Super Martin Level Editor

Generated by Doxygen 1.8.6

Mon Jun 2 2014 13:59:31

Contents

1 Data Structure Documentation

1.1 Cursor Struct Reference

`#include <structures.h>`

Data Fields

- int x
- int y
- char tileID

1.1.1 Detailed Description

The cursor structure

1.1.2 Field Documentation

1.1.2.1 char tileID

The tile ID

1.1.2.2 int x

The x coordinate

1.1.2.3 int y

The y coordinate

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

- structures.h

1.2 Input Struct Reference

`#include <structures.h>`

Data Fields

- char key [SDLK_LAST]
- char mouse [6]
- int mouseX
- int mouseY
- int quit

1.2.1 Detailed Description

The input structure

1.2.2 Field Documentation

1.2.2.1 char key[SDLK_LAST]

all the keyboard keys : 1 the key is pushed, 0 isn't

1.2.2.2 char mouse[6]

all the mouse buttons : 1 the button is pushed, 0 isn't

1.2.2.3 int mouseX

The x coordinate of the mouse

1.2.2.4 int mouseY

The y coordinate of the mouse

1.2.2.5 int quit

is 1 is the SDL_QUIT event happens

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

- structures.h

1.3 Level Struct Reference

`#include <structures.h>`

Data Fields

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

1.3.1 Detailed Description

The level structure

1.3.2 Field Documentation

1.3.2.1 char background[MAX_LENGTH_FILE_NAME]

The background

1.3.2.2 int height

The height

1.3.2.3 unsigned char** map

The map

1.3.2.4 char music[MAX_LENGTH_FILE_NAME]

The music

1.3.2.5 char tileSet[MAX_LENGTH_FILE_NAME]

The tilset

1.3.2.6 int timer_level

The timer level

1.3.2.7 int width

The width

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

- structures.h

1.4 Map Struct Reference

```
#include <structures.h>
```

Collaboration diagram for Map:

Data Fields

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

1.4.1 Detailed Description

The map structure

1.4.2 Field Documentation

1.4.2.1 Level* lvl

The level

1.4.2.2 int screenHeight

The screen height

1.4.2.3 int screenWidth

The Screen width

1.4.2.4 int xScroll

The xscroll

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

- structures.h

2 File Documentation

2.1 const.h File Reference

contient les constantes du programme

Macros

- #define TILE_SIZE 16
- #define NB_TILES_X 80
- #define NB_TILES_Y 45
- #define SCREEN_WIDTH TILE_SIZE * NB_TILES_X
- #define SCREEN_HEIGHT TILE_SIZE * NB_TILES_Y
- #define TILESET_SIZE 20
- #define OPTIONS_PER_COLUMN 9
- #define FPS 60
- #define SCROLLING_MARGIN 2
- #define MAX_LENGTH_FILE_NAME 100
- #define min(a, b) (a<=b?a:b)

Enumerations

- enum {
  VOID =0, **GROUND**, **COIN** =7, **ROCK**,
  SPRING, **HAMMER**, **HEART**, **ADDLIFE**,
  ENEMY, **CANON_R**, **CANON_L**, **CANON_B**,
  TREE, **FLOWER**, **CLOUD**, **PLATFORMV**,
  PLATFORMH }
- enum { **RIGHT**, **LEFT**, **UP**, **DOWN** }

2.1.1 Detailed Description

contient les constantes du programme

Author

Xavier COPONET, Glenn HERROU

Date

2014-02-27

2.1.2 Macro Definition Documentation

2.1.2.1 #define FPS 60

The FPS

2.1.2.2 #define MAX_LENGTH_FILE_NAME 100

The size max of the filenames

2.1.2.3 #define min(*a*, *b*) (a<=b?a:b)

mix

2.1.2.4 #define NB_TILES_X 80

The number of tile on x

2.1.2.5 #define NB_TILES_Y 45

The number of tile on y

2.1.2.6 #define OPTIONS_PER_COLUMN 9

The number of options per column

2.1.2.7 #define SCREEN_HEIGHT TILE_SIZE * NB_TILES_Y

The screen height

2.1.2.8 #define SCREEN_WIDTH TILE_SIZE * NB_TILES_X

The screen width

2.1.2.9 #define SCROLLING_MARGIN 2

The scrolling margin

2.1.2.10 #define TILE_SIZE 16

The tile size

2.1.2.11 #define TILESET_SIZE 20

The tilset size

2.2 file.c File Reference

Functions to access to the files.

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

Functions

- FILE * openFile (char name[], char mode[])
- int closeFile (FILE *ptr_file)

2.2.1 Detailed Description

Functions to access to the files.

Author

Remi BERTHO, Glenn HERROU

Date

2014-05-12

2.2.2 Function Documentation

2.2.2.1 int closeFile (FILE * *ptr_file*)

Close the given file

Parameters

in	*ptr_file	the file

Returns

0 if the file has been succesfully closed, 1 otherwise

2.2.2.2 FILE * openFile (char *name[]*, char *mode[]*)

Open a file which path is name with the given mode

Parameters

in	*name[]*	name of the file

in	*mode[]*	the opening mode

Returns

a pointer if the opening has been succesfull, NULL otherwise

2.3 file.h File Reference

file.c header

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

Functions

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

2.3.1 Detailed Description

file.c header

Author

Remi BERTHO, Glenn HERROU

Date

2014-05-12

2.3.2 Function Documentation

2.3.2.1 int closeFile (FILE * *ptr_file*)

Close the given file

Parameters

in	**ptr_file*	the file

Returns

0 if the file has been succesfully closed, 1 otherwise

2.3.2.2 FILE* openFile (char *name[]*, char *mode[]*)

Open a file which path is name with the given mode

Parameters

in	*name[]*	name of the file

in	*mode[]*	the opening mode

Returns

a pointer if the opening has been succesfull, NULL otherwise

2.4 file_level.c File Reference

Management of the level files.

```
#include "file_level.h"
```

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)
- Level ∗ adaptSizeLevel (Level ∗lvl)
- int searchEndLevel (Level ∗lvl)

2.4.1 Detailed Description

Management of the level files.

Author

Remi BERTHO, Glenn HERROU

Date

2014-05-12

Version

2.0

2.4.2 Function Documentation

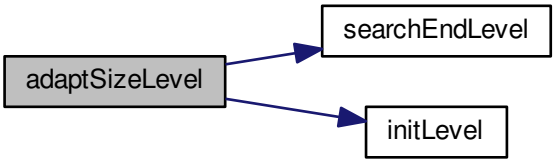
2.4.2.1 Level ∗ adaptSizeLevel (Level ∗ *lvl*)

Adapt the width of a level

Parameters

in,out	*lvl*	the level to adapt

Here is the call graph for this function:



2.4.2.2 void closeLevel (Level * *lvl*)

Close a level by freeing its allocated memory

Parameters

out	*lvl*	The level

2.4.2.3 void closeLevelList (char ** *level_names*, int *nb_lvl*)

Free the array containing the list of the levels

Parameters

in,out	*level_names*	the list of existing levels
in	*nb_lvl*	the number of existing levels

2.4.2.4 Level * initLevel (Level * *lvl*)

Initialize a level. The width and the Height of the level must be stored in the level before calling this function.

Parameters

out	*lvl*	The level

Returns

a pointer on the level initialized

2.4.2.5 Level * openLevel (char * *file_name*)

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

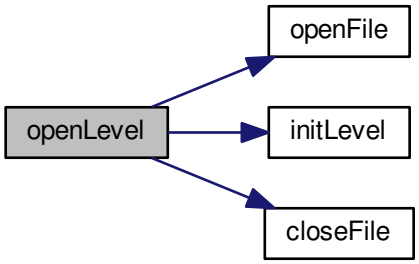
Parameters

in	*file_name*	the name of the level file

Returns

a pointer to the level created

Here is the call graph for this function:



2.4.2.6 char ** readLevelFile (int * *nb_lvl*)

Read the file including the list of existing levels

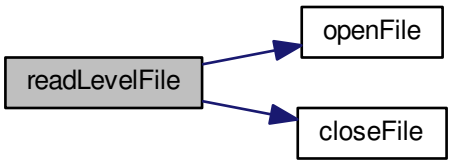
Parameters

out	*nb_lvl*	the number of level in the file

Returns

a pointer to an array of strings containing the list of the levels

Here is the call graph for this function:



2.4.2.7 int searchEndLevel (Level * *lvl*)

Search the end of a level

Parameters

in,out	*lvl*	the level

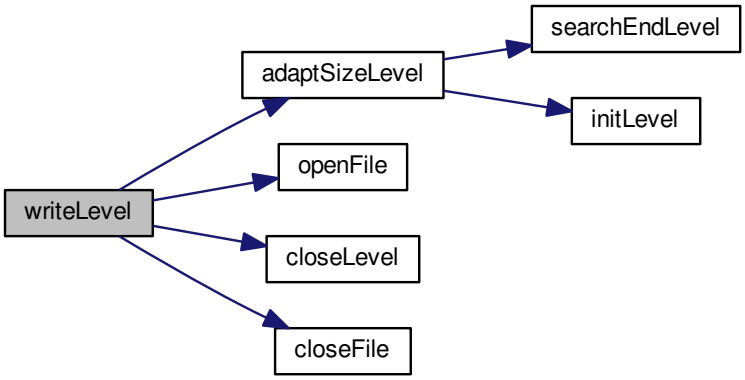
2.4.2.8 void writeLevel (char * *file_name*, Level * *lvl*)

Write the given level in the given file

Parameters

in	*lvl*	the file
in	*file_name*	the name of the file

Here is the call graph for this function:



2.5 file_level.h File Reference

file_level.c header

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

Macros

- #define BUFFER_SIZE 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)
- Level * adaptSizeLevel (Level *lvl)
- int searchEndLevel (Level *lvl)

2.5.1 Detailed Description

file_level.c header

Author

Remi BERTHO, Glenn HERROU

Date

2014-05-12

Version

2.0

2.5.2 Macro Definition Documentation

2.5.2.1 #define BUFFER_SIZE 2

The buffer size

2.5.3 Function Documentation

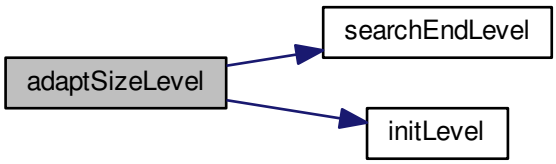
2.5.3.1 Level* adaptSizeLevel (Level * *lvl*)

Adapt the width of a level

Parameters

in,out	*lvl*	the level to adapt

Here is the call graph for this function:



2.5.3.2 void closeLevel (Level * *lvl*)

Close a level by freeing its allocated memory

Parameters

out	*lvl*	The level

2.5.3.3 void closeLevelList (char ** *level_names*, int *nb_lvl*)

Free the array containing the list of the levels

Parameters

in,out	*level_names*	the list of existing levels
in	*nb_lvl*	the number of existing levels

2.5.3.4 Level* initLevel (Level * *lvl*)

Initialize a level. The width and the Height of the level must be stored in the level before calling this function.

Parameters

out	*lvl*	The level

Returns

a pointer on the level initialized

2.5.3.5 Level* openLevel (char * *file_name*)

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

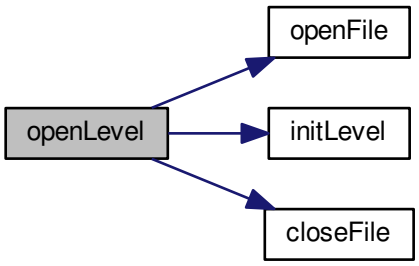
Parameters

in	*file_name*	the name of the level file

Returns

a pointer to the level created

Here is the call graph for this function:



2.5.3.6 char** readLevelFile (int * *nb_lvl*)

Read the file including the list of existing levels

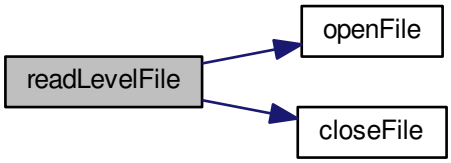
Parameters

out	*nb_lvl*	the number of level in the file

Returns

a pointer to an array of strings containing the list of the levels

Here is the call graph for this function:



2.5.3.7 int searchEndLevel (Level * *lvl*)

Search the end of a level

Parameters

in,out	*lvl*	the level

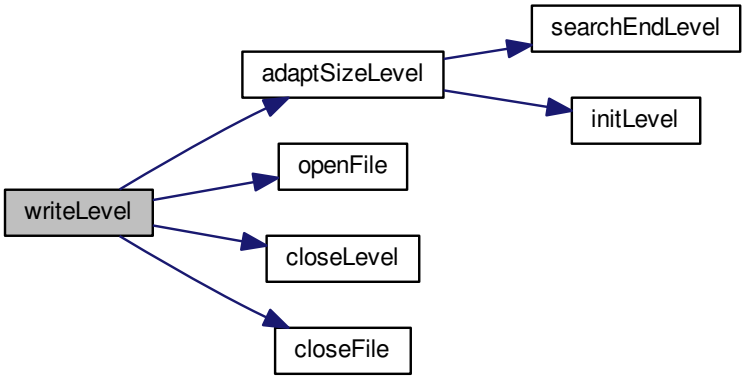
2.5.3.8 void writeLevel (char * *file_name*, Level * *lvl*)

Write the given level in the given file

Parameters

in	*lvl*	the file
in	*file_name*	the name of the file

Here is the call graph for this function:



2.6 game.c File Reference

Contain the main functions of the game.

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

Functions

- void play (SDL_Surface ∗screen, char ∗level_name, SDLKey ∗kc)
- void printConfirmation (SDL_Surface ∗screen, Input ∗in, int ∗go)

2.6.1 Detailed Description

Contain the main functions of the game.

Author

Xavier COPONET, Glenn HERROU

Date

2014-05-15

2.6.2 Function Documentation

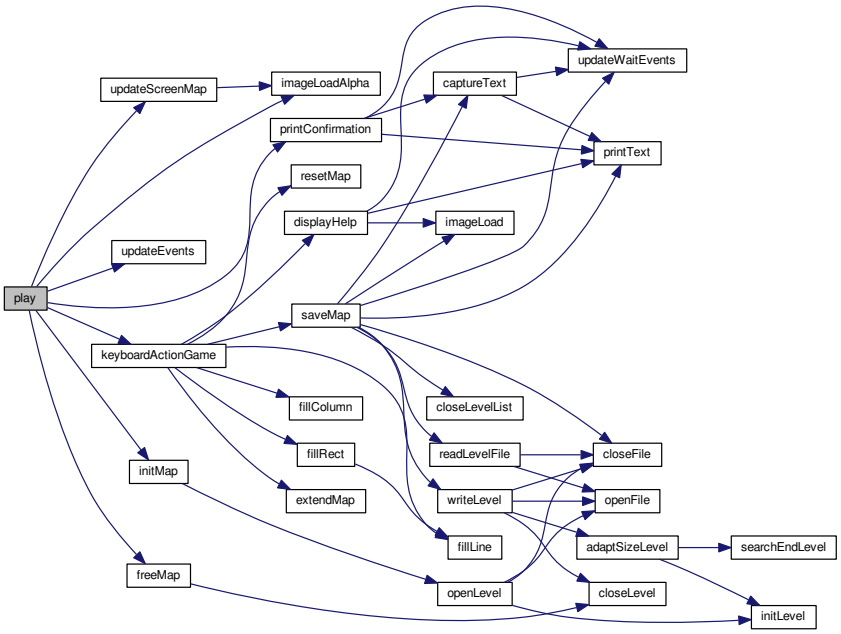
2.6.2.1 void play (SDL_Surface ∗ *screen,* char ∗ *level_name,* SDLKey ∗ *kc*)

Include the main loop of the game

Parameters

in,out	*screen*	The screen of the game
in	*level_name*	The name of the level
in	*kc*	The keyboard configuration

Here is the call graph for this function:



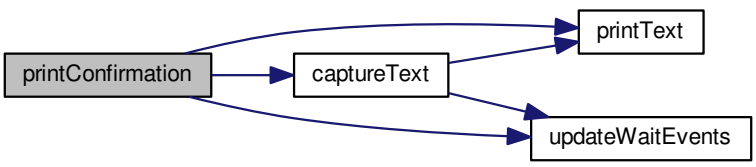
2.6.2.2 void printConfirmation (SDL_Surface * *screen*, Input * *in*, int * *go*)

Display the confirmation screen, before leaving the edition of a level

Parameters

out	*screen*	The screen of the game
in	*in*	The input structure
out	*go*	The main loop activation

Here is the call graph for this function:



2.7 game.h File Reference

game.c header

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include "const.h"
#include "structures.h"
#include "text.h"
#include "share.h"
#include "file_level.h"
#include "image.h"
#include "map.h"
#include "input.h"
```

Functions

- void play (SDL_Surface *screen, char *level_name, SDLKey *kc)
- void printConfirmation (SDL_Surface *screen, Input *in, int *go)

2.7.1 Detailed Description

game.c header

Author

Xavier COPONET, Glenn HERROU

Date

2014-05-15

2.7.2 Function Documentation

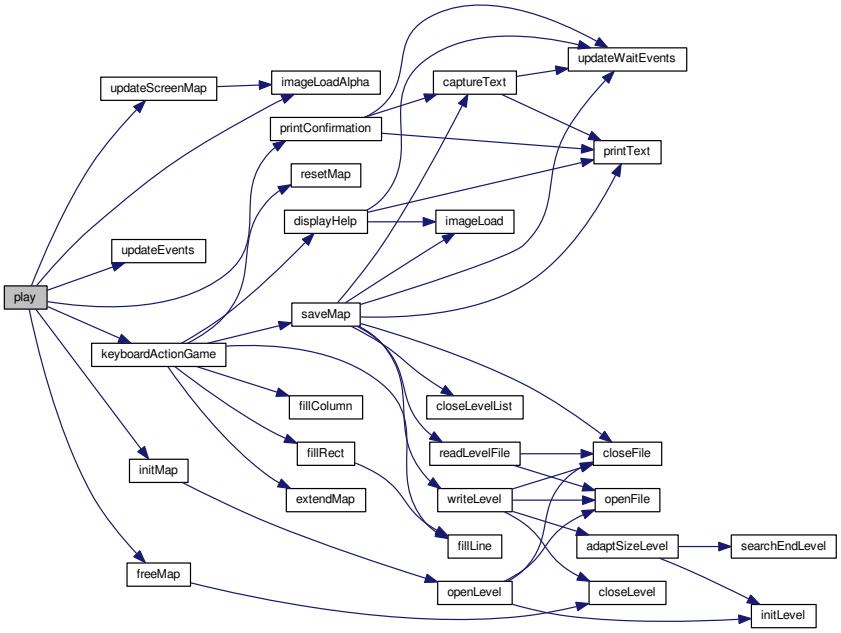
2.7.2.1 void play (SDL_Surface * *screen*, char * *level_name*, SDLKey * *kc*)

Include the main loop of the game

Parameters

in,out	*screen*	The screen of the game
in	*level_name*	The name of the level
in	*kc*	The keyboard configuration

Here is the call graph for this function:



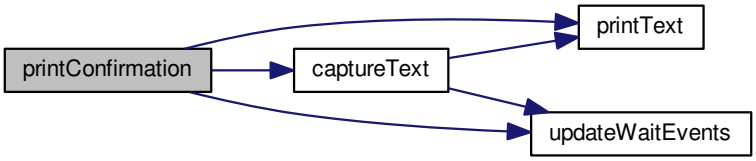
2.7.2.2 void printConfirmation (SDL_Surface * *screen*, Input * *in*, int * *go*)

Display the confirmation screen, before leaving the edition of a level

Parameters

out	*screen*	The screen of the game
in	*in*	The input structure
out	*go*	The main loop activation

Here is the call graph for this function:



2.8 image.c File Reference

Contain the functions managing the images.

```
#include "image.h"
```

Functions

- SDL_Surface * imageLoad (char *file_name)
- SDL_Surface * imageLoadAlpha (char *file_name)

2.8.1 Detailed Description

Contain the functions managing the images.

Author

Rémi BERTHO

Date

2014-02-27

2.8.2 Function Documentation

2.8.2.1 SDL_Surface * imageLoad (char * *file_name*)

Load an image

Parameters

in	*file_name*	the name of the image file

Returns

a pointer on the SDL_Surface created

2.8.2.2 SDL_Surface * imageLoadAlpha (char * *file_name*)

Load an image with alpha management

Parameters

in	*file_name*	the name of the image file

Returns

a pointer on the SDL_Surface created

2.9 image.h File Reference

contient les fonction liées aux images

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include "const.h"
```

Functions

- SDL_Surface * imageLoad (char *file_name)
- SDL_Surface * imageLoadAlpha (char *file_name)

2.9.1 Detailed Description

contient les fonction liées aux images

Author

Rémi BERTHO

Date

2014-02-27

2.9.2 Function Documentation

2.9.2.1 SDL_Surface* imageLoad (char * *file_name*)

Load an image

Parameters

in	*file_name*	the name of the image file

Returns

a pointer on the SDL_Surface created

2.9.2.2 SDL_Surface* imageLoadAlpha (char * *file_name*)

Load an image with alpha management

Parameters

in	*file_name*	the name of the image file

Returns

a pointer on the SDL_Surface created

2.10 input.c File Reference

```
#include "input.h"
```

Functions

- void updateEvents (Input *in)
- void keyboardActionGame (SDL_Surface *screen, Input *in, Map *m, Cursor *cursor, SDLKey *kc)
- int updateWaitEvents (Input *in)
- int keyboardActionMenu (Input *in, int *cursorPos, int *select, int nb_options)

2.10.1 Detailed Description

Author

Xavier COPONET, Glenn HERROU

Date

2014-03-18

2.10.2 Function Documentation

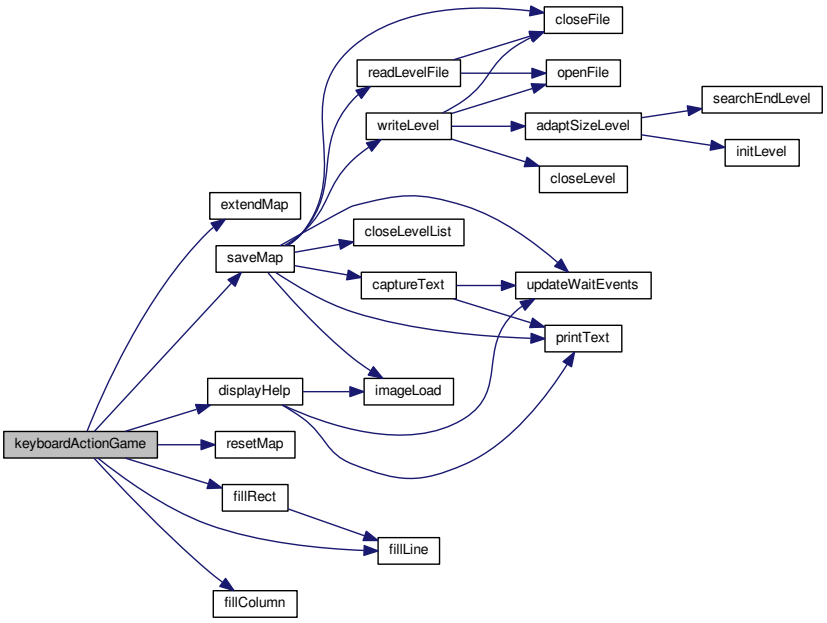
2.10.2.1 void keyboardActionGame (SDL_Surface * *screen*, Input * *in*, Map * *m*, Cursor * *cursor*, SDLKey * *kc*)

perform action commanded by keyboard action

Parameters

in,out	*screen*	The screen of the game
in,out	*in*	the input structure
in,out	*m*	the map to update
in,out	*cursor*	the cursor structure
in	*kc*	the keyboard bindings

Here is the call graph for this function:



2.10.2.2 void keyboardActionMenu (Input * *in*, int * *cursorPos*, int * *select*, int *nb_options*)

perform menu action commanded by keyboard action

Parameters

in	*in*	the input structure
out	*cursorPos*	the cursor position
out	*select*	boolean about selecting the option or quit to title screen
in	*nb_options*	the number of options of the menu

2.10.2.3 void updateEvents (Input * *in*)

get keyboard input with a SDL_PollEvent

Parameters

out	*in*	the input structure

2.10.2.4 int updateWaitEvents (Input * *in*)

get keyboard input with a SDL_WaitEvent

Parameters

out	*in*	the input structure

Returns

1 if a key is activated

2.11 main.c File Reference

```
#include "game.h"
#include "const.h"
#include "menu.h"
#include "menu_level.h"
#include "menu_option.h"
#include "option.h"
```

Functions

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

2.11.1 Detailed Description

Author

Xavier COPONET

Date

2014-02-27

2.11.2 Function Documentation

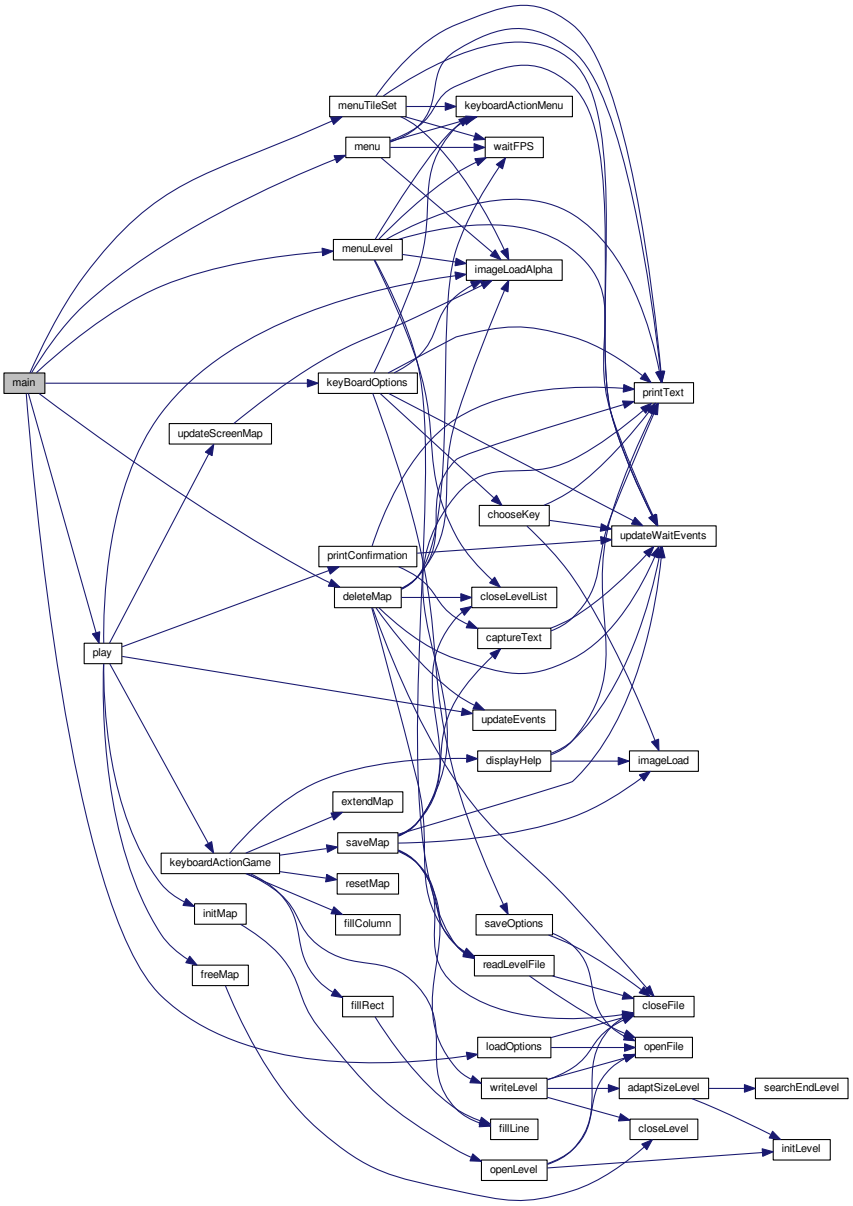
2.11.2.1 int main (int *argc*, char * *argv[]*)

Main

Parameters

in,out	*argc*	argc
in,out	*argv*	argv

Here is the call graph for this function:



2.12 map.c File Reference

Management of the map.

```
#include "map.h"
```

Functions

- void updateScreenMap (SDL_Surface *screen, Map *m, char *tileset, Cursor *cursor)
- Map * initMap (SDL_Surface *screen, char *level_name)
- void fillLine (Map *m, int line, int column, char tileID)
- void fillColumn (Map *m, int line, int column, char tileID)

- void fillRect (Map *m, int line, int column, char tileID)
- void displayHelp (SDL_Surface *screen, SDLKey *kc)
- void saveMap (SDL_Surface *screen, Map *m)
- void deleteMap (SDL_Surface *screen, char *map_name, char *map_path)
- void extendMap (Map *m)
- void resetMap (Map *m)
- void freeMap (Map *m)

2.12.1 Detailed Description

Management of the map.

Author

Xavier COPONET, Glenn HERROU

Date

2014-05-18

2.12.2 Function Documentation

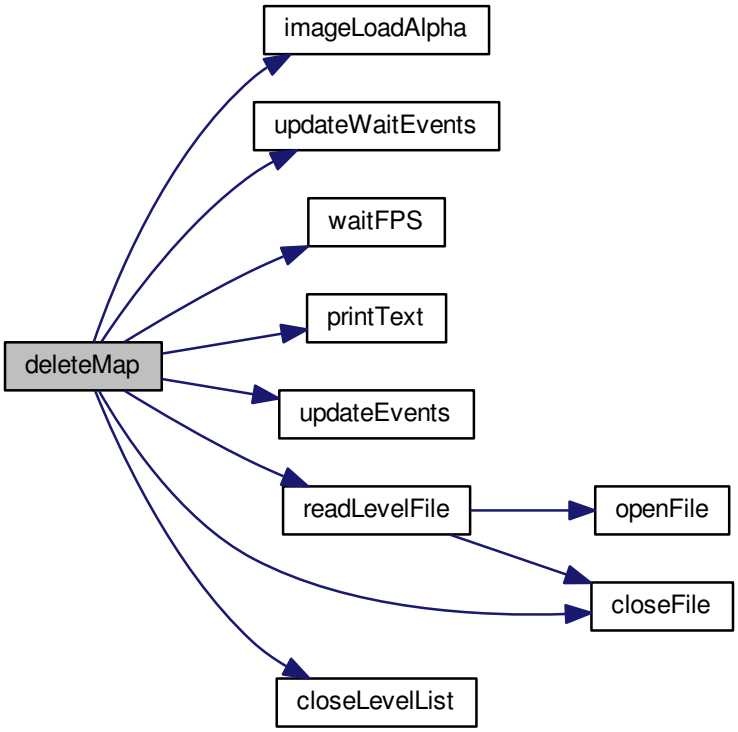
2.12.2.1 void deleteMap (SDL_Surface * *screen*, char * *map_name*, char * *map_path*)

Delete a the map file and update the level list file

Parameters

in,out	*screen*	The screen of the game
in	*map_name*	The name of the map to delete
in	*map_path*	The path to the file of the map to delete

Here is the call graph for this function:



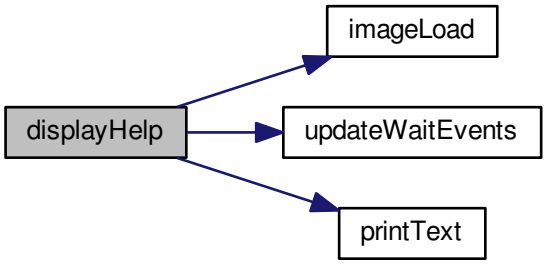
2.12.2.2 void displayHelp (SDL_Surface * *screen*, SDLKey * *kc*)

Display the list of keybindings on the screen

Parameters

in,out	*screen*	The screen of the game
in	*kc*	The array containing the keybindings

Here is the call graph for this function:



2.12.2.3 void extendMap (Map * *m*)

Extend the width of a map

Parameters

out	*m*	The map to extend

2.12.2.4 void fillColumn (Map * *m*, int *line*, int *column*, char *tileID*)

Fill a column with the current tile. The filling stops if another non-void tile is reached. This function works only with ground tiles

Parameters

in	*m*	The map
in	*line*	The line pointed by the cursor
in	*column*	The column pointed by the cursor
in	*tileID*	the tile ID

2.12.2.5 void fillLine (Map * *m*, int *line*, int *column*, char *tileID*)

Fill a line with the current tile. The filling stops if another non-void tile is reached. This function works only with ground tiles

Parameters

in	*m*	The map
in	*line*	The line pointed by the cursor
in	*column*	The column pointed by the cursor
in	*tileID*	the tile ID

2.12.2.6 void fillRect (Map * *m*, int *line*, int *column*, char *tileID*)

Fill a rectangle with the current tile. The filling stops if another non-void tile is reached. This function works only with ground tiles

Parameters

in	*m*	The map
in	*line*	The line pointed by the cursor
in	*column*	The column pointed by the cursor
in	*tileID*	the tile ID

Here is the call graph for this function:

2.12.2.7 void freeMap (Map * *m*)

Free memory allocated to the map

Parameters

in,out	*m*	the map

Here is the call graph for this function:

2.12.2.8 Map * initMap (SDL_Surface * *screen*, char * *level_name*)

initialize the map

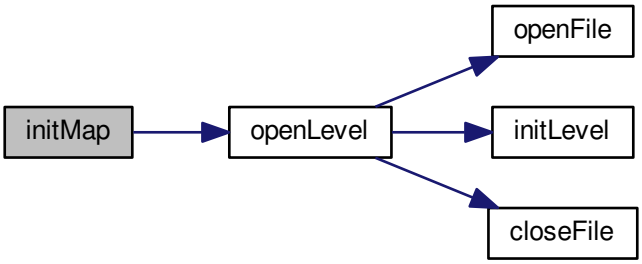
Parameters

in	*screen*	game screen
in	*level_name*	lvl name

Returns

a pointer on the map

Here is the call graph for this function:



2.12.2.9 void resetMap (Map * *m*)

Fill a map with blank tiles. This function doesn't change the current map file.

Parameters

in,out	*m*	The map to reinit

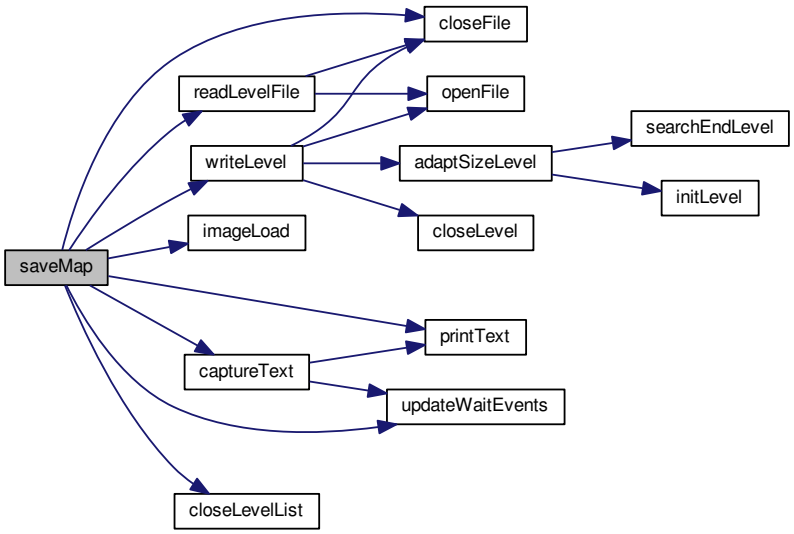
2.12.2.10 void saveMap (SDL_Surface * *screen*, Map * *m*)

Save the map in a new file and update the file 'level' containing the map list

Parameters

in,out	*screen*	The screen of the game
in	*m*	The map to save

Here is the call graph for this function:



2.12.2.11 void updateScreenMap (SDL_Surface * *screen*, Map * *m*, char * *tileset*, Cursor * *cursor*)

update and display the map on the screen

Parameters

in,out	*screen*	The screen of the game
in	*m*	The map
in	*tileset*	The level tileset
in	*cursor*	The mouse cursor

Here is the call graph for this function:

2.13 map.h File Reference

map.c header

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include "image.h"
#include "file_level.h"
#include "input.h"
#include "text.h"
#include "share.h"
```

Functions

- void updateScreenMap (SDL_Surface *screen, Map *m, char *tileset, Cursor *cursor)
- Map * initMap (SDL_Surface *screen, char *level_name)
- void fillLine (Map *m, int line, int column, char tileID)
- void fillColumn (Map *m, int line, int column, char tileID)
- void fillRect (Map *m, int line, int column, char tileID)
- void displayHelp (SDL_Surface *screen, SDLKey *kc)
- void saveMap (SDL_Surface *screen, Map *m)
- void deleteMap (SDL_Surface *screen, char *map_name, char *map_path)
- void extendMap (Map *m)
- void resetMap (Map *m)
- void freeMap (Map *m)

2.13.1 Detailed Description

map.c header

Author

Xavier COPONET, Glenn HERROU

Date

2014-05-18

2.13.2 Function Documentation

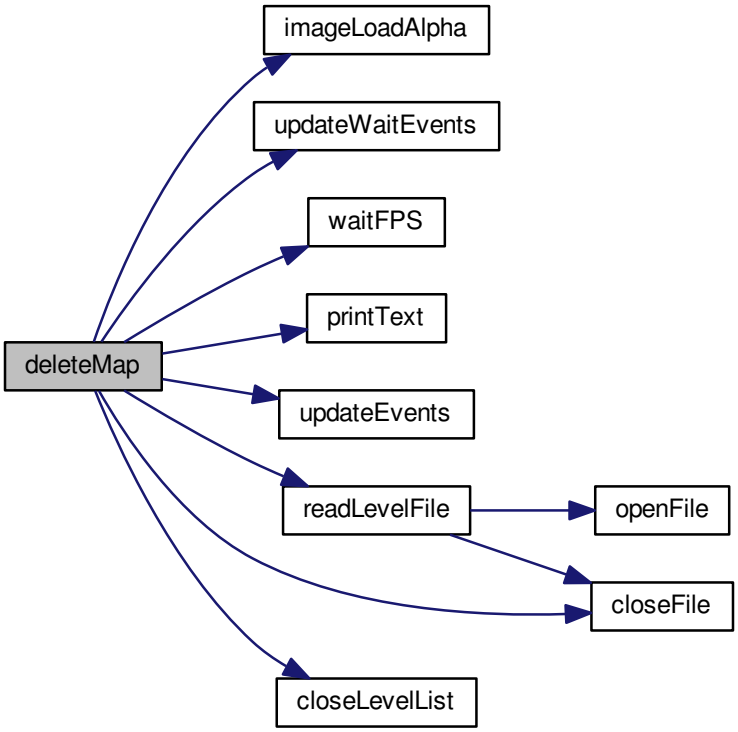
2.13.2.1 void deleteMap (SDL_Surface * *screen*, char * *map_name*, char * *map_path*)

Delete a the map file and update the level list file

Parameters

in,out	*screen*	The screen of the game
in	*map_name*	The name of the map to delete
in	*map_path*	The path to the file of the map to delete

Here is the call graph for this function:



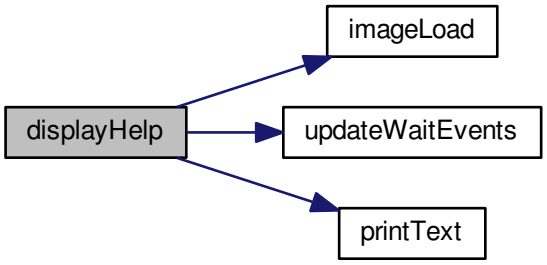
2.13.2.2 void displayHelp (SDL_Surface * *screen*, SDLKey * *kc*)

Display the list of keybindings on the screen

Parameters

in,out	*screen*	The screen of the game
in	*kc*	The array containing the keybindings

Here is the call graph for this function:



2.13.2.3 void extendMap (Map * *m*)

Extend the width of a map

Parameters

out	*m*	The map to extend

2.13.2.4 void fillColumn (Map * *m*, int *line*, int *column*, char *tileID*)

Fill a column with the current tile. The filling stops if another non-void tile is reached. This function works only with ground tiles

Parameters

in	*m*	The map
in	*line*	The line pointed by the cursor
in	*column*	The column pointed by the cursor
in	*tileID*	the tile ID

2.13.2.5 void fillLine (Map * *m*, int *line*, int *column*, char *tileID*)

Fill a line with the current tile. The filling stops if another non-void tile is reached. This function works only with ground tiles

Parameters

in	*m*	The map
in	*line*	The line pointed by the cursor
in	*column*	The column pointed by the cursor
in	*tileID*	the tile ID

2.13.2.6 void fillRect (Map * *m*, int *line*, int *column*, char *tileID*)

Fill a rectangle with the current tile. The filling stops if another non-void tile is reached. This function works only with ground tiles

Parameters

in	*m*	The map
in	*line*	The line pointed by the cursor
in	*column*	The column pointed by the cursor
in	*tileID*	the tile ID

Here is the call graph for this function:

2.13.2.7 void freeMap (Map * *m*)

Free memory allocated to the map

Parameters

in,out	*m*	the map

Here is the call graph for this function:

2.13.2.8 Map* initMap (SDL_Surface * *screen*, char * *level_name*)

initialize the map

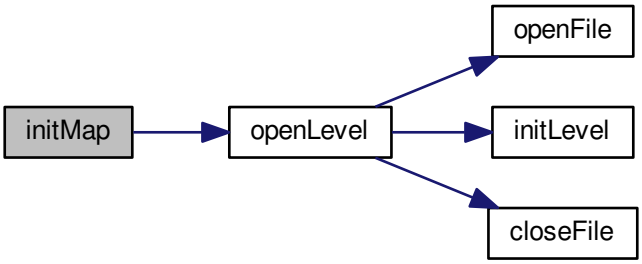
Parameters

in	*screen*	game screen
in	*level_name*	lvl name

Returns

a pointer on the map

Here is the call graph for this function:



2.13.2.9 void resetMap (Map * *m*)

Fill a map with blank tiles. This function doesn't change the current map file.

Parameters

in,out	*m*	The map to reinit

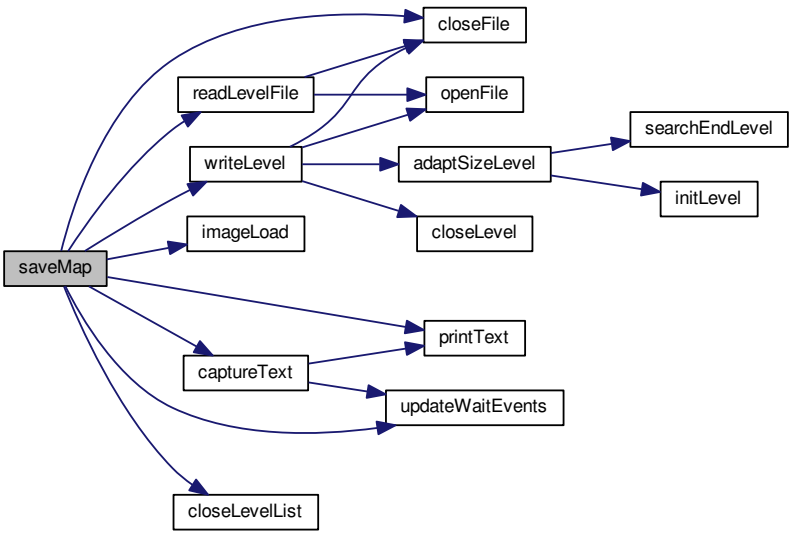
2.13.2.10 void saveMap (SDL_Surface * *screen*, Map * *m*)

Save the map in a new file and update the file 'level' containing the map list

Parameters

in,out	*screen*	The screen of the game
in	*m*	The map to save

Here is the call graph for this function:



2.13.2.11 void updateScreenMap (SDL_Surface * *screen*, Map * *m*, char * *tileset*, Cursor * *cursor*)

update and display the map on the screen

Parameters

in,out	*screen*	The screen of the game
in	*m*	The map
in	*tileset*	The level tileset
in	*cursor*	The mouse cursor

Here is the call graph for this function:

2.14 menu.c File Reference

Contain the main menu management.

```
#include "menu.h"
```

Functions

- int menu (SDL_Surface *screen, int *choice, int *go)

- int menuTileSet (SDL_Surface *screen, char tileSet_name[MAX_LENGTH_FILE_NAME])

2.14.1 Detailed Description

Contain the main menu management.

Author

Glenn HERROU

Date

2014-04-20

2.14.2 Function Documentation

2.14.2.1 int menu (SDL_Surface * *screen*, int * *choice*, int * *go*)

Display the menu on the screen

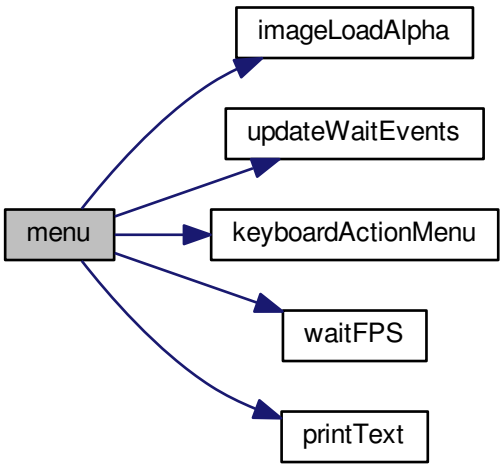
Parameters

out	*screen*	the screen of the game
out	*choice*	the option selected
out	*go*	the main loop validation

Returns

1 if an option has been selected

Here is the call graph for this function:



2.14.2.2 int menuTileSet (SDL_Surface * *screen*, char *tileSet_name[MAX_LENGTH_FILE_NAME]*)

Display the tileset menu on the screen

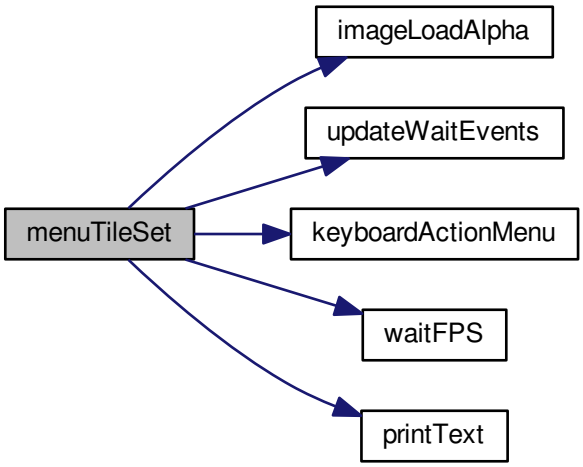
Parameters

out	*screen*	the screen of the game
out	*tileSet_name*	The name of the tileSet selected

Returns

1 if a tileset has been selected

Here is the call graph for this function:



2.15 menu.h File Reference

header de menu.c

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include <SDL/SDL_ttf.h>
#include "const.h"
#include "text.h"
#include "share.h"
#include "image.h"
#include "input.h"
```

Functions

- int menu (SDL_Surface *screen, int *choice, int *go)
- int menuTileSet (SDL_Surface *screen, char tileSet_name[MAX_LENGTH_FILE_NAME])

2.15.1 Detailed Description

header de menu.c

Author

Xavier COPONET

Date

2014-02-27

2.15.2 Function Documentation

2.15.2.1 int menu (SDL_Surface * *screen*, int * *choice*, int * *go*)

Display the menu on the screen

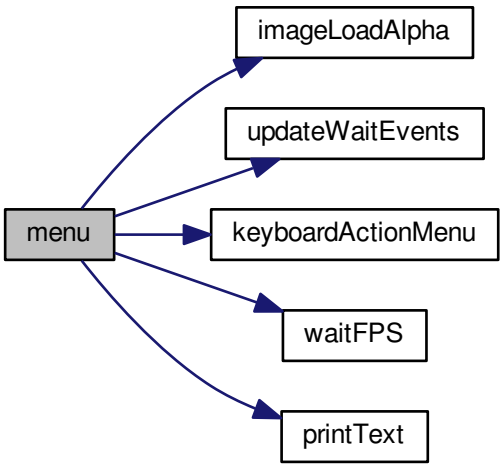
Parameters

out	*screen*	the screen of the game
out	*choice*	the option selected
out	*go*	the main loop validation

Returns

1 if an option has been selected

Here is the call graph for this function:



2.15.2.2 int menuTileSet (SDL_Surface * *screen*, char *tileSet_name[MAX_LENGTH_FILE_NAME]*)

Display the tileset menu on the screen

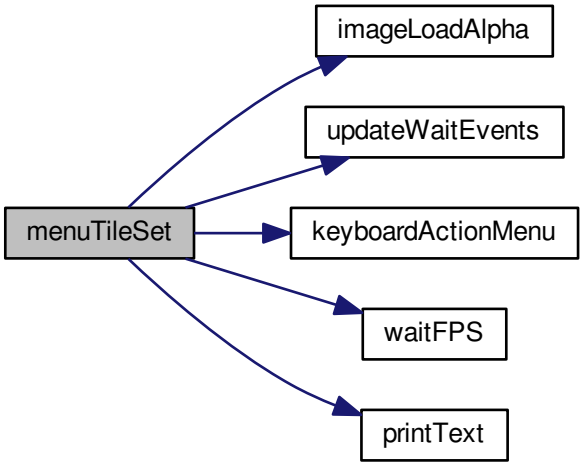
Parameters

out	*screen*	the screen of the game
out	*tileSet_name*	The name of the tileSet selected

Returns

1 if a tileset has been selected

Here is the call graph for this function:



2.16 menu_level.c File Reference

Contain the level menu management.

```
#include "menu_level.h"
```

Functions

- int menuLevel (SDL_Surface *screen, char level_name[MAX_LENGTH_FILE_NAME], char level_path[MAX_LENGTH_FILE_NAME])

2.16.1 Detailed Description

Contain the level menu management.

Author

Remi BERTHO, Glenn HERROU

Date

2014-05-10

Version

2.0

2.16.2 Function Documentation

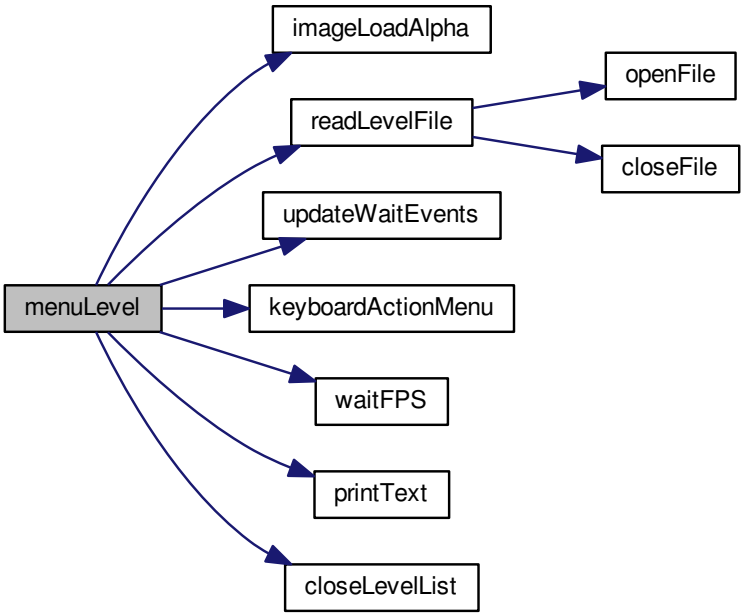
2.16.2.1 int menuLevel (SDL_Surface * *screen*, char *level_name[MAX_LENGTH_FILE_NAME]*, char *level_path[MAX_LENGTH_FILE_NAME]*)

Display the level menu on the screen

Parameters

out	*screen*	The screen of the game
out	*level_name*	The level name
out	*level_path*	The level file path

Here is the call graph for this function:



2.17 menu_level.h File Reference

menu_level.c header

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include <SDL/SDL_ttf.h>
#include "const.h"
#include "file_level.h"
#include "share.h"
#include "text.h"
#include "image.h"
#include "input.h"
```

Functions

- int menuLevel (SDL_Surface *screen, char level_name[MAX_LENGTH_FILE_NAME], char level_path[MAX_LENGTH_FILE_NAME])

2.17.1 Detailed Description

menu_level.c header

Author

Remi BERTHO, Glenn HERROU

Date

2014-05-10

Version

2.0

2.17.2 Function Documentation

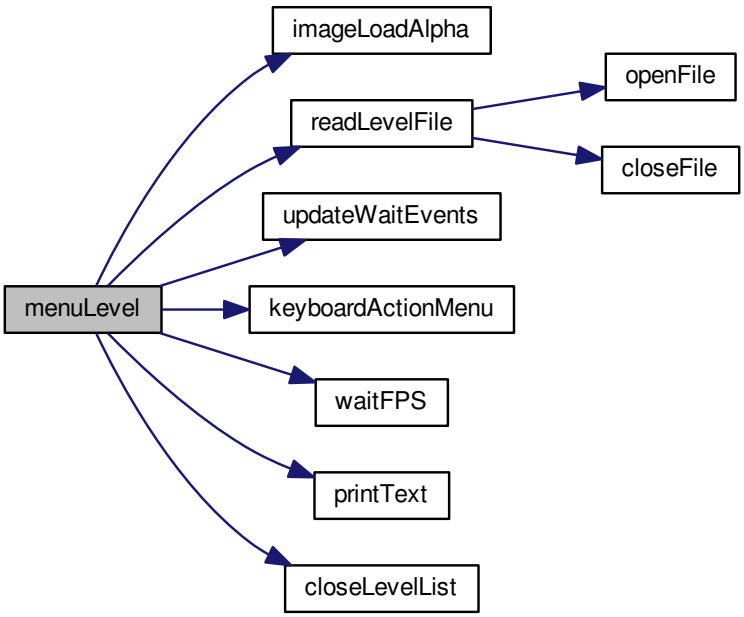
2.17.2.1 int menuLevel (SDL_Surface * *screen*, char *level_name[MAX_LENGTH_FILE_NAME]*, char *level_path[MAX_LENGTH_FILE_NAME]*)

Display the level menu on the screen

Parameters

out	*screen*	The screen of the game
out	*level_name*	The level name
out	*level_path*	The level file path

Here is the call graph for this function:



2.18 menu_option.c File Reference

contain the option menu functions

```
#include "menu_option.h"
```

Functions

- int menuOptions (SDL_Surface ∗screen, int ∗go, SDLKey ∗kc)
- void keyBoardOptions (SDL_Surface ∗screen, int ∗go, SDLKey ∗kc)
- void chooseKey (SDL_Surface ∗screen, Input ∗in, char ∗action, SDLKey ∗kc, int nb)

2.18.1 Detailed Description

contain the option menu functions

Author

Xavier COPONET

Date

2014-04-27

2.18.2 Function Documentation

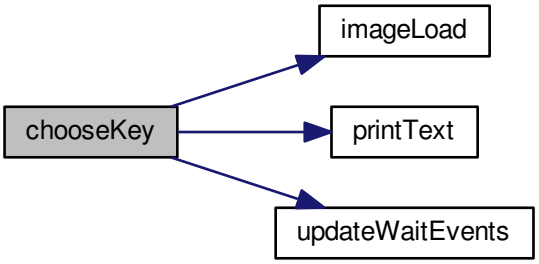
2.18.2.1 void chooseKey (SDL_Surface * *screen*, Input * *in*, char * *action*, SDLKey * *kc*, int *nb*)

print the message asking the player to choose a key and wait until the player press a key and deals with this key

Parameters

out	*screen*	the game screen
in,out	*in*	the input structure
in	*action*	the action which the key has to be choosen
out	*kc*	the keyboard configuration
in	*nb*	the number of the action

Here is the call graph for this function:



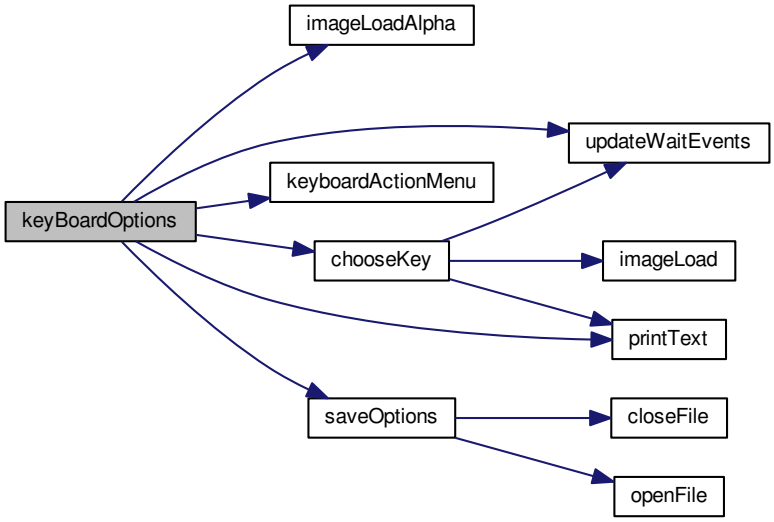
2.18.2.2 void keyBoardOptions (SDL_Surface * *screen*, int * *go*, SDLKey * *kc*)

print the keyboard options and deals with the user choises

Parameters

out	*screen*	the game screen
in,out	*go*	main loop validation
in,out	*kc*	the keyboard config

Here is the call graph for this function:



2.18.2.3 int menuOptions (SDL_Surface * *screen*, int * *go*, SDLKey * *kc*)

print the option menu on the screen

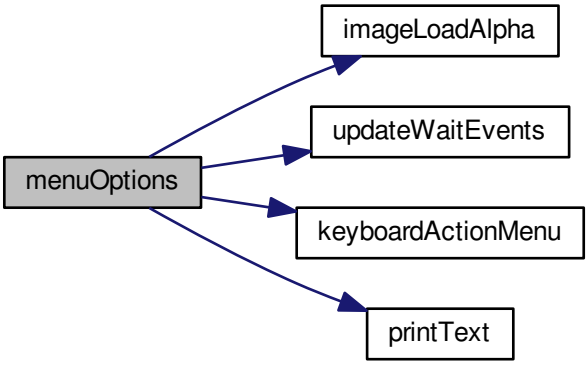
Parameters

out	*screen*	the game screen
in,out	*go*	main loop validation
in,out	*kc*	the keyboard configuration

Returns

the number of the option which is choosen, -1 if esc

Here is the call graph for this function:



2.19 menu_option.h File Reference

menu_option.c header

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include <SDL/SDL_ttf.h>
#include "const.h"
#include "text.h"
#include "share.h"
#include "image.h"
#include "input.h"
#include "option.h"
```

Functions

- int menuOptions (SDL_Surface ∗screen, int ∗go, SDLKey ∗kc)
- void keyBoardOptions (SDL_Surface ∗screen, int ∗go, SDLKey ∗kc)
- void chooseKey (SDL_Surface ∗screen, Input ∗in, char ∗action, SDLKey ∗kc, int nb)

2.19.1 Detailed Description

menu_option.c header

Author

Xavier COPONET

Date

2014-04-27

2.19.2 Function Documentation

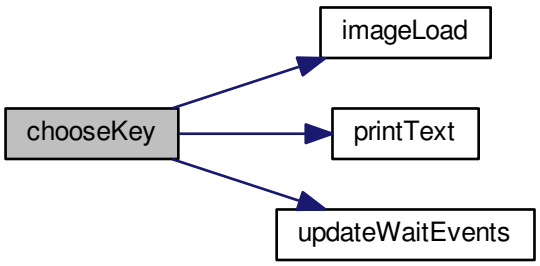
2.19.2.1 void chooseKey (SDL_Surface * *screen*, Input * *in*, char * *action*, SDLKey * *kc*, int *nb*)

print the message asking the player to choose a key and wait until the player press a key and deals with this key

Parameters

out	*screen*	the game screen
in,out	*in*	the input structure
in	*action*	the action which the key has to be choosen
out	*kc*	the keyboard configuration
in	*nb*	the number of the action

Here is the call graph for this function:



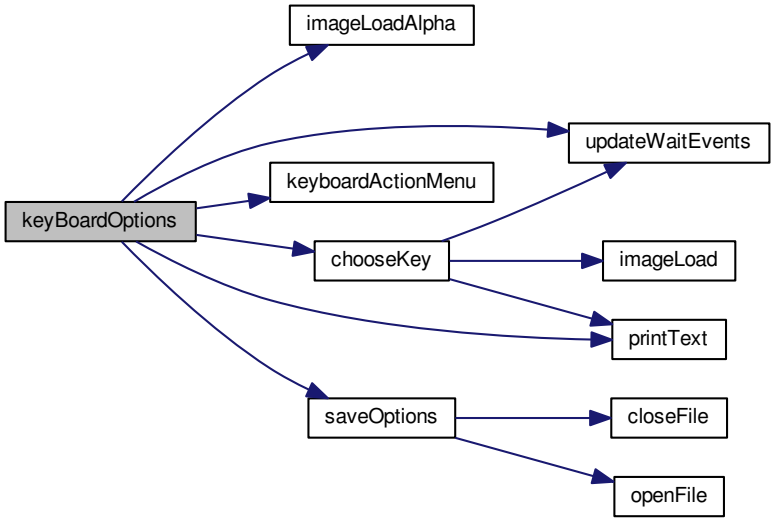
2.19.2.2 void keyBoardOptions (SDL_Surface * *screen*, int * *go*, SDLKey * *kc*)

print the keyboard options and deals with the user choises

Parameters

out	*screen*	the game screen
in,out	*go*	main loop validation
in,out	*kc*	the keyboard config

Here is the call graph for this function:



2.19.2.3 int menuOptions (SDL_Surface * *screen*, int * *go*, SDLKey * *kc*)

print the option menu on the screen

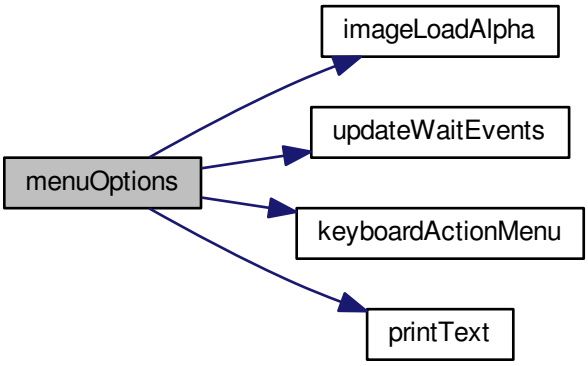
Parameters

out	*screen*	the game screen
in,out	*go*	main loop validation
in,out	*kc*	the keyboard configuration

Returns

the number of the option which is choosen, -1 if esc

Here is the call graph for this function:



2.20 option.c File Reference

contains the functions that manipulate the options

```
#include "option.h"
```

Functions

- void loadOptions (char confFile[], SDLKey *kc)
- void saveOptions (char confFile[], SDLKey *kc)

2.20.1 Detailed Description

contains the functions that manipulate the options

Author

Xavier COPONET, Glenn HERROU

Date

2014-04-28

2.20.2 Function Documentation

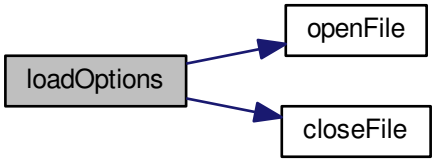
2.20.2.1 void loadOptions (char *confFile[],* SDLKey * *kc*)

Load the options from the config file

Parameters

in	*confFile*	the config file path
out	*kc*	the keyboard configuration structure

Here is the call graph for this function:



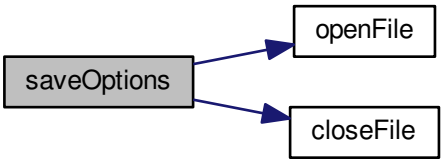
2.20.2.2 void saveOptions (char *confFile[]*, SDLKey * *kc*)

save the options to the config file

Parameters

in	*confFile*	the config file path
in	*kc*	the keyboard configuration structure

Here is the call graph for this function:



2.21 option.h File Reference

option.c header

```
#include "file.h"
#include "structures.h"
#include "input.h"
```

Functions

- void loadOptions (char confFile[], SDLKey *kc)
- void saveOptions (char confFile[], SDLKey *kc)

2.21.1 Detailed Description

option.c header

Author

Xavier COPONET

Date

2014-04-28

2.21.2 Function Documentation

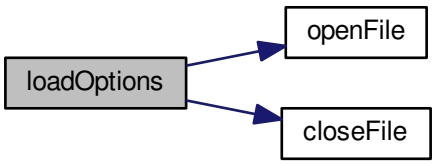
2.21.2.1 void loadOptions (char *confFile[]*, SDLKey * *kc*)

Load the options from the config file

Parameters

in	*confFile*	the config file path
out	*kc*	the keyboard configuration structure

Here is the call graph for this function:



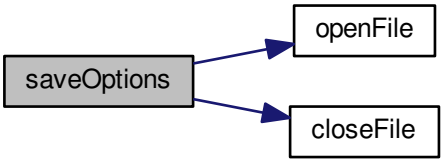
2.21.2.2 void saveOptions (char *confFile[]*, SDLKey * *kc*)

save the options to the config file

Parameters

in	*confFile*	the config file path
in	*kc*	the keyboard configuration structure

Here is the call graph for this function:



2.22 share.c File Reference

Management of FPS rate.

```
#include "share.h"
```

Functions

- void waitFPS (int *previous_time, int *current_time)

2.22.1 Detailed Description

Management of FPS rate.

Author

Remi BERTHO

Date

15/03/14

Version

1.0

2.22.2 Function Documentation

2.22.2.1 void waitFPS (int * *previous_time*, int * *current_time*)

Function managing the fps rate

Parameters

in,out	*previous_time*	The previous time
in,out	*current_time*	The current time

2.23 share.h File Reference

share.c header

```
#include "const.h"
#include <SDL/SDL.h>
```

Functions

- void waitFPS (int *previous_time, int *current_time)

2.23.1 Detailed Description

share.c header

Author

Remi BERTHO

Date

15/03/14

Version

1.0

2.23.2 Function Documentation

2.23.2.1 void waitFPS (int * *previous_time*, int * *current_time*)

Function managing the fps rate

Parameters

in,out	*previous_time*	The previous time
in,out	*current_time*	The current time

2.24 text.c File Reference

Management of the display of text on the screen.

```
#include "text.h"
```

Functions

- void printText (SDL_Surface *screen, SDL_Rect *posText, char *text, int r, int g, int b, char *font, int ptSize, int mode)
- void captureText (SDL_Surface *screen, SDL_Rect posText, char *text, int text_length, int r, int g, int b, char *font, int text_size, int *go)

2.24.1 Detailed Description

Management of the display of text on the screen.

Author

Xavier COPONET, Glenn HERROU

Date

2014-04-27

2.24.2 Function Documentation

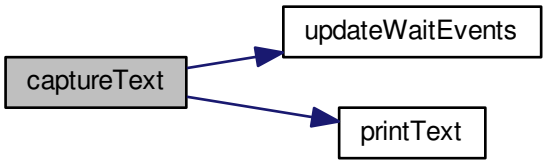
2.24.2.1 void captureText (SDL_Surface * *screen*, SDL_Rect *posText*, char * *text*, int *text_length*, int *r*, int *g*, int *b*, char * *font*, int *text_size*, int * *go*)

Capture the text corresponding to the keyboard inputs and display it on the screen at the given position

Parameters

out	*screen*	The screen of the game
in	*posText*	The position of the text. If NULL, the text is centered
out	*text*	The text to display
in	*r*	red value
in	*g*	green value
in	*b*	blue value
in	*text_length*	the text length
in	*font*	The path to the font file
in	*text_size*	The text size
out	*go*	The main loop validation

Here is the call graph for this function:



2.24.2.2 void printText (SDL_Surface * *screen*, SDL_Rect * *posText*, char * *text*, int *r*, int *g*, int *b*, char * *font*, int *ptSize*, int *mode*)

Display the given text on the screen, at the given position

Parameters

out	*screen*	The screen of the game
in	*posText*	The position of the text. If NULL, the text is centered
in	*text*	The text to display
in	*r*	red value
in	*g*	green value
in	*b*	blue value
in	*font*	The path to the font file
in	*ptSize*	The text size
in	*mode*	The writing mode : 0 (Solid), 1 (Blended)

2.25 text.h File Reference

text.c header

```
#include <stdlib.h>
#include <stdio.h>
#include <errno.h>
#include "structures.h"
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include <SDL/SDL_ttf.h>
#include "input.h"
```

Functions

- void printText (SDL_Surface ∗screen, SDL_Rect ∗posText, char ∗text, int r, int g, int b, char ∗font, int ptSize, int mode)
- void captureText (SDL_Surface ∗screen, SDL_Rect posText, char ∗text, int text_length, int r, int g, int b, char ∗font, int text_size, int ∗go)

2.25.1 Detailed Description

text.c header

Author

Xavier COPONET, Glenn HERROU

Date

2014-04-27

2.25.2 Function Documentation

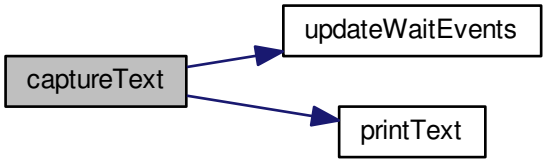
2.25.2.1 void captureText (SDL_Surface ∗ *screen*, SDL_Rect *posText*, char ∗ *text*, int *text_length*, int *r*, int *g*, int *b*, char ∗ *font*, int *text_size*, int ∗ *go*)

Capture the text corresponding to the keyboard inputs and display it on the screen at the given position

Parameters

out	*screen*	The screen of the game
in	*posText*	The position of the text. If NULL, the text is centered
out	*text*	The text to display
in	*r*	red value
in	*g*	green value
in	*b*	blue value
in	*text_length*	the text length
in	*font*	The path to the font file
in	*text_size*	The text size
out	*go*	The main loop validation

Here is the call graph for this function:



2.25.2.2 void printText (SDL_Surface * *screen*, SDL_Rect * *posText*, char * *text*, int *r*, int *g*, int *b*, char * *font*, int *ptSize*, int *mode*)

Display the given text on the screen, at the given position

Parameters

out	*screen*	The screen of the game
in	*posText*	The position of the text. If NULL, the text is centered
in	*text*	The text to display
in	*r*	red value
in	*g*	green value
in	*b*	blue value
in	*font*	The path to the font file
in	*ptSize*	The text size
in	*mode*	The writing mode : 0 (Solid), 1 (Blended)

Index