timer_game_engine

Generated by Doxygen 1.8.1.2

Thu Nov 27 2014 15:02:11

Contents

Chapter 1

Timer Game Engine

Timer Game Engine is a curses library for making games where events occur on a regular periodic basis, while allowing for input at any time.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Data Structure Documentation

4.1 gw Struct Reference

Structure for main game window.

Data Fields

- WINDOW ∗ window
- struct winsize ws
- int old_cursor
- bool initialized

4.1.1 Detailed Description

Structure for main game window.

4.1.2 Field Documentation

4.1.2.1 bool gw::initialized

true if initialized, false otherwise

4.1.2.2 int gw::old_cursor

To store the old cursor

4.1.2.3 WINDOW∗ gw::window

Pointer to main curses window

4.1.2.4 struct winsize gw::ws

Contains dimensions of terminal

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

- src/tge_curses_routines.c

4.2 tge_parameters Struct Reference

Structure for containing game parameters.

`#include <tge_main_game.h>`

Data Fields

- void(∗ setup_function)(void)
- void(∗ draw_function)(void)
- void(∗ input_function)(int)
- void(∗ teardown_function)(int)
- double timer_interval

4.2.1 Detailed Description

Structure for containing game parameters.

4.2.2 Field Documentation

4.2.2.1 void(∗ tge_parameters::draw_function)(void)

Draw function

4.2.2.2 void(∗ tge_parameters::input_function)(int)

Input handling function

4.2.2.3 void(∗ tge_parameters::setup_function)(void)

Setup/initialization function

4.2.2.4 void(∗ tge_parameters::teardown_function)(int)

Cleanup function

4.2.2.5 double tge_parameters::timer_interval

Timer interval, in seconds

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

- include/tge_main_game.h

Chapter 5

File Documentation

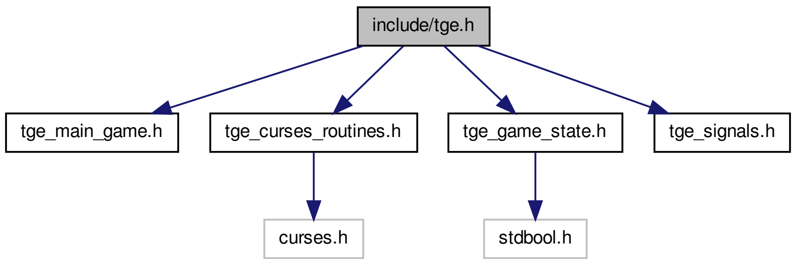
5.1 docs/tge.dox File Reference

5.2 include/tge.h File Reference

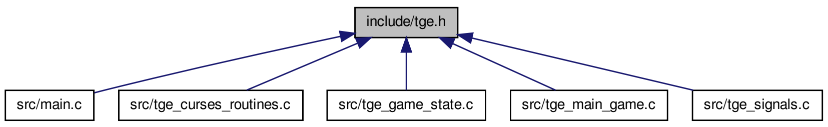
Aggregated header for curses timer game engine functions.

```
#include "tge_main_game.h"
#include "tge_curses_routines.h"
#include "tge_game_state.h"
#include "tge_signals.h"
```

Include dependency graph for tge.h:



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



5.2.1 Detailed Description

Aggregated header for curses timer game engine functions. Aggregated header for curses timer game engine functions.

A "timer game" is here defined as a game with two properties:

- the game and display regularly updates at a specified time interval; and
- the game can receive keyboard input at any time.

The rationale for providing this framework is twofold. Firstly, to encapsulate some of the overhead of setting up the game, including initializing curses, setting up signal handlers, and controlling the overall flow of the game. Secondly, to solve the programming problem of asynchronously updating the game while waiting on user input.

An initial naive implementation may be to implement the game timer using `SIGALRM`, and update the screen in the signal handler, while simply blocking on user input in the main thread. However, the curses functions are not re-entrant, and neither are many standard library functions, so this approach is not reliable. Similar objections would apply to a threads-based approach where one thread blocks on input, and a second thread updates the screen.

A second naive implementation would be to simply poll for input without blocking, which would be an unnecessary waste of processor time.

The solution here is to wait for input with `select()` to avoid blocking on any input. The game timer is implemented with `SIGALRM`, and the signal handler simply sets an "updated needed" variable and has the automatic side-effect of interrupting `select()` (note: portable curses programs cannot make any assumptions about whether handled signals will interrupt any curses input functions, so `select()` is necessary).

The library implements a game loop which begins by checking if a screen update is necessary. If it is, the screen is updated, and the loop re-entered. If it is not, the loop waits on `select()`. If no input is entered, `select()` will interrupt on handling `SIGALRM` and continue to the next iteration of the loop, and the screen will be updated again. If input is entered, `select()` will return and the input can be obtained without blocking.

Author

Paul Griffiths

Copyright

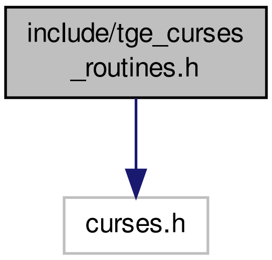
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. `http://www.gnu.org/licenses/`

5.3 include/tge_curses_routines.h File Reference

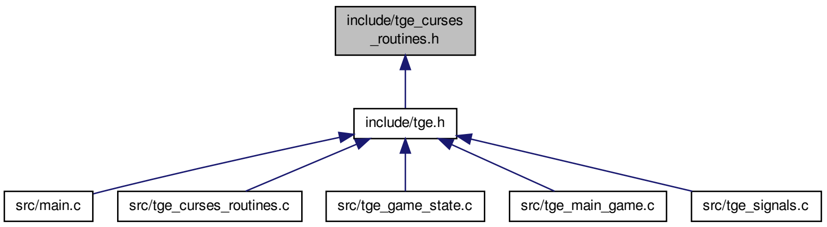
Interface to TGE general curses functions.

```
#include <curses.h>
```

Include dependency graph for tge_curses_routines.h:



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



Functions

- void tge_initialize_screen (void)
- void tge_free_screen (void)
- WINDOW ∗ tge_main_window (void)

  Returns a pointer to the main curses window.
- int tge_get_character (void)

  Gets a character input by the player.
- int tge_term_rows (void)

  Returns the number of rows in the terminal.
- int tge_term_cols (void)

  Returns the number of columns in the terminal.

5.3.1 Detailed Description

Interface to TGE general curses functions. Interface to TGE general curses functions.

Author

Paul Griffiths

Copyright

5.3.2 Function Documentation

5.3.2.1 void tge_free_screen (void)

Frees and destroys the game screen.

5.3.2.2 int tge_get_character (void)

Gets a character input by the player.

Gets a character input by the player. This function will not block if no input is ready.

Returns

The character input by the player, or -1 if no character was ready.

5.3.2.3 void tge_initialize_screen (void)

Initializes the game screen.

5.3.2.4 WINDOW∗ tge_main_window (void)

Returns a pointer to the main curses window.

Returns a pointer to the main curses window.

Returns

A pointer to the main curses window.

5.3.2.5 int tge_term_cols (void)

Returns the number of columns in the terminal.

Returns the number of columns in the terminal.

Returns

The number of columns in the terminal.

5.3.2.6 int tge_term_rows (void)

Returns the number of rows in the terminal.

Returns the number of rows in the terminal.

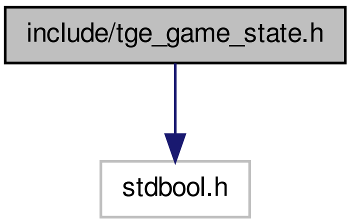
Returns

The number of rows in the terminal.

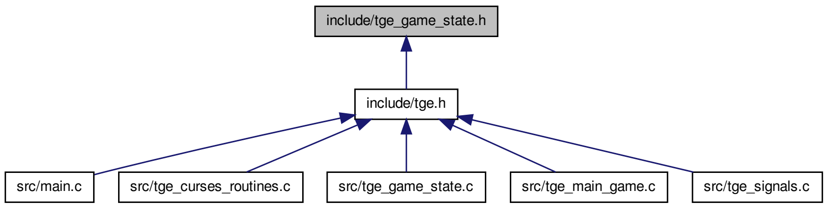
5.4 include/tge_game_state.h File Reference

Interface to TGE game state functions.

`#include <stdbool.h>`
Include dependency graph for tge_game_state.h:



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



Functions

- void tge_start_game (void)
- bool tge_game_started (void)

    Tests if the timer game has started.
- void tge_end_game (const int status)

    Ends the timer game.
- bool tge_game_ended (void)

    Tests if the timer game has ended.
- int tge_end_status (void)

    Returns the timer game exit status.
- void tge_set_needs_refresh (const bool status)

    Notifies the library that the game needs updating.
- bool tge_needs_refresh (void)

    Checks if the game needs updating.

5.4.1 Detailed Description

Interface to TGE game state functions. Interface to TGE game state functions.

Author

Paul Griffiths

Copyright

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

5.4.2 Function Documentation

5.4.2.1 void tge_end_game (const int *status*)

Ends the timer game.

Ends the timer game.

Parameters

status	The exit status of the game.

5.4.2.2 int tge_end_status (void)

Returns the timer game exit status.

Returns the timer game exit status. The meaning of this value is defined by the application using the TGE library.

Returns

The timer game exit status.

5.4.2.3 bool tge_game_ended (void)

Tests if the timer game has ended.

Tests if the timer game has ended.

Returns

`true` if the timer game has ended, `false` otherwise.

5.4.2.4 bool tge_game_started (void)

Tests if the timer game has started.

Tests if the timer game has started.

Returns

`true` if the timer game has started, `false` otherwise.

5.4.2.5 bool tge_needs_refresh (void)

Checks if the game needs updating.

Checks if the game needs updating. This function returns the state set by `tge_set_needs_refresh()`.

Returns

`true` is the game needs updating, `false` otherwise.

5.4.2.6 void tge_set_needs_refresh (const bool *status*)

Notifies the library that the game needs updating.

Notifies the library that the game needs updating. This is typically called by the `SIGALRM` signal handler in response to the game timer, but could be called by the application.

Parameters

status	`true` to turn on the "needs updating" flag, `false` to turn it off.

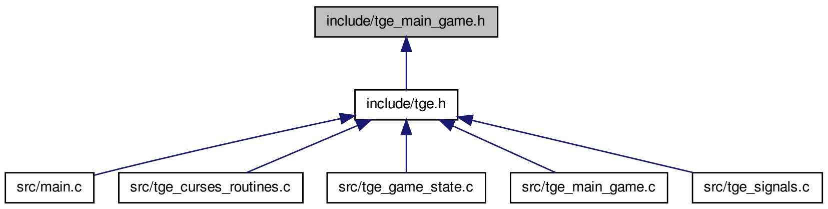
5.4.2.7 void tge_start_game (void)

Starts the timer game.

5.5 include/tge_main_game.h File Reference

Interface to curses timer game engine main game functions.

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



Data Structures

- struct tge_parameters

  Structure for containing game parameters.

Functions

- int tge_begin_game (const struct tge_parameters ∗parameters)

  Begins the timer game.
- void tge_end_game (const int status)

  Ends the timer game.

5.5.1 Detailed Description

Interface to curses timer game engine main game functions. Interface to curses timer game engine main game functions.

Author

Paul Griffiths

Copyright

5.5.2 Function Documentation

5.5.2.1 int tge_begin_game (const struct **tge_parameters** * *parameters*)

Begins the timer game.

Begins the timer game.

Parameters

parameters	A pointer to a `struct tge_parameters` object containing the desired game parameters.

Returns

The exit status of the game.

5.5.2.2 void tge_end_game (const int *status*)

Ends the timer game.

Ends the timer game.

Parameters

status	The exit status for the game.

Ends the timer game.

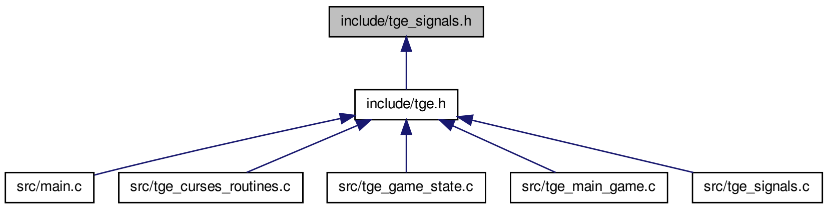
Parameters

status	The exit status of the game.

5.6 include/tge_signals.h File Reference

Interface to TGE signals functions.

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



Functions

- void tge_set_signal_handlers (void)
- void tge_timer_start (const double start, const double interval)

  Starts the game timer.
- void tge_timer_stop (void)

5.6.1 Detailed Description

Interface to TGE signals functions. Interface to TGE signals functions.

Author

Paul Griffiths

Copyright

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

5.6.2 Function Documentation

5.6.2.1 void tge_set_signal_handlers (void)

Registers the signal handlers.

5.6.2.2 void tge_timer_start (const double *start,* const double *interval*)

Starts the game timer.

Starts the game timer.

Parameters

start	The time until the first alarm, in seconds.
interval	The time interval between subsequent alarms, in seconds.

5.6.2.3 void tge_timer_stop (void)

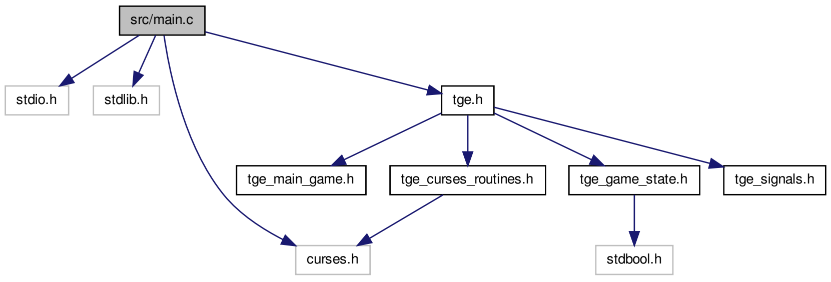
Stops the game timer.

5.7 src/main.c File Reference

Main function for timer_game_engine.

```
#include <stdio.h>
#include <stdlib.h>
#include <curses.h>
#include "tge.h"
```

Include dependency graph for main.c:



Functions

- void game_init (void)

  Initializes the game.
- void game_destroy (int exit_status)

  Cleans up after the game.
- void game_draw (void)

  Draws the game screen.
- void process_input (const int ch)

  Processes the game input.
- int main (void)

  Main function.

Variables

- static const int game_normal_exit = 0
- static const int game_early_exit = 1

5.7.1 Detailed Description

Main function for timer_game_engine. Main function for timer_game_engine.

Author

Paul Griffiths

Copyright

5.7.2 Function Documentation

5.7.2.1 void game_destroy (int *exit_status*)

Cleans up after the game.

This function is called back by the TGE engine.

5.7.2.2 void game_draw (void)

Draws the game screen.

This function is called back by the TGE engine.

5.7.2.3 void game_init (void)

Initializes the game.

This function is called back by the TGE engine.

5.7.2.4 int main (void)

Main function.

Returns

Exit status.

5.7.2.5 void process_input (const int *ch*)

Processes the game input.

This function is called back by the TGE engine.

5.7.3 Variable Documentation

5.7.3.1 const int game_early_exit = 1 [static]

File scope constant for early exit status

5.7.3.2 const int game_normal_exit = 0 [static]

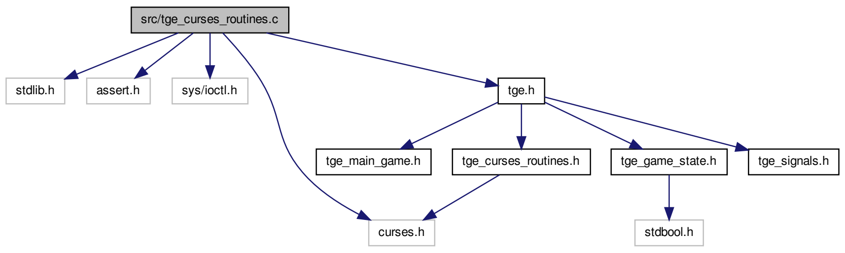
File scope constant for normal exit status

5.8 src/tge_curses_routines.c File Reference

Implementation of TGE general curses functions.

```
#include <stdlib.h>
#include <assert.h>
#include <sys/ioctl.h>
#include <curses.h>
#include "tge.h"
```

Include dependency graph for tge_curses_routines.c:



Data Structures

- struct gw

    Structure for main game window.

Macros

- #define _POSIX_C_SOURCE 200809L

Functions

- void tge_initialize_screen (void)
- void tge_free_screen (void)
- WINDOW ∗ tge_main_window (void)

    Returns a pointer to the main curses window.
- int tge_get_character (void)

    Gets a character input by the player.
- int tge_term_rows (void)

    Returns the number of rows in the terminal.
- int tge_term_cols (void)

    Returns the number of columns in the terminal.

Variables

- static struct gw game_window = {NULL, {0, 0, 0, 0}, 0, false}

5.8.1 Detailed Description

Implementation of TGE general curses functions. Implementation of TGE general curses functions.

Author

Paul Griffiths

Copyright

5.8.2 Macro Definition Documentation

5.8.2.1 #define _POSIX_C_SOURCE 200809L

POSIX feature test macro

5.8.3 Function Documentation

5.8.3.1 void tge_free_screen (void)

Frees and destroys the game screen.

5.8.3.2 int tge_get_character (void)

Gets a character input by the player.

Gets a character input by the player. This function will not block if no input is ready.

Returns

The character input by the player, or -1 if no character was ready.

5.8.3.3 void tge_initialize_screen (void)

Initializes the game screen.

5.8.3.4 WINDOW∗ tge_main_window (void)

Returns a pointer to the main curses window.

Returns a pointer to the main curses window.

Returns

A pointer to the main curses window.

5.8.3.5 int tge_term_cols (void)

Returns the number of columns in the terminal.

Returns the number of columns in the terminal.

Returns

The number of columns in the terminal.

5.8.3.6 int tge_term_rows (void)

Returns the number of rows in the terminal.

Returns the number of rows in the terminal.

Returns

The number of rows in the terminal.

5.8.4 Variable Documentation

5.8.4.1 struct gw game_window = {NULL, {0, 0, 0, 0}, 0, false} [static]

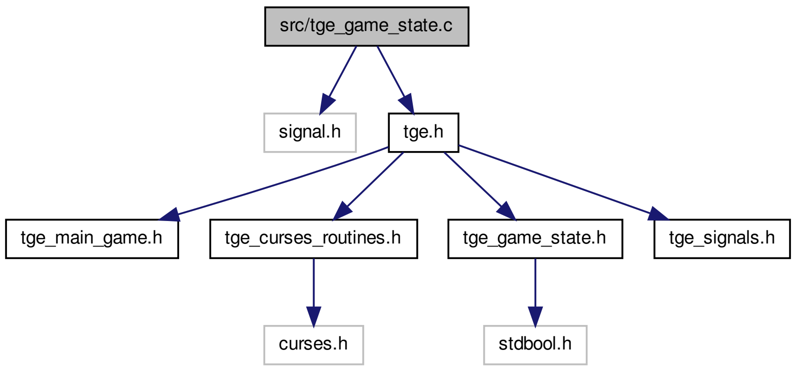
File scope variable to hold main game window

5.9 src/tge_game_state.c File Reference

Implementation of TGE game state functions.

```
#include <signal.h>
#include "tge.h"
```

Include dependency graph for tge_game_state.c:



Macros

- #define _POSIX_C_SOURCE 200809L

Enumerations

- enum game_state { TGE_GAME_STATE_NOTSTARTED, TGE_GAME_STATE_RUNNING, TGE_GAME_-STATE_ENDED }

    Enumeration constants for game state.

Functions

- void tge_start_game (void)
- bool tge_game_started (void)

    Tests if the timer game has started.
- void tge_end_game (const int status)

    Ends the timer game.
- bool tge_game_ended (void)

    Tests if the timer game has ended.
- int tge_end_status (void)

    Returns the timer game exit status.

- void tge_set_needs_refresh (const bool status)

  Notifies the library that the game needs updating.
- bool tge_needs_refresh (void)

  Checks if the game needs updating.

Variables

- static volatile sig_atomic_t tge_end_status_var = 0
- static volatile sig_atomic_t game_state = TGE_GAME_STATE_NOTSTARTED
- static volatile sig_atomic_t refresh_flag = true

5.9.1 Detailed Description

Implementation of TGE game state functions. Implementation of TGE game state functions.

Author

Paul Griffiths

Copyright

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

5.9.2 Macro Definition Documentation

5.9.2.1 #define _POSIX_C_SOURCE 200809L

POSIX feature test macro

5.9.3 Enumeration Type Documentation

5.9.3.1 enum **game_state**

Enumeration constants for game state.

Enumerator:

TGE_GAME_STATE_NOTSTARTED	Game has not started
TGE_GAME_STATE_RUNNING	Game has started
TGE_GAME_STATE_ENDED	Game has ended

5.9.4 Function Documentation

5.9.4.1 void tge_end_game (const int *status*)

Ends the timer game.

Ends the timer game.

Parameters

status	The exit status of the game.

5.9.4.2 int tge_end_status (void)

Returns the timer game exit status.

Returns the timer game exit status. The meaning of this value is defined by the application using the TGE library.

Returns

The timer game exit status.

5.9.4.3 bool tge_game_ended (void)

Tests if the timer game has ended.

Tests if the timer game has ended.

Returns

`true` if the timer game has ended, `false` otherwise.

5.9.4.4 bool tge_game_started (void)

Tests if the timer game has started.

Tests if the timer game has started.

Returns

`true` if the timer game has started, `false` otherwise.

5.9.4.5 bool tge_needs_refresh (void)

Checks if the game needs updating.

Checks if the game needs updating. This function returns the state set by `tge_set_needs_refresh()`.

Returns

`true` is the game needs updating, `false` otherwise.

5.9.4.6 void tge_set_needs_refresh (const bool *status*)

Notifies the library that the game needs updating.

Notifies the library that the game needs updating. This is typically called by the `SIGALRM` signal handler in response to the game timer, but could be called by the application.

Parameters

status	`true` to turn on the "needs updating" flag, `false` to turn it off.

5.9.4.7 void tge_start_game (void)

Starts the timer game.

5.9.5 Variable Documentation

5.9.5.1 volatile sig_atomic_t game_state = TGE_GAME_STATE_NOTSTARTED [static]

File scope variable for current game state

5.9.5.2 volatile sig_atomic_t refresh_flag = true [static]

File scope variable for "needs updating" flag

5.9.5.3 volatile sig_atomic_t tge_end_status_var = 0 [static]

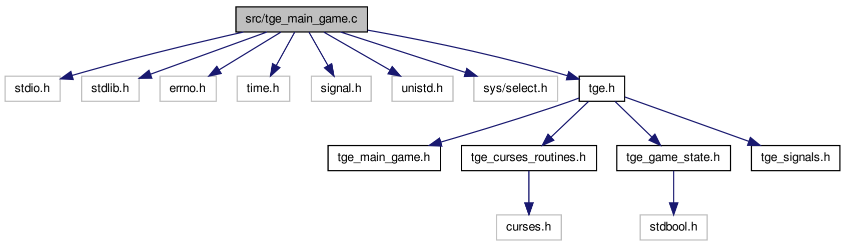
File scope variable for current exit status

5.10 src/tge_main_game.c File Reference

Implementation of TGE main game functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <time.h>
#include <signal.h>
#include <unistd.h>
#include <sys/select.h>
#include "tge.h"
```

Include dependency graph for tge_main_game.c:



Macros

- #define _POSIX_C_SOURCE 200809L

Functions

- static void tge_game_loop (void)
- int tge_begin_game (const struct tge_parameters ∗parameters)

  Begins the timer game.

Variables

- static struct tge_parameters game_param

5.10.1 Detailed Description

Implementation of TGE main game functions. Implementation of TGE main game functions.

Author

Paul Griffiths

Copyright

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

5.10.2 Macro Definition Documentation

5.10.2.1 #define _POSIX_C_SOURCE 200809L

POSIX feature test macro

5.10.3 Function Documentation

5.10.3.1 int tge_begin_game (const struct tge_parameters * *parameters*)

Begins the timer game.

Begins the timer game.

Parameters

parameters	A pointer to a `struct tge_parameters` object containing the desired game parameters.

Returns

The exit status of the game.

5.10.3.2 static void tge_game_loop (void) `[static]`

Main game loop function.

5.10.4 Variable Documentation

5.10.4.1 struct tge_parameters game_param `[static]`

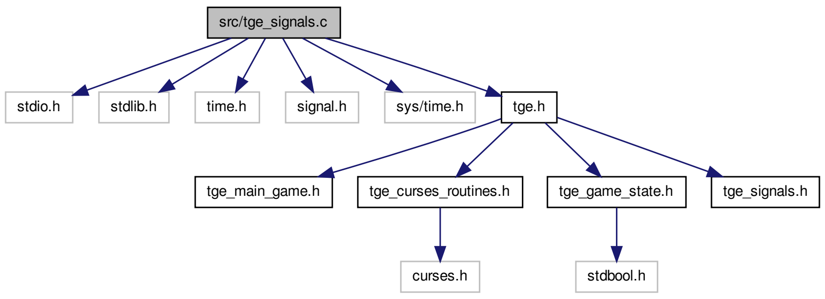
File scope variable for game parameters

5.11 src/tge_signals.c File Reference

Implementation of TGE signals functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <signal.h>
#include <sys/time.h>
#include "tge.h"
```

Include dependency graph for tge_signals.c:



Macros

- #define _POSIX_C_SOURCE 200809L

Functions

- static long tge_get_whole_seconds (const double seconds)

  Returns the whole seconds part of a seconds value.
- static long tge_get_fractional_microseconds (const double seconds)

  Returns the fractional microseconds part of a seconds value.
- static void tge_handler (int signum)

  Generic signal handler.
- void tge_set_signal_handlers (void)
- void tge_timer_start (const double start, const double interval)

  Starts the game timer.
- void tge_timer_stop (void)

5.11.1 Detailed Description

Implementation of TGE signals functions. Implementation of TGE signals functions.

Author

Paul Griffiths

Copyright

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

5.11.2 Macro Definition Documentation

5.11.2.1 #define _POSIX_C_SOURCE 200809L

POSIX feature test macro

5.11.3 Function Documentation

5.11.3.1 static long tge_get_fractional_microseconds (const double *seconds*) [static]

Returns the fractional microseconds part of a seconds value.

Returns the whole microseconds part of a seconds value.

Parameters

seconds	The seconds representation.

Returns

The fractional microseconds part of the seconds value.

5.11.3.2 static long tge_get_whole_seconds (const double *seconds*) [static]

Returns the whole seconds part of a seconds value.

Returns the whole seconds part of a seconds value.

Parameters

seconds	The seconds value.

Returns

The whole seconds part of the seconds value.

5.11.3.3 static void tge_handler (int *signum*) [static]

Generic signal handler.

Generic signal handler.

Parameters

signum	The signal number.

5.11.3.4 void tge_set_signal_handlers (void)

Registers the signal handlers.

5.11.3.5 void tge_timer_start (const double *start,* const double *interval*)

Starts the game timer.

Starts the game timer.

Parameters

start	The time until the first alarm, in seconds.
interval	The time interval between subsequent alarms, in seconds.

5.11.3.6 void tge_timer_stop (void)

Stops the game timer.

Index