\_logging.h, [14](#)

tests\_get\_total\_tests

test\_logging.c, [12](#)

test\_logging.h, [14](#)

tests\_log\_test

test\_logging.c, [12](#)

test\_logging.h, [14](#)

vector, [9](#)