Super Martin Level Editor

Generated by Doxygen 1.8.6

Thu Mar 20 2014 15:02:22

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 documented files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 Cursor Struct Reference

Data Fields

- int **x**
- int **y**
- int **tileID**

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

- const.h

3.2 Input Struct Reference

Data Fields

- int **left**
- int **right**
- int **add**
- int **remove**
- int **previous**
- int **next**
- int **load**
- int **save**
- int **copy**
- int **reinit**
- int **mouseX**
- int **mouseY**

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

- const.h

3.3 Level Struct Reference

Data Fields

- unsigned char $**$ **map**

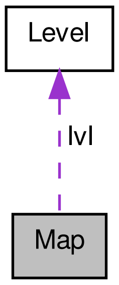
- int **width**
- int **height**
- int **timer_level**
- char **music** [MAX_LENGTH_FILE_NAME]
- char **background** [MAX_LENGTH_FILE_NAME]

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

- const.h

3.4 Map Struct Reference

Collaboration diagram for Map:



Data Fields

- Level ∗ **lvl**
- int **xScroll**
- int **screenWidth**
- int **screenHeight**

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

- const.h

Chapter 4

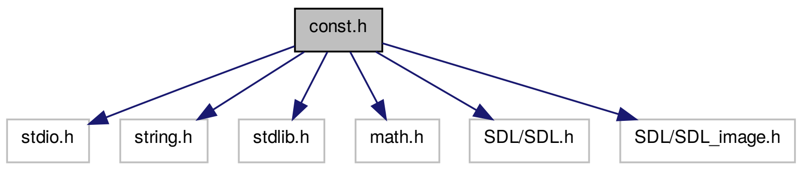
File Documentation

4.1 const.h File Reference

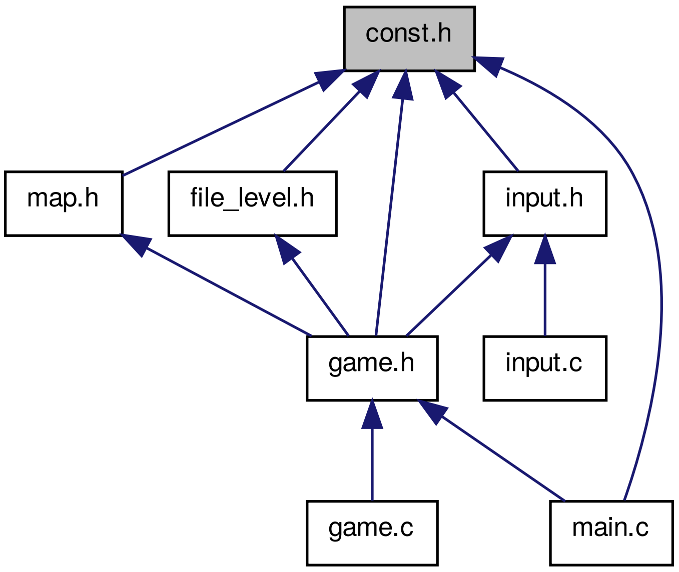
Definitions of every constants and structures used.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
```

Include dependency graph for const.h:



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



Data Structures

- struct Level
- struct Map
- struct Cursor
- struct Input

Macros

- #define **TILE_SIZE** 16
- #define **NB_TILES_X** 60
- #define **NB_TILES_Y** 34
- #define **SCREEN_WIDTH** TILE_SIZE $*$ NB_TILES_X
- #define **SCREEN_HEIGHT** TILE_SIZE $*$ NB_TILES_Y
- #define **TILESET_LAST** 8
- #define **FPS** 60
- #define **MAX_LENGTH_FILE_NAME** 100
- #define **TRANS_R** 255
- #define **TRANS_G** 255
- #define **TRANS_B** 255

Typedefs

- typedef struct Level **Level**
- typedef struct Map **Map**
- typedef struct Cursor **Cursor**
- typedef struct Input **Input**

Enumerations

- enum { **RIGHT**, **LEFT**, **UP**, **DOWN** }

4.1.1 Detailed Description

Definitions of every constants and structures used.

Author

Xavier COPONET, Glenn HERROU

Date

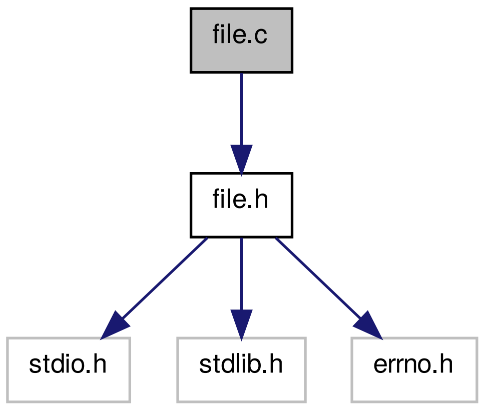
2014-03-18

4.2 file.c File Reference

Contains the functions that handle the files of the game.

```
#include "file.h"
```

Include dependency graph for file.c:



Functions

- FILE ∗ openFile (char nom[ ], char mode[ ])
- int closeFile (FILE ∗ptr_fichier)
- int readFileSize (FILE ∗ptr_fichier)

4.2.1 Detailed Description

Contains the functions that handle the files of the game.

Author

Remi BERTHO, Glenn HERROU

Date

15/03/14

4.2.2 Function Documentation

4.2.2.1 int closeFile (FILE ∗ *ptr_fichier*)

Ferme le fichier

Parameters

in	∗*ptr_fichier*	le fichier

Returns

entier 0 si tout s'est bien passe, 1 sinon

4.2.2.2 FILE ∗ openFile (char *nom[ ]*, char *mode[ ]*)

Ouvre un fichier a partir de son nom (nom[]) et du mode voulu (mode[])

Parameters

in	*nom[ ]*	le nom du fichier
in	*mode[ ]*	le mode voulu

Returns

un pointeur sur le fichier ouvert, NULL s'il y a eut un probleme

4.2.2.3 int readFileSize (FILE ∗ *ptr_fichier*)

Lis la taille du fichier

Parameters

in	∗*ptr_fichier*	le fichier

Returns

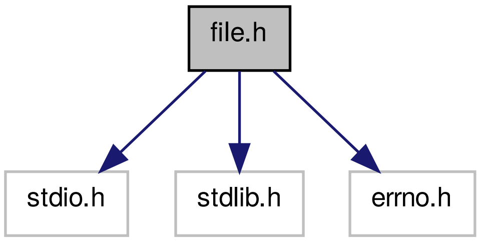
entier ayant la taille du fichier

4.3 file.h File Reference

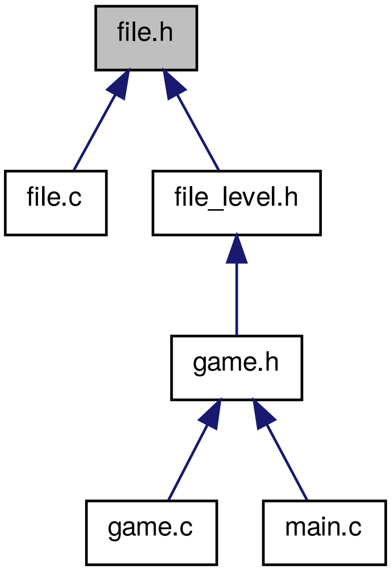
Header of file.c.

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

Include dependency graph for file.h:



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



Functions

- FILE ∗ openFile (char nome[ ], char mode[ ])
- int closeFile (FILE ∗ptr_fichier)
- int readFileSize (FILE ∗ptr_fichier)

4.3.1 Detailed Description

Header of file.c.

Author

Remi BERTHO

Date

15/03/14

4.3.2 Function Documentation

4.3.2.1 int closeFile (FILE ∗ *ptr_fichier*)

Ferme le fichier

Parameters

in	∗*ptr_fichier*	le fichier

Returns

entier 0 si tout s'est bien passe, 1 sinon

4.3.2.2 FILE∗ openFile (char *nom[ ]*, char *mode[ ]*)

Ouvre un fichier a partir de son nom (nom[]) et du mode voulu (mode[])

Parameters

in	*nom[]*	le nom du fichier
in	*mode[]*	le mode voulu

Returns

un pointeur sur le fichier ouvert, NULL s'il y a eut un probleme

4.3.2.3 int readFileSize (FILE ∗ *ptr_fichier*)

Lis la taille du fichier

Parameters

in	∗*ptr_fichier*	le fichier

Returns

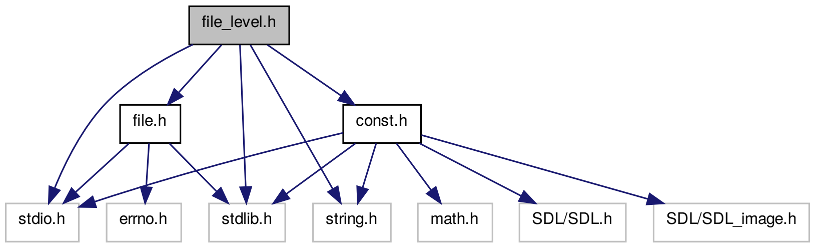
entier ayant la taille du fichier

4.4 file_level.h File Reference

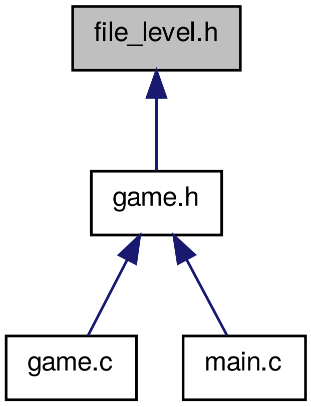
Header of file_level.c.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "file.h"
#include "const.h"
```

Include dependency graph for file_level.h:



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



Macros

- #define **TAILLE_BUFFER** 2

Functions

- Level ∗ openLevel (char ∗file_name)
- void closeLevel (Level ∗lvl)
- Level ∗ initLevel (Level ∗lvl)
- void writeLevel (char ∗file_name, Level ∗lvl)
- char ∗∗ readLevelFile (int ∗nb_lvl)
- void closeLevelList (char ∗∗level_names, int nb_lvl)

4.4.1 Detailed Description

Header of file_level.c.

Author

Remi BERTHO

Date

15/03/14

Version

1.0

4.4.2 Function Documentation

4.4.2.1 void closeLevel (Level ∗ *lvl*)

Close a level by deallocating its map

Parameters

`out`	*lvl*	The level to close

4.4.2.2 void closeLevelList (char ∗∗ *level_names*, int *nb_lvl*)

Deallocate the array of level names created by the function readLevelFile

Parameters

`in,out`	*level_names*	la liste des noms de niveau
`in`	*nb_lvl*	le nombre de niveau

4.4.2.3 Level∗ initLevel (Level ∗ *lvl*)

Initialize a level. The width and the height of the level must be initiated before calling this function

Parameters

`out`	*lvl*	The level to initialize

Returns

a pointer on the level initialized

4.4.2.4 Level∗ openLevel (char ∗ *file_name*)

Open a level file and create the level corresponding to the file

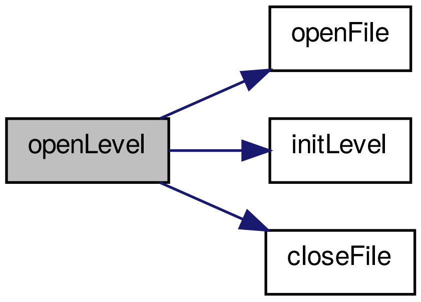
Parameters

`in`	*file_name*	The name of the file to open

Returns

a pointer on the level created

Here is the call graph for this function:



4.4.2.5 char∗∗ readLevelFile (int ∗ *nb_lvl*)

Read the file "level" which contains the list of all existing levels. The first line of this file contains the length of the list, each other line contains one, and only one, level file name

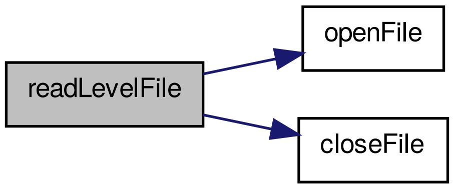
Parameters

out	*nb_lvl*	The number of existing levels

Returns

a pointer on the array containing all existing level names

Here is the call graph for this function:



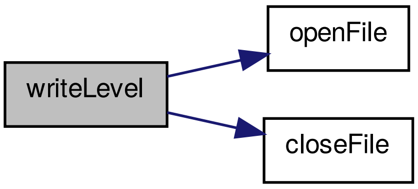
4.4.2.6 void writeLevel (char ∗ *file_name,* Level ∗ *lvl*)

Write a level in a file

Parameters

in	*lvl*	The level to write
in	*file_name*	The file to write in

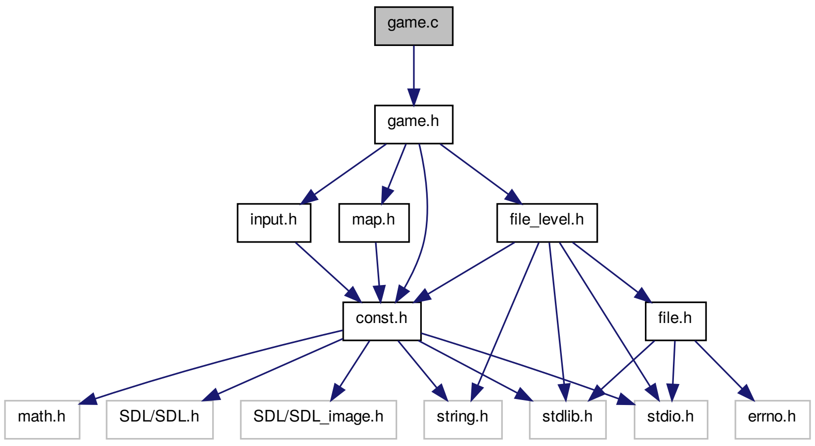
Here is the call graph for this function:



4.5 game.c File Reference

Contains the principal function of the game.

`#include "game.h"`
Include dependency graph for game.c:



Functions

- void play (SDL_Surface ∗screen, char ∗level_name)

4.5.1 Detailed Description

Contains the principal function of the game. Contains the functions managing the maps.

Author

Xavier COPONET, Glenn HERROU

Date

2014-03-18

4.5.2 Function Documentation

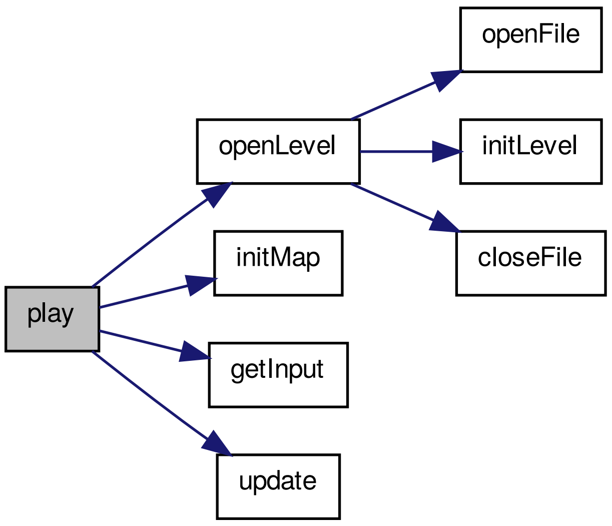
4.5.2.1 void play (SDL_Surface ∗ *screen*, char ∗ *level_name*)

Contains the infinite loop of the game, call the main functions

Parameters

`in,out`	*screen*	Game screen
`in`	*level_name*	Level name

Here is the call graph for this function:

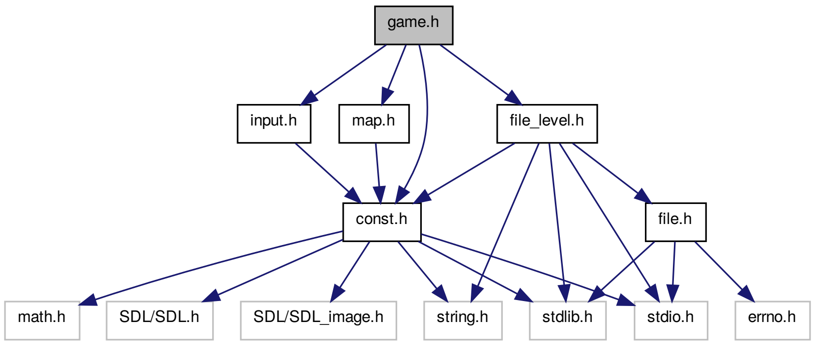


4.6 game.h File Reference

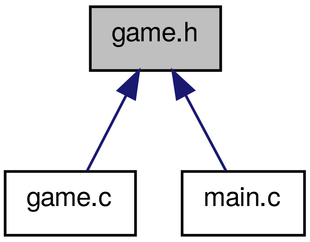
Header of game.c.

```
#include "const.h"
#include "file_level.h"
#include "input.h"
#include "map.h"
```

Include dependency graph for game.h:



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



Functions

- void play (SDL_Surface ∗screen, char ∗level_name)

4.6.1 Detailed Description

Header of game.c.

Author

Xavier COPONET, Glenn HERROU

Date

2014-03-18

4.6.2 Function Documentation

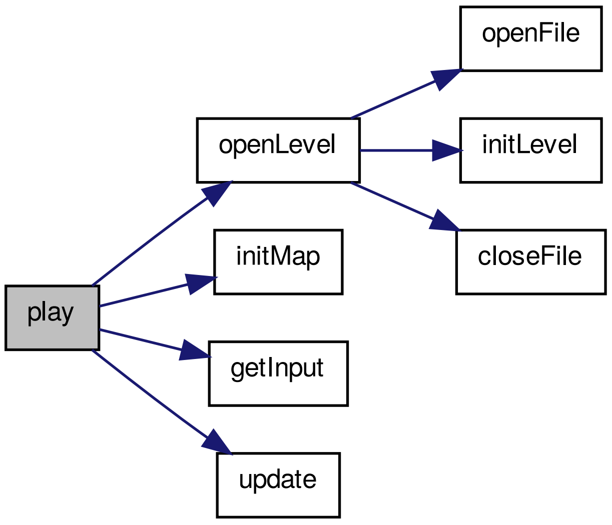
4.6.2.1 void play (SDL_Surface ∗ *screen,* char ∗ *level_name*)

Contains the infinite loop of the game, call the main functions

Parameters

`in,out`	*screen*	Game screen
`in`	*level_name*	Level name

Here is the call graph for this function:

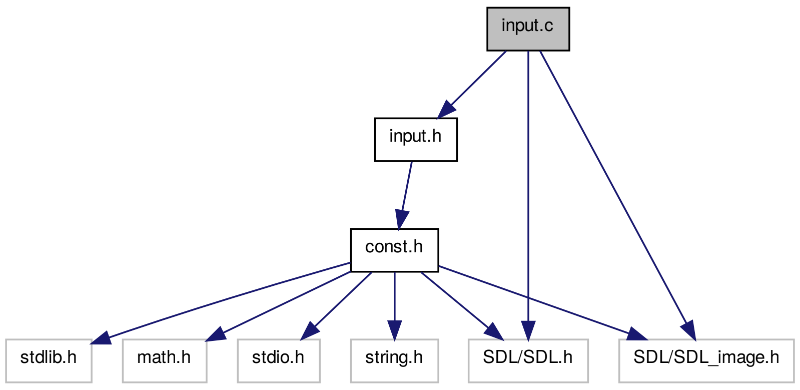


4.7 input.c File Reference

Management of keyboard and mouse inputs handled by the game.

```
#include "input.h"
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
```

Include dependency graph for input.c:



Functions

- void getInput (Input ∗input)
- void update (Map ∗m, Input ∗input, Cursor ∗cursor)

4.7.1 Detailed Description

Management of keyboard and mouse inputs handled by the game.

Author

Glenn HERROU

Date

2014-03-18

4.7.2 Function Documentation

4.7.2.1 void getInput (**Input** ∗ *input*)

Set/reset the right variable of the input structure depending on the event polled

Parameters

`in,out`	*input*	The structure where inputs are saved

4.7.2.2 void update (**Map** ∗ *m,* **Input** ∗ *input,* **Cursor** ∗ *cursor*)

Update the map and the screen following variables of the input structure

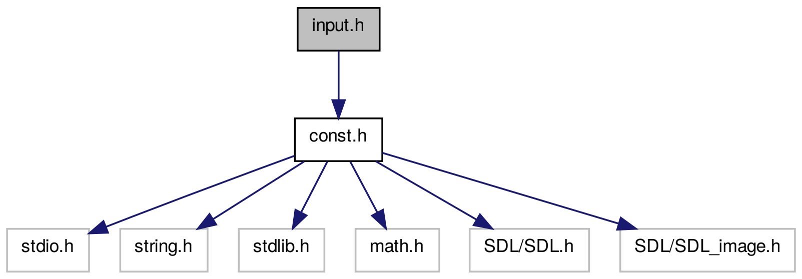
Parameters

`in,out`	*m*	Map to update
`in,out`	*input*	The structure where inputs has been saved
`in,out`	*cursor*	The cursor of the mouse

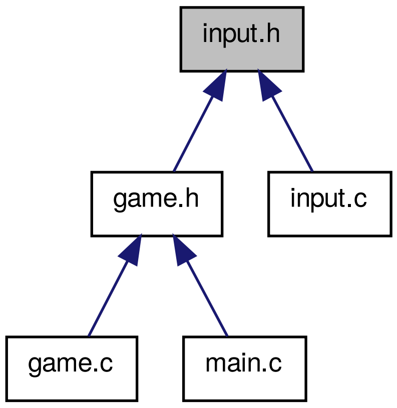
4.8 input.h File Reference

Header of input.c.

```
#include "const.h"
```
Include dependency graph for input.h:



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



Functions

- void getInput (Input ∗input)
- void update (Map ∗m, Input ∗input, Cursor ∗cursor)

Variables

- Input **input**
- Map **map**
- Cursor **cursor**

4.8.1 Detailed Description

Header of input.c.

Author

Glenn HERROU

Date

2014-03-18

4.8.2 Function Documentation

4.8.2.1 void getInput (**Input** ∗ *input*)

Set/reset the right variable of the input structure depending on the event polled

Parameters

`in,out`	*input*	The structure where inputs are saved

4.8.2.2 void update (**Map** ∗ *m,* **Input** ∗ *input,* **Cursor** ∗ *cursor*)

Update the map and the screen following variables of the input structure

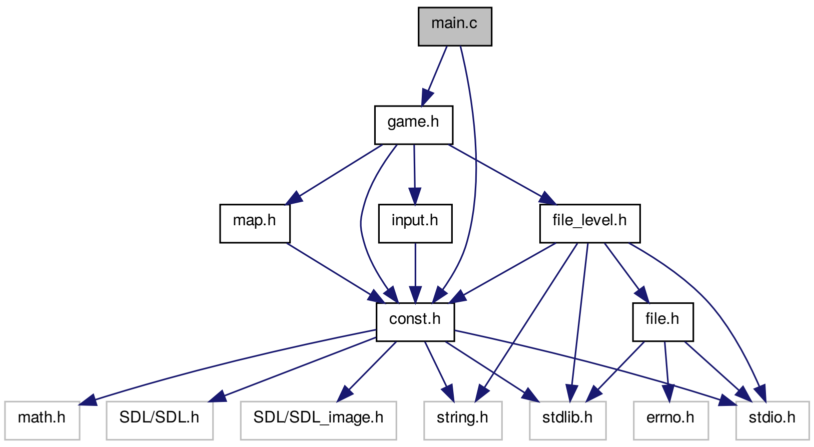
Parameters

`in,out`	*m*	Map to update
`in,out`	*input*	The structure where inputs has been saved
`in,out`	*cursor*	The cursor of the mouse

4.9 main.c File Reference

```
#include "game.h"
#include "const.h"
```

Include dependency graph for main.c:



Functions

- int **main** (int argc, char ∗argv[])

4.9.1 Detailed Description

Author

Xavier COPONET, Glenn HERROU

Date

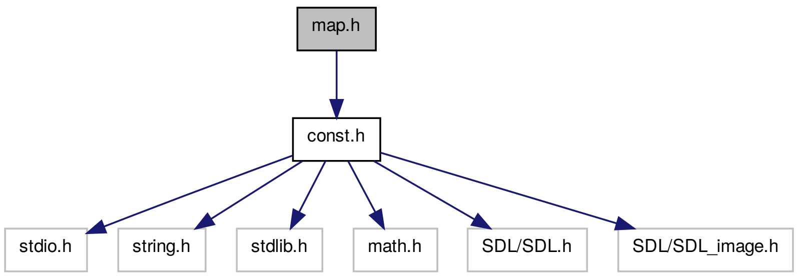
2014-03-18

4.10 map.h File Reference

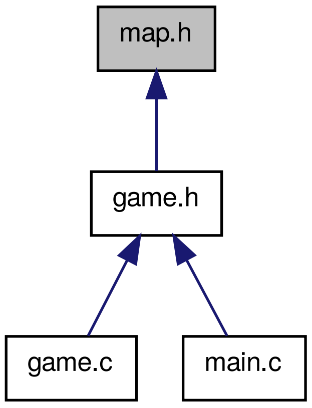
Header of map.c.

`#include "const.h"`

Include dependency graph for map.h:



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



Functions

- void **updateScreenMap** (SDL_Surface ∗screen, Map ∗m, char ∗tileset, Cursor ∗cursor)
- void **scrolling** (Map ∗m, int direction)
- Map ∗ initMap (Level ∗lvl, SDL_Surface ∗screen)
- void **saveMap** (Map ∗m)
- void **cleanString** (const char ∗buffer, FILE ∗fp)
- void **clean_stdin** (void)
- void **freeMap** (Map ∗m)

4.10.1 Detailed Description

Header of map.c.

Author

Xavier COPONET, Glenn HERROU

Date

2014-03-18

4.10.2 Function Documentation

4.10.2.1 Map∗ initMap (Level ∗ *lvl,* SDL_Surface ∗ *screen*)

Initialize the map

Parameters

`in`	*screen*	The screen of the game
`in`	*level*	The level of the map

Returns

a pointer on the map initialized

Index