

Zappy architecture

Generated by Doxygen 1.10.0

Chapter 1

Jenkins CI/CD Pipeline for Zappy

This directory contains the configuration for a Jenkins CI/CD pipeline that automates building, testing, and reporting for the Zappy project.

1.1 Features

- **Multi-Job Pipeline:** Sequential jobs organized in branch-specific folders
 - Coding Style Check
 - Build
 - Tests
 - Dashboard
- **Comprehensive Testing:** Unit tests with coverage reports for server, [GUI](#), and AI components
- **Automated Reporting:** Visual dashboards and summary reports

1.2 Setup Instructions

1.2.1 Requirements

- Docker and Docker Compose
- Git access to the Zappy repository

1.2.2 Getting Started

1. Start Jenkins:

```
cd jenkins
docker-compose up -d
```

2. Access Jenkins:

- Open <http://localhost:8080> in your browser
- Jenkins is pre-configured with necessary plugins and jobs

3. Run the Pipeline:

- Navigate to the "Pull Branch from Zappy" job
- Provide the branch name (default is "main")
- Click "Build"

1.3 Troubleshooting

1.3.1 Plugin Issues

If you encounter issues with plugins:

1. Run the plugin diagnostics script inside the container:

```
docker exec zappipicaca /var/jenkins_home/plugin_diagnostics.sh
```
2. Verify the plugins are properly installed:

```
docker exec zappipicaca ls -la /var/jenkins_home/plugins/
```

1.3.2 Fork Test Issues

The pipeline automatically disables problematic fork tests that cause crashes in the Jenkins environment. This is handled by the `fix_fork_tests.sh` script.

1.3.3 Coverage Issues

If tests fail but you still want coverage reports, the pipeline uses `run_coverage_with_workaround.sh` to ensure coverage reports are generated even if some tests fail.

1.4 Custom Scripts

- **fix_fork_tests.sh**: Disables problematic fork tests in Jenkins
- **fix_audio_issues.sh**: Addresses audio driver issues in [GUI](#) tests
- **run_coverage_with_workaround.sh**: Generates coverage even when tests fail
- **pipeline_summary.sh**: Creates textual summary of pipeline execution
- **pipeline_visualization.sh**: Creates visual ASCII representation of pipeline
- **plugin_diagnostics.sh**: Diagnoses plugin installation issues
- **memory_diagnostics.sh**: Reports memory usage and leaks

1.5 Maintenance

1.5.1 Adding New Jobs

To add a new job to the pipeline:

1. Update the `job_dsl.groovy` file with your new job definition
2. Ensure proper downstream triggering for sequential execution
3. Rebuild the "Pull Branch from Zappy" job

1.5.2 Updating Plugins

To update or add plugins:

1. Add the plugin to `plugins.txt`
2. Rebuild the Docker image:

```
docker-compose down
docker-compose build --no-cache
docker-compose up -d
```

Chapter 2

README

2.1 ZAPPY

A multiplayer network strategy game where teams compete for supremacy!

[[LICENSE) "" "[Languages](https://img.shields.io/badge/Languages-C%2B%2B%20%7C%20C%20%7C%20Python-orange?style=for-the-badge)"]

2.1.1 About The Project

Zappy is an exciting network-based strategy game where multiple teams compete on a tile-based map filled with resources. The objective is strategic: be the first team to get **at least 6 players** to reach the **maximum elevation level**.

2.1.1.1 Key Features

- **Multiplayer Network Game** - Real-time competition between teams
- **Dynamic Tile Map** - Resource-rich environment for strategic gameplay
- **Team-Based Strategy** - Collaborate with teammates to achieve victory
- **Multiple Interfaces** - Server, GUI client, and AI bot components
- **Real-time Visualization** - Watch the action unfold with the GUI
- **AI Integration** - Develop and deploy intelligent bots

2.1.2 Architecture

The project consists of three main components:

```
Zappy
  Server - Core game engine and network management
  GUI Client - Real-time game visualization interface
  AI Bot - Intelligent automated players
```

2.1.2.1 Technologies Used

Component	Language	Framework/Libraries
Server	C	Custom networking
GUI	C++	Graphics libraries
AI Bot	Python	Socket programming

2.1.3 Quick Start

2.1.3.1 Prerequisites

Before running Zappy, ensure you have:

- **C/C++ Compiler** (gcc/g++)
- **Python 3.x**
- **Make** build system
- **PDF-LaTeX** (for documentation generation)

2.1.3.2 Installation

1. Clone the repository

```
git clone <repository-url>
cd zappy
```

2. Build all components

```
make
```

This will compile:

- `zappy_server` - The game server
- `zappy_gui` - The graphical interface
- `zappy_ai` - The AI bot

3. Run the game

Start the server:

```
./zappy_server -p <port> -x <width> -y <height> -n <team1> <team2> ... -c <nb_clients> -f <freq>
```

Launch the **GUI**:

```
./zappy_gui -p <port> -h <hostname>
```

Deploy AI team:

```
./zappy_ai -p <port> -n <team_name> -h <hostname>
```

2.1.4 Documentation

2.1.4.1 Docusaurus Documentation

Start the interactive documentation:

```
cd documentation/my-zappy-doc
npx docusaurus start
```

Troubleshooting: If you encounter npm error could not determine executable to run, run:

```
npm install --save-dev @docusaurus/types
```

2.1.4.2 PDF Documentation (Doxygen)

Generate comprehensive PDF documentation:

Important: Move the `my-zappy-doc` folder out of the repository before generation due to Unicode emoji conflicts.

```
./generateDoc.sh
```

Requirements: Ensure `pdf-latex` library is installed on your system.

2.1.5 Contributing

We follow a structured commit convention to maintain code quality and project organization.

2.1.5.1 Commit Convention

Format: [Gitmoji] : [Element/Module] : [MESSAGE]

- **Gitmoji:** Appropriate emoji for the modification type
- **Element/Module:** The component you modified
- **MESSAGE:** Detailed description of changes

2.1.5.2 Gitmoji Reference

Code Features

Emoji	Code	Usage
	:sparkles:	Introduce new features
	:recycle:	Refactor/update code
	:bug:	Fix a bug
	:poop:	Remove coding style errors or temporary fix
	:rotating_↩ light:	Fix compiling warnings
	:fire:	Remove code or files

Testing

Emoji	Code	Usage
	:white_check_↩ mark:	Add, update, or pass tests

Architecture

Emoji	Code	Usage
	:see_no_evil:	Add or update .gitignore files
	:construction_worker:	Add or update CI build system
	:building_↩ construction:	Make architectural changes
	:memo:	Add or update documentation

Pull Requests

Emoji	Code	Usage
	:tada:	Must be used for each PR created!
	↩ :lipstick_↩ :	Must be used for each PR merged!
	:rewind:	Must be used for each revert done!

2.1.6 Git Commands Reference

2.1.6.1 Commit Management

Modify commit message (before push):

```
git commit --amend -m "New commit message"
```

Modify commit message (after push):

```
git commit --amend -m "New commit message"
git push --force
```

2.1.6.2 File Management

Unstage accidentally added file (not yet pushed):

```
git restore --staged <file>
```

Remove file from commit (after commit):

```
git reset --soft HEAD~1
git restore --staged file-to-remove.txt
git commit -m "New commit message (without the file)"
```

2.1.7 Testing

Run the comprehensive test suite:

```
# Unit tests
make tests_run
```

```
# Functional tests
cd tests/functional
python3 Tester.py
```

Coverage reports are automatically generated in `coverage_report/`.

2.1.8 Jenkins CI/CD

This project includes a fully configured Jenkins pipeline to automate building, testing, and quality checking.

2.1.8.1 Getting Started with Jenkins

1. **Start the Jenkins container:**

```
make jenkins
```

2. **Access the Jenkins interface:**

- Open <http://localhost:8080> in your browser
- Login with credentials (check the `.env` file or ask a team member)

3. **Run the pipeline:**

- Navigate to the "Pull Branch from Zappy" job
- Enter your branch name (default is "main")
- Click "Build"

4. **Stop the Jenkins container:**

```
make jenkins_stop
docker-compose -f jenkins/docker-compose.yml down -v // to remove volumes
```

2.1.8.2 Pipeline Jobs

The CI/CD pipeline consists of the following sequential jobs:

1 Coding Style Check

- **Purpose:** Verifies adherence to Epitech coding standards
- **Components Checked:**
 - C coding style (`cStyleChecker.sh`)
 - C++ coding style (`cppStyleChecker.sh`)
 - Python coding style (`pythonStyleChecker.sh`)
- **Trigger:** Automatic on each commit or manual execution

2 Build

- **Purpose:** Compiles all project components
- **Steps:**
 - Build server (`make zappy_server`)
 - Build GUI (`make zappy_gui`)
 - Build AI (`make zappy_ai`)
 - Verify clean/rebuild works (`make clean,make fclean,make re`)
- **Trigger:** Automatic after successful style check

3 Tests

- **Purpose:** Runs comprehensive test suite with coverage
- **Features:**
 - Sets up testing environment (GUI tests, audio configuration)
 - Runs unit tests with coverage reporting
 - Fixes common testing issues automatically
- **Trigger:** Automatic after successful build

4 Dashboard

- **Purpose:** Generates reports and visualizations
- **Outputs:**
 - Pipeline summary report
 - Visual pipeline progress representation
 - Test results and coverage statistics
- **Trigger:** Automatic after tests (even on failure)

2.1.8.3 Troubleshooting

If the pipeline fails:

1. **Check the console output** for the failing job
2. **View artifact reports** for detailed error information
3. **Run specific diagnostic scripts:**

```
# From host machine
docker exec zappipicaca /var/jenkins_home/plugin_diagnostics.sh
docker exec zappipicaca /var/jenkins_home/memory_diagnostics.sh
```

2.1.8.4 Viewing Reports

- Access coverage reports and artifacts from the Jenkins job page
- Click on "Artifacts" in the left sidebar of a completed job
- Download and view generated reports locally

2.1.9 Team

Project developed by EPITECH students

- Elliott Tesnier
- Albane Merian
- Nolan Papa
- Matisse Marsac
- Alban Roussée
- Noa Roussière

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

action_queue_s	??
action_request_s	??
App.App	??
BoundingBox3D	??
Broadcaster.Broadcaster	??
buffer_s	??
CameraManager	??
CLI	??
CLI.CLI	??
Client	??
Color32	??
Utils.Colors	??
command_info_t	??
command_pf_s	??
Communication.Communication	??
zappy::structs::Config	??
zappy::structs::Egg	??
egg_s	??
Exception	
Exceptions.CLIParsingException	??
Exceptions::CLIHostException	??
Exceptions.CLIInvalidArgumentException	??
Exceptions.CLIInvalidArgumentException	??
Exceptions.CLIMachineException	??
Exceptions.CLIMissingArgumentException	??
Exceptions.CLIMissingArgumentException	??
Exceptions.CLINameException	??
Exceptions.CLIPortException	??
Exceptions.CLIPortException	??
Exceptions.CommunicationException	??
Exceptions.CommunicationHandshakeException	??
Exceptions.CommunicationInvalidResponseException	??
Exceptions.PlayerDead	??
Exceptions.SocketException	??
std::exception	
Exceptions.CLIParsingException	??
Exceptions::ModuleError	??
Exceptions::NetworkException	??
Exceptions::ConnectionFailedException	??
Exceptions::ConnectionTimeoutException	??

Exceptions::ReceiveException	??
Exceptions::SendException	??
Exceptions::SocketCreationException	??
FloatRect	??
game_s	??
graph_net_s	??
graphic_pf_s	??
GUI	??
Hash.Hash	??
Help	??
IAudio	??
Audio	??
MockAudio	??
ICommunication	??
Communication	??
MockCommunication	??
MockCommunication	??
IContainers	??
AContainers	??
Containers	??
IDisplay	??
MockDisplay	??
Raylib	??
ILoader	??
DLLoader< std::shared_ptr< IDisplay > >	??
DLLoader< T >	??
zappy::structs::Incantation	??
incantation_s	??
IntRect	??
zappy::structs::Inventory	??
inventory_s	??
IObserver	??
ConcreteObserver	??
GuiObserver	??
HUD	??
MockObserver	??
MockObserver	??
TestObserver	??
ISubject	??
Subject	??
GameInfos	??
MockGameInfos	??
TestObserver::TestableGameInfos	??
item_handler_t	??
IUIElement	??
AUIElement	??
Button	??
Checkbox	??
Image	??
ImageButton	??
Slider	??
Text	??
Logger.Logger	??
Map	??
MockMap	??

map_t	??
MockGUI	??
MockServer	??
RayLibEnc::ModelData	??
MsgHandler	??
network_s	??
OutputRedirector	??
params_s	??
Parser.Parser	??
Player.Player	??
zappy::structs::Player	??
player_s	??
zappy::gui::PlayerModelInfo	??
PlayerPositionState	??
PlayerRotationState	??
Ray3D	??
RayCollision3D	??
RayLibEnc	??
RelativePosition	??
server_s	??
Settings	??
Socket.Socket	??
std::streambuf	
OutputRedirector::NullBuffer	??
team_s	??
testing::Test	
CLITest	??
CameraManagerTest	??
ClientTest	??
CommunicationTest	??
ExceptionsTest	??
GameInfosAdditionalTest	??
GameInfosObserverTest	??
GameInfosTest	??
GuiObserverTest	??
IObserverTest	??
MsgHandlerTest	??
SubjectTest	??
TestCase.TestCase	??
unittest.TestCase	
test_hash.TestHash	??
test_cli.TestCLI	??
test_com.TestCommunication	??
test_player.TestPlayer	??
test_socket.TestSocket	??
zappy::structs::Tile	??
tiles_s	??
UIRelativePosition	??
unified_poll_s	??
Vector2f	??
Vector2i	??
Vector3f	??
zappy_s	??

Chapter 4

Class Index

4.1 Class List

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

AContainers	??
action_queue_s	??
action_request_s	??
App.App	??
Audio	??
AUIElement	??
BoundingBox3D	??
Broadcaster.Broadcaster	??
buffer_s	??
Button	??
CameraManager	??
CameraManagerTest	??
Checkbox	??
CLI	??
CLI.CLI	??
Client	??
ClientTest	??
Exceptions::CLIHostException	??
Exceptions.CLIInvalidArgumentException	??
Exceptions.CLIMachineException	??
Exceptions.CLIMissingArgumentException	??
Exceptions.CLINameException	??
Exceptions.CLIParsingException	??
EPITECH PROJECT, 2025 zappy File description: Exceptions	??
Exceptions.CLIPortException	??
CLITest	??
Color32	??
Utils.Colors	??
command_info_t	??
command_pf_s	??
Communication	??
Communication.Communication	??
Exceptions.CommunicationException	??
Exceptions.CommunicationHandshakeException	??
Exceptions.CommunicationInvalidResponseException	??
CommunicationTest	??
ConcreteObserver	??
zappy::structs::Config	??
Exceptions::ConnectionFailedException	??
Exceptions::ConnectionTimeoutException	??

Containers	??
DLLoader< T >	??
zappy::structs::Egg	??
egg_s	??
ExceptionsTest	??
FloatRect	??
game_s	??
GameInfos	??
GameInfosAdditionalTest	??
GameInfosObserverTest	??
GameInfosTest	??
graph_net_s	??
graphic_pf_s	??
GUI	??
GuiObserver	??
GuiObserverTest	??
Hash.Hash	??
Help	??
HUD	??
IAudio	??
ICommunication	??
IContainers	??
IDisplay	??
ILoader	??
Image	??
ImageButton	??
zappy::structs::Incantation	??
incantation_s	??
IntRect	??
zappy::structs::Inventory	??
inventory_s	??
IObserver	??
IObserverTest	??
ISubject	??
item_handler_t	??
IUIElement	??
Logger.Logger	??
Map	??
map_t	??
MockAudio	??
MockCommunication	??
MockDisplay	??
MockGameInfos	??
MockGUI	??
MockMap	??
MockObserver	??
MockServer	??
RayLibEnc::ModelData	??
Exceptions::ModuleError	??
MsgHandler	??
MsgHandlerTest	??
network_s	??
Exceptions::NetworkException	??
OutputRedirector::NullBuffer	??
OutputRedirector	??
params_s	??
Parser.Parser	??
Player.Player	??

zappy::structs::Player	??
player_s	??
Exceptions.PlayerDead	??
zappy::gui::PlayerModelInfo	??
PlayerPositionState	??
PlayerRotationState	??
Ray3D	??
RayCollision3D	??
Raylib	??
RayLibEnc	??
Exceptions::ReceiveException	??
RelativePosition	??
Exceptions::SendException	??
server_s	??
Settings	??
Slider	??
Socket.Socket	??
Exceptions::SocketCreationException	??
Exceptions.SocketException	??
Subject	??
SubjectTest	??
team_s	??
TestObserver::TestableGameInfos	??
TestCase.TestCase	??
test_cli.TestCLI	??
test_com.TestCommunication	??
test_hash.TestHash	??
TestObserver	??
test_player.TestPlayer	??
test_socket.TestSocket	??
Text	??
zappy::structs::Tile	??
tiles_s	??
UIRelativePosition	??
unified_poll_s	??
Vector2f	??
Vector2i	??
Vector3f	??
zappy_s	??

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

gui/src/IDisplay.hpp	??
gui/src/Audio/Audio.hpp	??
gui/src/Audio/IAudio.hpp	??
gui/src/CLI/CLI.hpp	??
gui/src/Client/Client.hpp	??
gui/src/Client/MsgHandler.hpp	??
gui/src/Communication/Communication.hpp	??
gui/src/Communication/ICommunication.hpp	??
gui/src/DLLoader/DLLoader.hpp	??
gui/src/DLLoader/ILoader.hpp	??
gui/src/DLLoader/LoaderType.hpp	??
gui/src/Exceptions/Exceptions.hpp	??
gui/src/Game/GameInfos.hpp	??
gui/src/Graphic/GUI.hpp	??
gui/src/Graphic/Map.hpp	??
gui/src/Graphic/Camera/CameraManager.hpp	??
gui/src/Graphic/HUD/HUD.hpp	??
gui/src/Graphic/HUD/Button/Button.hpp	??
gui/src/Graphic/HUD/Checkbox/Checkbox.hpp	??
gui/src/Graphic/HUD/Containers/AContainers.hpp	??
gui/src/Graphic/HUD/Containers/Containers.hpp	??
gui/src/Graphic/HUD/Containers/IContainers.hpp	??
gui/src/Graphic/HUD/Help/Help.hpp	??
gui/src/Graphic/HUD/Image/Image.hpp	??
gui/src/Graphic/HUD/ImageButton/ImageButton.hpp	??
gui/src/Graphic/HUD/Settings/Settings.hpp	??
gui/src/Graphic/HUD/Slider/Slider.hpp	??
gui/src/Graphic/HUD/Text/Text.hpp	??
gui/src/Graphic/HUD/UIElement/AUIElement.hpp	??
gui/src/Graphic/HUD/UIElement/UIElement.hpp	??
gui/src/Observer/GuiObserver.hpp	??
gui/src/Observer/IObserver.hpp	??
gui/src/Observer/ISubject.hpp	??
gui/src/Observer/Subject.hpp	??
gui/src/RayLib/Raylib.hpp	??
gui/src/RayLib/RaylibEnc/RayLibEnc.hpp	??
gui/src/Utils/Constants.hpp	??
gui/src/Utils/GamepadConstants.hpp	??
gui/src/Utils/HelpText.hpp	??
gui/src/Utils/InputType.hpp	??

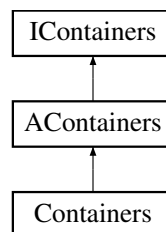
server/include/ algo.h	??
server/include/ buffer.h	??
server/include/ game.h	??
server/include/ my.h	??
server/include/ network.h	??
server/include/ zappy.h	??
server/lib/my/ my.h	??
server/src/network/ buffer.h	??
server/src/network/ network.h	??
tests/unit/server/ fake_malloc.h	??

Chapter 6

Class Documentation

6.1 AContainers Class Reference

Inheritance diagram for AContainers:



Public Member Functions

- **AContainers** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void [setPosition](#) (float x, float y) override
- void [setSize](#) (float width, float height) override
- [FloatRect](#) [getBounds](#) () const override
- bool [contains](#) (float x, float y) const override
- void [setVisible](#) (bool visible) override
- bool [isVisible](#) () const override
- void **setRelativePosition** (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [RelativePosition](#) [getRelativePosition](#) () const
- void **updatePositionFromRelative** ()
- float **getWidth** () const
- float **getHeight** () const

Public Member Functions inherited from [IDisplay](#)

- virtual void **draw** ()=0
- virtual void **update** ()=0

Protected Attributes

- std::shared_ptr< [IDisplay](#) > **_display**
- [FloatRect](#) **_bounds**
- [RelativePosition](#) **_relativePos**
- [Color32](#) **_backgroundColor**
- bool **_visible**
- bool **_hasBackground**

6.1.1 Member Function Documentation

6.1.1.1 contains()

```
bool AContainers::contains (
    float x,
    float y ) const [override], [virtual]
```

Implements [IContainers](#).

6.1.1.2 getBounds()

```
FloatRect AContainers::getBounds ( ) const [override], [virtual]
```

Implements [IContainers](#).

6.1.1.3 isVisible()

```
bool AContainers::isVisible ( ) const [override], [virtual]
```

Implements [IContainers](#).

6.1.1.4 setPosition()

```
void AContainers::setPosition (
    float x,
    float y ) [override], [virtual]
```

Implements [IContainers](#).

6.1.1.5 setSize()

```
void AContainers::setSize (
    float width,
    float height ) [override], [virtual]
```

Implements [IContainers](#).

6.1.1.6 setVisible()

```
void AContainers::setVisible (
    bool visible ) [override], [virtual]
```

Implements [IContainers](#).

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

- `gui/src/Graphic/HUD/Containers/AContainers.hpp`
- `gui/src/Graphic/HUD/Containers/AContainers.cpp`

6.2 action_queue_s Struct Reference

Public Attributes

- [action_request_t](#) * **head**
- [action_request_t](#) * **tail**
- int **count**

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

- `server/include/game.h`

6.3 action_request_s Struct Reference

Public Attributes

- char * **command**

- `time_t` **timestamp**
- `float` **time_limit**
- `action_priority_t` **priority**
- `player_t` * **player**
- `struct action_request_s` * **next**

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

- `server/include/game.h`

6.4 App.App Class Reference

Public Member Functions

- `__init__` (self, dict[str] config)
- `__del__` (self)
- `int` **create_new_player** (self)
- `run` (self)

Public Attributes

- **port**
- **name**
- **ip**
- **running**
- **is_main_process**
- **logger**
- **childs**

Protected Member Functions

- `_signal_handler` (self, signum, frame)
- `_cleanup_children` (self)
- `_child_signal_handler` (self, signum, frame)

Protected Attributes

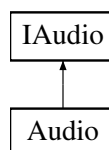
- `_signal_handler`
- `_child_signal_handler`

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

- `ai/src/App/App.py`

6.5 Audio Class Reference

Inheritance diagram for Audio:



Public Member Functions

- float [getSFXVolumeLevel](#) ()
- float [getMusicVolumeLevel](#) ()
- void [setSFXVolumeLevel](#) (float)
- void [setMusicVolumeLevel](#) (float)
- bool [loadSound](#) (const std::string &id, const std::string &filepath)
- void [playMainTheme](#) (float volume)
- void [playNextTheme](#) (float volume)
- void [playSound](#) (const std::string &id, float volume)
- void [stopSound](#) (const std::string &id)
- bool [isSoundPlaying](#) (const std::string &id) const
- void [setSoundLooping](#) (const std::string &id, bool looping)
- void [setSoundVolume](#) (const std::string &id, float volume)

Private Attributes

- std::vector< std::string > **_musicId** = {"main_theme", "main_theme2"}
- std::vector< std::string > **_sfxId** = {"click", "clickPlayer"}
- std::map< std::string, std::unique_ptr< sf::Music > > **_sounds**
- float **_levelSFX** = 75.f
- float **_levelMusic** = 50.f
- int **_themeIndex** = 0

6.5.1 Member Function Documentation

6.5.1.1 [getMusicVolumeLevel\(\)](#)

float [Audio::getMusicVolumeLevel](#) () [virtual]
Implements [IAudio](#).

6.5.1.2 [getSFXVolumeLevel\(\)](#)

float [Audio::getSFXVolumeLevel](#) () [virtual]
Implements [IAudio](#).

6.5.1.3 [isSoundPlaying\(\)](#)

bool [Audio::isSoundPlaying](#) (
 const std::string & id) const [virtual]
Implements [IAudio](#).

6.5.1.4 [loadSound\(\)](#)

bool [Audio::loadSound](#) (
 const std::string & id,
 const std::string & filepath) [virtual]
Implements [IAudio](#).

6.5.1.5 [playMainTheme\(\)](#)

void [Audio::playMainTheme](#) (
 float volume) [virtual]
Implements [IAudio](#).

6.5.1.6 [playNextTheme\(\)](#)

void [Audio::playNextTheme](#) (
 float volume) [virtual]
Implements [IAudio](#).

6.5.1.7 playSound()

```
void Audio::playSound (
    const std::string & id,
    float volume ) [virtual]
```

Implements [IAudio](#).

6.5.1.8 setMusicVolumeLevel()

```
void Audio::setMusicVolumeLevel (
    float level ) [virtual]
```

Implements [IAudio](#).

6.5.1.9 setSFXVolumeLevel()

```
void Audio::setSFXVolumeLevel (
    float level ) [virtual]
```

Implements [IAudio](#).

6.5.1.10 setSoundLooping()

```
void Audio::setSoundLooping (
    const std::string & id,
    bool looping ) [virtual]
```

Implements [IAudio](#).

6.5.1.11 setSoundVolume()

```
void Audio::setSoundVolume (
    const std::string & id,
    float volume ) [virtual]
```

Implements [IAudio](#).

6.5.1.12 stopSound()

```
void Audio::stopSound (
    const std::string & id ) [virtual]
```

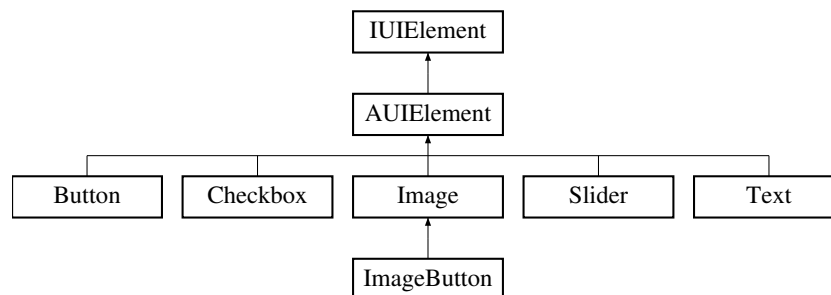
Implements [IAudio](#).

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

- gui/src/Audio/Audio.hpp
- gui/src/Audio/Audio.cpp

6.6 AUIElement Class Reference

Inheritance diagram for AUIElement:



Public Member Functions

- **AUIElement** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void [setPosition](#) (float x, float y) override
- [FloatRect](#) [getBounds](#) () const override
- bool [contains](#) (float x, float y) const override
- void [setVisible](#) (bool visible) override
- bool [isVisible](#) () const override
- virtual void [setSize](#) (float width, float height)
- void **setRelativePosition** (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [UIRelativePosition](#) [getRelativePosition](#) () const

Public Member Functions inherited from [IUIElement](#)

- virtual void **draw** ()=0
- virtual void **update** ()=0

Protected Attributes

- std::shared_ptr< [IDisplay](#) > **_display**
- [FloatRect](#) **_bounds**
- [UIRelativePosition](#) **_relativePos**
- bool **_visible**

6.6.1 Member Function Documentation

6.6.1.1 contains()

```
bool AUIElement::contains (
    float x,
    float y ) const [override], [virtual]
```

Implements [IUIElement](#).

6.6.1.2 getBounds()

```
FloatRect AUIElement::getBounds ( ) const [override], [virtual]
```

Implements [IUIElement](#).

6.6.1.3 isVisible()

```
bool AUIElement::isVisible ( ) const [override], [virtual]
```

Implements [IUIElement](#).

6.6.1.4 setPosition()

```
void AUIElement::setPosition (
    float x,
    float y ) [override], [virtual]
```

Implements [IUIElement](#).

6.6.1.5 setSize()

```
void AUIElement::setSize (
    float width,
    float height ) [virtual]
```

Implements [IUIElement](#).

6.6.1.6 setVisible()

```
void AUIElement::setVisible (
    bool visible ) [override], [virtual]
```

Implements [IUIElement](#).

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

- gui/src/Graphic/HUD/UIElement/AUIElement.hpp
- gui/src/Graphic/HUD/UIElement/AUIElement.cpp

6.7 BoundingBox3D Struct Reference

Public Attributes

- [Vector3f](#) min
- [Vector3f](#) max

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

- gui/src/IDisplay.hpp

6.8 Broadcaster.Broadcaster Class Reference

Public Member Functions

- None `__init__` (self, [Communication](#) com, str team)
- str `revealMessage` (self, str message)
- None `broadcastMessage` (self, str message)

Public Attributes

- com
- hasher

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

- ai/src/Broadcaster/Broadcaster.py

6.9 buffer_s Struct Reference

Public Attributes

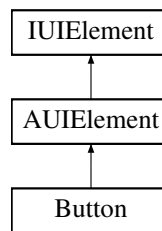
- char **data** [BUFFER_SIZE]
- int **head**
- int **tail**
- int **full**

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

- server/include/buffer.h
- server/src/network/buffer.h

6.10 Button Class Reference

Inheritance diagram for Button:



Public Member Functions

- **Button** (std::shared_ptr< [IDisplay](#) > display, std::shared_ptr< [IAudio](#) > audio, float x, float y, float width, float height, const std::string &text, std::function< void()> callback)
- void [draw](#) () override
- void [update](#) () override
- void **setText** (const std::string &text)
- std::string **getText** () const
- void **setCallback** (std::function< void()> callback)
- void **setColors** ([Color32](#) normal, [Color32](#) hover, [Color32](#) pressed, [Color32](#) textColor)
- void [setSize](#) (float width, float height) override

Public Member Functions inherited from [AUIElement](#)

- **AUIElement** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void [setPosition](#) (float x, float y) override
- [FloatRect](#) [getBounds](#) () const override
- bool [contains](#) (float x, float y) const override
- void [setVisible](#) (bool visible) override
- bool [isVisible](#) () const override
- void **setRelativePosition** (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [UIRelativePosition](#) **getRelativePosition** () const

Private Attributes

- std::string **_text**
- std::function< void()> **_callback**
- [Color32](#) **_normalColor**
- [Color32](#) **_hoverColor**
- [Color32](#) **_pressedColor**
- [Color32](#) **_textColor**
- bool **_isHovered**
- bool **_isPressed**
- std::shared_ptr< [IDisplay](#) > **_display**
- std::shared_ptr< [IAudio](#) > **_audio**

Additional Inherited Members

Protected Attributes inherited from [AUIElement](#)

- std::shared_ptr< [IDisplay](#) > **_display**
- [FloatRect](#) **_bounds**
- [UIRelativePosition](#) **_relativePos**
- bool **_visible**

6.10.1 Member Function Documentation

6.10.1.1 draw()

void Button::draw () [override], [virtual]

Implements [IUIElement](#).

6.10.1.2 setSize()

```
void Button::setSize (
    float width,
    float height ) [override], [virtual]
```

Reimplemented from [AUIElement](#).

6.10.1.3 update()

void Button::update () [override], [virtual]

Implements [IUIElement](#).

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

- gui/src/Graphic/HUD/Button/Button.hpp
- gui/src/Graphic/HUD/Button/Button.cpp

6.11 CameraManager Class Reference

Public Member Functions

- **CameraManager** (std::shared_ptr< [IDisplay](#) > display)
- void **updateCamera** (zappy::gui::CameraMode mode)
- void **updateCameraFreeMode** ()
- void **updateCameraTargetMode** ()
- void **updateCameraPlayerMode** ()
- void **setMapCenter** (const [Vector3f](#) ¢er)
- void **setMapSize** (int width, int height)
- void **setTargetDistance** (float distance)
- void **initTargetPositionFromCurrentCamera** ()
- void **setPlayerId** (int playerId)
- int **getPlayerId** () const
- void **setGameInfos** (std::shared_ptr< [GameInfos](#) > gameInfos)
- void **setMapInstance** (std::shared_ptr< [Map](#) > map)
- float **getCameraMovingSpeed** ()
- void **setCameraMovingSpeed** (float)
- float **getCameraRotaSpeed** ()
- void **setCameraRotaSpeed** (float)
- float **getCameraZoomSpeed** ()
- void **setCameraZoomSpeed** (float)
- [Vector3f](#) **calculatePlayerPosition** (const zappy::structs::Player &player)
- [Vector3f](#) **calculateCameraPosition** (const [Vector3f](#) &playerPos, float angleXZ)

Private Member Functions

- void **handlePlayerCameraMouseInput** ()

Private Attributes

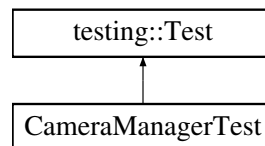
- float **_cameraMovingSpeed** = 15.0f
- float **_cameraRotaSpeed** = 2.0f
- float **_cameraZoomSpeed** = 120.0f
- std::shared_ptr< [IDisplay](#) > **_display**
- std::shared_ptr< [GameInfos](#) > **_gameInfos**
- std::shared_ptr< [Map](#) > **_map**
- [Vector3f](#) **_mapCenter**
- int **_mapWidth**
- int **_mapHeight**
- float **_targetDistance**
- float **_targetAngleXZ**
- float **_targetAngleY**
- bool **_isDragging**
- int **_playerId**
- float **_playerAngleXZ**
- bool **_isPlayerViewDragging**

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

- gui/src/Graphic/Camera/CameraManager.hpp
- gui/src/Graphic/Camera/CameraManager.cpp

6.12 CameraManagerTest Class Reference

Inheritance diagram for CameraManagerTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

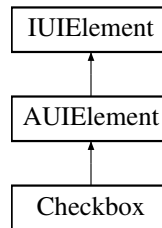
- std::unique_ptr< [CameraManager](#) > **cameraManager**
- std::shared_ptr< testing::NiceMock< [MockDisplay](#) > > **mockDisplay**
- std::shared_ptr< testing::NiceMock< [MockGameInfos](#) > > **mockGameInfos**
- std::shared_ptr< testing::NiceMock< [MockMap](#) > > **mockMap**
- std::vector< [zappy::structs::Player](#) > **testPlayersList**
- std::vector< [zappy::structs::Player](#) > **emptyPlayersList**

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

- tests/unit/gui/Camera_manager/Camera_manager_test.cpp

6.13 Checkbox Class Reference

Inheritance diagram for Checkbox:



Public Member Functions

- **Checkbox** (std::shared_ptr< [IDisplay](#) > display, std::shared_ptr< [IAudio](#) > audio, float x, float y, float width, float height, bool initialValue, std::function< void(bool)> callback)
- void [draw](#) () override
- void [update](#) () override
- void **setCallback** (std::function< void(bool)> callback)
- void **setValue** (bool value)
- bool **getValue** () const
- void **setColors** ([Color32](#) normalColor, [Color32](#) hoverColor, [Color32](#) pressedColor, [Color32](#) checkColor)
- void [setSize](#) (float width, float height) override

Public Member Functions inherited from [AUIElement](#)

- **AUIElement** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void [setPosition](#) (float x, float y) override
- [FloatRect](#) [getBounds](#) () const override
- bool [contains](#) (float x, float y) const override
- void [setVisible](#) (bool visible) override
- bool [isVisible](#) () const override
- void **setRelativePosition** (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [UIRelativePosition](#) [getRelativePosition](#) () const

Private Attributes

- bool **_value**
- std::function< void(bool)> **_callback**
- [Color32](#) **_normalColor**
- [Color32](#) **_hoverColor**
- [Color32](#) **_pressedColor**
- [Color32](#) **_checkColor**
- bool **_isHovered**
- bool **_isPressed**
- std::shared_ptr< [IDisplay](#) > **_display**
- std::shared_ptr< [IAudio](#) > **_audio**
- float **_checkboxSize**

Additional Inherited Members

Protected Attributes inherited from [AUIElement](#)

- std::shared_ptr< [IDisplay](#) > **_display**
- [FloatRect](#) **_bounds**
- [UIRelativePosition](#) **_relativePos**
- bool **_visible**

6.13.1 Member Function Documentation

6.13.1.1 draw()

void Checkbox::draw () [override], [virtual]
Implements [UIElement](#).

6.13.1.2 setSize()

void Checkbox::setSize (
float width,
float height) [override], [virtual]
Reimplemented from [AUIElement](#).

6.13.1.3 update()

void Checkbox::update () [override], [virtual]
Implements [UIElement](#).

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

- gui/src/Graphic/HUD/Checkbox/Checkbox.hpp
- gui/src/Graphic/HUD/Checkbox/Checkbox.cpp

6.14 CLI Class Reference

Public Member Functions

- **CLI** (int ac, const char *const *av)
- [zappy::structs::Config](#) **parseArguments** (int ac, const char *const *av) const

Private Member Functions

- bool **hasCorrectNumberOfArguments** (int ac) const
- int **parsePort** (const char *portStr) const
- std::string **parseHostname** (const char *hostnameStr) const
- void **validateConfig** (bool portFound, bool hostFound) const

Private Attributes

- int **_ac**
- const char *const * **_av**

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

- gui/src/CLI/CLI.hpp
- gui/src/CLI/CLI.cpp

6.15 CLI.CLI Class Reference

Public Member Functions

- **__init__** (self)
- **parse_args** (self, args)
- **parse_port** (self, port_str)
- **parse_name** (self, name)
- **parse_machine** (self, machine_str)
- **validate_config** (self, port_found, name_found)

Public Attributes

- **port**
- **name**
- **machine**

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

- ai/src/CLI/CLI.py

6.16 Client Class Reference

Public Member Functions

- **Client** (int ac, const char *const *av)
- void **tryToCreateGuiWithSharedLibInFolder** (const std::string &libPath)

Private Member Functions

- void **initialize** (int ac, const char *const *av)

Private Attributes

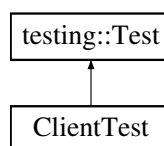
- [zappy::structs::Config](#) **_config**
- std::shared_ptr< [ICommunication](#) > **_communication**
- std::shared_ptr< [GameInfos](#) > **_gameInfos**
- std::unique_ptr< [MsgHandler](#) > **_msgHandler**
- std::shared_ptr< [GUI](#) > **_gui**
- std::shared_ptr< [GuiObserver](#) > **_guiObserver**

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

- gui/src/Client/Client.hpp
- gui/src/Client/Client.cpp

6.17 ClientTest Class Reference

Inheritance diagram for ClientTest:

**Protected Member Functions**

- void **SetUp** () override
- void **TearDown** () override
- char ** **createArgv** (const std::vector< std::string > &args)
- void **cleanupArgv** (char **argv, int argc)

Protected Attributes

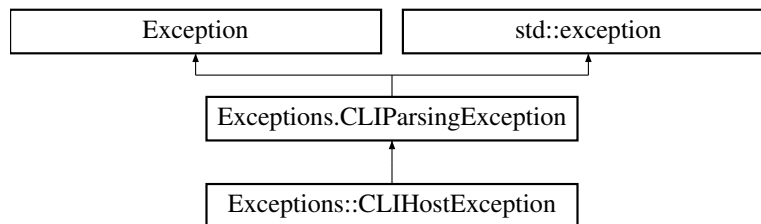
- std::stringstream **buffer**
- std::streambuf * **originalCout**

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

- tests/unit/gui/Client/Client_test.cpp

6.18 Exceptions::CLIHostException Class Reference

Inheritance diagram for Exceptions::CLIHostException:



Public Member Functions

- **CLIHostException** (const std::string &message)

Public Member Functions inherited from [Exceptions.CLIParsingException](#)

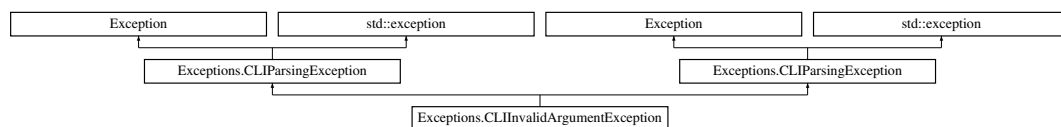
- `__init__` (self, str message)
- **CLIParsingException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.19 Exceptions.CLIInvalidArgumentException Class Reference

Inheritance diagram for Exceptions.CLIInvalidArgumentException:



Public Member Functions

- `__init__` (self, str message)
- **CLIInvalidArgumentException** (const std::string &message)

Public Member Functions inherited from [Exceptions.CLIParsingException](#)

- **CLIParsingException** (const std::string &message)
- const char * **what** () const noexcept override

6.19.1 Constructor & Destructor Documentation

6.19.1.1 `__init__()`

```

Exceptions.CLIInvalidArgumentException.__init__ (
    self,
    str message )
  
```

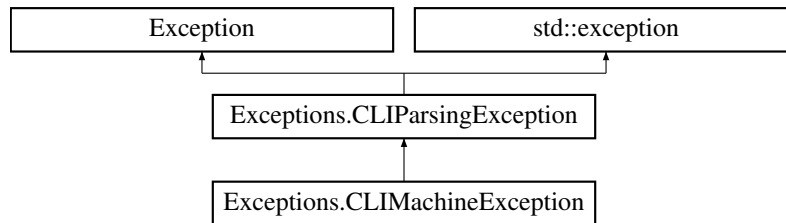
Reimplemented from [Exceptions.CLIParsingException](#).

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

- ai/src/Exceptions/Exceptions.py
- gui/src/Exceptions/Exceptions.hpp

6.20 Exceptions.CLIException Class Reference

Inheritance diagram for Exceptions.CLIException:



Public Member Functions

- [__init__](#) (self, str message)

Public Member Functions inherited from [Exceptions.CLIException](#)

- **CLIException** (const std::string &message)
- const char * **what** () const noexcept override

6.20.1 Constructor & Destructor Documentation

6.20.1.1 [__init__](#)()

```
Exceptions.CLIException.__init__ (
    self,
    str message )
```

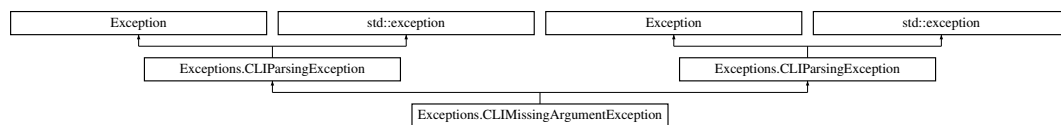
Reimplemented from [Exceptions.CLIException](#).

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

- ai/src/Exceptions/Exceptions.py

6.21 Exceptions.CLIMissingArgumentException Class Reference

Inheritance diagram for Exceptions.CLIMissingArgumentException:



Public Member Functions

- [__init__](#) (self, str message)
- **CLIMissingArgumentException** (const std::string &message)

Public Member Functions inherited from [Exceptions.CLIException](#)

- **CLIException** (const std::string &message)
- const char * **what** () const noexcept override

6.21.1 Constructor & Destructor Documentation

6.21.1.1 `__init__()`

```
Exceptions.CLIMissingArgumentException.__init__ (
    self,
    str message )
```

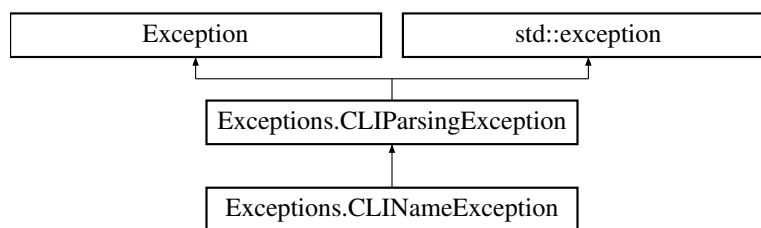
Reimplemented from [Exceptions.CLIParsingException](#).

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

- ai/src/Exceptions/Exceptions.py
- gui/src/Exceptions/Exceptions.hpp

6.22 Exceptions.CLINameException Class Reference

Inheritance diagram for Exceptions.CLINameException:



Public Member Functions

- [__init__](#) (self, str message)

Public Member Functions inherited from [Exceptions.CLIParsingException](#)

- **CLIParsingException** (const std::string &message)
- const char * **what** () const noexcept override

6.22.1 Constructor & Destructor Documentation

6.22.1.1 `__init__()`

```
Exceptions.CLINameException.__init__ (
    self,
    str message )
```

Reimplemented from [Exceptions.CLIParsingException](#).

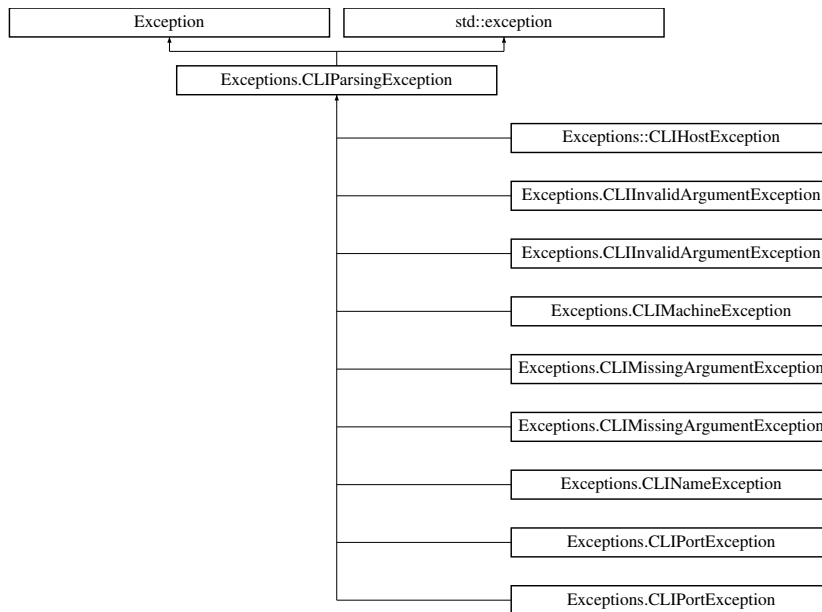
The documentation for this class was generated from the following file:

- ai/src/Exceptions/Exceptions.py

6.23 Exceptions.CLIParsingException Class Reference

EPITECH PROJECT, 2025 zappy File description: Exceptions.

Inheritance diagram for Exceptions.CLIParsingException:



Public Member Functions

- `__init__` (self, str message)
- **CLIParsingException** (const std::string &message)
- const char * **what** () const noexcept override

Private Attributes

- std::string **_message**

6.23.1 Detailed Description

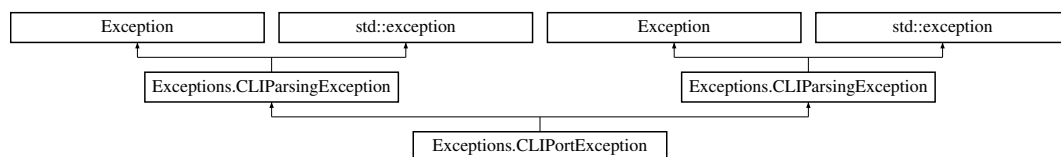
EPITECH PROJECT, 2025 zappy File description: Exceptions.

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

- ai/src/Exceptions/Exceptions.py
- gui/src/Exceptions/Exceptions.hpp

6.24 Exceptions.CLIPortException Class Reference

Inheritance diagram for Exceptions.CLIPortException:



Public Member Functions

- `__init__` (self, str message)
- **CLIPortException** (const std::string &message)

Public Member Functions inherited from [Exceptions.CLIParsingException](#)

- **CLIParsingException** (const std::string &message)
- const char * **what** () const noexcept override

6.24.1 Constructor & Destructor Documentation

6.24.1.1 `__init__()`

```
Exceptions.CLIPortException.__init__ (
    self,
    str message )
```

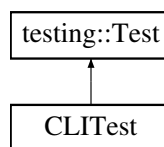
Reimplemented from [Exceptions.CLIParsingException](#).

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

- `ai/src/Exceptions/Exceptions.py`
- `gui/src/Exceptions/Exceptions.hpp`

6.25 CLITest Class Reference

Inheritance diagram for CLITest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override
- char ** **createArgv** (const std::vector< std::string > &args)
- void **cleanupArgv** (char **argv, int argc)

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

- `tests/unit/gui/CLI/CLI_test.cpp`

6.26 Color32 Struct Reference

Public Attributes

- unsigned char **r**
- unsigned char **g**
- unsigned char **b**
- unsigned char **a**

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

- `gui/src/IDisplay.hpp`

6.27 Utils.Colors Class Reference

Static Public Attributes

- str **BOLD** = "\033[1m"
- str **RED** = "\033[1m\033[31m"
- str **GREEN** = "\033[1m\033[32m"
- str **YELLOW** = "\033[1m\033[33m"
- str **BLUE** = "\033[1m\033[34m"
- str **MAGENTA** = "\033[1m\033[35m"
- str **CYAN** = "\033[1m\033[36m"

- str **WHITE** = "\033[1m\033[37m"
- str **RESET** = "\033[0m"

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

- `ai/src/Utils/Utils.py`

6.28 `command_info_t` Struct Reference

Public Attributes

- char * **command**
- float **base_time**
- action_priority_t **priority**
- int(* **handler**)([player_t](#) *, char *, [zappy_t](#) *)

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

- `server/include/zappy.h`

6.29 `command_pf_s` Struct Reference

Public Attributes

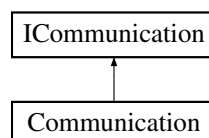
- char const * **flag**
- bool(* **checker**)(const char *, const char *, [params_t](#) *)

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

- `server/include/zappy.h`

6.30 Communication Class Reference

Inheritance diagram for Communication:



Public Member Functions

- **Communication** ([zappy::structs::Config](#) config)
- void [sendMessage](#) (const std::string &message) override
- bool [hasMessages](#) () const override
- std::string [popMessage](#) () override
- bool [isConnected](#) () const override
- void [disconnect](#) () override

Private Member Functions

- void **setupConnection** ()
- void **createSocket** ()
- void **connectToServer** ()
- void **setupNonBlocking** ()
- void **startCommunicationThread** ()

- void **communicationLoop** ()
- bool **handlePoll** ()
- void **processWrite** ()
- void **processRead** ()
- void **parseReceivedData** ()

Private Attributes

- [zappy::structs::Config](#) **_config**
- std::thread **_thread**
- std::mutex **_mutex**
- std::condition_variable **_cv**
- std::atomic< bool > **_running**
- std::atomic< bool > **_connected**
- std::queue< std::string > **_outgoingMessages**
- std::queue< std::string > **_incomingMessages**
- std::string **_receiveBuffer**
- std::string **_sendBuffer**
- int **_socket**
- struct pollfd **_pollfd**

Static Private Attributes

- static const int **BUFFER_SIZE** = 4096
- static const int **POLL_TIMEOUT** = 100
- static const char **MESSAGE_DELIMITER** = '\n'

6.30.1 Member Function Documentation

6.30.1.1 disconnect()

void Communication::disconnect () [override], [virtual]
Implements [ICommunication](#).

6.30.1.2 hasMessages()

bool Communication::hasMessages () const [override], [virtual]
Implements [ICommunication](#).

6.30.1.3 isConnected()

bool Communication::isConnected () const [override], [virtual]
Implements [ICommunication](#).

6.30.1.4 popMessage()

std::string Communication::popMessage () [override], [virtual]
Implements [ICommunication](#).

6.30.1.5 sendMessage()

void Communication::sendMessage (
const std::string & message) [override], [virtual]
Implements [ICommunication](#).

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

- gui/src/Communication/Communication.hpp
- gui/src/Communication/Communication.cpp

6.31 Communication.Communication Class Reference

Public Member Functions

- **__init__** (self, str name, str host, int port)
- **__del__** (self)
- None **stopLoop** (self)
- None **loop** (self)
- dict[str, int]|None **tryGetInventory** (self, str response)
- list[dict[str, int]]|None **tryGetLook** (self, str response)
- str **handleResponse** (self, str response)
- str **receiveData** (self)
- None **receive** (self)
- dict[str, int] **getInventory** (self)
- list[dict[str, int]] **getLook** (self)
- int **lenMessageQueue** (self)
- bool **hasMessages** (self)
- tuple[int, str] **getLastMessage** (self)
- int **lenResponseQueue** (self)
- bool **hasResponses** (self)
- None **addResponse** (self, str response)
- str **getLastResponse** (self)
- int **lenPendingQueue** (self)
- bool **hasPendingCommands** (self)
- int **lenRequestQueue** (self)
- bool **playerIsDead** (self)
- **connectToServer** (self)
- None **sendCommand** (self, str message)
- **sendForward** (self)
- **sendRight** (self)
- **sendLeft** (self)
- None **sendLook** (self)
- None **sendInventory** (self)
- **sendBroadcast** (self, str message)
- None **sendGetConnectNbr** (self)
- **sendFork** (self)
- **sendEject** (self)
- **sendTakeObject** (self, str object_name)
- **sendSetObject** (self, str object_name)
- **sendIncantation** (self)

Public Attributes

- **name**
- **host**
- **port**
- **socket**
- **mutex**
- **logger**
- **playerDead**
- **lastInventory**
- **lastLook**
- **responseBuffer**
- **messageQueue**
- **responseQueue**
- **pendingQueue**

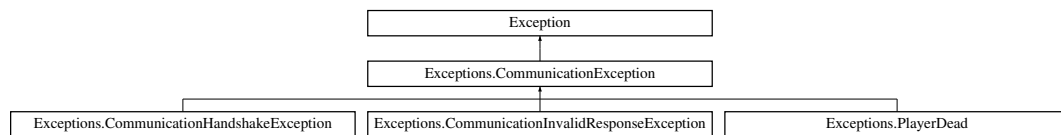
- `requestQueue`

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

- `ai/src/Communication/Communication.py`

6.32 Exceptions.CommunicationException Class Reference

Inheritance diagram for `Exceptions.CommunicationException`:



Public Member Functions

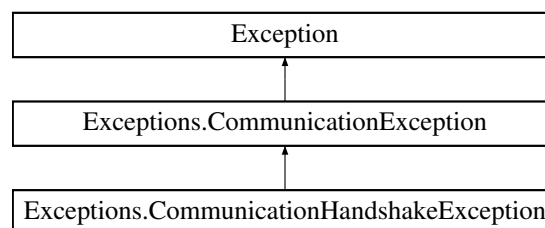
- `__init__` (self, str message)

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

- `ai/src/Exceptions/Exceptions.py`

6.33 Exceptions.CommunicationHandshakeException Class Reference

Inheritance diagram for `Exceptions.CommunicationHandshakeException`:



Public Member Functions

- `__init__` (self, str message)

6.33.1 Constructor & Destructor Documentation

6.33.1.1 `__init__`()

```
Exceptions.CommunicationHandshakeException.__init__ (
    self,
    str message )
```

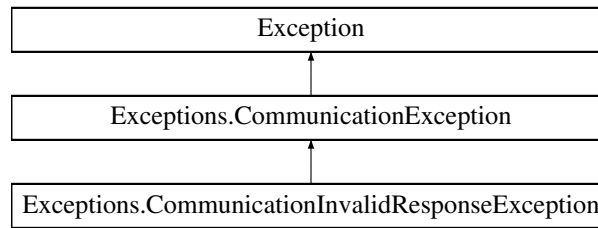
Reimplemented from [Exceptions.CommunicationException](#).

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

- `ai/src/Exceptions/Exceptions.py`

6.34 Exceptions.CommunicationInvalidResponseException Class Reference

Inheritance diagram for `Exceptions.CommunicationInvalidResponseException`:



Public Member Functions

- [__init__](#) (self, str message)

6.34.1 Constructor & Destructor Documentation

6.34.1.1 __init__()

```
Exceptions.CommunicationInvalidResponseException.__init__ (
    self,
    str message )
```

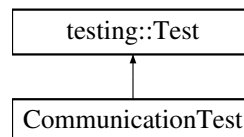
Reimplemented from [Exceptions.CommunicationException](#).

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

- ai/src/Exceptions/Exceptions.py

6.35 CommunicationTest Class Reference

Inheritance diagram for CommunicationTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override
- [zappy::structs::Config](#) **createValidConfig** ()

Protected Attributes

- std::unique_ptr< [MockServer](#) > **mockServer**

Static Protected Attributes

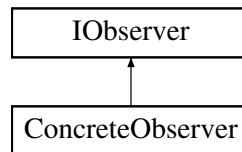
- static const int **TEST_PORT** = 9876

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

- tests/unit/gui/Communication/Communication_test.cpp

6.36 ConcreteObserver Class Reference

Inheritance diagram for ConcreteObserver:



Public Member Functions

- **MOCK_METHOD** (void, update,(),(override))
- **MOCK_METHOD** (void, onGameEvent,(GameEventType eventType, const std::string &teamName),(override))

Public Member Functions inherited from [IObserver](#)

- virtual void **update** ()=0
- virtual void **onGameEvent** (GameEventType eventType, const std::string &teamName)

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

- tests/unit/gui/Observer/IObserver_test.cpp

6.37 zappy::structs::Config Struct Reference

Public Attributes

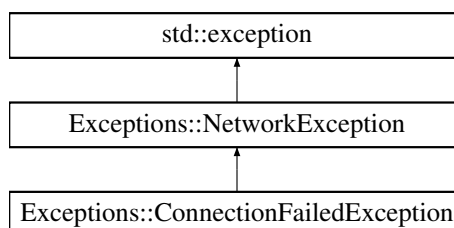
- int **port**
- std::string **hostname**

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

- gui/src/Utils/Constants.hpp

6.38 Exceptions::ConnectionFailedException Class Reference

Inheritance diagram for Exceptions::ConnectionFailedException:



Public Member Functions

- **ConnectionFailedException** (const std::string &message)

Public Member Functions inherited from [Exceptions::NetworkException](#)

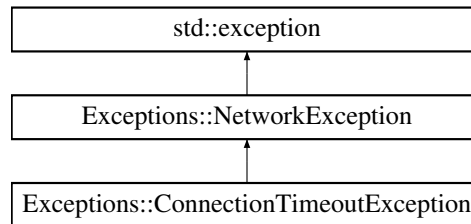
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.39 Exceptions::ConnectionTimeoutException Class Reference

Inheritance diagram for Exceptions::ConnectionTimeoutException:



Public Member Functions

- **ConnectionTimeoutException** (const std::string &message)

Public Member Functions inherited from Exceptions::NetworkException

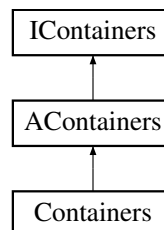
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.40 Containers Class Reference

Inheritance diagram for Containers:



Public Member Functions

- **Containers** (std::shared_ptr< [IDisplay](#) > display, std::shared_ptr< [IAudio](#) > audio, float x, float y, float width, float height, [Color32](#) backgroundColor={40, 40, 40, 200})
- void **draw** () override
- void **update** () override
- void **setBackgroundColor** ([Color32](#) color)
- bool **addElement** (const std::string &id, std::shared_ptr< [IUElement](#) > element)
- std::shared_ptr< [IUElement](#) > **getElement** (const std::string &id) const
- bool **removeElement** (const std::string &id)
- std::shared_ptr< [Button](#) > **addButton** (const std::string &id, float x, float y, float width, float height, const std::string &text, std::function< void()> callback)
- std::shared_ptr< [Button](#) > **addButton** (const std::string &id, float x, float y, float width, float height, const std::string &text, std::function< void()> callback, [Color32](#) normalColor, [Color32](#) hoverColor, [Color32](#) pressedColor, [Color32](#) textColor)
- std::shared_ptr< [Text](#) > **addText** (const std::string &id, float x, float y, const std::string &text, float font↵Size=20.0f, [Color32](#) color=CBLACK)
- std::shared_ptr< [Slider](#) > **addSlider** (const std::string &id, float x, float y, float width, float height, float min↵Value, float maxValue, float initialValue, const std::string &text, std::function< void(float)> onValueChanged)

- `std::shared_ptr< Slider > addSliderPercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, float minVal, float maxVal, float initialValue, const std::string &text, std::function< void(float)> onValueChanged)
- void **clearElements** ()
- void **handleResize** (int oldWidth, int oldHeight, int newWidth, int newHeight)
- `std::shared_ptr< Button > addButtonPercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &text, std::function< void()> callback)
- `std::shared_ptr< Button > addButtonPercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &text, std::function< void()> callback, [Color32](#) normalColor, [Color32](#) hoverColor, [Color32](#) pressedColor, [Color32](#) textColor)
- `std::shared_ptr< Text > addTextPercent` (const std::string &id, float xPercent, float yPercent, const std::string &text, float fontSizePercent=5.0f, [Color32](#) color=CBLACK)
- `std::shared_ptr< Image > addImage` (const std::string &id, float x, float y, float width, float height, const std::string &imagePath)
- `std::shared_ptr< Image > addImage` (const std::string &id, float x, float y, float width, float height, const std::string &imagePath, [Color32](#) tint)
- `std::shared_ptr< Image > addImagePercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &imagePath)
- `std::shared_ptr< Image > addImagePercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &imagePath, [Color32](#) tint)
- `std::shared_ptr< ImageButton > addImageButton` (const std::string &id, float x, float y, float width, float height, const std::string &imagePath, std::function< void()> callback)
- `std::shared_ptr< ImageButton > addImageButton` (const std::string &id, float x, float y, float width, float height, const std::string &imagePath, std::function< void()> callback, [Color32](#) tint)
- `std::shared_ptr< ImageButton > addImageButtonPercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &imagePath, std::function< void()> callback)
- `std::shared_ptr< ImageButton > addImageButtonPercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &imagePath, std::function< void()> callback, [Color32](#) tint)
- `std::shared_ptr< Checkbox > addCheckbox` (const std::string &id, float x, float y, float width, float height, bool initialValue, std::function< void(bool)> callback)
- `std::shared_ptr< Checkbox > addCheckboxPercent` (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, bool initialValue, std::function< void(bool)> callback)
- float **getWidth** () const
- float **getHeight** () const

Public Member Functions inherited from [AContainers](#)

- **AContainers** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void **setPosition** (float x, float y) override
- void **setSize** (float width, float height) override
- [FloatRect](#) **getBounds** () const override
- bool **contains** (float x, float y) const override
- void **setVisible** (bool visible) override
- bool **isVisible** () const override
- void **setRelativePosition** (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [RelativePosition](#) **getRelativePosition** () const
- void **updatePositionFromRelative** ()
- float **getWidth** () const
- float **getHeight** () const

Private Attributes

- std::shared_ptr< [IAudio](#) > **_audio**
- std::unordered_map< std::string, std::shared_ptr< [UIElement](#) > > **_elements**

Additional Inherited Members**Protected Attributes inherited from [AContainers](#)**

- `std::shared_ptr< IDisplay > _display`
- `FloatRect _bounds`
- `RelativePosition _relativePos`
- `Color32 _backgroundColor`
- `bool _visible`
- `bool _hasBackground`

6.40.1 Member Function Documentation**6.40.1.1 draw()**

`void Containers::draw () [override], [virtual]`
 Implements [IContainers](#).

6.40.1.2 update()

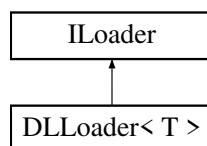
`void Containers::update () [override], [virtual]`
 Implements [IContainers](#).

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

- `gui/src/Graphic/HUD/Containers/Containers.hpp`
- `gui/src/Graphic/HUD/Containers/Containers.cpp`

6.41 DLLoader< T > Class Template Reference

Inheritance diagram for DLLoader< T >:

**Public Member Functions**

- `void * getHandler () const override`
- `void * Open (const char *path, int flag=RTLD_LAZY) override`
- `void * Symbol (const char *symbolName) override`
- `T getSymbol (const char *symbolName)`
- `int Close () override`
- `const char * Error () override`

Private Attributes

- `void * _handler = nullptr`

6.41.1 Member Function Documentation**6.41.1.1 Close()**

`template<typename T >`
`int DLLoader< T >::Close () [inline], [override], [virtual]`
 Implements [ILoader](#).

6.41.1.2 Error()

```
template<typename T >
const char * DLLoader< T >::Error ( ) [inline], [override], [virtual]
Implements ILoader.
```

6.41.1.3 getHandler()

```
template<typename T >
void * DLLoader< T >::getHandler ( ) const [inline], [override], [virtual]
Implements ILoader.
```

6.41.1.4 Open()

```
template<typename T >
void * DLLoader< T >::Open (
    const char * path,
    int flag = RTLD_LAZY ) [inline], [override], [virtual]
Implements ILoader.
```

6.41.1.5 Symbol()

```
template<typename T >
void * DLLoader< T >::Symbol (
    const char * symbolName ) [inline], [override], [virtual]
Implements ILoader.
```

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

- gui/src/DLLoader/DLLoader.hpp

6.42 zappy::structs::Egg Struct Reference

Public Member Functions

- **Egg** (int _eggNumber=0, int _playerNumber=0, int _x=0, int _y=0, bool _hatched=false, const std::string &_teamName="")

Public Attributes

- int **eggNumber**
- int **playerNumber**
- int **x**
- int **y**
- bool **hatched**
- std::string **teamName**

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

- gui/src/Utils/Constants.hpp

6.43 egg_s Struct Reference

Public Attributes

- int **id**
- int **posX**
- int **posY**
- char * **teamName**
- int **idLayer**

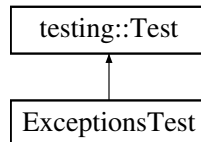
- bool **isHatched**
- struct [egg_s](#) * **next**

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

- server/include/game.h

6.44 ExceptionsTest Class Reference

Inheritance diagram for ExceptionsTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

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

- tests/unit/gui/Exceptions/Exceptions_test.cpp

6.45 FloatRect Struct Reference

Public Attributes

- float **x**
- float **y**
- float **width**
- float **height**

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

- gui/src/IDisplay.hpp

6.46 game_s Struct Reference

Public Attributes

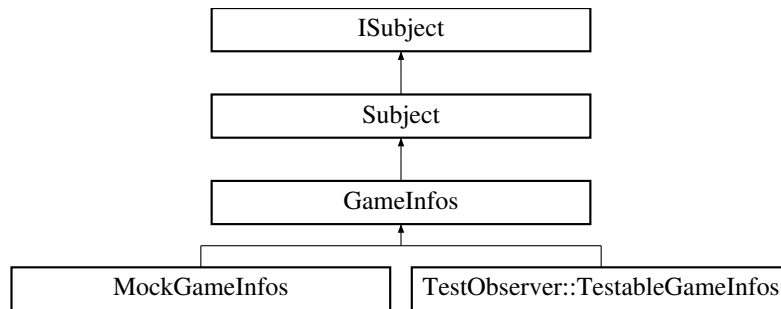
- [team_t](#) * **teams**
- [map_t](#) * **map**

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

- server/include/game.h

6.47 GameInfos Class Reference

Inheritance diagram for GameInfos:



Public Member Functions

- **GameInfos** (std::shared_ptr< [ICommunication](#) > communication)
- void **setAudio** (std::shared_ptr< [IAudio](#) > audio)
- void **setCurrentCameraMode** (zappy::gui::CameraMode cameraMode)
- void **setCurrentPlayerFocus** (int playerId)
- void **setMapSize** (int width, int height)
- std::pair< int, int > **getMapSize** () const
- void **setTimeUnit** (int timeUnit, bool sendToServer=false)
- int **getTimeUnit** () const
- void **updateTile** (const [zappy::structs::Tile](#) tile)
- const [zappy::structs::Tile](#) **getTile** (int x, int y) const
- const [zappy::structs::Tile](#) & **getTileRef** (int x, int y) const
- void **initializeTileMatrix** ()
- void **updateTeamName** (const std::string &teamName)
- const std::vector< std::string > **getTeamNames** () const
- void **setTeamVisibility** (const std::string &teamName, bool visible)
- bool **isTeamVisible** (const std::string &teamName) const
- const std::unordered_map< std::string, bool > **getTeamVisibilities** () const
- void **addPlayer** (const [zappy::structs::Player](#) player)
- void **killPlayer** (int playerNumber)
- void **updatePlayerPosition** (int playerNumber, int x, int y)
- void **updatePlayerOrientation** (int playerNumber, int orientation)
- void **updatePlayerLevel** (int playerNumber, int level)
- void **updatePlayerInventory** (int playerNumber, const [zappy::structs::Inventory](#) inventory)
- void **updatePlayerExpulsion** (int playerNumber)
- void **updatePlayerDeath** (int playerNumber)
- void **updatePlayerResourceAction** (int playerNumber, int resourceId, bool isCollecting)
- void **updatePlayerFork** (int playerNumber)
- const std::vector< [zappy::structs::Player](#) > **getPlayers** () const
- const [zappy::structs::Player](#) **getPlayer** (int playerNumber) const
- void **addPlayerBroadcast** (int playerNumber, const std::string &message)
- const std::vector< std::pair< int, std::string > > **getPlayersBroadcasting** ()
- void **addIncantation** (const [zappy::structs::Incantation](#) incantation)
- void **removeIncantation** (int x, int y, int result)
- const std::vector< [zappy::structs::Incantation](#) > **getIncantations** ()
- void **addEgg** (const [zappy::structs::Egg](#) egg)
- void **updateEggHatched** (int eggNumber)
- void **updateEggDeath** (int eggNumber)
- const std::vector< [zappy::structs::Egg](#) > **getEggs** () const
- void **setGameOver** (const std::string &winningTeam)
- void **playDefeatSound** (const std::string &teamName)
- std::pair< bool, std::string > **isGameOver** () const
- void **addServerMessage** (const std::string &message)

- const std::vector< std::string > **getServerMessages** () const
- void **securityActualisation** ()
- void **incrementPlayerLevel** (int playerNumber)
- void **decrementPlayerLevel** (int playerNumber)
- void **incrementPlayerInventoryItem** (int playerNumber, int resourceId)
- void **decrementPlayerInventoryItem** (int playerNumber, int resourceId)
- void **incrementTileInventoryItem** (int x, int y, int resourceId)
- void **decrementTileInventoryItem** (int x, int y, int resourceId)

Public Member Functions inherited from **Subject**

- void **addObserver** (std::shared_ptr< **IObserver** > observer) override
- void **removeObserver** (std::shared_ptr< **IObserver** > observer) override
- void **notifyObservers** () override
- void **notifyGameEvent** (GameEventType eventType, const std::string &teamName)

Private Member Functions

- void **notifyStateChange** ()

Private Attributes

- int **_mapWidth**
- int **_mapHeight**
- int **_timeUnit**
- std::vector< std::vector< **zappy::structs::Tile** > > **_tileMatrix**
- bool **_matrixInitialized**
- std::vector< std::string > **_teamNames**
- std::unordered_map< std::string, bool > **_teamVisibilities**
- std::vector< **zappy::structs::Player** > **_players**
- std::vector< std::pair< int, bool > > **_playersExpulsing**
- std::vector< std::tuple< int, std::string, std::chrono::steady_clock::time_point > > **_playersBroadcasting**
- std::vector< **zappy::structs::Incantation** > **_incantations**
- std::vector< **zappy::structs::Egg** > **_eggs**
- std::vector< std::string > **_serverMessages**
- bool **_gameOver**
- std::string **_winningTeam**
- bool **_victorySoundPlayed**
- std::mutex **_dataMutex**
- std::shared_ptr< **ICommunication** > **_communication**
- std::shared_ptr< **IAudio** > **_audio**
- **zappy::gui::CameraMode** **_currentCameraMode**
- int **_currentPlayerFocus**

Additional Inherited Members

Protected Attributes inherited from **ISubject**

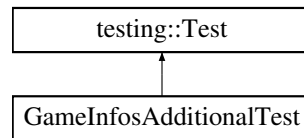
- std::vector< std::weak_ptr< **IObserver** > > **_observers**

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

- gui/src/Game/GameInfos.hpp
- gui/src/Game/GameInfos.cpp

6.48 GameInfosAdditionalTest Class Reference

Inheritance diagram for GameInfosAdditionalTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

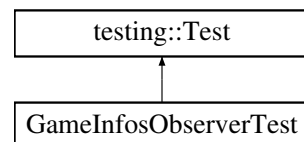
- std::unique_ptr< [GameInfos](#) > **gameInfos**
- std::shared_ptr< testing::NiceMock< [MockCommunication](#) > > **mockCommunication**
- std::shared_ptr< testing::NiceMock< [MockAudio](#) > > **mockAudio**
- std::shared_ptr< testing::NiceMock< [MockObserver](#) > > **mockObserver**

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

- tests/unit/gui/Game/GameInfos_test.cpp

6.49 GameInfosObserverTest Class Reference

Inheritance diagram for GameInfosObserverTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

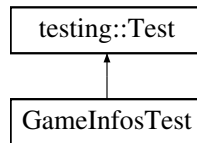
- std::unique_ptr< [TestObserver::TestableGameInfos](#) > **gameInfos**
- std::shared_ptr< testing::NiceMock< [MockCommunication](#) > > **mockCommunication**
- std::shared_ptr< testing::NiceMock< [MockAudio](#) > > **mockAudio**
- std::shared_ptr< [TestObserver](#) > **mockObserver**

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

- tests/unit/gui/Game/GameInfos_test.cpp

6.50 GameInfosTest Class Reference

Inheritance diagram for GameInfosTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

- std::unique_ptr< [GameInfos](#) > **gameInfos**
- std::shared_ptr< testing::NiceMock< [MockCommunication](#) > > **mockCommunication**
- std::shared_ptr< testing::NiceMock< [MockAudio](#) > > **mockAudio**

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

- tests/unit/gui/Game/GameInfos_test.cpp

6.51 graph_net_s Struct Reference

Public Attributes

- int **fd**
- bool **mapSent**
- struct [graph_net_s](#) * **next**

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

- server/include/zappy.h

6.52 graphic_pf_s Struct Reference

Public Attributes

- char * **command**
- int(* **handler**)(zappy_t *zappy, [graph_net_t](#) *graphic, char *message)

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

- server/include/zappy.h

6.53 GUI Class Reference

Public Member Functions

- **GUI** (std::shared_ptr< [GameInfos](#) > gameInfos, const std::string &libPath)
- void **run** ()
- void **refresh** ()
- void **handleVictory** (const std::string &teamName)
- int **getWindowWidth** () const
- int **getWindowHeight** () const

- void **setWindowWidth** (int width)
- void **setWindowHeight** (int height)
- void **switchCameraMode** (zappy::gui::CameraMode mode)
- void **switchCameraModeNext** ()
- void **setPlayerToFollow** (int playerId)
- int **getPlayerToFollow** () const
- bool **selectFirstAvailablePlayer** ()
- void **switchToNextPlayer** ()
- void **switchToPreviousPlayer** ()

Private Member Functions

- void **updateCamera** ()
- virtual void **update** ()
- virtual void **draw** ()
- virtual bool **isRunning** ()
- bool **playerExists** (int playerId) const
- void **initModels** ()
- void **initPlayers** ()
- void **handlePlayerClicks** ()
- int **getPlayerUnderMouse** () const
- [BoundingBox3D](#) **getPlayerBoundingBox** (const [zappy::structs::Player](#) &player) const
- void **handleTileClicks** ()
- std::pair< int, int > **getTileUnderMouse** () const
- [BoundingBox3D](#) **getTileBoundingBox** (int x, int y) const

Private Attributes

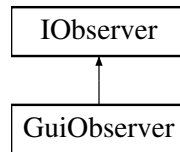
- std::string **_currentLibLoaded**
- bool **_isRunning**
- [DLLoader](#)< std::shared_ptr< [IDisplay](#) > > **_dlLoader**
- std::shared_ptr< [IDisplay](#) > **_display**
- std::shared_ptr< [GameInfos](#) > **_gameInfos**
- std::unique_ptr< [Map](#) > **_map**
- std::unique_ptr< [HUD](#) > **_hud**
- std::shared_ptr< [IAudio](#) > **_audio**
- std::shared_ptr< [CameraManager](#) > **_cameraManager**
- int **_windowWidth**
- int **_windowHeight**
- zappy::gui::CameraMode **_cameraMode**
- bool **_isHUDVisible** = true
- bool **_backgroundLoaded**
- bool **_skyboxLoaded**
- int **_hoveredPlayerId**
- std::pair< int, int > **_selectedTile**
- bool **_performanceMode** = false

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

- gui/src/Graphic/GUI.hpp
- gui/src/Graphic/GUI.cpp

6.54 GuiObserver Class Reference

Inheritance diagram for GuiObserver:



Public Member Functions

- **GuiObserver** (std::shared_ptr< GUI > gui)
- void **update** () override
- void **onGameEvent** (GameEventType eventType, const std::string &teamName) override

Private Attributes

- std::weak_ptr< GUI > **_gui**

6.54.1 Member Function Documentation

6.54.1.1 onGameEvent()

```
void GuiObserver::onGameEvent (
    GameEventType eventType,
    const std::string & teamName ) [override], [virtual]
```

Reimplemented from [IObserver](#).

6.54.1.2 update()

```
void GuiObserver::update ( ) [override], [virtual]
```

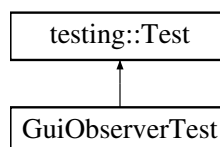
Implements [IObserver](#).

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

- gui/src/Observer/GuiObserver.hpp
- gui/src/Observer/GuiObserver.cpp

6.55 GuiObserverTest Class Reference

Inheritance diagram for GuiObserverTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

- std::shared_ptr< [MockGUI](#) > **mockGui**

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

- tests/unit/gui/Observer/GuiObserver_test.cpp

6.56 Hash.Hash Class Reference

Public Member Functions

- `__init__` (self, str hash_key)
- bytes `simple_xor` (self, bytes data)
- str `hashMessage` (self, str message)
- str `unHashMessage` (self, str hex_message)

Public Attributes

- `key`

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

- ai/src/Hash/Hash.py

6.57 Help Class Reference

Public Member Functions

- `Help` (std::shared_ptr< [IDisplay](#) > display, std::shared_ptr< [IAudio](#) > audio)
- void `show` ()
- void `hide` ()
- bool `isVisible` () const
- bool `containsPoint` (float x, float y) const
- void `update` ()
- void `draw` ()
- void `handleResize` (int oldWidth, int oldHeight, int newWidth, int newHeight)

Private Member Functions

- void `initHelpContainer` ()

Private Attributes

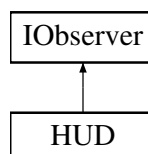
- std::shared_ptr< [IDisplay](#) > `_display`
- std::shared_ptr< [IAudio](#) > `_audio`
- std::shared_ptr< [Containers](#) > `_helpContainer`
- bool `_visible`

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

- gui/src/Graphic/HUD/Help/Help.hpp
- gui/src/Graphic/HUD/Help/Help.cpp

6.58 HUD Class Reference

Inheritance diagram for HUD:



Public Member Functions

- **HUD** (std::shared_ptr< [IDisplay](#) > display, std::shared_ptr< [GameInfos](#) > gameInfos, std::shared_ptr< [IAudio](#) > audio, std::shared_ptr< [CameraManager](#) >, std::function< void()> resetCameraFunc=nullptr)
- void **draw** ()
- std::shared_ptr< [Containers](#) > **addContainer** (const std::string &id, float x, float y, float width, float height, [Color32](#) backgroundColor={40, 40, 40, 200})
- std::shared_ptr< [Containers](#) > **getContainer** (const std::string &id) const
- bool **removeContainer** (const std::string &id)
- void **handleResize** (int oldWidth, int oldHeight, int newWidth, int newHeight)
- void **clearAllContainers** ()
- void **initDefaultLayout** (float sideWidthPercent=15.0f, float bottomHeightPercent=20.0f)
- std::shared_ptr< [Containers](#) > **getSideContainer** () const
- std::shared_ptr< [Containers](#) > **getBottomContainer** () const
- std::shared_ptr< [Containers](#) > **getSquareContainer** () const
- std::shared_ptr< [Containers](#) > **getTpsContainer** () const
- std::shared_ptr< [Containers](#) > **getSecurityContainer** () const
- std::shared_ptr< [Containers](#) > **getServerMessagesContainer** () const
- void **initExitButton** ()
- void **initSettingsButton** ()
- void **initHelpButton** ()
- void **initCameraResetButton** ()
- void **initTeamPlayersDisplay** (std::shared_ptr< [GameInfos](#) > gameInfos)
- void **updateTeamPlayersDisplay** (std::shared_ptr< [GameInfos](#) > gameInfos)
- void **initTpsSlider** (std::shared_ptr< [GameInfos](#) > gameInfos, std::shared_ptr< [IDisplay](#) > raylib, std::shared_ptr< [IAudio](#) > audio)
- void **updateTpsSlider** (std::shared_ptr< [GameInfos](#) > gameInfos)
- void **initServerMessagesDisplay** (std::shared_ptr< [GameInfos](#) > gameInfos)
- void **updateServerMessagesDisplay** (std::shared_ptr< [GameInfos](#) > gameInfos)
- void **initPlayerInventoryDisplay** (int playerId)
- void **updatePlayerInventoryDisplay** (int playerId, zappy::gui::CameraMode cameraMode)
- void **updateHelpInformationHUD** (zappy::gui::CameraMode cameraMode)
- void **clearPlayerInventoryElements** ()
- void **setSelectedTile** (int x, int y)
- void **initTileResourceDisplay** ()
- void **updateTileResourceDisplay** (int x, int y)
- void **clearTileResourceElements** ()
- void **initFpsDisplay** ()
- void **updateFpsDisplay** ()
- [zappy::structs::Player](#) **getPlayerById** (int playerId) const
- bool **isPlayerInIncantation** (int playerId) const
- void **setResetCameraCallback** (std::function< void()> resetFunc)
- void **displayWinMessage** (const std::string &teamName)
- void **update** () override
- void **onGameEvent** (GameEventType eventType, const std::string &teamName) override
- bool **isMouseOverHUD** () const

Private Member Functions

- void **_initHelpInformation** ()
- std::string **_camModeToText** (zappy::gui::CameraMode, bool isGamePadAvailable)
- std::string **_camKeyHelp** (zappy::gui::CameraMode, bool isGamePadAvailable)
- std::shared_ptr< [Containers](#) > **createSquareContainer** (float squareSize, float sideWidthPercent)
- std::shared_ptr< [Containers](#) > **createSideContainer** (float sideYStart, float sideWidth, float sideHeight, float sideWidthPercent, float bottomHeightPercent)

- `std::shared_ptr< Containers > createBottomContainer` (int screenWidth, int screenHeight, float bottomHeight, float bottomHeightPercent)
- `std::shared_ptr< Containers > createTpsContainer` (int screenWidth, int screenHeight, float bottomHeight, float bottomHeightPercent)
- `std::shared_ptr< Containers > createSecurityContainer` (int screenWidth, int screenHeight, float bottomHeight, float bottomHeightPercent)
- `std::shared_ptr< Containers > createServerMessagesContainer` (int screenWidth, int screenHeight, float bottomHeight, float bottomHeightPercent)
- `void updateElementPositions` (std::shared_ptr< [Containers](#) > container, const std::unordered_map< std::string, float > &initialYPositions, float offset)
- `std::pair< float, float > calculateContentMetrics` (std::shared_ptr< [Containers](#) > container, const std::unordered_map< std::string, float > &initialYPositions)
- `void clearTeamDisplayElements` (std::shared_ptr< [Containers](#) > container)
- `std::vector< int > getTeamPlayerNumbers` (const std::string &teamName, const std::vector< [zappy::structs::Player](#) > &players)
- `std::string createPlayerListText` (const std::vector< int > &playerNumbers)
- `void addPlayerListText` (std::shared_ptr< [Containers](#) > container, const std::string &teamId, float yPos, const std::vector< int > &playerNumbers)
- `void addIncrementDecrementButtons` (std::shared_ptr< [Containers](#) > container, int playerId)

Private Attributes

- `std::unordered_map< std::string, std::shared_ptr< Containers > > _containers`
- `std::shared_ptr< IDisplay > _display`
- `std::shared_ptr< GameInfos > _gameInfos`
- `std::shared_ptr< IAudio > _audio`
- `std::shared_ptr< CameraManager > _camera`
- `std::shared_ptr< Help > _help`
- `std::shared_ptr< Settings > _settings`
- `std::function< void() > _resetCameraFunc`
- `bool _showVictoryMessage`
- `std::string _winningTeam`
- `Color32 _victoryColor`
- `std::pair< int, int > _selectedTile`

6.58.1 Member Function Documentation

6.58.1.1 onGameEvent()

```
void HUD::onGameEvent (
    GameEventType eventType,
    const std::string & teamName ) [override], [virtual]
```

Reimplemented from [IObserver](#).

6.58.1.2 update()

```
void HUD::update ( ) [override], [virtual]
```

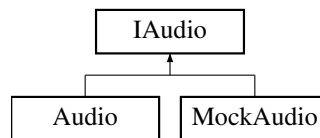
Implements [IObserver](#).

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

- `gui/src/Graphic/HUD/HUD.hpp`
- `gui/src/Graphic/HUD/HUD.cpp`

6.59 IAudio Class Reference

Inheritance diagram for IAudio:



Public Member Functions

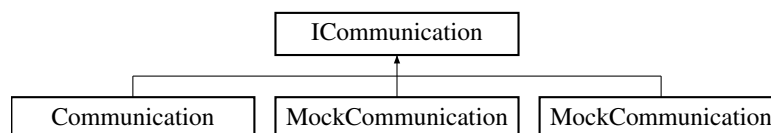
- virtual float **getSFXVolumeLevel** ()=0
- virtual float **getMusicVolumeLevel** ()=0
- virtual void **setSFXVolumeLevel** (float)=0
- virtual void **setMusicVolumeLevel** (float)=0
- virtual bool **loadSound** (const std::string &id, const std::string &filepath)=0
- virtual void **playMainTheme** (float volume)=0
- virtual void **playNextTheme** (float volume)=0
- virtual void **playSound** (const std::string &id, float volume)=0
- virtual void **stopSound** (const std::string &id)=0
- virtual bool **isSoundPlaying** (const std::string &id) const =0
- virtual void **setSoundLooping** (const std::string &id, bool looping)=0
- virtual void **setSoundVolume** (const std::string &id, float volume)=0

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

- gui/src/Audio/IAudio.hpp

6.60 ICommunication Class Reference

Inheritance diagram for ICommunication:



Public Member Functions

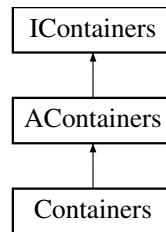
- virtual void **sendMessage** (const std::string &message)=0
- virtual bool **hasMessages** () const =0
- virtual std::string **popMessage** ()=0
- virtual bool **isConnected** () const =0
- virtual void **disconnect** ()=0

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

- gui/src/Communication/ICommunication.hpp

6.61 IContainers Class Reference

Inheritance diagram for IContainers:



Public Member Functions

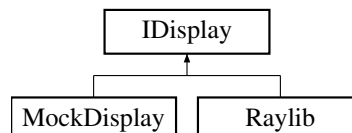
- virtual void **draw** ()=0
- virtual void **update** ()=0
- virtual void **setPosition** (float x, float y)=0
- virtual void **setSize** (float width, float height)=0
- virtual [FloatRect](#) **getBounds** () const =0
- virtual bool **contains** (float x, float y) const =0
- virtual void **setVisible** (bool visible)=0
- virtual bool **isVisible** () const =0

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

- [gui/src/Graphic/HUD/Containers/IContainers.hpp](#)

6.62 IDisplay Class Reference

Inheritance diagram for IDisplay:



Public Member Functions

- virtual [Vector2i](#) **getMonitorSize** ()=0
- virtual [Vector2i](#) **getScreenSize** ()=0
- virtual void **initWindow** (int width, int height, std::string)=0
- virtual void **initCamera** ()=0
- virtual bool **isWindowReady** ()=0
- virtual void **setTargetFPS** (unsigned int FPS)=0
- virtual bool **isOpen** ()=0
- virtual void **closeWindow** ()=0
- virtual int **getKeyId** (enum Key)=0
- virtual bool **isKeyReleased** (int key)=0
- virtual bool **isKeyPressed** (int key)=0
- virtual bool **isKeyDown** (int key)=0
- virtual bool **isGamepadAvailable** ()=0
- virtual bool **isGamepadButtonReleased** (int key)=0
- virtual bool **isGamepadButtonPressed** (int key)=0
- virtual bool **isGamepadButtonDown** (int key)=0

- virtual bool **isMouseButtonDown** (int key)=0
- virtual bool **isMouseButtonReleased** (int key)=0
- virtual bool **isMouseButtonPressed** (int key)=0
- virtual [Vector2f](#) **getMousePosition** ()=0
- virtual void **setMousePosition** ([Vector2f](#))=0
- virtual float **getMouseWheelMove** ()=0
- virtual float **getGamepadAxisMovement** (int key)=0
- virtual void **setCameraPosition** ([Vector3f](#))=0
- virtual void **setCameraTarget** ([Vector3f](#))=0
- virtual [Vector2f](#) **getMouseDelta** ()=0
- virtual float **vector3DDistanceFromCamera** ([Vector3f](#) target)=0
- virtual [Vector3f](#) **vector3SubtractFromCamera** ([Vector3f](#) target)=0
- virtual [Vector3f](#) **vector3Normalize** ([Vector3f](#))=0
- virtual void **enableCursor** ()=0
- virtual void **disableCursor** ()=0
- virtual float **getFrameTime** ()=0
- virtual int **getFPS** ()=0
- virtual void **updateCameraFreeMode** (float camMovingSpeed, float camRotaSpeed)=0
- virtual InputType **getLastInputType** () const =0
- virtual void **updateLastInputType** ()=0
- virtual float **measureText** (const std::string &text, float fontSize) const =0
- virtual bool **checkCollisionPointRec** ([Vector2f](#) point, [FloatRect](#) rec)=0
- virtual [Ray3D](#) **getMouseRay** ([Vector2f](#) mousePosition)=0
- virtual [RayCollision3D](#) **getRayCollisionBox** ([Ray3D](#) ray, [BoundingBox3D](#) box)=0
- virtual [RayCollision3D](#) **getRayCollisionSphere** ([Ray3D](#) ray, [Vector3f](#) center, float radius)=0
- virtual bool **checkCollisionBoxes** ([BoundingBox3D](#) box1, [BoundingBox3D](#) box2)=0
- virtual [Ray3D](#) **getMouseRayFromCurrent** ()=0
- virtual [BoundingBox3D](#) **createBoundingBox** ([Vector3f](#) center, [Vector3f](#) size)=0
- virtual [BoundingBox3D](#) **createBoundingBoxFromMinMax** ([Vector3f](#) min, [Vector3f](#) max)=0
- virtual void **beginDrawing** ()=0
- virtual void **endDrawing** ()=0
- virtual void **clearBackground** ([Color32](#))=0
- virtual void **begin3DMode** ()=0
- virtual void **end3DMode** ()=0
- virtual void **endScissorMode** ()=0
- virtual void **beginScissorMode** ([IntRect](#))=0
- virtual bool **loadModel** (const std::string &id, const std::string &filepath, [Vector3f](#) center={0.0f, 0.0f, 0.0f})=0
- virtual void **drawCube** ([Vector3f](#) position, float width, float height, float length, [Color32](#) color)=0
- virtual void **drawCubeWires** ([Vector3f](#) position, float width, float height, float length, [Color32](#) color)=0
- virtual void **drawSphere** ([Vector3f](#) position, float radius, [Color32](#) color)=0
- virtual void **drawSphereWires** ([Vector3f](#) position, float radius, int rings, int slices, [Color32](#) color)=0
- virtual void **drawCylinder** ([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color)=0
- virtual void **drawCylinderWires** ([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color)=0
- virtual void **drawCylinderEx** ([Vector3f](#) startPos, [Vector3f](#) endPos, float startRadius, float endRadius, int sides, [Color32](#) color)=0
- virtual void **drawPlane** ([Vector3f](#) position, [Vector2f](#) size, [Color32](#) color)=0
- virtual void **drawLine3D** ([Vector3f](#) startPos, [Vector3f](#) endPos, [Color32](#) color)=0
- virtual void **drawModelEx** (const std::string &id, [Vector3f](#) position, [Vector3f](#) rotationAxis, float rotationAngle, [Vector3f](#) scale, [Color32](#) tint=CWHITE)=0
- virtual void **drawCircle** (float centerX, float centerY, float radius, [Color32](#) color)=0
- virtual void **drawCircleLines** (float centerX, float centerY, float radius, [Color32](#) color)=0
- virtual void **drawText** (const std::string &text, float x, float y, float fontSize, [Color32](#) color)=0
- virtual void **drawTextEx** (const std::string &text, float x, float y, float fontSize, float spacing, [Color32](#) color)=0

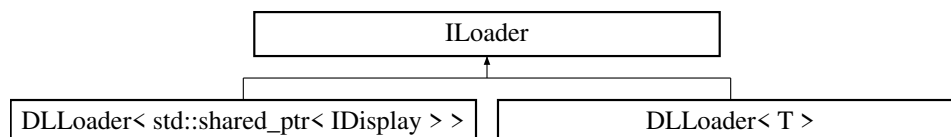
- virtual void **drawRectangleRec** ([FloatRect](#) rec, [Color32](#) color)=0
- virtual bool **loadTexture** (const std::string &id, const std::string &filepath)=0
- virtual bool **loadFont** (const std::string &id, const std::string &filepath)=0
- virtual void **drawTexture** (const std::string &id, float x, float y, [Color32](#) tint=CWHITE)=0
- virtual void **drawTextureScaled** (const std::string &id, float x, float y, float width, float height, [Color32](#) tint=CWHITE)=0
- virtual [Vector2f](#) **getTextureSize** (const std::string &id) const =0
- virtual bool **loadSkybox** (const std::string &id, const std::string &filepath)=0
- virtual void **drawSkybox** (const std::string &id)=0
- virtual float **getTime** () const =0

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

- gui/src/IDisplay.hpp

6.63 ILoader Class Reference

Inheritance diagram for ILoader:



Public Member Functions

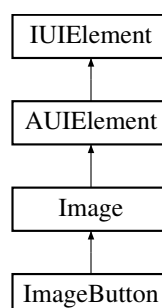
- virtual void * **Open** (const char *path, int flag)=0
- virtual void * **Symbol** (const char *symbolName)=0
- virtual int **Close** ()=0
- virtual const char * **Error** ()=0
- virtual void * **getHandler** () const =0

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

- gui/src/DLoader/ILoader.hpp

6.64 Image Class Reference

Inheritance diagram for Image:



Public Member Functions

- **Image** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height, const std::string &imagePath)
- void [draw](#) () override
- void [update](#) () override
- void [setImagePath](#) (const std::string &imagePath)
- std::string [getImagePath](#) () const
- void [setTint](#) ([Color32](#) tint)
- [Color32](#) [getTint](#) () const
- void [setSize](#) (float width, float height) override
- void [setMaintainAspectRatio](#) (bool maintain)
- bool [getMaintainAspectRatio](#) () const

Public Member Functions inherited from [AUIElement](#)

- **AUIElement** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void [setPosition](#) (float x, float y) override
- [FloatRect](#) [getBounds](#) () const override
- bool [contains](#) (float x, float y) const override
- void [setVisible](#) (bool visible) override
- bool [isVisible](#) () const override
- void [setRelativePosition](#) (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [UIRelativePosition](#) [getRelativePosition](#) () const

Private Member Functions

- void [loadImage](#) ()

Private Attributes

- std::string [_imagePath](#)
- [Color32](#) [_tint](#)
- bool [_maintainAspectRatio](#)
- bool [_imageLoaded](#)

Additional Inherited Members**Protected Attributes inherited from [AUIElement](#)**

- std::shared_ptr< [IDisplay](#) > [_display](#)
- [FloatRect](#) [_bounds](#)
- [UIRelativePosition](#) [_relativePos](#)
- bool [_visible](#)

6.64.1 Member Function Documentation**6.64.1.1 draw()**

void Image::draw () [override], [virtual]
Implements [AUIElement](#).

6.64.1.2 setSize()

```
void Image::setSize (
    float width,
    float height ) [override], [virtual]
```

Reimplemented from [AUIElement](#).

6.64.1.3 update()

`void Image::update () [override], [virtual]`

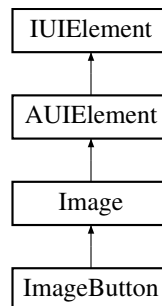
Implements [IUIElement](#).

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

- `gui/src/Graphic/HUD/Image/Image.hpp`
- `gui/src/Graphic/HUD/Image/Image.cpp`

6.65 ImageButton Class Reference

Inheritance diagram for ImageButton:



Public Member Functions

- **ImageButton** (`std::shared_ptr< IDisplay > display`, `std::shared_ptr< IAudio > audio`, `float x`, `float y`, `float width`, `float height`, `const std::string &imagePath`, `std::function< void()> callback`)
- void [update](#) () override
- void **setCallback** (`std::function< void()> callback`)
- `std::function< void()> getCallback () const`

Public Member Functions inherited from [Image](#)

- **Image** (`std::shared_ptr< IDisplay > display`, `float x`, `float y`, `float width`, `float height`, `const std::string &imagePath`)
- void [draw](#) () override
- void **setImagePath** (`const std::string &imagePath`)
- `std::string getImagePath () const`
- void **setTint** (`Color32 tint`)
- `Color32 getTint () const`
- void [setSize](#) (`float width`, `float height`) override
- void **setMaintainAspectRatio** (`bool maintain`)
- `bool getMaintainAspectRatio () const`

Public Member Functions inherited from [AUIElement](#)

- **AUIElement** (`std::shared_ptr< IDisplay > display`, `float x`, `float y`, `float width`, `float height`)
- void [setPosition](#) (`float x`, `float y`) override
- `FloatRect getBounds () const` override
- `bool contains (float x, float y) const` override
- void [setVisible](#) (`bool visible`) override
- `bool isVisible () const` override
- void **setRelativePosition** (`float xPercent`, `float yPercent`, `float widthPercent`, `float heightPercent`)
- `UIRelativePosition getRelativePosition () const`

Private Attributes

- `std::function< void()> _callback`
- `std::shared_ptr< IAudio > _audio`
- `bool _isHovered`
- `bool _isPressed`

Additional Inherited Members**Protected Attributes inherited from [AUIElement](#)**

- `std::shared_ptr< IDisplay > _display`
- `FloatRect _bounds`
- `UIRelativePosition _relativePos`
- `bool _visible`

6.65.1 Member Function Documentation**6.65.1.1 update()**

`void ImageButton::update () [override], [virtual]`

Reimplemented from [Image](#).

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

- `gui/src/Graphic/HUD/ImageButton/ImageButton.hpp`
- `gui/src/Graphic/HUD/ImageButton/ImageButton.cpp`

6.66 zappy::structs::Incantation Struct Reference**Public Member Functions**

- **Incantation** (int `_x`=0, int `_y`=0, int `_level`=1, const `std::vector< int > &_players`={})

Public Attributes

- int **x**
- int **y**
- int **level**
- `std::vector< int > players`

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

- `gui/src/Utils/Constants.hpp`

6.67 incantation_s Struct Reference**Public Attributes**

- int **levelt_to_reach**
- int **nb_players**
- `inventory_t required_inventory`

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

- `server/include/game.h`

6.68 IntRect Struct Reference

Public Attributes

- int **x**
- int **y**
- int **width**
- int **height**

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

- gui/src/IDisplay.hpp

6.69 zappy::structs::Inventory Struct Reference

Public Member Functions

- **Inventory** (int _food=0, int _linemate=0, int _deraumere=0, int _sibur=0, int _mendiane=0, int _phiras=0, int _thystame=0)

Public Attributes

- int **food**
- int **linemate**
- int **deraumere**
- int **sibur**
- int **mendiane**
- int **phiras**
- int **thystame**

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

- gui/src/Utils/Constants.hpp

6.70 inventory_s Struct Reference

Public Attributes

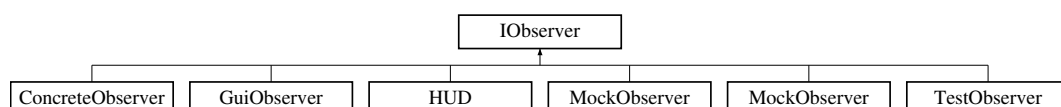
- int **nbFood**
- int **nbLinemate**
- int **nbDeraumere**
- int **nbSibur**
- int **nbMendiane**
- int **nbPhiras**
- int **nbThystame**

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

- server/include/game.h

6.71 IObserver Class Reference

Inheritance diagram for IObserver:



Public Member Functions

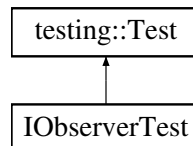
- virtual void **update** ()=0
- virtual void **onGameEvent** (GameEventType eventType, const std::string &teamName)

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

- gui/src/Observer/IObserver.hpp

6.72 IObserverTest Class Reference

Inheritance diagram for IObserverTest:

**Protected Member Functions**

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

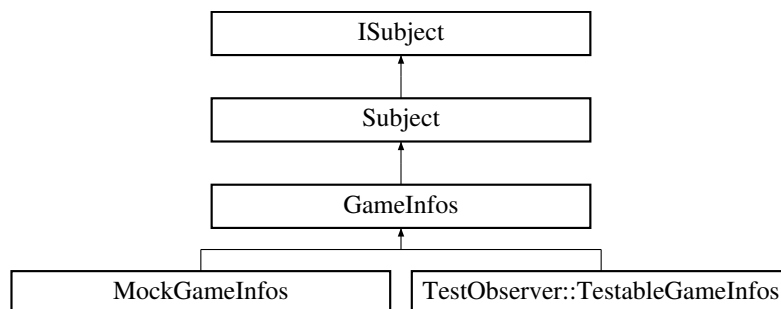
- std::shared_ptr< [ConcreteObserver](#) > **observer**

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

- tests/unit/gui/Observer/IObserver_test.cpp

6.73 ISubject Class Reference

Inheritance diagram for ISubject:

**Public Member Functions**

- virtual void **addObserver** (std::shared_ptr< [IObserver](#) > observer)=0
- virtual void **removeObserver** (std::shared_ptr< [IObserver](#) > observer)=0
- virtual void **notifyObservers** ()=0
- virtual void **notifyGameEvent** (GameEventType eventType, const std::string &teamName)=0

Protected Attributes

- std::vector< std::weak_ptr< [IObserver](#) > > **_observers**

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

- gui/src/Observer/ISubject.hpp

6.74 item_handler_t Struct Reference

Public Attributes

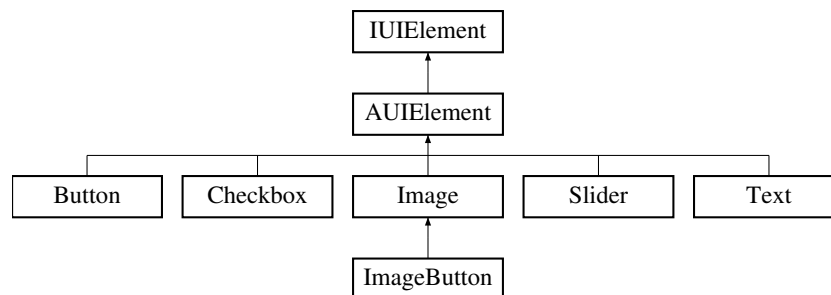
- char * **name**
- void(* **add_func**)(inventory_t *)

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

- server/include/zappy.h

6.75 UIElement Class Reference

Inheritance diagram for UIElement:



Public Member Functions

- virtual void **draw** ()=0
- virtual void **update** ()=0
- virtual void **setPosition** (float x, float y)=0
- virtual void **setSize** (float width, float height)=0
- virtual FloatRect **getBounds** () const =0
- virtual bool **contains** (float x, float y) const =0
- virtual void **setVisible** (bool visible)=0
- virtual bool **isVisible** () const =0

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

- gui/src/Graphic/HUD/UIElement/UIElement.hpp

6.76 Logger.Logger Class Reference

Public Member Functions

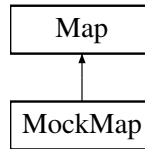
- None **error** (self, str message)
- None **info** (self, str message)
- None **help** (self, str message)
- None **debug** (self, str message)
- None **success** (self, str message)
- None **display** (self, str message)

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

- ai/src/Logger/Logger.py

6.77 Map Class Reference

Inheritance diagram for Map:



Public Member Functions

- **Map** (std::shared_ptr< [GameInfos](#) > gameInfos, std::shared_ptr< [IDisplay](#) > display)
- void **draw** (bool performanceMode=false)
- void **drawBroadcastingPlayers** ()
- void **drawIncantations** ()
- void **drawTile** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawPerformanceTile** (const [zappy::structs::Tile](#) &tile)
- void **drawRock** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawPerformanceRock** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawFood** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawPerformanceFood** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawAllPlayers** ()
- void **drawEggs** (int x, int y)
- [Color32](#) **getTeamColor** (const std::string &teamName)
- float **getOffset** (DisplayPriority priority, int x, int y, size_t stackIndex=0)
- void **updatePlayerRotations** ()
- float **getPlayerInterpolatedRotation** (int playerId, int serverOrientation)
- void **updatePlayerPositions** ()
- [Vector3f](#) **getPlayerInterpolatedPosition** (int playerId, int serverX, int serverY)

Private Member Functions

- void **drawTorus** (const [Vector3f](#) &position, float radius, float thickness, int radialSegments, [Color32](#) color)
- float **orientationToRotation** (int orientation)
- float **normalizeAngle** (float angle)
- float **getShortestAngleDifference** (float from, float to)
- [Vector3f](#) **calculatePlayerWorldPosition** (int x, int y)
- float **getDistance** (const [Vector3f](#) &from, const [Vector3f](#) &to)
- [Vector3f](#) **lerpVector3f** (const [Vector3f](#) &from, const [Vector3f](#) &to, float t)

Private Attributes

- std::shared_ptr< [GameInfos](#) > **_gameInfos**
- std::shared_ptr< [IDisplay](#) > **_display**
- std::unordered_map< std::string, [Color32](#) > **_teamColors**
- std::vector< [Color32](#) > **_colors**
- int **_colorIndex** = 0
- std::unordered_map< int, std::chrono::steady_clock::time_point > **_broadcastStartTimes**
- std::unordered_map< int, [PlayerRotationState](#) > **_playerRotations**
- std::unordered_map< int, [PlayerPositionState](#) > **_playerPositions**
- bool **_performanceMode** = false

Static Private Attributes

- static constexpr float **BASE_HEIGHT_TILE** = 0.0f
- static constexpr float **BASE_HEIGHT_PLAYER** = 0.0f
- static constexpr float **PLAYER_HEIGHT** = 0.95f
- static constexpr float **BASE_HEIGHT_EGG** = 0.0f
- static constexpr float **EGG_HEIGHT** = 0.2f
- static constexpr float **BASE_HEIGHT_FOOD** = 0.1f
- static constexpr float **FOOD_HEIGHT** = 0.7f
- static constexpr float **BASE_HEIGHT_ROCK** = 0.1f
- static constexpr float **ROCK_HEIGHT** = 0.7f

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

- gui/src/Graphic/Map.hpp
- gui/src/Graphic/Map.cpp

6.78 map_t Struct Reference

Public Attributes

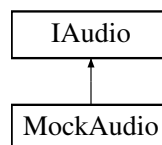
- int **width**
- int **height**
- [egg_t](#) * **currentEggs**
- [inventory_t](#) ** **tiles**

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

- server/include/game.h

6.79 MockAudio Class Reference

Inheritance diagram for MockAudio:



Public Member Functions

- **MOCK_METHOD** (float, getSFXVolumeLevel,(),(override))
- **MOCK_METHOD** (float, getMusicVolumeLevel,(),(override))
- **MOCK_METHOD** (void, setSFXVolumeLevel,(float),(override))
- **MOCK_METHOD** (void, setMusicVolumeLevel,(float),(override))
- **MOCK_METHOD** (bool, loadSound,(const std::string &id, const std::string &filepath),(override))
- **MOCK_METHOD** (void, playMainTheme,(float volume),(override))
- **MOCK_METHOD** (void, playNextTheme,(float volume),(override))
- **MOCK_METHOD** (void, playSound,(const std::string &id, float volume),(override))
- **MOCK_METHOD** (void, stopSound,(const std::string &id),(override))
- **MOCK_METHOD** (bool, isSoundPlaying,(const std::string &id),(const, override))
- **MOCK_METHOD** (void, setSoundLooping,(const std::string &id, bool looping),(override))
- **MOCK_METHOD** (void, setSoundVolume,(const std::string &id, float volume),(override))

Public Member Functions inherited from [IAudio](#)

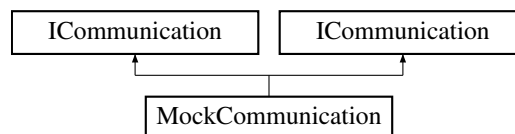
- virtual float **getSFXVolumeLevel** ()=0
- virtual float **getMusicVolumeLevel** ()=0
- virtual void **setSFXVolumeLevel** (float)=0
- virtual void **setMusicVolumeLevel** (float)=0
- virtual bool **loadSound** (const std::string &id, const std::string &filepath)=0
- virtual void **playMainTheme** (float volume)=0
- virtual void **playNextTheme** (float volume)=0
- virtual void **playSound** (const std::string &id, float volume)=0
- virtual void **stopSound** (const std::string &id)=0
- virtual bool **isSoundPlaying** (const std::string &id) const =0
- virtual void **setSoundLooping** (const std::string &id, bool looping)=0
- virtual void **setSoundVolume** (const std::string &id, float volume)=0

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

- tests/unit/gui/Game/GameInfos_test.cpp

6.80 MockCommunication Class Reference

Inheritance diagram for MockCommunication:



Public Member Functions

- **MOCK_METHOD** (void, sendMessage,(const std::string &message),(override))
- **MOCK_METHOD** (bool, hasMessages,(),(const, override))
- **MOCK_METHOD** (std::string, popMessage,(),(override))
- **MOCK_METHOD** (bool, isConnected,(),(const, override))
- **MOCK_METHOD** (void, disconnect,(),(override))
- **MOCK_METHOD** (void, sendMessage,(const std::string &message),(override))
- **MOCK_METHOD** (bool, hasMessages,(),(const, override))
- **MOCK_METHOD** (std::string, popMessage,(),(override))
- **MOCK_METHOD** (bool, isConnected,(),(const, override))
- **MOCK_METHOD** (void, disconnect,(),(override))

Public Member Functions inherited from [ICommunication](#)

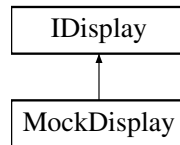
- virtual void **sendMessage** (const std::string &message)=0
- virtual bool **hasMessages** () const =0
- virtual std::string **popMessage** ()=0
- virtual bool **isConnected** () const =0
- virtual void **disconnect** ()=0

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

- tests/unit/gui/Client/MsgHandler_test.cpp
- tests/unit/gui/Game/GameInfos_test.cpp

6.81 MockDisplay Class Reference

Inheritance diagram for MockDisplay:



Public Member Functions

- **MOCK_METHOD** ([Vector2i](#), getMonitorSize,(),(override))
- **MOCK_METHOD** ([Vector2i](#), getScreenSize,(),(override))
- **MOCK_METHOD** (void, initWindow,(int width, int height, std::string title),(override))
- **MOCK_METHOD** (void, initCamera,(),(override))
- **MOCK_METHOD** (bool, isWindowReady,(),(override))
- **MOCK_METHOD** (void, setTargetFPS,(unsigned int FPS),(override))
- **MOCK_METHOD** (bool, isOpen,(),(override))
- **MOCK_METHOD** (void, closeWindow,(),(override))
- **MOCK_METHOD** (int, getKeyId,(enum Key),(override))
- **MOCK_METHOD** (bool, isKeyReleased,(int key),(override))
- **MOCK_METHOD** (bool, isKeyPressed,(int key),(override))
- **MOCK_METHOD** (bool, isKeyDown,(int key),(override))
- **MOCK_METHOD** (bool, isGamepadAvailable,(),(override))
- **MOCK_METHOD** (bool, isGamepadButtonReleased,(int key),(override))
- **MOCK_METHOD** (bool, isGamepadButtonPressed,(int key),(override))
- **MOCK_METHOD** (bool, isGamepadButtonDown,(int key),(override))
- **MOCK_METHOD** (bool, isMouseButtonDown,(int key),(override))
- **MOCK_METHOD** (bool, isMouseButtonReleased,(int key),(override))
- **MOCK_METHOD** (bool, isMouseButtonPressed,(int key),(override))
- **MOCK_METHOD** ([Vector2f](#), getMousePosition,(),(override))
- **MOCK_METHOD** (void, setMousePosition,([Vector2f](#)),(override))
- **MOCK_METHOD** (float, getMouseWheelMove,(),(override))
- **MOCK_METHOD** (float, getGamepadAxisMovement,(int key),(override))
- **MOCK_METHOD** (void, setCameraPosition,([Vector3f](#)),(override))
- **MOCK_METHOD** (void, setCameraTarget,([Vector3f](#)),(override))
- **MOCK_METHOD** ([Vector2f](#), getMouseDelta,(),(override))
- **MOCK_METHOD** (float, vector3DDistanceFromCamera,([Vector3f](#) target),(override))
- **MOCK_METHOD** ([Vector3f](#), vector3DSubtractFromCamera,([Vector3f](#) target),(override))
- **MOCK_METHOD** ([Vector3f](#), vector3DNormalize,([Vector3f](#)),(override))
- **MOCK_METHOD** (void, enableCursor,(),(override))
- **MOCK_METHOD** (void, disableCursor,(),(override))
- **MOCK_METHOD** (float, getFrameTime,(),(override))
- **MOCK_METHOD** (int, getFPS,(),(override))
- **MOCK_METHOD** (void, updateCameraFreeMode,(float camMovingSpeed, float camRotaSpeed),(override))
- **MOCK_METHOD** (InputType, getLastInputType,(),(const, override))
- **MOCK_METHOD** (void, updateLastInputType,(),(override))
- **MOCK_METHOD** (float, measureText,(const std::string &text, float fontSize),(const, override))
- **MOCK_METHOD** (bool, checkCollisionPointRec,([Vector2f](#) point, [FloatRect](#) rec),(override))
- **MOCK_METHOD** ([Ray3D](#), getMouseRay,([Vector2f](#) mousePosition),(override))
- **MOCK_METHOD** ([RayCollision3D](#), getRayCollisionBox,([Ray3D](#) ray, [BoundingBox3D](#) box),(override))
- **MOCK_METHOD** ([RayCollision3D](#), getRayCollisionSphere,([Ray3D](#) ray, [Vector3f](#) center, float radius),(override))
- **MOCK_METHOD** (bool, checkCollisionBoxes,([BoundingBox3D](#) box1, [BoundingBox3D](#) box2),(override))

- **MOCK_METHOD** ([Ray3D](#), getMouseRayFromCurrent,(),(override))
- **MOCK_METHOD** ([BoundingBox3D](#), createBoundingBox,([Vector3f](#) center, [Vector3f](#) size),(override))
- **MOCK_METHOD** ([BoundingBox3D](#), createBoundingBoxFromMinMax,([Vector3f](#) min, [Vector3f](#) max),(override))
- **MOCK_METHOD** (void, beginDrawing,(),(override))
- **MOCK_METHOD** (void, endDrawing,(),(override))
- **MOCK_METHOD** (void, clearBackground,([Color32](#)),(override))
- **MOCK_METHOD** (void, begin3DMode,(),(override))
- **MOCK_METHOD** (void, end3DMode,(),(override))
- **MOCK_METHOD** (void, endScissorMode,(),(override))
- **MOCK_METHOD** (void, beginScissorMode,([IntRect](#)),(override))
- **MOCK_METHOD** (bool, loadModel,(const std::string &id, const std::string &filepath, [Vector3f](#) center),(override))
- **MOCK_METHOD** (void, drawCube,([Vector3f](#) position, float width, float height, float length, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawCubeWires,([Vector3f](#) position, float width, float height, float length, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawSphere,([Vector3f](#) position, float radius, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawSphereWires,([Vector3f](#) position, float radius, int rings, int slices, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawCylinder,([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawCylinderWires,([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawCylinderEx,([Vector3f](#) startPos, [Vector3f](#) endPos, float startRadius, float endRadius, int sides, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawPlane,([Vector3f](#) position, [Vector2f](#) size, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawLine3D,([Vector3f](#) startPos, [Vector3f](#) endPos, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawModelEx,(const std::string &id, [Vector3f](#) position, [Vector3f](#) rotationAxis, float rotationAngle, [Vector3f](#) scale, [Color32](#) tint),(override))
- **MOCK_METHOD** (void, drawCircle,(float centerX, float centerY, float radius, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawCircleLines,(float centerX, float centerY, float radius, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawText,(const std::string &text, float x, float y, float fontSize, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawTextEx,(const std::string &text, float x, float y, float fontSize, float spacing, [Color32](#) color),(override))
- **MOCK_METHOD** (void, drawRectangleRec,([FloatRect](#) rec, [Color32](#) color),(override))
- **MOCK_METHOD** (bool, loadTexture,(const std::string &id, const std::string &filepath),(override))
- **MOCK_METHOD** (bool, loadFont,(const std::string &id, const std::string &filepath),(override))
- **MOCK_METHOD** (void, drawTexture,(const std::string &id, float x, float y, [Color32](#) tint),(override))
- **MOCK_METHOD** (void, drawTextureScaled,(const std::string &id, float x, float y, float width, float height, [Color32](#) tint),(override))
- **MOCK_METHOD** ([Vector2f](#), getTextureSize,(const std::string &id),(const, override))
- **MOCK_METHOD** (bool, loadSkybox,(const std::string &id, const std::string &filepath),(override))
- **MOCK_METHOD** (void, drawSkybox,(const std::string &id),(override))
- **MOCK_METHOD** (float, getTime,(),(const, override))

Public Member Functions inherited from [IDisplay](#)

- virtual [Vector2i](#) **getMonitorSize** ()=0
- virtual [Vector2i](#) **getScreenSize** ()=0
- virtual void **initWindow** (int width, int height, std::string)=0
- virtual void **initCamera** ()=0
- virtual bool **isWindowReady** ()=0
- virtual void **setTargetFPS** (unsigned int FPS)=0
- virtual bool **isOpen** ()=0

- virtual void **closeWindow** ()=0
- virtual int **getKeyId** (enum Key)=0
- virtual bool **isKeyReleased** (int key)=0
- virtual bool **isKeyPressed** (int key)=0
- virtual bool **isKeyDown** (int key)=0
- virtual bool **isGamepadAvailable** ()=0
- virtual bool **isGamepadButtonReleased** (int key)=0
- virtual bool **isGamepadButtonPressed** (int key)=0
- virtual bool **isGamepadButtonDown** (int key)=0
- virtual bool **isMouseButtonDown** (int key)=0
- virtual bool **isMouseButtonReleased** (int key)=0
- virtual bool **isMouseButtonPressed** (int key)=0
- virtual [Vector2f](#) **getMousePosition** ()=0
- virtual void **setMousePosition** ([Vector2f](#))=0
- virtual float **getMouseWheelMove** ()=0
- virtual float **getGamepadAxisMovement** (int key)=0
- virtual void **setCameraPosition** ([Vector3f](#))=0
- virtual void **setCameraTarget** ([Vector3f](#))=0
- virtual [Vector2f](#) **getMouseDelta** ()=0
- virtual float **vector3DDistanceFromCamera** ([Vector3f](#) target)=0
- virtual [Vector3f](#) **vector3SubtractFromCamera** ([Vector3f](#) target)=0
- virtual [Vector3f](#) **vector3Normalize** ([Vector3f](#))=0
- virtual void **enableCursor** ()=0
- virtual void **disableCursor** ()=0
- virtual float **getFrameTime** ()=0
- virtual int **getFPS** ()=0
- virtual void **updateCameraFreeMode** (float camMovingSpeed, float camRotaSpeed)=0
- virtual InputType **getLastInputType** () const =0
- virtual void **updateLastInputType** ()=0
- virtual float **measureText** (const std::string &text, float fontSize) const =0
- virtual bool **checkCollisionPointRec** ([Vector2f](#) point, [FloatRect](#) rec)=0
- virtual [Ray3D](#) **getMouseRay** ([Vector2f](#) mousePosition)=0
- virtual [RayCollision3D](#) **getRayCollisionBox** ([Ray3D](#) ray, [BoundingBox3D](#) box)=0
- virtual [RayCollision3D](#) **getRayCollisionSphere** ([Ray3D](#) ray, [Vector3f](#) center, float radius)=0
- virtual bool **checkCollisionBoxes** ([BoundingBox3D](#) box1, [BoundingBox3D](#) box2)=0
- virtual [Ray3D](#) **getMouseRayFromCurrent** ()=0
- virtual [BoundingBox3D](#) **createBoundingBox** ([Vector3f](#) center, [Vector3f](#) size)=0
- virtual [BoundingBox3D](#) **createBoundingBoxFromMinMax** ([Vector3f](#) min, [Vector3f](#) max)=0
- virtual void **beginDrawing** ()=0
- virtual void **endDrawing** ()=0
- virtual void **clearBackground** ([Color32](#))=0
- virtual void **begin3DMode** ()=0
- virtual void **end3DMode** ()=0
- virtual void **endScissorMode** ()=0
- virtual void **beginScissorMode** ([IntRect](#))=0
- virtual bool **loadModel** (const std::string &id, const std::string &filepath, [Vector3f](#) center={0.0f, 0.0f, 0.0f})=0
- virtual void **drawCube** ([Vector3f](#) position, float width, float height, float length, [Color32](#) color)=0
- virtual void **drawCubeWires** ([Vector3f](#) position, float width, float height, float length, [Color32](#) color)=0
- virtual void **drawSphere** ([Vector3f](#) position, float radius, [Color32](#) color)=0
- virtual void **drawSphereWires** ([Vector3f](#) position, float radius, int rings, int slices, [Color32](#) color)=0
- virtual void **drawCylinder** ([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color)=0
- virtual void **drawCylinderWires** ([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color)=0

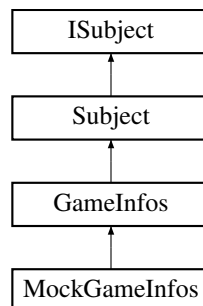
- virtual void **drawCylinderEx** ([Vector3f](#) startPos, [Vector3f](#) endPos, float startRadius, float endRadius, int sides, [Color32](#) color)=0
- virtual void **drawPlane** ([Vector3f](#) position, [Vector2f](#) size, [Color32](#) color)=0
- virtual void **drawLine3D** ([Vector3f](#) startPos, [Vector3f](#) endPos, [Color32](#) color)=0
- virtual void **drawModelEx** (const std::string &id, [Vector3f](#) position, [Vector3f](#) rotationAxis, float rotationAngle, [Vector3f](#) scale, [Color32](#) tint=CWHITE)=0
- virtual void **drawCircle** (float centerX, float centerY, float radius, [Color32](#) color)=0
- virtual void **drawCircleLines** (float centerX, float centerY, float radius, [Color32](#) color)=0
- virtual void **drawText** (const std::string &text, float x, float y, float fontSize, [Color32](#) color)=0
- virtual void **drawTextEx** (const std::string &text, float x, float y, float fontSize, float spacing, [Color32](#) color)=0
- virtual void **drawRectangleRec** ([FloatRect](#) rec, [Color32](#) color)=0
- virtual bool **loadTexture** (const std::string &id, const std::string &filepath)=0
- virtual bool **loadFont** (const std::string &id, const std::string &filepath)=0
- virtual void **drawTexture** (const std::string &id, float x, float y, [Color32](#) tint=CWHITE)=0
- virtual void **drawTextureScaled** (const std::string &id, float x, float y, float width, float height, [Color32](#) tint=CWHITE)=0
- virtual [Vector2f](#) **getTextureSize** (const std::string &id) const =0
- virtual bool **loadSkybox** (const std::string &id, const std::string &filepath)=0
- virtual void **drawSkybox** (const std::string &id)=0
- virtual float **getTime** () const =0

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

- tests/unit/gui/Camera_manager/Camera_manager_test.cpp

6.82 MockGameInfos Class Reference

Inheritance diagram for MockGameInfos:



Public Member Functions

- **MOCK_METHOD** (const std::vector< [zappy::structs::Player](#) > &, getPlayers,(),(const))
- **MOCK_METHOD** (bool, isTeamVisible,(const std::string &teamName),(const))

Public Member Functions inherited from [GameInfos](#)

- **GameInfos** (std::shared_ptr< [ICommunication](#) > communication)
- void **setAudio** (std::shared_ptr< [IAudio](#) > audio)
- void **setCurrentCameraMode** ([zappy::gui::CameraMode](#) cameraMode)
- void **setCurrentPlayerFocus** (int playerId)
- void **setMapSize** (int width, int height)
- std::pair< int, int > **getMapSize** () const
- void **setTimeUnit** (int timeUnit, bool sendToServer=false)
- int **getTimeUnit** () const
- void **updateTile** (const [zappy::structs::Tile](#) tile)

- const [zappy::structs::Tile](#) **getTile** (int x, int y) const
- const [zappy::structs::Tile](#) & **getTileRef** (int x, int y) const
- void **initializeTileMatrix** ()
- void **updateTeamName** (const std::string &teamName)
- const std::vector< std::string > **getTeamNames** () const
- void **setTeamVisibility** (const std::string &teamName, bool visible)
- bool **isTeamVisible** (const std::string &teamName) const
- const std::unordered_map< std::string, bool > **getTeamVisibilities** () const
- void **addPlayer** (const [zappy::structs::Player](#) player)
- void **killPlayer** (int playerNumber)
- void **updatePlayerPosition** (int playerNumber, int x, int y)
- void **updatePlayerOrientation** (int playerNumber, int orientation)
- void **updatePlayerLevel** (int playerNumber, int level)
- void **updatePlayerInventory** (int playerNumber, const [zappy::structs::Inventory](#) inventory)
- void **updatePlayerExpulsion** (int playerNumber)
- void **updatePlayerDeath** (int playerNumber)
- void **updatePlayerResourceAction** (int playerNumber, int resourceId, bool isCollecting)
- void **updatePlayerFork** (int playerNumber)
- const std::vector< [zappy::structs::Player](#) > **getPlayers** () const
- const [zappy::structs::Player](#) **getPlayer** (int playerNumber) const
- void **addPlayerBroadcast** (int playerNumber, const std::string &message)
- const std::vector< std::pair< int, std::string > > **getPlayersBroadcasting** ()
- void **addIncantation** (const [zappy::structs::Incantation](#) incantation)
- void **removeIncantation** (int x, int y, int result)
- const std::vector< [zappy::structs::Incantation](#) > **getIncantations** ()
- void **addEgg** (const [zappy::structs::Egg](#) egg)
- void **updateEggHatched** (int eggNumber)
- void **updateEggDeath** (int eggNumber)
- const std::vector< [zappy::structs::Egg](#) > **getEggs** () const
- void **setGameOver** (const std::string &winningTeam)
- void **playDefeatSound** (const std::string &teamName)
- std::pair< bool, std::string > **isGameOver** () const
- void **addServerMessage** (const std::string &message)
- const std::vector< std::string > **getServerMessages** () const
- void **securityActualisation** ()
- void **incrementPlayerLevel** (int playerNumber)
- void **decrementPlayerLevel** (int playerNumber)
- void **incrementPlayerInventoryItem** (int playerNumber, int resourceId)
- void **decrementPlayerInventoryItem** (int playerNumber, int resourceId)
- void **incrementTileInventoryItem** (int x, int y, int resourceId)
- void **decrementTileInventoryItem** (int x, int y, int resourceId)

Public Member Functions inherited from [Subject](#)

- void **addObserver** (std::shared_ptr< [IObserver](#) > observer) override
- void **removeObserver** (std::shared_ptr< [IObserver](#) > observer) override
- void **notifyObservers** () override
- void **notifyGameEvent** (GameEventType eventType, const std::string &teamName)

Additional Inherited Members

Protected Attributes inherited from [ISubject](#)

- std::vector< std::weak_ptr< [IObserver](#) > > **_observers**

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

- tests/unit/gui/Camera_manager/Camera_manager_test.cpp

6.83 MockGUI Class Reference

Public Member Functions

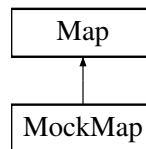
- **MOCK_METHOD** (void, refresh,())
- **MOCK_METHOD** (void, handleVictory,(const std::string &teamName))

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

- tests/unit/gui/Observer/GuiObserver_test.cpp

6.84 MockMap Class Reference

Inheritance diagram for MockMap:



Public Member Functions

- **MOCK_METHOD** ([Vector3f](#), getPlayerInterpolatedPosition,(int playerNumber, int x, int y))
- **MOCK_METHOD** (float, getOffset,(DisplayPriority priority, int x, int y, size_t index))

Public Member Functions inherited from [Map](#)

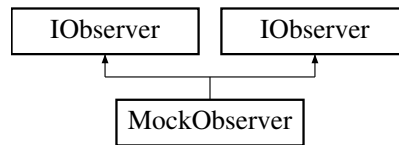
- **Map** (std::shared_ptr< [GameInfos](#) > gameInfos, std::shared_ptr< [IDisplay](#) > display)
- void **draw** (bool performanceMode=false)
- void **drawBroadcastingPlayers** ()
- void **drawIncantations** ()
- void **drawTile** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawPerformanceTile** (const [zappy::structs::Tile](#) &tile)
- void **drawRock** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawPerformanceRock** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawFood** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawPerformanceFood** (int x, int y, const [zappy::structs::Tile](#) &tile)
- void **drawAllPlayers** ()
- void **drawEggs** (int x, int y)
- [Color32](#) **getTeamColor** (const std::string &teamName)
- float **getOffset** (DisplayPriority priority, int x, int y, size_t stackIndex=0)
- void **updatePlayerRotations** ()
- float **getPlayerInterpolatedRotation** (int playerId, int serverOrientation)
- void **updatePlayerPositions** ()
- [Vector3f](#) **getPlayerInterpolatedPosition** (int playerId, int serverX, int serverY)

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

- tests/unit/gui/Camera_manager/Camera_manager_test.cpp

6.85 MockObserver Class Reference

Inheritance diagram for MockObserver:



Public Member Functions

- **MOCK_METHOD** (void, update,(),(override))
- **MOCK_METHOD** (void, onGameEvent,(GameEventType, const std::string &),(override))
- **MOCK_METHOD** (void, update,(),(override))
- **MOCK_METHOD** (void, onGameEvent,(GameEventType eventType, const std::string &teamName),(override))

Public Member Functions inherited from [IObserver](#)

- virtual void **update** ()=0
- virtual void **onGameEvent** (GameEventType eventType, const std::string &teamName)

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

- tests/unit/gui/Game/GameInfos_test.cpp
- tests/unit/gui/Observer/Subject_test.cpp

6.86 MockServer Class Reference

Public Member Functions

- **MockServer** (int port)
- bool **start** ()
- void **stop** ()
- bool **sendToAllClients** (const std::string &message)
- bool **hasClients** () const

Private Member Functions

- void **acceptLoop** ()

Private Attributes

- int **_port**
- bool **_running**
- int **_serverSocket**
- std::thread **_thread**
- std::vector< int > **_clientSockets**

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

- tests/unit/gui/Communication/Communication_test.cpp

6.87 RayLibEnc::ModelData Struct Reference

Public Attributes

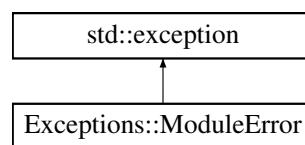
- Model **model**
- unsigned int **animationCount**
- Vector3 **center**

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

- gui/src/RayLib/RaylibEnc/RayLibEnc.hpp

6.88 Exceptions::ModuleError Class Reference

Inheritance diagram for Exceptions::ModuleError:



Public Member Functions

- **ModuleError** (const std::string &msg)
- const char * **what** () const noexcept override

Private Attributes

- std::string **_message** = ""

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

- gui/src/Exceptions/Exceptions.hpp

6.89 MsgHandler Class Reference

Public Member Functions

- **MsgHandler** (std::shared_ptr< [GameInfos](#) > gameInfos, std::shared_ptr< [ICommunication](#) > communication)
- void **start** ()
- void **stop** ()
- void **handleMessage** (const std::string &message)

Protected Member Functions

- void **messageLoop** ()
- bool **handleWelcomeMessage** (const std::string &message)
- bool **handleMszMMessage** (const std::string &message)
- bool **handleBctMessage** (const std::string &message)
- bool **handleTnaMessage** (const std::string &message)
- bool **handlePnwMessage** (const std::string &message)
- bool **handlePpoMessage** (const std::string &message)
- bool **handlePlvMessage** (const std::string &message)
- bool **handlePinMessage** (const std::string &message)
- bool **handlePexMessage** (const std::string &message)

- bool **handlePbcMessage** (const std::string &message)
- bool **handlePicMessage** (const std::string &message)
- bool **handlePieMessage** (const std::string &message)
- bool **handlePfkMessage** (const std::string &message)
- bool **handlePdrMessage** (const std::string &message)
- bool **handlePgtMessage** (const std::string &message)
- bool **handlePdiMessage** (const std::string &message)
- bool **handleEnwMessage** (const std::string &message)
- bool **handleEboMessage** (const std::string &message)
- bool **handleEdiMessage** (const std::string &message)
- bool **handleSgtMessage** (const std::string &message)
- bool **handleSstMessage** (const std::string &message)
- bool **handleSegMessage** (const std::string &message)
- bool **handleSmgMessage** (const std::string &message)
- bool **handleSucMessage** (const std::string &message)
- bool **handleSbpMessage** (const std::string &message)

Private Attributes

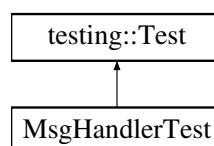
- std::thread **_thread**
- std::atomic< bool > **_running**
- std::mutex **_mutex**
- std::condition_variable **_condition**
- std::shared_ptr< [GameInfos](#) > **_gameInfos**
- std::shared_ptr< [ICommunication](#) > **_communication**
- std::mutex **_gameInfosMutex**
- std::map< std::string, std::function< bool(const std::string &)> > **_messageHandlers**)

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

- gui/src/Client/MsgHandler.hpp
- gui/src/Client/MsgHandler.cpp

6.90 MsgHandlerTest Class Reference

Inheritance diagram for MsgHandlerTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

- std::shared_ptr< [GameInfos](#) > **gameInfos**
- std::shared_ptr< [MockCommunication](#) > **mockCommunication**
- std::unique_ptr< [MsgHandler](#) > **msgHandler**

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

- tests/unit/gui/Client/MsgHandler_test.cpp

6.91 network_s Struct Reference

Public Attributes

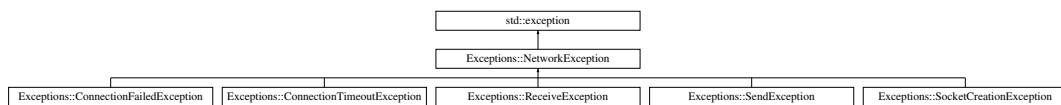
- int **fd**
- [buffer_t](#) * **buffer**

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

- server/include/game.h

6.92 Exceptions::NetworkException Class Reference

Inheritance diagram for Exceptions::NetworkException:



Public Member Functions

- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

Private Attributes

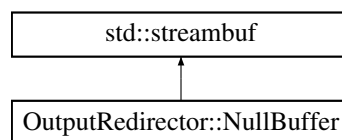
- std::string **_message**

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

- gui/src/Exceptions/Exceptions.hpp

6.93 OutputRedirector::NullBuffer Class Reference

Inheritance diagram for OutputRedirector::NullBuffer:



Protected Member Functions

- int **overflow** (int c) override

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

- tests/unit/gui/main_test.cpp

6.94 OutputRedirector Class Reference

Classes

- class [NullBuffer](#)

Private Attributes

- `std::streambuf *` **originalCout**
- `std::streambuf *` **originalCerr**
- `NullBuffer` **nullBuffer**

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

- `tests/unit/gui/main_test.cpp`

6.95 `params_s` Struct Reference

Public Attributes

- `int` **port**
- `int` **x**
- `int` **y**
- `int` **nb_team**
- `char **` **teams**
- `int` **nb_client**
- `int` **freq**
- `bool` **is_debug**

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

- `server/include/zappy.h`

6.96 `Parser.Parser` Class Reference

Public Member Functions

- `__init__` (self)
- `run` (self)
- `parseConfig` (self)
- `parseJsons` (self)
- `getTests` (self)

Public Attributes

- **tests_folder**
- **tests_files_names**
- **tests_files**
- **output_folder**
- **testsObjects**

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

- `tests/functional/Parser.py`

6.97 `Player.Player` Class Reference

Public Member Functions

- `None` `__init__` (self, str name, str ip, int port=4242)
- `__del__` (self)
- `__str__` (self)
- `int` `create_child` (self)
- `None` `startComThread` (self)

- None **setMapSize** (self, int x, int y)
- list[(str, int)] **getNeededStonesByPriority** (self)
- None **dropStonesForSurvival** (self)
- bool **hasEnoughFoodForIncantation** (self)
- None **roombaAction** (self)
- None **incantationAction** (self)
- list[()] **getStepsFromDirection** (self)
- None **goToIncantationAction** (self)
- None **handleResponseInventory** (self)
- None **handleResponseLook** (self)
- None **handleResponseKO** (self)
- None **handleResponseOK** (self)
- None **handleResponseElevationUnderway** (self)
- None **handleResponseCurrentLevel** (self, str rest)
- None **handleCommandResponse** (self, str response)
- None **handleMessages** (self, int direction, str message)
- None **loop** (self)

Public Attributes

- **logger**
- **is_child_process**
- **x**
- **y**
- **level**
- **look**
- **incantationPhase**
- **incantationLastCommand**
- **canIncant**
- **incantationDirection**
- **inIncantation**
- **inventory**
- **goToIncantation**
- **handleResponseInventory**
- **handleResponseLook**
- **handleResponseKO**
- **handleResponseOK**
- **handleResponseElevationUnderway**
- **handleResponseCurrentLevel**

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

- ai/src/Player/Player.py

6.98 zappy::structs::Player Struct Reference

Public Member Functions

- **Player** (int _number=0, int _x=0, int _y=0, int _orientation=0, int _level=1, const std::string &_teamName="", struct [Inventory](#) _inventory=[Inventory](#)())

Public Attributes

- int **number**
- int **x**
- int **y**
- int **orientation**
- int **level**
- std::string **teamName**
- struct [Inventory](#) **inventory**

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

- `gui/src/Utils/Constants.hpp`

6.99 player_s Struct Reference**Public Attributes**

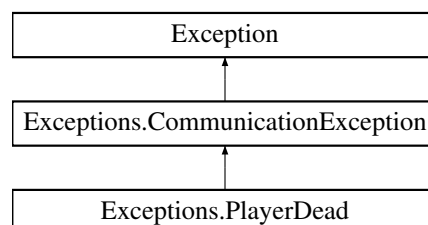
- int **id**
- [network_t](#) * **network**
- int **level**
- int **posX**
- int **posY**
- [direction_t](#) **direction**
- [inventory_t](#) * **inventory**
- char * **team**
- [action_queue_t](#) * **pending_actions**
- [time_t](#) **last_action_time**
- bool **is_busy**
- int **remaining_cooldown**
- char * **current_action**
- int **food_timer**
- [time_t](#) **last_food_check**
- struct [player_s](#) * **next**

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

- `server/include/game.h`

6.100 Exceptions.PlayerDead Class Reference

Inheritance diagram for Exceptions.PlayerDead:

**Public Member Functions**

- [__init__](#) (self)

6.100.1 Constructor & Destructor Documentation

6.100.1.1 `__init__()`

```
Exceptions.PlayerDead.__init__ (
    self )
```

Reimplemented from [Exceptions.CommunicationException](#).

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

- `ai/src/Exceptions/Exceptions.py`

6.101 zappy::gui::PlayerModelInfo Struct Reference

Public Attributes

- `std::string` **name**
- `std::string` **modelPath**
- [Vector3f](#) **center**
- [Vector3f](#) **scale**
- `float` **rotation**

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

- `gui/src/Utils/Constants.hpp`

6.102 PlayerPositionState Struct Reference

Public Attributes

- [Vector3f](#) **currentPosition**
- [Vector3f](#) **targetPosition**
- `bool` **isMoving**
- `std::chrono::steady_clock::time_point` **lastUpdateTime**

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

- `gui/src/Graphic/Map.hpp`

6.103 PlayerRotationState Struct Reference

Public Attributes

- `float` **currentRotation**
- `float` **targetRotation**
- `bool` **isRotating**
- `std::chrono::steady_clock::time_point` **lastUpdateTime**

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

- `gui/src/Graphic/Map.hpp`

6.104 Ray3D Struct Reference

Public Attributes

- [Vector3f](#) **position**
- [Vector3f](#) **direction**

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

- `gui/src/IDisplay.hpp`

6.105 RayCollision3D Struct Reference

Public Attributes

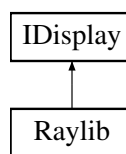
- bool **hit**
- float **distance**
- [Vector3f](#) **point**
- [Vector3f](#) **normal**

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

- `gui/src/IDisplay.hpp`

6.106 Raylib Class Reference

Inheritance diagram for Raylib:



Public Member Functions

- virtual [Vector2i](#) `getMonitorSize ()`
- virtual [Vector2i](#) `getScreenSize ()`
- virtual void `initWindow (int width, int height, std::string)`
- virtual void `initCamera ()`
- virtual bool `isWindowReady ()`
- virtual void `setTargetFPS (unsigned int FPS)`
- virtual bool `isOpen ()`
- virtual void `closeWindow ()`
- virtual int `getKeyId (enum Key)`
- virtual bool `isKeyReleased (int key)`
- virtual bool `isKeyPressed (int key)`
- virtual bool `isKeyDown (int key)`
- virtual bool `isGamepadAvailable ()`
- virtual bool `isGamepadButtonReleased (int key)`
- virtual bool `isGamepadButtonPressed (int key)`
- virtual bool `isGamepadButtonDown (int key)`
- virtual bool `isMouseButtonDown (int key)`
- virtual bool `isMouseButtonReleased (int key)`
- virtual bool `isMouseButtonPressed (int key)`
- virtual [Vector2f](#) `getMousePosition ()`
- virtual void `setMousePosition (Vector2f)`
- virtual float `getMouseWheelMove ()`
- virtual float `getGamepadAxisMovement (int key)`
- virtual void `setCameraPosition (Vector3f)`
- virtual void `setCameraTarget (Vector3f)`
- virtual [Vector2f](#) `getMouseDelta ()`
- virtual float `vector3DDistanceFromCamera (Vector3f target)`
- virtual [Vector3f](#) `vector3SubtractFromCamera (Vector3f target)`
- virtual [Vector3f](#) `vector3Normalize (Vector3f)`
- virtual void `enableCursor ()`
- virtual void `disableCursor ()`

- virtual float [getFrameTime](#) ()
- virtual int [getFPS](#) ()
- virtual void [updateCameraFreeMode](#) (float camMovingSpeed, float camRotaSpeed)
- virtual InputType [getLastInputType](#) () const
- virtual void [updateLastInputType](#) ()
- virtual float [measureText](#) (const std::string &text, float fontSize) const
- virtual bool [checkCollisionPointRec](#) (Vector2f point, [FloatRect](#) rec)
- virtual [Ray3D](#) [getMouseRay](#) (Vector2f mousePosition)
- virtual [RayCollision3D](#) [getRayCollisionBox](#) ([Ray3D](#) ray, [BoundingBox3D](#) box)
- virtual [RayCollision3D](#) [getRayCollisionSphere](#) ([Ray3D](#) ray, [Vector3f](#) center, float radius)
- virtual bool [checkCollisionBoxes](#) ([BoundingBox3D](#) box1, [BoundingBox3D](#) box2)
- virtual [Ray3D](#) [getMouseRayFromCurrent](#) ()
- virtual [BoundingBox3D](#) [createBoundingBox](#) ([Vector3f](#) center, [Vector3f](#) size)
- virtual [BoundingBox3D](#) [createBoundingBoxFromMinMax](#) ([Vector3f](#) min, [Vector3f](#) max)
- virtual void [beginScissorMode](#) ([IntRect](#))
- virtual void [endScissorMode](#) ()
- virtual void [beginDrawing](#) ()
- virtual void [endDrawing](#) ()
- virtual void [clearBackground](#) ([Color32](#))
- virtual void [begin3DMode](#) ()
- virtual void [end3DMode](#) ()
- virtual bool [loadModel](#) (const std::string &id, const std::string &filepath, [Vector3f](#) center={0.0f, 0.0f, 0.0f})
- virtual void [drawCube](#) ([Vector3f](#) position, float width, float height, float length, [Color32](#) color)
- virtual void [drawCubeWires](#) ([Vector3f](#) position, float width, float height, float length, [Color32](#) color)
- virtual void [drawSphere](#) ([Vector3f](#) position, float radius, [Color32](#) color)
- virtual void [drawSphereWires](#) ([Vector3f](#) position, float radius, int rings, int slices, [Color32](#) color)
- virtual void [drawCylinder](#) ([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color)
- virtual void [drawCylinderWires](#) ([Vector3f](#) position, float radiusTop, float radiusBottom, float height, int slices, [Color32](#) color)
- virtual void [drawCylinderEx](#) ([Vector3f](#) startPos, [Vector3f](#) endPos, float startRadius, float endRadius, int sides, [Color32](#) color)
- virtual void [drawPlane](#) ([Vector3f](#) position, [Vector2f](#) size, [Color32](#) color)
- virtual void [drawLine3D](#) ([Vector3f](#) startPos, [Vector3f](#) endPos, [Color32](#) color)
- virtual void [drawModelEx](#) (const std::string &id, [Vector3f](#) position, [Vector3f](#) rotationAxis, float rotationAngle, [Vector3f](#) scale, [Color32](#) tint=CWHITE)
- virtual void [drawText](#) (const std::string &text, float x, float y, float fontSize, [Color32](#) color)
- virtual void [drawTextEx](#) (const std::string &text, float x, float y, float fontSize, float spacing, [Color32](#) color)
- virtual void [drawCircle](#) (float centerX, float centerY, float radius, [Color32](#) color)
- virtual void [drawCircleLines](#) (float centerX, float centerY, float radius, [Color32](#) color)
- virtual void [drawRectangleRec](#) ([FloatRect](#) rec, [Color32](#) color)
- virtual bool [loadTexture](#) (const std::string &id, const std::string &filepath)
- virtual bool [loadFont](#) (const std::string &id, const std::string &filepath)
- virtual void [drawTexture](#) (const std::string &id, float x, float y, [Color32](#) tint=CWHITE)
- virtual void [drawTextureScaled](#) (const std::string &id, float x, float y, float width, float height, [Color32](#) tint=CWHITE)
- virtual [Vector2f](#) [getTextureSize](#) (const std::string &id) const
- virtual bool [loadSkybox](#) (const std::string &id, const std::string &filepath)
- virtual void [drawSkybox](#) (const std::string &id)
- virtual float [getTime](#) () const

Private Attributes

- std::unique_ptr< [RayLibEnc](#) > [_raylib](#)

6.106.1 Member Function Documentation

6.106.1.1 begin3DMode()

```
void Raylib::begin3DMode ( ) [virtual]  
Implements IDisplay.
```

6.106.1.2 beginDrawing()

```
void Raylib::beginDrawing ( ) [virtual]  
Implements IDisplay.
```

6.106.1.3 beginScissorMode()

```
void Raylib::beginScissorMode (   
    IntRect data ) [virtual]  
Implements IDisplay.
```

6.106.1.4 checkCollisionBoxes()

```
bool Raylib::checkCollisionBoxes (   
    BoundingBox3D box1,   
    BoundingBox3D box2 ) [virtual]  
Implements IDisplay.
```

6.106.1.5 checkCollisionPointRec()

```
bool Raylib::checkCollisionPointRec (   
    Vector2f point,   
    FloatRect rec ) [virtual]  
Implements IDisplay.
```

6.106.1.6 clearBackground()

```
void Raylib::clearBackground (   
    Color32 color ) [virtual]  
Implements IDisplay.
```

6.106.1.7 closeWindow()

```
void Raylib::closeWindow ( ) [virtual]  
Implements IDisplay.
```

6.106.1.8 createBoundingBox()

```
BoundingBox3D Raylib::createBoundingBox (   
    Vector3f center,   
    Vector3f size ) [virtual]  
Implements IDisplay.
```

6.106.1.9 createBoundingBoxFromMinMax()

```
BoundingBox3D Raylib::createBoundingBoxFromMinMax (   
    Vector3f min,   
    Vector3f max ) [virtual]  
Implements IDisplay.
```

6.106.1.10 disableCursor()

```
void Raylib::disableCursor ( ) [virtual]  
Implements IDisplay.
```


6.106.1.11 drawCircle()

```
void Raylib::drawCircle (
    float centerX,
    float centerY,
    float radius,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.12 drawCircleLines()

```
void Raylib::drawCircleLines (
    float centerX,
    float centerY,
    float radius,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.13 drawCube()

```
void Raylib::drawCube (
    Vector3f position,
    float width,
    float height,
    float length,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.14 drawCubeWires()

```
void Raylib::drawCubeWires (
    Vector3f position,
    float width,
    float height,
    float length,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.15 drawCylinder()

```
void Raylib::drawCylinder (
    Vector3f position,
    float radiusTop,
    float radiusBottom,
    float height,
    int slices,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.16 drawCylinderEx()

```
void Raylib::drawCylinderEx (
    Vector3f startPos,
    Vector3f endPos,
    float startRadius,
    float endRadius,
    int sides,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.17 drawCylinderWires()

```
void Raylib::drawCylinderWires (
    Vector3f position,
    float radiusTop,
    float radiusBottom,
    float height,
    int slices,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.18 drawLine3D()

```
void Raylib::drawLine3D (
    Vector3f startPos,
    Vector3f endPos,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.19 drawModelEx()

```
void Raylib::drawModelEx (
    const std::string & id,
    Vector3f position,
    Vector3f rotationAxis,
    float rotationAngle,
    Vector3f scale,
    Color32 tint = CWHITE ) [virtual]
```

Implements [IDisplay](#).

6.106.1.20 drawPlane()

```
void Raylib::drawPlane (
    Vector3f position,
    Vector2f size,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.21 drawRectangleRec()

```
void Raylib::drawRectangleRec (
    FloatRect rec,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.22 drawSkybox()

```
void Raylib::drawSkybox (
    const std::string & id ) [virtual]
```

Implements [IDisplay](#).

6.106.1.23 drawSphere()

```
void Raylib::drawSphere (
    Vector3f position,
    float radius,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.24 drawSphereWires()

```
void Raylib::drawSphereWires (
    Vector3f position,
    float radius,
    int rings,
    int slices,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.25 drawText()

```
void Raylib::drawText (
    const std::string & text,
    float x,
    float y,
    float fontSize,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.26 drawTextEx()

```
void Raylib::drawTextEx (
    const std::string & text,
    float x,
    float y,
    float fontSize,
    float spacing,
    Color32 color ) [virtual]
```

Implements [IDisplay](#).

6.106.1.27 drawTexture()

```
void Raylib::drawTexture (
    const std::string & id,
    float x,
    float y,
    Color32 tint = CWHITE ) [virtual]
```

Implements [IDisplay](#).

6.106.1.28 drawTextureScaled()

```
void Raylib::drawTextureScaled (
    const std::string & id,
    float x,
    float y,
    float width,
    float height,
    Color32 tint = CWHITE ) [virtual]
```

Implements [IDisplay](#).

6.106.1.29 enableCursor()

```
void Raylib::enableCursor ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.30 end3DMode()

```
void Raylib::end3DMode ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.31 endDrawing()

```
void Raylib::endDrawing ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.32 endScissorMode()

```
void Raylib::endScissorMode ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.33 getFPS()

```
int Raylib::getFPS ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.34 getFrameTime()

```
float Raylib::getFrameTime ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.35 getGamepadAxisMovement()

```
float Raylib::getGamepadAxisMovement (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.36 getKeyId()

```
int Raylib::getKeyId (
    enum Key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.37 getLastInputType()

```
InputType Raylib::getLastInputType ( ) const [virtual]
```

Implements [IDisplay](#).

6.106.1.38 getMonitorSize()

```
Vector2i Raylib::getMonitorSize ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.39 getMouseDelta()

```
Vector2f Raylib::getMouseDelta ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.40 getMousePosition()

```
Vector2f Raylib::getMousePosition ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.41 getMouseRay()

```
Ray3D Raylib::getMouseRay (
    Vector2f mousePosition ) [virtual]
```

Implements [IDisplay](#).

6.106.1.42 getMouseRayFromCurrent()

`Ray3D` Raylib::getMouseRayFromCurrent () [virtual]
Implements [IDisplay](#).

6.106.1.43 getMouseWheelMove()

`float` Raylib::getMouseWheelMove () [virtual]
Implements [IDisplay](#).

6.106.1.44 getRayCollisionBox()

`RayCollision3D` Raylib::getRayCollisionBox (
 `Ray3D` ray,
 `BoundingBox3D` box) [virtual]
Implements [IDisplay](#).

6.106.1.45 getRayCollisionSphere()

`RayCollision3D` Raylib::getRayCollisionSphere (
 `Ray3D` ray,
 `Vector3f` center,
 `float` radius) [virtual]
Implements [IDisplay](#).

6.106.1.46 getScreenSize()

`Vector2i` Raylib::getScreenSize () [virtual]
Implements [IDisplay](#).

6.106.1.47 getTextureSize()

`Vector2f` Raylib::getTextureSize (
 `const std::string & id`) `const` [virtual]
Implements [IDisplay](#).

6.106.1.48 getTime()

`float` Raylib::getTime () `const` [virtual]
Implements [IDisplay](#).

6.106.1.49 initCamera()

`void` Raylib::initCamera () [virtual]
Implements [IDisplay](#).

6.106.1.50 initWindow()

`void` Raylib::initWindow (
 `int` width,
 `int` height,
 `std::string` title) [virtual]
Implements [IDisplay](#).

6.106.1.51 isGamepadAvailable()

`bool` Raylib::isGamepadAvailable () [virtual]
Implements [IDisplay](#).

6.106.1.52 isGamepadButtonDown()

```
bool Raylib::isGamepadButtonDown (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.53 isGamepadButtonPressed()

```
bool Raylib::isGamepadButtonPressed (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.54 isGamepadButtonReleased()

```
bool Raylib::isGamepadButtonReleased (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.55 isKeyDown()

```
bool Raylib::isKeyDown (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.56 isKeyPressed()

```
bool Raylib::isKeyPressed (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.57 isKeyReleased()

```
bool Raylib::isKeyReleased (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.58 isMouseButtonDown()

```
bool Raylib::isMouseButtonDown (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.59 isMouseButtonPressed()

```
bool Raylib::isMouseButtonPressed (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.60 isMouseButtonReleased()

```
bool Raylib::isMouseButtonReleased (
    int key ) [virtual]
```

Implements [IDisplay](#).

6.106.1.61 isOpen()

```
bool Raylib::isOpen ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.62 isWindowReady()

```
bool Raylib::isWindowReady ( ) [virtual]
Implements IDisplay.
```

6.106.1.63 loadFont()

```
bool Raylib::loadFont (
    const std::string & id,
    const std::string & filepath ) [virtual]
Implements IDisplay.
```

6.106.1.64 loadModel()

```
bool Raylib::loadModel (
    const std::string & id,
    const std::string & filepath,
    Vector3f center = {0.0f, 0.0f, 0.0f} ) [virtual]
Implements IDisplay.
```

6.106.1.65 loadSkybox()

```
bool Raylib::loadSkybox (
    const std::string & id,
    const std::string & filepath ) [virtual]
Implements IDisplay.
```

6.106.1.66 loadTexture()

```
bool Raylib::loadTexture (
    const std::string & id,
    const std::string & filepath ) [virtual]
Implements IDisplay.
```

6.106.1.67 measureText()

```
float Raylib::measureText (
    const std::string & text,
    float fontSize ) const [virtual]
Implements IDisplay.
```

6.106.1.68 setCameraPosition()

```
void Raylib::setCameraPosition (
    Vector3f pos ) [virtual]
Implements IDisplay.
```

6.106.1.69 setCameraTarget()

```
void Raylib::setCameraTarget (
    Vector3f pos ) [virtual]
Implements IDisplay.
```

6.106.1.70 setMousePosition()

```
void Raylib::setMousePosition (
    Vector2f pos ) [virtual]
Implements IDisplay.
```

6.106.1.71 setTargetFPS()

```
void Raylib::setTargetFPS (
    unsigned int FPS ) [virtual]
```

Implements [IDisplay](#).

6.106.1.72 updateCameraFreeMode()

```
void Raylib::updateCameraFreeMode (
    float camMovingSpeed,
    float camRotaSpeed ) [virtual]
```

Implements [IDisplay](#).

6.106.1.73 updateLastInputType()

```
void Raylib::updateLastInputType ( ) [virtual]
```

Implements [IDisplay](#).

6.106.1.74 vector3DDistanceFromCamera()

```
float Raylib::vector3DDistanceFromCamera (
    Vector3f target ) [virtual]
```

Implements [IDisplay](#).

6.106.1.75 vector3Normalize()

```
Vector3f Raylib::vector3Normalize (
    Vector3f vec ) [virtual]
```

Implements [IDisplay](#).

6.106.1.76 vector3SubtractFromCamera()

```
Vector3f Raylib::vector3SubtractFromCamera (
    Vector3f target ) [virtual]
```

Implements [IDisplay](#).

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

- `gui/src/RayLib/Raylib.hpp`
- `gui/src/RayLib/Raylib.cpp`

6.107 RayLibEnc Class Reference**Classes**

- struct [ModelData](#)

Public Member Functions

- void **initWindow** (int width, int height, const std::string &title)
- void **closeWindow** ()
- bool **windowShouldClose** () const
- void **beginDrawing** ()
- void **endDrawing** ()
- void **clearBackground** (Color color=WHITE)
- bool **isWindowReady** () const
- int **getMonitorWidth** (int monitor) const
- int **getMonitorHeight** (int monitor) const
- void **waitTime** (float seconds) const
- void **setTargetFPS** (int fps) const

- int **getFPS** () const
- float **getFrameTime** () const
- bool **checkCollisionPointRec** (Vector2 point, Rectangle rec) const
- Ray **getMouseRay** (Vector2 mousePosition)
- RayCollision **getRayCollisionBox** (Ray ray, BoundingBox box)
- RayCollision **getRayCollisionSphere** (Ray ray, Vector3 center, float radius)
- bool **checkCollisionBoxes** (BoundingBox box1, BoundingBox box2)
- Ray **getMouseRayFromCurrent** ()
- BoundingBox **createBoundingBox** (Vector3 center, Vector3 size)
- BoundingBox **createBoundingBoxFromMinMax** (Vector3 min, Vector3 max)
- void **drawTextureRec** (Texture2D texture, Rectangle source, Vector2 position, Color tint)
- void **unloadTexture** (Texture2D texture)
- Texture2D **loadTextureFromFile** (const std::string &filepath)
- void **drawTextureEx** (Texture2D texture, Vector2 position, Color tint)
- void **drawTextureScaled** (Texture2D texture, float x, float y, float width, float height, Color tint)
- bool **hasTexture** (const std::string &id) const
- Texture2D **getTexture** (const std::string &id) const
- void **addTexture** (const std::string &id, Texture2D texture)
- bool **isMouseButtonDown** (int button) const
- bool **isMouseButtonPressed** (int button) const
- bool **isMouseButtonReleased** (int button) const
- bool **isKeyDown** (int key) const
- bool **isKeyPressed** (int key) const
- bool **isKeyReleased** (int key) const
- Vector2 **getMouseDelta** ()
- Vector2 **getMousePosition** () const
- void **setMousePosition** (int x, int y)
- void **disableCursor** ()
- void **enableCursor** ()
- int **getScreenWidth** () const
- int **getScreenHeight** () const
- float **getMouseWheelMove** () const
- bool **isGamepadAvailable** (int gamepad) const
- bool **isGamepadButtonPressed** (int gamepad, int button) const
- bool **isGamepadButtonDown** (int gamepad, int button) const
- bool **isGamepadButtonReleased** (int gamepad, int button) const
- float **getGamepadAxisMovement** (int gamepad, int axis) const
- InputType **getLastInputType** () const
- void **updateLastInputType** ()
- void **beginScissorMode** (int x, int y, int width, int height)
- void **endScissorMode** ()
- void **begin3DMode** ()
- void **end3DMode** ()
- float **vector3Distance** (Vector3 v1, Vector3 v2) const
- Vector3 **vector3Normalize** (Vector3 v) const
- Vector3 **vector3Subtract** (Vector3 v1, Vector3 v2) const
- Vector3 **vector3Add** (Vector3 v1, Vector3 v2) const
- void **initCamera** ()
- void **setCameraPosition** (Vector3 position)
- void **setCameraTarget** (Vector3 target)
- void **setCameraUp** (Vector3 up)
- void **setCameraFovy** (float fovy)
- void **setCameraProjection** (int projection)
- void **updateCamera** (int mode=CAMERA_FREE)
- void **updateCameraFreeMode** (float camMovingSpeed, float camRotaSpeed)

- Camera3D **getCamera** () const
- void **drawGrid** (int slices, float spacing)
- void **drawCube** (Vector3 position, float width, float height, float length, Color color)
- void **drawCubeWires** (Vector3 position, float width, float height, float length, Color color)
- void **drawSphere** (Vector3 position, float radius, Color color)
- void **drawSphereWires** (Vector3 position, float radius, int rings, int slices, Color color)
- void **drawCylinder** (Vector3 position, float radiusTop, float radiusBottom, float height, int slices, Color color)
- void **drawCylinderWires** (Vector3 position, float radiusTop, float radiusBottom, float height, int slices, Color color)
- void **drawCylinderEx** (Vector3 startPos, Vector3 endPos, float startRadius, float endRadius, int sides, Color color)
- void **drawPlane** (Vector3 position, Vector2 size, Color color)
- void **drawLine3D** (Vector3 startPos, Vector3 endPos, Color color)
- bool **loadModel** (const std::string &id, const std::string &filepath, Vector3 center={0.0f, 0.0f, 0.0f})
- void **drawModel** (const std::string &id, Vector3 position, float scale, Color tint=WHITE)
- void **drawModelEx** (const std::string &id, Vector3 position, Vector3 rotationAxis, float rotationAngle, Vector3 scale, Color tint=WHITE)
- void **drawModelWires** (const std::string &id, Vector3 position, float scale, Color tint=WHITE)
- void **drawModelWiresEx** (const std::string &id, Vector3 position, Vector3 rotationAxis, float rotationAngle, Vector3 scale, Color tint=WHITE)
- void **unloadModel** (const std::string &id)
- void **unloadAllModels** ()
- bool **modelExists** (const std::string &id) const
- bool **loadSkybox** (const std::string &id, const std::string &filepath)
- void **drawSkybox** (const std::string &id)
- Color **getDayNightColor** (float cycleTime)
- float **getTime** () const
- void **drawRectangleRec** (Rectangle rec, Color color)
- void **drawText** (const std::string &text, float x, float y, float fontSize, Color color)
- void **drawTextEx** (const std::string &text, float x, float y, float fontSize, float spacing, Color color)
- void **drawCircle** (float centerX, float centerY, float radius, Color color)
- void **drawCircleLines** (float centerX, float centerY, float radius, Color color)
- float **measureText** (const std::string &text, float fontSize) const
- float **measureTextEx** (const std::string &text, float fontSize, float spacing) const
- bool **loadFont** (const std::string &id, const std::string &filepath)
- void **unloadFont** (const std::string &id)
- bool **hasFontLoaded** (const std::string &id) const
- Font **getFont** (const std::string &id) const
- void **unloadAllFonts** ()

Private Member Functions

- float **getScaledFontSize** (float fontSize) const
- float **getFontSpacing** (float scaledFontSize) const
- float **getScaledSpacing** (float spacing) const

Private Attributes

- bool **_isInitialized**
- Camera3D **_camera**
- Vector2 **_previousMousePosition**
- bool **_isCursorLocked**
- InputType **_lastInputType**
- std::map< std::string, [ModelData](#) > **_models**
- std::map< std::string, Texture2D > **_textures**
- std::map< std::string, Sound > **_sounds**
- std::map< std::string, Music > **_musics**
- std::map< std::string, Font > **_fonts**

Static Private Attributes

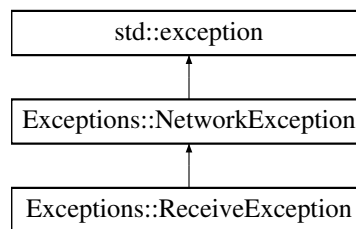
- static constexpr float **FONT_SCALE_FACTOR** = 4.0f
- static constexpr float **FONT_RENDER_SCALE** = 0.25f
- static constexpr float **FONT_SPACING_RATIO** = 0.1f

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

- gui/src/RayLib/RaylibEnc/RayLibEnc.hpp
- gui/src/RayLib/RaylibEnc/Raylib3dDrawing.cpp
- gui/src/RayLib/RaylibEnc/Raylib3dEnv.cpp
- gui/src/RayLib/RaylibEnc/Raylib3dModel.cpp
- gui/src/RayLib/RaylibEnc/RaylibCamera.cpp
- gui/src/RayLib/RaylibEnc/RaylibCollision3D.cpp
- gui/src/RayLib/RaylibEnc/RayLibEnc.cpp
- gui/src/RayLib/RaylibEnc/RaylibGamepad.cpp
- gui/src/RayLib/RaylibEnc/RaylibInput.cpp
- gui/src/RayLib/RaylibEnc/RaylibSkybox.cpp
- gui/src/RayLib/RaylibEnc/RaylibTextures.cpp
- gui/src/RayLib/RaylibEnc/RaylibWindow.cpp

6.108 Exceptions::ReceiveException Class Reference

Inheritance diagram for Exceptions::ReceiveException:

**Public Member Functions**

- **ReceiveException** (const std::string &message)

Public Member Functions inherited from [Exceptions::NetworkException](#)

- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.109 RelativePosition Struct Reference**Public Attributes**

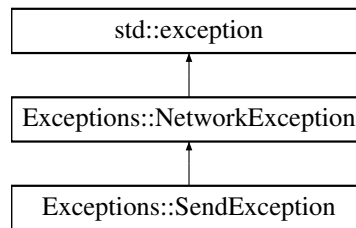
- float **xPercent**
- float **yPercent**
- float **widthPercent**
- float **heightPercent**

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

- gui/src/Graphic/HUD/Containers/AContainers.hpp

6.110 Exceptions::SendException Class Reference

Inheritance diagram for Exceptions::SendException:



Public Member Functions

- **SendException** (const std::string &message)

Public Member Functions inherited from [Exceptions::NetworkException](#)

- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.111 server_s Struct Reference

Public Attributes

- int **sockfd**
- struct pollfd **pollserver**

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

- server/include/zappy.h

6.112 Settings Class Reference

Public Member Functions

- bool **isVisible** () const
- bool **containsPoint** (float x, float y) const
- void **show** ()
- void **hide** ()
- void **update** ()
- void **draw** ()
- void **handleResize** (int oldWidth, int oldHeight, int newWidth, int newHeight)
- **Settings** (std::shared_ptr< [IDisplay](#) > display, std::shared_ptr< [IAudio](#) > audio, std::shared_ptr< [CameraManager](#) > camera)

Private Attributes

- std::shared_ptr< [IDisplay](#) > **_display**
- std::shared_ptr< [IAudio](#) > **_audio**
- std::shared_ptr< [CameraManager](#) > **_camera**
- float **_sfxLevel**
- float **_musicLevel**

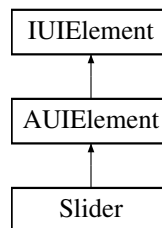
- float **_cameraMovingSpeed**
- float **_cameraRotaSpeed**
- float **_cameraZoomSpeed**
- std::shared_ptr< [Containers](#) > **_settingsContainer**
- bool **_visible**

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

- gui/src/Graphic/HUD/Settings/Settings.hpp
- gui/src/Graphic/HUD/Settings/Settings.cpp

6.113 Slider Class Reference

Inheritance diagram for Slider:



Public Member Functions

- **Slider** (std::shared_ptr< [IDisplay](#) > raylib, float x, float y, float width, float height, float min**Value**, float max↔
Value, float initial**Value**, const std::string &text, std::function< void(float)> on**Value**Changed)
- void [draw](#) () override
- void [update](#) () override
- bool **isDragging** () const
- void **setValue** (float value)
- float **getValue** () const
- void **setMinValue** (float min**Value**)
- void **setMaxValue** (float max**Value**)
- float **getMinValue** () const
- float **getMaxValue** () const
- void **setText** (const std::string &text)
- std::string **getText** () const
- void [setSize](#) (float width, float height) override

Public Member Functions inherited from [AUIElement](#)

- **AUIElement** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void [setPosition](#) (float x, float y) override
- [FloatRect](#) [getBounds](#) () const override
- bool [contains](#) (float x, float y) const override
- void [setVisible](#) (bool visible) override
- bool [isVisible](#) () const override
- void **setRelativePosition** (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [UIRelativePosition](#) [getRelativePosition](#) () const

Private Member Functions

- void **updateValueFromMousePosition** (float mouseX)
- float **getHandlePosition** () const
- bool **isMouseOverHandle** (float mouseX, float mouseY) const
- bool **isMouseOverTrack** (float mouseX, float mouseY) const

Private Attributes

- float **_value**
- float **_minValue**
- float **_maxValue**
- std::string **_text**
- std::function< void(float)> **_onValueChanged**
- bool **_isDragging**
- float **_sliderTrackWidth**
- float **_sliderHandleRadius**
- [Color32](#) **_trackColor**
- [Color32](#) **_fillColor**
- [Color32](#) **_handleColor**
- [Color32](#) **_textColor**

Additional Inherited Members**Protected Attributes inherited from [UIElement](#)**

- std::shared_ptr< [IDisplay](#) > **_display**
- [FloatRect](#) **_bounds**
- [UIRelativePosition](#) **_relativePos**
- bool **_visible**

6.113.1 Member Function Documentation**6.113.1.1 draw()**

void Slider::draw () [override], [virtual]
Implements [UIElement](#).

6.113.1.2 setSize()

```
void Slider::setSize (
    float width,
    float height ) [override], [virtual]
```

Reimplemented from [UIElement](#).

6.113.1.3 update()

void Slider::update () [override], [virtual]
Implements [UIElement](#).

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

- gui/src/Graphic/HUD/Slider/Slider.hpp
- gui/src/Graphic/HUD/Slider/Slider.cpp

6.114 Socket.Socket Class Reference**Public Member Functions**

- **__init__** (self, str host, int port)
- **connect** (self)
- int **get_fd** (self)
- **send** (self, str content)
- str **receive** (self)
- **close** (self)

Protected Attributes

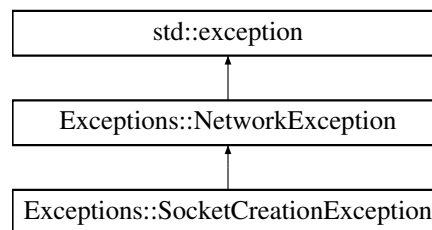
- `_host`
- `_port`
- `_address`
- `_socket`

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

- `ai/src/Communication/Socket.py`

6.115 Exceptions::SocketCreationException Class Reference

Inheritance diagram for Exceptions::SocketCreationException:

**Public Member Functions**

- **SocketCreationException** (const std::string &message)

Public Member Functions inherited from [Exceptions::NetworkException](#)

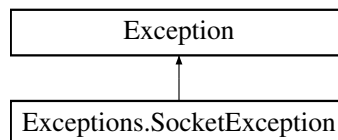
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- `gui/src/Exceptions/Exceptions.hpp`

6.116 Exceptions.SocketException Class Reference

Inheritance diagram for Exceptions.SocketException:

**Public Member Functions**

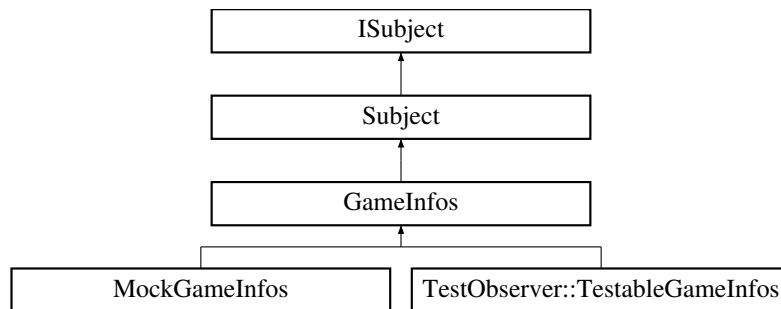
- **__init__** (self, str message)

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

- `ai/src/Exceptions/Exceptions.py`

6.117 Subject Class Reference

Inheritance diagram for Subject:



Public Member Functions

- void [addObserver](#) (std::shared_ptr< [IObserver](#) > observer) override
- void [removeObserver](#) (std::shared_ptr< [IObserver](#) > observer) override
- void [notifyObservers](#) () override
- void [notifyGameEvent](#) (GameEventType eventType, const std::string &teamName)

Private Attributes

- std::vector< std::weak_ptr< [IObserver](#) > > [_observers](#)

Additional Inherited Members

Protected Attributes inherited from [ISubject](#)

- std::vector< std::weak_ptr< [IObserver](#) > > [_observers](#)

6.117.1 Member Function Documentation

6.117.1.1 addObserver()

```
void Subject::addObserver (
    std::shared_ptr< IObserver > observer ) [override], [virtual]
```

Implements [ISubject](#).

6.117.1.2 notifyGameEvent()

```
void Subject::notifyGameEvent (
    GameEventType eventType,
    const std::string & teamName ) [virtual]
```

Implements [ISubject](#).

6.117.1.3 notifyObservers()

```
void Subject::notifyObservers ( ) [override], [virtual]
```

Implements [ISubject](#).

6.117.1.4 removeObserver()

```
void Subject::removeObserver (
    std::shared_ptr< IObserver > observer ) [override], [virtual]
```

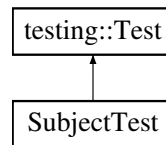
Implements [ISubject](#).

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

- gui/src/Observer/Subject.hpp
- gui/src/Observer/Subject.cpp

6.118 SubjectTest Class Reference

Inheritance diagram for SubjectTest:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

- std::unique_ptr< [Subject](#) > **subject**
- std::shared_ptr< [MockObserver](#) > **observer1**
- std::shared_ptr< [MockObserver](#) > **observer2**
- std::shared_ptr< [MockObserver](#) > **observer3**

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

- tests/unit/gui/Observer/Subject_test.cpp

6.119 team_s Struct Reference

Public Attributes

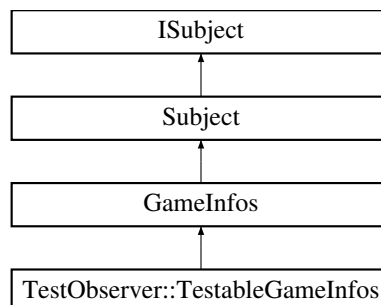
- char * **name**
- int **nbPlayers**
- int **nbPlayerAlive**
- [player_t](#) * **players**
- struct [team_s](#) * **next**

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

- server/include/game.h

6.120 TestObserver::TestableGameInfos Class Reference

Inheritance diagram for TestObserver::TestableGameInfos:



Public Member Functions

- **TestableGameInfos** (std::shared_ptr< [ICommunication](#) > communication)
- void **testNotifyObservers** ()

Public Member Functions inherited from [GameInfos](#)

- **GameInfos** (std::shared_ptr< [ICommunication](#) > communication)
- void **setAudio** (std::shared_ptr< [IAudio](#) > audio)
- void **setCurrentCameraMode** (zappy::gui::CameraMode cameraMode)
- void **setCurrentPlayerFocus** (int playerId)
- void **setMapSize** (int width, int height)
- std::pair< int, int > **getMapSize** () const
- void **setTimeUnit** (int timeUnit, bool sendToServer=false)
- int **getTimeUnit** () const
- void **updateTile** (const [zappy::structs::Tile](#) tile)
- const [zappy::structs::Tile](#) **getTile** (int x, int y) const
- const [zappy::structs::Tile](#) & **getTileRef** (int x, int y) const
- void **initializeTileMatrix** ()
- void **updateTeamName** (const std::string &teamName)
- const std::vector< std::string > **getTeamNames** () const
- void **setTeamVisibility** (const std::string &teamName, bool visible)
- bool **isTeamVisible** (const std::string &teamName) const
- const std::unordered_map< std::string, bool > **getTeamVisibilities** () const
- void **addPlayer** (const [zappy::structs::Player](#) player)
- void **killPlayer** (int playerNumber)
- void **updatePlayerPosition** (int playerNumber, int x, int y)
- void **updatePlayerOrientation** (int playerNumber, int orientation)
- void **updatePlayerLevel** (int playerNumber, int level)
- void **updatePlayerInventory** (int playerNumber, const [zappy::structs::Inventory](#) inventory)
- void **updatePlayerExpulsion** (int playerNumber)
- void **updatePlayerDeath** (int playerNumber)
- void **updatePlayerResourceAction** (int playerNumber, int resourceId, bool isCollecting)
- void **updatePlayerFork** (int playerNumber)
- const std::vector< [zappy::structs::Player](#) > **getPlayers** () const
- const [zappy::structs::Player](#) **getPlayer** (int playerNumber) const
- void **addPlayerBroadcast** (int playerNumber, const std::string &message)
- const std::vector< std::pair< int, std::string > > **getPlayersBroadcasting** ()
- void **addIncantation** (const [zappy::structs::Incantation](#) incantation)
- void **removeIncantation** (int x, int y, int result)
- const std::vector< [zappy::structs::Incantation](#) > **getIncantations** ()
- void **addEgg** (const [zappy::structs::Egg](#) egg)
- void **updateEggHatched** (int eggNumber)
- void **updateEggDeath** (int eggNumber)
- const std::vector< [zappy::structs::Egg](#) > **getEggs** () const
- void **setGameOver** (const std::string &winningTeam)
- void **playDefeatSound** (const std::string &teamName)
- std::pair< bool, std::string > **isGameOver** () const
- void **addServerMessage** (const std::string &message)
- const std::vector< std::string > **getServerMessages** () const
- void **securityActualisation** ()
- void **incrementPlayerLevel** (int playerNumber)
- void **decrementPlayerLevel** (int playerNumber)
- void **incrementPlayerInventoryItem** (int playerNumber, int resourceId)
- void **decrementPlayerInventoryItem** (int playerNumber, int resourceId)
- void **incrementTileInventoryItem** (int x, int y, int resourceId)
- void **decrementTileInventoryItem** (int x, int y, int resourceId)

Public Member Functions inherited from [Subject](#)

- void [addObserver](#) (std::shared_ptr< [IObserver](#) > observer) override
- void [removeObserver](#) (std::shared_ptr< [IObserver](#) > observer) override
- void [notifyObservers](#) () override
- void [notifyGameEvent](#) (GameEventType eventType, const std::string &teamName)

Additional Inherited Members

Protected Attributes inherited from [ISubject](#)

- std::vector< std::weak_ptr< [IObserver](#) > > **_observers**

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

- tests/unit/gui/Game/GameInfos_test.cpp

6.121 TestCase.TestCase Class Reference

Public Member Functions

- **__init__** (self, name, desc, input, output, value, output_folder)
- **execute** (self)
- **check** (self)
- **displayPassed** (self, index)
- **displayFailed** (self, index)

Public Attributes

- **name**
- **desc**
- **input**
- **output**
- **value**
- **tty_mode**
- **tty_input**
- **succeed_after**
- **succeed_forced**
- **real_output**
- **real_value**
- **raw_output**

Protected Member Functions

- **_execute_normal** (self)
- **_execute_tty** (self)

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

- tests/functional/TestCase.py

6.122 test_cli.TestCLI Class Reference

Public Member Functions

- [test_parse_args_valid](#) (self)
- [test_parse_args_valid_ip](#) (self)
- [test_parse_args_invalid_option](#) (self)

- [test_parse_args_missing_value](#) (self)
- [test_parse_args_not_enough_args](#) (self)
- [test_parse_port_invalid](#) (self)
- [test_parse_port_negative](#) (self)
- [test_parse_port_too_large](#) (self)
- [test_parse_name_empty](#) (self)
- [test_parse_name_whitespace](#) (self)
- [test_parse_machine_empty](#) (self)
- [test_parse_machine_invalid_ip_format](#) (self)
- [test_parse_machine_invalid_ip_value](#) (self)
- [test_parse_machine_invalid_ip_chars](#) (self)
- [test_validate_config_missing_port](#) (self)
- [test_validate_config_missing_name](#) (self)

6.122.1 Member Function Documentation

6.122.1.1 test_parse_args_invalid_option()

```
test_cli.TestCLI.test_parse_args_invalid_option (
    self )
```

Test parsing invalid option

6.122.1.2 test_parse_args_missing_value()

```
test_cli.TestCLI.test_parse_args_missing_value (
    self )
```

Test parsing missing value for option

6.122.1.3 test_parse_args_not_enough_args()

```
test_cli.TestCLI.test_parse_args_not_enough_args (
    self )
```

Test parsing not enough arguments

6.122.1.4 test_parse_args_valid()

```
test_cli.TestCLI.test_parse_args_valid (
    self )
```

Test parsing valid command line arguments

6.122.1.5 test_parse_args_valid_ip()

```
test_cli.TestCLI.test_parse_args_valid_ip (
    self )
```

Test parsing valid IP address

6.122.1.6 test_parse_machine_empty()

```
test_cli.TestCLI.test_parse_machine_empty (
    self )
```

Test parsing empty machine name

6.122.1.7 test_parse_machine_invalid_ip_chars()

```
test_cli.TestCLI.test_parse_machine_invalid_ip_chars (
    self )
```

Test parsing IP with invalid characters

6.122.1.8 test_parse_machine_invalid_ip_format()

```
test_cli.TestCLI.test_parse_machine_invalid_ip_format (
    self )
```

Test parsing invalid IP format

6.122.1.9 test_parse_machine_invalid_ip_value()

```
test_cli.TestCLI.test_parse_machine_invalid_ip_value (
    self )
```

Test parsing invalid IP value

6.122.1.10 test_parse_name_empty()

```
test_cli.TestCLI.test_parse_name_empty (
    self )
```

Test parsing empty team name

6.122.1.11 test_parse_name_whitespace()

```
test_cli.TestCLI.test_parse_name_whitespace (
    self )
```

Test parsing whitespace team name

6.122.1.12 test_parse_port_invalid()

```
test_cli.TestCLI.test_parse_port_invalid (
    self )
```

Test parsing invalid port

6.122.1.13 test_parse_port_negative()

```
test_cli.TestCLI.test_parse_port_negative (
    self )
```

Test parsing negative port

6.122.1.14 test_parse_port_too_large()

```
test_cli.TestCLI.test_parse_port_too_large (
    self )
```

Test parsing port that is too large

6.122.1.15 test_validate_config_missing_name()

```
test_cli.TestCLI.test_validate_config_missing_name (
    self )
```

Test validating config with missing name

6.122.1.16 test_validate_config_missing_port()

```
test_cli.TestCLI.test_validate_config_missing_port (
    self )
```

Test validating config with missing port

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

- tests/unit/ai/CLI/test_cli.py

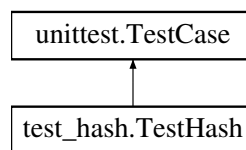
6.123 test_com.TestCommunication Class Reference

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

- tests/unit/ai/Communication/test_com.py

6.124 test_hash.TestHash Class Reference

Inheritance diagram for test_hash.TestHash:

**Public Member Functions**

- **setUp** (self)
- **test_hash_initialization** (self)
- **test_simple_xor** (self)
- **test_hash_message** (self)
- **test_unhash_message** (self)
- **test_hash_unhash_roundtrip** (self)
- **test_different_keys_produce_different_hashes** (self)

Public Attributes

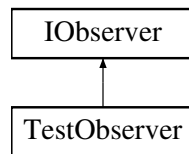
- **hash_obj**

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

- tests/unit/ai/Hash/test_hash.py

6.125 TestObserver Class Reference

Inheritance diagram for TestObserver:



Classes

- class [TestableGameInfos](#)

Public Member Functions

- **MOCK_METHOD** (void, update,(),(override))
- **MOCK_METHOD** (void, onGameEvent,(GameEventType, const std::string &),(override))

Public Member Functions inherited from [IObserver](#)

- virtual void **update** ()=0
- virtual void **onGameEvent** (GameEventType eventType, const std::string &teamName)

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

- tests/unit/gui/Game/GameInfos_test.cpp

6.126 test_player.TestPlayer Class Reference

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

- tests/unit/ai/Player/test_player.py

6.127 test_socket.TestSocket Class Reference

Public Member Functions

- [test_socket_init](#) (self)
- [test_socket_connect_success](#) (self, mock_socket)
- [test_socket_connect_failure](#) (self, mock_socket)
- [test_socket_send_success](#) (self, mock_socket)
- [test_socket_send_unicode](#) (self, mock_socket)
- [test_socket_receive_connection_closed](#) (self, mock_socket)
- [test_socket_receive_unicode](#) (self, mock_socket)
- [test_socket_close](#) (self, mock_socket)
- [test_socket_different_hosts_and_ports](#) (self)

6.127.1 Member Function Documentation

6.127.1.1 test_socket_close()

```

test_socket.TestSocket.test_socket_close (
    self,
    mock_socket )

```

Test socket close

6.127.1.2 test_socket_connect_failure()

```
test_socket.TestSocket.test_socket_connect_failure (
    self,
    mock_socket )
```

Test socket connection failure

6.127.1.3 test_socket_connect_success()

```
test_socket.TestSocket.test_socket_connect_success (
    self,
    mock_socket )
```

Test successful socket connection

6.127.1.4 test_socket_different_hosts_and_ports()

```
test_socket.TestSocket.test_socket_different_hosts_and_ports (
    self )
```

Test socket creation with different hosts and ports

6.127.1.5 test_socket_init()

```
test_socket.TestSocket.test_socket_init (
    self )
```

Test socket initialization

6.127.1.6 test_socket_receive_connection_closed()

```
test_socket.TestSocket.test_socket_receive_connection_closed (
    self,
    mock_socket )
```

Test handling closed connection during receive

6.127.1.7 test_socket_receive_unicode()

```
test_socket.TestSocket.test_socket_receive_unicode (
    self,
    mock_socket )
```

Test receiving unicode messages

6.127.1.8 test_socket_send_success()

```
test_socket.TestSocket.test_socket_send_success (
    self,
    mock_socket )
```

Test successful message sending

6.127.1.9 test_socket_send_unicode()

```
test_socket.TestSocket.test_socket_send_unicode (
    self,
    mock_socket )
```

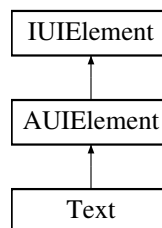
Test sending unicode messages

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

- tests/unit/ai/Communication/test_socket.py

6.128 Text Class Reference

Inheritance diagram for Text:



Public Member Functions

- **Text** (std::shared_ptr< [IDisplay](#) > raylib, float x, float y, const std::string &text, float fontSize=20.0f, [Color32](#) color=CBLACK)
- void [draw](#) () override
- void [update](#) () override
- void **setText** (const std::string &text)
- std::string **getText** () const
- void **setFontSize** (float fontSize)
- float **getFontSize** () const
- void **setColor** ([Color32](#) color)
- [Color32](#) **getColor** () const
- void [setSize](#) (float width, float height) override
- float **getWidth** () const
- void **setX** (float x)
- void **setY** (float y)

Public Member Functions inherited from [AUIElement](#)

- **AUIElement** (std::shared_ptr< [IDisplay](#) > display, float x, float y, float width, float height)
- void [setPosition](#) (float x, float y) override
- [FloatRect](#) [getBounds](#) () const override
- bool [contains](#) (float x, float y) const override
- void [setVisible](#) (bool visible) override
- bool [isVisible](#) () const override
- void **setRelativePosition** (float xPercent, float yPercent, float widthPercent, float heightPercent)
- [UIRelativePosition](#) **getRelativePosition** () const

Private Attributes

- std::string **_text**
- float **_fontSize**
- [Color32](#) **_color**
- std::shared_ptr< [IDisplay](#) > **_display**

Additional Inherited Members

Protected Attributes inherited from [AUIElement](#)

- `std::shared_ptr< IDisplay > _display`
- `FloatRect _bounds`
- `UIRelativePosition _relativePos`
- `bool _visible`

6.128.1 Member Function Documentation

6.128.1.1 `draw()`

`void Text::draw () [override], [virtual]`

Implements [IUIElement](#).

6.128.1.2 `setSize()`

```
void Text::setSize (
    float width,
    float height ) [override], [virtual]
```

Reimplemented from [AUIElement](#).

6.128.1.3 `update()`

`void Text::update () [override], [virtual]`

Implements [IUIElement](#).

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

- `gui/src/Graphic/HUD/Text/Text.hpp`
- `gui/src/Graphic/HUD/Text/Text.cpp`

6.129 `zappy::structs::Tile` Struct Reference

Public Member Functions

- **Tile** (`int _x=0`, `int _y=0`, `int _food=0`, `int _linemate=0`, `int _deraumere=0`, `int _sibur=0`, `int _mendiane=0`, `int _phiras=0`, `int _thystame=0`)

Public Attributes

- `int x`
- `int y`
- `int food`
- `int linemate`
- `int deraumere`
- `int sibur`
- `int mendiane`
- `int phiras`
- `int thystame`

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

- `gui/src/Utils/Constants.hpp`

6.130 tiles_s Struct Reference

Public Attributes

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

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

- server/include/algo.h

6.131 UIRelativePosition Struct Reference

Public Attributes

- float **xPercent**
- float **yPercent**
- float **widthPercent**
- float **heightPercent**

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

- gui/src/Graphic/HUD/UIElement/AUIElement.hpp

6.132 unified_poll_s Struct Reference

Public Attributes

- struct pollfd * **fds**
- int **count**
- int **capacity**

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

- server/include/zappy.h

6.133 Vector2f Struct Reference

Public Attributes

- float **x**
- float **y**

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

- gui/src/IDisplay.hpp

6.134 Vector2i Struct Reference

Public Attributes

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

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

- gui/src/IDisplay.hpp

6.135 Vector3f Struct Reference

Public Member Functions

- bool **operator==** (const [Vector3f](#) &other) const
- bool **operator!=** (const [Vector3f](#) &other) const

Public Attributes

- float **x**
- float **y**
- float **z**

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

- gui/src/IDisplay.hpp

6.136 zappy_s Struct Reference

Public Attributes

- [server_t](#) * **network**
- [game_t](#) * **game**
- [graph_net_t](#) * **graph**
- [params_t](#) * **params**
- [unified_poll_t](#) * **unified_poll**

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

- server/include/zappy.h

Chapter 7

File Documentation

7.1 Audio.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Audio
00006 */
00007
00008 #ifndef AUDIO_HPP_
00009 #define AUDIO_HPP_
00010
00011 #include <string>
00012 #include <map>
00013 #include <memory>
00014 #include <vector>
00015 #include <SFML/Audio.hpp>
00016 #include "IAudio.hpp"
00017
00018 class Audio : public IAudio {
00019     private:
00020         std::vector<std::string> _musicId = {"main_theme", "main_theme2"};
00021         std::vector<std::string> _sfxId = {"click", "clickPlayer"};
00022         std::map<std::string, std::unique_ptr<sf::Music> _sounds;
00023         float _levelSFX = 75.f;
00024         float _levelMusic = 50.f;
00025         int _themeIndex = 0;
00026
00027     public:
00028         Audio();
00029         ~Audio();
00030
00031         float getSFXVolumeLevel();
00032         float getMusicVolumeLevel();
00033
00034         void setSFXVolumeLevel(float);
00035         void setMusicVolumeLevel(float);
00036
00037         bool loadSound(const std::string& id, const std::string& filepath);
00038
00039         void playMainTheme(float volume);
00040         void playNextTheme(float volume);
00041
00042         void playSound(const std::string& id, float volume);
00043         void stopSound(const std::string& id);
00044         bool isSoundPlaying(const std::string& id) const;
00045
00046         void setSoundLooping(const std::string& id, bool looping);
00047         void setSoundVolume(const std::string& id, float volume);
00048 };
00049
00050 #endif /* !AUDIO_HPP_ */
```

7.2 IAudio.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** IAudio
00006 */
```

```

00007
00008 #ifndef IAUDIO_HPP_
00009 #define IAUDIO_HPP_
00010
00011 #include <string>
00012
00013 class IAudio {
00014     public:
00015         virtual ~IAudio() = default;
00016
00017         virtual float getSFXVolumeLevel() = 0;
00018         virtual float getMusicVolumeLevel() = 0;
00019
00020         virtual void setSFXVolumeLevel(float) = 0;
00021         virtual void setMusicVolumeLevel(float) = 0;
00022
00023         virtual bool loadSound(const std::string& id, const std::string& filepath) = 0;
00024
00025         virtual void playMainTheme(float volume) = 0;
00026         virtual void playNextTheme(float volume) = 0;
00027
00028         virtual void playSound(const std::string& id, float volume) = 0;
00029         virtual void stopSound(const std::string& id) = 0;
00030         virtual bool isSoundPlaying(const std::string& id) const = 0;
00031
00032         virtual void setSoundLooping(const std::string& id, bool looping) = 0;
00033         virtual void setSoundVolume(const std::string& id, float volume) = 0;
00034 };
00035
00036 #endif /* !IAUDIO_HPP_ */

```

7.3 CLI.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** CLI
00006 */
00007
00008 #ifndef CLI_HPP_
00009 #define CLI_HPP_
00010
00011 #include <string>
00012 #include "../Utils/Constants.hpp"
00013
00014 class CLI {
00015     public:
00016         CLI(int ac, const char *const *av);
00017         ~CLI();
00018
00019         zappy::structs::Config parseArguments(int ac, const char *const *av) const;
00020
00021     private:
00022         int _ac;
00023         const char *const *_av;
00024
00025         bool hasCorrectNumberOfArguments(int ac) const;
00026         int parsePort(const char *portStr) const;
00027         std::string parseHostname(const char *hostnameStr) const;
00028         void validateConfig(bool portFound, bool hostFound) const;
00029 };
00030
00031 #endif /* !CLI_HPP_ */

```

7.4 Client.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Client
00006 */
00007
00008 #ifndef CLIENT_HPP_
00009 #define CLIENT_HPP_
00010
00011 #include <memory>
00012 #include <filesystem>
00013 #include <string>
00014 #include "../Utils/Constants.hpp"
00015 #include "../Communication/ICommunication.hpp"

```

```

00016 #include "../Game/GameInfos.hpp"
00017 #include "../Graphic/GUI.hpp"
00018 #include "MsgHandler.hpp"
00019 #include "../Observer/GuiObserver.hpp"
00020 #include "../Observer/IObserver.hpp"
00021
00022 class Client {
00023     public:
00024         Client(int ac, const char *const *av);
00025         ~Client();
00026
00027         void tryToCreateGuiWithSharedLibInFolder(const std::string &libPath);
00028
00029     private:
00030         zappy::structs::Config _config;
00031         void initialize(int ac, const char * const *av);
00032
00033         std::shared_ptr<ICommunication> _communication;
00034         std::shared_ptr<GameInfos> _gameInfos;
00035         std::unique_ptr<MsgHandler> _msgHandler;
00036         std::shared_ptr<GUI> _gui;
00037         std::shared_ptr<GuiObserver> _guiObserver;
00038 };
00039
00040 #endif /* !CLIENT_HPP_ */

```

7.5 MsgHandler.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** MsgHandler
00006 */
00007
00008 #ifndef MSGHANDLER_HPP_
00009 #define MSGHANDLER_HPP_
00010
00011 #include <memory>
00012 #include <map>
00013 #include <functional>
00014 #include <thread>
00015 #include <mutex>
00016 #include <atomic>
00017 #include <queue>
00018 #include <condition_variable>
00019 #include <string>
00020
00021 #include "../Game/GameInfos.hpp"
00022 #include "../Communication/ICommunication.hpp"
00023 #include "../Utils/Constants.hpp"
00024
00025 class MsgHandler {
00026     public:
00027         MsgHandler(std::shared_ptr<GameInfos> gameInfos,
00028                 std::shared_ptr<ICommunication> communication);
00029         ~MsgHandler();
00030
00031         void start();
00032         void stop();
00033
00034         void handleMessage(const std::string& message);
00035
00036     protected:
00037         void messageLoop();
00038
00039         bool handleWelcomeMessage(const std::string& message);
00040         bool handleMszMMessage(const std::string& message);
00041         bool handleBctMessage(const std::string& message);
00042         bool handleTnaMessage(const std::string& message);
00043         bool handlePnwMessage(const std::string& message);
00044         bool handlePpoMessage(const std::string& message);
00045         bool handlePlvMessage(const std::string& message);
00046         bool handlePinMessage(const std::string& message);
00047         bool handlePexMessage(const std::string& message);
00048         bool handlePbcMessage(const std::string& message);
00049         bool handlePicMessage(const std::string& message);
00050         bool handlePieMessage(const std::string& message);
00051         bool handlePfkMessage(const std::string& message);
00052         bool handlePdrMessage(const std::string& message);
00053         bool handlePgtMessage(const std::string& message);
00054         bool handlePdiMessage(const std::string& message);
00055         bool handleEnwMessage(const std::string& message);
00056         bool handleEboMessage(const std::string& message);
00057         bool handleEdiMessage(const std::string& message);

```

```

00058         bool handleSgtMessage(const std::string& message);
00059         bool handleSstMessage(const std::string& message);
00060         bool handleSegMessage(const std::string& message);
00061         bool handleSmgMessage(const std::string& message);
00062         bool handleSucMessage(const std::string& message);
00063         bool handleSbpMessage(const std::string& message);
00064
00065     private:
00066         std::thread _thread;
00067         std::atomic<bool> _running;
00068         std::mutex _mutex;
00069         std::condition_variable _condition;
00070
00071         std::shared_ptr<GameInfos> _gameInfos;
00072         std::shared_ptr<ICommunication> _communication;
00073         std::mutex _gameInfosMutex;
00074
00075         std::map<std::string, std::function<bool(const std::string&)>> _messageHandlers;
00076 };
00077
00078 #endif /* !MSGHANDLER_HPP_ */

```

7.6 Communication.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Communication
00006  */
00007
00008 #ifndef COMMUNICATION_HPP_
00009 #define COMMUNICATION_HPP_
00010
00011 #include <sys/socket.h>
00012 #include <netinet/in.h>
00013 #include <arpa/inet.h>
00014 #include <unistd.h>
00015 #include <fcntl.h>
00016 #include <poll.h>
00017 #include <netdb.h>
00018 #include <thread>
00019 #include <mutex>
00020 #include <atomic>
00021 #include <condition_variable>
00022 #include <queue>
00023 #include <string>
00024 #include <vector>
00025
00026 #include "../Utils/Constants.hpp"
00027 #include "../Exceptions/Exceptions.hpp"
00028 #include "ICommunication.hpp"
00029
00030 class Communication : public ICommunication {
00031     public:
00032         explicit Communication(zappy::structs::Config config);
00033         ~Communication();
00034
00035         void sendMessage(const std::string &message) override;
00036         bool hasMessages() const override;
00037         std::string popMessage() override;
00038         bool isConnected() const override;
00039         void disconnect() override;
00040
00041     private:
00042         void setupConnection();
00043         void createSocket();
00044         void connectToServer();
00045         void setupNonBlocking();
00046
00047         void startCommunicationThread();
00048         void communicationLoop();
00049         bool handlePoll();
00050         void processWrite();
00051         void processRead();
00052
00053         void parseReceivedData();
00054
00055         zappy::structs::Config _config;
00056         std::thread _thread;
00057         std::mutex _mutex;
00058         std::condition_variable _cv;
00059         std::atomic<bool> _running;
00060         std::atomic<bool> _connected;
00061

```



```

00062         std::queue<std::string> _outgoingMessages;
00063         std::queue<std::string> _incomingMessages;
00064
00065         std::string _receiveBuffer;
00066         std::string _sendBuffer;
00067
00068         int _socket;
00069         struct pollfd _pollfd;
00070         static const int BUFFER_SIZE = 4096;
00071         static const int POLL_TIMEOUT = 100;
00072         static const char MESSAGE_DELIMITER = '\n';
00073     };
00074
00075 #endif /* !COMMUNICATION_HPP_ */

```

7.7 ICommunication.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** ICommunication
00006 */
00007
00008 #ifndef ICOMMUNICATION_HPP_
00009 #define ICOMMUNICATION_HPP_
00010
00011 #include <string>
00012
00013 class ICommunication {
00014     public:
00015         virtual ~ICommunication() = default;
00016
00017         virtual void sendMessage(const std::string &message) = 0;
00018         virtual bool hasMessages() const = 0;
00019         virtual std::string popMessage() = 0;
00020         virtual bool isConnected() const = 0;
00021         virtual void disconnect() = 0;
00022 };
00023
00024 #endif /* !ICOMMUNICATION_HPP_ */

```

7.8 DLLoader.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** DLLoader
00006 */
00007
00008 #ifndef DLLOADER_HPP_
00009 #define DLLOADER_HPP_
00010
00011 #include <dlfcn.h>
00012 #include <iostream>
00013 #include <ostream>
00014 #include <memory>
00015 #include "ILoader.hpp"
00016
00017 template <typename T>
00018
00019 class DLLoader : public ILoader {
00020     private:
00021         void *_handler = nullptr;
00022
00023     public:
00024         ~DLLoader() = default;
00025
00026         void *getHandler() const override {
00027             return _handler;
00028         };
00029         void *Open(const char *path, int flag = RTLD_LAZY) override {
00030             _handler = dlopen(path, flag);
00031             return _handler;
00032         };
00033         void *Symbol(const char *symbolName) override {
00034             void *symbol = dlsym(_handler, symbolName);
00035             const char *error = dlerror();
00036             if (error) {
00037                 std::cerr << "dlerror: " << error << std::endl;
00038                 return nullptr;
00039             }
00040         };
00041     };

```

```

00039         }
00040         return symbol;
00041     };
00042     T getSymbol(const char *symbolName) {
00043         return reinterpret_cast<T>(dlsym(_handler, symbolName));
00044     };
00045     int Close() override{
00046         if (_handler == nullptr)
00047             return -1;
00048         return dlclose(_handler);
00049     };
00050     const char *Error() override {
00051         return dlerror();
00052     };
00053 };
00054
00055 #endif /* !DLLOADER_HPP_ */

```

7.9 ILoader.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** ILoader
00006 */
00007
00008 #ifndef ILoader_HPP_
00009 #define ILoader_HPP_
00010
00011 class ILoader {
00012     public:
00013         ~ILoader() = default;
00014
00015         virtual void *Open(const char *path, int flag) = 0;
00016         virtual void *Symbol(const char *symbolName) = 0;
00017         virtual int Close() = 0;
00018         virtual const char *Error() = 0;
00019         virtual void *getHandler() const = 0;
00020
00021     protected:
00022     private:
00023 };
00024
00025 #endif /* !ILoader_HPP_ */

```

7.10 LoaderType.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** LoaderType
00006 */
00007
00008 #ifndef LOADERTYPE_HPP_
00009 #define LOADERTYPE_HPP_
00010
00011 enum ModuleType_t{
00012     DISPLAY_MODULE,
00013     NONE
00014 };
00015
00016 #endif /* !LOADERTYPE_HPP_ */

```

7.11 Exceptions.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Exceptions
00006 */
00007
00008 #ifndef EXCEPTIONS_HPP_
00009 #define EXCEPTIONS_HPP_
00010
00011 #include <exception>

```

```

00012 #include <string>
00013 #include "../Utils/Constants.hpp"
00014
00015 namespace Exceptions {
00016
00017     // CLI Exceptions
00018     class CLIParsingException : public std::exception {
00019     public:
00020         explicit CLIParsingException(const std::string &message)
00021             : _message(std::string(colors::T_RED) +
00022                 "CLI Parsing Error: " + message +
00023                 colors::RESET) {}
00024
00025         const char *what() const noexcept override {
00026             return _message.c_str();
00027         }
00028
00029     private:
00030         std::string _message;
00031     };
00032
00033     class CLIPortException : public CLIParsingException {
00034     public:
00035         explicit CLIPortException(const std::string &message)
00036             : CLIParsingException(std::string(colors::T_CYAN) +
00037                 "Port Error: " + message +
00038                 colors::RESET) {}
00039     };
00040
00041     class CLIHostException : public CLIParsingException {
00042     public:
00043         explicit CLIHostException(const std::string &message)
00044             : CLIParsingException(std::string(colors::T_CYAN) +
00045                 "Hostname Error: " + message +
00046                 colors::RESET) {}
00047     };
00048
00049     class CLIMissingArgumentException : public CLIParsingException {
00050     public:
00051         explicit CLIMissingArgumentException(const std::string &message)
00052             : CLIParsingException(std::string(colors::T_CYAN) +
00053                 "Missing Argument: " + message +
00054                 colors::RESET) {}
00055     };
00056
00057     class CLIInvalidArgumentException : public CLIParsingException {
00058     public:
00059         explicit CLIInvalidArgumentException(const std::string &message)
00060             : CLIParsingException(std::string(colors::T_CYAN) +
00061                 "Invalid Argument: " + message +
00062                 colors::RESET) {}
00063     };
00064
00065     class NetworkException : public std::exception {
00066     public:
00067         explicit NetworkException(const std::string &message)
00068             : _message(std::string(colors::T_RED) +
00069                 "Network Error: " + message +
00070                 colors::RESET) {}
00071
00072         const char *what() const noexcept override {
00073             return _message.c_str();
00074         }
00075
00076     private:
00077         std::string _message;
00078     };
00079
00080     class ConnectionFailedException : public NetworkException {
00081     public:
00082         explicit ConnectionFailedException(const std::string &message)
00083             : NetworkException(std::string(colors::T_CYAN) +
00084                 "Connection Failed: " + message +
00085                 colors::RESET) {}
00086     };
00087
00088     class SocketCreationException : public NetworkException {
00089     public:
00090         explicit SocketCreationException(const std::string &message)
00091             : NetworkException(std::string(colors::T_CYAN) +
00092                 "Socket Creation Failed: " + message +
00093                 colors::RESET) {}
00094     };
00095
00096     class ConnectionTimeoutException : public NetworkException {
00097     public:
00098         explicit ConnectionTimeoutException(const std::string &message)

```

```

00099         : NetworkException(std::string(colors::T_CYAN) +
00100                             "Connection Timeout: " + message +
00101                             colors::RESET) {}
00102     };
00103
00104     class SendException : public NetworkException {
00105     public:
00106         explicit SendException(const std::string &message)
00107             : NetworkException(std::string(colors::T_CYAN) +
00108                               "Send Error: " + message +
00109                               colors::RESET) {}
00110     };
00111
00112     class ReceiveException : public NetworkException {
00113     public:
00114         explicit ReceiveException(const std::string &message)
00115             : NetworkException(std::string(colors::T_CYAN) +
00116                               "Receive Error: " + message +
00117                               colors::RESET) {}
00118     };
00119
00120     class ModuleError : public std::exception {
00121     private:
00122         std::string _message = "";
00123     public:
00124         explicit ModuleError(const std::string &msg) : _message(msg) {};
00125         const char *what() const noexcept override {
00126             return this->_message.c_str();
00127         }
00128     };
00129 }
00130
00131 #endif /* !EXCEPTIONS_HPP_ */

```

7.12 GameInfos.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GameInfos
00006 */
00007
00008 #ifndef GAMEINFOS_HPP_
00009 #define GAMEINFOS_HPP_
00010
00011 #include <utility>
00012 #include <vector>
00013 #include <memory>
00014 #include <mutex>
00015 #include <string>
00016 #include <chrono>
00017 #include <unordered_map>
00018
00019 #include "../Utils/Constants.hpp"
00020 #include "../Communication/ICommunication.hpp"
00021 #include "../Observer/Subject.hpp"
00022 #include "../Audio/IAudio.hpp"
00023
00024 class GameInfos : public Subject {
00025     public:
00026         explicit GameInfos(std::shared_ptr<ICommunication> communication);
00027         ~GameInfos();
00028
00029         void setAudio(std::shared_ptr<IAudio> audio);
00030         void setCurrentCameraMode(zappy::gui::CameraMode cameraMode);
00031         void setCurrentPlayerFocus(int playerId);
00032
00033         void setMapSize(int width, int height);
00034         std::pair<int, int> getMapSize() const;
00035
00036         void setTimeUnit(int timeUnit, bool sendToServer = false);
00037         int getTimeUnit() const;
00038
00039         void updateTile(const zappy::structs::Tile tile);
00040         const zappy::structs::Tile getTile(int x, int y) const;
00041         const zappy::structs::Tile& getTileRef(int x, int y) const;
00042         void initializeTileMatrix();
00043
00044         void updateTeamName(const std::string &teamName);
00045         const std::vector<std::string> getTeamNames() const;
00046
00047         void setTeamVisibility(const std::string &teamName, bool visible);
00048         bool isTeamVisible(const std::string &teamName) const;
00049         const std::unordered_map<std::string, bool> getTeamVisibilities() const;

```

```

00050
00051     void addPlayer(const zappy::structs::Player player);
00052     void killPlayer(int playerNumber);
00053     void updatePlayerPosition(int playerNumber, int x, int y);
00054     void updatePlayerOrientation(int playerNumber, int orientation);
00055     void updatePlayerLevel(int playerNumber, int level);
00056     void updatePlayerInventory(int playerNumber,
00057         const zappy::structs::Inventory inventory);
00058     void updatePlayerExpulsion(int playerNumber);
00059     void updatePlayerDeath(int playerNumber);
00060     void updatePlayerResourceAction(int playerNumber, int resourceId, bool isCollecting);
00061     void updatePlayerFork(int playerNumber);
00062     const std::vector<zappy::structs::Player> getPlayers() const;
00063     const zappy::structs::Player getPlayer(int playerNumber) const;
00064
00065     void addPlayerBroadcast(int playerNumber, const std::string &message);
00066     const std::vector<std::pair<int, std::string>> getPlayersBroadcasting();
00067
00068     void addIncantation(const zappy::structs::Incantation incantation);
00069     void removeIncantation(int x, int y, int result);
00070     const std::vector<zappy::structs::Incantation> getIncantations();
00071
00072     void addEgg(const zappy::structs::Egg egg);
00073     void updateEggHatched(int eggNumber);
00074     void updateEggDeath(int eggNumber);
00075     const std::vector<zappy::structs::Egg> getEggs() const;
00076
00077     void setGameOver(const std::string &winningTeam);
00078     void playDefeatSound(const std::string &teamName);
00079     std::pair<bool, std::string> isGameOver() const;
00080
00081     void addServerMessage(const std::string &message);
00082     const std::vector<std::string> getServerMessages() const;
00083
00084     void securityActualisation();
00085     void incrementPlayerLevel(int playerNumber);
00086     void decrementPlayerLevel(int playerNumber);
00087     void incrementPlayerInventoryItem(int playerNumber, int resourceId);
00088     void decrementPlayerInventoryItem(int playerNumber, int resourceId);
00089     void incrementTileInventoryItem(int x, int y, int resourceId);
00090     void decrementTileInventoryItem(int x, int y, int resourceId);
00091
00092 private:
00093     int _mapWidth;
00094     int _mapHeight;
00095     int _timeUnit;
00096
00097     std::vector<std::vector<zappy::structs::Tile>> _tileMatrix;
00098     bool _matrixInitialized;
00099     std::vector<std::string> _teamNames;
00100     std::unordered_map<std::string, bool> _teamVisibilities;
00101     std::vector<zappy::structs::Player> _players;
00102     std::vector<std::pair<int, bool>> _playersExpulsing;
00103     std::vector<std::tuple<int, std::string, std::chrono::steady_clock::time_point>>
00104         _playersBroadcasting;
00105     std::vector<zappy::structs::Incantation> _incantations;
00106     std::vector<zappy::structs::Egg> _eggs;
00107     std::vector<std::string> _serverMessages;
00108
00109     bool _gameOver;
00110     std::string _winningTeam;
00111     bool _victorySoundPlayed;
00112
00113     mutable std::mutex _dataMutex;
00114
00115     std::shared_ptr<ICommunication> _communication;
00116     std::shared_ptr<IAudio> _audio;
00117     zappy::gui::CameraMode _currentCameraMode;
00118     int _currentPlayerFocus;
00119
00120     void notifyStateChange();
00121 };
00122
00123 #endif /* !GAMEINFOS_HPP_ */

```

7.13 CameraManager.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** CameraManager
00006  */
00007
00008 #ifndef CAMERA_MANAGER_HPP_

```

```

00009 #define CAMERA_MANAGER_HPP_
00010
00011 #include <memory>
00012 #include "../Utils/Constants.hpp"
00013 #include "../Game/GameInfos.hpp"
00014 #include "../Map.hpp"
00015
00016 class CameraManager {
00017 public:
00018     explicit CameraManager(std::shared_ptr<IDisplay> display);
00019     ~CameraManager();
00020
00021     void updateCamera(zappy::gui::CameraMode mode);
00022     void updateCameraFreeMode();
00023     void updateCameraTargetMode();
00024     void updateCameraPlayerMode();
00025
00026     void setMapCenter(const Vector3f &center);
00027     void setMapSize(int width, int height);
00028
00029     void setTargetDistance(float distance);
00030     void initTargetPositionFromCurrentCamera();
00031
00032     void setPlayerId(int playerId);
00033     int getPlayerId() const;
00034     void setGameInfos(std::shared_ptr<GameInfos> gameInfos);
00035     void setMapInstance(std::shared_ptr<Map> map);
00036
00037     float getCameraMovingSpeed();
00038     void setCameraMovingSpeed(float);
00039     float getCameraRotaSpeed();
00040     void setCameraRotaSpeed(float);
00041     float getCameraZoomSpeed();
00042     void setCameraZoomSpeed(float);
00043
00044     Vector3f calculatePlayerPosition(const zappy::structs::Player& player);
00045     Vector3f calculateCameraPosition(const Vector3f& playerPos, float angleXZ);
00046
00047 private:
00048     float _cameraMovingSpeed = 15.0f;
00049     float _cameraRotaSpeed = 2.0f;
00050     float _cameraZoomSpeed = 120.0f;
00051     std::shared_ptr<IDisplay> _display;
00052     std::shared_ptr<GameInfos> _gameInfos;
00053     std::shared_ptr<Map> _map;
00054     Vector3f _mapCenter;
00055     int _mapWidth;
00056     int _mapHeight;
00057
00058     float _targetDistance;
00059     float _targetAngleXZ;
00060     float _targetAngleY;
00061     bool _isDragging;
00062     int _playerId;
00063
00064     float _playerAngleXZ;
00065     bool _isPlayerViewDragging;
00066
00067     void handlePlayerCameraMouseInput();
00068 };
00069
00070 #endif /* !CAMERA_MANAGER_HPP_ */

```

7.14 GUI.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** GUI
00006  */
00007
00008 #ifndef GUI_HPP_
00009 #define GUI_HPP_
00010
00011 #include <memory>
00012 #include <string>
00013 #include <utility>
00014 #include "../Game/GameInfos.hpp"
00015 #include "Map.hpp"
00016 #include "HUD/HUD.hpp"
00017 #include "../Audio/IAudio.hpp"
00018 #include "../Utils/Constants.hpp"
00019 #include "Camera/CameraManager.hpp"
00020 #include "../IDisplay.hpp"

```

```

00021 #include "../DLLoader/DLLoader.hpp"
00022
00023 class GUI {
00024     public:
00025         GUI(std::shared_ptr<GameInfos> gameInfos, const std::string &libPath);
00026         ~GUI();
00027
00028         void run();
00029         void refresh();
00030         void handleVictory(const std::string &teamName);
00031
00032         int getWindowWidth() const;
00033         int getWindowHeight() const;
00034         void setWindowWidth(int width);
00035         void setWindowHeight(int height);
00036
00037         void switchCameraMode(zappy::gui::CameraMode mode);
00038         void switchCameraModeNext();
00039         void setPlayerToFollow(int playerId);
00040         int getPlayerToFollow() const;
00041         bool selectFirstAvailablePlayer();
00042         void switchToNextPlayer();
00043         void switchToPreviousPlayer();
00044
00045     private:
00046         void updateCamera();
00047         virtual void update();
00048         virtual void draw();
00049         virtual bool isRunning();
00050         bool playerExists(int playerId) const;
00051
00052         void initModels();
00053         void initPlayers();
00054         void handlePlayerClicks();
00055         int getPlayerUnderMouse() const;
00056         BoundingBox3D getPlayerBoundingBox(const zappy::structs::Player& player) const;
00057
00058         void handleTileClicks();
00059         std::pair<int, int> getTileUnderMouse() const;
00060         BoundingBox3D getTileBoundingBox(int x, int y) const;
00061
00062         std::string _currentLibLoaded;
00063         bool _isRunning;
00064
00065         DLLoader<std::shared_ptr<IDisplay>> _dlLoader;
00066         std::shared_ptr<IDisplay> _display;
00067         std::shared_ptr<GameInfos> _gameInfos;
00068         std::unique_ptr<Map> _map;
00069         std::unique_ptr<HUD> _hud;
00070         std::shared_ptr<IAudio> _audio;
00071         std::shared_ptr<CameraManager> _cameraManager;
00072
00073         int _windowWidth;
00074         int _windowHeight;
00075
00076         zappy::gui::CameraMode _cameraMode;
00077         bool _isHUDVisible = true;
00078         bool _backgroundLoaded;
00079         bool _skyboxLoaded;
00080         int _hoveredPlayerId;
00081         std::pair<int, int> _selectedTile;
00082
00083         bool _performanceMode = false;
00084 };
00085
00086 #endif /* !GUI_HPP_ */

```

7.15 Button.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Button
00006  */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <functional>
00012 #include <memory>
00013
00014 #include "../UIElement/AUIElement.hpp"
00015 #include "../../Audio/IAudio.hpp"
00016 #include "../../IDisplay.hpp"

```

```

00017
00018 class Button : public AUIElement {
00019     public:
00020         Button(
00021             std::shared_ptr<IDisplay> display,
00022             std::shared_ptr<IAudio> audio,
00023             float x, float y,
00024             float width, float height,
00025             const std::string& text,
00026             std::function<void()> callback
00027         );
00028
00029         ~Button() override = default;
00030
00031         void draw() override;
00032
00033         void update() override;
00034
00035         void setText(const std::string& text);
00036
00037         std::string getText() const;
00038
00039         void setCallback(std::function<void()> callback);
00040
00041         void setColors(
00042             Color32 normal,
00043             Color32 hover,
00044             Color32 pressed,
00045             Color32 textColor
00046         );
00047
00048         void setSize(float width, float height) override;
00049
00050     private:
00051         std::string _text;
00052         std::function<void()> _callback;
00053
00054         Color32 _normalColor;
00055         Color32 _hoverColor;
00056         Color32 _pressedColor;
00057         Color32 _textColor;
00058
00059         bool _isHovered;
00060         bool _isPressed;
00061
00062         std::shared_ptr<IDisplay> _display;
00063         std::shared_ptr<IAudio> _audio;
00064 };

```

7.16 Checkbox.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Checkbox
00006  */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <functional>
00012 #include <memory>
00013
00014 #include "../UIElement/AUIElement.hpp"
00015 #include "../../Audio/IAudio.hpp"
00016 #include "../../IDisplay.hpp"
00017
00018 class Checkbox : public AUIElement {
00019     public:
00020         Checkbox(
00021             std::shared_ptr<IDisplay> display,
00022             std::shared_ptr<IAudio> audio,
00023             float x, float y,
00024             float width, float height,
00025             bool initialValue,
00026             std::function<void(bool)> callback
00027         );
00028
00029         ~Checkbox() override = default;
00030
00031         void draw() override;
00032
00033         void update() override;
00034

```



```

00035         void setCallback(std::function<void(bool)> callback);
00036
00037         void setValue(bool value);
00038
00039         bool getValue() const;
00040
00041         void setColors(
00042             Color32 normalColor,
00043             Color32 hoverColor,
00044             Color32 pressedColor,
00045             Color32 checkColor
00046         );
00047
00048         void setSize(float width, float height) override;
00049
00050     private:
00051         bool _value;
00052         std::function<void(bool)> _callback;
00053
00054         Color32 _normalColor;
00055         Color32 _hoverColor;
00056         Color32 _pressedColor;
00057         Color32 _checkColor;
00058
00059         bool _isHovered;
00060         bool _isPressed;
00061
00062         std::shared_ptr<IDisplay> _display;
00063         std::shared_ptr<IAudio> _audio;
00064
00065         float _checkboxSize;
00066 };

```

7.17 AContainers.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** AContainers
00006  */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <vector>
00012 #include <memory>
00013
00014 #include "IContainers.hpp"
00015
00016 struct RelativePosition {
00017     float xPercent;
00018     float yPercent;
00019     float widthPercent;
00020     float heightPercent;
00021 };
00022
00023 class AContainers : public IContainers {
00024     public:
00025         AContainers(std::shared_ptr<IDisplay> display, float x, float y, float width,
00026             float height);
00027
00028         virtual ~AContainers() = default;
00029
00030         void setPosition(float x, float y) override;
00031         void setSize(float width, float height) override;
00032         FloatRect getBounds() const override;
00033         bool contains(float x, float y) const override;
00034         void setVisible(bool visible) override;
00035         bool isVisible() const override;
00036
00037         void setRelativePosition(float xPercent, float yPercent, float widthPercent,
00038             float heightPercent);
00039
00040         RelativePosition getRelativePosition() const;
00041
00042         void updatePositionFromRelative();
00043
00044         float getWidth() const;
00045         float getHeight() const;
00046
00047     protected:
00048         std::shared_ptr<IDisplay> _display;
00049         FloatRect _bounds;
00050         RelativePosition _relativePos;

```

```

00051         Color32 _backgroundColor;
00052         bool _visible;
00053         bool _hasBackground;
00054     };

```

7.18 Containers.hpp

```

00001  /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Containers
00006  */
00007
00008  #pragma once
00009
00010  #include <vector>
00011  #include <functional>
00012  #include <unordered_map>
00013  #include <memory>
00014  #include <string>
00015
00016  #include "AContainers.hpp"
00017  #include "../UIElement/UIElement.hpp"
00018  #include "../Button/Button.hpp"
00019  #include "../Text/Text.hpp"
00020  #include "../Slider/Slider.hpp"
00021  #include "../Image/Image.hpp"
00022  #include "../ImageButton/ImageButton.hpp"
00023  #include "../Checkbox/Checkbox.hpp"
00024  #include "../../Audio/IAudio.hpp"
00025  #include "../../IDisplay.hpp"
00026
00027  class Containers : public AContainers {
00028  public:
00029      Containers(std::shared_ptr<IDisplay> display, std::shared_ptr<IAudio> audio,
00030                float x, float y, float width, float height,
00031                Color32 backgroundColor = {40, 40, 40, 200});
00032
00033      ~Containers() override = default;
00034
00035      void draw() override;
00036
00037      void update() override;
00038
00039      void setBackgroundColor(Color32 color);
00040
00041      bool addElement(const std::string& id, std::shared_ptr<UIElement> element);
00042
00043      std::shared_ptr<UIElement> getElement(const std::string& id) const;
00044
00045      bool removeElement(const std::string& id);
00046
00047      std::shared_ptr<Button> addButton(
00048          const std::string& id,
00049          float x, float y,
00050          float width, float height,
00051          const std::string& text,
00052          std::function<void()> callback
00053      );
00054
00055      std::shared_ptr<Button> addButton(
00056          const std::string& id,
00057          float x, float y,
00058          float width, float height,
00059          const std::string& text,
00060          std::function<void()> callback,
00061          Color32 normalColor,
00062          Color32 hoverColor,
00063          Color32 pressedColor,
00064          Color32 textColor
00065      );
00066
00067      std::shared_ptr<Text> addText(
00068          const std::string& id,
00069          float x, float y,
00070          const std::string& text,
00071          float fontSize = 20.0f,
00072          Color32 color = CBLACK
00073      );
00074
00075      std::shared_ptr<Slider> addSlider(
00076          const std::string& id,
00077          float x, float y,
00078          float width, float height,

```

```

00079         float minValue, float maxValue,
00080         float initialValue,
00081         const std::string& text,
00082         std::function<void(float)> onValueChanged
00083     );
00084
00085     std::shared_ptr<Slider> addSliderPercent (
00086         const std::string& id,
00087         float xPercent, float yPercent,
00088         float widthPercent, float heightPercent,
00089         float minValue, float maxValue,
00090         float initialValue,
00091         const std::string& text,
00092         std::function<void(float)> onValueChanged
00093     );
00094
00095     void clearElements();
00096
00097     void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00098
00099     std::shared_ptr<Button> addButtonPercent (
00100         const std::string& id,
00101         float xPercent, float yPercent,
00102         float widthPercent, float heightPercent,
00103         const std::string& text,
00104         std::function<void()> callback
00105     );
00106
00107     std::shared_ptr<Button> addButtonPercent (
00108         const std::string& id,
00109         float xPercent, float yPercent,
00110         float widthPercent, float heightPercent,
00111         const std::string& text,
00112         std::function<void()> callback,
00113         Color32 normalColor,
00114         Color32 hoverColor,
00115         Color32 pressedColor,
00116         Color32 textColor
00117     );
00118
00119     std::shared_ptr<Text> addTextPercent (
00120         const std::string& id,
00121         float xPercent, float yPercent,
00122         const std::string& text,
00123         float fontSizePercent = 5.0f,
00124         Color32 color = CBLACK
00125     );
00126
00127     std::shared_ptr<Image> addImage (
00128         const std::string& id,
00129         float x, float y,
00130         float width, float height,
00131         const std::string& imagePath
00132     );
00133
00134     std::shared_ptr<Image> addImage (
00135         const std::string& id,
00136         float x, float y,
00137         float width, float height,
00138         const std::string& imagePath,
00139         Color32 tint
00140     );
00141
00142     std::shared_ptr<Image> addImagePercent (
00143         const std::string& id,
00144         float xPercent, float yPercent,
00145         float widthPercent, float heightPercent,
00146         const std::string& imagePath
00147     );
00148
00149     std::shared_ptr<Image> addImagePercent (
00150         const std::string& id,
00151         float xPercent, float yPercent,
00152         float widthPercent, float heightPercent,
00153         const std::string& imagePath,
00154         Color32 tint
00155     );
00156
00157     std::shared_ptr<ImageButton> addImageButton (
00158         const std::string& id,
00159         float x, float y,
00160         float width, float height,
00161         const std::string& imagePath,
00162         std::function<void()> callback
00163     );
00164
00165     std::shared_ptr<ImageButton> addImageButton (
00166         const std::string& id,

```

```

00166         float x, float y,
00167         float width, float height,
00168         const std::string& imagePath,
00169         std::function<void()> callback,
00170         Color32 tint
00171     );
00172
00173     std::shared_ptr<ImageButton> addImageButtonPercent (
00174         const std::string& id,
00175         float xPercent, float yPercent,
00176         float widthPercent, float heightPercent,
00177         const std::string& imagePath,
00178         std::function<void()> callback
00179     );
00180
00181     std::shared_ptr<ImageButton> addImageButtonPercent (
00182         const std::string& id,
00183         float xPercent, float yPercent,
00184         float widthPercent, float heightPercent,
00185         const std::string& imagePath,
00186         std::function<void()> callback,
00187         Color32 tint
00188     );
00189
00190     std::shared_ptr<Checkbox> addCheckbox (
00191         const std::string& id,
00192         float x, float y,
00193         float width, float height,
00194         bool initialValue,
00195         std::function<void(bool)> callback
00196     );
00197
00198     std::shared_ptr<Checkbox> addCheckboxPercent (
00199         const std::string& id,
00200         float xPercent, float yPercent,
00201         float widthPercent, float heightPercent,
00202         bool initialValue,
00203         std::function<void(bool)> callback
00204     );
00205
00206         float getWidth() const;
00207         float getHeight() const;
00208
00209     private:
00210         std::shared_ptr<IAudio> _audio;
00211         std::unordered_map<std::string, std::shared_ptr<IUElement>> _elements;
00212 };

```

7.19 IContainers.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** IContainers
00006  */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <memory>
00012 #include <vector>
00013 #include ".././../IDisplay.hpp"
00014
00015 class IContainers {
00016     public:
00017         virtual ~IContainers() = default;
00018
00019         virtual void draw() = 0;
00020
00021         virtual void update() = 0;
00022
00023         virtual void setPosition(float x, float y) = 0;
00024
00025         virtual void setSize(float width, float height) = 0;
00026
00027         virtual FloatRect getBounds() const = 0;
00028
00029         virtual bool contains(float x, float y) const = 0;
00030
00031         virtual void setVisible(bool visible) = 0;
00032
00033         virtual bool isVisible() const = 0;
00034 };

```

7.20 Help.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Help
00006 */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include <string>
00012 #include "../Containers/Containers.hpp"
00013 #include "../../IDisplay.hpp"
00014 #include "../../Audio/IAudio.hpp"
00015
00016 class Help {
00017     public:
00018         Help(std::shared_ptr<IDisplay> display, std::shared_ptr<IAudio> audio);
00019
00020         ~Help() = default;
00021
00022         void show();
00023
00024         void hide();
00025
00026         bool isVisible() const;
00027
00028         bool containsPoint(float x, float y) const;
00029
00030         void update();
00031
00032         void draw();
00033
00034         void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00035
00036     private:
00037         void initHelpContainer();
00038
00039         std::shared_ptr<IDisplay> _display;
00040         std::shared_ptr<IAudio> _audio;
00041         std::shared_ptr<Containers> _helpContainer;
00042         bool _visible;
00043 };

```

7.21 HUD.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** HUD
00006 */
00007
00008 #pragma once
00009
00010 #include <vector>
00011 #include <unordered_map>
00012 #include <memory>
00013 #include <string>
00014 #include <utility>
00015 #include <functional>
00016 #include <chrono>
00017 #include "Containers/Containers.hpp"
00018 #include "../../Game/GameInfos.hpp"
00019 #include "../../Audio/IAudio.hpp"
00020 #include "../../Utils/Constants.hpp"
00021 #include "Help/Help.hpp"
00022 #include "Settings/Settings.hpp"
00023 #include "../../IDisplay.hpp"
00024 #include "../../Observer/IObserver.hpp"
00025 #include "Graphic/Camera/CameraManager.hpp"
00026
00027 class HUD : public IObserver {
00028     public:
00029         HUD(std::shared_ptr<IDisplay> display, std::shared_ptr<GameInfos> gameInfos,
00030             std::shared_ptr<IAudio> audio,
00031             std::shared_ptr<CameraManager>,
00032             std::function<void()> resetCameraFunc = nullptr);
00033
00034         ~HUD();
00035
00036         void draw();

```

```

00037
00038     std::shared_ptr<Containers> addContainer(
00039         const std::string& id,
00040         float x, float y,
00041         float width, float height,
00042         Color32 backgroundColor = {40, 40, 40, 200}
00043     );
00044
00045     std::shared_ptr<Containers> getContainer(const std::string& id) const;
00046
00047     bool removeContainer(const std::string& id);
00048
00049     void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00050
00051     void clearAllContainers();
00052
00053     void initDefaultLayout(float sideWidthPercent = 15.0f,
00054         float bottomHeightPercent = 20.0f);
00055
00056     std::shared_ptr<Containers> getSideContainer() const;
00057
00058     std::shared_ptr<Containers> getBottomContainer() const;
00059
00060     std::shared_ptr<Containers> getSquareContainer() const;
00061
00062     std::shared_ptr<Containers> getTpsContainer() const;
00063
00064     std::shared_ptr<Containers> getSecurityContainer() const;
00065
00066     std::shared_ptr<Containers> getServerMessagesContainer() const;
00067
00068     void initExitButton();
00069
00070     void initSettingsButton();
00071
00072     void initHelpButton();
00073
00074     void initCameraResetButton();
00075
00076     void initTeamPlayersDisplay(std::shared_ptr<GameInfos> gameInfos);
00077
00078     void updateTeamPlayersDisplay(std::shared_ptr<GameInfos> gameInfos);
00079
00080     void initTpsSlider(std::shared_ptr<GameInfos> gameInfos,
00081         std::shared_ptr<IDisplay> raylib, std::shared_ptr<IAudio> audio);
00082
00083     void updateTpsSlider(std::shared_ptr<GameInfos> gameInfos);
00084
00085     void initServerMessagesDisplay(std::shared_ptr<GameInfos> gameInfos);
00086
00087     void updateServerMessagesDisplay(std::shared_ptr<GameInfos> gameInfos);
00088
00089     void initPlayerInventoryDisplay(int playerId);
00090
00091     void updatePlayerInventoryDisplay(int playerId, zappy::gui::CameraMode cameraMode);
00092
00093     void updateHelpInformationHUD(zappy::gui::CameraMode cameraMode);
00094
00095     void clearPlayerInventoryElements();
00096
00097     void setSelectedTile(int x, int y);
00098
00099     void initTileResourceDisplay();
00100
00101     void updateTileResourceDisplay(int x, int y);
00102
00103     void clearTileResourceElements();
00104
00105     void initFpsDisplay();
00106
00107     void updateFpsDisplay();
00108
00109     zappy::structs::Player getPlayerById(int playerId) const;
00110
00111     bool isPlayerInIncantation(int playerId) const;
00112
00113     void setResetCameraCallback(std::function<void()> resetFunc);
00114
00115     void displayWinMessage(const std::string& teamName);
00116
00117     void update() override;
00118     void onGameEvent(GameEventType eventType, const std::string& teamName) override;
00119
00120     bool isMouseOverHUD() const;
00121
00122 private:
00123     void _initHelpInformation();

```

```

00124
00125     std::string _camModeToText(zappy::gui::CameraMode, bool isGamePadAvailable);
00126
00127     std::string _camKeyHelp(zappy::gui::CameraMode, bool isGamePadAvailable);
00128
00129     std::shared_ptr<Containers> createSquareContainer(float squareSize,
00130         float sideWidthPercent);
00131
00132     std::shared_ptr<Containers> createSideContainer(
00133         float sideYStart,
00134         float sideWidth,
00135         float sideHeight,
00136         float sideWidthPercent,
00137         float bottomHeightPercent);
00138
00139     std::shared_ptr<Containers> createBottomContainer(
00140         int screenWidth,
00141         int screenHeight,
00142         float bottomHeight,
00143         float bottomHeightPercent);
00144
00145     std::shared_ptr<Containers> createTpsContainer(
00146         int screenWidth,
00147         int screenHeight,
00148         float bottomHeight,
00149         float bottomHeightPercent);
00150
00151     std::shared_ptr<Containers> createSecurityContainer(
00152         int screenWidth,
00153         int screenHeight,
00154         float bottomHeight,
00155         float bottomHeightPercent);
00156
00157     std::shared_ptr<Containers> createServerMessagesContainer(
00158         int screenWidth,
00159         int screenHeight,
00160         float bottomHeight,
00161         float bottomHeightPercent);
00162
00163     void updateElementPositions(
00164         std::shared_ptr<Containers> container,
00165         const std::unordered_map<std::string, float>& initialYPositions,
00166         float offset);
00167
00168     std::pair<float, float> calculateContentMetrics(
00169         std::shared_ptr<Containers> container,
00170         const std::unordered_map<std::string, float>& initialYPositions);
00171
00172     void clearTeamDisplayElements(std::shared_ptr<Containers> container);
00173
00174     std::vector<int> getTeamPlayerNumbers(const std::string& teamName,
00175         const std::vector<zappy::structs::Player>& players);
00176
00177     std::string createPlayerListText(const std::vector<int>& playerNumbers);
00178
00179     void addPlayerListText(std::shared_ptr<Containers> container,
00180         const std::string& teamId,
00181         float yPos, const std::vector<int>& playerNumbers);
00182
00183     void addIncrementDecrementButtons(std::shared_ptr<Containers> container, int playerId);
00184
00185     std::unordered_map<std::string, std::shared_ptr<Containers>> _containers;
00186     std::shared_ptr<IDisplay> _display;
00187     std::shared_ptr<GameInfos> _gameInfos;
00188     std::shared_ptr<IAudio> _audio;
00189     std::shared_ptr<CameraManager> _camera;
00190     std::shared_ptr<Help> _help;
00191     std::shared_ptr<Settings> _settings;
00192     std::function<void()> _resetCameraFunc;
00193     bool _showVictoryMessage;
00194     std::string _winningTeam;
00195     Color32 _victoryColor;
00196     std::pair<int, int> _selectedTile;
00197 };

```

7.22 Image.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Image
00006  */
00007
00008 #pragma once

```

```

00009
00010 #include <string>
00011 #include <memory>
00012
00013 #include "../UIElement/AUIElement.hpp"
00014 #include "../../IDisplay.hpp"
00015
00016 class Image : public AUIElement {
00017     public:
00018         Image(
00019             std::shared_ptr<IDisplay> display,
00020             float x, float y,
00021             float width, float height,
00022             const std::string& imagePath
00023         );
00024
00025         ~Image() override = default;
00026
00027         void draw() override;
00028
00029         void update() override;
00030
00031         void setImagePath(const std::string& imagePath);
00032
00033         std::string getImagePath() const;
00034
00035         void setTint(Color32 tint);
00036
00037         Color32 getTint() const;
00038
00039         void setSize(float width, float height) override;
00040
00041         void setMaintainAspectRatio(bool maintain);
00042
00043         bool getMaintainAspectRatio() const;
00044
00045     private:
00046         std::string _imagePath;
00047         Color32 _tint;
00048         bool _maintainAspectRatio;
00049         bool _imageLoaded;
00050
00051         void loadImage();
00052 };

```

7.23 ImageButton.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** ImageButton
00006 */
00007
00008 #pragma once
00009
00010 #include <functional>
00011 #include <string>
00012 #include <memory>
00013 #include "../Image/Image.hpp"
00014 #include "../../Audio/IAudio.hpp"
00015
00016 class ImageButton : public Image {
00017     public:
00018         ImageButton(
00019             std::shared_ptr<IDisplay> display,
00020             std::shared_ptr<IAudio> audio,
00021             float x, float y,
00022             float width, float height,
00023             const std::string& imagePath,
00024             std::function<void()> callback
00025         );
00026
00027         ~ImageButton() override = default;
00028
00029         void update() override;
00030
00031         void setCallback(std::function<void()> callback);
00032
00033         std::function<void()> getCallback() const;
00034
00035     private:
00036         std::function<void()> _callback;
00037         std::shared_ptr<IAudio> _audio;
00038         bool _isHovered;

```



```
00039         bool _isPressed;
00040     };
```

7.24 Settings.hpp

```
00001  /*
00002  ** EPITECH PROJECT, 2025
00003  ** zappy
00004  ** File description:
00005  ** Settings
00006  */
00007
00008  #ifndef SETTINGS_HPP_
00009  #define SETTINGS_HPP_
00010  #include <memory>
00011  #include "../Containers/Containers.hpp"
00012  #include "../../../IDisplay.hpp"
00013  #include "../../../Audio/IAudio.hpp"
00014  #include "Graphic/Camera/CameraManager.hpp"
00015
00016  class Settings {
00017  private:
00018      std::shared_ptr<IDisplay> _display;
00019      std::shared_ptr<IAudio> _audio;
00020      std::shared_ptr<CameraManager> _camera;
00021      float _sfxLevel;
00022      float _musicLevel;
00023      float _cameraMovingSpeed;
00024      float _cameraRotaSpeed;
00025      float _cameraZoomSpeed;
00026      std::shared_ptr<Containers> _settingsContainer;
00027      bool _visible;
00028
00029  public:
00030      bool isVisible() const;
00031
00032      bool containsPoint(float x, float y) const;
00033
00034      void show();
00035
00036      void hide();
00037
00038      void update();
00039
00040      void draw();
00041
00042      void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00043
00044      Settings(
00045          std::shared_ptr<IDisplay> display,
00046          std::shared_ptr<IAudio> audio,
00047          std::shared_ptr<CameraManager> camera
00048      );
00049      ~Settings();
00050  };
00051
00052  #endif /* !SETTINGS_HPP_ */
```

7.25 Slider.hpp

```
00001  /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Slider
00006  */
00007
00008  #ifndef SLIDER_HPP_
00009  #define SLIDER_HPP_
00010
00011  #include <string>
00012  #include <functional>
00013  #include <memory>
00014
00015  #include "../../../IDisplay.hpp"
00016  #include "../UIElement/AUIElement.hpp"
00017
00018  class Slider : public AUIElement {
00019  public:
00020      Slider(
00021          std::shared_ptr<IDisplay> raylib,
00022          float x, float y,
```

```

00023         float width, float height,
00024         float minValue, float maxValue,
00025         float initialValue,
00026         const std::string& text,
00027         std::function<void(float)> onValueChanged
00028     );
00029
00030     ~Slider() override = default;
00031
00032     void draw() override;
00033     void update() override;
00034     bool isDragging() const;
00035
00036     void setValue(float value);
00037     float getValue() const;
00038     void setMinValue(float minValue);
00039     void setMaxValue(float maxValue);
00040     float getMinValue() const;
00041     float getMaxValue() const;
00042     void setText(const std::string& text);
00043     std::string getText() const;
00044
00045     void setSize(float width, float height) override;
00046
00047     private:
00048         float _value;
00049         float _minValue;
00050         float _maxValue;
00051         std::string _text;
00052         std::function<void(float)> _onValueChanged;
00053
00054         bool _isDragging;
00055         float _sliderTrackWidth;
00056         float _sliderHandleRadius;
00057
00058         Color32 _trackColor;
00059         Color32 _fillColor;
00060         Color32 _handleColor;
00061         Color32 _textColor;
00062
00063         void updateValueFromMousePosition(float mouseX);
00064         float getHandlePosition() const;
00065         bool isMouseOverHandle(float mouseX, float mouseY) const;
00066         bool isMouseOverTrack(float mouseX, float mouseY) const;
00067 };
00068
00069 #endif /* !SLIDER_HPP_ */

```

7.26 Text.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Text
00006  */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include <string>
00012
00013 #include "../UIElement/AUIElement.hpp"
00014 #include "../IDisplay.hpp"
00015
00016 class Text : public AUIElement {
00017     public:
00018         Text(
00019             std::shared_ptr<IDisplay> raylib,
00020             float x, float y,
00021             const std::string& text,
00022             float fontSize = 20.0f,
00023             Color32 color = CBLACK
00024         );
00025
00026         ~Text() override = default;
00027
00028         void draw() override;
00029
00030         void update() override;
00031
00032         void setText(const std::string& text);
00033
00034         std::string getText() const;
00035

```

```

00036         void setFontSize(float fontSize);
00037
00038         float getFontSize() const;
00039
00040         void setColor(Color32 color);
00041
00042         Color32 getColor() const;
00043
00044         void setSize(float width, float height) override;
00045
00046         float getWidth() const;
00047         void setX(float x);
00048         void setY(float y);
00049
00050     private:
00051         std::string _text;
00052         float _fontSize;
00053         Color32 _color;
00054         std::shared_ptr<IDisplay> _display;
00055 };

```

7.27 AUIElement.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** AUIElement
00006  */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include "IUIElement.hpp"
00012
00013 struct UIRelativePosition {
00014     float xPercent;
00015     float yPercent;
00016     float widthPercent;
00017     float heightPercent;
00018 };
00019
00020 class AUIElement : public IUIElement {
00021     public:
00022         AUIElement(std::shared_ptr<IDisplay> display, float x, float y, float width,
00023             float height);
00024
00025         virtual ~AUIElement() = default;
00026
00027         // IUIElement implementation
00028         void setPosition(float x, float y) override;
00029         FloatRect getBounds() const override;
00030         bool contains(float x, float y) const override;
00031         void setVisible(bool visible) override;
00032         bool isVisible() const override;
00033
00034         virtual void setSize(float width, float height);
00035
00036         void setRelativePosition(float xPercent, float yPercent, float widthPercent,
00037             float heightPercent);
00038
00039         UIRelativePosition getRelativePosition() const;
00040
00041     protected:
00042         std::shared_ptr<IDisplay> _display;
00043         FloatRect _bounds;
00044         UIRelativePosition _relativePos;
00045         bool _visible;
00046 };

```

7.28 IUIElement.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** IUIElement
00006  */
00007
00008 #pragma once
00009
00010 #include ".././././IDisplay.hpp"

```

```

00011
00012 class UIElement {
00013     public:
00014         virtual ~UIElement() = default;
00015
00016         virtual void draw() = 0;
00017
00018         virtual void update() = 0;
00019
00020         virtual void setPosition(float x, float y) = 0;
00021
00022         virtual void setSize(float width, float height) = 0;
00023
00024         virtual FloatRect getBounds() const = 0;
00025
00026         virtual bool contains(float x, float y) const = 0;
00027
00028         virtual void setVisible(bool visible) = 0;
00029
00030         virtual bool isVisible() const = 0;
00031 };

```

7.29 Map.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** Map
00006  */
00007
00008 #ifndef MAP_HPP_
00009 #define MAP_HPP_
00010
00011 #include <memory>
00012 #include <unordered_map>
00013 #include <vector>
00014 #include <string>
00015 #include <chrono>
00016 #include "../Game/GameInfos.hpp"
00017 #include "../IDisplay.hpp"
00018
00019 enum class DisplayPriority {
00020     TILE = 0,
00021     EGG = 1,
00022     PLAYER = 2,
00023     FOOD = 3,
00024     ROCK = 4,
00025 };
00026
00027 struct PlayerRotationState {
00028     float currentRotation;
00029     float targetRotation;
00030     bool isRotating;
00031     std::chrono::steady_clock::time_point lastUpdateTime;
00032
00033     PlayerRotationState() : currentRotation(0.0f), targetRotation(0.0f),
00034         isRotating(false), lastUpdateTime(std::chrono::steady_clock::now()) {}
00035 };
00036
00037 struct PlayerPositionState {
00038     Vector3f currentPosition;
00039     Vector3f targetPosition;
00040     bool isMoving;
00041     std::chrono::steady_clock::time_point lastUpdateTime;
00042
00043     PlayerPositionState() : currentPosition({0.0f, 0.0f, 0.0f}),
00044         targetPosition({0.0f, 0.0f, 0.0f}),
00045         isMoving(false), lastUpdateTime(std::chrono::steady_clock::now()) {}
00046 };
00047
00048 class Map {
00049     public:
00050         Map(std::shared_ptr<GameInfos> gameInfos, std::shared_ptr<IDisplay> display);
00051         ~Map();
00052
00053         void draw(bool performanceMode = false);
00054         void drawBroadcastingPlayers();
00055         void drawIncantations();
00056
00057         void drawTile(int x, int y, const zappy::structs::Tile &tile);
00058         void drawPerformanceTile(const zappy::structs::Tile &tile);
00059
00060         void drawRock(int x, int y, const zappy::structs::Tile &tile);
00061         void drawPerformanceRock(int x, int y, const zappy::structs::Tile &tile);

```

```

00062
00063     void drawFood(int x, int y, const zappy::structs::Tile &tile);
00064     void drawPerformanceFood(int x, int y, const zappy::structs::Tile &tile);
00065
00066     void drawAllPlayers();
00067     void drawEggs(int x, int y);
00068     Color32 getTeamColor(const std::string &teamName);
00069
00070     float getOffset(DisplayPriority priority, int x, int y, size_t stackIndex = 0);
00071     void updatePlayerRotations();
00072     float getPlayerInterpolatedRotation(int playerId, int serverOrientation);
00073     void updatePlayerPositions();
00074     Vector3f getPlayerInterpolatedPosition(int playerId, int serverX, int serverY);
00075
00076 private:
00077     std::shared_ptr<GameInfos> _gameInfos;
00078     std::shared_ptr<IDisplay> _display;
00079     std::unordered_map<std::string, Color32> _teamColors;
00080     std::vector<Color32> _colors;
00081     int _colorIndex = 0;
00082
00083     std::unordered_map<int, std::chrono::steady_clock::time_point> _broadcastStartTimes;
00084     std::unordered_map<int, PlayerRotationState> _playerRotations;
00085     std::unordered_map<int, PlayerPositionState> _playerPositions;
00086
00087     static constexpr float BASE_HEIGHT_TILE = 0.0f;
00088
00089     static constexpr float BASE_HEIGHT_PLAYER = 0.0f;
00090     static constexpr float PLAYER_HEIGHT = 0.95f;
00091
00092     static constexpr float BASE_HEIGHT_EGG = 0.0f;
00093     static constexpr float EGG_HEIGHT = 0.2f;
00094
00095     static constexpr float BASE_HEIGHT_FOOD = 0.1f;
00096     static constexpr float FOOD_HEIGHT = 0.7f;
00097
00098     static constexpr float BASE_HEIGHT_ROCK = 0.1f;
00099     static constexpr float ROCK_HEIGHT = 0.7f;
00100
00101
00102     void drawTorus(const Vector3f &position, float radius, float thickness,
00103                   int radialSegments, Color32 color);
00104     float orientationToRotation(int orientation);
00105     float normalizeAngle(float angle);
00106     float getShortestAngleDifference(float from, float to);
00107     Vector3f calculatePlayerWorldPosition(int x, int y);
00108     float getDistance(const Vector3f& from, const Vector3f& to);
00109     Vector3f lerpVector3f(const Vector3f& from, const Vector3f& to, float t);
00110
00111     bool _performanceMode = false;
00112 };
00113
00114 #endif /* !MAP_HPP_ */

```

7.30 IDisplay.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** IDisplay
00006 */
00007
00008 #ifndef IDISPLAY_HPP_
00009 #define IDISPLAY_HPP_
00010 #include <utility>
00011 #include <string>
00012 #include "Utils/InputType.hpp"
00013
00014 enum Key {
00015     TAB,
00016     ESC,
00017     UP,
00018     DOWN,
00019     RIGHT,
00020     LEFT,
00021     H,
00022     C,
00023     GM_PD_LEFT_SHOULDER,
00024     GM_PD_RIGHT_SHOULDER,
00025     GM_PD_LEFT_TRIGGER,
00026     GM_PD_RIGHT_TRIGGER,
00027     GM_PD_UP,
00028     GM_PD_DOWN,
00029     GM_PD_AXIS_RIGHT_X,

```

```

00030     GM_PD_AXIS_RIGHT_Y,
00031     GM_PD_H,
00032     MOUSE_LEFT,
00033     MOUSE_RIGHT,
00034 };
00035
00036 typedef struct Vector3f {
00037     float x;
00038     float y;
00039     float z;
00040
00041     bool operator==(const Vector3f& other) const {
00042         return x == other.x && y == other.y && z == other.z;
00043     }
00044
00045     bool operator!=(const Vector3f& other) const {
00046         return !(*this == other);
00047     }
00048 } Vector3f;
00049
00050 typedef struct Vector2f {
00051     float x;
00052     float y;
00053 } Vector2f;
00054
00055 typedef struct Vector2i {
00056     int x;
00057     int y;
00058 } Vector2i;
00059
00060 typedef struct Color32 {
00061     unsigned char r;
00062     unsigned char g;
00063     unsigned char b;
00064     unsigned char a;
00065 } Color32;
00066
00067 typedef struct FloatRect {
00068     float x;
00069     float y;
00070     float width;
00071     float height;
00072 } FloatRect;
00073
00074 typedef struct IntRect {
00075     int x;
00076     int y;
00077     int width;
00078     int height;
00079 } IntRect;
00080
00081 typedef struct Ray3D {
00082     Vector3f position;
00083     Vector3f direction;
00084 } Ray3D;
00085
00086 typedef struct RayCollision3D {
00087     bool hit;
00088     float distance;
00089     Vector3f point;
00090     Vector3f normal;
00091 } RayCollision3D;
00092
00093 typedef struct BoundingBox3D {
00094     Vector3f min;
00095     Vector3f max;
00096 } BoundingBox3D;
00097
00098 #define COLOR(r, g, b) Color32{ r, g, b, 255 }
00099 #define CLIGHTGRAY COLOR(200, 200, 200)
00100 #define CBLACK COLOR(0, 0, 0)
00101 #define CRED COLOR(230, 41, 55)
00102 #define CBROWN COLOR(127, 106, 79)
00103 #define CBLUE COLOR(0, 121, 241)
00104 #define CWHITE COLOR(255, 255, 255)
00105
00106 #define CRAYWHITE COLOR(245, 245, 245)
00107 #define CPINK COLOR(255, 109, 194)
00108 #define CGREEN COLOR(0, 228, 48)
00109 #define CMAROON COLOR(190, 33, 55)
00110 #define CPURPLE COLOR(200, 122, 255)
00111 #define CORANGE COLOR(255, 161, 0)
00112 #define CYELLOW COLOR(253, 249, 0)
00113
00114 class IDisplay {
00115 public:
00116     virtual Vector2i getMonitorSize() = 0;

```

```

00117     virtual Vector2i getScreenSize() = 0;
00118
00119     virtual void initWindow(int width, int height, std::string) = 0;
00120     virtual void initCamera() = 0;
00121
00122     virtual bool isWindowReady() = 0;
00123     virtual void setTargetFPS(unsigned int FPS) = 0;
00124
00125     virtual bool isOpen() = 0;
00126     virtual void closeWindow() = 0;
00127
00128     virtual int getKeyId(enum Key) = 0;
00129
00130     virtual bool isKeyReleased(int key) = 0;
00131     virtual bool isKeyPressed(int key) = 0;
00132     virtual bool isKeyDown(int key) = 0;
00133
00134     virtual bool isGamepadAvailable() = 0;
00135
00136     virtual bool isGamepadButtonReleased(int key) = 0;
00137     virtual bool isGamepadButtonPressed(int key) = 0;
00138     virtual bool isGamepadButtonDown(int key) = 0;
00139
00140     virtual bool isMouseButtonDown(int key) = 0;
00141     virtual bool isMouseButtonReleased(int key) = 0;
00142     virtual bool isMouseButtonPressed(int key) = 0;
00143
00144     virtual Vector2f getMousePosition() = 0;
00145     virtual void setMousePosition(Vector2f) = 0;
00146
00147     virtual float getMouseWheelMove() = 0;
00148
00149     virtual float getGamepadAxisMovement(int key) = 0;
00150
00151     virtual void setCameraPosition(Vector3f) = 0;
00152
00153     virtual void setCameraTarget(Vector3f) = 0;
00154
00155     virtual Vector2f getMouseDelta() = 0;
00156
00157     virtual float vector3DDistanceFromCamera(Vector3f target) = 0;
00158     virtual Vector3f vector3SubtractFromCamera(Vector3f target) = 0;
00159
00160     virtual Vector3f vector3Normalize(Vector3f) = 0;
00161
00162
00163     virtual void enableCursor() = 0;
00164     virtual void disableCursor() = 0;
00165
00166     virtual float getFrameTime() = 0;
00167     virtual int getFPS() = 0;
00168
00169     virtual void updateCameraFreeMode(float camMovingSpeed, float camRotaSpeed) = 0;
00170
00171     virtual InputType getLastInputType() const = 0;
00172     virtual void updateLastInputType() = 0;
00173
00174     virtual float measureText(const std::string& text, float fontSize) const = 0;
00175
00176     virtual bool checkCollisionPointRec(Vector2f point, FloatRect rec) = 0;
00177
00178     virtual Ray3D getMouseRay(Vector2f mousePosition) = 0;
00179     virtual RayCollision3D getRayCollisionBox(Ray3D ray, BoundingBox3D box) = 0;
00180     virtual RayCollision3D getRayCollisionSphere(Ray3D ray, Vector3f center,
00181         float radius) = 0;
00182     virtual bool checkCollisionBoxes(BoundingBox3D box1, BoundingBox3D box2) = 0;
00183
00184     virtual Ray3D getMouseRayFromCurrent() = 0;
00185     virtual BoundingBox3D createBoundingBox(Vector3f center, Vector3f size) = 0;
00186     virtual BoundingBox3D createBoundingBoxFromMinMax(Vector3f min, Vector3f max) = 0;
00187
00188     virtual void beginDrawing() = 0;
00189     virtual void endDrawing() = 0;
00190     virtual void clearBackground(Color32) = 0;
00191
00192     virtual void begin3DMode() = 0;
00193     virtual void end3DMode() = 0;
00194
00195     virtual void endScissorMode() = 0;
00196     virtual void beginScissorMode(IntRect) = 0;
00197
00198     virtual bool loadModel(const std::string& id, const std::string& filepath,
00199         Vector3f center = {0.0f, 0.0f, 0.0f}) = 0;
00200
00201     virtual void drawCube(Vector3f position, float width, float height, float length,
00202         Color32 color) = 0;
00203     virtual void drawCubeWires(Vector3f position, float width, float height, float length,

```

```

00204         Color32 color) = 0;
00205
00206     virtual void drawSphere(Vector3f position, float radius, Color32 color) = 0;
00207     virtual void drawSphereWires(Vector3f position, float radius, int rings, int slices,
00208         Color32 color) = 0;
00209
00210     virtual void drawCylinder(Vector3f position, float radiusTop, float radiusBottom,
00211         float height, int slices, Color32 color) = 0;
00212     virtual void drawCylinderWires(Vector3f position, float radiusTop, float radiusBottom,
00213         float height, int slices, Color32 color) = 0;
00214     virtual void drawCylinderEx(Vector3f startPos, Vector3f endPos, float startRadius,
00215         float endRadius, int sides, Color32 color) = 0;
00216
00217     virtual void drawPlane(Vector3f position, Vector2f size, Color32 color) = 0;
00218
00219     virtual void drawLine3D(Vector3f startPos, Vector3f endPos, Color32 color) = 0;
00220
00221     virtual void drawModelEx(const std::string& id, Vector3f position,
00222         Vector3f rotationAxis, float rotationAngle, Vector3f scale,
00223         Color32 tint = CWHITE) = 0;
00224
00225     virtual void drawCircle(float centerX, float centerY, float radius,
00226         Color32 color) = 0;
00227     virtual void drawCircleLines(float centerX, float centerY, float radius,
00228         Color32 color) = 0;
00229
00230     virtual void drawText(const std::string& text, float x, float y, float fontSize,
00231         Color32 color) = 0;
00232
00233     virtual void drawTextEx(const std::string& text, float x, float y, float fontSize,
00234         float spacing, Color32 color) = 0;
00235
00236     virtual void drawRectangleRec(FloatRect rec, Color32 color) = 0;
00237
00238     virtual bool loadTexture(const std::string& id, const std::string& filepath) = 0;
00239
00240     virtual bool loadFont(const std::string& id, const std::string& filepath) = 0;
00241
00242     virtual void drawTexture(const std::string& id, float x, float y,
00243         Color32 tint = CWHITE) = 0;
00244
00245     virtual void drawTextureScaled(const std::string& id, float x, float y, float width,
00246         float height, Color32 tint = CWHITE) = 0;
00247
00248     virtual Vector2f getTextureSize(const std::string& id) const = 0;
00249
00250     virtual bool loadSkybox(const std::string& id, const std::string& filepath) = 0;
00251
00252     virtual void drawSkybox(const std::string& id) = 0;
00253
00254     virtual float getTime() const = 0;
00255
00256     ~IDisplay() = default;
00257 };
00258
00259 #endif /* !IDISPLAY_HPP_ */

```

7.31 GuiObserver.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004  ** File description:
00005  ** GuiObserver
00006  */
00007
00008 #ifndef GUIOBSERVER_HPP_
00009 #define GUIOBSERVER_HPP_
00010
00011 #include <memory>
00012 #include <string>
00013 #include "IObserver.hpp"
00014
00015 class GUI;
00016
00017 class GuiObserver : public IObserver {
00018     public:
00019         GuiObserver(std::shared_ptr<GUI> gui);
00020         virtual ~GuiObserver() = default;
00021
00022         void update() override;
00023         void onGameEvent(GameEventType eventType, const std::string& teamName) override;
00024
00025     private:
00026         std::weak_ptr<GUI> _gui;

```



```

00027 };
00028
00029 #endif /* !GUIOBSERVER_HPP_ */

```

7.32 IObserver.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** IObserver
00006 */
00007
00008 #ifndef IOBSERVER_HPP_
00009 #define IOBSERVER_HPP_
00010
00011 #include <string>
00012
00013 enum class GameEventType {
00014     STATE_CHANGED,
00015     TEAM_WIN,
00016     TEAM_DEFEAT
00017 };
00018
00019 class IObserver {
00020     public:
00021         virtual ~IObserver() = default;
00022         virtual void update() = 0;
00023         virtual void onGameEvent(GameEventType eventType, const std::string& teamName) {
00024             (void)eventType;
00025             (void)teamName;
00026         }
00027 };
00028
00029 #endif /* !IOBSERVER_HPP_ */

```

7.33 ISubject.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** ISubject
00006 */
00007
00008 #ifndef ISUBJECT_HPP_
00009 #define ISUBJECT_HPP_
00010
00011 #include <vector>
00012 #include <memory>
00013 #include <string>
00014 #include "IObserver.hpp"
00015
00016 class ISubject {
00017     public:
00018         virtual ~ISubject() = default;
00019         virtual void addObserver(std::shared_ptr<IObserver> observer) = 0;
00020         virtual void removeObserver(std::shared_ptr<IObserver> observer) = 0;
00021         virtual void notifyObservers() = 0;
00022         virtual void notifyGameEvent(GameEventType eventType, const std::string& teamName) = 0;
00023
00024     protected:
00025         std::vector<std::weak_ptr<IObserver>> _observers;
00026 };
00027
00028 #endif /* !ISUBJECT_HPP_ */

```

7.34 Subject.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Subject
00006 */
00007
00008 #include <algorithm>
00009 #include <memory>
00010 #include <vector>

```

```

00011 #include <string>
00012
00013 #include "ISubject.hpp"
00014
00015 #ifndef SUBJECT_HPP_
00016 #define SUBJECT_HPP_
00017
00018 class Subject : public ISubject {
00019     public:
00020         virtual ~Subject() = default;
00021
00022         void addObserver(std::shared_ptr<IObserver> observer) override;
00023
00024         void removeObserver(std::shared_ptr<IObserver> observer) override;
00025
00026         void notifyObservers() override;
00027
00028         void notifyGameEvent(GameEventType eventType, const std::string& teamName);
00029
00030     private:
00031         std::vector<std::weak_ptr<IObserver> _observers;
00032 };
00033
00034 #endif /* !SUBJECT_HPP_ */

```

7.35 Raylib.hpp

```

00001 /*
00002  ** EPITECH PROJECT, 2025
00003  ** zappy
00004  ** File description:
00005  ** Raylib
00006  */
00007
00008 #ifndef RAYLIB_HPP_
00009 #define RAYLIB_HPP_
00010 #include <memory>
00011 #include <string>
00012 #include "../IDisplay.hpp"
00013 #include "RaylibEnc/RayLibEnc.hpp"
00014
00015 class Raylib : public IDisplay {
00016     private:
00017         std::unique_ptr<RayLibEnc> _raylib;
00018
00019     public:
00020         virtual Vector2i getMonitorSize();
00021         virtual Vector2i getScreenSize();
00022
00023         virtual void initWindow(int width, int height, std::string);
00024         virtual void initCamera();
00025
00026         virtual bool isWindowReady();
00027         virtual void setTargetFPS(unsigned int FPS);
00028
00029         virtual bool isOpen();
00030         virtual void closeWindow();
00031
00032         virtual int getKeyId(enum Key);
00033
00034         virtual bool isKeyReleased(int key);
00035         virtual bool isKeyPressed(int key);
00036         virtual bool isKeyDown(int key);
00037
00038         virtual bool isGamepadAvailable();
00039
00040         virtual bool isGamepadButtonReleased(int key);
00041         virtual bool isGamepadButtonPressed(int key);
00042         virtual bool isGamepadButtonDown(int key);
00043
00044
00045         virtual bool isMouseButtonDown(int key);
00046         virtual bool isMouseButtonReleased(int key);
00047         virtual bool isMouseButtonPressed(int key);
00048
00049         virtual Vector2f getMousePosition();
00050         virtual void setMousePosition(Vector2f);
00051
00052         virtual float getMouseWheelMove();
00053
00054         virtual float getGamepadAxisMovement(int key);
00055
00056         virtual void setCameraPosition(Vector3f);
00057
00058         virtual void setCameraTarget(Vector3f);

```

```

00059
00060     virtual Vector2f getMouseDelta();
00061
00062     virtual float vector3DDistanceFromCamera(Vector3f target);
00063     virtual Vector3f vector3SubtractFromCamera(Vector3f target);
00064
00065     virtual Vector3f vector3Normalize(Vector3f);
00066
00067     virtual void enableCursor();
00068     virtual void disableCursor();
00069
00070     virtual float getFrameTime();
00071     virtual int getFPS();
00072
00073     virtual void updateCameraFreeMode(float camMovingSpeed, float camRotaSpeed);
00074
00075     virtual InputType getLastInputType() const;
00076     virtual void updateLastInputType();
00077
00078     virtual float measureText(const std::string& text, float fontSize) const;
00079
00080     virtual bool checkCollisionPointRec(Vector2f point, FloatRect rec);
00081
00082     virtual Ray3D getMouseRay(Vector2f mousePosition);
00083     virtual RayCollision3D getRayCollisionBox(Ray3D ray, BoundingBox3D box);
00084     virtual RayCollision3D getRayCollisionSphere(Ray3D ray, Vector3f center, float radius);
00085     virtual bool checkCollisionBoxes(BoundingBox3D box1, BoundingBox3D box2);
00086
00087     virtual Ray3D getMouseRayFromCurrent();
00088     virtual BoundingBox3D createBoundingBox(Vector3f center, Vector3f size);
00089     virtual BoundingBox3D createBoundingBoxFromMinMax(Vector3f min, Vector3f max);
00090
00091     virtual void beginScissorMode(IntRect);
00092     virtual void endScissorMode();
00093
00094     virtual void beginDrawing();
00095     virtual void endDrawing();
00096
00097     virtual void clearBackground(Color32);
00098
00099     virtual void begin3DMode();
00100     virtual void end3DMode();
00101
00102     virtual bool loadModel(const std::string& id, const std::string& filepath,
00103                          Vector3f center = {0.0f, 0.0f, 0.0f});
00104
00105     virtual void drawCube(Vector3f position, float width, float height, float length,
00106                          Color32 color);
00107     virtual void drawCubeWires(Vector3f position, float width, float height, float length,
00108                                Color32 color);
00109
00110     virtual void drawSphere(Vector3f position, float radius, Color32 color);
00111     virtual void drawSphereWires(Vector3f position, float radius, int rings, int slices,
00112                                  Color32 color);
00113
00114     virtual void drawCylinder(Vector3f position, float radiusTop, float radiusBottom,
00115                               float height, int slices, Color32 color);
00116     virtual void drawCylinderWires(Vector3f position, float radiusTop, float radiusBottom,
00117                                    float height, int slices, Color32 color);
00118     virtual void drawCylinderEx(Vector3f startPos, Vector3f endPos, float startRadius,
00119                                float endRadius, int sides, Color32 color);
00120
00121     virtual void drawPlane(Vector3f position, Vector2f size, Color32 color);
00122
00123     virtual void drawLine3D(Vector3f startPos, Vector3f endPos, Color32 color);
00124
00125     virtual void drawModelEx(const std::string& id, Vector3f position,
00126                             Vector3f rotationAxis, float rotationAngle, Vector3f scale,
00127                             Color32 tint = CWHITE);
00128
00129     virtual void drawText(const std::string& text, float x, float y, float fontSize,
00130                           Color32 color);
00131
00132     virtual void drawTextEx(const std::string& text, float x, float y, float fontSize,
00133                             float spacing, Color32 color);
00134
00135     virtual void drawCircle(float centerX, float centerY, float radius,
00136                             Color32 color);
00137     virtual void drawCircleLines(float centerX, float centerY,
00138                                  float radius, Color32 color);
00139
00140     virtual void drawRectangleRec(FloatRect rec, Color32 color);
00141
00142     virtual bool loadTexture(const std::string& id, const std::string& filepath);
00143
00144     virtual bool loadFont(const std::string& id, const std::string& filepath);
00145

```

```

00146     virtual void drawTexture(const std::string& id, float x, float y,
00147                             Color32 tint = CWHITE);
00148
00149     virtual void drawTextureScaled(const std::string& id, float x, float y, float width,
00150                                   float height, Color32 tint = CWHITE);
00151
00152     virtual Vector2f getTextureSize(const std::string& id) const;
00153
00154     virtual bool loadSkybox(const std::string& id, const std::string& filepath);
00155
00156     virtual void drawSkybox(const std::string& id);
00157
00158     virtual float getTime() const;
00159
00160     Raylib();
00161     ~Raylib() = default;
00162 };
00163
00164 #endif /* !RAYLIB_HPP_ */

```

7.36 RayLibEnc.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** RayLibEnc
00006 */
00007
00008 #ifndef RAYLIBENC_HPP_
00009 #define RAYLIBENC_HPP_
00010
00011 #include <string>
00012 #include <map>
00013 #include <memory>
00014 #include "raylib.h"
00015 #include "../Utils/InputType.hpp"
00016
00017 class RayLibEnc {
00018     public:
00019         RayLibEnc();
00020         ~RayLibEnc();
00021
00022         // Window management methods
00023         void initWindow(int width, int height, const std::string &title);
00024         void closeWindow();
00025         bool windowShouldClose() const;
00026         void beginDrawing();
00027         void endDrawing();
00028         void clearBackground(Color color = WHITE);
00029         bool isWindowReady() const;
00030         int getMonitorWidth(int monitor) const;
00031         int getMonitorHeight(int monitor) const;
00032         void waitTime(float seconds) const;
00033         void setTargetFPS(int fps) const;
00034         int getFPS() const;
00035         float getFrameTime() const;
00036
00037         // Collision methods
00038         bool checkCollisionPointRec(Vector2 point, Rectangle rec) const;
00039
00040         // Ray and collision methods for 3D
00041         Ray getMouseRay(Vector2 mousePosition);
00042         RayCollision getRayCollisionBox(Ray ray, BoundingBox box);
00043         RayCollision getRayCollisionSphere(Ray ray, Vector3 center, float radius);
00044         bool checkCollisionBoxes(BoundingBox box1, BoundingBox box2);
00045
00046         // Utility methods for 3D collisions
00047         Ray getMouseRayFromCurrent();
00048         BoundingBox createBoundingBox(Vector3 center, Vector3 size);
00049         BoundingBox createBoundingBoxFromMinMax(Vector3 min, Vector3 max);
00050
00051         // Texture methods
00052         void drawTextureRec(Texture2D texture, Rectangle source, Vector2 position, Color tint);
00053         void unloadTexture(Texture2D texture);
00054         Texture2D loadTextureFromFile(const std::string& filepath);
00055         void drawTextureEx(Texture2D texture, Vector2 position, Color tint);
00056         void drawTextureScaled(Texture2D texture, float x, float y, float width, float height,
00057                               Color tint);
00058
00059         // Texture map accessor methods
00060         bool hasTexture(const std::string& id) const;
00061         Texture2D getTexture(const std::string& id) const;
00062         void addTexture(const std::string& id, Texture2D texture);
00063

```

```

00064     // Input methods
00065     bool isMouseButtonDown(int button) const;
00066     bool isMouseButtonPressed(int button) const;
00067     bool isMouseButtonReleased(int button) const;
00068     bool isKeyDown(int key) const;
00069     bool isKeyPressed(int key) const;
00070     bool isKeyReleased(int key) const;
00071     Vector2 getMouseDelta();
00072     Vector2 getMousePosition() const;
00073     void setMousePosition(int x, int y);
00074     void disableCursor();
00075     void enableCursor();
00076     int getScreenWidth() const;
00077     int getScreenHeight() const;
00078     float getMouseWheelMove() const;
00079
00080     // Gamepad methods
00081     bool isGamepadAvailable(int gamepad) const;
00082     bool isGamepadButtonPressed(int gamepad, int button) const;
00083     bool isGamepadButtonDown(int gamepad, int button) const;
00084     bool isGamepadButtonReleased(int gamepad, int button) const;
00085     float getGamepadAxisMovement(int gamepad, int axis) const;
00086
00087     // Input type tracking methods
00088     InputType getLastInputType() const;
00089     void updateLastInputType();
00090
00091     // Scissor mode methods for clipping
00092     void beginScissorMode(int x, int y, int width, int height);
00093     void endScissorMode();
00094
00095     // 3D Environment methods
00096     void begin3DMode();
00097     void end3DMode();
00098     float vector3Distance(Vector3 v1, Vector3 v2) const;
00099     Vector3 vector3Normalize(Vector3 v) const;
00100     Vector3 vector3Subtract(Vector3 v1, Vector3 v2) const;
00101     Vector3 vector3Add(Vector3 v1, Vector3 v2) const;
00102
00103     // Camera methods
00104     void initCamera();
00105     void setCameraPosition(Vector3 position);
00106     void setCameraTarget(Vector3 target);
00107     void setCameraUp(Vector3 up);
00108     void setCameraFovy(float fovy);
00109     void setCameraProjection(int projection);
00110     void updateCamera(int mode = CAMERA_FREE);
00111     void updateCameraFreeMode(float camMovingSpeed, float camRotaSpeed);
00112     Camera3D getCamera() const;
00113
00114     // 3D Drawing methods
00115     void drawGrid(int slices, float spacing);
00116     void drawCube(Vector3 position, float width, float height, float length, Color color);
00117     void drawCubeWires(Vector3 position, float width, float height, float length,
00118         Color color);
00119     void drawSphere(Vector3 position, float radius, Color color);
00120     void drawSphereWires(Vector3 position, float radius, int rings, int slices,
00121         Color color);
00122     void drawCylinder(Vector3 position, float radiusTop, float radiusBottom,
00123         float height, int slices, Color color);
00124     void drawCylinderWires(Vector3 position, float radiusTop, float radiusBottom,
00125         float height, int slices, Color color);
00126     void drawCylinderEx(Vector3 startPos, Vector3 endPos, float startRadius,
00127         float endRadius, int sides, Color color);
00128     void drawPlane(Vector3 position, Vector2 size, Color color);
00129     void drawLine3D(Vector3 startPos, Vector3 endPos, Color color);
00130
00131     // 3D Model methods
00132     bool loadModel(const std::string& id, const std::string& filepath,
00133         Vector3 center = {0.0f, 0.0f, 0.0f});
00134     void drawModel(const std::string& id, Vector3 position, float scale,
00135         Color tint = WHITE);
00136     void drawModelEx(const std::string& id, Vector3 position, Vector3 rotationAxis,
00137         float rotationAngle, Vector3 scale, Color tint = WHITE);
00138     void drawModelWires(const std::string& id, Vector3 position, float scale,
00139         Color tint = WHITE);
00140     void drawModelWiresEx(const std::string& id, Vector3 position, Vector3 rotationAxis,
00141         float rotationAngle, Vector3 scale, Color tint = WHITE);
00142     void unloadModel(const std::string& id);
00143     void unloadAllModels();
00144     bool modelExists(const std::string& id) const;
00145
00146     // Skybox methods
00147     bool loadSkybox(const std::string& id, const std::string& filepath);
00148     void drawSkybox(const std::string& id);
00149     Color getDayNightColor(float cycleTime);
00150     float getTime() const;

```



```

00036
00037     inline const int FAILURE_EXIT_CODE = 84;
00038     inline const int SUCCESS_EXIT_CODE = 0;
00039 };
00040
00041 namespace colors {
00042
00043     inline const char *T_BOLD = "\033[1m";
00044     inline const char *T_RED = "\033[1m\033[31m";
00045     inline const char *T_GREEN = "\033[1m\033[32m";
00046     inline const char *T_YELLOW = "\033[1m\033[33m";
00047     inline const char *T_BLUE = "\033[1m\033[34m";
00048     inline const char *T_MAGENTA = "\033[1m\033[35m";
00049     inline const char *T_CYAN = "\033[1m\033[36m";
00050     inline const char *T_WHITE = "\033[1m\033[37m";
00051     inline const char *RESET = "\033[0m";
00052
00053 };
00054
00055 namespace zappy::structs {
00056
00057     struct Config {
00058         int port;
00059         std::string hostname;
00060     };
00061
00062     struct Tile {
00063         int x;
00064         int y;
00065         int food;
00066         int linemate;
00067         int deraumere;
00068         int sibur;
00069         int mendiane;
00070         int phiras;
00071         int thystame;
00072
00073         Tile(int _x = 0, int _y = 0, int _food = 0, int _linemate = 0,
00074             int _deraumere = 0, int _sibur = 0, int _mendiane = 0,
00075             int _phiras = 0, int _thystame = 0)
00076             : x(_x), y(_y), food(_food), linemate(_linemate),
00077             deraumere(_deraumere), sibur(_sibur),
00078             mendiane(_mendiane), phiras(_phiras), thystame(_thystame) {}
00079     };
00080
00081     struct Inventory {
00082         int food;
00083         int linemate;
00084         int deraumere;
00085         int sibur;
00086         int mendiane;
00087         int phiras;
00088         int thystame;
00089
00090         Inventory(int _food = 0, int _linemate = 0, int _deraumere = 0,
00091             int _sibur = 0, int _mendiane = 0, int _phiras = 0,
00092             int _thystame = 0)
00093             : food(_food), linemate(_linemate), deraumere(_deraumere),
00094             sibur(_sibur), mendiane(_mendiane), phiras(_phiras),
00095             thystame(_thystame) {}
00096     };
00097
00098     struct Player {
00099         int number;
00100         int x;
00101         int y;
00102         int orientation;
00103         int level;
00104         std::string teamName;
00105         struct Inventory inventory;
00106
00107         Player(int _number = 0, int _x = 0, int _y = 0, int _orientation = 0,
00108             int _level = 1, const std::string &_teamName = "",
00109             struct Inventory _inventory = Inventory())
00110             : number(_number), x(_x), y(_y), orientation(_orientation),
00111             level(_level), teamName(_teamName), inventory(_inventory) {}
00112     };
00113
00114     struct Incantation {
00115         int x;
00116         int y;
00117         int level;
00118         std::vector<int> players;
00119
00120         Incantation(int _x = 0, int _y = 0, int _level = 1,
00121             const std::vector<int> &_players = {})
00122             : x(_x), y(_y), level(_level), players(_players) {}
00123     };

```

```

00123
00124     struct Egg {
00125         int eggNumber;
00126         int playerNumber;
00127         int x;
00128         int y;
00129         bool hatched;
00130         std::string teamName;
00131
00132         Egg(int _eggNumber = 0, int _playerNumber = 0, int _x = 0, int _y = 0,
00133             bool _hatched = false, const std::string &_teamName = "")
00134             : eggNumber(_eggNumber), playerNumber(_playerNumber), x(_x), y(_y),
00135             hatched(_hatched), teamName(_teamName) {}
00136     };
00137 };
00138
00139 namespace zappy::gui {
00140
00141     inline const std::string WINDOW_TITLE = "Zappy GUI";
00142     inline const std::string CUSTOM_FONT_PATH = "gui/assets/fonts/fall.ttf";
00143     inline const int FPS = 120;
00144     inline const float CAMERA_SENSITIVITY = 0.001f;
00145     inline const float GAMEPAD_STICK_SENSITIVITY = 3.0f;
00146     inline const float GAMEPAD_DEADZONE = 0.2f;
00147     inline const float POSITION_MULTIPLIER = 2.2f;
00148
00149     inline const float FOG_DISTANCE_MAX = 60.0f;
00150     inline const float DURATION_DAYNIGHT_CYCLE = 120.0f;
00151
00152     inline const float EGG_SCALE = 1.0f;
00153     inline const float FOOD_SCALE = 0.005f;
00154     inline const float FOOD_FLOAT_AMPLITUDE = 0.05f;
00155     inline const float FOOD_FLOAT_SPEED = 0.10f;
00156
00157     inline const float LINEMATE_SCALE = 0.2f; // soccerball
00158     inline const float DERAUMERE_SCALE = 0.15f; // beachball
00159     inline const float SIBUR_SCALE = 0.15f; // basketball
00160     inline const float MENDIANE_SCALE = 0.18f; // bowlingball
00161     inline const float PHIRAS_SCALE = 0.1f; // eightball
00162     inline const float THYSTAME_SCALE = 0.1f; // tennisball
00163
00164     inline const float PLAYER_ROTATION_SPEED = 720.0f;
00165     inline const float ROTATION_INTERPOLATION_THRESHOLD = 1.0f;
00166
00167     inline const float PLAYER_MOVEMENT_SPEED = 8.0f;
00168     inline const float MOVEMENT_INTERPOLATION_THRESHOLD = 0.05f;
00169
00170     enum class CameraMode {
00171         FREE = 0,
00172         TARGETED = 1,
00173         PLAYER = 2,
00174         NB_MODES = 3,
00175     };
00176
00177     struct PlayerModelInfo {
00178         std::string name;
00179         std::string modelPath;
00180         Vector3f center;
00181         Vector3f scale;
00182         float rotation;
00183     };
00184 };
00185
00186     inline const std::vector<PlayerModelInfo> PLAYER_MODELS_INFO = {
00187         {"playerLv11", "gui/assets/models/playerLv11.glb",
00188          {0.0f, -0.0f, 0.0f}, {0.005f, 0.005f, 0.005f}, 0.0f},
00189         {"playerLv12", "gui/assets/models/playerLv12.glb",
00190          {0.0f, -0.5f, 0.0f}, {0.25f, 0.25f, 0.25f}, 0.0f},
00191         {"playerLv13", "gui/assets/models/playerLv13.glb",
00192          {0.0f, 20.0f, 0.0f}, {0.0045f, 0.0045f, 0.0045f}, 0.0f},
00193         {"playerLv14", "gui/assets/models/playerLv14.glb",
00194          {0.0f, 0.0025f, 0.0f}, {40.0f, 40.0f, 40.0f}, -90.0f},
00195         {"playerLv15", "gui/assets/models/playerLv15.glb",
00196          {8.0f, -1.8f, 0.0f}, {0.2f, 0.2f, 0.2f}, 0.0f},
00197         {"playerLv16", "gui/assets/models/playerLv16.glb",
00198          {0.0f, 20.0f, 0.0f}, {0.009f, 0.009f, 0.009f}, 0.0f},
00199         {"playerLv17", "gui/assets/models/playerLv17.glb",
00200          {0.0f, 0.4f, 0.0f}, {0.25f, 0.25f, 0.25f}, 0.0f},
00201         {"playerLv18", "gui/assets/models/playerLv18.glb",
00202          {0.0f, 1.0f, 0.0f}, {0.085f, 0.085f, 0.085f}, 0.0f}
00203     };
00204 }
00205
00206 #endif /* !CONSTANTS_HPP_ */

```


7.38 GamepadConstants.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GamepadConstants
00006 */
00007
00008 #ifndef GAMEPAD_CONSTANTS_HPP_
00009 #define GAMEPAD_CONSTANTS_HPP_
00010
00011 #ifndef GAMEPAD_AXIS_LEFT_X
00012     #define GAMEPAD_AXIS_LEFT_X      0
00013     #define GAMEPAD_AXIS_LEFT_Y      1
00014     #define GAMEPAD_AXIS_RIGHT_X     2
00015     #define GAMEPAD_AXIS_RIGHT_Y     3
00016 #endif
00017
00018 #ifndef GAMEPAD_BUTTON_A
00019     #define GAMEPAD_AXIS_LEFT_TRIGGER 4
00020     #define GAMEPAD_AXIS_RIGHT_TRIGGER 5
00021     #define GAMEPAD_BUTTON_A         6
00022     #define GAMEPAD_BUTTON_B         5
00023     #define GAMEPAD_BUTTON_X         9
00024     #define GAMEPAD_BUTTON_Y         8
00025     #define GAMEPAD_BUTTON_START     17
00026     #define GAMEPAD_BUTTON_SELECT    16
00027     #define GAMEPAD_BUTTON_UP        1
00028     #define GAMEPAD_BUTTON_RIGHT     2
00029     #define GAMEPAD_BUTTON_DOWN      3
00030     #define GAMEPAD_BUTTON_LEFT      4
00031     #define GAMEPAD_BUTTON_LEFT_SHOULDER 10
00032     #define GAMEPAD_BUTTON_RIGHT_SHOULDER 12
00033     #define GAMEPAD_BUTTON_LEFT_TRIGGER 13
00034     #define GAMEPAD_BUTTON_RIGHT_TRIGGER 15
00035 #endif
00036
00037 #endif /* !GAMEPAD_CONSTANTS_HPP_ */

```

7.39 HelpText.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** HelpText
00006 */
00007
00008 #ifndef HELP_TEXT_HPP_
00009 #define HELP_TEXT_HPP_
00010
00011 namespace zappy::constants {
00012
00013     inline const char *HELP_TITLE =
00014         "HELP";
00015
00016     inline const char *HELP_SECTION_1 =
00017         "Game Overview";
00018
00019     inline const char *HELP_SECTION_1_CONTENT =
00020         "Zappy is a game where AI-controlled players compete to collect resources\n"
00021         "and level up on a dynamically changing map. The GUI allows you to visualize\n"
00022         "the game state, players, and resources in real-time.";
00023
00024     inline const char *HELP_SECTION_2 =
00025         "Controls";
00026
00027     inline const char *HELP_SECTION_2_CONTENT =
00028         "Camera Movement:\n"
00029         "  - Arrow keys or ZQSD: Move camera\n"
00030         "  - Controller: Use left stick to move camera\n"
00031         "  - Right mouse button + drag: Rotate camera\n"
00032         "Interface:\n"
00033         "  - Click on players to see their stats\n"
00034         "  - Click on tiles to see their stats\n"
00035         "  - Use the RESET CAMERA button to return to default view\n"
00036         "  - Use the Settings button to adjust game settings";
00037
00038     inline const char *HELP_SECTION_3 =
00039         "Teams and Players";
00040
00041     inline const char *HELP_SECTION_3_CONTENT =
00042         "The left panel shows all teams and their player IDs.\n"

```

```

00043         "Players have different levels based on collected resources.\n"
00044         "The team that first gets a player to level 8 wins the game.";
00045
00046     inline const char *HELP_SECTION_4 =
00047         "Resources";
00048
00049     inline const char *HELP_SECTION_4_CONTENT =
00050         "Resources on the map are represented by different colored objects.\n"
00051         "Players collect these resources to perform rituals and level up.";
00052
00053     inline const char *HELP_SECTION_5 =
00054         "Levels";
00055
00056     inline const char *HELP_SECTION_6 =
00057         "Items";
00058
00059 } // namespace zappy::constants
00060
00061 #endif /* !HELP_TEXT_HPP_ */

```

7.40 InputType.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** InputType
00006 */
00007
00008 #ifndef INPUTTYPE_HPP_
00009 #define INPUTTYPE_HPP_
00010
00011 enum class InputType {
00012     KEYBOARD_MOUSE,
00013     GAMEPAD,
00014     NONE
00015 };
00016
00017 #endif /* !INPUTTYPE_HPP_ */

```

7.41 algo.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** algo
00006 */
00007
00008 #ifndef ALGO_H_
00009 #define ALGO_H_
00010
00011 typedef struct tiles_s {
00012     int x;
00013     int y;
00014 } tiles_t;
00015
00016 /* Algo.c */
00017 tiles_t *shuffle_fisher(int width, int height);
00018
00019 #endif /* !ALGO_H_ */

```

7.42 game.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** game
00006 */
00007
00008 #include "buffer.h"
00009 #include <time.h>
00010 #include <pthread.h>
00011
00012 #ifndef GAME_H_
00013 #define GAME_H_
00014
00015 typedef struct action_request_s action_request_t;

```

```

00016 typedef struct action_queue_s action_queue_t;
00017 typedef struct player_s player_t;
00018
00019 /* Definition of the directions */
00020 typedef enum direction_e {
00021     NORTH = 1,
00022     EAST = 2,
00023     SOUTH = 3,
00024     WEST = 4
00025 } direction_t;
00026
00027 /* definition of the different element on the map */
00028 typedef enum crystal_e {
00029     FOOD,
00030     LINEMATE,
00031     DERAUMERE,
00032     SIBUR,
00033     MENDIANE,
00034     PHIRAS,
00035     THYSTAME
00036 } crystal_t;
00037
00038
00039 /* This enum defines the priority of the action in the queue */
00040 typedef enum action_priority_e {
00041     PRIORITY_CRITICAL = 0,
00042     PRIORITY_HIGH = 1,
00043     PRIORITY_MEDIUM = 2,
00044     PRIORITY_LOW = 3
00045 } action_priority_t;
00046
00047 /* This structure allows use to define a 'queue' of the requests */
00048 typedef struct action_queue_s {
00049     action_request_t *head;
00050     action_request_t *tail;
00051     int count;
00052 } action_queue_t;
00053
00054
00055 typedef struct egg_s {
00056     int id; /* Id of the egg */
00057     int posX;
00058     int posY;
00059     char *teamName; /* Name of the team that laid it */
00060     int idLayer; /* Id of the player that layed it */
00061     bool isHatched;
00062     struct egg_s *next;
00063 } egg_t;
00064
00065 /* Struct that "handles" the network element */
00066 typedef struct network_s {
00067     int fd;
00068     buffer_t *buffer;
00069 } network_t;
00070
00071 /* Struct defining the inventory of tiles and players */
00072 typedef struct inventory_s {
00073     int nbFood;
00074     int nbLinemate;
00075     int nbDeraumere;
00076     int nbSibur;
00077     int nbMendiane;
00078     int nbPhiras;
00079     int nbThystame;
00080 } inventory_t;
00081
00082 /* Definition of the incantation structure */
00083 typedef struct incantation_s {
00084     int levelt_to_reach;
00085     int nb_players;
00086     inventory_t required_inventory;
00087 } incantation_t;
00088
00089
00090 /* Player struct */
00091 typedef struct player_s {
00092     int id;
00093     network_t *network;
00094     int level;
00095     int posX;
00096     int posY;
00097     direction_t direction;
00098     inventory_t *inventory;
00099     char *team;
00100     /* New additions for the smart pollin */
00101     action_queue_t *pending_actions;
00102     time_t last_action_time;

```

```

00103     bool is_busy;
00104     int remaining_cooldown;
00105     char *current_action;
00106     /* Food timer for health system */
00107     int food_timer; /* Time units until next food consumption */
00108     time_t last_food_check; /* Last time food was checked */
00109
00110     struct player_s *next;
00111 } player_t;
00112
00113 /* This structure define the request struct */
00114 typedef struct action_request_s {
00115     char *command;
00116     time_t timestamp;
00117     float time_limit; // in game ticks (7/f, 42/f, etc.)
00118     action_priority_t priority;
00119     player_t *player;
00120     struct action_request_s *next;
00121 } action_request_t;
00122
00123 /* Team Struct */
00124 typedef struct team_s {
00125     char *name;
00126     int nbPlayers;
00127     int nbPlayerAlive;
00128     player_t *players;
00129     struct team_s *next;
00130 } team_t;
00131
00132
00133 /* Structure that holds the size and array of tiles */
00134 typedef struct map_t {
00135     int width;
00136     int height;
00137     egg_t *currentEggs; /* List of current eggs */
00138     inventory_t **tiles; /* Here we call inv for the tile*/
00139 } map_t;
00140
00141
00142 /* Map struct */
00143 typedef struct game_s {
00144     team_t *teams;
00145     map_t *map;
00146 } game_t;
00147
00148 #endif /* !GAME_H_ */

```

7.43 my.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** my
00006 */
00007
00008 #ifndef MY_H_
00009     #define MY_H_
00010
00011 int int_str_len(int value);
00012 char *my_itoa(unsigned int nb);
00013 int is_only_digits(const char *str);
00014 int my_unsignedlen(unsigned int nb);
00015
00016 #endif /* !MY_H_ */

```

7.44 my.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** my
00006 */
00007
00008 #ifndef MY_H_
00009     #define MY_H_
00010
00011 int int_str_len(int value);
00012 char *my_itoa(unsigned int nb);
00013 int is_only_digits(const char *str);
00014 int my_unsignedlen(unsigned int nb);

```

```

00015
00016 #endif /* !MY_H_ */

```

7.45 zappy.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** Zappy
00004 ** File description:
00005 ** Server :: Zappy header
00006 */
00007
00008 #include <stdbool.h>
00009 #include <poll.h>
00010 #include "game.h"
00011 #include "my.h"
00012
00013 #ifndef ZAPPY_H_
00014     #define ZAPPY_H_
00015
00016     /* items handler */
00017     typedef struct {
00018         char *name;
00019         void (*add_func)(inventory_t *);
00020     } item_handler_t;
00021
00022
00023     /* Cli parameter of the server */
00024     typedef struct params_s {
00025         int port;
00026         int x;
00027         int y;
00028         int nb_team;
00029         char **teams;
00030         int nb_client;
00031         int freq;
00032         bool is_debug;
00033     } params_t;
00034
00035     /* Structure to handle the network side of the gui*/
00036     typedef struct graph_net_s {
00037         int fd;
00038         bool mapSent;
00039         struct graph_net_s *next;
00040     } graph_net_t;
00041
00042     /* Unified polling structure for all clients */
00043     typedef struct unified_poll_s {
00044         struct pollfd *fds;
00045         int count;
00046         int capacity;
00047     } unified_poll_t;
00048
00049     /* Server part of the network */
00050     typedef struct server_s {
00051         int sockfd;
00052         struct pollfd pollserver;
00053     } server_t;
00054
00055     typedef struct zappy_s {
00056         server_t *network;
00057         game_t *game;
00058         graph_net_t *graph;
00059         params_t *params;
00060         unified_poll_t *unified_poll;
00061     } zappy_t;
00062
00063     typedef struct command_pf_s {
00064         char const *flag;
00065         bool (*checker)(const char *, const char *, params_t *);
00066     } command_pf_t;
00067
00068     typedef struct {
00069         char *command;
00070         float base_time;
00071         action_priority_t priority;
00072         int (*handler)(player_t *, char *, zappy_t *);
00073     } command_info_t;
00074
00075     typedef struct graphic_pf_s {
00076         char *command;
00077         int (*handler)(zappy_t *zappy, graph_net_t *graphic, char *message);
00078     } graphic_pf_t;
00079
00080     /* messages.c */

```

```

00081 int helper(void);
00082 void error_message(const char *message);
00083 void valid_message(char const *message);
00084
00085 /* checkers.c */
00086 bool check_port(char const *flag, char const *value, params_t *params);
00087 bool check_width(char const *flag, char const *value, params_t *params);
00088 bool check_height(char const *flag, char const *value, params_t *params);
00089 bool check_client(char const *flag, char const *value, params_t *params);
00090 bool check_freq(char const *flag, char const *value, params_t *params);
00091
00092 /* unified_poll.c */
00093 unified_poll_t *init_unified_poll(void);
00094 void free_unified_poll(unified_poll_t *poll_struct);
00095 int add_fd_to_poll(unified_poll_t *poll_struct, int fd, short events);
00096 int remove_fd_from_poll(unified_poll_t *poll_struct, int fd);
00097 void rebuild_poll_fds(zappy_t *zappy);
00098 void poll_all_clients(zappy_t *zappy);
00099
00100
00101 /* signal.c */
00102 void setup_signal(void);
00103 int *get_running_state(void);
00104
00105 /* params.c */
00106 params_t *check_args(int argc, char **argv);
00107 void *free_params(params_t *params);
00108
00109 /* params_checker.c */
00110 bool validate_no_extra_args(int argc, char **argv);
00111
00112 /* server.c */
00113 zappy_t *init_server(int argc, char **argv);
00114 void *free_zappy(zappy_t *server);
00115
00116 /* protocol.c */
00117 int start_protocol(zappy_t *server);
00118
00119 /* client.c */
00120 bool process_new_client(const char *team_name, int fd, zappy_t *server);
00121 team_t *add_client_to_team(const char *team_name, int fd, zappy_t *server);
00122 void check_player_status(zappy_t *zappy);
00123 void remove_player_by_fd(zappy_t *zappy, int fd);
00124
00125 /* init_map.c */
00126 void init_game(zappy_t *server);
00127 int distribute_resources(zappy_t *z);
00128
00129 /* init_team.c */
00130 void init_teams(zappy_t *server);
00131
00132 /* accept.c */
00133 int accept_client(zappy_t *server);
00134
00135 /* refill_food.c */
00136 void count_current_resources(zappy_t *z, int current_count[7]);
00137 void refill_food(zappy_t *zappy);
00138
00139 /* free server */
00140 void *free_zappy(zappy_t *server);
00141 void *free_params(params_t *params);
00142 void *free_player(player_t *player);
00143 void free_map(map_t *map);
00144
00145 /* Function to send info to the gui */
00146 int send_map_size(zappy_t *server);
00147 int send_entire_map(zappy_t *server);
00148 int send_map_tile(inventory_t **tiles, zappy_t *server,
00149 int posX, int posY);
00150 int send_team_name(zappy_t *server);
00151 int send_egg(zappy_t *zappy, egg_t *egg);
00152 int send_entire_egg_list(zappy_t *zappy);
00153 int send_time_message(zappy_t *zappy);
00154 int send_egg_death(zappy_t *zappy, egg_t *egg);
00155 int send_egg_connect(zappy_t *zappy, egg_t *currentEgg);
00156 int send_player_connect(zappy_t *zappy, player_t *player);
00157 int send_player_pos(zappy_t *zappy, player_t *player);
00158 int send_player_level(zappy_t *zappy, player_t *player);
00159 int send_player_connect_to_specific_gui(graph_net_t *fd, player_t *p);
00160 int send_player_inventory(zappy_t *zappy, player_t *player);
00161 int send_player_expelled(zappy_t *zappy, player_t *player);
00162 int send_broadcast_to_all(zappy_t *zappy, const char *message);
00163 int send_broadcast_to_player(zappy_t *zappy, player_t *player,
00164 const char *message);
00165 int send_player_laying_egg(zappy_t *zappy, player_t *player);
00166 int send_resource_dropped(zappy_t *zappy, player_t *player,
00167 int resourceType);

```

```

00168 int send_ressource_collected(zappy_t *zappy, player_t *player,
00169     int ressourceType);
00170 int send_player_death(zappy_t *zappy, player_t *player);
00171 int send_updated_time(zappy_t *zappy, int time);
00172 int send_end_game(zappy_t *zappy, const char *teamName);
00173 int send_str_message(zappy_t *zappy, const char *message);
00174 int send_unknown_command(zappy_t *zappy);
00175 int send_command_parameter(zappy_t *zappy);
00176 int send_start_incantation(zappy_t *zappy, player_t *player, int *player_list,
00177     int nb_player);
00178 int send_end_incantation(zappy_t *zappy, player_t *player, char *result);
00179
00180 /* init_egg.c */
00181 void init_egg(zappy_t *zappy);
00182 egg_t *add_egg_node(int id, int *pos, char *team_name, int id_layer);
00183 egg_t *kil_egg_node(egg_t **head, int egg_id);
00184
00185 /* AI messages */
00186 int forward_message(player_t *player, params_t *params);
00187
00188 /* Pollin handler */
00189 void process_player_actions(player_t *player, zappy_t *zappy);
00190 void process_player_actions_tick(zappy_t *zappy);
00191 void execute_action(player_t *player, action_request_t *action,
00192     zappy_t *zappy);
00193 void queue_action(player_t *player, char *command, zappy_t *zappy);
00194 action_queue_t *init_action_queue(void);
00195 void free_action_queue(action_queue_t *queue);
00196 action_request_t *create_action_request(char *command, player_t *player,
00197     int frequency);
00198 const command_info_t *find_command_info(char *command);
00199 action_request_t *dequeue_highest_priority_action(action_queue_t *queue);
00200 void free_action_request(action_request_t *action);
00201 void insert_action_by_priority(action_queue_t *queue,
00202     action_request_t *action);
00203
00204 /* Unified polling functions */
00205 unified_poll_t *init_unified_poll(void);
00206 void free_unified_poll(unified_poll_t *poll_struct);
00207 int add_fd_to_poll(unified_poll_t *poll_struct, int fd, short events);
00208 int remove_fd_from_poll(unified_poll_t *poll_struct, int fd);
00209 void poll_all_clients(zappy_t *zappy);
00210 void rebuild_poll_fds(zappy_t *zappy);
00211
00212 /* This is the definition of the array function of the commands */
00213 int handle_forward(player_t *player, char *command, zappy_t *zappy);
00214
00215 int handle_left(player_t *player, char *command, zappy_t *zappy);
00216 int left_message(player_t *player);
00217 int print_left_server(player_t *player);
00218
00219 int handle_right(player_t *player, char *command, zappy_t *zappy);
00220 int print_right_server(player_t *player);
00221 int right_message(player_t *player);
00222
00223 int handle_connect_nbr(player_t *player, char *command, zappy_t *zappy);
00224 int handle_eject(player_t *player, char *command, zappy_t *zappy);
00225
00226 /* fork */
00227 int handle_fork(player_t *player, char *command, zappy_t *zappy);
00228 int handle_fork_end(player_t *player, zappy_t *zappy);
00229
00230 int print_look_server(player_t *player);
00231
00232 /* Incantation handler */
00233 int handle_incantation(player_t *player, char *command, zappy_t *zappy);
00234 int check_player_on_tile(player_t *player, zappy_t *zappy);
00235 void increase_level_player(int *player_list, int nb_players, zappy_t *zappy);
00236 int *get_player_on_tile_id(int posX, int posY, zappy_t *zappy, int nb_players);
00237 int handle_end_incantation(player_t *player, zappy_t *zappy);
00238 int get_nb_player_on_tile(int posX, int posY, zappy_t *zappy, int level);
00239 void mark_players_incanting(int *player_list, int nb_players, zappy_t *zappy);
00240 void remove_crystal_from_tiles(int posX, int posY, int level, zappy_t *zappy);
00241 int validate_and_get_players(player_t *player, zappy_t *zappy,
00242     int **player_list);
00243
00244
00245 int handle_inventory(player_t *player, char *command, zappy_t *zappy);
00246 int inventory_message(player_t *player);
00247 int print_inventory_server(player_t *player, int len);
00248
00249 int handle_broadcast(player_t *player, char *command, zappy_t *zappy);
00250 int broadcast_text(player_t *source, player_t *dest, char *text,
00251     zappy_t *zappy);
00252
00253 int handle_look(player_t *player, char *command, zappy_t *zappy);
00254 int handle_set(player_t *player, char *command, zappy_t *zappy);

```

```

00255 int handle_take(player_t *player, char *command, zappy_t *zappy);
00256
00257 /* graphic_clinet.c */
00258 graph_net_t *add_graph_node(graph_net_t **head, int fd);
00259 graph_net_t *remove_graph_node(graph_net_t **head, int fd);
00260 int poll_graphic_commands(zappy_t *zappy, graph_net_t *current,
00261     char *buffer);
00262
00263
00264 /* Element handler.c */
00265 void add_food(inventory_t *inventory);
00266 void add_linemate(inventory_t *inventory);
00267 void add_deraumere(inventory_t *inventory);
00268 void add_sibur(inventory_t *inventory);
00269 void add_mendiane(inventory_t *inventory);
00270 void add_phiras(inventory_t *inventory);
00271 void add_thystame(inventory_t *inventory);
00272
00273 void rm_food(inventory_t *inventory);
00274 void rm_linemate(inventory_t *inventory);
00275 void rm_deraumere(inventory_t *inventory);
00276 void rm_sibur(inventory_t *inventory);
00277 void rm_mendiane(inventory_t *inventory);
00278 void rm_phiras(inventory_t *inventory);
00279 void rm_thystame(inventory_t *inventory);
00280
00281 /* Element handler.c */
00282 int msz(zappy_t *zappy, graph_net_t *graphic, char *message);
00283 int bct(zappy_t *zappy, graph_net_t *graphic, char *message);
00284 int mct(zappy_t *zappy, graph_net_t *graphic, char *message);
00285 int tna(zappy_t *zappy, graph_net_t *graphic, char *message);
00286 int ppo(zappy_t *zappy, graph_net_t *graphic, char *message);
00287 int plv(zappy_t *zappy, graph_net_t *graphic, char *message);
00288 int plu(zappy_t *zappy, graph_net_t *graphic, char *message);
00289 int pld(zappy_t *zappy, graph_net_t *graphic, char *message);
00290 int pin(zappy_t *zappy, graph_net_t *graphic, char *message);
00291 int sgt(zappy_t *zappy, graph_net_t *graphic, char *message);
00292 int sst(zappy_t *zappy, graph_net_t *graphic, char *message);
00293 int kil(zappy_t *zappy, graph_net_t *graphic, char *message);
00294 int tar(zappy_t *zappy, graph_net_t *graphic, char *message);
00295 int tsr(zappy_t *zappy, graph_net_t *graphic, char *message);
00296 int pia(zappy_t *zappy, graph_net_t *graphic, char *message);
00297 int pis(zappy_t *zappy, graph_net_t *graphic, char *message);
00298 int send_bct_message(graph_net_t *graphic, int x, int y,
00299     inventory_t *inventory);
00300 int send_pin_message(graph_net_t *graphic, player_t *player);
00301
00302 /* player_id.c */
00303 player_t *get_player_by_id(game_t *game, int player_id);
00304 int get_next_free_id(zappy_t *server);
00305 void verify_player_id(zappy_t *zappy, player_t *player);
00306 #endif /* !ZAPPY_H_ */

```

7.46 buffer.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** buffer
00006 */
00007
00008 #include <stddef.h>
00009
00010 #ifndef BUFFER_H_
00011     #define BUFFER_H_
00012
00013     #define BUFFER_SIZE 1024
00014
00015
00016 typedef struct buffer_s {
00017     char data[BUFFER_SIZE];
00018     int head;
00019     int tail;
00020     int full;
00021 } buffer_t;
00022
00023 /* buffer.c */
00024 int advance(int idx);
00025 void cb_write(buffer_t *cb, char c);
00026 int cb_getline(buffer_t *cb, char *line, int max_len);
00027
00028 #endif /* !BUFFER_H_ */

```


7.47 buffer.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** buffer
00006 */
00007
00008 #include <stddef.h>
00009
00010 #ifndef BUFFER_H_
00011     #define BUFFER_H_
00012
00013     #define BUFFER_SIZE 1024
00014
00015
00016 typedef struct buffer_s {
00017     char data[BUFFER_SIZE];
00018     int head;
00019     int tail;
00020     int full;
00021 } buffer_t;
00022
00023 /* buffer.c */
00024 int advance(int idx);
00025 void cb_write(buffer_t *cb, char c);
00026 int cb_getline(buffer_t *cb, char *line, int max_len);
00027
00028 #endif /* !BUFFER_H_ */

```

7.48 network.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** network
00006 */
00007
00008 #ifndef NETWORK_H_
00009     #define NETWORK_H_
00010
00011 /* Write an error message */
00012 void error_print(char const *message);
00013 /* Set the socket of the file descriptor */
00014 int set_socket(void);
00015 /* Bind the file descriptor to the port */
00016 int bind_socket(int fd, int port);
00017 /* Specify the queue the fd will use */
00018 int listen_socket(int fd, int backlog);
00019
00020 /* Close the server */
00021 void close_fd(int fd);
00022
00023 /* Accept new connection */
00024 int accept_connection(int server_fd);
00025 /* Handle Message input */
00026 char *get_message(int fd, int timeout);
00027 /* Hello */
00028 int write_message(int fd, const char *message);
00029 #endif /* !NETWORK_H_ */

```

7.49 network.h

```

00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** network
00006 */
00007
00008 #ifndef NETWORK_H_
00009     #define NETWORK_H_
00010
00011 /* Write an error message */
00012 void error_print(char const *message);
00013 /* Set the socket of the file descriptor */
00014 int set_socket(void);
00015 /* Bind the file descriptor to the port */
00016 int bind_socket(int fd, int port);
00017 /* Specify the queue the fd will use */

```

```
00018 int listen_socket(int fd, int backlog);
00019
00020 /* Close the server */
00021 void close_fd(int fd);
00022
00023 /* Accept new connetion */
00024 int accept_connection(int server_fd);
00025 /* Handle Message input */
00026 char *get_message(int fd, int timeout);
00027 /* Hello */
00028 int write_message(int fd, const char *message);
00029 #endif /* !NETWORK_H_ */
```

7.50 fake_malloc.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** fake_malloc
00006 */
00007
00008 #ifndef FAKE_MALLOC_H_
00009     #define FAKE_MALLOC_H_
00010
00011
00012 void enable_malloc_failure(int after_calls);
00013 void disable_malloc_failure(void);
00014 void reset_malloc_counter(void);
00015 void *malloc(size_t size);
00016 void *calloc(size_t nmemb, size_t size);
00017
00018 #endif /* !FAKE_MALLOC_H_ */
```