

Worms!

Generated by Doxygen 1.8.1.2

Thu Nov 27 2014 14:58:47

Contents

1	Worms!	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Data Structure Documentation	7
4.1	cell Struct Reference	7
4.1.1	Detailed Description	7
4.1.2	Field Documentation	7
4.1.2.1	back	7
4.1.2.2	front	7
4.1.2.3	x	8
4.1.2.4	y	8
4.2	game_window Struct Reference	8
4.2.1	Field Documentation	8
4.2.1.1	cols	8
4.2.1.2	rows	8
4.2.1.3	window	8
4.3	worm Struct Reference	9
4.3.1	Detailed Description	9
4.3.2	Field Documentation	9
4.3.2.1	head	9
4.3.2.2	last_direction	9
4.3.2.3	next_direction	9
4.3.2.4	tail	9
5	File Documentation	11
5.1	docs/worms.dox File Reference	11
5.2	include/worms.h File Reference	11
5.2.1	Detailed Description	11

5.3	include/worms_food.h File Reference	12
5.3.1	Detailed Description	12
5.3.2	Function Documentation	12
5.3.2.1	worms_draw_food	12
5.3.2.2	worms_food_here	13
5.3.2.3	worms_get_food_eaten	13
5.3.2.4	worms_place_new_food	13
5.4	include/worms_game.h File Reference	13
5.4.1	Detailed Description	14
5.4.2	Enumeration Type Documentation	14
5.4.2.1	worms_exit_status	14
5.4.3	Function Documentation	14
5.4.3.1	worms_draw_screen	14
5.4.3.2	worms_game_setup	14
5.4.3.3	worms_game_teardown	14
5.4.3.4	worms_process_input	14
5.5	include/worms_screen.h File Reference	15
5.5.1	Detailed Description	16
5.5.2	Enumeration Type Documentation	16
5.5.2.1	worms_cell_character	16
5.5.3	Function Documentation	16
5.5.3.1	worms_coords_in_game_arena	16
5.5.3.2	worms_draw_arena_border	16
5.5.3.3	worms_draw_sidebar	17
5.5.3.4	worms_game_area_destroy	17
5.5.3.5	worms_game_area_init	17
5.5.3.6	worms_game_arena_cols	17
5.5.3.7	worms_game_arena_rows	17
5.5.3.8	worms_get_arena_character	17
5.5.3.9	worms_refresh_game_area	17
5.5.3.10	worms_write_arena_character	17
5.6	include/worms_time.h File Reference	18
5.6.1	Detailed Description	18
5.6.2	Function Documentation	18
5.6.2.1	worms_game_time	18
5.6.2.2	worms_game_time_string	19
5.6.2.3	worms_time_init	19
5.7	include/worms_worm.h File Reference	19
5.7.1	Detailed Description	20
5.7.2	Enumeration Type Documentation	20

5.7.2.1	worm_direction	20
5.7.3	Function Documentation	20
5.7.3.1	worm_destroy	20
5.7.3.2	worm_init	20
5.7.3.3	worm_move_and_draw	21
5.7.3.4	worm_set_direction	21
5.8	src/main.c File Reference	21
5.8.1	Detailed Description	21
5.8.2	Function Documentation	22
5.8.2.1	main	22
5.8.2.2	print_quit_message	22
5.9	src/worms_food.c File Reference	22
5.9.1	Detailed Description	23
5.9.2	Function Documentation	23
5.9.2.1	worms_draw_food	23
5.9.2.2	worms_food_here	23
5.9.2.3	worms_get_food_eaten	23
5.9.2.4	worms_place_new_food	23
5.9.3	Variable Documentation	24
5.9.3.1	food_eaten	24
5.9.3.2	food_x	24
5.9.3.3	food_y	24
5.10	src/worms_game.c File Reference	24
5.10.1	Detailed Description	24
5.10.2	Function Documentation	25
5.10.2.1	worms_draw_screen	25
5.10.2.2	worms_game_setup	25
5.10.2.3	worms_game_teardown	25
5.10.2.4	worms_process_input	25
5.11	src/worms_screen.c File Reference	25
5.11.1	Detailed Description	26
5.11.2	Function Documentation	27
5.11.2.1	worms_coords_in_game_arena	27
5.11.2.2	worms_draw_arena_border	27
5.11.2.3	worms_draw_info_window	27
5.11.2.4	worms_draw_sidebar	27
5.11.2.5	worms_draw_title_window	27
5.11.2.6	worms_game_area_destroy	27
5.11.2.7	worms_game_area_init	27
5.11.2.8	worms_game_arena_cols	27

5.11.2.9	worms_game_arena_rows	28
5.11.2.10	worms_get_arena_character	28
5.11.2.11	worms_refresh_game_area	28
5.11.2.12	worms_write_arena_character	28
5.11.3	Variable Documentation	28
5.11.3.1	arena_window	28
5.11.3.2	info_window	28
5.11.3.3	sidebar_cols	28
5.11.3.4	title_rows	29
5.11.3.5	title_window	29
5.12	src/worms_time.c File Reference	29
5.12.1	Detailed Description	29
5.12.2	Function Documentation	30
5.12.2.1	worms_game_time	30
5.12.2.2	worms_game_time_string	30
5.12.2.3	worms_time_init	30
5.12.3	Variable Documentation	30
5.12.3.1	start_time	30
5.13	src/worms_worm.c File Reference	30
5.13.1	Detailed Description	32
5.13.2	Function Documentation	32
5.13.2.1	worm_add_cell_head	32
5.13.2.2	worm_cell_here	32
5.13.2.3	worm_clear	32
5.13.2.4	worm_delete_cell_head	32
5.13.2.5	worm_delete_cell_tail	32
5.13.2.6	worm_destroy	32
5.13.2.7	worm_draw	33
5.13.2.8	worm_init	33
5.13.2.9	worm_move	33
5.13.2.10	worm_move_and_draw	33
5.13.2.11	worm_set_direction	33
5.13.3	Variable Documentation	33
5.13.3.1	worm	33
5.13.3.2	WORM_START_LENGTH	33

Chapter 1

Worms!

Worms! is an ncurses based worms game.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

cell	7
game_window	
Structure to hold details of game windows	8
worm	9

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

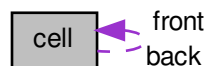
include/ worms.h	Aggregated header for worms game functions	11
include/ worms_food.h	Interface to worm food functions	12
include/ worms_game.h	Interface to TGE game engine callback functions	13
include/ worms_screen.h	Interface to worms game screen functions	15
include/ worms_time.h	Interface to worms game duration timer functions	18
include/ worms_worm.h	Interface to worms game worm functions	19
src/ main.c	Main() function for worms game	21
src/ worms_food.c	Implementation of worm food functions	22
src/ worms_game.c	Implementation of TGE game engine callback functions	24
src/ worms_screen.c	Implementation of worms game screen functions	25
src/ worms_time.c	Implementation of worms game duration timer functions	29
src/ worms_worm.c	Implementation of worms game worm functions	30

Chapter 4

Data Structure Documentation

4.1 cell Struct Reference

Collaboration diagram for cell:



Data Fields

- int `x`
- int `y`
- struct `cell` * `front`
- struct `cell` * `back`

4.1.1 Detailed Description

Structure to hold an individual cell of the worm's body

4.1.2 Field Documentation

4.1.2.1 struct `cell`* `cell::back`

Pointer to next cell towards tail

4.1.2.2 struct `cell`* `cell::front`

Pointer to next cell towards head

4.1.2.3 int cell::x

X coordinate of the cell

4.1.2.4 int cell::y

Y coordinate of the cell

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

- [src/worms_worm.c](#)

4.2 game_window Struct Reference

Structure to hold details of game windows.

Data Fields

- WINDOW * [window](#)
- int [rows](#)
- int [cols](#)

4.2.1 Field Documentation

4.2.1.1 int game_window::cols

Number of columns in the window

4.2.1.2 int game_window::rows

Number of rows in the window

4.2.1.3 WINDOW* game_window::window

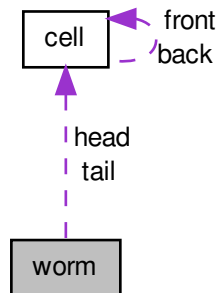
Pointer to curses WINDOW structure

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

- [src/worms_screen.c](#)

4.3 worm Struct Reference

Collaboration diagram for worm:



Data Fields

- struct [cell](#) * [head](#)
- struct [cell](#) * [tail](#)
- enum [worm_direction](#) [last_direction](#)
- enum [worm_direction](#) [next_direction](#)

4.3.1 Detailed Description

Structure to hold the worm

4.3.2 Field Documentation

4.3.2.1 struct [cell](#)* [worm::head](#)

Pointer to head cell

4.3.2.2 enum [worm_direction](#) [worm::last_direction](#)

Next direction to move

4.3.2.3 enum [worm_direction](#) [worm::next_direction](#)

Last direction moved

4.3.2.4 struct [cell](#)* [worm::tail](#)

Pointer to tail cell

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

- [src/worms_worm.c](#)

Chapter 5

File Documentation

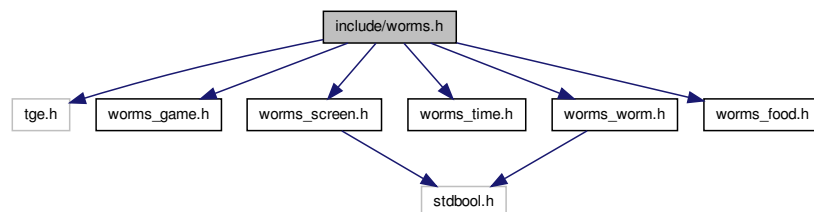
5.1 docs/worms.dox File Reference

5.2 include/worms.h File Reference

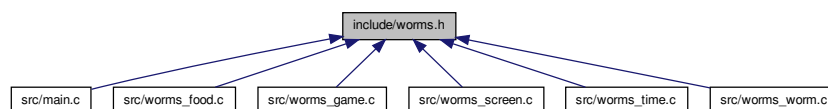
Aggregated header for worms game functions.

```
#include "tge.h"  
#include "worms_game.h"  
#include "worms_screen.h"  
#include "worms_time.h"  
#include "worms_worm.h"  
#include "worms_food.h"
```

Include dependency graph for worms.h:



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



5.2.1 Detailed Description

Aggregated header for worms game functions.

Author

Paul Griffiths

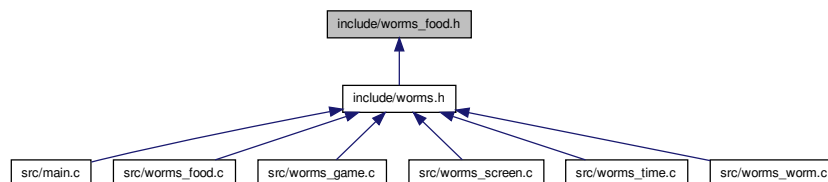
Copyright

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

5.3 include/worms_food.h File Reference

Interface to worm food functions.

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

**Functions**

- void [worms_place_new_food](#) (void)
- void [worms_draw_food](#) (void)
- bool [worms_food_here](#) (const int x, const int y)
Tests if a piece of food is at the specified coordinates.
- int [worms_get_food_eaten](#) (void)
Returns the total number of pieces of food eaten.

5.3.1 Detailed Description

Interface to worm food functions for curses worms game.

Author

Paul Griffiths

Copyright

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

5.3.2 Function Documentation

5.3.2.1 void worms_draw_food (void)

Draws the current piece of food.

5.3.2.2 bool worms_food_here (const int x, const int y)

Tests if a piece of food is at the specified coordinates.

Parameters

x	The specified x coordinate.
y	The specified y coordinate.

Returns

`true` if a piece of food is at the specified coordinates, `false` otherwise.

5.3.2.3 int worms_get_food_eaten (void)

Returns the total number of pieces of food eaten since the start of the game.

Returns

The total number of pieces of food eaten since the start of the game.

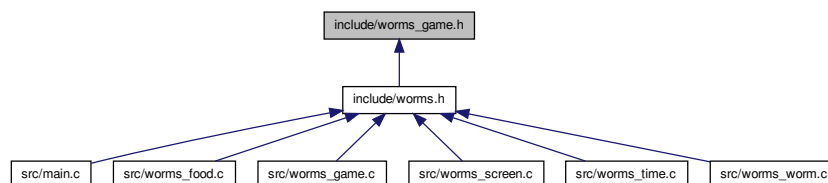
5.3.2.4 void worms_place_new_food (void)

Places a new piece of food at a random location.

5.4 include/worms_game.h File Reference

Interface to TGE game engine callback functions.

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



Enumerations

- enum `worms_exit_status` { `WORMS_EXIT_NORMAL`, `WORMS_EXIT_HITWALL`, `WORMS_EXIT_HITSELF` }

Functions

- void `worms_game_setup` (void)
Initializes the game.
- void `worms_game_teardown` (const int end_status)
Cleans up after the game ends.

- void `worms_draw_screen` (void)
- void `worms_process_input` (const int ch)

Processes keyboard input.

5.4.1 Detailed Description

Interface to TGE game engine callback functions, along with general game data structures and constants.

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 Enumeration Type Documentation

5.4.2.1 enum worms_exit_status

Enumeration constants for game exit status.

Enumerator:

`WORMS_EXIT_NORMAL` Player chose to quit
`WORMS_EXIT_HITWALL` Worm ran into the arena wall
`WORMS_EXIT_HITSELF` Worm ran into its own body

5.4.3 Function Documentation

5.4.3.1 void worms_draw_screen (void)

Draws the game screen at each timer interval.

5.4.3.2 void worms_game_setup (void)

Initializes the game area, worm, worm food, and game time.

5.4.3.3 void worms_game_teardown (const int end_status)

Destroys the worm and the game area.

Parameters

<code>end_status</code>	Status code for end-of-game reason.
-------------------------	-------------------------------------

5.4.3.4 void worms_process_input (const int ch)

Processes keyboard input to move the worm or quit the game.

Parameters

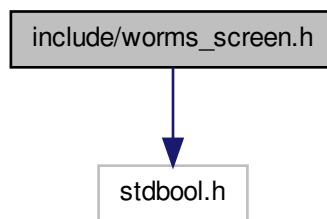
<i>ch</i>	Character code representing the key pressed.
-----------	--

5.5 include/worms_screen.h File Reference

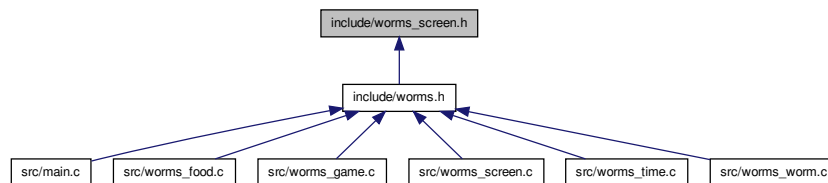
Interface to worms game screen functions.

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

Include dependency graph for worms_screen.h:



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



Enumerations

- enum `worms_cell_character` { `WORM_BODY_CHARACTER`, `WORM_EMPTY_CHARACTER`, `WORM_FOOD_CHARACTER` }

Functions

- void `worms_game_area_init` (void)
- void `worms_game_area_destroy` (void)
- void `worms_draw_arena_border` (void)
- void `worms_draw_sidebar` (void)
- void `worms_refresh_game_area` (void)
- int `worms_game_arena_cols` (void)
Returns the number of columns in the game arena.
- int `worms_game_arena_rows` (void)
Returns the number of rows in the game arena.

- void `worms_write_arena_character` (const enum `worms_cell_character` ch, const int x, const int y)
Writes a character to the game arena.
- enum `worms_cell_character` `worms_get_arena_character` (const int x, const int y)
Returns the character at the specified arena coordinates.
- bool `worms_coords_in_game_arena` (const int x, const int y)
Tests if the specified coordinates are within the arena.

5.5.1 Detailed Description

Interface to worms game screen functions. The game screen consists of two parts: the "arena", where the worm moves, and the "sidebar", containing the game title and game statistics. "Game area" can also refer to the entire game screen.

Author

Paul Griffiths

Copyright

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

5.5.2 Enumeration Type Documentation

5.5.2.1 enum worms_cell_character

Enumeration constants for game cell characters

Enumerator:

`WORM_BODY_CHARACTER` Character for worm body
`WORM_EMPTY_CHARACTER` Character for empty cell
`WORM_FOOD_CHARACTER` Character for piece of food

5.5.3 Function Documentation

5.5.3.1 bool worms_coords_in_game_arena (const int x, const int y)

Tests if the specified coordinates are within the arena. The coordinates are *not* considered to be within the arena if they are within the arena's border, so this test is suitable for determining whether the worm has hit the border at the edge of the arena.

Parameters

x	The specified x coordinate.
y	The specified y coordinate.

Returns

`true` if the specified coordinates are within the arena, excluding the arena border, and `false` otherwise.

5.5.3.2 void worms_draw_arena_border (void)

Draws a border around the game arena.

5.5.3.3 void worms_draw_sidebar (void)

Draws the sidebar.

5.5.3.4 void worms_game_area_destroy (void)

Destroys the game area, including the sidebar.

5.5.3.5 void worms_game_area_init (void)

Initializes the game area, including the sidebar.

5.5.3.6 int worms_game_arena_cols (void)

Returns the number of columns in the game arena. The number of columns *includes* the arena border, which is one character wide on all sides.

Returns

The number of columns in the game arena.

5.5.3.7 int worms_game_arena_rows (void)

Returns the number of rows in the game arena. The number of rows *includes* the arena border, which is one character wide on all sides.

Returns

The number of rows in the game arena.

5.5.3.8 enum worms_cell_character worms_get_arena_character (const int x, const int y)

Returns the character at the specified arena coordinates.

Parameters

x	The specified x coordinate.
y	The specified y coordinate.

Returns

The character at the specified coordinates in the arena.

5.5.3.9 void worms_refresh_game_area (void)

Refreshes the main arena and the sidebar.

5.5.3.10 void worms_write_arena_character (const enum worms_cell_character ch, const int x, const int y)

Writes a character to the game arena.

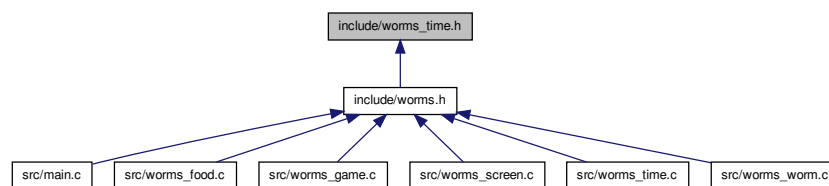
Parameters

<i>ch</i>	The character to write.
<i>x</i>	The x coordinate at which to write the character.
<i>y</i>	The y coordinate at which to write the character.

5.6 include/worms_time.h File Reference

Interface to worms game duration timer functions.

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



Functions

- void [worms_time_init](#) (void)
- long [worms_game_time](#) (void)
Returns the number of seconds since the game started.
- char * [worms_game_time_string](#) (const bool long_format)
Returns a string representation of total game time.

5.6.1 Detailed Description

Interface to worms game duration timer 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 long worms_game_time (void)

Returns the number of seconds since the game started.

Returns

The number of seconds since the game started.

5.6.2.2 char* worms_game_time_string (const bool *long_format*)

Returns a string representation of total game time.

Parameters

<i>long_format</i>	If this parameter is <code>true</code> , the string representation is of the form "[H] hours, [M] minutes, and [S] seconds". If <code>false</code> , the string representation is of the form "HH:MM:SS".
--------------------	---

Returns

A string representation of the total game time. The returned pointer points to statically allocated storage, and is overwritten each time this function is called.

5.6.2.3 void worms_time_init (void)

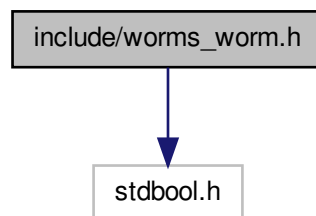
Initializes the game duration timer.

5.7 include/worms_worm.h File Reference

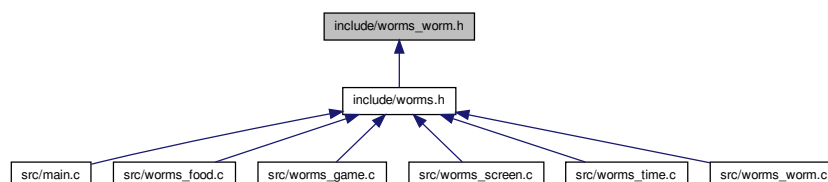
Interface to worms game worm functions.

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

Include dependency graph for worms_worm.h:



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



Enumerations

- enum `worm_direction` { `WORM_DIR_UP`, `WORM_DIR_RIGHT`, `WORM_DIR_DOWN`, `WORM_DIR_LEFT` }

Enumeration constants for worm movement direction.

Functions

- void `worm_init` (void)
Initializes the worm.
- void `worm_destroy` (void)
Destroys the worm.
- void `worm_set_direction` (enum `worm_direction` direction)
Attempts to change the next direction of the worm.
- bool `worm_move_and_draw` (void)
Moves and draws the worm.

5.7.1 Detailed Description

Interface to worms game worm functions.

Author

Paul Griffiths

Copyright

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

5.7.2 Enumeration Type Documentation

5.7.2.1 enum `worm_direction`

Enumeration constants for worm movement direction. The order in which these are defined is meaningful to the implementation, namely that pairs of opposing directions (i.e. up versus down, left versus right) have values with an absolute difference of 2.

Enumerator:

`WORM_DIR_UP` Up direction
`WORM_DIR_RIGHT` Right direction
`WORM_DIR_DOWN` Down direction
`WORM_DIR_LEFT` Left direction

5.7.3 Function Documentation

5.7.3.1 void `worm_destroy` (void)

Destroys and deallocates memory for the worm.

5.7.3.2 void `worm_init` (void)

Initializes and allocates memory for the worm.

5.7.3.3 bool worm_move_and_draw (void)

Moves and draws the worm.

Returns

`true` if the movement that was performed caused the worm to eat a piece of food, `false` in all other situations.

5.7.3.4 void worm_set_direction (enum worm_direction direction)

Attempts to change the next direction of the worm. The worm will not actually change direction until the game next updates, and a second call to this function in the interim will nullify the effect of any previous calls, so it is safe to call this function an arbitrary number of times in between game updates. This function will ignore any attempt to change the direction in a way which is not allowed, particularly that the worm is not allowed to turn 180 degrees as this would cause it to hit its own body, so it is safe to call this function with any direction, and to trust it to ignore any direction changes which are not allowed.

Parameters

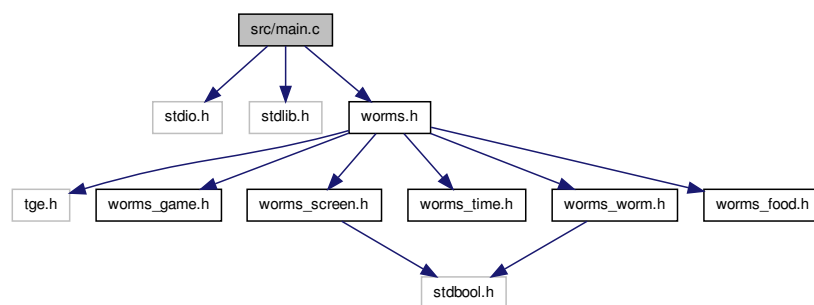
<i>direction</i>	The desired direction of the worm on the next update.
------------------	---

5.8 src/main.c File Reference

`main()` function for worms game.

```
#include <stdio.h>
#include <stdlib.h>
#include "worms.h"
```

Include dependency graph for main.c:



Functions

- static void `print_quit_message` (const int end_status)
Prints a quit message.
- int `main` (void)
`main()` function.

5.8.1 Detailed Description

Author

Paul Griffiths

Copyright

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

5.8.2 Function Documentation**5.8.2.1 int main (void)**

`main()` function.

Returns

The exit status of the program.

5.8.2.2 static void print_quit_message (const int *end_status*) [static]

Prints a quit message.

Parameters

<i>end_status</i>	The exit status of the game.
-------------------	------------------------------

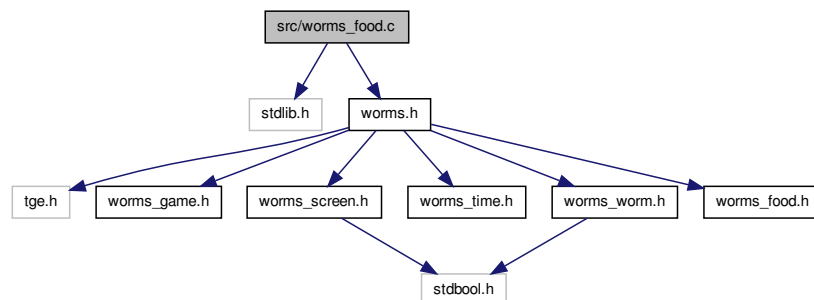
5.9 src/worms_food.c File Reference

Implementation of worm food functions.

```
#include <stdlib.h>
```

```
#include "worms.h"
```

Include dependency graph for worms_food.c:

**Functions**

- void `worms_place_new_food` (void)
- void `worms_draw_food` (void)
- bool `worms_food_here` (const int x, const int y)

Tests if a piece of food is at the specified coordinates.

- int `worms_get_food_eaten` (void)

Returns the total number of pieces of food eaten.

Variables

- static int `food_x` = 0
- static int `food_y` = 0
- static int `food_eaten` = -1

5.9.1 Detailed Description

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 Function Documentation

5.9.2.1 void worms_draw_food (void)

Draws the current piece of food.

5.9.2.2 bool worms_food_here (const int x, const int y)

Tests if a piece of food is at the specified coordinates.

Parameters

<code>x</code>	The specified x coordinate.
<code>y</code>	The specified y coordinate.

Returns

`true` if a piece of food is at the specified coordinates, `false` otherwise.

5.9.2.3 int worms_get_food_eaten (void)

Returns the total number of pieces of food eaten since the start of the game.

Returns

The total number of pieces of food eaten since the start of the game.

5.9.2.4 void worms_place_new_food (void)

Places a new piece of food at a random location.

5.9.3 Variable Documentation

5.9.3.1 `int food_eaten = -1` `[static]`

File scope variable for total food eaten since start of game

5.9.3.2 `int food_x = 0` `[static]`

File scope variable for x coordinate of current food piece

5.9.3.3 `int food_y = 0` `[static]`

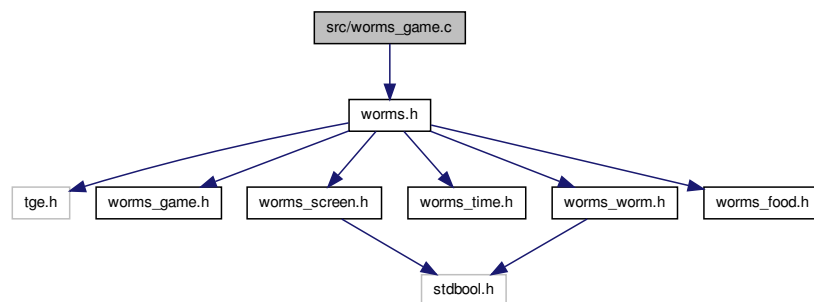
File scope variable for y coordinate of current food piece

5.10 `src/worms_game.c` File Reference

Implementation of TGE game engine callback functions.

```
#include "worms.h"
```

Include dependency graph for `worms_game.c`:



Functions

- void `worms_game_setup` (void)
Initializes the game.
- void `worms_game_teardown` (const int end_status)
Cleans up after the game ends.
- void `worms_draw_screen` (void)
- void `worms_process_input` (const int ch)
Processes keyboard input.

5.10.1 Detailed Description

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 Function Documentation

5.10.2.1 void worms_draw_screen (void)

Draws the game screen at each timer interval.

5.10.2.2 void worms_game_setup (void)

Initializes the game area, worm, worm food, and game time.

5.10.2.3 void worms_game_teardown (const int *end_status*)

Destroys the worm and the game area.

Parameters

<i>end_status</i>	Status code for end-of-game reason.
-------------------	-------------------------------------

5.10.2.4 void worms_process_input (const int *ch*)

Processes keyboard input to move the worm or quit the game.

Parameters

<i>ch</i>	Character code representing the key pressed.
-----------	--

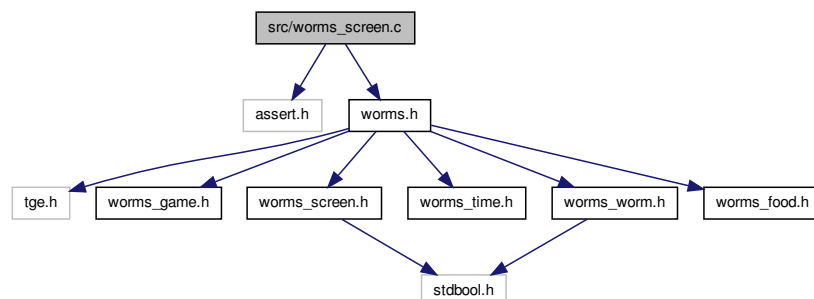
5.11 src/worms_screen.c File Reference

Implementation of worms game screen functions.

```
#include <assert.h>
```

```
#include "worms.h"
```

Include dependency graph for worms_screen.c:



Data Structures

- struct [game_window](#)

Structure to hold details of game windows.

Functions

- static void [worms_draw_title_window](#) (void)
- static void [worms_draw_info_window](#) (void)
- void [worms_game_area_init](#) (void)
- void [worms_game_area_destroy](#) (void)
- void [worms_draw_arena_border](#) (void)
- void [worms_draw_sidebar](#) (void)
- void [worms_refresh_game_area](#) (void)
- int [worms_game_arena_cols](#) (void)

Returns the number of columns in the game arena.

- int [worms_game_arena_rows](#) (void)

Returns the number of rows in the game arena.

- void [worms_write_arena_character](#) (const enum [worms_cell_character](#) ch, const int x, const int y)

Writes a character to the game arena.

- enum [worms_cell_character](#) [worms_get_arena_character](#) (const int x, const int y)

Returns the character at the specified arena coordinates.

- bool [worms_coords_in_game_arena](#) (const int x, const int y)

Tests if the specified coordinates are within the arena.

Variables

- static const int [sidebar_cols](#) = 20
- static const int [title_rows](#) = 7
- static struct [game_window](#) [arena_window](#)
- static struct [game_window](#) [title_window](#)
- static struct [game_window](#) [info_window](#)

5.11.1 Detailed Description

The game screen consists of two parts: the "arena", where the worm moves, and the "sidebar", containing the game title and game statistics. "Game area" can also refer to the entire game screen.

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 Function Documentation

5.11.2.1 `bool worms_coords_in_game_arena (const int x, const int y)`

Tests if the specified coordinates are within the arena. The coordinates are *not* considered to be within the arena if they are within the arena's border, so this test is suitable for determining whether the worm has hit the border at the edge of the arena.

Parameters

<code>x</code>	The specified x coordinate.
<code>y</code>	The specified y coordinate.

Returns

`true` if the specified coordinates are within the arena, excluding the arena border, and `false` otherwise.

5.11.2.2 `void worms_draw_arena_border (void)`

Draws a border around the game arena.

5.11.2.3 `static void worms_draw_info_window (void) [static]`

Draws the information window in the sidebar.

5.11.2.4 `void worms_draw_sidebar (void)`

Draws the sidebar.

5.11.2.5 `static void worms_draw_title_window (void) [static]`

Draws the title window in the sidebar.

5.11.2.6 `void worms_game_area_destroy (void)`

Destroys the game area, including the sidebar.

5.11.2.7 `void worms_game_area_init (void)`

Initializes the game area, including the sidebar.

5.11.2.8 `int worms_game_arena_cols (void)`

Returns the number of columns in the game arena. The number of columns *includes* the arena border, which is one character wide on all sides.

Returns

The number of columns in the game arena.

5.11.2.9 `int worms_game_arena_rows (void)`

Returns the number of rows in the game arena. The number of rows *includes* the arena border, which is one character wide on all sides.

Returns

The number of rows in the game arena.

5.11.2.10 `enum worms_cell_character worms_get_arena_character (const int x, const int y)`

Returns the character at the specified arena coordinates.

Parameters

<code>x</code>	The specified x coordinate.
<code>y</code>	The specified y coordinate.

Returns

The character at the specified coordinates in the arena.

5.11.2.11 `void worms_refresh_game_area (void)`

Refreshes the main arena and the sidebar.

5.11.2.12 `void worms_write_arena_character (const enum worms_cell_character ch, const int x, const int y)`

Writes a character to the game arena.

Parameters

<code>ch</code>	The character to write.
<code>x</code>	The x coordinate at which to write the character.
<code>y</code>	The y coordinate at which to write the character.

5.11.3 Variable Documentation

5.11.3.1 `struct game_window arena_window [static]`

File scope variable for main arena window

5.11.3.2 `struct game_window info_window [static]`

File scope variable for information window

5.11.3.3 `const int sidebar_cols = 20 [static]`

File scope variable for number of columns in sidebar

5.11.3.4 `const int title_rows = 7` `[static]`

File scope variable for number of rows in title window

5.11.3.5 `struct game_window title_window` `[static]`

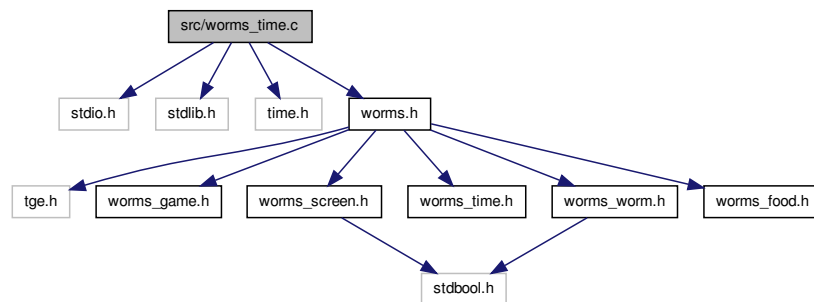
File scope variable for title window

5.12 src/worms_time.c File Reference

Implementation of worms game duration timer functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "worms.h"
```

Include dependency graph for worms_time.c:



Functions

- void `worms_time_init` (void)
- long `worms_game_time` (void)
Returns the number of seconds since the game started.
- char * `worms_game_time_string` (const bool long_format)
Returns a string representation of total game time.

Variables

- static time_t `start_time`

5.12.1 Detailed Description

Author

Paul Griffiths

Copyright

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

5.12.2 Function Documentation

5.12.2.1 `long worms_game_time (void)`

Returns the number of seconds since the game started.

Returns

The number of seconds since the game started.

5.12.2.2 `char* worms_game_time_string (const bool long_format)`

Returns a string representation of total game time.

Parameters

<i>long_format</i>	If this parameter is <code>true</code> , the string representation is of the form "[H] hours, [M] minutes, and [S] seconds". If <code>false</code> , the string representation is of the form "HH:MM:SS".
--------------------	---

Returns

A string representation of the total game time. The returned pointer points to statically allocated storage, and is overwritten each time this function is called.

5.12.2.3 `void worms_time_init (void)`

Initializes the game duration timer.

5.12.3 Variable Documentation

5.12.3.1 `time_t start_time` `[static]`

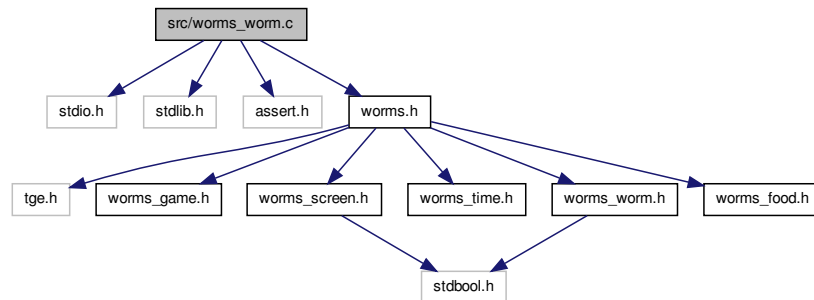
File scope variable to hold the time at the start of the game

5.13 `src/worms_worm.c` File Reference

Implementation of worms game worm functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#include "worms.h"
```

Include dependency graph for worms_worm.c:



Data Structures

- struct [cell](#)
- struct [worm](#)

Functions

- static void [worm_clear](#) (void)
Erases the worm from the arena.
- static bool [worm_move](#) (void)
Moves the worm.
- static void [worm_draw](#) (void)
- static bool [worm_cell_here](#) (const int x, const int y)
Tests if the specified coordinates contain a worm body cell.
- static void [worm_add_cell_head](#) (const int x, const int y)
Adds a cell at the head of the worm.
- static void [worm_delete_cell_head](#) (void)
- static void [worm_delete_cell_tail](#) (void)
- void [worm_init](#) (void)
Initializes the worm.
- void [worm_destroy](#) (void)
Destroys the worm.
- void [worm_set_direction](#) (enum [worm_direction](#) direction)
Attempts to change the next direction of the worm.
- bool [worm_move_and_draw](#) (void)
Moves and draws the worm.

Variables

- static struct [worm](#) [worm](#) = {NULL, NULL, [WORM_DIR_RIGHT](#), [WORM_DIR_RIGHT](#)}
- static const size_t [WORM_START_LENGTH](#) = 8

5.13.1 Detailed Description

The worm is implemented as a double-ended, doubly-linked list.

Author

Paul Griffiths

Copyright

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

5.13.2 Function Documentation

5.13.2.1 `static void worm_add_cell_head (const int x, const int y) [static]`

Adds a cell at the head of the worm.

Parameters

<code>x</code>	The x coordinate of the new head.
<code>y</code>	The y coordinate of the new head.

5.13.2.2 `static bool worm_cell_here (const int x, const int y) [static]`

Tests if the specified coordinates contain a worm body cell. This function is used to test for collision with the worm's own body.

Parameters

<code>x</code>	The specified x coordinate.
<code>y</code>	The specified y coordinate.

Returns

`true` if the specified coordinates contain a worm body cell, `false` otherwise.

5.13.2.3 `static void worm_clear (void) [static]`

Erases the worm from the arena, i.e. overwrites it with empty cell characters.

5.13.2.4 `static void worm_delete_cell_head (void) [static]`

Deletes the cell at the head of the worm.

5.13.2.5 `static void worm_delete_cell_tail (void) [static]`

Deletes the cell at the tail of the worm.

5.13.2.6 `void worm_destroy (void)`

Destroys and deallocates memory for the worm.

5.13.2.7 static void worm_draw (void) [static]

Draws the worm.

5.13.2.8 void worm_init (void)

Initializes and allocates memory for the worm.

5.13.2.9 static bool worm_move (void) [static]

Moves the worm. The worm moves in the next direction specified in the worm structure. The function detects if the move would cause the worm to eat some food, and whether it would cause the worm to hit the arena wall, or its own body. In the former case, the tail cell is not deleted, and the worm grows in length. In the latter case, the game is ended.

Returns

`true` if the move caused the worm to eat some food, `false` in all other situations.

5.13.2.10 bool worm_move_and_draw (void)

Moves and draws the worm.

Returns

`true` if the movement that was performed caused the worm to eat a piece of food, `false` in all other situations.

5.13.2.11 void worm_set_direction (enum worm_direction direction)

Attempts to change the next direction of the worm. The worm will not actually change direction until the game next updates, and a second call to this function in the interim will nullify the effect of any previous calls, so it is safe to call this function an arbitrary number of times in between game updates. This function will ignore any attempt to change the direction in a way which is not allowed, particularly that the worm is not allowed to turn 180 degrees as this would cause it to hit its own body, so it is safe to call this function with any direction, and to trust it to ignore any direction changes which are not allowed.

Parameters

<i>direction</i>	The desired direction of the worm on the next update.
------------------	---

5.13.3 Variable Documentation**5.13.3.1 struct worm worm = {NULL, NULL, WORM_DIR_RIGHT, WORM_DIR_RIGHT} [static]**

File scope variable to contain the worm

5.13.3.2 const size_t WORM_START_LENGTH = 8 [static]

File scope constant containing initial length of worm

Index

arena_window
worms_screen.c, 28

back
cell, 7

cell, 7
back, 7
front, 7
x, 7
y, 8

cols
game_window, 8

docs/worms.dox, 11

food_eaten
worms_food.c, 24

food_x
worms_food.c, 24

food_y
worms_food.c, 24

front
cell, 7

game_window, 8
cols, 8
rows, 8
window, 8

head
worm, 9

include/worms.h, 11
include/worms_food.h, 12
include/worms_game.h, 13
include/worms_screen.h, 15
include/worms_time.h, 18
include/worms_worm.h, 19
info_window
worms_screen.c, 28

last_direction
worm, 9

main
main.c, 22

main.c
main, 22
print_quit_message, 22

next_direction

worm, 9

print_quit_message
main.c, 22

rows
game_window, 8

sidebar_cols
worms_screen.c, 28

src/main.c, 21
src/worms_food.c, 22
src/worms_game.c, 24
src/worms_screen.c, 25
src/worms_time.c, 29
src/worms_worm.c, 30
start_time
worms_time.c, 30

tail
worm, 9

title_rows
worms_screen.c, 28

title_window
worms_screen.c, 29

WORM_BODY_CHARACTER
worms_screen.h, 16

WORM_DIR_DOWN
worms_worm.h, 20

WORM_DIR_LEFT
worms_worm.h, 20

WORM_DIR_RIGHT
worms_worm.h, 20

WORM_DIR_UP
worms_worm.h, 20

WORM_EMPTY_CHARACTER
worms_screen.h, 16

WORM_FOOD_CHARACTER
worms_screen.h, 16

WORMS_EXIT_HITSELF
worms_game.h, 14

WORMS_EXIT_HITWALL
worms_game.h, 14

WORMS_EXIT_NORMAL
worms_game.h, 14

WORM_START_LENGTH
worms_worm.c, 33

window
game_window, 8

worm, 9

- head, [9](#)
- last_direction, [9](#)
- next_direction, [9](#)
- tail, [9](#)
- worms_worm.c, [33](#)
- worm_add_cell_head
 - worms_worm.c, [32](#)
- worm_cell_here
 - worms_worm.c, [32](#)
- worm_clear
 - worms_worm.c, [32](#)
- worm_delete_cell_head
 - worms_worm.c, [32](#)
- worm_delete_cell_tail
 - worms_worm.c, [32](#)
- worm_destroy
 - worms_worm.c, [32](#)
 - worms_worm.h, [20](#)
- worm_direction
 - worms_worm.h, [20](#)
- worm_draw
 - worms_worm.c, [32](#)
- worm_init
 - worms_worm.c, [33](#)
 - worms_worm.h, [20](#)
- worm_move
 - worms_worm.c, [33](#)
- worm_move_and_draw
 - worms_worm.c, [33](#)
 - worms_worm.h, [20](#)
- worm_set_direction
 - worms_worm.c, [33](#)
 - worms_worm.h, [21](#)
- worms_game.h
 - WORMS_EXIT_HITSELF, [14](#)
 - WORMS_EXIT_HITWALL, [14](#)
 - WORMS_EXIT_NORMAL, [14](#)
- worms_screen.h
 - WORM_BODY_CHARACTER, [16](#)
 - WORM_EMPTY_CHARACTER, [16](#)
 - WORM_FOOD_CHARACTER, [16](#)
- worms_worm.h
 - WORM_DIR_DOWN, [20](#)
 - WORM_DIR_LEFT, [20](#)
 - WORM_DIR_RIGHT, [20](#)
 - WORM_DIR_UP, [20](#)
- worms_cell_character
 - worms_screen.h, [16](#)
- worms_coords_in_game_arena
 - worms_screen.c, [27](#)
 - worms_screen.h, [16](#)
- worms_draw_arena_border
 - worms_screen.c, [27](#)
 - worms_screen.h, [16](#)
- worms_draw_food
 - worms_food.c, [23](#)
 - worms_food.h, [12](#)
- worms_draw_info_window
 - worms_screen.c, [27](#)
 - worms_screen.h, [16](#)
- worms_draw_sidebar
 - worms_screen.c, [27](#)
 - worms_screen.h, [16](#)
- worms_draw_title_window
 - worms_screen.c, [27](#)
- worms_exit_status
 - worms_game.h, [14](#)
- worms_food.c
 - food_eaten, [24](#)
 - food_x, [24](#)
 - food_y, [24](#)
 - worms_draw_food, [23](#)
 - worms_food_here, [23](#)
 - worms_get_food_eaten, [23](#)
 - worms_place_new_food, [23](#)
- worms_food.h
 - worms_draw_food, [12](#)
 - worms_food_here, [12](#)
 - worms_get_food_eaten, [13](#)
 - worms_place_new_food, [13](#)
- worms_food_here
 - worms_food.c, [23](#)
 - worms_food.h, [12](#)
- worms_game.c
 - worms_draw_screen, [25](#)
 - worms_game_setup, [25](#)
 - worms_game_teardown, [25](#)
 - worms_process_input, [25](#)
- worms_game.h
 - worms_draw_screen, [14](#)
 - worms_exit_status, [14](#)
 - worms_game_setup, [14](#)
 - worms_game_teardown, [14](#)
 - worms_process_input, [14](#)
- worms_game_area_destroy
 - worms_screen.c, [27](#)
 - worms_screen.h, [17](#)
- worms_game_area_init
 - worms_screen.c, [27](#)
 - worms_screen.h, [17](#)
- worms_game_arena_cols
 - worms_screen.c, [27](#)
 - worms_screen.h, [17](#)
- worms_game_arena_rows
 - worms_screen.c, [27](#)
 - worms_screen.h, [17](#)
- worms_game_setup
 - worms_game.c, [25](#)
 - worms_game.h, [14](#)
- worms_game_teardown
 - worms_game.c, [25](#)
 - worms_game.h, [14](#)
- worms_game_time
 - worms_time.c, [30](#)

- worms_time.h, 18
- worms_game_time_string
 - worms_time.c, 30
 - worms_time.h, 18
- worms_get_arena_character
 - worms_screen.c, 28
 - worms_screen.h, 17
- worms_get_food_eaten
 - worms_food.c, 23
 - worms_food.h, 13
- worms_place_new_food
 - worms_food.c, 23
 - worms_food.h, 13
- worms_process_input
 - worms_game.c, 25
 - worms_game.h, 14
- worms_refresh_game_area
 - worms_screen.c, 28
 - worms_screen.h, 17
- worms_screen.c
 - arena_window, 28
 - info_window, 28
 - sidebar_cols, 28
 - title_rows, 28
 - title_window, 29
 - worms_coords_in_game_arena, 27
 - worms_draw_arena_border, 27
 - worms_draw_info_window, 27
 - worms_draw_sidebar, 27
 - worms_draw_title_window, 27
 - worms_game_area_destroy, 27
 - worms_game_area_init, 27
 - worms_game_arena_cols, 27
 - worms_game_arena_rows, 27
 - worms_get_arena_character, 28
 - worms_refresh_game_area, 28
 - worms_write_arena_character, 28
- worms_screen.h
 - worms_cell_character, 16
 - worms_coords_in_game_arena, 16
 - worms_draw_arena_border, 16
 - worms_draw_sidebar, 16
 - worms_game_area_destroy, 17
 - worms_game_area_init, 17
 - worms_game_arena_cols, 17
 - worms_game_arena_rows, 17
 - worms_get_arena_character, 17
 - worms_refresh_game_area, 17
 - worms_write_arena_character, 17
- worms_time.c
 - start_time, 30
 - worms_game_time, 30
 - worms_game_time_string, 30
 - worms_time_init, 30
- worms_time.h
 - worms_game_time, 18
 - worms_game_time_string, 18
 - worms_time_init, 19
- worms_time_init
 - worms_time.c, 30
 - worms_time.h, 19
- worms_worm.c
 - WORM_START_LENGTH, 33
 - worm, 33
 - worm_add_cell_head, 32
 - worm_cell_here, 32
 - worm_clear, 32
 - worm_delete_cell_head, 32
 - worm_delete_cell_tail, 32
 - worm_destroy, 32
 - worm_draw, 32
 - worm_init, 33
 - worm_move, 33
 - worm_move_and_draw, 33
 - worm_set_direction, 33
- worms_worm.h
 - worm_destroy, 20
 - worm_direction, 20
 - worm_init, 20
 - worm_move_and_draw, 20
 - worm_set_direction, 21
- worms_write_arena_character
 - worms_screen.c, 28
 - worms_screen.h, 17
- x
 - cell, 7
- y
 - cell, 8