

Jumpman

Generated by Doxygen 1.8.3.1

Tue Dec 10 2013 11:11:34

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	BasicStar Class Reference	7
4.1.1	Detailed Description	7
4.1.2	Constructor & Destructor Documentation	8
4.1.2.1	BasicStar	8
4.1.2.2	~BasicStar	8
4.1.3	Member Function Documentation	8
4.1.3.1	randomizeSpawn	8
4.1.3.2	takeAction	8
4.2	Game Class Reference	8
4.2.1	Detailed Description	9
4.2.2	Constructor & Destructor Documentation	9
4.2.2.1	Game	9
4.2.2.2	~Game	9
4.2.2.3	Game	9
4.2.3	Member Function Documentation	9
4.2.3.1	addStars	9
4.2.3.2	drawObjectsToScreen	10
4.2.3.3	gameOver	10
4.2.3.4	handlePlayerInput	10
4.2.3.5	letObjectsInteract	10
4.2.3.6	operator=	10
4.2.3.7	run	10

4.2.4	Member Data Documentation	10
4.2.4.1	graphics_	10
4.2.4.2	player_	11
4.2.4.3	star_list_	11
4.3	GraphicsEngine Class Reference	11
4.3.1	Detailed Description	12
4.3.2	Constructor & Destructor Documentation	12
4.3.2.1	GraphicsEngine	12
4.3.2.2	~GraphicsEngine	12
4.3.2.3	GraphicsEngine	12
4.3.3	Member Function Documentation	13
4.3.3.1	drawImage	13
4.3.3.2	drawText	13
4.3.3.3	getEvent	13
4.3.3.4	getLastError	13
4.3.3.5	loadImage	13
4.3.3.6	makeScreenBlack	14
4.3.3.7	operator=	14
4.3.3.8	screen_height	14
4.3.3.9	screen_width	14
4.3.3.10	updateScreen	14
4.3.3.11	waitForKeypress	14
4.3.4	Member Data Documentation	14
4.3.4.1	font_	14
4.3.4.2	FRAME_RATE	14
4.3.4.3	images_	14
4.3.4.4	screen_	14
4.3.4.5	SCREEN_BPP	15
4.3.4.6	SCREEN_HEIGHT	15
4.3.4.7	SCREEN_WIDTH	15
4.3.4.8	time_of_last_refresh_	15
4.3.4.9	TITLE	15
4.4	MovingStar Class Reference	15
4.4.1	Detailed Description	16
4.4.2	Constructor & Destructor Documentation	16
4.4.2.1	MovingStar	16
4.4.2.2	~MovingStar	16
4.4.3	Member Function Documentation	16
4.4.3.1	takeAction	16
4.4.4	Member Data Documentation	16

4.4.4.1	dx_	16
4.4.4.2	dy_	16
4.5	Player Class Reference	16
4.5.1	Detailed Description	17
4.5.2	Constructor & Destructor Documentation	17
4.5.2.1	Player	17
4.5.2.2	~Player	17
4.5.3	Member Function Documentation	18
4.5.3.1	handleGravity	18
4.5.3.2	imageX	18
4.5.3.3	imageY	18
4.5.3.4	jump	18
4.5.3.5	move	18
4.5.3.6	reset	18
4.5.3.7	score	18
4.5.3.8	touches	19
4.5.4	Member Data Documentation	19
4.5.4.1	dx_	19
4.5.4.2	dy_	19
4.5.4.3	facing_direction_	19
4.5.4.4	score_	19
4.5.4.5	standing_on_floor_	19
4.6	Sprite Class Reference	19
4.6.1	Detailed Description	20
4.6.2	Constructor & Destructor Documentation	20
4.6.2.1	Sprite	20
4.6.2.2	Sprite	21
4.6.2.3	~Sprite	21
4.6.3	Member Function Documentation	21
4.6.3.1	filename	21
4.6.3.2	height	21
4.6.3.3	imageX	21
4.6.3.4	initialY	21
4.6.3.5	modifyY	21
4.6.3.6	operator=	21
4.6.3.7	width	22
4.6.3.8	x	22
4.6.3.9	y	22
4.6.4	Member Data Documentation	22
4.6.4.1	current_image_	22

4.6.4.2	filename_	22
4.6.4.3	height_	22
4.6.4.4	initial_y_	22
4.6.4.5	num_images_	22
4.6.4.6	width_	22
4.6.4.7	x_	22
4.6.4.8	y_	23
5	File Documentation	25
5.1	src/BasicStar.cc File Reference	25
5.2	src/BasicStar.hh File Reference	25
5.2.1	Detailed Description	25
5.3	src/Game.cc File Reference	26
5.4	src/Game.hh File Reference	26
5.4.1	Detailed Description	26
5.5	src/GraphicsEngine.cc File Reference	26
5.6	src/GraphicsEngine.hh File Reference	27
5.6.1	Detailed Description	27
5.6.2	Typedef Documentation	27
5.6.2.1	rect_t	27
5.6.3	Enumeration Type Documentation	27
5.6.3.1	event_t	27
5.7	src/main.cc File Reference	28
5.7.1	Function Documentation	28
5.7.1.1	main	28
5.8	src/MovingStar.cc File Reference	28
5.9	src/MovingStar.hh File Reference	28
5.9.1	Detailed Description	28
5.10	src/Player.cc File Reference	29
5.11	src/Player.hh File Reference	29
5.11.1	Detailed Description	29
5.12	src/Sprite.cc File Reference	29
5.13	src/Sprite.hh File Reference	29
5.13.1	Detailed Description	29
	Index	30

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Game	8
GraphicsEngine	11
Sprite	19
BasicStar	7
MovingStar	15
Player	16

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

BasicStar	The most basic type of star, cannot move	7
Game	Main game class	8
GraphicsEngine	Class for managing graphics and events	11
MovingStar	A BasicStar which moves around	15
Player	Player class	16
Sprite	Base class for all images	19

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

src/ BasicStar.cc	25
src/ BasicStar.hh	
File containing the BasicStar class Header	25
src/ Game.cc	26
src/ Game.hh	
File containing the Game class Header	26
src/ GraphicsEngine.cc	26
src/ GraphicsEngine.hh	
File containing the GraphicsEngine class Header	27
src/ main.cc	28
src/ MovingStar.cc	28
src/ MovingStar.hh	
File containing the MovingStar class Header	28
src/ Player.cc	29
src/ Player.hh	
File containing the Player class Header	29
src/ Sprite.cc	29
src/ Sprite.hh	
File containing the Sprite class Header	29

Chapter 4

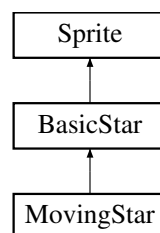
Class Documentation

4.1 BasicStar Class Reference

The most basic type of star, cannot move.

```
#include <BasicStar.hh>
```

Inheritance diagram for BasicStar:



Public Member Functions

- `BasicStar` (short `y`, int `edge_coord`)
Constructor.
- `~BasicStar` ()
Destructor.
- virtual void `takeAction` ()
The common star method for doing things.

Private Member Functions

- void `randomizeSpawn` (int `edge_coord`)
Set the enemy's x to a random number.

Additional Inherited Members

4.1.1 Detailed Description

The most basic type of star, cannot move.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 `BasicStar::BasicStar (short y, int edge_coord)`

Constructor.

Parameters

<i>y</i>	position of last BasicStar
<i>edge_coord</i>	how many pixels we need to move to escape the screen

4.1.2.2 `BasicStar::~~BasicStar ()`

Destructor.

4.1.3 Member Function Documentation

4.1.3.1 `void BasicStar::randomizeSpawn (int edge_coord)` `[private]`

Set the enemy's x to a random number.

Parameters

<i>edge_coord</i>	how many pixels we need to move to escape the screen
-------------------	--

4.1.3.2 `void BasicStar::takeAction ()` `[virtual]`

The common star method for doing things.

Reimplemented in [MovingStar](#).

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

- [src/BasicStar.hh](#)
- [src/BasicStar.cc](#)

4.2 Game Class Reference

Main game class.

```
#include <Game.hh>
```

Public Member Functions

- [Game](#) ()
Constructor.
- [~Game](#) ()
Destructor.
- int [run](#) ()
The game's main loop.

Private Member Functions

- int `handlePlayerInput` ()
handles player input
- int `letObjectsInteract` ()
lets objects interact with each other
- int `drawObjectsToScreen` ()
draw updates to screen
- void `addStars` ()
Add stars to star_list_ until they fill up the screen.
- int `gameOver` ()
Triggers when the player fails.
- `Game` (const `Game` &)
Disabled copy constructor.
- void `operator=` (const `Game` &)
Disabled copy constructor.

Private Attributes

- `GraphicsEngine` * `graphics_`
Instance for managing graphics.
- std::list< `BasicStar` * > `star_list_`
List of all flying objects that the player can hit.
- `Player` `player_`
Player instance.

4.2.1 Detailed Description

Main game class.

`Game` is the main class, which acts as the core of the application. Its role is glue the other projects together in a structured way without doing any work by itself.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 `Game::Game ()`

Constructor.

4.2.2.2 `Game::~~Game ()`

Destructor.

4.2.2.3 `Game::Game (const Game &) [private]`

Disabled copy constructor.

4.2.3 Member Function Documentation

4.2.3.1 `void Game::addStars () [private]`

Add stars to star_list_ until they fill up the screen.

4.2.3.2 `int Game::drawObjectsToScreen () [private]`

draw updates to screen

Returns

1 on graphics failure

4.2.3.3 `int Game::gameOver () [private]`

Triggers when the player fails.

Returns

always returns 2

4.2.3.4 `int Game::handlePlayerInput () [private]`

handles player input

Returns

1 if user want to quit the game

4.2.3.5 `int Game::letObjectsInteract () [private]`

lets objects interact with each other

All action happens here

Returns

1 if player has died

4.2.3.6 `void Game::operator= (const Game &) [private]`

Disabled copy constructor.

4.2.3.7 `int Game::run ()`

The game's main loop.

Returns

0 on success, 1 on failure and 2 on restart

4.2.4 Member Data Documentation

4.2.4.1 `GraphicsEngine* Game::graphics_ [private]`

Instance for managing graphics.

4.2.4.2 `Player` `Game::player_` [private]

`Player` instance.

4.2.4.3 `std::list<BasicStar*>` `Game::star_list_` [private]

List of all flying objects that the player can hit.

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

- `src/Game.hh`
- `src/Game.cc`

4.3 GraphicsEngine Class Reference

Class for managing graphics and events.

```
#include <GraphicsEngine.hh>
```

Public Member Functions

- `GraphicsEngine` (const std::string &title, const unsigned `screen_width`, const unsigned `screen_height`, const unsigned `screen_bpp`, const unsigned `frame_rate`)
Constructor.
- `~GraphicsEngine` ()
Destructor.
- bool `loadImage` (const std::string &filename)
Loads and image from the disk into the RAM.
- std::string `getLastError` () const
returns a string describing last error that occurred
- void `makeScreenBlack` ()
fills screen width black paint
- bool `drawImage` (const std::string &image, `rect_t` *srcrect, `rect_t` *dstrect)
Draw an image to the game screen.
- void `drawText` (const std::string &text, unsigned x, unsigned y)
Draw some text at the given location.
- bool `updateScreen` ()
Flushes the screen so it's visible to the user.
- bool `getEvent` (`event_t` &event) const
Non-blocking function to check event depending on user input.
- void `waitForKeypress` () const
Waits until a key is pressed.
- unsigned `screen_width` () const
Returns width of game screen.
- unsigned `screen_height` () const
Returns height of game screen.

Private Member Functions

- `GraphicsEngine` (const `GraphicsEngine` &)
Copy constructor (DO NOT USE)
- void `operator=` (const `GraphicsEngine` &)
Copy constructor (DO NOT USE)

Private Attributes

- const std::string [TITLE](#)
Title to display on the game's status bar.
- const unsigned [SCREEN_WIDTH](#)
Width of the game screen.
- const unsigned [SCREEN_HEIGHT](#)
Height of the game screen.
- const unsigned [SCREEN_BPP](#)
Screen bits per pixel (color)
- const unsigned short [FRAME_RATE](#)
Screen frames per second.
- std::map< std::string, SDL_Surface * > [images_](#)
Map of filename and image we have loaded from disk.
- size_t [time_of_last_refresh_](#)
The time [updateScreen\(\)](#) last was called.
- SDL_Surface * [screen_](#)
The game screen.
- TTF_Font * [font_](#)
Font to use.

4.3.1 Detailed Description

Class for managing graphics and events.

[GraphicsEngine](#) handles all input from the user and handles all drawing of images on the screen. Only one [GraphicsEngine](#) can be active at one time

4.3.2 Constructor & Destructor Documentation

4.3.2.1 [GraphicsEngine::GraphicsEngine](#) (const std::string & *title*, const unsigned *screen_width*, const unsigned *screen_height*, const unsigned *screen_bpp*, const unsigned *frame_rate*)

Contructor.

Parameters

<i>title</i>	The title that will be seen on the titlebar
<i>screen_width</i>	Size of game screen's width
<i>screen_height</i>	Size of the game screen's height
<i>screen_bpp</i>	The amount of bits per pixel (color)
<i>frame_rate</i>	The frames per second we want to display

4.3.2.2 [GraphicsEngine::~~GraphicsEngine](#) ()

Destructor.

4.3.2.3 [GraphicsEngine::GraphicsEngine](#) (const [GraphicsEngine](#) &) `[private]`

Copy constructor (DO NOT USE)

4.3.3 Member Function Documentation

4.3.3.1 `bool GraphicsEngine::drawImage (const std::string & image, rect_t * srcrect, rect_t * dstrect)`

Draw an image to the game screen.

Parameters

<i>image</i>	filename of image to draw
<i>srcrect</i>	rectangle of image to draw from
<i>dstrect</i>	part to screen to draw to

Returns

true on success

4.3.3.2 `void GraphicsEngine::drawText (const std::string & text, unsigned x, unsigned y)`

Draw some text at the given location.

Parameters

<i>text</i>	text to draw
<i>x</i>	the center on the x-axis where we will draw
<i>y</i>	the center on the y-axis where we will draw

4.3.3.3 `bool GraphicsEngine::getEvent (event_t & event) const`

Non-blocking function to check event depending on user input.

Parameters

<i>event</i>	The event received from user
--------------	------------------------------

Returns

true if there are more pending events

4.3.3.4 `string GraphicsEngine::getLastError () const`

returns a string describing last error that occurred

4.3.3.5 `bool GraphicsEngine::loadImage (const std::string & filename)`

Loads and image from the disk into the RAM.

Parameters

<i>filename</i>	Name of the image file inside graphics/ folder (example: to load graphics/player.png, filename should be "player")
-----------------	--

Returns

true on success

4.3.3.6 void GraphicsEngine::makeScreenBlack ()

fills screen with black paint

4.3.3.7 void GraphicsEngine::operator= (const GraphicsEngine &) [private]

Copy constructor (DO NOT USE)

4.3.3.8 unsigned GraphicsEngine::screen_height () const

Returns height of game screen.

4.3.3.9 unsigned GraphicsEngine::screen_width () const

Returns width of game screen.

4.3.3.10 bool GraphicsEngine::updateScreen ()

Flushes the screen so it's visible to the user.

Returns

true on success

4.3.3.11 void GraphicsEngine::waitForKeypress () const

Waits until a key is pressed.

4.3.4 Member Data Documentation**4.3.4.1 TTF_Font* GraphicsEngine::font_ [private]**

Font to use.

4.3.4.2 const unsigned short GraphicsEngine::FRAME_RATE [private]

Screen frames per second.

4.3.4.3 std::map<std::string, SDL_Surface*> GraphicsEngine::images_ [private]

Map of filename and image we have loaded from disk.

4.3.4.4 SDL_Surface* GraphicsEngine::screen_ [private]

The game screen.

4.3.4.5 `const unsigned GraphicsEngine::SCREEN.BPP` [private]

Screen bits per pixel (color)

4.3.4.6 `const unsigned GraphicsEngine::SCREEN.HEIGHT` [private]

Height of the game screen.

4.3.4.7 `const unsigned GraphicsEngine::SCREEN.WIDTH` [private]

Width of the game screen.

4.3.4.8 `size_t GraphicsEngine::time_of_last_refresh` [private]

The time [updateScreen\(\)](#) last was called.

4.3.4.9 `const std::string GraphicsEngine::TITLE` [private]

Title to display on the game's status bar.

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

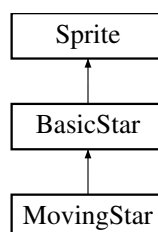
- [src/GraphicsEngine.hh](#)
- [src/GraphicsEngine.cc](#)

4.4 MovingStar Class Reference

A [BasicStar](#) which moves around.

```
#include <MovingStar.hh>
```

Inheritance diagram for MovingStar:



Public Member Functions

- [MovingStar](#) (short *y*, int *edge_coord*)
Constructor.
- [~MovingStar](#) ()
Destructor.
- void [takeAction](#) ()
Overloaded from [BasicStar](#).

Private Attributes

- short [dx_](#)
- short [dy_](#)

Additional Inherited Members

4.4.1 Detailed Description

A [BasicStar](#) which moves around.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 `MovingStar::MovingStar (short y, int edge_coord)`

Constructor.

Parameters

<i>y</i>	position of last BasicStar
<i>edge_coord</i>	how many pixels we need to move to escape the screen

4.4.2.2 `MovingStar::~~MovingStar ()`

Destructor.

4.4.3 Member Function Documentation

4.4.3.1 `void MovingStar::takeAction ()` [virtual]

Overloaded from [BasicStar](#).

Reimplemented from [BasicStar](#).

4.4.4 Member Data Documentation

4.4.4.1 `short MovingStar::dx_` [private]

4.4.4.2 `short MovingStar::dy_` [private]

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

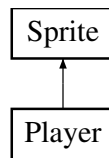
- [src/MovingStar.hh](#)
- [src/MovingStar.cc](#)

4.5 Player Class Reference

[Player](#) class.

```
#include <Player.hh>
```

Inheritance diagram for Player:



Public Member Functions

- [Player](#) ()
Constructor.
- [~Player](#) ()
Destructor.
- void [reset](#) ()
resets player to starting position
- bool [touches](#) ([Sprite](#) *other)
Check if player touches another [Sprite](#) class.
- void [handleGravity](#) (const signed screen_width)
Manages player's movement depending on dx and dy.
- void [jump](#) (bool force_push=false)
Jump a short distance into the air.
- void [move](#) (short dx)
Set player to in movement on the x-axis.
- size_t [score](#) () const
- short [imageX](#) ()
- short [imageY](#) () const

Private Attributes

- short [dx_](#)
- short [dy_](#)
- bool [standing_on_floor_](#)
- size_t [score_](#)
- bool [facing_direction_](#)

Additional Inherited Members

4.5.1 Detailed Description

[Player](#) class.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 [Player::Player](#) ()

Constructor.

4.5.2.2 [Player::~~Player](#) ()

Destructor.

4.5.3 Member Function Documentation

4.5.3.1 void Player::handleGravity (const signed *screen_width*)

Manages player's movement depending on dx and dy.

Parameters

<i>screen_width</i>	width of game screen
---------------------	----------------------

4.5.3.2 short Player::imageX () [virtual]

[Player](#) image is a bit special so it has its own [imageX\(\)](#)

Returns

x of the image the [Sprite](#) wants to draw

Reimplemented from [Sprite](#).

4.5.3.3 short Player::imageY () const

Returns

y of the image the [Sprite](#) wants to draw

4.5.3.4 void Player::jump (bool *force_push* = false)

Jump a short distance into the air.

Parameters

<i>force_push</i>	if true, this is not due to the player jumping
-------------------	--

4.5.3.5 void Player::move (short *dx*)

Set player to in movement on the x-axis.

Parameters

<i>dx</i>	-1 to move left, +1 to move right, 0 to stay still
-----------	--

4.5.3.6 void Player::reset ()

resets player to starting position

4.5.3.7 size_t Player::score () const

Returns

the current player score

4.5.3.8 `bool Player::touches (Sprite * other)`

Check if player touches another [Sprite](#) class.

Parameters

<i>other</i>	Sprite to check if they touch
--------------	---

Returns

true if they touch

4.5.4 Member Data Documentation

4.5.4.1 `short Player::dx_ [private]`

Current x-axis movement

4.5.4.2 `short Player::dy_ [private]`

Current y-axis movement

4.5.4.3 `bool Player::facing_direction_ [private]`

Direction the player is facing, false = right

4.5.4.4 `size_t Player::score_ [private]`

Current player score

4.5.4.5 `bool Player::standing_on_floor_ [private]`

True if player has not yet jumped

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

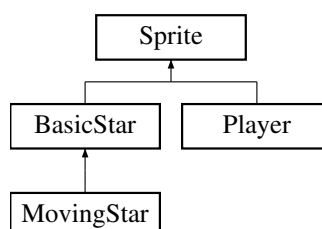
- [src/Player.hh](#)
- [src/Player.cc](#)

4.6 Sprite Class Reference

Base class for all images.

```
#include <Sprite.hh>
```

Inheritance diagram for Sprite:



Public Member Functions

- **Sprite** (std::string **filename**, short **x**, short **y**, unsigned short **width**, unsigned short **height**, short num_images)
Constructor.
- **Sprite** (const **Sprite** &other)
Copy Constructor.
- virtual **~Sprite** ()
Destructor.
- **Sprite** & **operator=** (const **Sprite** &other)
Copy Constructor.
- const std::string & **filename** () const
- short **x** () const
- short **y** () const
- unsigned short **width** () const
- unsigned short **height** () const
- virtual short **imageX** ()
- short **initialY** () const
the position of the y-axis this sprite was initiated at
- void **modifyY** (int mod)
*Modifies the **Sprite**'s position of the y-axis.*

Protected Attributes

- short **current_image_**
- short **num_images_**
- short **x_**
- short **y_**
- short **initial_y_**
- const unsigned short **width_**
- const unsigned short **height_**
- std::string **filename_**

4.6.1 Detailed Description

Base class for all images.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 **Sprite::Sprite** (std::string *filename*, short *x*, short *y*, unsigned short *width*, unsigned short *height*, short *num_images*)

Constructor.

Parameters

<i>filename</i>	Name of the object's image-file inside graphics/
<i>x</i>	starting x-position
<i>y</i>	starting y-position
<i>width</i>	width of image
<i>height</i>	height of image
<i>num_images</i>	how many images this Sprite has

4.6.2.2 `Sprite::Sprite (const Sprite & other)`

Copy Constructor.

Parameters

<i>other</i>	Sprite to copy
--------------	--------------------------------

4.6.2.3 `Sprite::~~Sprite () [virtual]`

Destructor.

4.6.3 Member Function Documentation

4.6.3.1 `const std::string & Sprite::filename () const`

Returns

filename of image file

4.6.3.2 `unsigned short Sprite::height () const`

Returns

[Sprite](#)'s image's height

4.6.3.3 `short Sprite::imageX () [virtual]`

Returns

x of the image the [Sprite](#) wants to draw

Reimplemented in [Player](#).

4.6.3.4 `short Sprite::initialY () const`

the position of the y-axis this sprite was initiated at

4.6.3.5 `void Sprite::modifyY (int mod)`

Modifies the [Sprite](#)'s position of the y-axis.

Parameters

<i>mod</i>	Y axis modifier
------------	-----------------

4.6.3.6 `Sprite & Sprite::operator= (const Sprite & other)`

Copy Constructor.

Parameters

<i>other</i>	Sprite to copy
--------------	--------------------------------

4.6.3.7 unsigned short `Sprite::width` () const

Returns

[Sprite](#)'s image's width

4.6.3.8 short `Sprite::x` () const

Returns

[Sprite](#)'s position of the x-axis

4.6.3.9 short `Sprite::y` () const

Returns

[Sprite](#)'s position of the y-axis

4.6.4 Member Data Documentation

4.6.4.1 short `Sprite::current_image_` [protected]

If the sprite has many images, this handles them

4.6.4.2 std::string `Sprite::filename_` [protected]

[Sprite](#)'s image's filename

4.6.4.3 const unsigned short `Sprite::height_` [protected]

[Sprite](#)'s image's height

4.6.4.4 short `Sprite::initial_y_` [protected]4.6.4.5 short `Sprite::num_images_` [protected]

How many images a [Sprite](#) has

4.6.4.6 const unsigned short `Sprite::width_` [protected]

[Sprite](#)'s original position on the y-axis [Sprite](#)'s image's width

4.6.4.7 short `Sprite::x_` [protected]

[Sprite](#)'s position on the x-axis

4.6.4.8 short Sprite::y_ [protected]

[Sprite](#)'s position on the y-acis

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

- src/[Sprite.hh](#)
- src/[Sprite.cc](#)

Chapter 5

File Documentation

5.1 src/BasicStar.cc File Reference

```
#include <random>
#include <chrono>
#include "BasicStar.hh"
```

5.2 src/BasicStar.hh File Reference

File containing the [BasicStar](#) class Header.

```
#include "Sprite.hh"
```

Classes

- class [BasicStar](#)

The most basic type of star, cannot move.

5.2.1 Detailed Description

File containing the [BasicStar](#) class Header.

Author

Olle Kvarnström

Date

5.3 src/Game.cc File Reference

```
#include <iostream>
#include <string>
#include <algorithm>
#include <chrono>
#include <random>
#include "Game.hh"
#include "MovingStar.hh"
```

5.4 src/Game.hh File Reference

File containing the [Game](#) class Header.

```
#include <list>
#include "GraphicsEngine.hh"
#include "Player.hh"
#include "BasicStar.hh"
```

Classes

- class [Game](#)
Main game class.

5.4.1 Detailed Description

File containing the [Game](#) class Header.

Author

Olle Kvarnström

Date

5.5 src/GraphicsEngine.cc File Reference

```
#include "GraphicsEngine.hh"
#include <iostream>
#include <algorithm>
```


5.6 src/GraphicsEngine.hh File Reference

File containing the [GraphicsEngine](#) class Header.

```
#include <string>
#include <map>
#include <SDL/SDL.h>
#include <SDL/SDL_image.h>
#include <SDL/SDL_ttf.h>
```

Classes

- class [GraphicsEngine](#)
Class for managing graphics and events.

Typedefs

- typedef SDL_Rect [rect_t](#)

Enumerations

- enum [event_t](#) {
 [LEFT](#), [RIGHT](#), [UP](#), [STILL](#),
 [NOTHING](#), [QUIT](#) }
events describing user input

5.6.1 Detailed Description

File containing the [GraphicsEngine](#) class Header.

Author

Olle Kvarnström

Date

5.6.2 Typedef Documentation

5.6.2.1 typedef SDL_Rect [rect_t](#)

5.6.3 Enumeration Type Documentation

5.6.3.1 enum [event_t](#)

events describing user input

Enumerator

LEFT [Player](#) wants to move left

RIGHT [Player](#) wants to move right

UP [Player](#) wants to jump
STILL [Player](#) wants to stop moving
NOTHING Unknown input received
QUIT User wants to exit the game

5.7 src/main.cc File Reference

```
#include "Game.hh"
```

Functions

- int [main](#) ()

5.7.1 Function Documentation

5.7.1.1 int main ()

5.8 src/MovingStar.cc File Reference

```
#include <chrono>
#include <random>
#include "MovingStar.hh"
```

5.9 src/MovingStar.hh File Reference

File containing the [MovingStar](#) class Header.

```
#include "BasicStar.hh"
```

Classes

- class [MovingStar](#)
A [BasicStar](#) which moves around.

5.9.1 Detailed Description

File containing the [MovingStar](#) class Header.

Author

Olle Kvarnström

Date

5.10 src/Player.cc File Reference

```
#include "Player.hh"
```

5.11 src/Player.hh File Reference

File containing the [Player](#) class Header.

```
#include "Sprite.hh"
```

Classes

- class [Player](#)
Player class.

5.11.1 Detailed Description

File containing the [Player](#) class Header.

Author

Olle Kvarnström

Date

5.12 src/Sprite.cc File Reference

```
#include "Sprite.hh"
```

5.13 src/Sprite.hh File Reference

File containing the [Sprite](#) class Header.

```
#include <string>
```

Classes

- class [Sprite](#)
Base class for all images.

5.13.1 Detailed Description

File containing the [Sprite](#) class Header.

Author

Olle Kvarnström

Date

Index

- ~BasicStar
 - BasicStar, 8
- ~Game
 - Game, 9
- ~GraphicsEngine
 - GraphicsEngine, 12
- ~MovingStar
 - MovingStar, 16
- ~Player
 - Player, 17
- ~Sprite
 - Sprite, 21
- addStars
 - Game, 9
- BasicStar, 7
 - ~BasicStar, 8
 - BasicStar, 8
 - BasicStar, 8
 - randomizeSpawn, 8
 - takeAction, 8
- current_image_
 - Sprite, 22
- drawImage
 - GraphicsEngine, 13
- drawObjectsToScreen
 - Game, 9
- drawText
 - GraphicsEngine, 13
- dx_
 - MovingStar, 16
 - Player, 19
- dy_
 - MovingStar, 16
 - Player, 19
- event_t
 - GraphicsEngine.hh, 27
- FRAME_RATE
 - GraphicsEngine, 14
- facing_direction_
 - Player, 19
- filename
 - Sprite, 21
- filename_
 - Sprite, 22
- font_

- GraphicsEngine, 14
- Game, 8
 - ~Game, 9
 - addStars, 9
 - drawObjectsToScreen, 9
 - Game, 9
 - gameOver, 10
 - graphics_, 10
 - handlePlayerInput, 10
 - letObjectsInteract, 10
 - operator=, 10
 - player_, 10
 - run, 10
 - star_list_, 11
- gameOver
 - Game, 10
- getEvent
 - GraphicsEngine, 13
- getLastError
 - GraphicsEngine, 13
- GraphicsEngine.hh
 - LEFT, 27
 - NOTHING, 28
 - QUIT, 28
 - RIGHT, 27
 - STILL, 28
 - UP, 27
- graphics_
 - Game, 10
- GraphicsEngine, 11
 - ~GraphicsEngine, 12
 - drawImage, 13
 - drawText, 13
 - FRAME_RATE, 14
 - font_, 14
 - getEvent, 13
 - getLastError, 13
 - GraphicsEngine, 12
 - GraphicsEngine, 12
 - images_, 14
 - loadImage, 13
 - makeScreenBlack, 14
 - operator=, 14
 - SCREEN_BPP, 14
 - SCREEN_HEIGHT, 15
 - SCREEN_WIDTH, 15
 - screen_, 14
 - screen_height, 14
 - screen_width, 14

- TITLE, 15
 - time_of_last_refresh_, 15
 - updateScreen, 14
 - waitForKeypress, 14
- GraphicsEngine.hh
 - event_t, 27
 - rect_t, 27
- handleGravity
 - Player, 18
- handlePlayerInput
 - Game, 10
- height
 - Sprite, 21
- height_
 - Sprite, 22
- imageX
 - Player, 18
 - Sprite, 21
- imageY
 - Player, 18
- images_
 - GraphicsEngine, 14
- initial_y_
 - Sprite, 22
- initialY
 - Sprite, 21
- jump
 - Player, 18
- LEFT
 - GraphicsEngine.hh, 27
- letObjectsInteract
 - Game, 10
- loadImage
 - GraphicsEngine, 13
- main
 - main.cc, 28
- main.cc
 - main, 28
- makeScreenBlack
 - GraphicsEngine, 14
- modifyY
 - Sprite, 21
- move
 - Player, 18
- MovingStar, 15
 - ~MovingStar, 16
 - dx_, 16
 - dy_, 16
 - MovingStar, 16
 - MovingStar, 16
 - takeAction, 16
- NOTHING
 - GraphicsEngine.hh, 28
- num_images_
 - Sprite, 22
- operator=
 - Game, 10
 - GraphicsEngine, 14
 - Sprite, 21
- Player, 16
 - ~Player, 17
 - dx_, 19
 - dy_, 19
 - facing_direction_, 19
 - handleGravity, 18
 - imageX, 18
 - imageY, 18
 - jump, 18
 - move, 18
 - Player, 17
 - reset, 18
 - score, 18
 - score_, 19
 - standing_on_floor_, 19
 - touches, 18
- player_
 - Game, 10
- QUIT
 - GraphicsEngine.hh, 28
- RIGHT
 - GraphicsEngine.hh, 27
- randomizeSpawn
 - BasicStar, 8
- rect_t
 - GraphicsEngine.hh, 27
- reset
 - Player, 18
- run
 - Game, 10
- STILL
 - GraphicsEngine.hh, 28
- SCREEN_BPP
 - GraphicsEngine, 14
- SCREEN_HEIGHT
 - GraphicsEngine, 15
- SCREEN_WIDTH
 - GraphicsEngine, 15
- score
 - Player, 18
- score_
 - Player, 19
- screen_
 - GraphicsEngine, 14
- screen_height
 - GraphicsEngine, 14
- screen_width
 - GraphicsEngine, 14
- Sprite, 19

- ~Sprite, [21](#)
- current_image_, [22](#)
- filename, [21](#)
- filename_, [22](#)
- height, [21](#)
- height_, [22](#)
- imageX, [21](#)
- initial_y_, [22](#)
- initialY, [21](#)
- modifyY, [21](#)
- num_images_, [22](#)
- operator=, [21](#)
- Sprite, [20](#)
- width, [22](#)
- width_, [22](#)
- x, [22](#)
- x_, [22](#)
- y, [22](#)
- y_, [22](#)
- src/BasicStar.cc, [25](#)
- src/BasicStar.hh, [25](#)
- src/Game.cc, [26](#)
- src/Game.hh, [26](#)
- src/GraphicsEngine.cc, [26](#)
- src/GraphicsEngine.hh, [27](#)
- src/MovingStar.cc, [28](#)
- src/MovingStar.hh, [28](#)
- src/Player.cc, [29](#)
- src/Player.hh, [29](#)
- src/Sprite.cc, [29](#)
- src/Sprite.hh, [29](#)
- src/main.cc, [28](#)
- standing_on_floor_
 - Player, [19](#)
- star_list_
 - Game, [11](#)
- TITLE
 - GraphicsEngine, [15](#)
- takeAction
 - BasicStar, [8](#)
 - MovingStar, [16](#)
- time_of_last_refresh_
 - GraphicsEngine, [15](#)
- touches
 - Player, [18](#)
- UP
 - GraphicsEngine.hh, [27](#)
- updateScreen
 - GraphicsEngine, [14](#)
- waitForKeypress
 - GraphicsEngine, [14](#)
- width
 - Sprite, [22](#)
- width_
 - Sprite, [22](#)
- x
 - Sprite, [22](#)
- x_
 - Sprite, [22](#)
- y
 - Sprite, [22](#)
- y_
 - Sprite, [22](#)