

## My Project

Generated by Doxygen 1.9.1



# Chapter 1

## Arcade Shared Library

Shared library for Arcade project

### 1.1 Description

This library contains shared code for the Arcade project in order to unify interfaces between collaborative groups.

### 1.2 Groups

- **G-Epitech**: Flavien Chenu, Math & Yann Masson
- **Carapace Retro**: Baptiste Moreau, Axel Fradet & Suceveanu Dragos

### 1.3 How ?

Our arcade simulator relies on a modular development model, utilizing dynamic libraries to separate game logic from graphical display. This system revolves around two types of libraries: game libraries and graphical libraries. These two libraries operate entirely independently, focusing solely on the principle of getters and setters for data manipulation.

#### 1. Game Libraries:

Game libraries are responsible for managing the internal game logic, including gameplay mechanics, collisions, levels, and more. They provide functionalities through getter and setter methods, allowing developers to access and modify game data as needed.

#### 1. Graphical Libraries:

On the other hand, graphical libraries handle the visual display of the game. They include features such as sprite rendering, animation management, and special effects. These libraries also interact with the simulator solely through getter and setter methods, enabling the modification of visual properties of game elements.

#### 1. Core:

To ensure consistency and communication between game libraries and graphical libraries, a "core" is necessary. This core acts as a central liaison, coordinating game data with graphical display. It retrieves information about the game state from game libraries and passes it to graphical libraries for display. Similarly, it monitors user interactions and updates game data accordingly, ensuring a smooth gaming experience.

In summary, our arcade simulator relies on a modular development model where game libraries, graphical libraries, and the core interact independently, providing maximum flexibility and enabling developers to create unique and dynamic gaming experiences.

## 1.4 Documentation

If you want the code documentation you can run this command :  
`doxygen Doxyfile`

This command allow you to generate Doxygen documentation.

## Chapter 2

# Add Game Library Guide

Get the score of the game

.

## 2.1 :notebook\_with\_decorative\_cover: Introduction

### 2.1.0.1 :briefcase: Project Explication:

In the Arcade projects you have the possibility to add your own game.

You will be able to play your games in the arcade, to do so you will have to follow the next steps carefully.

Be attentive it is important!

## 2.2 The way we do it !

### 2.2.1 Basic:

To do this please create your folder at this place "src/games" and named this repository in lowercase letter by the name of your games for exemples: `**"src/games/snake"**`.

In order for your game to be added to the arcade you will have to in the `**"src/games/CMakeLists.txt"**` file. To do this you just have to go to the end of the file and just do like the other games.

Use the example below:

```
## Create the games here "create_game(<game_name>)"
create_game(test_game)
```

Just add the line: `create_game(...)`

Of course has the place of `**"..."**` you have to put the name of your game.

:construction: But beware, it is absolutely necessary that the name of the game you put is the same name as the directory created at the beginning. Otherwise it will not work.

---

### 2.2.2 Let's build it:

Pour cette explication nous partirons du principe que nous voulons rajouter le jeux **Snake** :snake:

### 2.2.2.1 Provider

To start we will create our SnakeProvider class. This class is the starting class that means it serves either:

- To create the class of the games that we will see later via this function:

```
/**
 * @brief Provides a new instance of the game
 *
 * @return Created game instance
 */
virtual std::shared_ptr<shared::games::IGame> createInstance(void) = 0;
```

Or

- To create the game manifest with all this information via this function:

```
/**
 * @brief Provides the game manifest
 *
 * @return Manifest of current game
 */
virtual const GameManifest &getManifest() const noexcept = 0;
```

A manifest looks like this:

```
typedef struct {
    std::string name;           // Name of the author
    std::string email;         // Public contact email
    std::string website;       // Website of the author ('github', 'gitlab', etc.)
} Author;
typedef struct {
    const std::string name;     // Name of the game
    const std::string description; // Description of the game
    const std::string version;  // Version of the game
    const std::vector<Author> authors; // Authors
    const std::string iconPath; // Path of the icon game
} GameManifest;
```

### 2.2.2.2 Games

Back to the main, effectively in the provider you have the possibilities to create the class of the games that we will see now.

So I suggest you create a class that we will call Snake following our example from the beginning, this class will be the main class because it will manage everything.

It will inherit the interface: [shared::games::IGame](#).

Of course, it will override all the following functions:

```
/**
 * @brief Compute the game each tick of the program
 *
 * @param dt Time since last tick (Time in 'milliseconds')
 */
virtual void compute(DeltaTime dt) = 0;
```

This one above serves to recover the time elapsed since the last tick of the game and so perform certain action like moving or other.

/\*\* \*\*

#### Returns

The score of the game \*/ virtual const int getScore() const noexcept = 0;

This will be used to return the score to the core which will manage the system of high score.

There are other functions in this interface that I see invite to see here: [\\*\\*"arcade/common/games/IGame.hpp"](#)

But the most important function remains this:

```
/**
 * @brief Get map of entities
 *
 * @return Entities map of the game
 */
virtual const entity::EntitiesMap &getEntities(void) const = 0;
```

This function will be used to send all entities to the core. You surely wonder what an entity is? Don't worry, we'll come back to this right after. But this function is very important because it will allow to display images, texts and much more.

### 2.2.2.3 Entities

Finally, as I explained earlier the snake game class to take our example will be able to send entities.

An entity contains components that can be of various nature:

```
typedef enum {
    TEXTURE,
    TEXT,
    DISPLAYABLE,
    SOUND,
    COLLIDABLE,
    POSITIONABLE,
    KEYBOARD
} ComponentType;
```

As you see a component can be a texture, text something that can be displayed or not, a sound, have a collision, ... So you understood it for every thing you want to display, play or other, you will have to create a class that will inherit the interfaces select the components that you want to see in more detail the interfaces of the different components I let us direct you here: `**"arcade/common/games/components/"**`

If we take the example of our snake, then, for example, there would be a `tete` entity that would look like this:

```
class SnakeHeadEntity : public entity::IEntity {
public:
    SnakeHeadEntity();
    ~SnakeHeadEntity();
    const components::ComponentsMap &getComponents(void) const noexcept override;
private:
    components::ComponentsMap _components;
};
```

Of course in `components::ComponentsMap` there would be several class of components like:

The texture component:

```
class SnakeHeadDisplayable : public components::ITextureComponent {
public:
    explicit SnakeHeadDisplayable(const entity::IEntity &entity);
    ~SnakeHeadDisplayable();
    // IComponent
    const components::ComponentType getType() const noexcept override;
    const entity::IEntity &getEntity() noexcept override;
    ...
};
```

or even the keyboard component:

```
class SnakeHeadKeyboard : public components::IKeyboardComponent {
public:
    explicit SnakeHeadKeyboard(const entity::IEntity &entity);
    ~SnakeHeadKeyboard();
    // IComponent
    const components::ComponentType getType() const noexcept override;
    const entity::IEntity &getEntity() noexcept override;
    // Keyboard
    void onKeyPress(std::shared_ptr<IGame> ctx,
        shared::games::components::IKeyboardComponent::KeyData key) override;
    void onKeyRelease(std::shared_ptr<IGame> ctx,
        shared::games::components::IKeyboardComponent::KeyData key) override;
    ...
};
```

### 2.2.3 Shared Library

If you wish to have more information about the implementation of the games I suggest you go to the folder `**"common/"**` where you will find all the interfaces that will inherit your class. These interfaces are domesticated and explained. If the common folder does not exist I redirect you to this url: <https://github.com/G-Epitech/MAYBDF-ArcadeShared.git>





## Chapter 3

# Add Graphic Library Guide

### 3.1 Introduction

#### 3.1.0.1 Project Explication:

In the Arcade projects you have the possibility to add your own graphic library.

In order to be able to change in the middle of the games on the libraries you just added is not that great? Be careful not to add 2 times the same libraries although it is possible it is useless.

Of course there are certain rules to respect that we will address.

### 3.2 The way we do it !

#### 3.2.1 Basic:

To do this please create your folder at this place "src/graphics" and named this repertoire in lowercase letter by the name of your graphic libraries.

Then for your libraries to be added to the construction of our projects, I let you go in the file `**"src/graphics/CMakeLists.txt"`. When you are in this file, at the end you should see:

```
## Add the graphical libraries here "create_graphic_library(<library_name>)"
set(SFML_DEPENDENCY sfml-graphics sfml-system sfml-window sfml-audio)
set(SDL2_DEPENDENCY SDL2 SDL2_image SDL2_ttf SDL2_mixer)
create_graphic_library(ncurses ncurses)
create_graphic_library(sfml "${SFML_DEPENDENCY}")
create_graphic_library(SDL2 "${SDL2_DEPENDENCY}")
```

I will let you add your line with the name of the library and its dependencies, if any: `set(YOUR_LIBRARY_DEPENDENCY your_dependency1 your_dependency2 ...)`

Then you would add the line with the name of the library: `create_graphic_library(your_library "${YOUR_LIBRARY_DEPENDENCY}")`

---

#### 3.2.2 Let's build it:

It should be noted that in the architecture of graphic libraries there are 5 classes:

1. The main class on behalf of your libraries
2. The window class
3. The texture class
4. The font class
5. The Sound Class

Remember that each .cpp file you create must be added to the **CMakeLists.txt**.

---

##### 3.2.2.1 Your Library Class (Main Class):

Create your files .hpp and .cpp, create your class on behalf of your libraries that must inherit the `**"shared::graphics::IGraphicsProvider"` class. This class will relay to your other classes, as it will be used to create the

**Window, Sound, Texture and Font.** So implement your functions by overriding the functions of the class "shared::graphics::IGraphicsProvider". These functions must of course return class them by creating them.

Here are the functions a override:

```
/**
 * @brief Get the manifest of the graphics library
 *
 * @return Manifest of the graphics library
 */
virtual const GraphicsManifest &getManifest(void) const noexcept = 0;
/**
 * @brief Create a new window object
 *
 * @param windowProps Properties to use to init the window
 * @return Created window object
 */
virtual std::unique_ptr<IWindow> createWindow(const IWindow::WindowInitProps &windowProps) = 0;
/**
 * @brief Create a sound object
 *
 * @param path Path of the sound file
 * @return Created sound object
 */
virtual std::shared_ptr<ISound> createSound(const std::string &path) = 0;
/**
 * @brief Create a texture object
 *
 * @param bin Path of the binary texture file
 * @param ascii Path of the ascii texture file
 * @return Created texture object
 */
virtual std::shared_ptr<ITexture> createTexture(const std::string &bin, const std::string &ascii) = 0;
/**
 * @brief Create a font object
 *
 * @param path Path of the font file
 * @return Created font object
 */
virtual std::shared_ptr<IFont> createFont(const std::string &path) = 0;
```

---

### 3.2.2.2 The Window Class:

Create the Window directory, in which you can create your . hpp and . cpp files,

This class is the second main class, it serves to create the window, render the images, the texts, and manage the events. As the main class and for all the next classes you will have to override all functions.

This class inherit from **\*\*"shared::graphics::IWindow"\*\***

Here is a brief overview of the a override functions:

```
/**
 * @brief Render the texture of entity with given properties
 *
 * @param props Properties of the entity & texture to render
 */
virtual void render(const TextureProps &props) = 0;
/**
 * @brief Render the text of entity with given properties
 *
 * @param props Properties of the entity & text to render
 */
virtual void render(const TextProps &props) = 0;
/**
 * @brief Clear the content of the window
 */
virtual void clear(void) = 0;
/**
 * @brief Display the content of the window
 */
virtual void display(void) = 0;
/**
 * @brief Close the window
 */
virtual void close(void) = 0;
```

---

### 3.2.2.3 The texture Class:

This class is used to create the texture of the image you want to appear on your window and therefore this class will be used in the Window class and more especially by the render function which serves to give the images to the window for it to be displayed.

This class inherit from **\*\*"shared::graphics::ITexture"\*\***

---

#### 3.2.2.4 Font class

This class is used to create the font of the text you want to appear on your window and therefore this class will be used in the Window class and more especially by the render function which serves to give the images to the window for it to be displayed.

This class inherit from `**"shared::graphics::IFont"**`

---

#### 3.2.2.5 Sound Class

This class is used to manage the sound on the libraries you want to add, I let you create the files .cpp and .hpp, where you will override the functions of the ISound class by implementing the sound system with the library functions that you want to add to the arcade project.

This class inherit from `**"shared::graphics::ISound"**`

---

### 3.2.3 Shared Library

If you wish to have more information about the implementation of the libraries I suggest you go to the folder `**"common/"**` where you will find all the interfaces that will inherit your class. These interfaces are domesticated and explained. If the common folder does not exist I redirect you to this url: <https://github.com/G-epitech/MAYBDF-ArcadeShared.git>



# Chapter 4

## Build Guide

### 4.1 Introduction

#### 4.1.0.1 Project Requirements:

You can build your project using either a Makefile or CMake.

Your CMakeLists.txt must build a program, at least three graphics dynamic libraries and at least two game dynamic libraries at the root of the repository.

### 4.2 @code{bash}

```
/B-OOP-400> mkdir ./build/ && cd ./build/ /B-OOP-400> cmake .. -G "Unix Makefiles" -DCMAKE_BUILD_TYPE=Release [...] /B-OOP-400> cmake --build . [...] /B-OOP-400> cd .. /B-OOP-400> ls ./arcade ./lib/ ./arcade ./lib/: arcade_ncurses.so arcade_pacman.so arcade_sdl2.so arcade_sfml.so arcade_solarfox.so
```

#### 4.2.0.1 Why CMake for the project?

1. CMake provides a simpler and more abstracted way to describe the build process compared to writing Makefiles directly. This can make it easier for developers to understand and maintain the build system, especially for larger projects.
2. CMake supports more advanced features than traditional Makefiles, such as the ability to automatically locate dependencies, manage library dependencies, and generate package configuration files.

---

#### 4.2.0.2 The way we do it !

Currently we have a **Makefile** that calls our **CMake** files, thus making it easier for everyone on the project to build because we are more used to it.

We have a CMake file at the root of the repository **that doesnt need to be changed for the rest of the project**.

### 4.3 How to build the libraries and games

#### 4.3.0.1 Building a graphic library

To build our libraries we have a CMake file at in the `src/graphics` folder that has a function for creating the dynamic libraries.

To build a graphical library for the ncurses for example you need two things:

1. A folder named `ncurses` in `src/graphical`
2. Add this line after the comment in the CMake file: `create_graphic_library(ncurses)`

Everything will be taken care of.

#### 4.3.0.2 Building a game

The process will be the same. We have a CMake file at the `src/games`, that has a function for creating games. To build a game like snake you need two things:

1. A folder named `snake` in `src/games`
2. Add this line after the comment in the CMake file `create_game (snake)`

### 4.4 !!!Warning

The build wont work in these cases:

- You have two folders with the same name
- You have a folder with the incorrect name
- The CMake files are moved in the project

### 4.5 Building the core

The core has a CMake file that is normally fine, and shouldn't be changed throughout the project

## Chapter 5

# How to use arcade in game

### 5.1 :information\_source: Introduction

In the arcade when you play your games there will be opportunities to change games at any time.  
But there will also be the possibility to change graphic libraries while you are in the middle of games.  
And finally you will have the possibility to return to the menu to change nickname, games, graphic libraries or even see if you beat their best score.  
Without forgetting that you can obviously leave the arcade.

---

### 5.2 How that work ?

To exit the arcade, simply press the **a** button.  
To return to the menu, simply press the **e** key.  
To change graphics libraries and use the next one, simply press the **d** key.  
To change the graphic libraries and use the previous one, simply press the **q** key.

---





## Chapter 6

# How to use the menu

### 6.1 Introduction

#### 6.1.0.1 :information\_source: Menu Explication:

In the Arcade project when you launch it you will have the opportunity to choose the game you want to launch. You will also have the opportunity to choose the graphic libraries with which you want to launch the game. And you will have to indicate your name without it impossible to launch the game.

---

### 6.2 How that work ?

To indicate your nickname I would let you type on the keys of your keyboard nothing simpler :grin:

To delete the last letter of the nickname you will have to press the key **F1**

To exit the arcade you will need to press the **F2**

Finally to launch the arcade with the graphic library chooses, the game chooses and your nickname, you will have to press the key **F3**

---

### 6.3 What it looks like ?



# Chapter 7

## arcade

Cross-platform arcade game emulator

### 7.1 :information\_source: Overview

The Arcade project is an interactive and modular gaming platform developed as part of the curriculum at Epitech. The project challenges students to create a flexible gaming framework capable of dynamically loading different game libraries and graphical user interfaces (GUIs).

### 7.2 :book: Learning Objectives

- **Software Design:** Designing and implementing a modular and extensible software architecture.
- **Dynamic Linking:** Understanding dynamic linking and the use of dynamic libraries in C++.
- **Graphical Programming:** Gaining experience in graphical programming and interfacing with different graphics libraries.
- **Project Management:** Collaborating in teams, managing project timelines, and ensuring effective communication.

### 7.3 :page\_facing\_up: Project Info

- :package: [Epitech Private Repository](#)
- :package: [Group Remote Repository](#)

### 7.4 :computer: Developpers

- [Axel](#)
- [Baptiste](#)
- [Dragos](#)

### 7.5 :electric\_plug: Group Collaboration

The project has been realized in collaboration with this [other group](#). Their group master is [Mathéo Coquet](#) that can be contacted at `**mathéo .coquet@epitech.eu**`.



## Chapter 8

# Hierarchical Index

### 8.1 Class Hierarchy

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

shared::games::Author	??
shared::graphics::Author	??
shared::types::ColorType	??
Core	??
CoreUtils	??
Directory	??
DLLoader< T >	??
std::exception	
shared::graphics::exceptions::IGraphicsException	??
shared::graphics::exceptions::IFontException	??
SDL2FontException	??
SFMLFontException	??
shared::graphics::exceptions::ISoundException	??
SDL2SoundException	??
SFMSoundException	??
shared::graphics::exceptions::ITextureException	??
NcursesTextureException	??
SDL2TextureException	??
SFMLTextureException	??
shared::graphics::exceptions::IWindowException	??
NcursesWindowException	??
SDL2WindowException	??
SFMLWindowException	??
GameList	??
shared::games::GameManifest	??
GraphicList	??
shared::graphics::GraphicsManifest	??
shared::games::components::IComponent	??
AComponent	??
AKeyboardComponent	??
SolarFoxPlayerKeyboard	??
PositionableComponent	??
ACollidableComponent	??
SolarFoxPlayerCollidable	??
SolarFoxPowerupCollidable	??
SolarFoxProjectileCollidable	??
ADisplayableComponent	??
ATextComponent	??
TextComponent	??

TextureComponent	??
shared::games::components::IKeyboardComponent	??
AKeyboardComponent	??
SnakeHeadKeyboard	??
shared::games::components::IPositionableComponent	??
PositionableComponent	??
shared::games::components::ICollidableComponent	??
ACollidableComponent	??
AppleCollidable	??
SnakeBodyCollidable	??
SnakeHeadCollidable	??
SnakeTailCollidable	??
shared::games::components::IDisplayableComponent	??
ADisplayableComponent	??
shared::games::components::ITextComponent	??
ATextComponent	??
ScoreTextDisplayable	??
TextComponent	??
shared::games::components::ITextureComponent	??
AppleDisplayable	??
BackgroundDisplayable	??
SnakeBodyDisplayable	??
SnakeHeadDisplayable	??
SnakeTailDisplayable	??
TextureComponent	??
shared::games::components::ISoundComponent	??
ASoundComponent	??
shared::games::entity::IEntity	??
AEntity	??
SolarFoxEnemy	??
SolarFoxPlayer	??
SolarFoxPowerup	??
SolarFoxProjectile	??
SolarFoxScore	??
AppleEntity	??
BackgroundEntity	??
ScoreTextEntity	??
SnakeBodyEntity	??
SnakeHeadEntity	??
SnakeTailEntity	??
shared::graphics::events::IEvent	??
shared::graphics::events::IKeyEvent	??
shared::graphics::events::KeyPressedEvent	??
shared::graphics::events::KeyReleaseEvent	??
shared::graphics::events::IMouseEvent	??
shared::graphics::events::IMouseButtonEvent	??
shared::graphics::events::MouseButtonPressEvent	??
shared::graphics::events::MouseButtonReleaseEvent	??
shared::graphics::events::MouseMoveEvent	??
shared::graphics::events::WindowCloseEvent	??
shared::graphics::events::WindowResizeEvent	??
shared::graphics::IFont	??
NcursesFont	??
SDL2Font	??
SFMLFont	??
shared::games::IGame	??

SnakeGame	??
SolarFoxGame	??
shared::games::IGameProvider	??
SnakeGameProvider	??
SolarFoxProvider	??
shared::graphics::IGraphicsProvider	??
Ncurses	??
SDL2	??
SFML	??
shared::graphics::ISound	??
NcursesSound	??
SDL2Sound	??
SFMLSound	??
shared::graphics::ITexture	??
NcursesTexture	??
SDL2Texture	??
SFMLTexture	??
shared::graphics::IWindow	??
NcursesWindow	??
SDL2Window	??
SFMLWindow	??
shared::games::components::IKeyboardComponent::KeyCode	??
shared::graphics::events::IKeyEvent::KeyCode	??
shared::games::components::IKeyboardComponent::KeyData	??
Core::SoundMapProps	??
shared::games::components::ITextComponent::TextFontProps	??
shared::games::components::ITextComponent::TextProps	??
shared::graphics::TextProps	??
shared::games::components::TextureProps	??
shared::graphics::TextureProps	??
shared::games::components::TextureSources	??
shared::types::Vector< T >	??
shared::types::Vector< float >	??
shared::types::Vector< int >	??
shared::graphics::IWindow::WindowInitProps	??





## Chapter 9

# Class Index

### 9.1 Class List

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

<a href="#">ACollidableComponent</a>	??
<a href="#">AComponent</a>	??
<a href="#">ADisplayableComponent</a>	??
<a href="#">AEntity</a>	??
<a href="#">AKeyboardComponent</a>	??
<a href="#">AppleCollidable</a>	??
<a href="#">AppleDisplayable</a>	??
<a href="#">AppleEntity</a>	??
<a href="#">ASoundComponent</a>	??
<a href="#">ATextComponent</a>	??
<a href="#">shared::games::Author</a>	
Author of the game	??
<a href="#">shared::graphics::Author</a>	
Author of the graphics library	??
<a href="#">BackgroundDisplayable</a>	??
<a href="#">BackgroundEntity</a>	??
<a href="#">shared::types::ColorType</a>	
Color type	??
<a href="#">Core</a>	??
<a href="#">CoreUtils</a>	??
<a href="#">Directory</a>	??
<a href="#">DLLoader&lt; T &gt;</a>	??
<a href="#">GameList</a>	??
<a href="#">shared::games::GameManifest</a>	
Game manifest	??
<a href="#">GraphicList</a>	??
<a href="#">shared::graphics::GraphicsManifest</a>	
Graphics library manifest	??
<a href="#">shared::games::components::ICollidableComponent</a>	
Interface of a collidable component	??
<a href="#">shared::games::components::IComponent</a>	
Interface of a component	??
<a href="#">shared::games::components::IDisplayableComponent</a>	
Interface of a displayable component	??
<a href="#">shared::games::entity::IEntity</a>	
Interface of an entity	??
<a href="#">shared::graphics::events::IEvent</a>	
Interface for the event object	??
<a href="#">shared::graphics::IFont</a>	
Interface of a font	??

<a href="#">shared::graphics::exceptions::IFontException</a>	
Interface for the font exception object . . . . .	??
<a href="#">shared::games::IGame</a>	
Interface of a game . . . . .	??
<a href="#">shared::games::IGameProvider</a>	
Interface of a game provider . . . . .	??
<a href="#">shared::graphics::exceptions::IGraphicsException</a>	
Interface for the graphics exception object . . . . .	??
<a href="#">shared::graphics::IGraphicsProvider</a>	
Interface for the graphics provider . . . . .	??
<a href="#">shared::games::components::IKeyboardComponent</a>	
Interface of a keyboard component . . . . .	??
<a href="#">shared::graphics::events::IKeyEvent</a>	
Interface for the key event object . . . . .	??
<a href="#">shared::graphics::events::IMouseButtonEvent</a>	
Interface for the mouse button event object . . . . .	??
<a href="#">shared::graphics::events::IMouseEvent</a>	
Interface for the mouse event object . . . . .	??
<a href="#">shared::games::components::IPositionableComponent</a>	
Interface of a positionable component . . . . .	??
<a href="#">shared::graphics::ISound</a>	
Interface for the sound object . . . . .	??
<a href="#">shared::games::components::ISoundComponent</a>	
Interface of a sound component . . . . .	??
<a href="#">shared::graphics::exceptions::ISoundException</a>	
Interface for the sound exception object . . . . .	??
<a href="#">shared::games::components::ITextComponent</a>	
Interface of a text component . . . . .	??
<a href="#">shared::graphics::ITexture</a>	
Interface for the texture object . . . . .	??
<a href="#">shared::games::components::ITextureComponent</a>	
Interface of a texture component . . . . .	??
<a href="#">shared::graphics::exceptions::ITextureException</a>	
Interface for the texture exception object . . . . .	??
<a href="#">shared::graphics::IWindow</a>	
Interface for the window object . . . . .	??
<a href="#">shared::graphics::exceptions::IWindowException</a>	
Interface for the window exception object . . . . .	??
<a href="#">shared::games::components::IKeyboardComponent::KeyCode</a>	
Function key code union . . . . .	??
<a href="#">shared::graphics::events::IKeyEvent::KeyCode</a>	
Key code . . . . .	??
<a href="#">shared::games::components::IKeyboardComponent::KeyData</a>	
Key data . . . . .	??
<a href="#">shared::graphics::events::KeyPressedEvent</a>	??
<a href="#">shared::graphics::events::KeyReleaseEvent</a>	??
<a href="#">shared::graphics::events::MouseButtonPressEvent</a>	??
<a href="#">shared::graphics::events::MouseButtonReleaseEvent</a>	??
<a href="#">shared::graphics::events::MouseMoveEvent</a>	??
<a href="#">Ncurses</a>	??
<a href="#">NcursesFont</a>	??
<a href="#">NcursesSound</a>	??
<a href="#">NcursesTexture</a>	??
<a href="#">NcursesTextureException</a>	??
<a href="#">NcursesWindow</a>	??
<a href="#">NcursesWindowException</a>	??
<a href="#">PositionableComponent</a>	??
<a href="#">ScoreTextDisplayable</a>	??

ScoreTextEntity	??
SDL2	??
SDL2Font	??
SDL2FontException	??
SDL2Sound	??
SDL2SoundException	??
SDL2Texture	??
SDL2TextureException	??
SDL2Window	??
SDL2WindowException	??
SFML	??
SFMLFont	??
SFMLFontException	??
SFMLSound	??
SFMLSoundException	??
SFMLTexture	??
SFMLTextureException	??
SFMLWindow	??
SFMLWindowException	??
SnakeBodyCollidable	??
SnakeBodyDisplayable	??
SnakeBodyEntity	??
SnakeGame	??
SnakeGameProvider	??
SnakeHeadCollidable	??
SnakeHeadDisplayable	??
SnakeHeadEntity	??
SnakeHeadKeyboard	??
SnakeTailCollidable	??
SnakeTailDisplayable	??
SnakeTailEntity	??
SolarFoxEnemy	??
SolarFoxGame	??
SolarFoxPlayer	??
SolarFoxPlayerCollidable	??
SolarFoxPlayerKeyboard	??
SolarFoxPowerup	??
SolarFoxPowerupCollidable	??
SolarFoxProjectile	??
SolarFoxProjectileCollidable	??
SolarFoxProvider	??
SolarFoxScore	??
Core::SoundMapProps	??
TextComponent	??
shared::games::components::ITextComponent::TextFontProps	
Font properties	??
shared::games::components::ITextComponent::TextProps	
Text properties	??
shared::graphics::TextProps	
Text properties	??
TextureComponent	??
shared::games::components::TextureProps	
Texture properties	??
shared::graphics::TextureProps	
Texture properties	??
shared::games::components::TextureSources	
Texture sources	??

---

<a href="#">shared::types::Vector&lt; T &gt;</a>	
<a href="#">Vector</a> type . . . . .	??
<a href="#">shared::graphics::events::WindowCloseEvent</a> . . . . .	??
<a href="#">shared::graphics::IWindow::WindowInitProps</a>	
Window initial properties . . . . .	??
<a href="#">shared::graphics::events::WindowResizeEvent</a> . . . . .	??

# Chapter 10

## Class Documentation

### 10.1 ACollidableComponent Class Reference

Inheritance diagram for ACollidableComponent:

### 10.2 AComponent Class Reference

Inheritance diagram for AComponent:

Collaboration diagram for AComponent:

#### Public Member Functions

- **AComponent** ([entity::IEntity](#) &entity, components::ComponentType type)
- const components::ComponentType [getType](#) () const noexcept override  
*Get the type of the component.*
- const [entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the parent entity of the component.*

#### Protected Attributes

- components::ComponentType **\_type**
- [entity::IEntity](#) & **\_entity**

#### 10.2.1 Member Function Documentation

##### 10.2.1.1 [getEntity\(\)](#)

```
const entity::IEntity & AComponent::getEntity ( ) [override], [virtual], [noexcept]
```

Get the parent entity of the component.

##### Returns

Entity of the component

Implements [shared::games::components::IComponent](#).

##### 10.2.1.2 [getType\(\)](#)

```
const components::ComponentType AComponent::getType ( ) const [override], [virtual], [noexcept]
```

Get the type of the component.

**Returns**

Type of the component

Implements [shared::games::components::IComponent](#).

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

- [src/games/abstracts/components/AComponent.hpp](#)
- [src/games/abstracts/components/AComponent.cpp](#)

## 10.3 ADisplayableComponent Class Reference

Inheritance diagram for ADisplayableComponent:

Collaboration diagram for ADisplayableComponent:

**Public Member Functions**

- **ADisplayableComponent** ([shared::types::Vector2f](#) position, [shared::types::Vector2u](#) size, [entity::IEntity](#) &entity, unsigned int zIndex, components::ComponentType type)
- unsigned int & [getZIndex](#) () noexcept override  
*Get Z index that is usefull for display priority.*

**Protected Attributes**

- unsigned int **\_zIndex**

### 10.3.1 Member Function Documentation

#### 10.3.1.1 getZIndex()

```
unsigned int & ADisplayableComponent::getZIndex ( ) [override], [virtual], [noexcept]
```

Get Z index that is usefull for display priority.

**Returns**

Z index of the entity

Implements [shared::games::components::IDisplayableComponent](#).

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

- [src/games/abstracts/components/ADisplayableComponent.hpp](#)
- [src/games/abstracts/components/ADisplayableComponent.cpp](#)

## 10.4 AEntity Class Reference

Inheritance diagram for AEntity:

Collaboration diagram for AEntity:

**Public Member Functions**

- const components::ComponentsMap & [getComponents](#) (void) const noexcept override  
*Get the components of the entity.*

**Protected Attributes**

- components::ComponentsMap **\_components**

## 10.4.1 Member Function Documentation

### 10.4.1.1 getComponents()

```
const shared::games::components::ComponentsMap & AEntity::getComponents (
    void ) const [override], [virtual], [noexcept]
```

Get the components of the entity.

Returns

Components of the entity

Implements [shared::games::entity::IEntity](#).

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

- src/games/abstracts/entity/AEntity.hpp
- src/games/abstracts/entity/AEntity.cpp

## 10.5 AKeyboardComponent Class Reference

Inheritance diagram for AKeyboardComponent:

Collaboration diagram for AKeyboardComponent:

### Public Member Functions

- **AKeyboardComponent** ([entity::IEntity](#) &entity)

### Additional Inherited Members

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

- src/games/abstracts/components/AKeyboardComponent.hpp
- src/games/abstracts/components/AKeyboardComponent.cpp

## 10.6 AppleCollidable Class Reference

Inheritance diagram for AppleCollidable:

Collaboration diagram for AppleCollidable:

### Public Member Functions

- **AppleCollidable** (const [shared::games::entity::IEntity](#) &entity)  
*Construct a new Apple Collidable object.*
- **~AppleCollidable** ()  
*Destroy the Apple Collidable object.*
- const [shared::games::components::ComponentType](#) **getType** () const noexcept override  
*Get the Type object.*
- const [shared::games::entity::IEntity](#) & **getEntity** () noexcept override  
*Get the entity object.*
- void **setPosition** ([Vector2f](#) pos) noexcept
- [Vector2f](#) & **getPosition** (void) noexcept override  
*Get position of the entity (tiles)*
- [Vector2u](#) & **getSize** (void) noexcept override  
*Get size of the entity (tiles)*
- void **onCollide** (std::shared\_ptr< [shared::games::IGame](#) > ctx, std::shared\_ptr< [shared::games::components::ICollidableComp](#) > target) override  
*On collide event handler for the component.*

## Public Attributes

- unsigned int `_score`

## 10.6.1 Constructor & Destructor Documentation

### 10.6.1.1 AppleCollidable()

```
AppleCollidable::AppleCollidable (
    const shared::games::entity::IEntity & entity ) [explicit]
```

Construct a new Apple Collidable object.

#### Parameters

<i>entity</i>	
---------------	--

## 10.6.2 Member Function Documentation

### 10.6.2.1 getEntity()

```
const shared::games::entity::IEntity & AppleCollidable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

#### Returns

const [shared::games::entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.6.2.2 getType()

```
const shared::games::components::ComponentType AppleCollidable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

#### Returns

const [shared::games::components::ComponentType](#)

Implements [shared::games::components::IComponent](#).

### 10.6.2.3 onCollide()

```
void AppleCollidable::onCollide (
    std::shared_ptr< shared::games::IGame > ctx,
    std::shared_ptr< shared::games::components::ICollidableComponent > target ) [override],
[virtual]
```

On collide event handler for the component.

#### Parameters

<i>ctx</i>	Context of the game
<i>target</i>	Target entity



Implements [shared::games::components::ICollidableComponent](#).

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

- `src/games/snake/entities/apple/components/AppleCollidable.hpp`
- `src/games/snake/entities/apple/components/AppleCollidable.cpp`

## 10.7 AppleDisplayable Class Reference

Inheritance diagram for AppleDisplayable:

Collaboration diagram for AppleDisplayable:

### Public Member Functions

- [AppleDisplayable](#) (const [shared::games::entity::IEntity](#) &entity)  
*Construct a new Apple Displayable object.*
- [~AppleDisplayable](#) ()  
*Destroy the Apple Displayable object.*
- const [shared::games::components::ComponentType](#) [getType](#) () const noexcept override  
*Get the Type object.*
- const [shared::games::entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get the Size object.*
- unsigned int & [getZIndex](#) (void) noexcept override  
*Get the ZIndex object.*
- [shared::games::components::TextureProps](#) & [getTextureProps](#) (void) noexcept override  
*Get the TextureProps object.*
- void [onMousePress](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override  
*Handle the mouse press event.*
- void [onMouseHover](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override  
*Handle the mouse hover event.*
- void [onMouseRelease](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override  
*Handle the mouse release event.*
- [Vector2f](#) & [getPosition](#) (void) noexcept override  
*Get the Position object.*

### Public Attributes

- [Vector2f](#) [\\_position](#)

### 10.7.1 Constructor & Destructor Documentation

#### 10.7.1.1 AppleDisplayable()

```
AppleDisplayable::AppleDisplayable (
    const shared::games::entity::IEntity & entity ) [explicit]
```

Construct a new Apple Displayable object.

Parameters

<code>entity</code>	
---------------------	--

## 10.7.2 Member Function Documentation

### 10.7.2.1 getEntity()

```
const shared::games::entity::IEntity & AppleDisplayable::getEntity (  
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

Returns

const shared::games::entity::IEntity&

Implements [shared::games::components::IComponent](#).

### 10.7.2.2 getPosition()

```
Vector2f & AppleDisplayable::getPosition (  
    void ) [override], [virtual], [noexcept]
```

Get the Position object.

Returns

Vector2f&

Implements [shared::games::components::IPositionableComponent](#).

### 10.7.2.3 getSize()

```
Vector2u & AppleDisplayable::getSize (  
    void ) [override], [virtual], [noexcept]
```

Get the Size object.

Returns

const Vector2u&

Implements [shared::games::components::IPositionableComponent](#).

### 10.7.2.4 getTextureProps()

```
shared::games::components::TextureProps & AppleDisplayable::getTextureProps (  
    void ) [override], [virtual], [noexcept]
```

Get the TextureProps object.

Returns

[shared::games::components::TextureProps](#)&

Implements [shared::games::components::ITextureComponent](#).

### 10.7.2.5 getType()

```
const shared::games::components::ComponentType AppleDisplayable::getType (  
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

Returns

const shared::games::components::ComponentType

Implements [shared::games::components::IComponent](#).

**10.7.2.6 getZIndex()**

```
unsigned int & AppleDisplayable::getZIndex (
    void ) [override], [virtual], [noexcept]
```

Get the ZIndex object.

**Returns**

unsigned int&

Implements [shared::games::components::IDisplayableComponent](#).

**10.7.2.7 onMouseHover()**

```
void AppleDisplayable::onMouseHover (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Handle the mouse hover event.

**Parameters**

<i>ctx</i>	
------------	--

Implements [shared::games::components::IDisplayableComponent](#).

**10.7.2.8 onMousePress()**

```
void AppleDisplayable::onMousePress (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Handle the mouse press event.

**Parameters**

<i>ctx</i>	
------------	--

Implements [shared::games::components::IDisplayableComponent](#).

**10.7.2.9 onMouseRelease()**

```
void AppleDisplayable::onMouseRelease (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Handle the mouse release event.

**Parameters**

<i>ctx</i>	
------------	--

Implements [shared::games::components::IDisplayableComponent](#).

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

- `src/games/snake/entities/apple/components/AppleDisplayable.hpp`
- `src/games/snake/entities/apple/components/AppleDisplayable.cpp`

**10.8 AppleEntity Class Reference**

Inheritance diagram for AppleEntity:

Collaboration diagram for AppleEntity:

## Public Member Functions

- [AppleEntity](#) ()  
*Construct a new Apple Entity object.*
- [~AppleEntity](#) ()  
*Destroy the Apple Entity object.*
- const shared::games::components::ComponentsMap & [getComponents](#) (void) const noexcept override  
*Get the Components object.*

### 10.8.1 Member Function Documentation

#### 10.8.1.1 getComponents()

```
const shared::games::components::ComponentsMap & AppleEntity::getComponents (
    void ) const [override], [virtual], [noexcept]
```

Get the Components object.

#### Returns

const shared::games::components::ComponentsMap&

Implements [shared::games::entity::IEntity](#).

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

- src/games/snake/entities/apple/AppleEntity.hpp
- src/games/snake/entities/apple/AppleEntity.cpp

## 10.9 ASoundComponent Class Reference

Inheritance diagram for ASoundComponent:

Collaboration diagram for ASoundComponent:

## Public Member Functions

- **ASoundComponent** (const [entity::IEntity](#) &\_entity, const std::string &path)
- const std::string & [getPath](#) (void) const noexcept override  
*Get path of the sound.*
- components::SoundState & [getState](#) (void) noexcept override  
*Get state of the sound.*
- components::SoundVolume & [getVolume](#) (void) noexcept override  
*Get volume of the sound.*
- bool & [getLoop](#) (void) noexcept override  
*Get loop of the sound.*
- const components::ComponentType [getType](#) (void) const noexcept override  
*Get the type of the component.*
- const [entity::IEntity](#) & [getEntity](#) (void) noexcept override  
*Get the parent entity of the component.*

## Protected Attributes

- std::string **\_path**
- components::SoundState **\_state**
- components::SoundVolume **\_volume**
- bool **\_loop**
- const [entity::IEntity](#) & **\_entity**

## 10.9.1 Member Function Documentation

### 10.9.1.1 getEntity()

```
const entity::IEntity & ASoundComponent::getEntity (  
    void ) [override], [virtual], [noexcept]
```

Get the parent entity of the component.

#### Returns

Entity of the component

Implements [shared::games::components::IComponent](#).

### 10.9.1.2 getLoop()

```
bool & ASoundComponent::getLoop (  
    void ) [override], [virtual], [noexcept]
```

Get loop of the sound.

#### Returns

Sound loop

Implements [shared::games::components::ISoundComponent](#).

### 10.9.1.3 getPath()

```
const std::string & ASoundComponent::getPath (  
    void ) const [override], [virtual], [noexcept]
```

Get path of the sound.

#### Returns

Sound path

Implements [shared::games::components::ISoundComponent](#).

### 10.9.1.4 getState()

```
components::SoundState & ASoundComponent::getState (  
    void ) [override], [virtual], [noexcept]
```

Get state of the sound.

#### Returns

Sound state

Implements [shared::games::components::ISoundComponent](#).

### 10.9.1.5 getType()

```
const components::ComponentType ASoundComponent::getType (  
    void ) const [override], [virtual], [noexcept]
```

Get the type of the component.

#### Returns

Type of the component

Implements [shared::games::components::IComponent](#).

### 10.9.1.6 getVolume()

```
components::SoundVolume & ASoundComponent::getVolume (
    void ) [override], [virtual], [noexcept]
```

Get volume of the sound.

#### Returns

Sound volume

Implements [shared::games::components::ISoundComponent](#).

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

- [src/games/abstracts/components/ASoundComponent.hpp](#)
- [src/games/abstracts/components/ASoundComponent.cpp](#)

## 10.10 ATextComponent Class Reference

Inheritance diagram for ATextComponent:

Collaboration diagram for ATextComponent:

### Public Member Functions

- **ATextComponent** ([shared::types::Vector2f](#) position, [shared::types::Vector2u](#) size, [entity::IEntity](#) &entity, unsigned int zIndex, [components::ITextComponent::TextProps](#) &textProps)
- [components::ITextComponent::TextProps](#) **getTextProps** () noexcept override  
*Get text props of the entity.*

### Protected Attributes

- [components::ITextComponent::TextProps](#) **\_textProps**

### Additional Inherited Members

### 10.10.1 Member Function Documentation

#### 10.10.1.1 getTextProps()

```
components::ITextComponent::TextProps ATextComponent::getTextProps ( ) [override], [virtual],
[noexcept]
```

Get text props of the entity.

#### Returns

text props

Implements [shared::games::components::ITextComponent](#).

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

- [src/games/abstracts/components/ATextComponent.hpp](#)
- [src/games/abstracts/components/ATextComponent.cpp](#)

## 10.11 shared::games::Author Struct Reference

[Author](#) of the game.

```
#include <GameManifest.hpp>
```

## Public Attributes

- std::string [name](#)  
*Name of the author.*
- std::string [email](#)  
*Public contact email.*
- std::string [website](#)  
*Website of the author (github, gitlab, etc.)*

### 10.11.1 Detailed Description

[Author](#) of the game.

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

- common/games/types/GameManifest.hpp

## 10.12 shared::graphics::Author Struct Reference

[Author](#) of the graphics library.

```
#include <GraphicsManifest.hpp>
```

## Public Attributes

- std::string [name](#)  
*Name of the author.*
- std::string [email](#)  
*Public contact email.*
- std::string [website](#)  
*Website of the author (github, gitlab, etc.)*

### 10.12.1 Detailed Description

[Author](#) of the graphics library.

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

- common/graphics/types/GraphicsManifest.hpp

## 10.13 BackgroundDisplayable Class Reference

Inheritance diagram for BackgroundDisplayable:

Collaboration diagram for BackgroundDisplayable:

## Public Member Functions

- [BackgroundDisplayable](#) (const [shared::games::entity::IEntity](#) &entity)  
*Construct a new Background Displayable object.*
- [~BackgroundDisplayable](#) ()  
*Destroy the Background Displayable object.*
- const [shared::games::components::ComponentType](#) [getType](#) () const noexcept override  
*Get the Type object.*
- const [shared::games::entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get the Size object.*
- unsigned int & [getZIndex](#) (void) noexcept override

*Get the ZIndex object.*

- [shared::games::components::TextureProps](#) & [getTextureProps](#) (void) noexcept override

*Get the TextureProps object.*

- void [onMousePress](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override

*Handle the mouse press event.*

- void [onMouseHover](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override

*Handle the mouse hover event.*

- void [onMouseRelease](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override

*Handle the mouse release event.*

- [Vector2f](#) & [getPosition](#) (void) noexcept override

*Get the Position object.*

## Public Attributes

- [Vector2f](#) [\\_position](#)

## 10.13.1 Constructor & Destructor Documentation

### 10.13.1.1 BackgroundDisplayable()

```
BackgroundDisplayable::BackgroundDisplayable (
    const shared::games::entity::IEntity & entity ) [explicit]
```

Construct a new Background Displayable object.

Parameters

<a href="#">entity</a>	
------------------------	--

## 10.13.2 Member Function Documentation

### 10.13.2.1 getEntity()

```
const shared::games::entity::IEntity & BackgroundDisplayable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

Returns

const [shared::games::entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.13.2.2 getPosition()

```
Vector2f & BackgroundDisplayable::getPosition (
    void ) [override], [virtual], [noexcept]
```

Get the Position object.

Returns

[Vector2f](#)&

Implements [shared::games::components::IPositionableComponent](#).



**10.13.2.3 getSize()**

```
Vector2u & BackgroundDisplayable::getSize (
    void ) [override], [virtual], [noexcept]
```

Get the Size object.

Returns

const Vector2u&

Implements [shared::games::components::IPositionableComponent](#).

**10.13.2.4 getTextureProps()**

```
shared::games::components::TextureProps & BackgroundDisplayable::getTextureProps (
    void ) [override], [virtual], [noexcept]
```

Get the TextureProps object.

Returns

[shared::games::components::TextureProps](#)&

Implements [shared::games::components::ITextureComponent](#).

**10.13.2.5 getType()**

```
const shared::games::components::ComponentType BackgroundDisplayable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

Returns

const shared::games::components::ComponentType

Implements [shared::games::components::IComponent](#).

**10.13.2.6 getZIndex()**

```
unsigned int & BackgroundDisplayable::getZIndex (
    void ) [override], [virtual], [noexcept]
```

Get the ZIndex object.

Returns

unsigned int&

Implements [shared::games::components::IDisplayableComponent](#).

**10.13.2.7 onMouseHover()**

```
void BackgroundDisplayable::onMouseHover (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Handle the mouse hover event.

Parameters

<i>ctx</i>	
------------	--

Implements [shared::games::components::IDisplayableComponent](#).

### 10.13.2.8 onMousePress()

```
void BackgroundDisplayable::onMousePress (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Handle the mouse press event.

#### Parameters

<i>ctx</i>	
------------	--

Implements [shared::games::components::IDisplayableComponent](#).

### 10.13.2.9 onMouseRelease()

```
void BackgroundDisplayable::onMouseRelease (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Handle the mouse release event.

#### Parameters

<i>ctx</i>	
------------	--

Implements [shared::games::components::IDisplayableComponent](#).

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

- `src/games/snake/entities/background/components/BackgroundDisplayable.hpp`
- `src/games/snake/entities/background/components/BackgroundDisplayable.cpp`

## 10.14 BackgroundEntity Class Reference

Inheritance diagram for BackgroundEntity:

Collaboration diagram for BackgroundEntity:

### Public Member Functions

- [BackgroundEntity](#) ()  
*Construct a new Background Entity object.*
- [~BackgroundEntity](#) ()  
*Destroy a new Background Entity object.*
- `const shared::games::components::ComponentsMap & getComponents (void) const noexcept override`  
*Get the Components object.*

### 10.14.1 Member Function Documentation

#### 10.14.1.1 getComponents()

```
const shared::games::components::ComponentsMap & BackgroundEntity::getComponents (
    void ) const [override], [virtual], [noexcept]
```

Get the Components object.

**Returns**

const components::ComponentsMap&

Implements [shared::games::entity::IEntity](#).

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

- src/games/snake/entities/background/BackgroundEntity.hpp
- src/games/snake/entities/background/BackgroundEntity.cpp

## 10.15 shared::types::ColorType Struct Reference

Color type.

```
#include <Color.hpp>
```

**Public Member Functions**

- [ColorType](#) (unsigned char [r](#), unsigned char [g](#), unsigned char [b](#), unsigned char [a](#))  
*Construct a new Color Type object.*

**Public Attributes**

- unsigned char [r](#)  
*Red color value.*
- unsigned char [g](#)  
*Green color value.*
- unsigned char [b](#)  
*Blue color value.*
- unsigned char [a](#)  
*Alpha color value.*

### 10.15.1 Detailed Description

Color type.

### 10.15.2 Constructor & Destructor Documentation

#### 10.15.2.1 ColorType()

```
shared::types::ColorType::ColorType (
    unsigned char r,
    unsigned char g,
    unsigned char b,
    unsigned char a ) [inline]
```

Construct a new Color Type object.

**Parameters**

<i>r</i>	Red color value
<i>g</i>	Green color value
<i>b</i>	Blue color value
<i>a</i>	Alpha color value

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

- `common/types/Color.hpp`

## 10.16 Core Class Reference

### Classes

- struct `SoundMapProps`

### Public Member Functions

- `Core` (`std::string defaultLib`)  
*Constructor of `Core` Class.*
- `~Core` ()  
*Destructor of `Core` Class.*
- `std::size_t runArcade` ()  
*Run the arcade main function.*
- `bool getLaunchArcade` () const  
*Get a bool value to tell if the arcade should be launched.*
- `void runMenu` ()  
*run the menu*

### 10.16.1 Member Function Documentation

#### 10.16.1.1 `getLaunchArcade()`

`bool Core::getLaunchArcade ( ) const`

Get a bool value to tell if the arcade should be launched.

##### Returns

`bool`

#### 10.16.1.2 `runArcade()`

`std::size_t Core::runArcade ( )`

Run the arcade main function.

##### Returns

`std::size_t`

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

- `src/core/Core.hpp`
- `src/core/Core.cpp`
- `src/core/HighScore.cpp`
- `src/core/Menu.cpp`

## 10.17 CoreUtils Class Reference

### Public Member Functions

- `CoreUtils` ()  
*Construct a new `Core Utils` object.*
- `~CoreUtils` ()  
*Destroy the `Core Utils` object.*

## Static Public Member Functions

- static `components::IKeyboardComponent::KeyData` `convertKey` (const `events::IKeyEvent::KeyCode` &keyCode, const `events::IKeyEvent::KeyType` &keyType)  
*Convert a key data.*
- static `TextAlign` `mapTextAlign` (const `components::ITextComponent::TextAlign` &align)  
*Align a text component.*
- static `TextVerticalAlign` `mapTextVerticalAlign` (const `components::ITextComponent::TextVerticalAlign` &align)  
*Vertically align a text component.*
- static `bool` `isDisplayablePressed` (std::shared\_ptr< `components::IDisplayableComponent` > displayable, std::shared\_ptr< `events::IMouseEvent` > mouseEvent)  
*Check if a displayable component is pressed.*
- static `shared::graphics::ISound::SoundState` `mapSoundState` (const `shared::games::components::SoundState` state)  
*Map a sound state paused / stopped / playing.*
- static `bool` `checkCollision` (std::shared\_ptr< `components::ICollidableComponent` > collidable1, std::shared\_ptr< `components::ICollidableComponent` > collidable2)  
*Check if two collidable components are colliding.*

### 10.17.1 Member Function Documentation

#### 10.17.1.1 `checkCollision()`

```
bool CoreUtils::checkCollision (
    std::shared_ptr< components::ICollidableComponent > collidable1,
    std::shared_ptr< components::ICollidableComponent > collidable2 ) [static]
```

Check if two collidable components are colliding.

##### Parameters

<i>collidable1</i>	
<i>collidable2</i>	

##### Returns

true  
false

#### 10.17.1.2 `convertKey()`

```
components::IKeyboardComponent::KeyData CoreUtils::convertKey (
    const events::IKeyEvent::KeyCode & keyCode,
    const events::IKeyEvent::KeyType & keyType ) [static]
```

Convert a key data.

##### Parameters

<i>align</i>	
--------------	--

**Returns**

std::string

**10.17.1.3 isDisplayablePressed()**

```
bool CoreUtils::isDisplayablePressed (
    std::shared_ptr< components::IDisplayableComponent > displayable,
    std::shared_ptr< events::IMouseEvent > mouseEvent ) [static]
```

Check if a displayable component is pressed.

**Parameters**

<i>displayable</i>	
<i>mouseEvent</i>	

**Returns**

bool

**10.17.1.4 mapSoundState()**

```
shared::graphics::ISound::SoundState CoreUtils::mapSoundState (
    const shared::games::components::SoundState state ) [static]
```

Map a sound state paused / stopped / playing.

**Parameters**

<i>state</i>	
--------------	--

**Returns**

[shared::graphics::ISound::SoundState](#)

**10.17.1.5 mapTextAlign()**

```
TextAlign CoreUtils::mapTextAlign (
    const components::ITextComponent::TextAlign & align ) [static]
```

Align a text component.

**Parameters**

<i>align</i>	
--------------	--

**Returns**

TextAlign

**10.17.1.6 mapTextVerticalAlign()**

```
TextVerticalAlign CoreUtils::mapTextVerticalAlign (
    const components::ITextComponent::TextVerticalAlign & align ) [static]
```

Vertically align a text component.

## Parameters

<i>align</i>	
--------------	--

## Returns

TextVerticalAlign

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

- src/core/utls/CoreUtils.hpp
- src/core/utls/CoreUtils.cpp

## 10.18 Directory Class Reference

### Public Member Functions

- [Directory](#) (const std::string &directoryPath)  
*Construct a new [Directory](#) object.*
- [~Directory](#) ()  
*Destroy the [Directory](#) object.*
- std::vector< std::string > [getListLibraries](#) ()  
*Get the List Libraries object.*

### 10.18.1 Constructor & Destructor Documentation

#### 10.18.1.1 Directory()

```
Directory::Directory (
    const std::string & directoryPath ) [explicit]
```

Construct a new [Directory](#) object.

## Parameters

<i>directoryPath</i>	Path of the directory to open
----------------------	-------------------------------

### 10.18.2 Member Function Documentation

#### 10.18.2.1 getListLibraries()

```
std::vector< std::string > Directory::getListLibraries ( )
```

Get the List Libraries object.

## Returns

std::vector<std::string>

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

- src/core/library/loader/Directory.hpp
- src/core/library/loader/Directory.cpp



## 10.19 DLLoader< T > Class Template Reference

### Public Member Functions

- [DLLoader](#) (const std::string libraryPath)  
*Construct a new [DLLoader](#) object.*
- [~DLLoader](#) ()  
*Destroy the [DLLoader](#) object.*
- shared::types::LibraryType [getType](#) (const std::string functionName)  
*Get the *Type* object.*
- T [getInstance](#) (const std::string functionName)  
*Get the *Instance* object.*

### Protected Attributes

- std::string [\\_libraryPath](#)
- void \* [\\_libraryInstance](#)

### 10.19.1 Constructor & Destructor Documentation

#### 10.19.1.1 DLLoader()

```
template<typename T >
DLLoader< T >::DLLoader (
    const std::string libraryPath ) [inline], [explicit]
Construct a new DLLoader object.
```

##### Parameters

<i>libraryPath</i>	
--------------------	--

### 10.19.2 Member Function Documentation

#### 10.19.2.1 getInstance()

```
template<typename T >
T DLLoader< T >::getInstance (
    const std::string functionName ) [inline]
Get the Instance object.
```

##### Parameters

<i>functionName</i>	
---------------------	--

##### Returns

T

#### 10.19.2.2 getType()

```
template<typename T >
```

```
shared::types::LibraryType DLLoader< T >::getType (
    const std::string functionName ) [inline]
```

Get the Type object.

#### Parameters

<i>functionName</i>	
---------------------	--

#### Returns

shared::types::LibraryType

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

- src/core/library/loader/DLLoader.hpp

## 10.20 GameList Class Reference

### Public Member Functions

- [GameList](#) (std::vector< std::string > allLibrariesPath)  
*Construct a new Graphic List object.*
- [~GameList](#) ()  
*Destroy the Game List object.*
- std::size\_t [getNbGame](#) () const  
*Get the Nb Game object.*
- void [incrementIndex](#) ()  
*Incremente index of Library.*
- void [decrementIndex](#) ()  
*Decremente index of Library.*
- void [setIndex](#) (std::size\_t newIndex)  
*Set index of Library.*
- std::shared\_ptr< [shared::games::IGame](#) > [getCurrentGame](#) ()  
*Get the Current Game object.*
- std::size\_t [getIndex](#) () const noexcept  
*Gets the index of the current lib.*
- std::vector< std::shared\_ptr< [shared::games::IGame](#) > > [getLibraryList](#) ()  
*Return the entire game list.*

### Protected Attributes

- std::vector< std::shared\_ptr< [shared::games::IGame](#) > > [\\_libraryList](#)
- std::vector< std::shared\_ptr< [DLLoader](#)< [shared::games::IGameProvider](#) \* > > > [\\_libraryLoader](#)
- std::size\_t [\\_index](#)

### 10.20.1 Constructor & Destructor Documentation

#### 10.20.1.1 GameList()

```
GameList::GameList (
    std::vector< std::string > allLibrariesPath ) [explicit]
```

Construct a new Graphic List object.

## Parameters

<i>allLibrariesPath</i>	All Libraries path
-------------------------	--------------------

## 10.20.2 Member Function Documentation

### 10.20.2.1 getCurrentGame()

`std::shared_ptr< shared::games::IGame > GameList::getCurrentGame ( )`  
Get the Current Game object.

## Returns

`std::shared_ptr<shared::games::IGame>` Current Game Library

### 10.20.2.2 getIndex()

`std::size_t GameList::getIndex ( ) const [noexcept]`  
Gets the index of the current lib.

## Returns

`std::size_t` Index of Library

### 10.20.2.3 getLibraryList()

`std::vector< std::shared_ptr< shared::games::IGame > > GameList::getLibraryList ( )`  
Return the entire game list.

## Returns

`std::vector<shared::graphics::IGraphicsProvider*>`

### 10.20.2.4 getNbGame()

`std::size_t GameList::getNbGame ( ) const`  
Get the Nb Game object.

## Returns

`std::size_t` Number of Game Library

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

- `src/core/library/GameList.hpp`
- `src/core/library/GameList.cpp`

## 10.21 shared::games::GameManifest Struct Reference

Game manifest.

```
#include <GameManifest.hpp>
```

## Public Attributes

- `const std::string name`  
*Name of the game.*
- `const std::string description`  
*Description of the game.*
- `const std::string version`  
*Version of the game.*
- `const std::vector< Author > authors`  
*List of authors of the game.*
- `const std::string iconPath`  
*Path of the icon game.*

### 10.21.1 Detailed Description

Game manifest.

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

- `common/games/types/GameManifest.hpp`

## 10.22 GraphicList Class Reference

### Public Member Functions

- `GraphicList (std::vector< std::string > allLibrariesPath, std::string defaultLib)`  
*Construct a new Graphic List object.*
- `~GraphicList ()`  
*Destroy the Graphic List object.*
- `std::size_t getNbGraphic () const`  
*Get the Nb Graphic object.*
- `void incrementIndex ()`  
*Incremente index of Library.*
- `void decrementIndex ()`  
*Decremente index of Library.*
- `void setIndex (std::size_t newIndex)`  
*Set new index of Library.*
- `shared::graphics::IGraphicsProvider * getCurrentLibrary ()`  
*Get the Current Library object.*
- `std::size_t getIndex () const noexcept`  
*Gets the index of the current lib.*
- `std::vector< shared::graphics::IGraphicsProvider * > getLibraryList ()`  
*Get the Library List object.*

### Protected Attributes

- `std::vector< shared::graphics::IGraphicsProvider * > _libraryList`
- `std::vector< std::shared_ptr< DLLoader< shared::graphics::IGraphicsProvider * > > > _libraryLoader`
- `std::size_t _index`

### 10.22.1 Constructor & Destructor Documentation

### 10.22.1.1 GraphicList()

```
GraphicList::GraphicList (
    std::vector< std::string > allLibrariesPath,
    std::string defaultLib ) [explicit]
```

Construct a new Graphic List object.

#### Parameters

<i>allLibrariesPath</i>	All Libraries path
-------------------------	--------------------

## 10.22.2 Member Function Documentation

### 10.22.2.1 getCurrentLibrary()

```
shared::graphics::IGraphicsProvider * GraphicList::getCurrentLibrary ( )
```

Get the Current Library object.

#### Returns

std::shared\_ptr<shared::graphics::IGraphicsProvider> Current Library

### 10.22.2.2 getIndex()

```
std::size_t GraphicList::getIndex ( ) const [noexcept]
```

Gets the index of the current lib.

#### Returns

std::size\_t Index of Library

### 10.22.2.3 getLibraryList()

```
std::vector< shared::graphics::IGraphicsProvider * > GraphicList::getLibraryList ( )
```

Get the Library List object.

#### Returns

std::vector<shared::graphics::IGraphicsProvider \*>

### 10.22.2.4 getNbGraphic()

```
std::size_t GraphicList::getNbGraphic ( ) const
```

Get the Nb Graphic object.

#### Returns

std::size\_t Number of Graphic Library

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

- src/core/library/GraphicList.hpp
- src/core/library/GraphicList.cpp

## 10.23 shared::graphics::GraphicsManifest Struct Reference

Graphics library manifest.

```
#include <GraphicsManifest.hpp>
```

### Public Attributes

- const std::string [name](#)  
*Name of the graphics library.*
- const std::string [description](#)  
*Description of the graphics library.*
- const std::string [version](#)  
*Version of the graphics library.*
- const std::vector< [Author](#) > [authors](#)  
*List of authors of the graphics library.*

### 10.23.1 Detailed Description

Graphics library manifest.

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

- common/graphics/types/GraphicsManifest.hpp

## 10.24 shared::games::components::ICollidableComponent Class Reference

Interface of a collidable component.

```
#include <ICollidableComponent.hpp>
```

Inheritance diagram for shared::games::components::ICollidableComponent:

Collaboration diagram for shared::games::components::ICollidableComponent:

### Public Member Functions

- virtual void [onCollide](#) (std::shared\_ptr< [IGame](#) > ctx, std::shared\_ptr< [ICollidableComponent](#) > target)=0  
*On collide event handler for the component.*

### 10.24.1 Detailed Description

Interface of a collidable component.

### 10.24.2 Member Function Documentation

#### 10.24.2.1 onCollide()

```
virtual void shared::games::components::ICollidableComponent::onCollide (
    std::shared_ptr< IGame > ctx,
    std::shared_ptr< ICollidableComponent > target ) [pure virtual]
```

On collide event handler for the component.

#### Parameters

<i>ctx</i>	Context of the game
<i>target</i>	Target entity

Implemented in [SnakeTailCollidable](#), [SnakeHeadCollidable](#), [SnakeBodyCollidable](#), and [AppleCollidable](#).  
The documentation for this class was generated from the following file:

- common/games/components/ICollidableComponent.hpp

## 10.25 shared::games::components::IComponent Class Reference

Interface of a component.

```
#include <IComponent.hpp>
```

Inheritance diagram for shared::games::components::IComponent:

### Public Member Functions

- virtual const ComponentType [getType](#) () const noexcept=0  
*Get the type of the component.*
- virtual const [entity::IEntity](#) & [getEntity](#) () noexcept=0  
*Get the parent entity of the component.*

### 10.25.1 Detailed Description

Interface of a component.

### 10.25.2 Member Function Documentation

#### 10.25.2.1 getEntity()

```
virtual const entity::IEntity& shared::games::components::IComponent::getEntity ( ) [pure virtual], [noexcept]
```

Get the parent entity of the component.

#### Returns

Entity of the component

Implemented in [ASoundComponent](#), [ScoreTextDisplayable](#), [SnakeTailDisplayable](#), [SnakeTailCollidable](#), [SnakeHeadKeyboard](#), [SnakeHeadDisplayable](#), [SnakeHeadCollidable](#), [SnakeBodyDisplayable](#), [SnakeBodyCollidable](#), [BackgroundDisplayable](#), [AppleDisplayable](#), [AppleCollidable](#), and [AComponent](#).

#### 10.25.2.2 getType()

```
virtual const ComponentType shared::games::components::IComponent::getType ( ) const [pure virtual], [noexcept]
```

Get the type of the component.

#### Returns

Type of the component

Implemented in [ASoundComponent](#), [ScoreTextDisplayable](#), [SnakeTailDisplayable](#), [SnakeTailCollidable](#), [SnakeHeadKeyboard](#), [SnakeHeadDisplayable](#), [SnakeHeadCollidable](#), [SnakeBodyDisplayable](#), [SnakeBodyCollidable](#), [BackgroundDisplayable](#), [AppleDisplayable](#), [AppleCollidable](#), and [AComponent](#).

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

- common/games/components/IComponent.hpp

## 10.26 shared::games::components::IDisplayableComponent Class Reference

Interface of a displayable component.

`#include <IDisplayableComponent.hpp>`

Inheritance diagram for shared::games::components::IDisplayableComponent:

Collaboration diagram for shared::games::components::IDisplayableComponent:

### Public Member Functions

- virtual unsigned int & [getZIndex](#) () noexcept=0  
*Get Z index that is usefull for display priority.*
- virtual void [onMousePress](#) (std::shared\_ptr< [IGame](#) > ctx)=0  
*On click event handler for the entity.*
- virtual void [onMouseRelease](#) (std::shared\_ptr< [IGame](#) > ctx)=0  
*On release event handler for the entity.*
- virtual void [onMouseHover](#) (std::shared\_ptr< [IGame](#) > ctx)=0  
*On hover event handler for the entity.*

### 10.26.1 Detailed Description

Interface of a displayable component.

### 10.26.2 Member Function Documentation

#### 10.26.2.1 getZIndex()

```
virtual unsigned int& shared::games::components::IDisplayableComponent::getZIndex ( ) [pure virtual], [noexcept]
```

Get Z index that is usefull for display priority.

#### Returns

Z index of the entity

Implemented in [SnakeTailDisplayable](#), [SnakeHeadDisplayable](#), [SnakeBodyDisplayable](#), [BackgroundDisplayable](#), [AppleDisplayable](#), [ScoreTextDisplayable](#), and [ADisplayableComponent](#).

#### 10.26.2.2 onMouseHover()

```
virtual void shared::games::components::IDisplayableComponent::onMouseHover (
    std::shared_ptr< IGame > ctx ) [pure virtual]
```

On hover event handler for the entity.

#### Parameters

<code>ctx</code>	Context of the game
------------------	---------------------

Implemented in [ScoreTextDisplayable](#), [BackgroundDisplayable](#), [AppleDisplayable](#), [SnakeTailDisplayable](#), [SnakeHeadDisplayable](#), [SnakeBodyDisplayable](#), [TextureComponent](#), and [TextComponent](#).

#### 10.26.2.3 onMousePress()

```
virtual void shared::games::components::IDisplayableComponent::onMousePress (
```



```
std::shared_ptr< IGame > ctx ) [pure virtual]
```

On click event handler for the entity.

#### Parameters

<code>ctx</code>	Context of the game
------------------	---------------------

Implemented in [ScoreTextDisplayable](#), [BackgroundDisplayable](#), [AppleDisplayable](#), [SnakeTailDisplayable](#), [SnakeHeadDisplayable](#), [SnakeBodyDisplayable](#), [TextureComponent](#), and [TextComponent](#).

#### 10.26.2.4 onMouseRelease()

```
virtual void shared::games::components::IDisplayableComponent::onMouseRelease (
    std::shared_ptr< IGame > ctx ) [pure virtual]
```

On release event handler for the entity.

#### Parameters

<code>ctx</code>	Context of the game
------------------	---------------------

Implemented in [ScoreTextDisplayable](#), [BackgroundDisplayable](#), [AppleDisplayable](#), [SnakeTailDisplayable](#), [SnakeHeadDisplayable](#), [SnakeBodyDisplayable](#), [TextureComponent](#), and [TextComponent](#).

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

- common/games/components/IDisplayableComponent.hpp

## 10.27 shared::games::entity::IEntity Class Reference

Interface of an entity.

```
#include <IEntity.hpp>
```

Inheritance diagram for shared::games::entity::IEntity:

### Public Member Functions

- virtual const components::ComponentsMap & [getComponents](#) (void) const noexcept=0  
*Get the components of the entity.*

#### 10.27.1 Detailed Description

Interface of an entity.

#### 10.27.2 Member Function Documentation

##### 10.27.2.1 getComponents()

```
virtual const components::ComponentsMap& shared::games::entity::IEntity::getComponents (
    void ) const [pure virtual], [noexcept]
```

Get the components of the entity.

#### Returns

Components of the entity

Implemented in [ScoreTextEntity](#), [SnakeTailEntity](#), [SnakeHeadEntity](#), [SnakeBodyEntity](#), [BackgroundEntity](#), [AppleEntity](#), and [AEntity](#).

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

- common/games/IEntity.hpp

## 10.28 shared::graphics::events::IEvent Class Reference

Interface for the event object.

```
#include <IEvent.hpp>
```

Inheritance diagram for shared::graphics::events::IEvent:

### Public Member Functions

- virtual EventType [getType](#) () const noexcept=0  
*Event type.*

#### 10.28.1 Detailed Description

Interface for the event object.

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

- common/graphics/events/IEvent.hpp

## 10.29 shared::graphics::IFont Class Reference

Interface of a font.

```
#include <IFont.hpp>
```

Inheritance diagram for shared::graphics::IFont:

#### 10.29.1 Detailed Description

Interface of a font.

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

- common/graphics/IFont.hpp

## 10.30 shared::graphics::exceptions::IFontException Class Reference

Interface for the font exception object.

```
#include <IFontException.hpp>
```

Inheritance diagram for shared::graphics::exceptions::IFontException:

Collaboration diagram for shared::graphics::exceptions::IFontException:

### Additional Inherited Members

#### 10.30.1 Detailed Description

Interface for the font exception object.

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

- common/graphics/exceptions/IFontException.hpp

## 10.31 shared::games::IGame Class Reference

Interface of a game.

```
#include <IGame.hpp>
```

Inheritance diagram for shared::games::IGame:

## Public Member Functions

- virtual void [compute](#) (DeltaTime dt)=0  
*Compute the game each tick of the program.*
- virtual const [GameManifest](#) & [getManifest](#) () const noexcept=0  
*Manifest with informations of the game.*
- virtual const [Vector2u](#) [getSize](#) () const noexcept=0  
*Number of tiles that represent the game Tile size is managed by the renderer.*
- virtual const unsigned int [getFps](#) () const noexcept=0  
*Get fps of the game.*
- virtual const entity::EntitiesMap & [getEntities](#) (void) const =0  
*Get map of entities.*
- virtual const int [getScore](#) () const noexcept=0  
*Get the score of the game.*

### 10.31.1 Detailed Description

Interface of a game.

### 10.31.2 Member Function Documentation

#### 10.31.2.1 [compute\(\)](#)

```
virtual void shared::games::IGame::compute (
    DeltaTime dt ) [pure virtual]
```

Compute the game each tick of the program.

##### Parameters

<i>dt</i>	Time since last tick (Time in milliseconds)
-----------	---

Implemented in [SolarFoxGame](#), and [SnakeGame](#).

#### 10.31.2.2 [getEntities\(\)](#)

```
virtual const entity::EntitiesMap& shared::games::IGame::getEntities (
    void ) const [pure virtual]
```

Get map of entities.

##### Returns

Entities map of the game

Implemented in [SolarFoxGame](#), and [SnakeGame](#).

#### 10.31.2.3 [getFps\(\)](#)

```
virtual const unsigned int shared::games::IGame::getFps ( ) const [pure virtual], [noexcept]
```

Get fps of the game.

##### Returns

The number of frame per seconds of the game

Implemented in [SnakeGame](#), and [SolarFoxGame](#).

#### 10.31.2.4 getManifest()

`virtual const GameManifest& shared::games::IGame::getManifest ( ) const [pure virtual], [noexcept]`  
 Manifest with informations of the game.

##### Returns

Manifest of the game

Implemented in [SolarFoxGame](#), and [SnakeGame](#).

#### 10.31.2.5 getScore()

`virtual const int shared::games::IGame::getScore ( ) const [pure virtual], [noexcept]`  
 Get the score of the game.

##### Returns

The score of the game

Implemented in [SnakeGame](#), and [SolarFoxGame](#).

#### 10.31.2.6 getSize()

`virtual const Vector2u shared::games::IGame::getSize ( ) const [pure virtual], [noexcept]`  
 Number of tiles that represent the game Tile size is managed by the renderer.

##### Returns

The number of tiles of the game

Implemented in [SolarFoxGame](#), and [SnakeGame](#).

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

- `common/games/IGame.hpp`

## 10.32 shared::games::IGameProvider Class Reference

Interface of a game provider.

`#include <IGameProvider.hpp>`

Inheritance diagram for `shared::games::IGameProvider`:

### Public Member Functions

- `virtual const GameManifest & getManifest ( ) const noexcept=0`  
*Provides the game manifest.*
- `virtual std::shared_ptr< shared::games::IGame > createInstance (void)=0`  
*Provides a new instance of the game.*

#### 10.32.1 Detailed Description

Interface of a game provider.

#### 10.32.2 Member Function Documentation

### 10.32.2.1 createInstance()

```
virtual std::shared_ptr<shared::games::IGame> shared::games::IGameProvider::createInstance (
    void ) [pure virtual]
```

Provides a new instance of the game.

#### Returns

Created game instance

Implemented in [SolarFoxProvider](#), and [SnakeGameProvider](#).

### 10.32.2.2 getManifest()

```
virtual const GameManifest& shared::games::IGameProvider::getManifest ( ) const [pure virtual],
[noexcept]
```

Provides the game manifest.

#### Returns

Manifest of current game

Implemented in [SolarFoxProvider](#), and [SnakeGameProvider](#).

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

- common/games/IGameProvider.hpp

## 10.33 shared::graphics::exceptions::IGraphicsException Class Reference

Interface for the graphics exception object.

```
#include <IGraphicsException.hpp>
```

Inheritance diagram for shared::graphics::exceptions::IGraphicsException:

Collaboration diagram for shared::graphics::exceptions::IGraphicsException:

### Public Member Functions

- virtual const char \* [where](#) () const noexcept=0

*Get error location.*

### 10.33.1 Detailed Description

Interface for the graphics exception object.

### 10.33.2 Member Function Documentation

#### 10.33.2.1 where()

```
virtual const char* shared::graphics::exceptions::IGraphicsException::where ( ) const [pure
virtual], [noexcept]
```

Get error location.

#### Returns

String containing error location

Implemented in [SFMLWindowException](#), [SFMLTextureException](#), [SFMSoundException](#), [SFMLFontException](#), [SDL2WindowException](#), [SDL2TextureException](#), [SDL2SoundException](#), [SDL2FontException](#), [NcursesWindowException](#), and [NcursesTextureException](#).

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

- common/graphics/exceptions/IGraphicsException.hpp

## 10.34 shared::graphics::IGraphicsProvider Class Reference

Interface for the graphics provider.

```
#include <IGraphicsProvider.hpp>
```

Inheritance diagram for shared::graphics::IGraphicsProvider:

### Public Member Functions

- virtual const [GraphicsManifest](#) & [getManifest](#) (void) const noexcept=0  
*Get the manifest of the graphics library.*
- virtual std::unique\_ptr< [IWindow](#) > [createWindow](#) (const [IWindow::WindowInitProps](#) &windowProps)=0  
*Create a new window object.*
- virtual std::shared\_ptr< [ISound](#) > [createSound](#) (const std::string &path)=0  
*Create a sound object.*
- virtual std::shared\_ptr< [ITexture](#) > [createTexture](#) (const std::string &bin, const std::string &ascii)=0  
*Create a texture object.*
- virtual std::shared\_ptr< [IFont](#) > [createFont](#) (const std::string &path)=0  
*Create a font object.*

### 10.34.1 Detailed Description

Interface for the graphics provider.

### 10.34.2 Member Function Documentation

#### 10.34.2.1 createFont()

```
virtual std::shared_ptr<IFont> shared::graphics::IGraphicsProvider::createFont (
    const std::string & path ) [pure virtual]
```

Create a font object.

##### Parameters

<i>path</i>	Path of the font file
-------------	-----------------------

##### Returns

Created font object

Implemented in [SFML](#), [SDL2](#), and [Ncurses](#).

#### 10.34.2.2 createSound()

```
virtual std::shared_ptr<ISound> shared::graphics::IGraphicsProvider::createSound (
    const std::string & path ) [pure virtual]
```

Create a sound object.

##### Parameters

<i>path</i>	Path of the sound file
-------------	------------------------

**Returns**

Created sound object

Implemented in [SFML](#), [SDL2](#), and [Ncurses](#).

**10.34.2.3 createTexture()**

```
virtual std::shared_ptr<ITexture> shared::graphics::IGraphicsProvider::createTexture (
    const std::string & bin,
    const std::string & ascii ) [pure virtual]
```

Create a texture object.

**Parameters**

<i>bin</i>	Path of the binary texture file
<i>ascii</i>	Path of the ascii texture file

**Returns**

Created texture object

Implemented in [Ncurses](#), [SFML](#), and [SDL2](#).

**10.34.2.4 createWindow()**

```
virtual std::unique_ptr<IWindow> shared::graphics::IGraphicsProvider::createWindow (
    const IWindow::WindowInitProps & windowProps ) [pure virtual]
```

Create a new window object.

**Parameters**

<i>windowProps</i>	Properties to use to init the window
--------------------	--------------------------------------

**Returns**

Created window object

Implemented in [SFML](#), [SDL2](#), and [Ncurses](#).

**10.34.2.5 getManifest()**

```
virtual const GraphicsManifest& shared::graphics::IGraphicsProvider::getManifest (
    void ) const [pure virtual], [noexcept]
```

Get the manifest of the graphics library.

**Returns**

Manifest of the graphics library

Implemented in [SFML](#), [SDL2](#), and [Ncurses](#).

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

- common/graphics/IGraphicsProvider.hpp

## 10.35 shared::games::components::IKeyboardComponent Class Reference

Interface of a keyboard component.

```
#include <IKeyboardComponent.hpp>
```

Inheritance diagram for `shared::games::components::IKeyboardComponent`:

Collaboration diagram for `shared::games::components::IKeyboardComponent`:

## Classes

- union [KeyCode](#)  
*Function key code union.*
- struct [KeyData](#)  
*Key data.*

## Public Types

- enum [KeyType](#) {  
  [CONTROL](#) , [ARROW](#) , [FUNC](#) , [CHAR](#) ,  
  [UNKNOWN](#) }
  - enum [ControlCode](#) { [CTRL](#) , [SHIFT](#) , [ALT](#) }
  - enum [ArrowCode](#) { [UP](#) , [DOWN](#) , [LEFT](#) , [RIGHT](#) }
- Type of the key.*
- Control key code.*
- Arrow key code.*

## Public Member Functions

- virtual void [onKeyPress](#) (std::shared\_ptr< [IGame](#) > ctx, [KeyData](#) keyData)=0  
*On key pressed event handler for the entity.*
- virtual void [onKeyRelease](#) (std::shared\_ptr< [IGame](#) > ctx, [KeyData](#) keyData)=0  
*On key release event handler for the entity.*

### 10.35.1 Detailed Description

Interface of a keyboard component.

### 10.35.2 Member Enumeration Documentation

#### 10.35.2.1 ArrowCode

enum [shared::games::components::IKeyboardComponent::ArrowCode](#)  
Arrow key code.

##### Enumerator

UP	Up arrow key
DOWN	Down arrow key
LEFT	Left arrow key
RIGHT	Right arrow key

#### 10.35.2.2 ControlCode

enum [shared::games::components::IKeyboardComponent::ControlCode](#)  
Control key code.



## Enumerator

CTRL	Ctrl key
SHIFT	Shift key
ALT	Alt key

## 10.35.2.3 KeyType

enum [shared::games::components::IKeyboardComponent::KeyType](#)

Type of the key.

## Enumerator

CONTROL	Control key (Ctrl, Shift, Alt)
ARROW	Arrow key (Up, Down, Left, Right)
FUNC	Function key (F1, F2, F3, etc.)
CHAR	Character key (a, 1, &, etc.)
UNKNOWN	Unknown key.

## 10.35.3 Member Function Documentation

## 10.35.3.1 onKeyPress()

```
virtual void shared::games::components::IKeyboardComponent::onKeyPress (
    std::shared_ptr< IGame > ctx,
    KeyData keyData ) [pure virtual]
```

On key pressed event handler for the entity.

## Parameters

<i>ctx</i>	Context of the game
<i>keyData</i>	Key data of key pressed

Implemented in [SnakeHeadKeyboard](#), and [SolarFoxPlayerKeyboard](#).

## 10.35.3.2 onKeyRelease()

```
virtual void shared::games::components::IKeyboardComponent::onKeyRelease (
    std::shared_ptr< IGame > ctx,
    KeyData keyData ) [pure virtual]
```

On key release event handler for the entity.

## Parameters

<i>ctx</i>	Context of the game
<i>keyData</i>	Key data of key released

Implemented in [SnakeHeadKeyboard](#), and [SolarFoxPlayerKeyboard](#).

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

- `common/games/components/IKeyboardComponent.hpp`

## 10.36 `shared::graphics::events::IKeyEvent` Class Reference

Interface for the key event object.

```
#include <IKeyEvent.hpp>
```

Inheritance diagram for `shared::graphics::events::IKeyEvent`:

Collaboration diagram for `shared::graphics::events::IKeyEvent`:

### Classes

- union `KeyCode`

*Key code.*

### Public Types

- enum `KeyType` { `CONTROL` , `ARROW` , `FUNC` , `CHAR` , `UNKNOWN` }
- Key type.*
- enum `ControlCode` { `CTRL` , `SHIFT` , `ALT` }
- Control key code.*
- enum `ArrowCode` { `UP` , `DOWN` , `LEFT` , `RIGHT` }
- Arrow key code.*

### Public Member Functions

- virtual const `KeyCode` `getKeyCode` (void) const noexcept=0
- Key code content.*
- virtual const `KeyType` `getKeyType` (void) const noexcept=0
- Key type.*

#### 10.36.1 Detailed Description

Interface for the key event object.

#### 10.36.2 Member Enumeration Documentation

##### 10.36.2.1 `ArrowCode`

```
enum shared::graphics::events::IKeyEvent::ArrowCode
```

Arrow key code.

Enumerator

UP	UP arrow key.
DOWN	DOWN arrow key.
LEFT	LEFT arrow key.
RIGHT	RIGHT arrow key.

### 10.36.2.2 ControlCode

enum [shared::graphics::events::IKeyEvent::ControlCode](#)  
Control key code.

#### Enumerator

CTRL	CTRL key.
SHIFT	SHIFT key.
ALT	ALT key.

### 10.36.2.3 KeyType

enum [shared::graphics::events::IKeyEvent::KeyType](#)  
Key type.

#### Enumerator

CONTROL	Control key (Ctrl, Shift, Alt)
ARROW	Arrow key (Up, Down, Left, Right)
FUNC	Function key (F1, F2, F3, etc.)
CHAR	ASCII character key.
UNKNOWN	Unknown key.

## 10.36.3 Member Function Documentation

### 10.36.3.1 getKeyCode()

```
virtual const KeyCode shared::graphics::events::IKeyEvent::getKeyCode (
    void ) const [pure virtual], [noexcept]
```

Key code content.

#### Returns

Content of the key code

Implemented in [shared::graphics::events::KeyReleaseEvent](#), and [shared::graphics::events::KeyPressedEvent](#).

### 10.36.3.2 getKeyType()

```
virtual const KeyType shared::graphics::events::IKeyEvent::getKeyType (
    void ) const [pure virtual], [noexcept]
```

Key type.

#### Returns

Type of the key pressed

Implemented in [shared::graphics::events::KeyReleaseEvent](#), and [shared::graphics::events::KeyPressedEvent](#).  
The documentation for this class was generated from the following file:

- common/graphics/events/IKeyEvent.hpp

## 10.37 shared::graphics::events::IMouseButtonEvent Class Reference

Interface for the mouse button event object.

```
#include <IMouseButtonEvent.hpp>
```

Inheritance diagram for shared::graphics::events::IMouseButtonEvent:

Collaboration diagram for shared::graphics::events::IMouseButtonEvent:

### Public Types

- enum [MouseButton](#) { LEFT , RIGHT }  
*Mouse button.*

### Public Member Functions

- virtual const [MouseButton](#) [getButton](#) (void) const noexcept=0  
*Mouse button released.*

#### 10.37.1 Detailed Description

Interface for the mouse button event object.

#### 10.37.2 Member Function Documentation

##### 10.37.2.1 getButton()

```
virtual const MouseButton shared::graphics::events::IMouseButtonEvent::getButton (
    void ) const [pure virtual], [noexcept]
```

Mouse button released.

##### Returns

Button released or pressed

Implemented in [shared::graphics::events::MouseButtonReleaseEvent](#), and [shared::graphics::events::MouseButtonPressEvent](#).  
The documentation for this class was generated from the following file:

- common/graphics/events/IMouseButtonEvent.hpp

## 10.38 shared::graphics::events::IMouseEvent Class Reference

Interface for the mouse event object.

```
#include <IMouseEvent.hpp>
```

Inheritance diagram for shared::graphics::events::IMouseEvent:

Collaboration diagram for shared::graphics::events::IMouseEvent:

### Public Member Functions

- virtual const [shared::types::Vector2f](#) [getPosition](#) (void) const noexcept=0  
*Mouse position.*

#### 10.38.1 Detailed Description

Interface for the mouse event object.

#### 10.38.2 Member Function Documentation

### 10.38.2.1 getPosition()

```
virtual const shared::types::Vector2f shared::graphics::events::IMouseEvent::getPosition (
    void ) const [pure virtual], [noexcept]
```

Mouse position.

#### Returns

Position of the mouse

Implemented in [shared::graphics::events::MouseMoveEvent](#), [shared::graphics::events::MouseButtonReleaseEvent](#), and [shared::graphics::events::MouseButtonPressEvent](#).

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

- common/graphics/events/IMouseEvent.hpp

## 10.39 shared::games::components::IPositionableComponent Class Reference

Interface of a positionable component.

```
#include <IPositionableComponent.hpp>
```

Inheritance diagram for shared::games::components::IPositionableComponent:

Collaboration diagram for shared::games::components::IPositionableComponent:

### Public Member Functions

- virtual [types::Vector2f](#) & [getPosition](#) () noexcept=0  
*Get position of the entity (tiles)*
- virtual [types::Vector2u](#) & [getSize](#) () noexcept=0  
*Get size of the entity (tiles)*

### 10.39.1 Detailed Description

Interface of a positionable component.

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

- common/games/components/IPositionableComponent.hpp

## 10.40 shared::graphics::ISound Class Reference

Interface for the sound object.

```
#include <ISound.hpp>
```

Inheritance diagram for shared::graphics::ISound:

### Public Types

- enum [SoundState](#) { [PLAY](#) , [PAUSE](#) , [STOP](#) }  
*Sound state.*
- typedef unsigned char [SoundVolume](#)  
*Sound volume.*

### Public Member Functions

- virtual void [setState](#) ([SoundState](#) state)=0  
*Get the state of the sound.*
- virtual [SoundState](#) [getState](#) () const =0  
*Get the state of the sound.*

- virtual void [setVolume](#) ([SoundVolume](#) volume)=0  
*Set the volume of the sound.*
- virtual [SoundVolume](#) [getVolume](#) () const =0  
*Get the volume of the sound.*
- virtual void [setLoopState](#) (bool loop)=0  
*Set the loop state of sound.*
- virtual bool [getLoopState](#) (void) const =0  
*Get the loop state of sound.*

### 10.40.1 Detailed Description

Interface for the sound object.

### 10.40.2 Member Enumeration Documentation

#### 10.40.2.1 SoundState

enum [shared::graphics::ISound::SoundState](#)  
Sound state.

Enumerator

PLAY	Sound is playing.
PAUSE	Sound is paused.
STOP	Sound is stopped.

### 10.40.3 Member Function Documentation

#### 10.40.3.1 getLoopState()

```
virtual bool shared::graphics::ISound::getLoopState (
    void ) const [pure virtual]
```

Get the loop state of sound.

Returns

Loop state of sound

Implemented in [SFMLSound](#), [SDL2Sound](#), and [NcursesSound](#).

#### 10.40.3.2 getState()

```
virtual SoundState shared::graphics::ISound::getState ( ) const [pure virtual]
```

Get the state of the sound.

Returns

Current state of the sound

Implemented in [SFMLSound](#), [SDL2Sound](#), and [NcursesSound](#).

### 10.40.3.3 getVolume()

```
virtual SoundVolume shared::graphics::ISound::getVolume ( ) const [pure virtual]
```

Get the volume of the sound.

#### Returns

Volume of the sound

Implemented in [SFMLSound](#), [SDL2Sound](#), and [NcursesSound](#).

### 10.40.3.4 setLoopState()

```
virtual void shared::graphics::ISound::setLoopState (
    bool loop ) [pure virtual]
```

Set the loop state of sound.

#### Parameters

<i>loop</i>	Loop state of sound
-------------	---------------------

Implemented in [SFMLSound](#), [SDL2Sound](#), and [NcursesSound](#).

### 10.40.3.5 setState()

```
virtual void shared::graphics::ISound::setState (
    SoundState state ) [pure virtual]
```

Get the state of the sound.

#### Parameters

<i>state</i>	State of sound playing
--------------	------------------------

Implemented in [SFMLSound](#), [SDL2Sound](#), and [NcursesSound](#).

### 10.40.3.6 setVolume()

```
virtual void shared::graphics::ISound::setVolume (
    SoundVolume volume ) [pure virtual]
```

Set the volume of the sound.

#### Parameters

<i>volume</i>	Volume of the sound
---------------	---------------------

Implemented in [SFMLSound](#), [SDL2Sound](#), and [NcursesSound](#).

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

- common/graphics/ISound.hpp

## 10.41 shared::games::components::ISoundComponent Class Reference

Interface of a sound component.

```
#include <ISoundComponent.hpp>
```

Inheritance diagram for shared::games::components::ISoundComponent:

Collaboration diagram for shared::games::components::ISoundComponent:

## Public Member Functions

- virtual const std::string & [getPath](#) () const noexcept=0  
*Get path of the sound.*
- virtual SoundState & [getState](#) () noexcept=0  
*Get state of the sound.*
- virtual SoundVolume & [getVolume](#) () noexcept=0  
*Get volume of the sound.*
- virtual bool & [getLoop](#) () noexcept=0  
*Get loop of the sound.*
- virtual void [onStateChange](#) (std::shared\_ptr< [IGame](#) > ctx, SoundState state)=0  
*On state change event handler for the component.*

### 10.41.1 Detailed Description

Interface of a sound component.

### 10.41.2 Member Function Documentation

#### 10.41.2.1 [getLoop\(\)](#)

```
virtual bool& shared::games::components::ISoundComponent::getLoop ( ) [pure virtual], [noexcept]
```

Get loop of the sound.

##### Returns

Sound loop

Implemented in [ASoundComponent](#).

#### 10.41.2.2 [getPath\(\)](#)

```
virtual const std::string& shared::games::components::ISoundComponent::getPath ( ) const [pure virtual], [noexcept]
```

Get path of the sound.

##### Returns

Sound path

Implemented in [ASoundComponent](#).

#### 10.41.2.3 [getState\(\)](#)

```
virtual SoundState& shared::games::components::ISoundComponent::getState ( ) [pure virtual], [noexcept]
```

Get state of the sound.

##### Returns

Sound state

Implemented in [ASoundComponent](#).



### 10.41.2.4 getVolume()

```
virtual SoundVolume& shared::games::components::ISoundComponent::getVolume ( ) [pure virtual],
[noexcept]
```

Get volume of the sound.

#### Returns

Sound volume

Implemented in [ASoundComponent](#).

### 10.41.2.5 onStateChange()

```
virtual void shared::games::components::ISoundComponent::onStateChange (
    std::shared_ptr< IGame > ctx,
    SoundState state ) [pure virtual]
```

On state change event handler for the component.

#### Parameters

<i>ctx</i>	Context of the game
<i>state</i>	New state of the sound

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

- common/games/components/ISoundComponent.hpp

## 10.42 shared::graphics::exceptions::ISoundException Class Reference

Interface for the sound exception object.

```
#include <ISoundException.hpp>
```

Inheritance diagram for shared::graphics::exceptions::ISoundException:

Collaboration diagram for shared::graphics::exceptions::ISoundException:

### Additional Inherited Members

#### 10.42.1 Detailed Description

Interface for the sound exception object.

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

- common/graphics/exceptions/ISoundException.hpp

## 10.43 shared::games::components::ITextComponent Class Reference

Interface of a text component.

```
#include <ITextComponent.hpp>
```

Inheritance diagram for shared::games::components::ITextComponent:

Collaboration diagram for shared::games::components::ITextComponent:

### Classes

- struct [TextFontProps](#)  
*Font properties.*
- struct [TextProps](#)  
*Text properties.*

## Public Types

- enum `TextAlign` { `LEFT` , `CENTER` , `RIGHT` }  
*Horizontal text alignment.*
- enum `TextVerticalAlign` { `BOTTOM` , `MIDDLE` , `TOP` }  
*Vertical text alignment.*

## Public Member Functions

- virtual `TextProps` `getTextProps` () noexcept=0  
*Get text props of the entity.*

### 10.43.1 Detailed Description

Interface of a text component.

### 10.43.2 Member Enumeration Documentation

#### 10.43.2.1 TextAlign

enum `shared::games::components::ITextComponent::TextAlign`  
Horizontal text alignment.

Enumerator

LEFT	Align text to the left.
CENTER	Align text to the center.
RIGHT	Align text to the right.

#### 10.43.2.2 TextVerticalAlign

enum `shared::games::components::ITextComponent::TextVerticalAlign`  
Vertical text alignment.

Enumerator

BOTTOM	Align text to the bottom.
MIDDLE	Align text to the middle.
TOP	Align text to the top.

### 10.43.3 Member Function Documentation

#### 10.43.3.1 getTextProps()

virtual `TextProps` `shared::games::components::ITextComponent::getTextProps` ( ) [pure virtual],  
[noexcept]  
Get text props of the entity.

**Returns**

text props

Implemented in [ScoreTextDisplayable](#), [TextComponent](#), and [ATextComponent](#).  
The documentation for this class was generated from the following file:

- common/games/components/ITextComponent.hpp

## 10.44 shared::graphics::ITexture Class Reference

Interface for the texture object.

```
#include <ITexture.hpp>
```

Inheritance diagram for shared::graphics::ITexture:

### 10.44.1 Detailed Description

Interface for the texture object.

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

- common/graphics/ITexture.hpp

## 10.45 shared::games::components::ITextureComponent Class Reference

Interface of a texture component.

```
#include <ITextureComponent.hpp>
```

Inheritance diagram for shared::games::components::ITextureComponent:

Collaboration diagram for shared::games::components::ITextureComponent:

### Public Member Functions

- virtual [TextureProps](#) & [getTextureProps](#) () noexcept=0  
*Get texture properties.*

### 10.45.1 Detailed Description

Interface of a texture component.

### 10.45.2 Member Function Documentation

#### 10.45.2.1 getTextureProps()

```
virtual TextureProps& shared::games::components::ITextureComponent::getTextureProps ( ) [pure virtual], [noexcept]
```

Get texture properties.

**Returns**

[TextureProps](#) &Texture properties

Implemented in [SnakeTailDisplayable](#), [SnakeHeadDisplayable](#), [SnakeBodyDisplayable](#), [BackgroundDisplayable](#), [AppleDisplayable](#), and [TextureComponent](#).

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

- common/games/components/ITextureComponent.hpp

## 10.46 shared::graphics::exceptions::ITextureException Class Reference

Interface for the texture exception object.

```
#include <ITextureException.hpp>
```

Inheritance diagram for shared::graphics::exceptions::ITextureException:

Collaboration diagram for shared::graphics::exceptions::ITextureException:

### Additional Inherited Members

#### 10.46.1 Detailed Description

Interface for the texture exception object.

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

- common/graphics/exceptions/ITextureException.hpp

## 10.47 shared::graphics::IWindow Class Reference

Interface for the window object.

```
#include <IWindow.hpp>
```

Inheritance diagram for shared::graphics::IWindow:

### Classes

- struct [WindowInitProps](#)  
*Window initial properties.*

### Public Types

- enum [WindowMode](#) { [WINDOWED](#) , [FULLSCREEN](#) }  
*Window mode.*

### Public Member Functions

- virtual void [setTitle](#) (const std::string &title)=0  
*Set the title of current window.*
- virtual void [setSize](#) ([Vector2u](#) size)=0  
*Set the size of the window.*
- virtual [Vector2u](#) [getSize](#) () const =0  
*Get the size of the window.*
- virtual void [setFramerateLimit](#) (unsigned int fps)=0  
*Set the framerate Limit of the window.*
- virtual unsigned int [getFramerateLimit](#) () const =0  
*Get the framerate Limit of the window.*
- virtual void [setMode](#) ([WindowMode](#) mode)=0  
*Set the mode of the window.*
- virtual [WindowMode](#) [getMode](#) (void) const =0  
*Get the mode of the window.*
- virtual void [setIcon](#) (const std::string &icon)=0  
*Set the icon of the window.*
- virtual void [render](#) (const [TextureProps](#) &props)=0  
*Render the texture of entity with given properties.*
- virtual void [render](#) (const [TextProps](#) &props)=0  
*Render the text of entity with given properties.*
- virtual void [clear](#) (void)=0

- Clear the content of the window.*
- virtual void [display](#) (void)=0
- Display the content of the window.*
- virtual void [close](#) (void)=0
- Close the window.*
- virtual bool [isOpen](#) (void) const =0
- Check if the window is open.*
- virtual std::vector< events::EventPtr > [getEvents](#) (void)=0
- Get the events object.*

## 10.47.1 Detailed Description

Interface for the window object.

## 10.47.2 Member Enumeration Documentation

### 10.47.2.1 WindowMode

enum [shared::graphics::IWindow::WindowMode](#)  
Window mode.

Enumerator

WINDOWED	Windowed mode.
FULLSCREEN	Fullscreen mode.

## 10.47.3 Member Function Documentation

### 10.47.3.1 getEvents()

```
virtual std::vector<events::EventPtr> shared::graphics::IWindow::getEvents (
    void ) [pure virtual]
```

Get the events object.

Returns

Last events occurred

Warning

Call successively this method will result in losing events

Note

Call A return `eventsA` containing 2 events, but make another call B (directly after call A) `eventsB` will result to an empty vector

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

### 10.47.3.2 getFramerateLimit()

```
virtual unsigned int shared::graphics::IWindow::getFramerateLimit ( ) const [pure virtual]
```

Get the framerate Limit of the window.

**Returns**

Frame per seconds

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.3 getMode()**

```
virtual WindowMode shared::graphics::IWindow::getMode (
    void ) const [pure virtual]
```

Get the mode of the window.

**Returns**

Mode of the window

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.4 getSize()**

```
virtual Vector2u shared::graphics::IWindow::getSize ( ) const [pure virtual]
```

Get the size of the window.

**Returns**

Size of the window

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.5 isOpen()**

```
virtual bool shared::graphics::IWindow::isOpen (
    void ) const [pure virtual]
```

Check if the window is open.

**Returns**

Open status of the window

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.6 render() [1/2]**

```
virtual void shared::graphics::IWindow::render (
    const TextProps & props ) [pure virtual]
```

Render the text of entity with given properties.

**Parameters**

<i>props</i>	Properties of the entity & text to render
--------------	---

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.7 render() [2/2]**

```
virtual void shared::graphics::IWindow::render (
    const TextureProps & props ) [pure virtual]
```

Render the texture of entity with given properties.

## Parameters

<i>props</i>	Properties of the entity & texture to render
--------------	--

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.8 setFramerateLimit()**

```
virtual void shared::graphics::IWindow::setFramerateLimit (
    unsigned int fps ) [pure virtual]
```

Set the framerate Limit of the window.

## Parameters

<i>fps</i>	Frame per seconds
------------	-------------------

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.9 setIcon()**

```
virtual void shared::graphics::IWindow::setIcon (
    const std::string & icon ) [pure virtual]
```

Set the icon of the window.

## Parameters

<i>icon</i>	Icon to use
-------------	-------------

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.10 setMode()**

```
virtual void shared::graphics::IWindow::setMode (
    WindowMode mode ) [pure virtual]
```

Set the mode of the window.

## Parameters

<i>mode</i>	Mode to apply to the window
-------------	-----------------------------

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

**10.47.3.11 setSize()**

```
virtual void shared::graphics::IWindow::setSize (
    Vector2u size ) [pure virtual]
```

Set the size of the window.

## Parameters

<i>size</i>	Size of the window
-------------	--------------------

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

### 10.47.3.12 setTitle()

```
virtual void shared::graphics::IWindow::setTitle (
    const std::string & title ) [pure virtual]
```

Set the title of current window.

#### Parameters

<i>title</i>	Title of the window
--------------	---------------------

Implemented in [SFMLWindow](#), [SDL2Window](#), and [NcursesWindow](#).

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

- common/graphics/IWindow.hpp

## 10.48 shared::graphics::exceptions::IWindowException Class Reference

Interface for the window exception object.

```
#include <IWindowException.hpp>
```

Inheritance diagram for shared::graphics::exceptions::IWindowException:

Collaboration diagram for shared::graphics::exceptions::IWindowException:

### Additional Inherited Members

#### 10.48.1 Detailed Description

Interface for the window exception object.

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

- common/graphics/exceptions/IWindowException.hpp

## 10.49 shared::games::components::IKeyboardComponent::KeyCode Union Reference

Function key code union.

```
#include <IKeyboardComponent.hpp>
```

### Public Attributes

- [ControlCode](#) *control*  
*Function key number.*
- [ArrowCode](#) *arrow*  
*Control key code.*
- char [character](#)  
*Arrow key code.*
- unsigned char [func](#)  
*Character key code.*

#### 10.49.1 Detailed Description

Function key code union.

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

- common/games/components/IKeyboardComponent.hpp



## 10.50 shared::graphics::events::IKeyEvent::KeyCode Union Reference

Key code.

```
#include <IKeyEvent.hpp>
```

### Public Attributes

- [ControlCode](#) control  
*Control key.*
- [ArrowCode](#) arrow  
*Arrow key.*
- char [character](#)  
*ASCII character value.*
- unsigned char [func](#)  
*Function key number.*

### 10.50.1 Detailed Description

Key code.

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

- common/graphics/events/IKeyEvent.hpp

## 10.51 shared::games::components::IKeyboardComponent::KeyData Struct Reference

Key data.

```
#include <IKeyboardComponent.hpp>
```

Collaboration diagram for shared::games::components::IKeyboardComponent::KeyData:

### Public Attributes

- [KeyCode](#) code  
*Key code. Interpretation depends on the type.*
- [KeyType](#) type  
*Type of the key.*

### 10.51.1 Detailed Description

Key data.

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

- common/games/components/IKeyboardComponent.hpp

## 10.52 shared::graphics::events::KeyPressedEvent Class Reference

Inheritance diagram for shared::graphics::events::KeyPressedEvent:

Collaboration diagram for shared::graphics::events::KeyPressedEvent:

### Public Member Functions

- **KeyPressedEvent** ([KeyType](#) keyType, [KeyCode](#) keyCode)
- const [KeyCode](#) [getKeyCode](#) (void) const noexcept override  
*Key code content.*
- const [KeyType](#) [getKeyType](#) (void) const noexcept override

*Key type.*

- EventType [getType](#) () const noexcept override

*Event type.*

## Additional Inherited Members

### 10.52.1 Member Function Documentation

#### 10.52.1.1 getKeyCode()

```
const KeyCode shared::graphics::events::KeyPressedEvent::getKeyCode (
    void ) const [inline], [override], [virtual], [noexcept]
```

Key code content.

##### Returns

Content of the key code

Implements [shared::graphics::events::IKeyEvent](#).

#### 10.52.1.2 getKeyType()

```
const KeyType shared::graphics::events::KeyPressedEvent::getKeyType (
    void ) const [inline], [override], [virtual], [noexcept]
```

Key type.

##### Returns

Type of the key pressed

Implements [shared::graphics::events::IKeyEvent](#).

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

- shared/events/key/KeyPressedEvent.hpp

## 10.53 shared::graphics::events::KeyReleaseEvent Class Reference

Inheritance diagram for shared::graphics::events::KeyReleaseEvent:

Collaboration diagram for shared::graphics::events::KeyReleaseEvent:

### Public Member Functions

- **KeyReleaseEvent** ([KeyType](#) keyType, [KeyCode](#) keyCode)
- const [KeyCode](#) [getKeyCode](#) (void) const noexcept override
 

*Key code content.*
- const [KeyType](#) [getKeyType](#) (void) const noexcept override
 

*Key type.*
- EventType [getType](#) () const noexcept override
 

*Event type.*

## Additional Inherited Members

### 10.53.1 Member Function Documentation

### 10.53.1.1 getKeyCode()

```
const KeyCode shared::graphics::events::KeyReleaseEvent::getKeyCode (
    void ) const [inline], [override], [virtual], [noexcept]
```

Key code content.

#### Returns

Content of the key code

Implements [shared::graphics::events::IKeyEvent](#).

### 10.53.1.2 getKeyType()

```
const KeyType shared::graphics::events::KeyReleaseEvent::getKeyType (
    void ) const [inline], [override], [virtual], [noexcept]
```

Key type.

#### Returns

Type of the key pressed

Implements [shared::graphics::events::IKeyEvent](#).

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

- shared/events/key/KeyReleaseEvent.hpp

## 10.54 shared::graphics::events::MouseButtonPressEvent Class Reference

Inheritance diagram for shared::graphics::events::MouseButtonPressEvent:

Collaboration diagram for shared::graphics::events::MouseButtonPressEvent:

### Public Member Functions

- **MouseButtonPressEvent** ([MouseButton](#) button, [types::Vector2f](#) position)
- const [MouseButton](#) [getButton](#) (void) const noexcept override  
*Mouse button released.*
- const [shared::types::Vector2f](#) [getPosition](#) (void) const noexcept override  
*Mouse position.*
- EventType [getType](#) () const noexcept  
*Event type.*

### Additional Inherited Members

#### 10.54.1 Member Function Documentation

##### 10.54.1.1 getButton()

```
const MouseButton shared::graphics::events::MouseButtonPressEvent::getButton (
    void ) const [inline], [override], [virtual], [noexcept]
```

Mouse button released.

#### Returns

Button released or pressed

Implements [shared::graphics::events::IMouseButtonEvent](#).

### 10.54.1.2 getPosition()

```
const shared::types::Vector2f shared::graphics::events::MouseButtonPressEvent::getPosition (
    void ) const [inline], [override], [virtual], [noexcept]
```

Mouse position.

Returns

Position of the mouse

Implements [shared::graphics::events::IMouseEvent](#).

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

- [shared/events/mouse/MouseButtonPressEvent.hpp](#)

## 10.55 shared::graphics::events::MouseButtonReleaseEvent Class Reference

Inheritance diagram for [shared::graphics::events::MouseButtonReleaseEvent](#):

Collaboration diagram for [shared::graphics::events::MouseButtonReleaseEvent](#):

### Public Member Functions

- **MouseButtonReleaseEvent** ([MouseButton](#) button, [types::Vector2f](#) position)
- const [MouseButton](#) [getButton](#) (void) const noexcept override  
*Mouse button released.*
- const [shared::types::Vector2f](#) [getPosition](#) (void) const noexcept override  
*Mouse position.*
- EventType [getType](#) () const noexcept override  
*Event type.*

### Additional Inherited Members

#### 10.55.1 Member Function Documentation

##### 10.55.1.1 getButton()

```
const MouseButton shared::graphics::events::MouseButtonReleaseEvent::getButton (
    void ) const [inline], [override], [virtual], [noexcept]
```

Mouse button released.

Returns

Button released or pressed

Implements [shared::graphics::events::IMouseButtonEvent](#).

##### 10.55.1.2 getPosition()

```
const shared::types::Vector2f shared::graphics::events::MouseButtonReleaseEvent::getPosition (
    void ) const [inline], [override], [virtual], [noexcept]
```

Mouse position.

Returns

Position of the mouse

Implements [shared::graphics::events::IMouseEvent](#).

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

- [shared/events/mouse/MouseButtonReleaseEvent.hpp](#)

## 10.56 shared::graphics::events::MouseMoveEvent Class Reference

Inheritance diagram for shared::graphics::events::MouseMoveEvent:

Collaboration diagram for shared::graphics::events::MouseMoveEvent:

### Public Member Functions

- **MouseMoveEvent** ([types::Vector2f](#) position)
- EventType [getType](#) () const noexcept override  
*Event type.*
- const [shared::types::Vector2f](#) [getPosition](#) (void) const noexcept override  
*Mouse position.*

### 10.56.1 Member Function Documentation

#### 10.56.1.1 getPosition()

```
const shared::types::Vector2f shared::graphics::events::MouseMoveEvent::getPosition (
    void ) const [inline], [override], [virtual], [noexcept]
```

Mouse position.

#### Returns

Position of the mouse

Implements [shared::graphics::events::IMouseEvent](#).

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

- shared/events/mouse/MouseMoveEvent.hpp

## 10.57 Ncurses Class Reference

Inheritance diagram for Ncurses:

Collaboration diagram for Ncurses:

### Public Member Functions

- const [shared::graphics::GraphicsManifest](#) & [getManifest](#) (void) const noexcept override  
*Get the manifest of the graphics library.*
- std::unique\_ptr< [shared::graphics::IWindow](#) > [createWindow](#) (const [shared::graphics::IWindow::WindowInitProps](#) &windowProps) override  
*Create a new window object.*
- std::shared\_ptr< [shared::graphics::ISound](#) > [createSound](#) (const std::string &path) override  
*Create a sound object.*
- std::shared\_ptr< [shared::graphics::ITexture](#) > [createTexture](#) (const std::string &path, const std::string &ascii) override  
*Create a texture object.*
- std::shared\_ptr< [IFont](#) > [createFont](#) (const std::string &path) override  
*Create a font object.*

### 10.57.1 Member Function Documentation

### 10.57.1.1 createFont()

```
std::shared_ptr< IFont > Ncurses::createFont (
    const std::string & path ) [override], [virtual]
```

Create a font object.

#### Parameters

<i>path</i>	Path of the font file
-------------	-----------------------

#### Returns

Created font object

Implements [shared::graphics::IGraphicsProvider](#).

### 10.57.1.2 createSound()

```
std::shared_ptr< shared::graphics::ISound > Ncurses::createSound (
    const std::string & path ) [override], [virtual]
```

Create a sound object.

#### Parameters

<i>path</i>	Path of the sound file
-------------	------------------------

#### Returns

Created sound object

Implements [shared::graphics::IGraphicsProvider](#).

### 10.57.1.3 createTexture()

```
std::shared_ptr< shared::graphics::ITexture > Ncurses::createTexture (
    const std::string & bin,
    const std::string & ascii ) [override], [virtual]
```

Create a texture object.

#### Parameters

<i>bin</i>	Path of the binary texture file
<i>ascii</i>	Path of the ascii texture file

#### Returns

Created texture object

Implements [shared::graphics::IGraphicsProvider](#).

### 10.57.1.4 createWindow()

```
std::unique_ptr< shared::graphics::IWindow > Ncurses::createWindow (
    const shared::graphics::IWindow::WindowInitProps & windowProps ) [override],
[virtual]
```

Create a new window object.

## Parameters

<code>windowProps</code>	Properties to use to init the window
--------------------------	--------------------------------------

## Returns

Created window object

Implements [shared::graphics::IGraphicsProvider](#).

## 10.57.1.5 getManifest()

```
const shared::graphics::GraphicsManifest & Ncurses::getManifest (
    void ) const [override], [virtual], [noexcept]
```

Get the manifest of the graphics library.

## Returns

Manifest of the graphics library

Implements [shared::graphics::IGraphicsProvider](#).

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

- `src/graphics/ncurses/Ncurses.hpp`
- `src/graphics/ncurses/Ncurses.cpp`

## 10.58 NcursesFont Class Reference

Inheritance diagram for NcursesFont:

Collaboration diagram for NcursesFont:

## Public Member Functions

- **NcursesFont** (std::string pathFont)

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

- `src/graphics/ncurses/NcursesFont.hpp`
- `src/graphics/ncurses/NcursesFont.cpp`

## 10.59 NcursesSound Class Reference

Inheritance diagram for NcursesSound:

Collaboration diagram for NcursesSound:

## Public Member Functions

- **NcursesSound** (const std::string &path)
- void **setState** ([shared::graphics::ISound::SoundState](#)) override  
*Get the state of the sound.*
- [shared::graphics::ISound::SoundState](#) **getState** () const override  
*Get the state of the sound.*
- void **setVolume** ([shared::graphics::ISound::SoundVolume](#) volume) override  
*Set the volume of the sound.*
- [shared::graphics::ISound::SoundVolume](#) **getVolume** () const override  
*Get the volume of the sound.*
- void **setLoopState** (bool loop) override  
*Set the loop state of sound.*
- bool **getLoopState** (void) const override  
*Get the loop state of sound.*

## Additional Inherited Members

### 10.59.1 Member Function Documentation

#### 10.59.1.1 getLoopState()

```
bool NcursesSound::getLoopState (
    void ) const [override], [virtual]
```

Get the loop state of sound.

##### Returns

Loop state of sound

Implements [shared::graphics::ISound](#).

#### 10.59.1.2 getState()

```
shared::graphics::ISound::SoundState NcursesSound::getState ( ) const [override], [virtual]
```

Get the state of the sound.

##### Returns

Current state of the sound

Implements [shared::graphics::ISound](#).

#### 10.59.1.3 getVolume()

```
shared::graphics::ISound::SoundVolume NcursesSound::getVolume ( ) const [override], [virtual]
```

Get the volume of the sound.

##### Returns

Volume of the sound

Implements [shared::graphics::ISound](#).

#### 10.59.1.4 setLoopState()

```
void NcursesSound::setLoopState (
    bool loop ) [override], [virtual]
```

Set the loop state of sound.

##### Parameters

<i>loop</i>	Loop state of sound
-------------	---------------------

Implements [shared::graphics::ISound](#).

#### 10.59.1.5 setState()

```
void NcursesSound::setState (
    shared::graphics::ISound::SoundState state ) [override], [virtual]
```

Get the state of the sound.



## Parameters

<i>state</i>	State of sound playing
--------------	------------------------

Implements [shared::graphics::ISound](#).

**10.59.1.6 setVolume()**

```
void NcursesSound::setVolume (
    shared::graphics::ISound::SoundVolume volume ) [override], [virtual]
```

Set the volume of the sound.

## Parameters

<i>volume</i>	Volume of the sound
---------------	---------------------

Implements [shared::graphics::ISound](#).

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

- src/graphics/ncurses/NcursesSound.hpp
- src/graphics/ncurses/NcursesSound.cpp

**10.60 NcursesTexture Class Reference**

Inheritance diagram for NcursesTexture:

Collaboration diagram for NcursesTexture:

**Public Member Functions**

- **NcursesTexture** (const std::string &asciiPath)
- const std::string & **getAscii** () const

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

- src/graphics/ncurses/NcursesTexture.hpp
- src/graphics/ncurses/NcursesTexture.cpp

**10.61 NcursesTextureException Class Reference**

Inheritance diagram for NcursesTextureException:

Collaboration diagram for NcursesTextureException:

**Public Member Functions**

- **NcursesTextureException** (const char \*[where](#), const char \*what)
- const char \* [where](#) () const noexcept override
 

*Get error location.*
- const char \* **what** () const noexcept override

**10.61.1 Member Function Documentation**

### 10.61.1.1 where()

`const char* NcursesTextureException::where ( ) const [inline], [override], [virtual], [noexcept]`  
Get error location.

#### Returns

String containing error location

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- `src/graphics/ncurses/exceptions/NcursesTextureException.hpp`

## 10.62 NcursesWindow Class Reference

Inheritance diagram for NcursesWindow:

Collaboration diagram for NcursesWindow:

### Public Member Functions

- **NcursesWindow** (const [WindowInitProps](#) &props)
- void [setTitle](#) (const std::string &title) override  
*Set the title of current window.*
- void [setSize](#) ([Vector2u](#) size) override  
*Set the size of the window.*
- [Vector2u](#) [getSize](#) () const override  
*Get the size of the window.*
- void [setFramerateLimit](#) (unsigned int fps) override  
*Set the framerate Limit of the window.*
- unsigned int [getFramerateLimit](#) () const override  
*Get the framerate Limit of the window.*
- void [setMode](#) ([shared::graphics::IWindow::WindowMode](#) mode) override  
*Set the mode of the window.*
- [shared::graphics::IWindow::WindowMode](#) [getMode](#) (void) const override  
*Get the mode of the window.*
- void [setIcon](#) (const std::string &icon) override  
*Set the icon of the window.*
- void [render](#) (const [shared::graphics::TextureProps](#) &props) override  
*Render the texture of entity with given properties.*
- void [render](#) (const [shared::graphics::TextProps](#) &props) override  
*Render the text of entity with given properties.*
- void [clear](#) (void) override  
*Clear the content of the window.*
- void [display](#) (void) override  
*Display the content of the window.*
- void [close](#) (void) override  
*Close the window.*
- bool [isOpen](#) (void) const override  
*Check if the window is open.*
- std::vector< std::shared\_ptr< [shared::graphics::events::IEvent](#) > > [getEvents](#) (void) override  
*Get the events object.*

## Static Public Member Functions

- static [events::IKeyEvent::KeyType](#) `mapNcursesKeyToKeyType` (int key)
- static [events::IKeyEvent::KeyCode](#) `mapNcursesKeyToKeyCode` (int key, [events::IKeyEvent::KeyType](#) type)

## Additional Inherited Members

### 10.62.1 Member Function Documentation

#### 10.62.1.1 `getEvents()`

```
std::vector< events::EventPtr > NcursesWindow::getEvents (
    void ) [override], [virtual]
```

Get the events object.

##### Returns

Last events occurred

##### Warning

Call successively this method will result in losing events

##### Note

Call A return `eventsA` containing 2 events, but make another call B (directly after call A) `eventsB` will result to an empty vector

Implements [shared::graphics::IWindow](#).

#### 10.62.1.2 `getFramerateLimit()`

```
unsigned int NcursesWindow::getFramerateLimit ( ) const [override], [virtual]
```

Get the framerate Limit of the window.

##### Returns

Frame per seconds

Implements [shared::graphics::IWindow](#).

#### 10.62.1.3 `getMode()`

```
shared::graphics::IWindow::WindowMode NcursesWindow::getMode (
    void ) const [override], [virtual]
```

Get the mode of the window.

##### Returns

Mode of the window

Implements [shared::graphics::IWindow](#).

#### 10.62.1.4 `getSize()`

```
Vector2u NcursesWindow::getSize ( ) const [override], [virtual]
```

Get the size of the window.

##### Returns

Size of the window

Implements [shared::graphics::IWindow](#).

**10.62.1.5 isOpen()**

```
bool NcursesWindow::isOpen (
    void ) const [override], [virtual]
```

Check if the window is open.

**Returns**

Open status of the window

Implements [shared::graphics::IWindow](#).

**10.62.1.6 render() [1/2]**

```
void NcursesWindow::render (
    const shared::graphics::TextProps & props ) [override], [virtual]
```

Render the text of entity with given properties.

**Parameters**

<i>props</i>	Properties of the entity & text to render
--------------	---

Implements [shared::graphics::IWindow](#).

**10.62.1.7 render() [2/2]**

```
void NcursesWindow::render (
    const shared::graphics::TextureProps & props ) [override], [virtual]
```

Render the texture of entity with given properties.

**Parameters**

<i>props</i>	Properties of the entity & texture to render
--------------	--

Implements [shared::graphics::IWindow](#).

**10.62.1.8 setFrameRateLimit()**

```
void NcursesWindow::setFrameRateLimit (
    unsigned int fps ) [override], [virtual]
```

Set the framerate Limit of the window.

**Parameters**

<i>fps</i>	Frame per seconds
------------	-------------------

Implements [shared::graphics::IWindow](#).

**10.62.1.9 setIcon()**

```
void NcursesWindow::setIcon (
    const std::string & icon ) [override], [virtual]
```

Set the icon of the window.

## Parameters

<i>icon</i>	Icon to use
-------------	-------------

Implements [shared::graphics::IWindow](#).

**10.62.1.10 setMode()**

```
void NcursesWindow::setMode (
    shared::graphics::IWindow::WindowMode mode ) [override], [virtual]
```

Set the mode of the window.

## Parameters

<i>mode</i>	Mode to apply to the window
-------------	-----------------------------

Implements [shared::graphics::IWindow](#).

**10.62.1.11 setSize()**

```
void NcursesWindow::setSize (
    Vector2u size ) [override], [virtual]
```

Set the size of the window.

## Parameters

<i>size</i>	Size of the window
-------------	--------------------

Implements [shared::graphics::IWindow](#).

**10.62.1.12 setTitle()**

```
void NcursesWindow::setTitle (
    const std::string & title ) [override], [virtual]
```

Set the title of current window.

## Parameters

<i>title</i>	Title of the window
--------------	---------------------

Implements [shared::graphics::IWindow](#).

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

- src/graphics/ncurses/window/NcursesWindow.hpp
- src/graphics/ncurses/window/NcursesWindow.cpp

**10.63 NcursesWindowException Class Reference**

Inheritance diagram for NcursesWindowException:

Collaboration diagram for NcursesWindowException:

**Public Member Functions**

- **NcursesWindowException** (const char \*[where](#), const char \*what)
- const char \* [where](#) () const noexcept override

*Get error location.*

- `const char * what () const noexcept override`

## 10.63.1 Member Function Documentation

### 10.63.1.1 `where()`

```
const char* NcursesWindowException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get error location.

#### Returns

String containing error location

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- `src/graphics/ncurses/exceptions/NcursesWindowException.hpp`

## 10.64 PositionableComponent Class Reference

Inheritance diagram for PositionableComponent:

Collaboration diagram for PositionableComponent:

### Public Member Functions

- **PositionableComponent** ([shared::types::Vector2f](#) position, [shared::types::Vector2u](#) size, [entity::IEntity](#) &entity, [components::ComponentType](#) type)
- [shared::types::Vector2f](#) & [getPosition](#) () noexcept override  
*Get position of the entity (tiles)*
- [shared::types::Vector2u](#) & [getSize](#) () noexcept override  
*Get size of the entity (tiles)*

### Protected Attributes

- [shared::types::Vector2f](#) **\_position**
- [shared::types::Vector2u](#) **\_size**

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

- `src/games/abstracts/components/PositionableComponent.hpp`
- `src/games/abstracts/components/PositionableComponent.cpp`

## 10.65 ScoreTextDisplayable Class Reference

Inheritance diagram for ScoreTextDisplayable:

Collaboration diagram for ScoreTextDisplayable:

### Public Member Functions

- [ScoreTextDisplayable](#) (const std::string &content, const [shared::games::entity::IEntity](#) &entity)  
*Construct a new Score Text Displayable object.*
- [~ScoreTextDisplayable](#) ()  
*Destroy the Score Text Displayable object.*
- [shared::games::components::ITextComponent::TextProps](#) [getTextProps](#) () noexcept override

- Get the Text Props object.*
- [Vector2u](#) & [getSize](#) () noexcept override
- Get the Size object.*
- unsigned int & [getZIndex](#) () noexcept override
- Get the ZIndex object.*
- void [onMousePress](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override
- Get the Text object.*
- void [onMouseRelease](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override
- Get the Text object.*
- void [onMouseHover](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx) override
- Get the Text object.*
- [Vector2f](#) & [getPosition](#) (void) noexcept override
- Get the Position object.*
- const [shared::games::components::ComponentType](#) [getType](#) () const noexcept override
- Get the Type object.*
- const [shared::games::entity::IEntity](#) & [getEntity](#) () noexcept override
- Get the entity object.*

## Public Attributes

- const std::string & [\\_text](#)
- [shared::games::components::ITextComponent::TextProps](#) [\\_props](#)

## Additional Inherited Members

### 10.65.1 Constructor & Destructor Documentation

#### 10.65.1.1 ScoreTextDisplayable()

```
ScoreTextDisplayable::ScoreTextDisplayable (
    const std::string & content,
    const shared::games::entity::IEntity & entity ) [explicit]
```

Construct a new Score Text Displayable object.

#### Parameters

<i>content</i>	
<i>entity</i>	

### 10.65.2 Member Function Documentation

#### 10.65.2.1 getEntity()

```
const shared::games::entity::IEntity & ScoreTextDisplayable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

#### Returns

const [shared::games::entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.65.2.2 getPosition()

```
shared::types::Vector2f & ScoreTextDisplayable::getPosition (
    void ) [override], [virtual], [noexcept]
```

Get the Position object.

#### Returns

const Vector2f&

Implements [shared::games::components::IPositionableComponent](#).

### 10.65.2.3 getSize()

```
shared::types::Vector2u & ScoreTextDisplayable::getSize (
    void ) [override], [virtual], [noexcept]
```

Get the Size object.

#### Returns

const Vector2u&

Implements [shared::games::components::IPositionableComponent](#).

### 10.65.2.4 getTextProps()

```
shared::games::components::ITextComponent::TextProps ScoreTextDisplayable::getTextProps ( )
[override], [virtual], [noexcept]
```

Get the Text Props object.

#### Returns

[shared::games::components::ITextComponent::TextProps](#)

Implements [shared::games::components::ITextComponent](#).

### 10.65.2.5 getType()

```
const shared::games::components::ComponentType ScoreTextDisplayable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

#### Returns

const shared::games::components::ComponentType

Implements [shared::games::components::IComponent](#).

### 10.65.2.6 getZIndex()

```
unsigned int & ScoreTextDisplayable::getZIndex (
    void ) [override], [virtual], [noexcept]
```

Get the ZIndex object.

#### Returns

unsigned int&

Implements [shared::games::components::IDisplayableComponent](#).



### 10.65.2.7 onMouseHover()

```
void ScoreTextDisplayable::onMouseHover (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Get the Text object.

#### Returns

const std::string&

Implements [shared::games::components::IDisplayableComponent](#).

### 10.65.2.8 onMousePress()

```
void ScoreTextDisplayable::onMousePress (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Get the Text object.

#### Returns

const std::string&

Implements [shared::games::components::IDisplayableComponent](#).

### 10.65.2.9 onMouseRelease()

```
void ScoreTextDisplayable::onMouseRelease (
    std::shared_ptr< shared::games::IGame > ctx ) [override], [virtual]
```

Get the Text object.

#### Returns

const std::string&

Implements [shared::games::components::IDisplayableComponent](#).

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

- [src/games/snake/entities/texts/score/components/ScoreTextDisplayable.hpp](#)
- [src/games/snake/entities/texts/score/components/ScoreTextDisplayable.cpp](#)

## 10.66 ScoreTextEntity Class Reference

Inheritance diagram for ScoreTextEntity:

Collaboration diagram for ScoreTextEntity:

### Public Member Functions

- [ScoreTextEntity](#) ()  
*Construct a new Score Text Entity object.*
- [~ScoreTextEntity](#) ()  
*Destroy a new Score Text Entity object.*
- const [shared::games::components::ComponentsMap](#) & [getComponents](#) (void) const noexcept override  
*Get the Components object.*
- void [updateScore](#) (int score) noexcept

### 10.66.1 Member Function Documentation

### 10.66.1.1 getComponents()

```
const shared::games::components::ComponentsMap & ScoreTextEntity::getComponents (
    void ) const [override], [virtual], [noexcept]
```

Get the Components object.

#### Returns

const components::ComponentsMap&

Implements [shared::games::entity::IEntity](#).

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

- src/games/snake/entities/texts/score/ScoreTextEntity.hpp
- src/games/snake/entities/texts/score/ScoreTextEntity.cpp

## 10.67 SDL2 Class Reference

Inheritance diagram for SDL2:

Collaboration diagram for SDL2:

### Public Member Functions

- [SDL2](#) ()  
*Constructor of [SDL2](#) Class.*
- [~SDL2](#) ()  
*Destructor of [SDL2](#) Class.*
- const [shared::graphics::GraphicsManifest](#) & [getManifest](#) (void) const noexcept override  
*Get the Manifest object.*
- std::unique\_ptr< [shared::graphics::IWindow](#) > [createWindow](#) (const [shared::graphics::IWindow::WindowInitProps](#) &windowProps) override  
*Create a Window object.*
- std::shared\_ptr< [shared::graphics::ISound](#) > [createSound](#) (const std::string &path) override  
*Create a Sound object.*
- std::shared\_ptr< [shared::graphics::ITexture](#) > [createTexture](#) (const std::string &bin, const std::string &ascii) override  
*Create a Texture object.*
- std::shared\_ptr< [shared::graphics::IFont](#) > [createFont](#) (const std::string &path) override  
*Create a Font object.*

### 10.67.1 Member Function Documentation

#### 10.67.1.1 createFont()

```
std::shared_ptr< shared::graphics::IFont > SDL2::createFont (
    const std::string & path ) [override], [virtual]
```

Create a Font object.

#### Parameters

<i>path</i>	
-------------	--

**Returns**

`std::shared_ptr<shared::graphics::IFont>`

Implements [shared::graphics::IGraphicsProvider](#).

**10.67.1.2 createSound()**

```
std::shared_ptr< shared::graphics::ISound > SDL2::createSound (
    const std::string & path ) [override], [virtual]
```

Create a Sound object.

**Parameters**

<i>path</i>	
-------------	--

**Returns**

`std::shared_ptr<shared::graphics::ISound>`

Implements [shared::graphics::IGraphicsProvider](#).

**10.67.1.3 createTexture()**

```
std::shared_ptr< shared::graphics::ITexture > SDL2::createTexture (
    const std::string & bin,
    const std::string & ascii ) [override], [virtual]
```

Create a Texture object.

**Parameters**

<i>bin</i>	
<i>ascii</i>	

**Returns**

`std::shared_ptr<shared::graphics::ITexture>`

Implements [shared::graphics::IGraphicsProvider](#).

**10.67.1.4 createWindow()**

```
std::unique_ptr< shared::graphics::IWindow > SDL2::createWindow (
    const shared::graphics::IWindow::WindowInitProps & windowProps ) [override],
[virtual]
```

Create a Window object.

**Parameters**

<i>windowProps</i>	
--------------------	--

**Returns**

`std::unique_ptr<shared::graphics::IWindow>`

Implements [shared::graphics::IGraphicsProvider](#).

### 10.67.1.5 getManifest()

```
const shared::graphics::GraphicsManifest & SDL2::getManifest (
    void ) const [override], [virtual], [noexcept]
```

Get the Manifest object.

#### Returns

const [shared::graphics::GraphicsManifest](#)&

Implements [shared::graphics::IGraphicsProvider](#).

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

- `src/graphics/sdl2/SDL2.hpp`
- `src/graphics/sdl2/SDL2.cpp`

## 10.68 SDL2Font Class Reference

Inheritance diagram for SDL2Font:

Collaboration diagram for SDL2Font:

### Public Member Functions

- [SDL2Font](#) (std::string pathFont)  
*Construct a new [SDL2Font](#) object.*
- [~SDL2Font](#) ()  
*Destroy the [SDL2Font](#) object.*
- void [setFont](#) (unsigned int characterSize)  
*Set the Font object.*
- TTF\_Font \* [getFont](#) () const  
*Get the Font object.*
- void [setSurface](#) (std::string content, SDL\_Color textColor)  
*Set the Surface object.*
- SDL\_Surface \* [getSurface](#) () const  
*Get the Surface object.*
- void [setTexture](#) (SDL\_Renderer \*renderer)  
*Set the Texture object.*
- SDL\_Texture \* [getTexture](#) () const  
*Get the Texture object.*

### 10.68.1 Constructor & Destructor Documentation

#### 10.68.1.1 SDL2Font()

```
SDL2Font::SDL2Font (
    std::string pathFont ) [explicit]
```

Construct a new [SDL2Font](#) object.

#### Parameters

<i>pathFont</i>	
-----------------	--

## 10.68.2 Member Function Documentation

### 10.68.2.1 getFont()

```
TTF_Font * SDL2Font::getFont ( ) const
```

Get the Font object.

Returns

TTF\_Font\*

### 10.68.2.2 getSurface()

```
SDL_Surface * SDL2Font::getSurface ( ) const
```

Get the Surface object.

Returns

SDL\_Surface\*

### 10.68.2.3 getTexture()

```
SDL_Texture * SDL2Font::getTexture ( ) const
```

Get the Texture object.

Returns

SDL\_Texture\*

### 10.68.2.4 setFont()

```
void SDL2Font::setFont (
    unsigned int characterSize )
```

Set the Font object.

Parameters

<i>characterSize</i>	
----------------------	--

### 10.68.2.5 setSurface()

```
void SDL2Font::setSurface (
    std::string content,
    SDL_Color textColor )
```

Set the Surface object.

Parameters

<i>content</i>	
<i>textColor</i>	

### 10.68.2.6 setTexture()

```
void SDL2Font::setTexture (
    SDL_Renderer * renderer )
```

Set the Texture object.

#### Parameters

<i>renderer</i>	
-----------------	--

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

- src/graphics/sdl2/SDL2Font.hpp
- src/graphics/sdl2/SDL2Font.cpp

## 10.69 SDL2FontException Class Reference

Inheritance diagram for SDL2FontException:

Collaboration diagram for SDL2FontException:

### Public Member Functions

- [SDL2FontException](#) (const char \**where*, const char \**what*)  
*Constructor of [SDL2FontException](#) Class.*
- [~SDL2FontException](#) ()=default  
*Destructor of [SDL2FontException](#) Class.*
- const char \* [where](#) () const noexcept override  
*Get the where object.*
- const char \* [what](#) () const noexcept override  
*Get the what object.*

### 10.69.1 Constructor & Destructor Documentation

#### 10.69.1.1 SDL2FontException()

```
SDL2FontException::SDL2FontException (
    const char * where,
    const char * what ) [inline]
```

Constructor of [SDL2FontException](#) Class.

#### Parameters

<i>where</i>	
<i>what</i>	

### 10.69.2 Member Function Documentation

#### 10.69.2.1 what()

```
const char* SDL2FontException::what ( ) const [inline], [override], [noexcept]
```

Get the what object.

## Returns

const char\*

## 10.69.2.2 where()

```
const char* SDL2FontException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the where object.

## Returns

const char\*

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- src/graphics/sdl2/exceptions/SDL2FontException.hpp

## 10.70 SDL2Sound Class Reference

Inheritance diagram for SDL2Sound:

Collaboration diagram for SDL2Sound:

### Public Member Functions

- [SDL2Sound](#) (const std::string &path)  
*Construct a new [SDL2Sound](#) object.*
- [~SDL2Sound](#) ()  
*Destroy the [SDL2Sound](#) object.*
- void [setState](#) ([shared::graphics::ISound::SoundState](#)) override  
*Set the sound state.*
- [shared::graphics::ISound::SoundState](#) [getState](#) () const override  
*Get the sound state.*
- void [setVolume](#) ([shared::graphics::ISound::SoundVolume](#) volume) override  
*Set the volume of the sound.*
- [shared::graphics::ISound::SoundVolume](#) [getVolume](#) () const override  
*Get the volume of the sound.*
- void [setLoopState](#) (bool loop) override  
*Set the sound to loop or not.*
- bool [getLoopState](#) (void) const override  
*Get the loop state.*

### Additional Inherited Members

#### 10.70.1 Constructor & Destructor Documentation

##### 10.70.1.1 SDL2Sound()

```
SDL2Sound::SDL2Sound (
    const std::string & path ) [explicit]
```

Construct a new [SDL2Sound](#) object.

## Parameters

<i>path</i>	
-------------	--

## 10.70.2 Member Function Documentation

### 10.70.2.1 getLoopState()

```
bool SDL2Sound::getLoopState (
    void ) const [override], [virtual]
```

Get the loop state.

#### Returns

bool

Implements [shared::graphics::ISound](#).

### 10.70.2.2 getState()

```
shared::graphics::ISound::SoundState SDL2Sound::getState (
    void ) const [override], [virtual]
```

Get the sound state.

#### Returns

[shared::graphics::ISound::SoundState](#)

Implements [shared::graphics::ISound](#).

### 10.70.2.3 getVolume()

```
shared::graphics::ISound::SoundVolume SDL2Sound::getVolume (
    void ) const [override], [virtual]
```

Get the volume of the sound.

#### Returns

[shared::graphics::ISound::SoundVolume](#)

Implements [shared::graphics::ISound](#).

### 10.70.2.4 setLoopState()

```
void SDL2Sound::setLoopState (
    bool loop ) [override], [virtual]
```

Set the sound to loop or not.

#### Parameters

<i>loop</i>	
-------------	--

Implements [shared::graphics::ISound](#).

### 10.70.2.5 setState()

```
void SDL2Sound::setState (
    shared::graphics::ISound::SoundState state ) [override], [virtual]
```

Set the sound state.



## Parameters

<i>state</i>	
--------------	--

Implements [shared::graphics::ISound](#).

**10.70.2.6 setVolume()**

```
void SDL2Sound::setVolume (
    shared::graphics::ISound::SoundVolume volume ) [override], [virtual]
```

Set the volume of the sound.

## Parameters

<i>volume</i>	
---------------	--

Implements [shared::graphics::ISound](#).

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

- `src/graphics/sdl2/SDL2Sound.hpp`
- `src/graphics/sdl2/SDL2Sound.cpp`

**10.71 SDL2SoundException Class Reference**

Inheritance diagram for SDL2SoundException:

Collaboration diagram for SDL2SoundException:

**Public Member Functions**

- [SDL2SoundException](#) (const char \*[where](#), const char \*[what](#))  
*Constructor of [SDL2SoundException](#) Class.*
- [~SDL2SoundException](#) ()=default  
*Destructor of [SDL2SoundException](#) Class.*
- const char \* [where](#) () const noexcept override  
*Get the where object.*
- const char \* [what](#) () const noexcept override  
*Get the what object.*

**10.71.1 Constructor & Destructor Documentation****10.71.1.1 SDL2SoundException()**

```
SDL2SoundException::SDL2SoundException (
    const char * where,
    const char * what ) [inline]
```

Constructor of [SDL2SoundException](#) Class.

## Parameters

<i>where</i>	
<i>what</i>	

## 10.71.2 Member Function Documentation

### 10.71.2.1 what()

```
const char* SDL2SoundException::what ( ) const [inline], [override], [noexcept]
```

Get the what object.

#### Returns

const char\*

### 10.71.2.2 where()

```
const char* SDL2SoundException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the where object.

#### Returns

const char\*

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- [src/graphics/sdl2/exceptions/SDL2SoundException.hpp](#)

## 10.72 SDL2Texture Class Reference

Inheritance diagram for SDL2Texture:

Collaboration diagram for SDL2Texture:

### Public Member Functions

- [SDL2Texture](#) (std::string pathTexture)  
*Constructor of [SDL2Texture](#) Class.*
- [~SDL2Texture](#) ()  
*Destructor of [SDL2Texture](#) Class.*
- void [setTexture](#) (SDL\_Renderer \*renderer)  
*Get the Path object.*
- SDL\_Texture \* [getTexture](#) () const  
*Get the Texture object.*

### 10.72.1 Constructor & Destructor Documentation

#### 10.72.1.1 SDL2Texture()

```
SDL2Texture::SDL2Texture (
    std::string pathTexture ) [explicit]
```

Constructor of [SDL2Texture](#) Class.

#### Parameters

<i>pathTexture</i>	
--------------------	--

## 10.72.2 Member Function Documentation

### 10.72.2.1 getTexture()

`SDL_Texture * SDL2Texture::getTexture ( ) const`  
Get the Texture object.

Returns

`SDL_Texture*`

### 10.72.2.2 setTexture()

`void SDL2Texture::setTexture (`  
     `SDL_Renderer * renderer )`  
Get the Path object.

Parameters

<code>renderer</code>	
-----------------------	--

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

- `src/graphics/sdl2/SDL2Texture.hpp`
- `src/graphics/sdl2/SDL2Texture.cpp`

## 10.73 SDL2TextureException Class Reference

Inheritance diagram for `SDL2TextureException`:

Collaboration diagram for `SDL2TextureException`:

### Public Member Functions

- `SDL2TextureException (const char *where, const char *what)`  
     *Constructor of `SDL2TextureException` Class.*
- `~SDL2TextureException ()=default`  
     *Destructor of `SDL2TextureException` Class.*
- `const char * where () const noexcept override`  
     *Get the where object.*
- `const char * what () const noexcept override`  
     *Get the what object.*

### 10.73.1 Constructor & Destructor Documentation

#### 10.73.1.1 SDL2TextureException()

`SDL2TextureException::SDL2TextureException (`  
     `const char * where,`  
     `const char * what ) [inline]`

Constructor of `SDL2TextureException` Class.

## Parameters

<i>where</i>	
<i>what</i>	

## 10.73.2 Member Function Documentation

### 10.73.2.1 what()

```
const char* SDL2TextureException::what ( ) const [inline], [override], [noexcept]
```

Get the what object.

## Returns

const char\*

### 10.73.2.2 where()

```
const char* SDL2TextureException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the where object.

## Returns

const char\*

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- [src/graphics/sdl2/exceptions/SDL2TextureException.hpp](#)

## 10.74 SDL2Window Class Reference

Inheritance diagram for SDL2Window:

Collaboration diagram for SDL2Window:

### Public Member Functions

- [SDL2Window](#) (const [shared::graphics::IWindow::WindowInitProps](#) &windowProps)  
*Construct a new [SDL2Window](#) object.*
- [~SDL2Window](#) ()  
*Destroy the [SDL2Window](#) object.*
- void [setTitle](#) (const std::string &title) override  
*Set the Title of the Window object.*
- void [setSize](#) ([Vector2u](#) size) override  
*Set the Size of the Window object.*
- [Vector2u](#) [getSize](#) () const override  
*Get the Size of the Window object.*
- void [setFramerateLimit](#) (unsigned int fps) override  
*Set the Framerate Limit of the Window object.*
- unsigned int [getFramerateLimit](#) () const override  
*Get the Framerate Limit of the Window object.*
- void [setMode](#) ([shared::graphics::IWindow::WindowMode](#) mode) override  
*Set the Mode of the Window object.*

- [shared::graphics::IWindow::WindowMode](#) [getMode](#) (void) const override  
*Get the Mode of the Window object.*
- void [setIcon](#) (const std::string &icon) override  
*Set the Icon of the Window object.*
- void [render](#) (const [shared::graphics::TextureProps](#) &props) override  
*Render the texture props on the window.*
- void [render](#) (const [shared::graphics::TextProps](#) &props) override  
*Render the texts props on the window.*
- void [clear](#) (void) override  
*Clear the window.*
- void [display](#) (void) override  
*Display the window.*
- void [close](#) (void) override  
*Close the window.*
- bool [isOpen](#) (void) const override  
*Check if the window is open.*
- std::vector< std::shared\_ptr< [shared::graphics::events::IEvent](#) > > [getEvents](#) (void) override  
*Get the Events object.*
- [shared::graphics::events::IKeyEvent::KeyType](#) [mapSDL2KeyToKeyType](#) (SDL\_Keycode sdl2Key)  
*Map the key to keyTypes to make it match our key types.*
- [shared::graphics::events::IKeyEvent::KeyCode](#) [mapSDL2KeyToKeyCode](#) (SDL\_Keycode sdl2Key, [shared::graphics::events::IKeyEvent](#) type)  
*Map the key to keyCodes to make it match our key codes.*

## Additional Inherited Members

### 10.74.1 Constructor & Destructor Documentation

#### 10.74.1.1 SDL2Window()

```
SDL2Window::SDL2Window (
    const shared::graphics::IWindow::WindowInitProps & windowProps ) [explicit]
```

Construct a new [SDL2Window](#) object.

##### Parameters

<i>windowProps</i>	
--------------------	--

### 10.74.2 Member Function Documentation

#### 10.74.2.1 getEvents()

```
std::vector< std::shared_ptr< shared::graphics::events::IEvent > > SDL2Window::getEvents (
    void ) [override], [virtual]
```

Get the Events object.

##### Returns

std::vector<std::shared\_ptr<shared::graphics::events::IEvent>>

Implements [shared::graphics::IWindow](#).

**10.74.2.2 getFramerateLimit()**

`unsigned int SDL2Window::getFramerateLimit ( ) const [override], [virtual]`  
 Get the Framerate Limit of the Window object.

**Returns**

unsigned int

Implements [shared::graphics::IWindow](#).

**10.74.2.3 getMode()**

`shared::graphics::IWindow::WindowMode SDL2Window::getMode ( void ) const [override], [virtual]`

Get the Mode of the Window object.

**Returns**

[shared::graphics::IWindow::WindowMode](#)

Implements [shared::graphics::IWindow](#).

**10.74.2.4 getSize()**

`Vector2u SDL2Window::getSize ( void ) const [override], [virtual]`

Get the Size of the Window object.

**Returns**

Vector2u

Implements [shared::graphics::IWindow](#).

**10.74.2.5 isOpen()**

`bool SDL2Window::isOpen ( void ) const [override], [virtual]`

Check if the window is open.

**Returns**

bool

Implements [shared::graphics::IWindow](#).

**10.74.2.6 mapSDL2KeyToKeyCode()**

`shared::graphics::events::IKeyEvent::KeyCode SDL2Window::mapSDL2KeyToKeyCode ( SDL_Keycode sdl2Key, shared::graphics::events::IKeyEvent::KeyType type )`

Map the key to keyCodes to make it match our key codes.

**Parameters**

<i>sflmKey</i>	
<i>type</i>	

## Returns

[shared::graphics::events::IKeyEvent::KeyCode](#)

**10.74.2.7 mapSDL2KeyToKeyType()**

```
shared::graphics::events::IKeyEvent::KeyType SDL2Window::mapSDL2KeyToKeyType (
    SDL_Keycode sdl2Key )
```

Map the key to keyTypes to make it match our key types.

## Parameters

<i>sfmIKey</i>	
----------------	--

## Returns

[shared::graphics::events::IKeyEvent::KeyType](#)

**10.74.2.8 render() [1/2]**

```
void SDL2Window::render (
    const shared::graphics::TextProps & props ) [override], [virtual]
```

Render the texts props on the window.

## Parameters

<i>props</i>	
--------------	--

Implements [shared::graphics::IWindow](#).

**10.74.2.9 render() [2/2]**

```
void SDL2Window::render (
    const shared::graphics::TextureProps & props ) [override], [virtual]
```

Render the texture props on the window.

## Parameters

<i>props</i>	
--------------	--

Implements [shared::graphics::IWindow](#).

**10.74.2.10 setFrameRateLimit()**

```
void SDL2Window::setFrameRateLimit (
    unsigned int fps ) [override], [virtual]
```

Set the Framerate Limit of the Window object.

## Parameters

<i>fps</i>	
------------	--

Implements [shared::graphics::IWindow](#).

#### 10.74.2.11 setIcon()

```
void SDL2Window::setIcon (
    const std::string & icon ) [override], [virtual]
```

Set the Icon of the Window object.

##### Parameters

<i>icon</i>	
-------------	--

Implements [shared::graphics::IWindow](#).

#### 10.74.2.12 setMode()

```
void SDL2Window::setMode (
    shared::graphics::IWindow::WindowMode mode ) [override], [virtual]
```

Set the Mode of the Window object.

##### Parameters

<i>mode</i>	
-------------	--

Implements [shared::graphics::IWindow](#).

#### 10.74.2.13 setSize()

```
void SDL2Window::setSize (
    Vector2u size ) [override], [virtual]
```

Set the Size of the Window object.

##### Parameters

<i>size</i>	
-------------	--

Implements [shared::graphics::IWindow](#).

#### 10.74.2.14 setTitle()

```
void SDL2Window::setTitle (
    const std::string & title ) [override], [virtual]
```

Set the Title of the Window object.

##### Parameters

<i>title</i>	
--------------	--

Implements [shared::graphics::IWindow](#).

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

- src/graphics/sdl2/Window/SDL2Window.hpp
- src/graphics/sdl2/Window/SDL2Window.cpp



## 10.75 SDL2WindowException Class Reference

Inheritance diagram for SDL2WindowException:

Collaboration diagram for SDL2WindowException:

### Public Member Functions

- [SDL2WindowException](#) (const char \*[where](#), const char \*[what](#))  
*Constructor of [SDL2WindowException](#) Class.*
- [~SDL2WindowException](#) ()=default  
*Destructor of [SDL2WindowException](#) Class.*
- const char \* [where](#) () const noexcept override  
*Get the where object.*
- const char \* [what](#) () const noexcept override  
*Get the what object.*

### 10.75.1 Constructor & Destructor Documentation

#### 10.75.1.1 SDL2WindowException()

```
SDL2WindowException::SDL2WindowException (
    const char * where,
    const char * what ) [inline]
```

Constructor of [SDL2WindowException](#) Class.

##### Parameters

<i>where</i>	
<i>what</i>	

### 10.75.2 Member Function Documentation

#### 10.75.2.1 what()

```
const char* SDL2WindowException::what ( ) const [inline], [override], [noexcept]
```

Get the what object.

##### Returns

const char\*

#### 10.75.2.2 where()

```
const char* SDL2WindowException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the where object.

##### Returns

const char\*

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- src/graphics/sdl2/exceptions/SDL2WindowException.hpp

## 10.76 SFML Class Reference

Inheritance diagram for SFML:

Collaboration diagram for SFML:

### Public Member Functions

- [SFML \(\)](#)  
*Construct a new [SFML](#) object.*
- [~SFML \(\)](#)  
*Destroy the [SFML](#) object.*
- const [shared::graphics::GraphicsManifest](#) & [getManifest](#) (void) const noexcept override  
*Get the Manifest object.*
- std::unique\_ptr< [shared::graphics::IWindow](#) > [createWindow](#) (const [shared::graphics::IWindow::WindowInitProps](#) &windowProps) override  
*Create a Window object.*
- std::shared\_ptr< [shared::graphics::ISound](#) > [createSound](#) (const std::string &path) override  
*Create a Sound object.*
- std::shared\_ptr< [shared::graphics::ITexture](#) > [createTexture](#) (const std::string &bin, const std::string &ascii) override  
*Create a Texture object.*
- std::shared\_ptr< [shared::graphics::IFont](#) > [createFont](#) (const std::string &path) override  
*Create a Font object.*

### 10.76.1 Member Function Documentation

#### 10.76.1.1 createFont()

```
std::shared_ptr< shared::graphics::IFont > SFML::createFont (
    const std::string & path ) [override], [virtual]
```

Create a Font object.

##### Parameters

<i>path</i>	
-------------	--

##### Returns

std::shared\_ptr<shared::graphics::IFont>

Implements [shared::graphics::IGraphicsProvider](#).

#### 10.76.1.2 createSound()

```
std::shared_ptr< shared::graphics::ISound > SFML::createSound (
    const std::string & path ) [override], [virtual]
```

Create a Sound object.

##### Parameters

<i>path</i>	
-------------	--

## Returns

`std::shared_ptr<shared::graphics::ISound>`

Implements [shared::graphics::IGraphicsProvider](#).

**10.76.1.3 createTexture()**

```
std::shared_ptr< shared::graphics::ITexture > SFML::createTexture (
    const std::string & bin,
    const std::string & ascii ) [override], [virtual]
```

Create a Texture object.

## Parameters

<i>bin</i>	
<i>ascii</i>	

## Returns

`std::shared_ptr<shared::graphics::ITexture>`

Implements [shared::graphics::IGraphicsProvider](#).

**10.76.1.4 createWindow()**

```
std::unique_ptr< shared::graphics::IWindow > SFML::createWindow (
    const shared::graphics::IWindow::WindowInitProps & windowProps ) [override],
[virtual]
```

Create a Window object.

## Parameters

<i>windowProps</i>	
--------------------	--

## Returns

`std::unique_ptr<shared::graphics::IWindow>`

Implements [shared::graphics::IGraphicsProvider](#).

**10.76.1.5 getManifest()**

```
const shared::graphics::GraphicsManifest & SFML::getManifest (
    void ) const [override], [virtual], [noexcept]
```

Get the Manifest object.

## Returns

[shared::graphics::GraphicsManifest](#)

Implements [shared::graphics::IGraphicsProvider](#).

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

- `src/graphics/sfml/SFML.hpp`
- `src/graphics/sfml/SFML.cpp`

## 10.77 SFMLFont Class Reference

Inheritance diagram for SFMLFont:

Collaboration diagram for SFMLFont:

### Public Member Functions

- [SFMLFont](#) (std::string pathFont)  
*Construct a new [SFMLFont](#) object.*
- [~SFMLFont](#) ()  
*Destroy the [SFMLFont](#) object.*
- std::string [getPath](#) () const  
*Get the path to the Font object.*

### 10.77.1 Constructor & Destructor Documentation

#### 10.77.1.1 SFMLFont()

```
SFMLFont::SFMLFont (
    std::string pathFont ) [explicit]
```

Construct a new [SFMLFont](#) object.

Parameters

<i>pathFont</i>	
-----------------	--

### 10.77.2 Member Function Documentation

#### 10.77.2.1 getPath()

```
std::string SFMLFont::getPath (
    void ) const
```

Get the path to the Font object.

Returns

path

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

- src/graphicsals/sfml/SFMLFont.hpp
- src/graphicsals/sfml/SFMLFont.cpp

## 10.78 SFMLFontException Class Reference

Inheritance diagram for SFMLFontException:

Collaboration diagram for SFMLFontException:

### Public Member Functions

- [SFMLFontException](#) (const char \*where, const char \*what)  
*Construct a new [SFMLFontException](#) object.*
- [~SFMLFontException](#) ()=default

Destroy the [SFMLFontException](#) object.

- `const char * where () const` noexcept override

Get the *where* object.

- `const char * what () const` noexcept override

Get the *what* object.

## 10.78.1 Constructor & Destructor Documentation

### 10.78.1.1 SFMLFontException()

```
SFMLFontException::SFMLFontException (
    const char * where,
    const char * what ) [inline]
```

Construct a new [SFMLFontException](#) object.

#### Parameters

<i>where</i>	
<i>what</i>	

## 10.78.2 Member Function Documentation

### 10.78.2.1 what()

```
const char* SFMLFontException::what ( ) const [inline], [override], [noexcept]
```

Get the *what* object.

#### Returns

`const char*`

### 10.78.2.2 where()

```
const char* SFMLFontException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the *where* object.

#### Returns

`const char*`

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- `src/graphicsals/sfml/exceptions/SFMLFontException.hpp`

## 10.79 SFMLSound Class Reference

Inheritance diagram for SFMLSound:

Collaboration diagram for SFMLSound:

## Public Member Functions

- [SFMLSound](#) (const std::string &path)  
*Construct a new [SFMLSound](#) object.*
- [~SFMLSound](#) ()  
*Destroy the [SFMLSound](#) object.*
- void [setState](#) (shared::graphics::ISound::SoundState) override  
*Set the sound state.*
- [shared::graphics::ISound::SoundState getState](#) () const override  
*Get the sound state.*
- void [setVolume](#) (shared::graphics::ISound::SoundVolume volume) override  
*Set the volume of the sound.*
- [shared::graphics::ISound::SoundVolume getVolume](#) () const override  
*Get the volume of the sound.*
- void [setLoopState](#) (bool loop) override  
*Set the sound to loop or not.*
- bool [getLoopState](#) (void) const override  
*Get the loop state of the sound.*

## Additional Inherited Members

### 10.79.1 Constructor & Destructor Documentation

#### 10.79.1.1 SFMLSound()

```
SFMLSound::SFMLSound (
    const std::string & path ) [explicit]
```

Construct a new [SFMLSound](#) object.

##### Parameters

<i>path</i>	
-------------	--

### 10.79.2 Member Function Documentation

#### 10.79.2.1 getLoopState()

```
bool SFMLSound::getLoopState (
    void ) const [override], [virtual]
```

Get the loop state of the sound.

##### Returns

bool

Implements [shared::graphics::ISound](#).

#### 10.79.2.2 getState()

```
shared::graphics::ISound::SoundState SFMLSound::getState (
    void ) const [override], [virtual]
```

Get the sound state.

Returns

[shared::graphics::ISound::SoundState](#)

Implements [shared::graphics::ISound](#).

### 10.79.2.3 getVolume()

```
shared::graphics::ISound::SoundVolume SFMLSound::getVolume (
    void ) const [override], [virtual]
```

Get the volume of the sound.

Returns

[shared::graphics::ISound::SoundVolume](#)

Implements [shared::graphics::ISound](#).

### 10.79.2.4 setState()

```
void SFMLSound::setState (
    shared::graphics::ISound::SoundState state ) [override], [virtual]
```

Set the sound state.

Parameters

<i>state</i>	
--------------	--

Implements [shared::graphics::ISound](#).

### 10.79.2.5 setVolume()

```
void SFMLSound::setVolume (
    shared::graphics::ISound::SoundVolume volume ) [override], [virtual]
```

Set the volume of the sound.

Parameters

<i>volume</i>	
---------------	--

Implements [shared::graphics::ISound](#).

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

- [src/graphics/sfml/SFMLSound.hpp](#)
- [src/graphics/sfml/SFMLSound.cpp](#)

## 10.80 SFMLSoundException Class Reference

Inheritance diagram for SFMLSoundException:

Collaboration diagram for SFMLSoundException:

### Public Member Functions

- [SFMLSoundException](#) (const char \*where, const char \*what)  
*Construct a new [SFMLSoundException](#) object.*
- [~SFMLSoundException](#) ()=default  
*Destroy the [SFMLSoundException](#) object.*

- `const char * where () const noexcept override`  
*Get the where object.*
- `const char * what () const noexcept override`  
*Get the what object.*

## 10.80.1 Constructor & Destructor Documentation

### 10.80.1.1 SFMLSoundException()

```
SFMLSoundException::SFMLSoundException (
    const char * where,
    const char * what ) [inline]
```

Construct a new [SFMLSoundException](#) object.

#### Parameters

<i>where</i>	
<i>what</i>	

## 10.80.2 Member Function Documentation

### 10.80.2.1 what()

```
const char* SFMLSoundException::what ( ) const [inline], [override], [noexcept]
```

Get the what object.

#### Returns

`const char*`

### 10.80.2.2 where()

```
const char* SFMLSoundException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the where object.

#### Returns

`const char*`

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- `src/graphicsals/sfml/exceptions/SFMLSoundException.hpp`

## 10.81 SFMLTexture Class Reference

Inheritance diagram for SFMLTexture:

Collaboration diagram for SFMLTexture:

### Public Member Functions

- [SFMLTexture](#) (`std::string pathTexture`)  
*Construct a new [SFMLTexture](#) object.*



- [~SFMLTexture](#) ()  
*Destroy the [SFMLTexture](#) object.*
- `sf::Texture` [getTexture](#) () const  
*Get the texture.*

## 10.81.1 Constructor & Destructor Documentation

### 10.81.1.1 SFMLTexture()

```
SFMLTexture::SFMLTexture (
    std::string pathTexture ) [explicit]
```

Construct a new [SFMLTexture](#) object.

Parameters

<i>pathTexture</i>	
--------------------	--

## 10.81.2 Member Function Documentation

### 10.81.2.1 getTexture()

```
sf::Texture SFMLTexture::getTexture ( ) const
```

Get the texture.

Returns

`sf::Texture`

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

- `src/graphicsals/sfml/SFMLTexture.hpp`
- `src/graphicsals/sfml/SFMLTexture.cpp`

## 10.82 SFMLTextureException Class Reference

Inheritance diagram for SFMLTextureException:

Collaboration diagram for SFMLTextureException:

### Public Member Functions

- [SFMLTextureException](#) (const char \*[where](#), const char \*[what](#))  
*Construct a new [SFMLTextureException](#) object.*
- [~SFMLTextureException](#) ()=default  
*Destroy the [SFMLTextureException](#) object.*
- const char \* [where](#) () const noexcept override  
*Get the where object.*
- const char \* [what](#) () const noexcept override  
*Get the what object.*

## 10.82.1 Constructor & Destructor Documentation

### 10.82.1.1 SFMLTextureException()

```
SFMLTextureException::SFMLTextureException (
    const char * where,
    const char * what ) [inline]
```

Construct a new [SFMLTextureException](#) object.

#### Parameters

<i>where</i>	
<i>what</i>	

## 10.82.2 Member Function Documentation

### 10.82.2.1 what()

```
const char* SFMLTextureException::what ( ) const [inline], [override], [noexcept]
```

Get the what object.

#### Returns

const char\*

### 10.82.2.2 where()

```
const char* SFMLTextureException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the where object.

#### Returns

const char\*

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- `src/graphicsals/sfml/exceptions/SFMLTextureException.hpp`

## 10.83 SFMLWindow Class Reference

Inheritance diagram for SFMLWindow:

Collaboration diagram for SFMLWindow:

### Public Member Functions

- [SFMLWindow](#) (const [shared::graphics::IWindow::WindowInitProps](#) &windowProps)  
*Construct a new [SFMLWindow](#) object.*
- [~SFMLWindow](#) ()  
*Destroy the [SFMLWindow](#) object.*
- void [setTitle](#) (const std::string &title) override  
*Set the Window Title object.*
- void [setSize](#) ([Vector2u](#) size) override  
*Set the size of the window.*
- [Vector2u](#) [getSize](#) () const override  
*Get the size of the window.*

- void `setFramerateLimit` (unsigned int fps) override  
*Set the framerate limit.*
- unsigned int `getFramerateLimit` () const override  
*Get the framerate limit.*
- void `setMode` (shared::graphics::IWindow::WindowMode mode) override  
*Set the mode of the window.*
- shared::graphics::IWindow::WindowMode `getMode` (void) const override  
*Get the mode of the window.*
- void `setIcon` (const std::string &icon) override  
*Set the Window Icon object.*
- void `render` (const shared::graphics::TextureProps &props) override  
*Render the texture props on the window.*
- void `render` (const shared::graphics::TextProps &props) override  
*Render the texts props on the window.*
- void `clear` (void) override  
*Clear the window.*
- void `display` (void) override  
*Display the window.*
- void `close` (void) override  
*Close the window.*
- bool `isOpen` (void) const override  
*Check if the window is open.*
- std::vector< std::shared\_ptr< shared::graphics::events::IEvent > > `getEvents` (void) override  
*Get the events object.*
- shared::graphics::events::IKeyEvent::KeyType `mapSFMLKeyToKeyType` (sf::Keyboard::Key sfmlKey)  
*Map the key to keyTypes to make it match our key types.*
- shared::graphics::events::IKeyEvent::KeyCode `mapSFMLKeyToKeyCode` (sf::Keyboard::Key sfmlKey, shared::graphics::events::IKeyEvent::KeyType type)  
*Map the key to keyCodes to make it match our key codes.*

## Additional Inherited Members

### 10.83.1 Constructor & Destructor Documentation

#### 10.83.1.1 SFMLWindow()

```
SFMLWindow::SFMLWindow (
    const shared::graphics::IWindow::WindowInitProps & windowProps ) [explicit]
```

Construct a new `SFMLWindow` object.

#### Parameters

<code>windowProps</code>	
--------------------------	--

### 10.83.2 Member Function Documentation

### 10.83.2.1 `getEvents()`

```
std::vector< shared::graphics::events::EventPtr > SFMLWindow::getEvents (
    void ) [override], [virtual]
```

Get the events object.

Returns

`std::vector<std::shared_ptr<shared::graphics::events::IEvent>>`

Implements [shared::graphics::IWindow](#).

### 10.83.2.2 `getFramerateLimit()`

```
unsigned int SFMLWindow::getFramerateLimit ( ) const [override], [virtual]
```

Get the framerate limit.

Returns

`unsigned int`

Implements [shared::graphics::IWindow](#).

### 10.83.2.3 `getMode()`

```
shared::graphics::IWindow::WindowMode SFMLWindow::getMode (
    void ) const [override], [virtual]
```

Get the mode of the window.

Returns

[shared::graphics::IWindow::WindowMode](#)

Implements [shared::graphics::IWindow](#).

### 10.83.2.4 `getSize()`

```
Vector2u SFMLWindow::getSize (
    void ) const [override], [virtual]
```

Get the size of the window.

Returns

`Vector2u`

Implements [shared::graphics::IWindow](#).

### 10.83.2.5 `isOpen()`

```
bool SFMLWindow::isOpen (
    void ) const [override], [virtual]
```

Check if the window is open.

Returns

`bool`

Implements [shared::graphics::IWindow](#).

### 10.83.2.6 mapSFMLKeyToKeyCode()

```
shared::graphics::events::IKeyEvent::KeyCode SFMLWindow::mapSFMLKeyToKeyCode (
    sf::Keyboard::Key sFMLKey,
    shared::graphics::events::IKeyEvent::KeyType type )
```

Map the key to keyCodes to make it match our key codes.

## Parameters

<i>sfmlKey</i>	
<i>type</i>	

## Returns

[shared::graphics::events::IKeyEvent::KeyCode](#)

**10.83.2.7 mapSFMLKeyToKeyType()**

```
shared::graphics::events::IKeyEvent::KeyType SFMLWindow::mapSFMLKeyToKeyType (
    sf::Keyboard::Key sfmlKey )
```

Map the key to keyTypes to make it match our key types.

## Parameters

<i>sfmlKey</i>	
----------------	--

## Returns

[shared::graphics::events::IKeyEvent::KeyType](#)

**10.83.2.8 render() [1/2]**

```
void SFMLWindow::render (
    const shared::graphics::TextProps & props ) [override], [virtual]
```

Render the texts props on the window.

## Parameters

<i>props</i>	
--------------	--

Implements [shared::graphics::IWindow](#).

**10.83.2.9 render() [2/2]**

```
void SFMLWindow::render (
    const shared::graphics::TextureProps & props ) [override], [virtual]
```

Render the texture props on the window.

## Parameters

<i>props</i>	
--------------	--

Implements [shared::graphics::IWindow](#).

**10.83.2.10 setFrameRateLimit()**

```
void SFMLWindow::setFrameRateLimit (
    unsigned int fps ) [override], [virtual]
```

Set the framerate limit.

## Parameters

<i>fps</i>	
------------	--

Implements [shared::graphics::IWindow](#).

**10.83.2.11 setIcon()**

```
void SFMLWindow::setIcon (
    const std::string & icon ) [override], [virtual]
```

Set the Window Icon object.

## Parameters

<i>icon</i>	
-------------	--

Implements [shared::graphics::IWindow](#).

**10.83.2.12 setMode()**

```
void SFMLWindow::setMode (
    shared::graphics::IWindow::WindowMode mode ) [override], [virtual]
```

Set the mode of the window.

## Parameters

<i>mode</i>	
-------------	--

Implements [shared::graphics::IWindow](#).

**10.83.2.13 setTitle()**

```
void SFMLWindow::setTitle (
    const std::string & title ) [override], [virtual]
```

Set the Window Title object.

## Parameters

<i>title</i>	
--------------	--

Implements [shared::graphics::IWindow](#).

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

- src/graphics/sfml/Window/SFMLWindow.hpp
- src/graphics/sfml/Window/SFMLWindow.cpp

**10.84 SFMLWindowException Class Reference**

Inheritance diagram for SFMLWindowException:

Collaboration diagram for SFMLWindowException:

**Public Member Functions**

- [SFMLWindowException](#) (const char \*where, const char \*what)

Constructor of [SFMLWindowException](#) Class.

- [~SFMLWindowException](#) ()=default

Destructor of [SFMLWindowException](#) Class.

- const char \* [where](#) () const noexcept override

Get the where object.

- const char \* [what](#) () const noexcept override

Get the what object.

## 10.84.1 Constructor & Destructor Documentation

### 10.84.1.1 SFMLWindowException()

```
SFMLWindowException::SFMLWindowException (
    const char * where,
    const char * what ) [inline]
```

Constructor of [SFMLWindowException](#) Class.

Parameters

<i>where</i>	
<i>what</i>	

## 10.84.2 Member Function Documentation

### 10.84.2.1 what()

```
const char* SFMLWindowException::what ( ) const [inline], [override], [noexcept]
```

Get the what object.

Returns

const char\*

### 10.84.2.2 where()

```
const char* SFMLWindowException::where ( ) const [inline], [override], [virtual], [noexcept]
```

Get the where object.

Returns

const char\*

Implements [shared::graphics::exceptions::IGraphicsException](#).

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

- src/graphicsals/sfml/exceptions/SFMLWindowException.hpp

## 10.85 SnakeBodyCollidable Class Reference

Inheritance diagram for SnakeBodyCollidable:

Collaboration diagram for SnakeBodyCollidable:



## Public Member Functions

- [SnakeBodyCollidable](#) (const [shared::games::entity::IEntity](#) &entity, unsigned int id)  
*Construct a new SnakeBody Collidable object.*
- [~SnakeBodyCollidable](#) ()  
*Destroy the SnakeBody Collidable object.*
- const [shared::games::components::ComponentType](#) [getType](#) () const noexcept override  
*Get the Type object.*
- const [shared::games::entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- [Vector2f](#) & [getPosition](#) (void) noexcept override  
*Get position of the entity (tiles)*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get size of the entity (tiles)*
- unsigned int [getId](#) (void) noexcept
- void [setPosition](#) ([Vector2f](#) pos) noexcept
- void [onCollide](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx, std::shared\_ptr< [shared::games::components::ICollidableComp](#)  
> target) override  
*On collide event handler for the component.*

## 10.85.1 Constructor & Destructor Documentation

### 10.85.1.1 SnakeBodyCollidable()

```
SnakeBodyCollidable::SnakeBodyCollidable (
    const shared::games::entity::IEntity & entity,
    unsigned int id ) [explicit]
```

Construct a new SnakeBody Collidable object.

#### Parameters

<a href="#">entity</a>	
------------------------	--

## 10.85.2 Member Function Documentation

### 10.85.2.1 getEntity()

```
const shared::games::entity::IEntity & SnakeBodyCollidable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

#### Returns

const [shared::games::entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.85.2.2 getType()

```
const shared::games::components::ComponentType SnakeBodyCollidable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

**Returns**

const shared::games::components::ComponentType

Implements [shared::games::components::IComponent](#).

**10.85.2.3 onCollide()**

```
void SnakeBodyCollidable::onCollide (
    std::shared_ptr< shared::games::IGame > ctx,
    std::shared_ptr< shared::games::components::ICollidableComponent > target ) [override],
[virtual]
```

On collide event handler for the component.

**Parameters**

<i>ctx</i>	Context of the game
<i>target</i>	Target entity

Implements [shared::games::components::ICollidableComponent](#).

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

- src/games/snake/entities/snake\_body/components/SnakeBodyCollidable.hpp
- src/games/snake/entities/snake\_body/components/SnakeBodyCollidable.cpp

**10.86 SnakeBodyDisplayable Class Reference**

Inheritance diagram for SnakeBodyDisplayable:

Collaboration diagram for SnakeBodyDisplayable:

**Public Member Functions**

- [SnakeBodyDisplayable](#) (const [entity::IEntity](#) &entity, [Vector2f](#) position)  
*Construct a new Snake Body Displayable object.*
- [~SnakeBodyDisplayable](#) ()  
*Destroy a new Snake Body Displayable object.*
- const components::ComponentType [getType](#) () const noexcept override  
*Get the Type object.*
- const [entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get the Size object.*
- unsigned int & [getZIndex](#) (void) noexcept override  
*Get the ZIndex object.*
- components::TextureProps & [getTextureProps](#) (void) noexcept override  
*Get the TextureProps object.*
- void [onMousePress](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*handle the mouse press event*
- void [onMouseHover](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*handle the mouse hover event*
- void [onMouseRelease](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*handle the mouse release event*
- void [setPosition](#) ([Vector2f](#) pos) noexcept  
*Set the Position object.*

- void [setOldPosition](#) ([Vector2f](#) pos) noexcept  
*Set the Old Position object.*
- [Vector2f](#) & [getPosition](#) (void) noexcept override  
*Get the Position object.*
- [Vector2f](#) & [getOldPosition](#) (void) noexcept  
*Get the Old Position object.*

## Public Attributes

- [Vector2f](#) [\\_position](#)
- [Vector2f](#) [\\_oldPosition](#)
- [components::TextureProps](#) [\\_textureProps](#)

## 10.86.1 Constructor & Destructor Documentation

### 10.86.1.1 SnakeBodyDisplayable()

```
SnakeBodyDisplayable::SnakeBodyDisplayable (
    const entity::IEntity & entity,
    Vector2f position ) [explicit]
```

Construct a new Snake Body Displayable object.

#### Parameters

<i>entity</i>	
<i>position</i>	

## 10.86.2 Member Function Documentation

### 10.86.2.1 getEntity()

```
const entity::IEntity & SnakeBodyDisplayable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

#### Returns

const [entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.86.2.2 getOldPosition()

```
Vector2f & SnakeBodyDisplayable::getOldPosition (
    void ) [noexcept]
```

Get the Old Position object.

#### Returns

[Vector2f](#)&

### 10.86.2.3 getPosition()

```
Vector2f & SnakeBodyDisplayable::getPosition (
    void ) [override], [virtual], [noexcept]
```

Get the Position object.

Returns

Vector2f&

Implements [shared::games::components::IPositionableComponent](#).

### 10.86.2.4 getSize()

```
Vector2u & SnakeBodyDisplayable::getSize (
    void ) [override], [virtual], [noexcept]
```

Get the Size object.

Returns

const Vector2u&

Implements [shared::games::components::IPositionableComponent](#).

### 10.86.2.5 getTextureProps()

```
components::TextureProps & SnakeBodyDisplayable::getTextureProps (
    void ) [override], [virtual], [noexcept]
```

Get the TextureProps object.

Returns

components::TextureProps&

Implements [shared::games::components::ITextureComponent](#).

### 10.86.2.6 getType()

```
const components::ComponentType SnakeBodyDisplayable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

Returns

const components::ComponentType

Implements [shared::games::components::IComponent](#).

### 10.86.2.7 getZIndex()

```
unsigned int & SnakeBodyDisplayable::getZIndex (
    void ) [override], [virtual], [noexcept]
```

Get the ZIndex object.

Returns

unsigned int&

Implements [shared::games::components::IDisplayableComponent](#).

### 10.86.2.8 onMouseHover()

```
void SnakeBodyDisplayable::onMouseHover (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse hover event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.86.2.9 onMousePress()

```
void SnakeBodyDisplayable::onMousePress (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse press event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.86.2.10 onMouseRelease()

```
void SnakeBodyDisplayable::onMouseRelease (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse release event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.86.2.11 setOldPosition()

```
void SnakeBodyDisplayable::setOldPosition (
    Vector2f pos ) [noexcept]
```

Set the Old Position object.

#### Parameters

<i>pos</i>	
------------	--

**Returns**

void

**10.86.2.12 setPosition()**

```
void SnakeBodyDisplayable::setPosition (
    Vector2f pos ) [noexcept]
```

Set the Position object.

**Parameters**

<i>pos</i>	
------------	--

**Returns**

void

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

- src/games/snake/entities/snake\_body/components/SnakeBodyDisplayable.hpp
- src/games/snake/entities/snake\_body/components/SnakeBodyDisplayable.cpp

**10.87 SnakeBodyEntity Class Reference**

Inheritance diagram for SnakeBodyEntity:

Collaboration diagram for SnakeBodyEntity:

**Public Member Functions**

- [SnakeBodyEntity](#) ([Vector2f](#) position, unsigned int id)  
*Construct a new Snake Body Entity object.*
- [~SnakeBodyEntity](#) ()  
*Destroy a new Snake Body Entity object.*
- const components::ComponentsMap & [getComponents](#) (void) const noexcept override  
*Get the Components object.*

**10.87.1 Constructor & Destructor Documentation****10.87.1.1 SnakeBodyEntity()**

```
SnakeBodyEntity::SnakeBodyEntity (
    Vector2f position,
    unsigned int id )
```

Construct a new Snake Body Entity object.

**Parameters**

<i>position</i>	
-----------------	--

**10.87.2 Member Function Documentation**

### 10.87.2.1 getComponents()

```
const components::ComponentsMap & SnakeBodyEntity::getComponents (
    void ) const [override], [virtual], [noexcept]
```

Get the Components object.

Returns

const components::ComponentsMap&

Implements [shared::games::entity::IEntity](#).

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

- src/games/snake/entities/snake\_body/SnakeBodyEntity.hpp
- src/games/snake/entities/snake\_body/SnakeBodyEntity.cpp

## 10.88 SnakeGame Class Reference

Inheritance diagram for SnakeGame:

Collaboration diagram for SnakeGame:

### Public Member Functions

- [SnakeGame](#) ()  
*Construct a new Snake Game object.*
- [~SnakeGame](#) ()  
*Destroy a new Snake Game object.*
- void [compute](#) (DeltaTime dt) override  
*Compute the game based on the DeltaTime dt.*
- const [GameManifest](#) & [getManifest](#) () const noexcept override  
*Get the game manifest object.*
- const [Vector2u](#) [getSize](#) (void) const noexcept override  
*Get the Size object.*
- const entity::EntitiesMap & [getEntities](#) (void) const override  
*Get the Entities object.*
- const int [getScore](#) () const noexcept override  
*Get the score of the game.*
- const unsigned int [getFps](#) (void) const noexcept override  
*Get the Fps object.*
- void [moveSnake](#) ()  
*Move the snake.*
- void [updatePosition](#) ()  
*Update the position of the snake.*
- bool [hasHeadMoved](#) (auto it)  
*Check if the head has moved.*
- [Vector2f](#) [updateBodyPositions](#) (auto it)  
*Update the body positions.*
- void [updateTailPosition](#) ([Vector2f](#))  
*Update the tail position.*
- void [checkMapExit](#) (std::shared\_ptr< [SnakeHeadDisplayable](#) > head)  
*Check if the snake has gone outside of the map and replace at it at the opposite side.*
- void [increaseSnakeSize](#) ()  
*Increases snake score and length.*
- int [increaseDifficulty](#) (int score)  
*Increases the difficulty of the game.*

- bool [findDirection](#) (auto it)  
*Search for the direction of the snake.*
- [Vector2f updateHeadPosition](#) (auto it)  
*Update the head position.*
- void [updateHeadCollidablePosition](#) (auto it, [Vector2f](#) pos)  
*Update the head collidable position.*
- void [updateBodyCollidablePosition](#) (auto it, [Vector2f](#) pos)  
*Update the body collidable position.*
- void [updateTailCollidablePosition](#) (auto it, [Vector2f](#) pos)  
*Update the tail collidable position.*
- void [updateApplePosition](#) ()  
*Update the apple position.*
- bool [checkLose](#) ()  
*Check if the player has lost.*
- void [gameInit](#) ()  
*Init the game.*

## 10.88.1 Member Function Documentation

### 10.88.1.1 [checkLose\(\)](#)

```
bool SnakeGame::checkLose (
    void )
```

Check if the player has lost.

#### Returns

true  
false

### 10.88.1.2 [checkMapExit\(\)](#)

```
void SnakeGame::checkMapExit (
    std::shared_ptr< SnakeHeadDisplayable > head )
```

Check if the snake has gone outside of the map and replace at it at the opposite side.

#### Parameters

<a href="#">head</a>	
----------------------	--

### 10.88.1.3 [compute\(\)](#)

```
void SnakeGame::compute (
    DeltaTime dt ) [override], [virtual]
```

Compute the game based on the DeltaTime dt.

#### Parameters

<a href="#">dt</a>	
--------------------	--

Implements [shared::games::IGame](#).



#### 10.88.1.4 findDirection()

```
bool SnakeGame::findDirection (
    auto it )
```

Search for the direction of the snake.

##### Parameters

<i>it</i>	
-----------	--

##### Returns

true

false

#### 10.88.1.5 getEntities()

```
const entity::EntitiesMap & SnakeGame::getEntities (
    void ) const [override], [virtual]
```

Get the Entities object.

##### Returns

const entity::EntitiesMap&

Implements [shared::games::IGame](#).

#### 10.88.1.6 getFps()

```
const unsigned int SnakeGame::getFps (
    void ) const [override], [virtual], [noexcept]
```

Get the Fps object.

##### Returns

const unsigned int

Implements [shared::games::IGame](#).

#### 10.88.1.7 getManifest()

```
const GameManifest & SnakeGame::getManifest (
    void ) const [override], [virtual], [noexcept]
```

Get the game manifest object.

##### Returns

const GameManifest&

Implements [shared::games::IGame](#).

#### 10.88.1.8 getScore()

```
const int SnakeGame::getScore ( ) const [override], [virtual], [noexcept]
```

Get the score of the game.

**Returns**

The score of the game

Implements [shared::games::IGame](#).

**10.88.1.9 getSize()**

```
const Vector2u SnakeGame::getSize (
    void ) const [override], [virtual], [noexcept]
```

Get the Size object.

**Returns**

const Vector2u

Implements [shared::games::IGame](#).

**10.88.1.10 hasHeadMoved()**

```
bool SnakeGame::hasHeadMoved (
    auto it )
```

Check if the head has moved.

**Parameters**

<i>it</i>	
-----------	--

**Returns**

bool

**10.88.1.11 increaseDifficulty()**

```
int SnakeGame::increaseDifficulty (
    int score )
```

Increases the difficulty of the game.

**Parameters**

<i>score</i>	return _moveSpeed
--------------	-------------------

**10.88.1.12 updateBodyCollidablePosition()**

```
void SnakeGame::updateBodyCollidablePosition (
    auto it,
    Vector2f pos )
```

Update the body collidable position.

**Parameters**

<i>it</i>	
<i>pos</i>	

#### 10.88.1.13 updateBodyPositions()

```
Vector2f SnakeGame::updateBodyPositions (
    auto it )
```

Update the body positions.

##### Parameters

<i>it</i>	
-----------	--

##### Returns

The tail position after the update

#### 10.88.1.14 updateHeadCollidablePosition()

```
void SnakeGame::updateHeadCollidablePosition (
    auto it,
    Vector2f pos )
```

Update the head collidable position.

##### Parameters

<i>it</i>	
<i>pos</i>	

#### 10.88.1.15 updateHeadPosition()

```
Vector2f SnakeGame::updateHeadPosition (
    auto it )
```

Update the head position.

##### Parameters

<i>it</i>	
-----------	--

##### Returns

Vector2f

#### 10.88.1.16 updateTailCollidablePosition()

```
void SnakeGame::updateTailCollidablePosition (
    auto it,
    Vector2f pos )
```

Update the tail collidable position.

##### Parameters

<i>it</i>	
<i>pos</i>	

### 10.88.1.17 updateTailPosition()

```
void SnakeGame::updateTailPosition (
    Vector2f tailNewPosition )
```

Update the tail position.

#### Parameters

<i>position</i>	
-----------------	--

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

- src/games/snake/game/SnakeGame.hpp
- src/games/snake/game/SnakeGame.cpp

## 10.89 SnakeGameProvider Class Reference

Inheritance diagram for SnakeGameProvider:

Collaboration diagram for SnakeGameProvider:

### Public Member Functions

- const [shared::games::GameManifest](#) & [getManifest](#) () const noexcept override  
*Provides the game manifest.*
- std::shared\_ptr< [shared::games::IGame](#) > [createInstance](#) () override  
*Provides a new instance of the game.*

### 10.89.1 Member Function Documentation

#### 10.89.1.1 createInstance()

```
std::shared_ptr< IGame > SnakeGameProvider::createInstance ( ) [override], [virtual]
```

Provides a new instance of the game.

#### Returns

Created game instance

Implements [shared::games::IGameProvider](#).

#### 10.89.1.2 getManifest()

```
const GameManifest & SnakeGameProvider::getManifest ( ) const [override], [virtual], [noexcept]
```

Provides the game manifest.

#### Returns

Manifest of current game

Implements [shared::games::IGameProvider](#).

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

- src/games/snake/SnakeGameProvider.hpp
- src/games/snake/SnakeGameProvider.cpp

## 10.90 SnakeHeadCollidable Class Reference

Inheritance diagram for SnakeHeadCollidable:

Collaboration diagram for SnakeHeadCollidable:

### Public Member Functions

- [SnakeHeadCollidable](#) (const [shared::games::entity::IEntity](#) &entity)  
*Construct a new SnakeHead Collidable object.*
- [~SnakeHeadCollidable](#) ()  
*Destroy the SnakeHead Collidable object.*
- const [shared::games::components::ComponentType](#) [getType](#) () const noexcept override  
*Get the Type object.*
- const [shared::games::entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- void [setPosition](#) ([Vector2f](#) pos) noexcept
- [Vector2f](#) & [getPosition](#) (void) noexcept override  
*Get position of the entity (tiles)*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get size of the entity (tiles)*
- bool [getLose](#) (void) noexcept
- void [onCollide](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx, std::shared\_ptr< [shared::games::components::ICollidableComp](#) > target) override  
*On collide event handler for the component.*

### 10.90.1 Constructor & Destructor Documentation

#### 10.90.1.1 SnakeHeadCollidable()

```
SnakeHeadCollidable::SnakeHeadCollidable (
    const shared::games::entity::IEntity & entity ) [explicit]
```

Construct a new SnakeHead Collidable object.

Parameters

<a href="#">entity</a>	
------------------------	--

### 10.90.2 Member Function Documentation

#### 10.90.2.1 getEntity()

```
const shared::games::entity::IEntity & SnakeHeadCollidable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

Returns

const [shared::games::entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.90.2.2 `getType()`

```
const shared::games::components::ComponentType SnakeHeadCollidable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

#### Returns

const shared::games::components::ComponentType

Implements [shared::games::components::IComponent](#).

### 10.90.2.3 `onCollide()`

```
void SnakeHeadCollidable::onCollide (
    std::shared_ptr< shared::games::IGame > ctx,
    std::shared_ptr< shared::games::components::ICollidableComponent > target ) [override],
[virtual]
```

On collide event handler for the component.

#### Parameters

<i>ctx</i>	Context of the game
<i>target</i>	Target entity

Implements [shared::games::components::ICollidableComponent](#).

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

- src/games/snake/entities/snake\_head/components/SnakeHeadCollidable.hpp
- src/games/snake/entities/snake\_head/components/SnakeHeadCollidable.cpp

## 10.91 SnakeHeadDisplayable Class Reference

Inheritance diagram for SnakeHeadDisplayable:

Collaboration diagram for SnakeHeadDisplayable:

### Public Member Functions

- [SnakeHeadDisplayable](#) (const [entity::IEntity](#) &entity)  
*Construct a new Snake Head Displayable object.*
- [~SnakeHeadDisplayable](#) ()  
*Destroy a new Snake Head Keyboard object.*
- const components::ComponentType [getType](#) () const noexcept override  
*Get the Type object.*
- const [entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get the Size object.*
- unsigned int & [getZIndex](#) (void) noexcept override  
*Get the ZIndex object.*
- components::TextureProps & [getTextureProps](#) (void) noexcept override  
*Get the TextureProps object.*
- void [onMousePress](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*handle the mouse press event*
- void [onMouseHover](#) (std::shared\_ptr< [IGame](#) > ctx) override

- handle the mouse hover event*
- void [onMouseRelease](#) (std::shared\_ptr< [IGame](#) > ctx) override
- handle the mouse release event*
- void [setPosition](#) ([Vector2f](#) pos) noexcept
- Set the Position object.*
- void [setOldPosition](#) ([Vector2f](#) pos) noexcept
- Set the Old Position object.*
- [Vector2f](#) & [getPosition](#) (void) noexcept override
- Get the Position object.*
- [Vector2f](#) & [getOldPosition](#) (void) noexcept
- Get the Old Position object.*

## Public Attributes

- [Vector2f](#) [\\_position](#)
- [Vector2f](#) [\\_oldPosition](#)
- [components::TextureProps](#) [\\_textureProps](#)

## 10.91.1 Constructor & Destructor Documentation

### 10.91.1.1 SnakeHeadDisplayable()

`SnakeHeadDisplayable::SnakeHeadDisplayable (`  
     const [entity::IEntity](#) & *entity* ) [explicit]  
`)`  
Construct a new Snake Head Displayable object.

#### Parameters

<i>entity</i>	
---------------	--

## 10.91.2 Member Function Documentation

### 10.91.2.1 getEntity()

`const entity::IEntity & SnakeHeadDisplayable::getEntity (`  
     void ) [override], [virtual], [noexcept]  
`)`  
Get the entity object.

#### Returns

const [entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.91.2.2 getOldPosition()

`Vector2f & SnakeHeadDisplayable::getOldPosition (`  
     void ) [noexcept]  
`)`  
Get the Old Position object.

#### Returns

[Vector2f](#)&

### 10.91.2.3 getPosition()

```
Vector2f & SnakeHeadDisplayable::getPosition (
    void ) [override], [virtual], [noexcept]
```

Get the Position object.

Returns

Vector2f&

Implements [shared::games::components::IPositionableComponent](#).

### 10.91.2.4 getSize()

```
Vector2u & SnakeHeadDisplayable::getSize (
    void ) [override], [virtual], [noexcept]
```

Get the Size object.

Returns

const Vector2u&

Implements [shared::games::components::IPositionableComponent](#).

### 10.91.2.5 getTextureProps()

```
components::TextureProps & SnakeHeadDisplayable::getTextureProps (
    void ) [override], [virtual], [noexcept]
```

Get the TextureProps object.

Returns

components::TextureProps&

Implements [shared::games::components::ITextureComponent](#).

### 10.91.2.6 getType()

```
const components::ComponentType SnakeHeadDisplayable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

Returns

const components::ComponentType

Implements [shared::games::components::IComponent](#).

### 10.91.2.7 getZIndex()

```
unsigned int & SnakeHeadDisplayable::getZIndex (
    void ) [override], [virtual], [noexcept]
```

Get the ZIndex object.

Returns

unsigned int&

Implements [shared::games::components::IDisplayableComponent](#).



### 10.91.2.8 onMouseHover()

```
void SnakeHeadDisplayable::onMouseHover (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse hover event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.91.2.9 onMousePress()

```
void SnakeHeadDisplayable::onMousePress (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse press event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.91.2.10 onMouseRelease()

```
void SnakeHeadDisplayable::onMouseRelease (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse release event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.91.2.11 setOldPosition()

```
void SnakeHeadDisplayable::setOldPosition (
    Vector2f pos ) [noexcept]
```

Set the Old Position object.

#### Parameters

<i>pos</i>	
------------	--

**Returns**

void

**10.91.2.12 setPosition()**

```
void SnakeHeadDisplayable::setPosition (
    Vector2f pos ) [noexcept]
```

Set the Position object.

**Parameters**

<i>pos</i>	
------------	--

**Returns**

void

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

- src/games/snake/entities/snake\_head/components/SnakeHeadDisplayable.hpp
- src/games/snake/entities/snake\_head/components/SnakeHeadDisplayable.cpp

**10.92 SnakeHeadEntity Class Reference**

Inheritance diagram for SnakeHeadEntity:

Collaboration diagram for SnakeHeadEntity:

**Public Member Functions**

- [SnakeHeadEntity](#) ()  
*Construct a new Snake Head Entity object.*
- [~SnakeHeadEntity](#) ()  
*Destroy a new Snake Head Entity object.*
- const components::ComponentsMap & [getComponents](#) (void) const noexcept override  
*Get the Components object.*

**10.92.1 Member Function Documentation****10.92.1.1 getComponents()**

```
const components::ComponentsMap & SnakeHeadEntity::getComponents (
    void ) const [override], [virtual], [noexcept]
```

Get the Components object.

**Returns**

const components::ComponentsMap&amp;

Implements [shared::games::entity::IEntity](#).

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

- src/games/snake/entities/snake\_head/SnakeHeadEntity.hpp
- src/games/snake/entities/snake\_head/SnakeHeadEntity.cpp

## 10.93 SnakeHeadKeyboard Class Reference

Inheritance diagram for SnakeHeadKeyboard:

Collaboration diagram for SnakeHeadKeyboard:

### Public Types

- enum [Direction](#) { [UP](#) , [DOWN](#) , [LEFT](#) , [RIGHT](#) }
- Enum for the direction of the snake.*

### Public Member Functions

- [SnakeHeadKeyboard](#) (const [entity::IEntity](#) &entity)  
*Construct a new Snake Head Keyboard object.*
- [~SnakeHeadKeyboard](#) ()  
*Destroy a new Snake Head Keyboard object.*
- const components::ComponentType [getType](#) () const noexcept override  
*Get the Type object.*
- const [entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- void [onKeyPress](#) (std::shared\_ptr< [IGame](#) > ctx, [KeyData](#) keyData) override  
*handle the key press event*
- void [onKeyRelease](#) (std::shared\_ptr< [IGame](#) > ctx, [KeyData](#) key) override  
*handle the key release event*

### Public Attributes

- [Direction](#) \_direction

## 10.93.1 Constructor & Destructor Documentation

### 10.93.1.1 SnakeHeadKeyboard()

```
SnakeHeadKeyboard::SnakeHeadKeyboard (
    const entity::IEntity & entity ) [explicit]
```

Construct a new Snake Head Keyboard object.

Parameters

<a href="#">entity</a>	
------------------------	--

## 10.93.2 Member Function Documentation

### 10.93.2.1 getEntity()

```
const entity::IEntity & SnakeHeadKeyboard::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

**Returns**

const entity::IEntity&

Implements [shared::games::components::IComponent](#).

**10.93.2.2 getType()**

```
const components::ComponentType SnakeHeadKeyboard::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

**Returns**

const components::ComponentType

Implements [shared::games::components::IComponent](#).

**10.93.2.3 onKeyPress()**

```
void SnakeHeadKeyboard::onKeyPress (
    std::shared_ptr< IGame > ctx,
    KeyData keyData ) [override], [virtual]
```

handle the key press event

**Parameters**

<i>ctx</i>	
<i>keyData</i>	

**Returns**

void

Implements [shared::games::components::IKeyboardComponent](#).

**10.93.2.4 onKeyRelease()**

```
void SnakeHeadKeyboard::onKeyRelease (
    std::shared_ptr< IGame > ctx,
    KeyData key ) [override], [virtual]
```

handle the key release event

**Parameters**

<i>ctx</i>	
<i>key</i>	

**Returns**

void

Implements [shared::games::components::IKeyboardComponent](#).

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

- src/games/snake/entities/snake\_head/components/SnakeHeadKeyboard.hpp
- src/games/snake/entities/snake\_head/components/SnakeHeadKeyboard.cpp

## 10.94 SnakeTailCollidable Class Reference

Inheritance diagram for SnakeTailCollidable:

Collaboration diagram for SnakeTailCollidable:

### Public Member Functions

- [SnakeTailCollidable](#) (const [shared::games::entity::IEntity](#) &entity)  
*Construct a new SnakeTail Collidable object.*
- [~SnakeTailCollidable](#) ()  
*Destroy the SnakeTail Collidable object.*
- const [shared::games::components::ComponentType](#) [getType](#) () const noexcept override  
*Get the Type object.*
- const [shared::games::entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- void [setPosition](#) ([Vector2f](#) pos) noexcept
- [Vector2f](#) & [getPosition](#) (void) noexcept override  
*Get position of the entity (tiles)*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get size of the entity (tiles)*
- void [onCollide](#) (std::shared\_ptr< [shared::games::IGame](#) > ctx, std::shared\_ptr< [shared::games::components::ICollidableComp](#)  
> target) override  
*On collide event handler for the component.*

### 10.94.1 Constructor & Destructor Documentation

#### 10.94.1.1 SnakeTailCollidable()

```
SnakeTailCollidable::SnakeTailCollidable (
    const shared::games::entity::IEntity & entity ) [explicit]
```

Construct a new SnakeTail Collidable object.

#### Parameters

<i>entity</i>	
---------------	--

### 10.94.2 Member Function Documentation

#### 10.94.2.1 getEntity()

```
const shared::games::entity::IEntity & SnakeTailCollidable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

#### Returns

const [shared::games::entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.94.2.2 `getType()`

```
const shared::games::components::ComponentType SnakeTailCollidable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

#### Returns

const shared::games::components::ComponentType

Implements [shared::games::components::IComponent](#).

### 10.94.2.3 `onCollide()`

```
void SnakeTailCollidable::onCollide (
    std::shared_ptr< shared::games::IGame > ctx,
    std::shared_ptr< shared::games::components::ICollidableComponent > target ) [override],
[virtual]
```

On collide event handler for the component.

#### Parameters

<i>ctx</i>	Context of the game
<i>target</i>	Target entity

Implements [shared::games::components::ICollidableComponent](#).

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

- src/games/snake/entities/snake\_tail/components/SnakeTailCollidable.hpp
- src/games/snake/entities/snake\_tail/components/SnakeTailCollidable.cpp

## 10.95 SnakeTailDisplayable Class Reference

Inheritance diagram for SnakeTailDisplayable:

Collaboration diagram for SnakeTailDisplayable:

### Public Member Functions

- [SnakeTailDisplayable](#) (const [entity::IEntity](#) &entity)  
*Construct a new Snake Tail Displayable object.*
- [~SnakeTailDisplayable](#) ()  
*Destroy a new Snake Tail Keyboard object.*
- const components::ComponentType [getType](#) () const noexcept override  
*Get the Type object.*
- const [entity::IEntity](#) & [getEntity](#) () noexcept override  
*Get the entity object.*
- [Vector2u](#) & [getSize](#) (void) noexcept override  
*Get the Size object.*
- unsigned int & [getZIndex](#) (void) noexcept override  
*Get the ZIndex object.*
- components::TextureProps & [getTextureProps](#) (void) noexcept override  
*Get the TextureProps object.*
- void [onMousePress](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*handle the mouse press event*
- void [onMouseHover](#) (std::shared\_ptr< [IGame](#) > ctx) override

- handle the mouse hover event*
- void [onMouseRelease](#) (std::shared\_ptr< [IGame](#) > ctx) override
- handle the mouse release event*
- void [setPosition](#) ([Vector2f](#) pos) noexcept
- Set the Position object.*
- void [setOldPosition](#) ([Vector2f](#) pos) noexcept
- Set the Old Position object.*
- [Vector2f](#) & [getPosition](#) (void) noexcept override
- Get the Position object.*
- [Vector2f](#) & [getOldPosition](#) (void) noexcept
- Get the Old Position object.*

## Public Attributes

- [Vector2f](#) [\\_position](#)
- [Vector2f](#) [\\_oldPosition](#)
- [components::TextureProps](#) [\\_textureProps](#)

## 10.95.1 Constructor & Destructor Documentation

### 10.95.1.1 SnakeTailDisplayable()

```
SnakeTailDisplayable::SnakeTailDisplayable (
    const entity::IEntity & entity ) [explicit]
```

Construct a new Snake Tail Displayable object.

#### Parameters

<a href="#">entity</a>	
------------------------	--

## 10.95.2 Member Function Documentation

### 10.95.2.1 getEntity()

```
const entity::IEntity & SnakeTailDisplayable::getEntity (
    void ) [override], [virtual], [noexcept]
```

Get the entity object.

#### Returns

const [entity::IEntity](#)&

Implements [shared::games::components::IComponent](#).

### 10.95.2.2 getOldPosition()

```
Vector2f & SnakeTailDisplayable::getOldPosition (
    void ) [noexcept]
```

Get the Old Position object.

#### Returns

[Vector2f](#)&

### 10.95.2.3 getPosition()

```
Vector2f & SnakeTailDisplayable::getPosition (
    void ) [override], [virtual], [noexcept]
```

Get the Position object.

#### Returns

Vector2f&

Implements [shared::games::components::IPositionableComponent](#).

### 10.95.2.4 getSize()

```
Vector2u & SnakeTailDisplayable::getSize (
    void ) [override], [virtual], [noexcept]
```

Get the Size object.

#### Returns

const Vector2u&

Implements [shared::games::components::IPositionableComponent](#).

### 10.95.2.5 getTextureProps()

```
components::TextureProps & SnakeTailDisplayable::getTextureProps (
    void ) [override], [virtual], [noexcept]
```

Get the TextureProps object.

#### Returns

components::TextureProps&

Implements [shared::games::components::ITextureComponent](#).

### 10.95.2.6 getType()

```
const components::ComponentType SnakeTailDisplayable::getType (
    void ) const [override], [virtual], [noexcept]
```

Get the Type object.

#### Returns

const components::ComponentType

Implements [shared::games::components::IComponent](#).

### 10.95.2.7 getZIndex()

```
unsigned int & SnakeTailDisplayable::getZIndex (
    void ) [override], [virtual], [noexcept]
```

Get the ZIndex object.

#### Returns

unsigned int&

Implements [shared::games::components::IDisplayableComponent](#).



### 10.95.2.8 onMouseHover()

```
void SnakeTailDisplayable::onMouseHover (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse hover event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.95.2.9 onMousePress()

```
void SnakeTailDisplayable::onMousePress (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse press event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.95.2.10 onMouseRelease()

```
void SnakeTailDisplayable::onMouseRelease (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

handle the mouse release event

#### Parameters

<i>ctx</i>	
------------	--

#### Returns

void

Implements [shared::games::components::IDisplayableComponent](#).

### 10.95.2.11 setOldPosition()

```
void SnakeTailDisplayable::setOldPosition (
    Vector2f pos ) [noexcept]
```

Set the Old Position object.

#### Parameters

<i>pos</i>	
------------	--

**Returns**

void

**10.95.2.12 setPosition()**

```
void SnakeTailDisplayable::setPosition (
    Vector2f pos ) [noexcept]
```

Set the Position object.

**Parameters**

<i>pos</i>	
------------	--

**Returns**

void

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

- src/games/snake/entities/snake\_tail/components/SnakeTailDisplayable.hpp
- src/games/snake/entities/snake\_tail/components/SnakeTailDisplayable.cpp

**10.96 SnakeTailEntity Class Reference**

Inheritance diagram for SnakeTailEntity:

Collaboration diagram for SnakeTailEntity:

**Public Member Functions**

- [SnakeTailEntity](#) ()  
*Construct a new Snake Tail Entity object.*
- [~SnakeTailEntity](#) ()  
*Destroy a new Snake Tail Entity object.*
- const components::ComponentsMap & [getComponents](#) (void) const noexcept override  
*Get the Components object.*

**10.96.1 Member Function Documentation****10.96.1.1 getComponents()**

```
const components::ComponentsMap & SnakeTailEntity::getComponents (
    void ) const [override], [virtual], [noexcept]
```

Get the Components object.

**Returns**

const components::ComponentsMap&amp;

Implements [shared::games::entity::IEntity](#).

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

- src/games/snake/entities/snake\_tail/SnakeTailEntity.hpp
- src/games/snake/entities/snake\_tail/SnakeTailEntity.cpp

## 10.97 SolarFoxEnemy Class Reference

Inheritance diagram for SolarFoxEnemy:

Collaboration diagram for SolarFoxEnemy:

### Public Member Functions

- **SolarFoxEnemy** ([shared::types::Vector2f](#) position, [shared::types::Vector2u](#) size, [shared::types::Vector2u](#) origin, [shared::types::Vector2i](#) direction)
- void **inverseDirection** ()
- [shared::types::Vector2i](#) **getDirection** () const
- void **move** ()
- void **incrementShootingStage** ()
- bool **isReadyToShoot** ()

### Additional Inherited Members

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

- src/games/solarfox/entities/enemy/SolarFoxEnemy.hpp
- src/games/solarfox/entities/enemy/SolarFoxEnemy.cpp

## 10.98 SolarFoxGame Class Reference

Inheritance diagram for SolarFoxGame:

Collaboration diagram for SolarFoxGame:

### Public Member Functions

- void [compute](#) (DeltaTime dt) override  
*Compute the game each tick of the program.*
- const [GameManifest](#) & [getManifest](#) (void) const noexcept override  
*Manifest with informations of the game.*
- const [Vector2u](#) [getSize](#) (void) const noexcept override  
*Number of tiles that represent the game Tile size is managed by the renderer.*
- const entity::EntitiesMap & [getEntities](#) (void) const override  
*Get map of entities.*
- const unsigned int [getFps](#) () const noexcept override  
*Get fps of the game.*
- const int [getScore](#) () const noexcept  
*Get the score of the game.*
- void **addProjectile** (ProjectileType type, [shared::types::Vector2f](#) position, [shared::types::Vector2f](#) direction)

### 10.98.1 Member Function Documentation

#### 10.98.1.1 compute()

```
void SolarFoxGame::compute (
    DeltaTime dt ) [override], [virtual]
```

Compute the game each tick of the program.

#### Parameters

<i>dt</i>	Time since last tick (Time in milliseconds)
-----------	---

Implements [shared::games::IGame](#).

#### 10.98.1.2 getEntities()

```
const entity::EntitiesMap & SolarFoxGame::getEntities (
    void ) const [override], [virtual]
```

Get map of entities.

Returns

Entities map of the game

Implements [shared::games::IGame](#).

#### 10.98.1.3 getFps()

```
const unsigned int SolarFoxGame::getFps ( ) const [override], [virtual], [noexcept]
```

Get fps of the game.

Returns

The number of frame per seconds of the game

Implements [shared::games::IGame](#).

#### 10.98.1.4 getManifest()

```
const GameManifest & SolarFoxGame::getManifest (
    void ) const [override], [virtual], [noexcept]
```

Manifest with informations of the game.

Returns

Manifest of the game

Implements [shared::games::IGame](#).

#### 10.98.1.5 getScore()

```
const int SolarFoxGame::getScore ( ) const [virtual], [noexcept]
```

Get the score of the game.

Returns

The score of the game

Implements [shared::games::IGame](#).

#### 10.98.1.6 getSize()

```
const Vector2u SolarFoxGame::getSize (
    void ) const [override], [virtual], [noexcept]
```

Number of tiles that represent the game Tile size is managed by the renderer.

Returns

The number of tiles of the game

Implements [shared::games::IGame](#).

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

- src/games/solarfox/game/SolarFoxGame.hpp
- src/games/solarfox/game/SolarFoxGame.cpp

## 10.99 SolarFoxPlayer Class Reference

Inheritance diagram for SolarFoxPlayer:

Collaboration diagram for SolarFoxPlayer:

### Public Member Functions

- bool **isShooting** () const
- bool **isDestroyed** () const

### Additional Inherited Members

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

- src/games/solarfox/entities/player/SolarFoxPlayer.hpp
- src/games/solarfox/entities/player/SolarFoxPlayer.cpp

## 10.100 SolarFoxPlayerCollidable Class Reference

Inheritance diagram for SolarFoxPlayerCollidable:

Collaboration diagram for SolarFoxPlayerCollidable:

### Public Member Functions

- **SolarFoxPlayerCollidable** ([shared::types::Vector2f](#) &position, [entity::IEntity](#) &entity)
- void **onCollide** (std::shared\_ptr< [IGame](#) > ctx, std::shared\_ptr< [ICollidableComponent](#) > target)
- bool **isDestroyed** () const

### Additional Inherited Members

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

- src/games/solarfox/entities/player/components/SolarFoxPlayerCollidable.hpp
- src/games/solarfox/entities/player/components/SolarFoxPlayerCollidable.cpp

## 10.101 SolarFoxPlayerKeyboard Class Reference

Inheritance diagram for SolarFoxPlayerKeyboard:

Collaboration diagram for SolarFoxPlayerKeyboard:

### Public Member Functions

- **SolarFoxPlayerKeyboard** ([entity::IEntity](#) &entity)
- void **onKeyPress** (std::shared\_ptr< [IGame](#) > ctx, [components::IKeyboardComponent::KeyData](#) keyData) override  
*On key pressed event handler for the entity.*
- void **onKeyRelease** (std::shared\_ptr< [IGame](#) > ctx, [components::IKeyboardComponent::KeyData](#) keyData) override  
*On key release event handler for the entity.*
- [components::IKeyboardComponent::ArrowCode](#) **getLastDirection** () const
- bool **isBoost** () const
- bool **isShooting** () const

### Additional Inherited Members

#### 10.101.1 Member Function Documentation

### 10.101.1.1 onKeyPress()

```
void SolarFoxPlayerKeyboard::onKeyPress (
    std::shared_ptr< IGame > ctx,
    components::IKeyboardComponent::KeyData keyData ) [override], [virtual]
```

On key pressed event handler for the entity.

#### Parameters

<i>ctx</i>	Context of the game
<i>keyData</i>	Key data of key pressed

Implements [shared::games::components::IKeyboardComponent](#).

### 10.101.1.2 onKeyRelease()

```
void SolarFoxPlayerKeyboard::onKeyRelease (
    std::shared_ptr< IGame > ctx,
    components::IKeyboardComponent::KeyData keyData ) [override], [virtual]
```

On key release event handler for the entity.

#### Parameters

<i>ctx</i>	Context of the game
<i>keyData</i>	Key data of key released

Implements [shared::games::components::IKeyboardComponent](#).

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

- `src/games/solarfox/entities/player/components/SolarFoxPlayerKeyboard.hpp`
- `src/games/solarfox/entities/player/components/SolarFoxPlayerKeyboard.cpp`

## 10.102 SolarFoxPowerup Class Reference

Inheritance diagram for SolarFoxPowerup:

Collaboration diagram for SolarFoxPowerup:

### Public Member Functions

- **SolarFoxPowerup** ([Vector2f](#) pos, PowerupType type)
- PowerupType **getPowerupType** () const

### Additional Inherited Members

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

- `src/games/solarfox/entities/powerup/SolarFoxPowerup.hpp`
- `src/games/solarfox/entities/powerup/SolarFoxPowerup.cpp`

## 10.103 SolarFoxPowerupCollidable Class Reference

Inheritance diagram for SolarFoxPowerupCollidable:

Collaboration diagram for SolarFoxPowerupCollidable:

## Public Member Functions

- **SolarFoxPowerupCollidable** ([shared::types::Vector2f](#) position, [entity::IEntity](#) &entity, PowerupType type)
- void **onCollide** (std::shared\_ptr< [IGame](#) > ctx, std::shared\_ptr< [ICollidableComponent](#) > target)
- bool **isDestroyed** () const
- PowerupType **getPowerupType** () const

## Additional Inherited Members

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

- src/games/solarfox/entities/powerup/components/SolarFoxPowerupCollidable.hpp
- src/games/solarfox/entities/powerup/components/SolarFoxPowerupCollidable.cpp

## 10.104 SolarFoxProjectile Class Reference

Inheritance diagram for SolarFoxProjectile:

Collaboration diagram for SolarFoxProjectile:

## Public Member Functions

- **SolarFoxProjectile** (ProjectileType type, [shared::types::Vector2f](#) position, [shared::types::Vector2f](#) direction)
- [shared::types::Vector2f](#) & **getDirection** ()
- [shared::types::Vector2f](#) & **getPosition** ()
- [shared::types::Vector2f](#) & **getStartingPosition** ()
- const ProjectileType **getType** () const
- void **moveProjectile** ()
- unsigned int **getProjectileTravelDistance** ()
- bool **isDestroyed** () const

## Additional Inherited Members

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

- src/games/solarfox/entities/projectile/SolarFoxProjectile.hpp
- src/games/solarfox/entities/projectile/SolarFoxProjectile.cpp

## 10.105 SolarFoxProjectileCollidable Class Reference

Inheritance diagram for SolarFoxProjectileCollidable:

Collaboration diagram for SolarFoxProjectileCollidable:

## Public Member Functions

- **SolarFoxProjectileCollidable** ([shared::types::Vector2f](#) &position, [entity::IEntity](#) &entity, ProjectileType type)
- void **onCollide** (std::shared\_ptr< [IGame](#) > ctx, std::shared\_ptr< [ICollidableComponent](#) > target)
- bool **isDestroyed** () const
- ProjectileType **getProjectileType** () const

## Additional Inherited Members

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

- src/games/solarfox/entities/projectile/components/SolarFoxProjectileCollidable.hpp
- src/games/solarfox/entities/projectile/components/SolarFoxProjectileCollidable.cpp

## 10.106 SolarFoxProvider Class Reference

Inheritance diagram for SolarFoxProvider:

Collaboration diagram for SolarFoxProvider:

### Public Member Functions

- const [GameManifest](#) & [getManifest](#) () const noexcept override  
*Provides the game manifest.*
- std::shared\_ptr< [IGame](#) > [createInstance](#) (void) override  
*Provides a new instance of the game.*

### 10.106.1 Member Function Documentation

#### 10.106.1.1 createInstance()

```
std::shared_ptr< IGame > SolarFoxProvider::createInstance (
    void ) [override], [virtual]
```

Provides a new instance of the game.

#### Returns

Created game instance

Implements [shared::games::IGameProvider](#).

#### 10.106.1.2 getManifest()

```
const GameManifest & SolarFoxProvider::getManifest ( ) const [override], [virtual], [noexcept]
```

Provides the game manifest.

#### Returns

Manifest of current game

Implements [shared::games::IGameProvider](#).

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

- src/games/solarfox/SolarFoxProvider.hpp
- src/games/solarfox/SolarFoxProvider.cpp

## 10.107 SolarFoxScore Class Reference

Inheritance diagram for SolarFoxScore:

Collaboration diagram for SolarFoxScore:

### Public Member Functions

- void **increaseScore** (std::size\_t score)
- void **decreaseScore** (std::size\_t score)
- std::size\_t **getScore** () const
- void **resetScore** ()

### Additional Inherited Members

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

- src/games/solarfox/entities/ui/SolarFoxScore.hpp
- src/games/solarfox/entities/ui/SolarFoxScore.cpp



## 10.108 Core::SoundMapProps Struct Reference

### Public Attributes

- `std::shared_ptr< ISound > graphicSound`
- `components::SoundState gameState`
- `ISound::SoundState graphicState`

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

- `src/core/Core.hpp`

## 10.109 TextComponent Class Reference

Inheritance diagram for TextComponent:

Collaboration diagram for TextComponent:

### Public Member Functions

- **TextComponent** (`shared::types::Vector2f` position, `shared::types::Vector2u` size, `entity::IEntity` &entity, unsigned int zIndex, `components::ITextComponent::TextProps` &textProps)
- `components::ITextComponent::TextProps` `getTextProps` () noexcept override  
*Get text props of the entity.*
- void `onMousePress` (`std::shared_ptr< IGame >` ctx) override  
*On click event handler for the entity.*
- void `onMouseRelease` (`std::shared_ptr< IGame >` ctx) override  
*On release event handler for the entity.*
- void `onMouseHover` (`std::shared_ptr< IGame >` ctx) override  
*On hover event handler for the entity.*

### Protected Attributes

- `components::ITextComponent::TextProps` & `_textProps`

### Additional Inherited Members

#### 10.109.1 Member Function Documentation

##### 10.109.1.1 `getTextProps()`

```
components::ITextComponent::TextProps TextComponent::getTextProps ( ) [override], [virtual], [noexcept]
```

Get text props of the entity.

Returns

text props

Implements `shared::games::components::ITextComponent`.

##### 10.109.1.2 `onMouseHover()`

```
void TextComponent::onMouseHover (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

On hover event handler for the entity.

**Parameters**

<code>ctx</code>	Context of the game
------------------	---------------------

Implements [shared::games::components::IDisplayableComponent](#).

**10.109.1.3 onMousePress()**

```
void TextComponent::onMousePress (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

On click event handler for the entity.

**Parameters**

<code>ctx</code>	Context of the game
------------------	---------------------

Implements [shared::games::components::IDisplayableComponent](#).

**10.109.1.4 onMouseRelease()**

```
void TextComponent::onMouseRelease (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

On release event handler for the entity.

**Parameters**

<code>ctx</code>	Context of the game
------------------	---------------------

Implements [shared::games::components::IDisplayableComponent](#).

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

- `src/games/abstracts/components/TextComponent.hpp`
- `src/games/abstracts/components/TextComponent.cpp`

## 10.110 shared::games::components::ITextComponent::TextFontProps Struct Reference

Font properties.

```
#include <ITextComponent.hpp>
```

**Public Attributes**

- `std::string path`  
*Path of the font.*
- `unsigned int size`  
*Size of the font.*

**10.110.1 Detailed Description**

Font properties.

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

- `common/games/components/ITextComponent.hpp`

## 10.111 shared::games::components::ITextComponent::TextProps Struct Reference

Text properties.

```
#include <ITextComponent.hpp>
```

Collaboration diagram for shared::games::components::ITextComponent::TextProps:

### Public Attributes

- `std::string content`  
*Content of the text.*
- `TextAlign align`  
*Horizontal alignment of the text.*
- `TextVerticalAlign verticalAlign`  
*Vertical alignment of the text.*
- `TextFontProps font`  
*Font of the text.*
- `types::Color color`  
*Color of the text.*

### 10.111.1 Detailed Description

Text properties.

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

- common/games/components/ITextComponent.hpp

## 10.112 shared::graphics::TextProps Struct Reference

Text properties.

```
#include <TextProps.hpp>
```

Collaboration diagram for shared::graphics::TextProps:

### Public Attributes

- `std::shared_ptr< IFont > font`  
*Font of the text.*
- `unsigned int fontSize`  
*Font size.*
- `std::string content`  
*Content of the text.*
- `TextAlign align`  
*Horizontal alignment of the text.*
- `TextVerticalAlign verticalAlign`  
*Vertical alignment of the text.*
- `types::Color color`  
*Color of the text.*
- `Vector2u size`  
*Size of the entity.*
- `Vector2f position`  
*Position of the entity.*

### 10.112.1 Detailed Description

Text properties.

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

- `common/graphics/types/TextProps.hpp`

## 10.113 TextureComponent Class Reference

Inheritance diagram for TextureComponent:

Collaboration diagram for TextureComponent:

### Public Member Functions

- **TextureComponent** ([shared::types::Vector2f](#) position, [shared::types::Vector2u](#) size, [entity::IEntity](#) &entity, unsigned int zIndex, [components::TextureProps](#) &textureProps)
- [components::TextureProps](#) & [getTextureProps](#) () noexcept override  
*Get texture properties.*
- void [onMouseDown](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*On click event handler for the entity.*
- void [onMouseRelease](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*On release event handler for the entity.*
- void [onMouseHover](#) (std::shared\_ptr< [IGame](#) > ctx) override  
*On hover event handler for the entity.*

### Protected Attributes

- [components::TextureProps](#) & `_textureProps`

### 10.113.1 Member Function Documentation

#### 10.113.1.1 [getTextureProps\(\)](#)

```
components::TextureProps & TextureComponent::getTextureProps ( ) [override], [virtual], [noexcept]
```

Get texture properties.

Returns

TextureProps &Texture properties

Implements [shared::games::components::ITextureComponent](#).

#### 10.113.1.2 [onMouseHover\(\)](#)

```
void TextureComponent::onMouseHover (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

On hover event handler for the entity.

Parameters

<code>ctx</code>	Context of the game
------------------	---------------------

Implements [shared::games::components::IDisplayableComponent](#).

### 10.113.1.3 onMousePress()

```
void TextureComponent::onMousePress (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

On click event handler for the entity.

#### Parameters

<code>ctx</code>	Context of the game
------------------	---------------------

Implements [shared::games::components::IDisplayableComponent](#).

### 10.113.1.4 onMouseRelease()

```
void TextureComponent::onMouseRelease (
    std::shared_ptr< IGame > ctx ) [override], [virtual]
```

On release event handler for the entity.

#### Parameters

<code>ctx</code>	Context of the game
------------------	---------------------

Implements [shared::games::components::IDisplayableComponent](#).

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

- `src/games/abstracts/components/TextureComponent.hpp`
- `src/games/abstracts/components/TextureComponent.cpp`

## 10.114 shared::games::components::TextureProps Struct Reference

Texture properties.

```
#include <ITextureComponent.hpp>
```

Collaboration diagram for `shared::games::components::TextureProps`:

### Public Attributes

- [TextureSources](#) `sources`  
*Sources of textures.*
- [Vector2u](#) `origin`  
*Size of the texture.*

### 10.114.1 Detailed Description

Texture properties.

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

- `common/games/components/ITextureComponent.hpp`

## 10.115 shared::graphics::TextureProps Struct Reference

Texture properties.

```
#include <TextureProps.hpp>
```

Collaboration diagram for `shared::graphics::TextureProps`:

## Public Attributes

- `std::shared_ptr< ITexture > texture`  
*Texture of the entity.*
- `Vector2f binTileSize`  
*Size of a binary tile.*
- `Vector2u origin`  
*Origin of the texture.*
- `Vector2u size`  
*Size of the entity.*
- `Vector2f position`  
*Position of the entity.*

### 10.115.1 Detailed Description

Texture properties.

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

- `common/graphics/types/TextureProps.hpp`

## 10.116 shared::games::components::TextureSources Struct Reference

Texture sources.

```
#include <ITextureComponent.hpp>
```

Collaboration diagram for `shared::games::components::TextureSources`:

## Public Attributes

- `const std::string ascii`  
*ASCII image representation path.*
- `const std::string bin`  
*Binary image path.*
- `Vector2f binTileSize`  
*Size of the binary tile.*

### 10.116.1 Detailed Description

Texture sources.

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

- `common/games/components/ITextureComponent.hpp`

## 10.117 shared::types::Vector< T > Struct Template Reference

`Vector` type.

```
#include <Vector.hpp>
```

## Public Member Functions

- `Vector (T x, T y)`  
*Construct a new `Vector` object.*

## Public Attributes

- [T \*x\*](#)  
*X value In this Graphical Project, it can be refer as :*
- [T \*y\*](#)  
*Y value In this Graphical Project, it can be refer as :*

### 10.117.1 Detailed Description

```
template<typename T>
struct shared::types::Vector< T >
```

[Vector](#) type.

#### Template Parameters

<i>T</i>	Type of the vector
----------	--------------------

### 10.117.2 Constructor & Destructor Documentation

#### 10.117.2.1 Vector()

```
template<typename T >
shared::types::Vector< T >::Vector (
    T x,
    T y ) [inline]
```

Construct a new [Vector](#) object.

#### Parameters

<i>x</i>	X value
<i>y</i>	Y value

### 10.117.3 Member Data Documentation

#### 10.117.3.1 *x*

```
template<typename T >
T shared::types::Vector< T >::x
```

X value In this Graphical Project, it can be refer as :

- Width
- Longitude

#### 10.117.3.2 *y*

```
template<typename T >
T shared::types::Vector< T >::y
```

Y value In this Graphical Project, it can be refer as :

- Height

- Latitude

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

- common/types/Vector.hpp

## 10.118 shared::graphics::events::WindowCloseEvent Class Reference

Inheritance diagram for shared::graphics::events::WindowCloseEvent:

Collaboration diagram for shared::graphics::events::WindowCloseEvent:

### Public Member Functions

- EventType [getType](#) () const noexcept override  
*Event type.*

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

- shared/events/WindowCloseEvent.hpp

## 10.119 shared::graphics::IWindow::WindowInitProps Struct Reference

Window initial properties.

```
#include <IWindow.hpp>
```

Collaboration diagram for shared::graphics::IWindow::WindowInitProps:

### Public Attributes

- [Vector2u](#) [size](#)  
*Initial size of the window.*
- [WindowMode](#) [mode](#)  
*Initial mode of the window.*
- unsigned int [fps](#)  
*Initial framerate of the window.*
- const std::string [title](#)  
*Initial title of the window.*
- const std::string [icon](#)  
*Initial icon of the window.*

### 10.119.1 Detailed Description

Window initial properties.

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

- common/graphics/IWindow.hpp

## 10.120 shared::graphics::events::WindowResizeEvent Class Reference

Inheritance diagram for shared::graphics::events::WindowResizeEvent:

Collaboration diagram for shared::graphics::events::WindowResizeEvent:

### Public Member Functions

- **WindowResizeEvent** ([types::Vector2u](#) newSize)
- EventType [getType](#) () const noexcept override  
*Event type.*
- const [shared::types::Vector2u](#) & [getNewSize](#) () const noexcept  
*Get the new window size.*



## Protected Attributes

- [shared::types::Vector2u](#) \_newSize

## 10.120.1 Member Function Documentation

### 10.120.1.1 getNewSize()

```
const shared::types::Vector2u& shared::graphics::events::WindowResizeEvent::getNewSize ( )  
const [inline], [noexcept]
```

Get the new window size.

#### Returns

New window size

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

- [shared/events/WindowResizeEvent.hpp](#)

