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%20↩
Python-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

- Eliott 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:

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

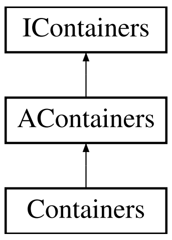
Here is a list of all documented files with brief descriptions:

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 IContainers

- 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

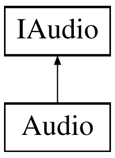
- **_signal_handler**
- **_child_signal_handler**

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()

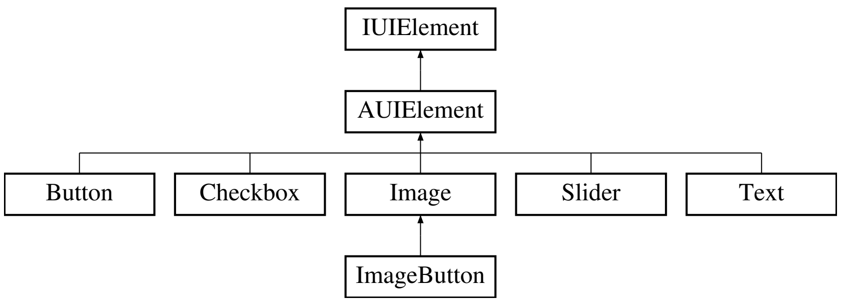
```
void Audio::stopSound (
             const std::string & id )  [virtual]
```

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

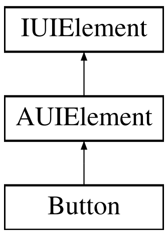
- char **data** [BUFFER_SIZE]
- int **head**
- int **tail**
- int **full**

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 &center)
- 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

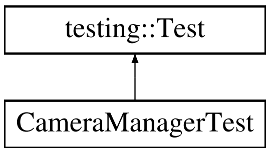
- float **_cameraMovingSpeed** = 15.0f
- float **_cameraRotaSpeed** = 2.0f
- float **_cameraZoomSpeed** = 120.0f
- std::shared_ptr< IDisplay > **_display**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< Map > **_map**
- Vector3f **_mapCenter**
- int **_mapWidth**
- int **_mapHeight**
- 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

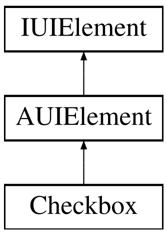
- std::unique_ptr< CameraManager > **cameraManager**
- std::shared_ptr< testing::NiceMock< MockDisplay > > **mockDisplay**
- std::shared_ptr< testing::NiceMock< MockGameInfos > > **mockGameInfos**
- std::shared_ptr< testing::NiceMock< MockMap > > **mockMap**
- std::vector< zappy::structs::Player > **testPlayersList**
- 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 IUIElement.

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 IUIElement.

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

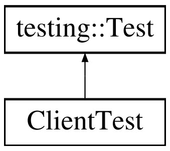
- zappy::structs::Config **_config**
- std::shared_ptr< ICommunication > **_communication**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::unique_ptr< MsgHandler > **_msgHandler**
- std::shared_ptr< GUI > **_gui**
- std::shared_ptr< GuiObserver > **_guiObserver**

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

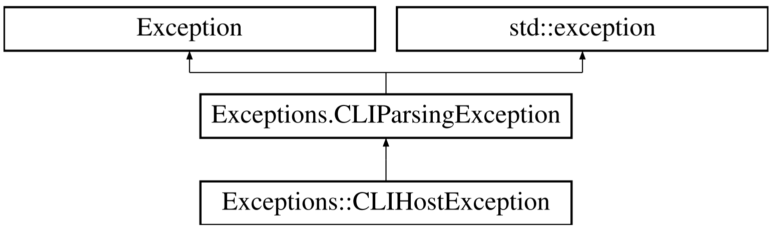
- std::stringstream **buffer**
- std::streambuf ∗ **originalCout**

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

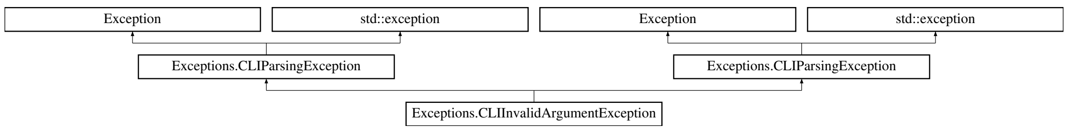
- **__init__** (self, str message)
- **CLIParsingException** (const std::string &message)
- const char ∗ **what** () const noexcept override

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:


Exception
std::exception
Exception
std::exception
Exceptions.CLIParsingException
Exceptions.CLIParsingException
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__()

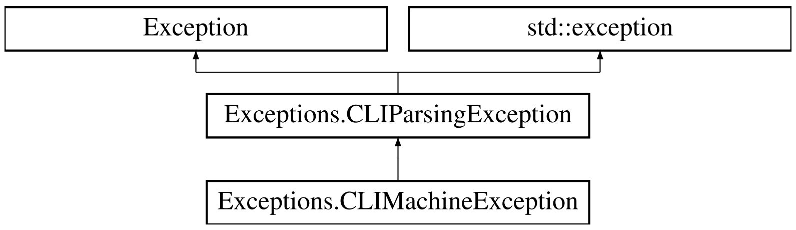
```
Exceptions.CLIInvalidArgumentException.__init__ (
              self,
             str message )
```

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.CLIMachineException Class Reference

Inheritance diagram for Exceptions.CLIMachineException:


Exception
std::exception
Exceptions.CLIParsingException
Exceptions.CLIMachineException

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

6.20.1 Constructor & Destructor Documentation

6.20.1.1 __init__()

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

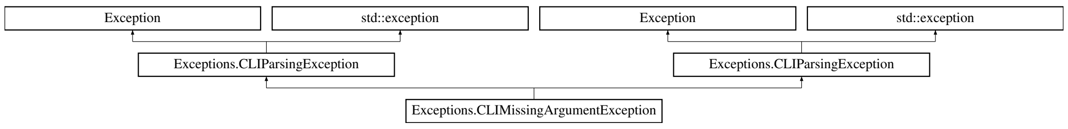
Reimplemented from Exceptions.CLIParsingException.

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

- ai/src/Exceptions/Exceptions.py

6.21 Exceptions.CLIMissingArgumentException Class Reference

Inheritance diagram for Exceptions.CLIMissingArgumentException:


Exception
std::exception
Exception
std::exception
Exceptions.CLIParsingException
Exceptions.CLIParsingException
Exceptions.CLIMissingArgumentException

Public Member Functions

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

Public Member Functions inherited from Exceptions.CLIParsingException

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

6.21.1 Constructor & Destructor Documentation

6.21.1.1 __init__()

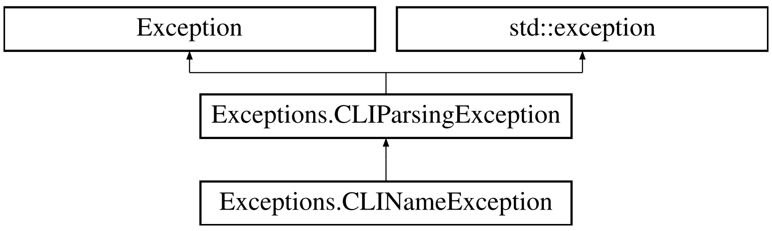
```
Exceptions.CLIMissingArgumentException.__init__ (
                 self,
                str message )
```

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__()

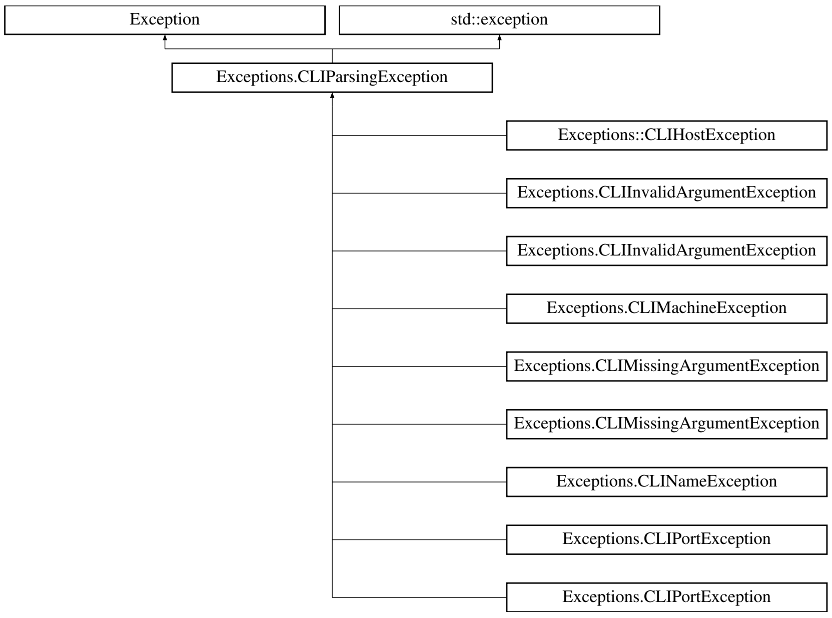
```
Exceptions.CLINameException.__init__ (
                 self,
                str message )
```

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:


Exception
std::exception
Exceptions.CLIParsingException
Exceptions::CLIHostException
Exceptions.CLIInvalidArgumentException
Exceptions.CLIInvalidArgumentException
Exceptions.CLIMachineException
Exceptions.CLIMissingArgumentException
Exceptions.CLIMissingArgumentException
Exceptions.CLINameException
Exceptions.CLIPortException
Exceptions.CLIPortException

Public Member Functions

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

Private Attributes

- std::string **_message**

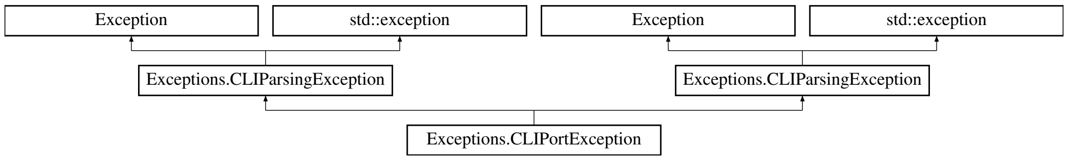
6.23.1 Detailed Description

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:


Exception
std::exception
Exception
std::exception
Exceptions.CLIParsingException
Exceptions.CLIParsingException
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 )
```

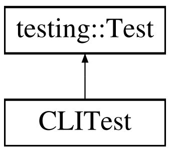
Reimplemented from Exceptions.CLIParsingException.

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

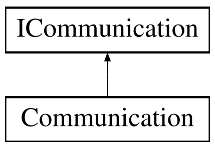
- char const ∗ **flag**
- bool(∗ **checker**)(const char ∗, const char ∗, params_t ∗)

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**

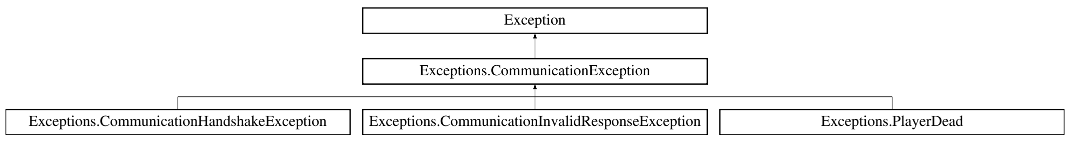
- **requestQueue**

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:


Exception
Exceptions.CommunicationException
Exceptions.CommunicationHandshakeException
Exceptions.CommunicationInvalidResponseException
Exceptions.PlayerDead

Public Member Functions

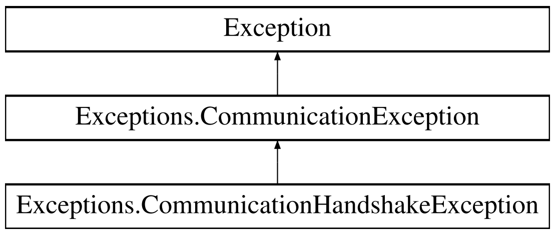
- **__init__** (self, str message)

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:


Exception
Exceptions.CommunicationException
Exceptions.CommunicationHandshakeException

Public Member Functions

- __init__ (self, str message)

6.33.1 Constructor & Destructor Documentation

6.33.1.1 __init__()

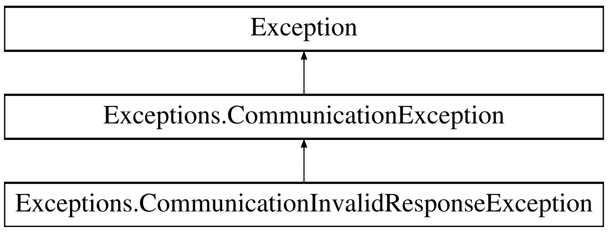
```
Exceptions.CommunicationHandshakeException.__init__ (
              self,
             str message )
```

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:


Exception
Exceptions.CommunicationException
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 )
```

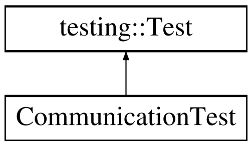
Reimplemented from Exceptions.CommunicationException.

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

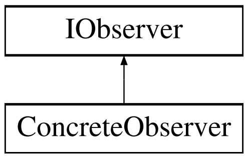
- static const int **TEST_PORT** = 9876

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

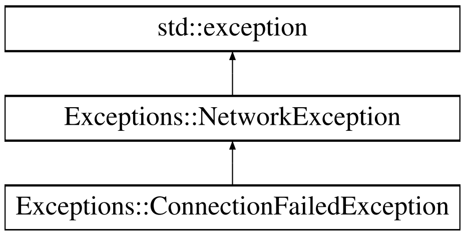
- int **port**
- std::string **hostname**

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:


std::exception
Exceptions::NetworkException
Exceptions::ConnectionFailedException

Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

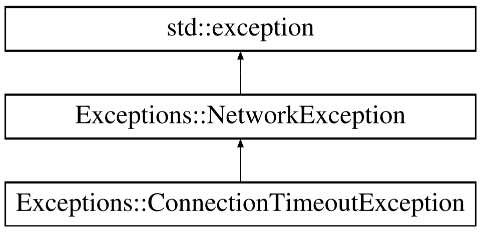
- **NetworkException** (const std::string &message)
- const char ∗ **what** () const noexcept override

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:


std::exception
Exceptions::NetworkException
Exceptions::ConnectionTimeoutException

Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

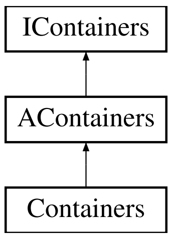
- **NetworkException** (const std::string &message)
- const char ∗ **what** () const noexcept override

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< IUIElement > element)
- std::shared_ptr< IUIElement > **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 minValue, float maxValue, 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 normal↩ Color, 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 y↩ Percent, float widthPercent, float heightPercent, const std::string &imagePath, std::function< void()> callback)
- std::shared_ptr< ImageButton > **addImageButtonPercent** (const std::string &id, float xPercent, float y↩ Percent, 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< IUIElement > > **_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]
```

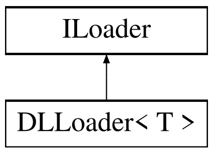
Implements IContainers.

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**

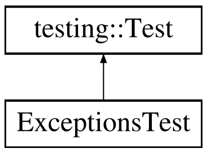
- bool **isHatched**
- struct egg_s ∗ **next**

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

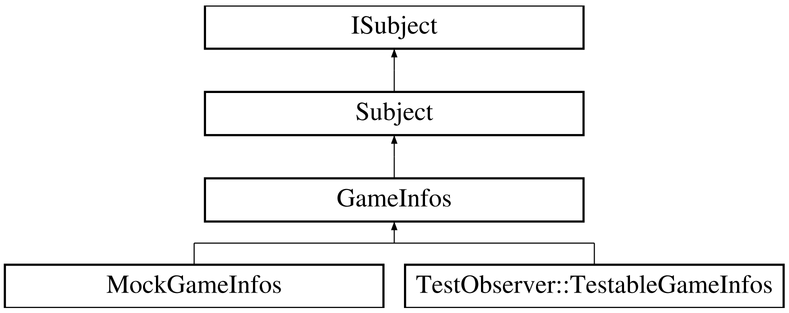
- team_t ∗ **teams**
- map_t ∗ **map**

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

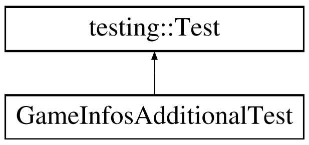
- std::vector< std::weak_ptr< IObserver > > **_observers**

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

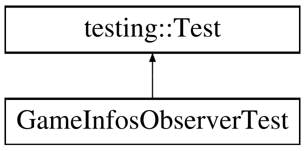
- std::unique_ptr< GameInfos > **gameInfos**
- std::shared_ptr< testing::NiceMock< MockCommunication > > **mockCommunication**
- std::shared_ptr< testing::NiceMock< MockAudio > > **mockAudio**
- std::shared_ptr< testing::NiceMock< MockObserver > > **mockObserver**

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

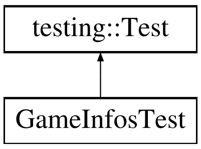
- std::unique_ptr< TestObserver::TestableGameInfos > **gameInfos**
- std::shared_ptr< testing::NiceMock< MockCommunication > > **mockCommunication**
- std::shared_ptr< testing::NiceMock< MockAudio > > **mockAudio**
- std::shared_ptr< TestObserver > **mockObserver**

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

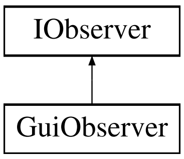
- std::string **_currentLibLoaded**
- bool **_isRunning**
- DLLoader< std::shared_ptr< IDisplay > > **_dlLoader**
- std::shared_ptr< IDisplay > **_display**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::unique_ptr< Map > **_map**
- std::unique_ptr< HUD > **_hud**
- std::shared_ptr< IAudio > **_audio**
- 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]
```

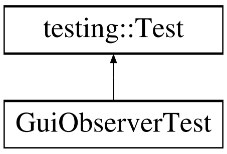
Implements IObserver.

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

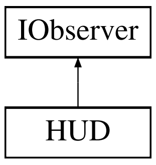
- std::shared_ptr< IDisplay > **_display**
- std::shared_ptr< IAudio > **_audio**
- std::shared_ptr< Containers > **_helpContainer**
- bool **_visible**

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 bottom←↩ Height, 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 bottom←↩ Height, 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]
```

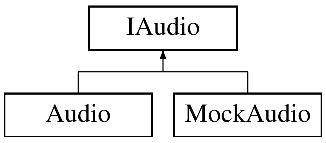
Implements IObserver.

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

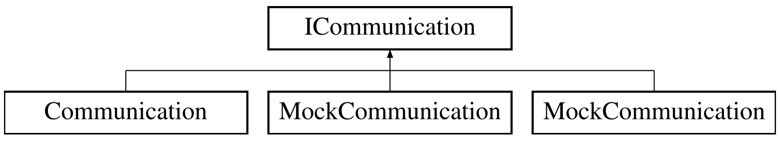
- virtual float **getSFXVolumeLevel** ()=0
- virtual float **getMusicVolumeLevel** ()=0
- virtual void **setSFXVolumeLevel** (float)=0
- virtual void **setMusicVolumeLevel** (float)=0
- virtual bool **loadSound** (const std::string &id, const std::string &filepath)=0
- virtual void **playMainTheme** (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

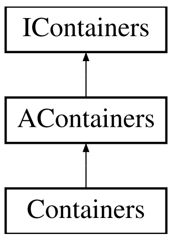
- virtual void **sendMessage** (const std::string &message)=0
- virtual bool **hasMessages** () const =0
- virtual std::string **popMessage** ()=0
- virtual bool **isConnected** () const =0
- virtual void **disconnect** ()=0

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

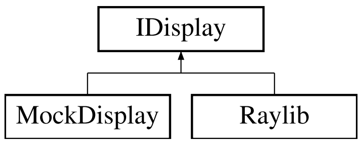
- virtual void **draw** ()=0
- virtual void **update** ()=0
- virtual void **setPosition** (float x, float y)=0
- virtual void **setSize** (float width, float height)=0
- virtual FloatRect **getBounds** () const =0
- virtual bool **contains** (float x, float y) const =0
- virtual void **setVisible**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

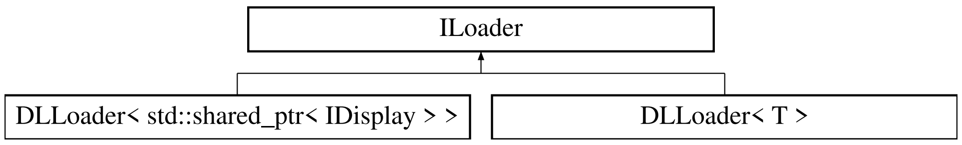
- virtual void **drawRectangleRec** (FloatRect rec, Color32 color)=0
- virtual bool **loadTexture** (const std::string &id, const std::string &filepath)=0
- virtual bool **loadFont** (const std::string &id, const std::string &filepath)=0
- virtual void **drawTexture** (const std::string &id, float x, 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

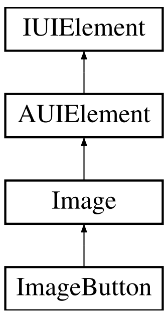
- virtual void ∗ **Open** (const char ∗path, int flag)=0
- virtual void ∗ **Symbol** (const char ∗symbolName)=0
- virtual int **Close** ()=0
- virtual const char ∗ **Error** ()=0
- virtual void ∗ **getHandler** () const =0

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

- gui/src/DLLoader/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 &image↩ Path)
- 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 IUIElement.

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]
```

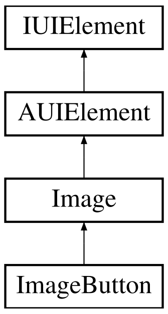
Implements IUIElement.

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 &image↩ Path)
- 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

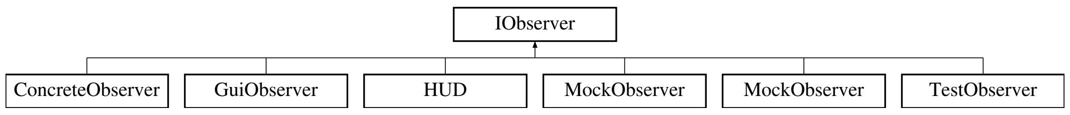
- int **nbFood**
- int **nbLinemate**
- int **nbDeraumere**
- int **nbSibur**
- int **nbMendiane**
- int **nbPhiras**
- int **nbThystame**

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

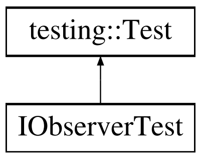
- virtual void **update** ()=0
- virtual void **onGameEvent** (GameEventType eventType, const std::string &teamName)

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

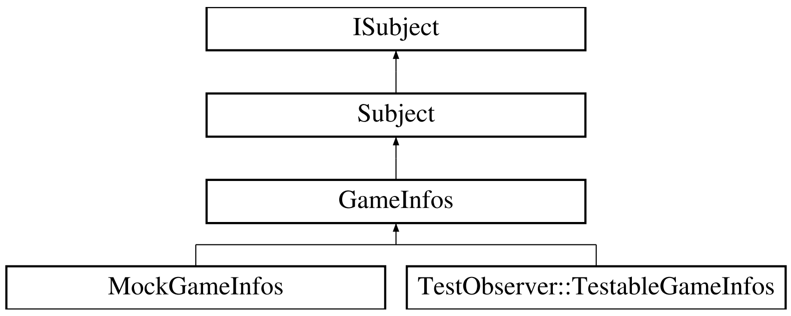
- std::shared_ptr< ConcreteObserver > **observer**

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

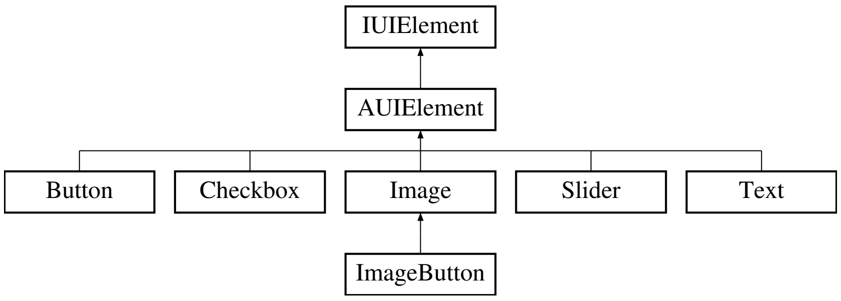
- char ∗ **name**
- void(∗ **add_func**)(inventory_t ∗)

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

- server/include/zappy.h

6.75 IUIElement Class Reference

Inheritance diagram for IUIElement:



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/IUIElement.hpp

6.76 Logger.Logger Class Reference

Public Member Functions

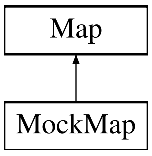
- None **error** (self, str message)
- None **info** (self, str message)
- None **help** (self, str message)
- None **debug** (self, str message)
- None **success** (self, str message)
- None **display** (self, str message)

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

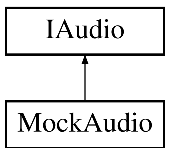
- int **width**
- int **height**
- egg_t ∗ **currentEggs**
- inventory_t ∗∗ **tiles**

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

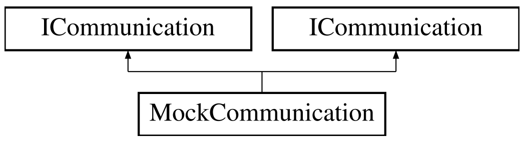
- virtual float **getSFXVolumeLevel** ()=0
- virtual float **getMusicVolumeLevel** ()=0
- virtual void **setSFXVolumeLevel** (float)=0
- virtual void **setMusicVolumeLevel** (float)=0
- virtual bool **loadSound** (const std::string &id, const std::string &filepath)=0
- virtual void **playMainTheme** (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

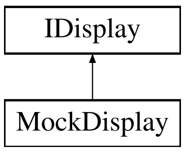
- virtual void **sendMessage** (const std::string &message)=0
- virtual bool **hasMessages** () const =0
- virtual std::string **popMessage** ()=0
- virtual bool **isConnected** () const =0
- virtual void **disconnect** ()=0

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, vector3SubtractFromCamera,(Vector3f target),(override))
- **MOCK_METHOD** (Vector3f, vector3Normalize,(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 end↩Radius, 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

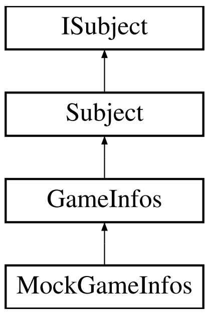
- virtual void **drawCylinderEx** (Vector3f startPos, Vector3f endPos, float startRadius, float endRadius, int sides, Color32 color)=0
- virtual void **drawPlane** (Vector3f position, Vector2f size, Color32 color)=0
- virtual void **drawLine3D** (Vector3f startPos, Vector3f endPos, Color32 color)=0
- 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

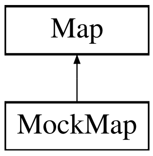
- **MOCK_METHOD** (void, refresh,())
- **MOCK_METHOD** (void, handleVictory,(const std::string &teamName))

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

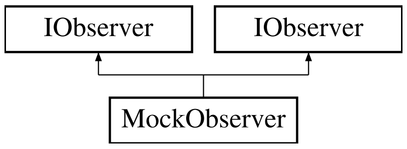
- **Map** (std::shared_ptr< GameInfos > gameInfos, std::shared_ptr< IDisplay > display)
- void **draw** (bool performanceMode=false)
- void **drawBroadcastingPlayers** ()
- void **drawIncantations** ()
- void **drawTile** (int x, int y, const zappy::structs::Tile &tile)
- void **drawPerformanceTile** (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

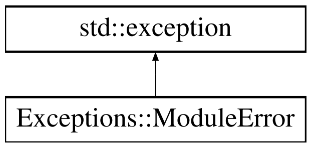
- Model **model**
- unsigned int **animationCount**
- Vector3 **center**

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 **handleMszMessage** (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

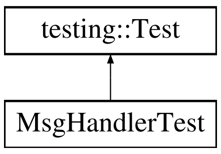
- std::thread **_thread**
- std::atomic< bool > **_running**
- std::mutex **_mutex**
- std::condition_variable **_condition**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< ICommunication > **_communication**
- std::mutex **_gameInfosMutex**
- std::map< std::string, std::function< bool(const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

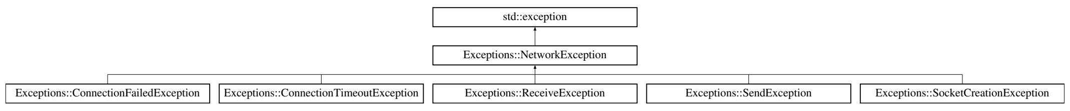
- int **fd**
- buffer_t ∗ **buffer**

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:


std::exception
Exceptions::NetworkException
Exceptions::ConnectionFailedException
Exceptions::ConnectionTimeoutException
Exceptions::ReceiveException
Exceptions::SendException
Exceptions::SocketCreationException

Public Member Functions

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

Private Attributes

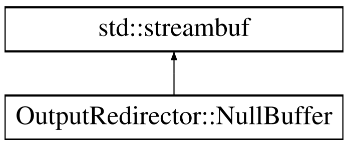
- std::string **_message**

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

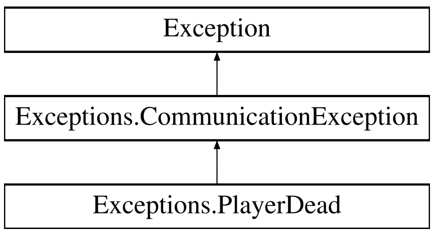
- int **id**
- network_t ∗ **network**
- int **level**
- int **posX**
- int **posY**
- direction_t **direction**
- inventory_t ∗ **inventory**
- char ∗ **team**
- action_queue_t ∗ **pending_actions**
- time_t **last_action_time**
- bool **is_busy**
- int **remaining_cooldown**
- char ∗ **current_action**
- 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

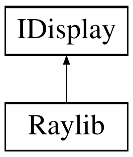
- bool **hit**
- float **distance**
- Vector3f **point**
- Vector3f **normal**

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

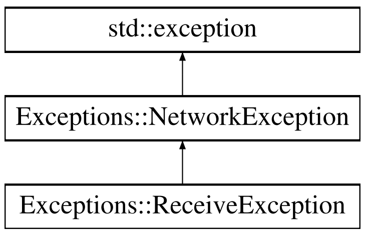
- static constexpr float **FONT_SCALE_FACTOR** = 4.0f
- static constexpr float **FONT_RENDER_SCALE** = 0.25f
- static constexpr float **FONT_SPACING_RATIO** = 0.1f

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

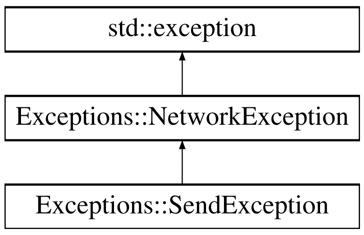
- float **xPercent**
- float **yPercent**
- float **widthPercent**
- float **heightPercent**

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**

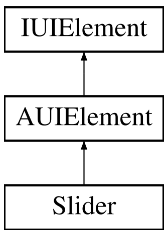
- float **_cameraMovingSpeed**
- float **_cameraRotaSpeed**
- float **_cameraZoomSpeed**
- std::shared_ptr< Containers > **_settingsContainer**
- bool **_visible**

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 minValue, float max←
  Value, float initialValue, const std::string &text, std::function< void(float)> onValueChanged)
- void draw () override
- void update () override
- bool **isDragging** () const
- void **setValue** (float value)
- float **getValue** () const
- void **setMinValue** (float minValue)
- void **setMaxValue** (float maxValue)
- 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 AUIElement

- 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 IUIElement.

6.113.1.2 setSize()

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

Reimplemented from AUIElement.

6.113.1.3 update()

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

Implements IUIElement.

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

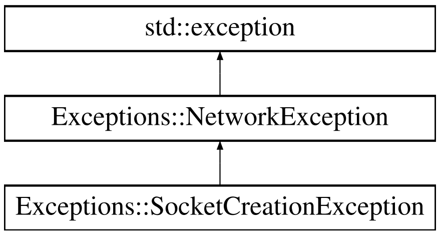
- **_host**
- **_port**
- **_address**
- **_socket**

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:


std::exception
Exceptions::NetworkException
Exceptions::SocketCreationException

Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

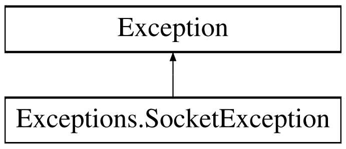
- **NetworkException** (const std::string &message)
- const char ∗ **what** () const noexcept override

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

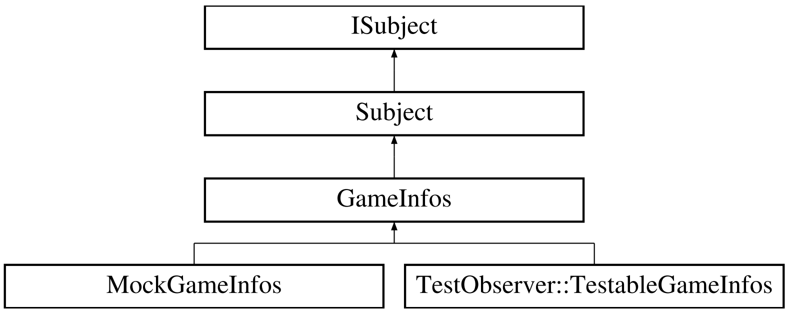
- **__init__** (self, str message)

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]
```

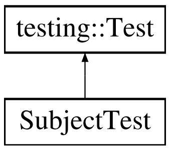
Implements ISubject.

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

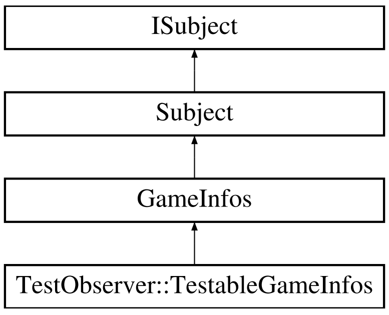
- char ∗ **name**
- int **nbPlayers**
- int **nbPlayerAlive**
- player_t ∗ **players**
- struct team_s ∗ **next**

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

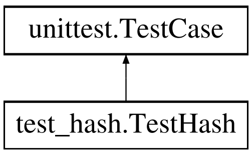
6.123 test_com.TestCommunication Class Reference

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

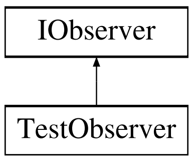
- **hash_obj**

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 )
```

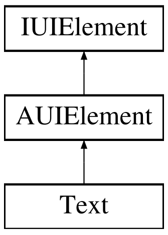
```
Test sending unicode messages
```

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

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Audio
*/

#ifndef AUDIO_HPP_
#define AUDIO_HPP_

#include <string>
#include <map>
#include <memory>
#include <vector>
#include <SFML/Audio.hpp>
#include "IAudio.hpp"

class Audio : public IAudio {
    private:
        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;

    public:
        Audio();
        ~Audio();

        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);
};

#endif /* !AUDIO_HPP_ */
```

7.2 IAudio.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** IAudio
*/
```

```

#ifndef IAUDIO_HPP_
#define IAUDIO_HPP_

#include <string>

class IAudio {
    public:
        virtual ~IAudio() = default;

        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;
};

#endif /* !IAUDIO_HPP_ */
```

7.3 CLI.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** CLI
*/

#ifndef CLI_HPP_
#define CLI_HPP_

#include <string>
#include "../Utils/Constants.hpp"

class CLI {
    public:
        CLI(int ac, const char *const *av);
        ~CLI();

        zappy::structs::Config parseArguments(int ac, const char *const *av) const;

    private:
        int _ac;
        const char *const *_av;

        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;
};

#endif /* !CLI_HPP_ */
```

7.4 Client.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Client
*/

#ifndef CLIENT_HPP_
#define CLIENT_HPP_

#include <memory>
#include <filesystem>
#include <string>
#include "../Utils/Constants.hpp"
#include "../Communication/ICommunication.hpp"
```

```
#include "../Game/GameInfos.hpp"
#include "../Graphic/GUI.hpp"
#include "MsgHandler.hpp"
#include "../Observer/GuiObserver.hpp"
#include "../Observer/IObserver.hpp"

class Client {
    public:
        Client(int ac, const char *const *av);
        ~Client();

        void tryToCreateGuiWithSharedLibInFolder(const std::string &libPath);

    private:
        zappy::structs::Config _config;
        void initialize(int ac, const char * const *av);

        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;
};

#endif /* !CLIENT_HPP_ */
```

7.5 MsgHandler.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** MsgHandler
*/

#ifndef MSGHANDLER_HPP_
#define MSGHANDLER_HPP_

#include <memory>
#include <map>
#include <functional>
#include <thread>
#include <mutex>
#include <atomic>
#include <queue>
#include <condition_variable>
#include <string>

#include "../Game/GameInfos.hpp"
#include "../Communication/ICommunication.hpp"
#include "../Utils/Constants.hpp"

class MsgHandler {
    public:
        MsgHandler(std::shared_ptr<GameInfos> gameInfos,
            std::shared_ptr<ICommunication> communication);
        ~MsgHandler();

        void start();
        void stop();

        void handleMessage(const std::string& message);

    protected:
        void messageLoop();

        bool handleWelcomeMessage(const std::string& message);
        bool handleMszMessage(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:
        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;
};

#endif /* !MSGHANDLER_HPP_ */
```

7.6 Communication.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Communication
*/

#ifndef COMMUNICATION_HPP_
#define COMMUNICATION_HPP_

#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <fcntl.h>
#include <poll.h>
#include <netdb.h>
#include <thread>
#include <mutex>
#include <atomic>
#include <condition_variable>
#include <queue>
#include <string>
#include <vector>

#include "../Utils/Constants.hpp"
#include "../Exceptions/Exceptions.hpp"
#include "ICommunication.hpp"

class Communication : public ICommunication {
    public:
        explicit Communication(zappy::structs::Config config);
        ~Communication();

        void sendMessage(const std::string &message) override;
        bool hasMessages() const override;
        std::string popMessage() override;
        bool isConnected() const override;
        void disconnect() override;

    private:
        void setupConnection();
        void createSocket();
        void connectToServer();
        void setupNonBlocking();

        void startCommunicationThread();
        void communicationLoop();
        bool handlePoll();
        void processWrite();
        void processRead();

        void parseReceivedData();

        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 const int BUFFER_SIZE = 4096;
        static const int POLL_TIMEOUT = 100;
        static const char MESSAGE_DELIMITER = '\n';
};

#endif /* !COMMUNICATION_HPP_ */
```

7.7 ICommunication.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** ICommunication
*/

#ifndef ICOMMUNICATION_HPP_
#define ICOMMUNICATION_HPP_

#include <string>

class ICommunication {
    public:
        virtual ~ICommunication() = default;

        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;
};

#endif /* !ICOMMUNICATION_HPP_ */
```

7.8 DLLoader.hpp

```
/*
** EPITECH PROJECT, 2025
** zappy
** File description:
** DLLoader
*/

#ifndef DLLOADER_HPP_
#define DLLOADER_HPP_

#include <dlfcn.h>
#include <iostream>
#include <ostream>
#include <memory>
#include "ILoader.hpp"

template <typename T>

class DLLoader : public ILoader {
    private:
        void *_handler = nullptr;

    public:
        ~DLLoader() = default;

        void *getHandler() const override {
            return _handler;
        };
        void *Open(const char *path, int flag = RTLD_LAZY) override {
            _handler = dlopen(path, flag);
            return _handler;
        };
        void *Symbol(const char *symbolName) override {
            void *symbol = dlsym(_handler, symbolName);
            const char *error = dlerror();
            if (error) {
                std::cerr « "dlerror: " « error « std::endl;
                return nullptr;
            }
            return symbol;
        };
        T getSymbol(const char *symbolName) {
            return reinterpret_cast<T>(dlsym(_handler, symbolName));
        };
        int Close() override{
            if (_handler == nullptr)
                return -1;
            return dlclose(_handler);
        };
        const char *Error() override {
            return dlerror();
        };
};

#endif /* !DLLOADER_HPP_ */
```

7.9 ILoader.hpp

```
/*
** EPITECH PROJECT, 2025
** zappy
** File description:
** ILoader
*/

#ifndef ILoader_HPP_
#define ILoader_HPP_


class ILoader {
    public:
        ~ILoader() = default;

        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;

    protected:
    private:
};

#endif /* !ILoader_HPP_ */
```

7.10 LoaderType.hpp

```
/*
** EPITECH PROJECT, 2025
** zappy
** File description:
** LoaderType
*/

#ifndef LOADERTYPE_HPP_
#define LOADERTYPE_HPP_

enum ModuleType_t{
    DISPLAY_MODULE,
    NONE
};

#endif /* !LOADERTYPE_HPP_ */
```

7.11 Exceptions.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Exceptions
*/

#ifndef EXCEPTIONS_HPP_
#define EXCEPTIONS_HPP_

#include <exception>
```

```cpp
#include <string>
#include "../Utils/Constants.hpp"

namespace Exceptions {

    // CLI Exceptions
    class CLIParsingException : public std::exception {
        public:
            explicit CLIParsingException(const std::string &message)
                : _message(std::string(colors::T_RED) +
                           "CLI Parsing Error: " + message +
                           colors::RESET) {}

            const char *what() const noexcept override {
                return _message.c_str();
            }

        private:
            std::string _message;
    };

    class CLIPortException : public CLIParsingException {
        public:
            explicit CLIPortException(const std::string &message)
                : CLIParsingException(std::string(colors::T_CYAN) +
                                      "Port Error: " + message +
                                      colors::RESET) {}
    };

    class CLIHostException : public CLIParsingException {
        public:
            explicit CLIHostException(const std::string &message)
                : CLIParsingException(std::string(colors::T_CYAN) +
                                      "Hostname Error: " + message +
                                      colors::RESET) {}
    };

    class CLIMissingArgumentException : public CLIParsingException {
        public:
            explicit CLIMissingArgumentException(const std::string &message)
                : CLIParsingException(std::string(colors::T_CYAN) +
                                      "Missing Argument: " + message +
                                      colors::RESET) {}
    };

    class CLIInvalidArgumentException : public CLIParsingException {
        public:
            explicit CLIInvalidArgumentException(const std::string &message)
                : CLIParsingException(std::string(colors::T_CYAN) +
                                      "Invalid Argument: " + message +
                                      colors::RESET) {}
    };

    class NetworkException : public std::exception {
        public:
            explicit NetworkException(const std::string &message)
                : _message(std::string(colors::T_RED) +
                           "Network Error: " + message +
                           colors::RESET) {}

            const char *what() const noexcept override {
                return _message.c_str();
            }

        private:
            std::string _message;
    };

    class ConnectionFailedException : public NetworkException {
        public:
            explicit ConnectionFailedException(const std::string &message)
                : NetworkException(std::string(colors::T_CYAN) +
                                   "Connection Failed: " + message +
                                   colors::RESET) {}
    };

    class SocketCreationException : public NetworkException {
        public:
            explicit SocketCreationException(const std::string &message)
                : NetworkException(std::string(colors::T_CYAN) +
                                   "Socket Creation Failed: " + message +
                                   colors::RESET) {}
    };

    class ConnectionTimeoutException : public NetworkException {
        public:
            explicit ConnectionTimeoutException(const std::string &message)
```

```
                : NetworkException(std::string(colors::T_CYAN) +
                                   "Connection Timeout: " + message +
                                   colors::RESET) {}
    };

    class SendException : public NetworkException {
        public:
            explicit SendException(const std::string &message)
                : NetworkException(std::string(colors::T_CYAN) +
                                   "Send Error: " + message +
                                   colors::RESET) {}
    };

    class ReceiveException : public NetworkException {
        public:
            explicit ReceiveException(const std::string &message)
                : NetworkException(std::string(colors::T_CYAN) +
                                   "Receive Error: " + message +
                                   colors::RESET) {}
    };

    class ModuleError : public std::exception {
        private:
            std::string _message = "";
        public:
            explicit ModuleError(const std::string &msg) : _message(msg) {};
            const char *what() const noexcept override {
                return this->_message.c_str();
            }
    };
}

#endif /* !EXCEPTIONS_HPP_ */
```

7.12 GameInfos.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** GameInfos
*/

#ifndef GAMEINFOS_HPP_
#define GAMEINFOS_HPP_

#include <utility>
#include <vector>
#include <memory>
#include <mutex>
#include <string>
#include <chrono>
#include <unordered_map>

#include "../Utils/Constants.hpp"
#include "../Communication/ICommunication.hpp"
#include "../Observer/Subject.hpp"
#include "../Audio/IAudio.hpp"

class GameInfos : public Subject {
    public:
        explicit GameInfos(std::shared_ptr<ICommunication> communication);
        ~GameInfos();

        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);

    private:
        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;

        mutable std::mutex _dataMutex;

        std::shared_ptr<ICommunication> _communication;
        std::shared_ptr<IAudio> _audio;
        zappy::gui::CameraMode _currentCameraMode;
        int _currentPlayerFocus;

        void notifyStateChange();
};

#endif /* !GAMEINFOS_HPP_ */
```

7.13 CameraManager.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** CameraManager
*/

#ifndef CAMERA_MANAGER_HPP_
```

```cpp
#define CAMERA_MANAGER_HPP_

#include <memory>
#include "../../Utils/Constants.hpp"
#include "../../Game/GameInfos.hpp"
#include "../Map.hpp"

class CameraManager {
    public:
        explicit CameraManager(std::shared_ptr<IDisplay> display);
        ~CameraManager();

        void updateCamera(zappy::gui::CameraMode mode);
        void updateCameraFreeMode();
        void updateCameraTargetMode();
        void updateCameraPlayerMode();

        void setMapCenter(const Vector3f &center);
        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:
        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;

        void handlePlayerCameraMouseInput();
};

#endif /* !CAMERA_MANAGER_HPP_ */
```

7.14 GUI.hpp

```cpp
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** GUI
*/

#ifndef GUI_HPP_
#define GUI_HPP_

#include <memory>
#include <string>
#include <utility>
#include "../Game/GameInfos.hpp"
#include "Map.hpp"
#include "HUD/HUD.hpp"
#include "../Audio/IAudio.hpp"
#include "../Utils/Constants.hpp"
#include "Camera/CameraManager.hpp"
#include "../IDisplay.hpp"
```

```
#include "../DLLoader/DLLoader.hpp"

class GUI {
    public:
        GUI(std::shared_ptr<GameInfos> gameInfos, const std::string &libPath);
        ~GUI();

        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:
        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;

        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;
};

#endif /* !GUI_HPP_ */
```

7.15 Button.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Button
*/

#pragma once

#include <string>
#include <functional>
#include <memory>

#include "../UIElement/AUIElement.hpp"
#include "../../../Audio/IAudio.hpp"
#include "../../../IDisplay.hpp"
```

```

class Button : public AUIElement {
    public:
        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
        );

        ~Button() override = default;

        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;

    private:
        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;
};
```

7.16 Checkbox.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Checkbox
*/

#pragma once

#include <string>
#include <functional>
#include <memory>

#include "../UIElement/AUIElement.hpp"
#include "../../../Audio/IAudio.hpp"
#include "../../../IDisplay.hpp"

class Checkbox : public AUIElement {
    public:
        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
        );

        ~Checkbox() override = default;

        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;

    private:
        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;
};
```

7.17 AContainers.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** AContainers
*/

#pragma once

#include <string>
#include <vector>
#include <memory>

#include "IContainers.hpp"

struct RelativePosition {
    float xPercent;
    float yPercent;
    float widthPercent;
    float heightPercent;
};

class AContainers : public IContainers {
    public:
        AContainers(std::shared_ptr<IDisplay> display, float x, float y, float width,
            float height);

        virtual ~AContainers() = default;

        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;

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

7.18 Containers.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Containers
*/

#pragma once

#include <vector>
#include <functional>
#include <unordered_map>
#include <memory>
#include <string>

#include "AContainers.hpp"
#include "../UIElement/IUIElement.hpp"
#include "../Button/Button.hpp"
#include "../Text/Text.hpp"
#include "../Slider/Slider.hpp"
#include "../Image/Image.hpp"
#include "../ImageButton/ImageButton.hpp"
#include "../Checkbox/Checkbox.hpp"
#include "../../../Audio/IAudio.hpp"
#include "../../../IDisplay.hpp"

class Containers : public AContainers {
    public:
        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});

        ~Containers() override = default;

        void draw() override;

        void update() override;

        void setBackgroundColor(Color32 color);

        bool addElement(const std::string& id, std::shared_ptr<IUIElement> element);

        std::shared_ptr<IUIElement> 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 fontSize = 20.0f,
            Color32 color = CBLACK
        );

        std::shared_ptr<Slider> addSlider(
            const std::string& id,
            float x, float y,
            float width, float height,
            float minValue, 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 minValue, float maxValue,
            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;

    private:
        std::shared_ptr<IAudio> _audio;
        std::unordered_map<std::string, std::shared_ptr<IUIElement» _elements;
};
```

7.19 IContainers.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** IContainers
*/

#pragma once

#include <string>
#include <memory>
#include <vector>
#include "../../../IDisplay.hpp"

class IContainers {
    public:
        virtual ~IContainers() = default;

        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;
};
```

7.20 Help.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Help
*/

#pragma once

#include <memory>
#include <string>
#include "../Containers/Containers.hpp"
#include "../../../IDisplay.hpp"
#include "../../../Audio/IAudio.hpp"

class Help {
    public:
        Help(std::shared_ptr<IDisplay> display, std::shared_ptr<IAudio> audio);

        ~Help() = default;

        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:
        void initHelpContainer();

        std::shared_ptr<IDisplay> _display;
        std::shared_ptr<IAudio> _audio;
        std::shared_ptr<Containers> _helpContainer;
        bool _visible;
};
```

7.21 HUD.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** HUD
*/

#pragma once

#include <vector>
#include <unordered_map>
#include <memory>
#include <string>
#include <utility>
#include <functional>
#include <chrono>
#include "Containers/Containers.hpp"
#include "../../Game/GameInfos.hpp"
#include "../../Audio/IAudio.hpp"
#include "../../Utils/Constants.hpp"
#include "Help/Help.hpp"
#include "Settings/Settings.hpp"
#include "../../IDisplay.hpp"
#include "../../Observer/IObserver.hpp"
#include "Graphic/Camera/CameraManager.hpp"

class HUD : public IObserver {
    public:
        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);

        ~HUD();

        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:
        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);

        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;
};
```

7.22 Image.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Image
*/

#pragma once
```

```

#include <string>
#include <memory>

#include "../UIElement/AUIElement.hpp"
#include "../../../IDisplay.hpp"

class Image : public AUIElement {
    public:
        Image(
            std::shared_ptr<IDisplay> display,
            float x, float y,
            float width, float height,
            const std::string& imagePath
        );

        ~Image() override = default;

        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;

    private:
        std::string _imagePath;
        Color32 _tint;
        bool _maintainAspectRatio;
        bool _imageLoaded;

        void loadImage();
};
```

7.23 ImageButton.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** ImageButton
*/

#pragma once

#include <functional>
#include <string>
#include <memory>
#include "../Image/Image.hpp"
#include "../../../Audio/IAudio.hpp"

class ImageButton : public Image {
    public:
        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
        );

        ~ImageButton() override = default;

        void update() override;

        void setCallback(std::function<void()> callback);

        std::function<void()> getCallback() const;

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

7.24 Settings.hpp

```
/*
** EPITECH PROJECT, 2025
** zappy
** File description:
** Settings
*/

#ifndef SETTINGS_HPP_
#define SETTINGS_HPP_
#include <memory>
#include "../Containers/Containers.hpp"
#include "../../../IDisplay.hpp"
#include  "../../../Audio/IAudio.hpp"
#include "Graphic/Camera/CameraManager.hpp"

class Settings {
    private:
        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;

    public:
        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
        );
        ~Settings();
};

#endif /* !SETTINGS_HPP_ */
```

7.25 Slider.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Slider
*/

#ifndef SLIDER_HPP_
#define SLIDER_HPP_

#include <string>
#include <functional>
#include <memory>

#include "../../../IDisplay.hpp"
#include "../UIElement/AUIElement.hpp"

class Slider : public AUIElement {
    public:
        Slider(
            std::shared_ptr<IDisplay> raylib,
            float x, float y,
            float width, float height,
            float minValue, float maxValue,
            float initialValue,
            const std::string& text,
            std::function<void(float)> onValueChanged
        );

        ~Slider() override = default;

        void draw() override;
        void update() override;
        bool isDragging() const;

        void setValue(float value);
        float getValue() const;
        void setMinValue(float minValue);
        void setMaxValue(float maxValue);
        float getMinValue() const;
        float getMaxValue() const;
        void setText(const std::string& text);
        std::string getText() const;

        void setSize(float width, float height) override;

    private:
        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;

        void updateValueFromMousePosition(float mouseX);
        float getHandlePosition() const;
        bool isMouseOverHandle(float mouseX, float mouseY) const;
        bool isMouseOverTrack(float mouseX, float mouseY) const;
};

#endif /* !SLIDER_HPP_ */
```

7.26 Text.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Text
*/

#pragma once

#include <memory>
#include <string>

#include "../UIElement/AUIElement.hpp"
#include "../../../IDisplay.hpp"

class Text : public AUIElement {
    public:
        Text(
            std::shared_ptr<IDisplay> raylib,
            float x, float y,
            const std::string& text,
            float fontSize = 20.0f,
            Color32 color = CBLACK
        );

        ~Text() override = default;

        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);

    private:
        std::string _text;
        float _fontSize;
        Color32 _color;
        std::shared_ptr<IDisplay> _display;
};
```

7.27 AUIElement.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** AUIElement
*/

#pragma once

#include <memory>
#include "IUIElement.hpp"

struct UIRelativePosition {
    float xPercent;
    float yPercent;
    float widthPercent;
    float heightPercent;
};

class AUIElement : public IUIElement {
    public:
        AUIElement(std::shared_ptr<IDisplay> display, float x, float y, float width,
            float height);

        virtual ~AUIElement() = default;

        // IUIElement implementation
        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;

    protected:
        std::shared_ptr<IDisplay> _display;
        FloatRect _bounds;
        UIRelativePosition _relativePos;
        bool _visible;
};
```

7.28 IUIElement.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** IUIElement
*/

#pragma once

#include "../../../IDisplay.hpp"
```

```
class IUIElement {
    public:
        virtual ~IUIElement() = default;

        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;
};
```

7.29 Map.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Map
*/

#ifndef MAP_HPP_
#define MAP_HPP_

#include <memory>
#include <unordered_map>
#include <vector>
#include <string>
#include <chrono>
#include "../Game/GameInfos.hpp"
#include "../IDisplay.hpp"

enum class DisplayPriority {
    TILE = 0,
    EGG = 1,
    PLAYER = 2,
    FOOD = 3,
    ROCK = 4,
};

struct PlayerRotationState {
    float currentRotation;
    float targetRotation;
    bool isRotating;
    std::chrono::steady_clock::time_point lastUpdateTime;

    PlayerRotationState() : currentRotation(0.0f), targetRotation(0.0f),
                    isRotating(false), lastUpdateTime(std::chrono::steady_clock::now()) {}
};

struct PlayerPositionState {
    Vector3f currentPosition;
    Vector3f targetPosition;
    bool isMoving;
    std::chrono::steady_clock::time_point lastUpdateTime;

    PlayerPositionState() : currentPosition({0.0f, 0.0f, 0.0f}),
                    targetPosition({0.0f, 0.0f, 0.0f}),
                    isMoving(false), lastUpdateTime(std::chrono::steady_clock::now()) {}
};

class Map {
    public:
        Map(std::shared_ptr<GameInfos> gameInfos, std::shared_ptr<IDisplay> display);
        ~Map();

        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:
        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;

        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;


        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);

        bool _performanceMode = false;
};

#endif /* !MAP_HPP_ */
```

7.30 IDisplay.hpp

```
/*
** EPITECH PROJECT, 2025
** zappy
** File description:
** IDisplay
*/

#ifndef IDISPLAY_HPP_
#define IDISPLAY_HPP_
#include <utility>
#include <string>
#include "Utils/InputType.hpp"

enum Key {
    TAB,
    ESC,
    UP,
    DOWN,
    RIGHT,
    LEFT,
    H,
    C,
    GM_PD_LEFT_SHOULDER,
    GM_PD_RIGHT_SHOULDER,
    GM_PD_LEFT_TRIGGER,
    GM_PD_RIGHT_TRIGGER,
    GM_PD_UP,
    GM_PD_DOWN,
    GM_PD_AXIS_RIGHT_X,
    GM_PD_AXIS_RIGHT_Y,
    GM_PD_H,
    MOUSE_LEFT,
    MOUSE_RIGHT,
};

typedef struct Vector3f {
    float x;
    float y;
    float z;

    bool operator==(const Vector3f& other) const {
        return x == other.x && y == other.y && z == other.z;
    }

    bool operator!=(const Vector3f& other) const {
        return !(*this == other);
    }
} Vector3f;

typedef struct Vector2f {
    float x;
    float y;
} Vector2f;

typedef struct Vector2i {
    int x;
    int y;
} Vector2i;

typedef struct Color32 {
    unsigned char r;
    unsigned char g;
    unsigned char b;
    unsigned char a;
} Color32;

typedef struct FloatRect {
    float x;
    float y;
    float width;
    float height;
} FloatRect;

typedef struct IntRect {
    int x;
    int y;
    int width;
    int height;
} IntRect;

typedef struct Ray3D {
    Vector3f position;
    Vector3f direction;
} Ray3D;

typedef struct RayCollision3D {
    bool hit;
    float distance;
    Vector3f point;
    Vector3f normal;
} RayCollision3D;

typedef struct BoundingBox3D {
    Vector3f min;
    Vector3f max;
} BoundingBox3D;

#define COLOR(r, g, b) Color32{ r, g, b, 255 }
#define CLIGHTGRAY COLOR(200, 200, 200)
#define CBLACK COLOR(0, 0, 0)
#define CRED COLOR(230, 41, 55)
#define CBROWN COLOR(127, 106, 79)
#define CBLUE COLOR(0, 121, 241)
#define CWHITE COLOR(255, 255, 255)

#define CRAYWHITE COLOR(245, 245, 245)
#define CPINK COLOR(255, 109, 194)
#define CGREEN COLOR(0, 228, 48)
#define CMAROON COLOR(190, 33, 55)
#define CPURPLE COLOR(200, 122, 255)
#define CORANGE COLOR(255, 161, 0)
#define CYELLOW COLOR(253, 249, 0)

class IDisplay {
    public:
        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;

        ~IDisplay() = default;
};

#endif /* !IDISPLAY_HPP_ */
```

7.31 GuiObserver.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** GuiObserver
*/

#ifndef GUIOBSERVER_HPP_
#define GUIOBSERVER_HPP_

#include <memory>
#include <string>
#include "IObserver.hpp"

class GUI;

class GuiObserver : public IObserver {
    public:
        GuiObserver(std::shared_ptr<GUI> gui);
        virtual ~GuiObserver() = default;

        void update() override;
        void onGameEvent(GameEventType eventType, const std::string& teamName) override;

    private:
        std::weak_ptr<GUI> _gui;
};

#endif /* !GUIOBSERVER_HPP_ */
```

7.32 IObserver.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** IObserver
*/

#ifndef IOBSERVER_HPP_
#define IOBSERVER_HPP_

#include <string>

enum class GameEventType {
    STATE_CHANGED,
    TEAM_WIN,
    TEAM_DEFEAT
};

class IObserver {
    public:
        virtual ~IObserver() = default;
        virtual void update() = 0;
        virtual void onGameEvent(GameEventType eventType, const std::string& teamName) {
            (void)eventType;
            (void)teamName;
        }
};

#endif /* !IOBSERVER_HPP_ */
```

7.33 ISubject.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** ISubject
*/

#ifndef ISUBJECT_HPP_
#define ISUBJECT_HPP_

#include <vector>
#include <memory>
#include <string>
#include "IObserver.hpp"

class ISubject {
    public:
        virtual ~ISubject() = default;
        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:
        std::vector<std::weak_ptr<IObserver» _observers;
};

#endif /* !ISUBJECT_HPP_ */
```

7.34 Subject.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Subject
*/

#include <algorithm>
#include <memory>
#include <vector>
```

```cpp
#include <string>

#include "ISubject.hpp"

#ifndef SUBJECT_HPP_
#define SUBJECT_HPP_

class Subject : public ISubject {
    public:
        virtual ~Subject() = default;

        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:
        std::vector<std::weak_ptr<IObserver» _observers;
};

#endif /* !SUBJECT_HPP_ */
```

7.35 Raylib.hpp

```cpp
/*
** EPITECH PROJECT, 2025
** zappy
** File description:
** Raylib
*/

#ifndef RAYLIB_HPP_
#define RAYLIB_HPP_
#include <memory>
#include <string>
#include "../IDisplay.hpp"
#include "RaylibEnc/RayLibEnc.hpp"

class Raylib : public IDisplay {
    private:
        std::unique_ptr<RayLibEnc> _raylib;

    public:
        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;

        Raylib();
        ~Raylib() = default;
};

#endif /* !RAYLIB_HPP_ */
```

7.36 RayLibEnc.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** RayLibEnc
*/

#ifndef RAYLIBENC_HPP_
#define RAYLIBENC_HPP_

#include <string>
#include <map>
#include <memory>
#include "raylib.h"
#include "../../Utils/InputType.hpp"

class RayLibEnc {
    public:
        RayLibEnc();
        ~RayLibEnc();

        // Window management methods
        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;

        // Collision methods
        bool checkCollisionPointRec(Vector2 point, Rectangle rec) const;

        // Ray and collision methods for 3D
        Ray getMouseRay(Vector2 mousePosition);
        RayCollision getRayCollisionBox(Ray ray, BoundingBox box);
        RayCollision getRayCollisionSphere(Ray ray, Vector3 center, float radius);
        bool checkCollisionBoxes(BoundingBox box1, BoundingBox box2);

        // Utility methods for 3D collisions
        Ray getMouseRayFromCurrent();
        BoundingBox createBoundingBox(Vector3 center, Vector3 size);
        BoundingBox createBoundingBoxFromMinMax(Vector3 min, Vector3 max);

        // Texture methods
        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);

        // Texture map accessor methods
        bool hasTexture(const std::string& id) const;
        Texture2D getTexture(const std::string& id) const;
        void addTexture(const std::string& id, Texture2D texture);

        // Input methods
        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;

        // Gamepad methods
        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;

        // Input type tracking methods
        InputType getLastInputType() const;
        void updateLastInputType();

        // Scissor mode methods for clipping
        void beginScissorMode(int x, int y, int width, int height);
        void endScissorMode();

        // 3D Environment methods
        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;

        // Camera methods
        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;

        // 3D Drawing methods
        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);

        // 3D Model methods
        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;

        // Skybox methods
        bool loadSkybox(const std::string& id, const std::string& filepath);
        void drawSkybox(const std::string& id);
        Color getDayNightColor(float cycleTime);
        float getTime() const;
```

```cpp

        // 2D Drawing methods
        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;

        // Font methods
        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:
        bool _isInitialized;
        Camera3D _camera;
        Vector2 _previousMousePosition;
        bool _isCursorLocked;
        InputType _lastInputType;

        static constexpr float FONT_SCALE_FACTOR = 4.0f;
        static constexpr float FONT_RENDER_SCALE = 0.25f;
        static constexpr float FONT_SPACING_RATIO = 0.1f;

        float getScaledFontSize(float fontSize) const;
        float getFontSpacing(float scaledFontSize) const;
        float getScaledSpacing(float spacing) const;

        struct ModelData {
            Model model;
            unsigned int animationCount;
            Vector3 center;
        };

        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;
};

#endif /* !RAYLIBEnc_HPP_ */
```

7.37 Constants.hpp

```cpp
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** Constants
*/

#ifndef CONSTANTS_HPP_
#define CONSTANTS_HPP_

    inline const float PLAYER_SCALE = 0.005f;
    inline const float EGG_SCALE = 1.0f;
    inline const float FOOD_SCALE = 0.005f;
    inline const float FOOD_FLOAT_AMPLITUDE = 0.05f;
    inline const float FOOD_FLOAT_SPEED = 1.5f;
    inline const char *CUSTOM_FONT_PATH = "gui/assets/fonts/fall.ttf";
    inline const float ROCK_SCALE = 0.2f;
    inline const float LINEMATE_SCALE = 0.2f;     // soccerball
    inline const float DERAUMERE_SCALE = 0.15f;   // beachball
    inline const float SIBUR_SCALE = 0.15f;       // basketball
    inline const float MENDIANE_SCALE = 0.18f;    // bowlingball
    inline const float PHIRAS_SCALE = 0.1f;       // eightball
    inline const float THYSTAME_SCALE = 0.1f;     // tennisball

#include <string>
#include <vector>
#include "HelpText.hpp"
#include "../IDisplay.hpp"

namespace zappy::constants {

    inline const char *USAGE_STRING = "USAGE: ./zappy_gui -p port -h machine\n"
                                      "option\t\tdescription\n"
                                      "-p port\t\tport number\n"
                                      "-h machine\thostname of the server";

    inline const int FAILURE_EXIT_CODE = 84;
    inline const int SUCCESS_EXIT_CODE = 0;
};

namespace colors {

    inline const char *T_BOLD = "\033[1m";
    inline const char *T_RED = "\033[1m\033[31m";
    inline const char *T_GREEN = "\033[1m\033[32m";
    inline const char *T_YELLOW = "\033[1m\033[33m";
    inline const char *T_BLUE = "\033[1m\033[34m";
    inline const char *T_MAGENTA = "\033[1m\033[35m";
    inline const char *T_CYAN = "\033[1m\033[36m";
    inline const char *T_WHITE = "\033[1m\033[37m";
    inline const char *RESET = "\033[0m";

};

namespace zappy::structs {

    struct Config {
        int port;
        std::string hostname;
    };

    struct Tile {
        int x;
        int y;
        int food;
        int linemate;
        int deraumere;
        int sibur;
        int mendiane;
        int phiras;
        int thystame;

        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)
            : x(_x), y(_y), food(_food), linemate(_linemate),
              deraumere(_deraumere), sibur(_sibur),
              mendiane(_mendiane), phiras(_phiras), thystame(_thystame) {}
    };

    struct Inventory {
        int food;
        int linemate;
        int deraumere;
        int sibur;
        int mendiane;
        int phiras;
        int thystame;

        Inventory(int _food = 0, int _linemate = 0, int _deraumere = 0,
                  int _sibur = 0, int _mendiane = 0, int _phiras = 0,
                  int _thystame = 0)
            : food(_food), linemate(_linemate), deraumere(_deraumere),
              sibur(_sibur), mendiane(_mendiane), phiras(_phiras),
              thystame(_thystame) {}
    };
    struct Player {
        int number;
        int x;
        int y;
        int orientation;
        int level;
        std::string teamName;
        struct Inventory inventory;

        Player(int _number = 0, int _x = 0, int _y = 0, int _orientation = 0,
               int _level = 1, const std::string &_teamName = "",
               struct Inventory _inventory = Inventory())
            : number(_number), x(_x), y(_y), orientation(_orientation),
              level(_level), teamName(_teamName), inventory(_inventory) {}
    };

    struct Incantation {
        int x;
        int y;
        int level;
        std::vector<int> players;

        Incantation(int _x = 0, int _y = 0, int _level = 1,
                    const std::vector<int> &_players = {})
            : x(_x), y(_y), level(_level), players(_players) {}
    };

    struct Egg {
        int eggNumber;
        int playerNumber;
        int x;
        int y;
        bool hatched;
        std::string teamName;

        Egg(int _eggNumber = 0, int _playerNumber = 0, int _x = 0, int _y = 0,
            bool _hatched = false, const std::string &_teamName = "")
            : eggNumber(_eggNumber), playerNumber(_playerNumber), x(_x), y(_y),
              hatched(_hatched), teamName(_teamName) {}
    };
};

namespace zappy::gui {

    inline const std::string WINDOW_TITLE = "Zappy GUI";
    inline const std::string CUSTOM_FONT_PATH = "gui/assets/fonts/fall.ttf";
    inline const int FPS = 120;
    inline const float CAMERA_SENSITIVITY = 0.001f;
    inline const float GAMEPAD_STICK_SENSITIVITY = 3.0f;
    inline const float GAMEPAD_DEADZONE = 0.2f;
    inline const float POSITION_MULTIPLIER = 2.2f;

    inline const float FOG_DISTANCE_MAX = 60.0f;
    inline const float DURATION_DAYNIGHT_CYCLE = 120.0f;

    inline const float EGG_SCALE = 1.0f;
    inline const float FOOD_SCALE = 0.005f;
    inline const float FOOD_FLOAT_AMPLITUDE = 0.05f;
    inline const float FOOD_FLOAT_SPEED = 0.10f;

    inline const float LINEMATE_SCALE = 0.2f;     // soccerball
    inline const float DERAUMERE_SCALE = 0.15f;  // beachball
    inline const float SIBUR_SCALE = 0.15f;      // basketball
    inline const float MENDIANE_SCALE = 0.18f;   // bowlingball
    inline const float PHIRAS_SCALE = 0.1f;      // eightball
    inline const float THYSTAME_SCALE = 0.1f;    // tennisball

    inline const float PLAYER_ROTATION_SPEED = 720.0f;
    inline const float ROTATION_INTERPOLATION_THRESHOLD = 1.0f;

    inline const float PLAYER_MOVEMENT_SPEED = 8.0f;
    inline const float MOVEMENT_INTERPOLATION_THRESHOLD = 0.05f;

    enum class CameraMode {
        FREE = 0,
        TARGETED = 1,
        PLAYER = 2,
        NB_MODES = 3,
    };


    struct PlayerModelInfo {
        std::string name;
        std::string modelPath;
        Vector3f center;
        Vector3f scale;
        float rotation;
    };

    inline const std::vector<PlayerModelInfo> PLAYER_MODELS_INFO = {
        {"playerLvl1", "gui/assets/models/playerLvl1.glb",
            {0.0f, -0.0f, 0.0f}, {0.005f, 0.005f, 0.005f}, 0.0f},
        {"playerLvl2", "gui/assets/models/playerLvl2.glb",
            {0.0f, -0.5f, 0.0f}, {0.25f, 0.25f, 0.25f}, 0.0f},
        {"playerLvl3", "gui/assets/models/playerLvl3.glb",
            {0.0f, 20.0f, 0.0f}, {0.0045f, 0.0045f, 0.0045f}, 0.0f},
        {"playerLvl4", "gui/assets/models/playerLvl4.glb",
            {0.0f, 0.0025f, 0.0f}, {40.0f, 40.0f, 40.0f}, -90.0f},
        {"playerLvl5", "gui/assets/models/playerLvl5.glb",
            {8.0f, -1.8f, 0.0f}, {0.2f, 0.2f, 0.2f}, 0.0f},
        {"playerLvl6", "gui/assets/models/playerLvl6.glb",
            {0.0f, 20.0f, 0.0f}, {0.009f, 0.009f, 0.009f}, 0.0f},
        {"playerLvl7", "gui/assets/models/playerLvl7.glb",
            {0.0f, 0.4f, 0.0f}, {0.25f, 0.25f, 0.25f}, 0.0f},
        {"playerLvl8", "gui/assets/models/playerLvl8.glb",
            {0.0f, 1.0f, 0.0f}, {0.085, 0.085f, 0.085f}, 0.0f}
    };
}

#endif /* !CONSTANTS_HPP_ */
```

7.38 GamepadConstants.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** GamepadConstants
*/

#ifndef GAMEPAD_CONSTANTS_HPP_
#define GAMEPAD_CONSTANTS_HPP_

#ifndef GAMEPAD_AXIS_LEFT_X
    #define GAMEPAD_AXIS_LEFT_X       0
    #define GAMEPAD_AXIS_LEFT_Y       1
    #define GAMEPAD_AXIS_RIGHT_X      2
    #define GAMEPAD_AXIS_RIGHT_Y      3
#endif

#ifndef GAMEPAD_BUTTON_A
    #define GAMEPAD_AXIS_LEFT_TRIGGER 4
    #define GAMEPAD_AXIS_RIGHT_TRIGGER 5
    #define GAMEPAD_BUTTON_A          6
    #define GAMEPAD_BUTTON_B          5
    #define GAMEPAD_BUTTON_X          9
    #define GAMEPAD_BUTTON_Y          8
    #define GAMEPAD_BUTTON_START      17
    #define GAMEPAD_BUTTON_SELECT     16
    #define GAMEPAD_BUTTON_UP         1
    #define GAMEPAD_BUTTON_RIGHT      2
    #define GAMEPAD_BUTTON_DOWN       3
    #define GAMEPAD_BUTTON_LEFT       4
    #define GAMEPAD_BUTTON_LEFT_SHOULDER  10
    #define GAMEPAD_BUTTON_RIGHT_SHOULDER 12
    #define GAMEPAD_BUTTON_LEFT_TRIGGER   13
    #define GAMEPAD_BUTTON_RIGHT_TRIGGER  15
#endif

#endif /* !GAMEPAD_CONSTANTS_HPP_ */
```

7.39 HelpText.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** HelpText
*/

#ifndef HELP_TEXT_HPP_
#define HELP_TEXT_HPP_

namespace zappy::constants {

    inline const char *HELP_TITLE =
        "HELP";

    inline const char *HELP_SECTION_1 =
        "Game Overview";

    inline const char *HELP_SECTION_1_CONTENT =
        "Zappy is a game where AI-controlled players compete to collect resources\n"
        "and level up on a dynamically changing map. The GUI allows you to visualize\n"
        "the game state, players, and resources in real-time.";

    inline const char *HELP_SECTION_2 =
        "Controls";

    inline const char *HELP_SECTION_2_CONTENT =
        "Camera Movement:\n"
        "  - Arrow keys or ZQSD: Move camera\n"
        "  - Controller: Use left stick to move camera\n"
        "  - Right mouse button + drag: Rotate camera\n\n"
        "Interface:\n"
        "  - Click on players to see their stats\n"
        "  - Click on tiles to see their stats\n"
        "  - Use the RESET CAMERA button to return to default view\n"
        "  - Use the Settings button to adjust game settings";

    inline const char *HELP_SECTION_3 =
        "Teams and Players";

    inline const char *HELP_SECTION_3_CONTENT =
        "The left panel shows all teams and their player IDs.\n"
        "Players have different levels based on collected resources.\n"
        "The team that first gets a player to level 8 wins the game.";

    inline const char *HELP_SECTION_4 =
        "Resources";

    inline const char *HELP_SECTION_4_CONTENT =
        "Resources on the map are represented by different colored objects.\n"
        "Players collect these resources to perform rituals and level up.";

    inline const char *HELP_SECTION_5 =
        "Levels";

    inline const char *HELP_SECTION_6 =
        "Items";

}  // namespace zappy::constants

#endif /* !HELP_TEXT_HPP_ */
```

7.40 InputType.hpp

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** InputType
*/

#ifndef INPUTTYPE_HPP_
#define INPUTTYPE_HPP_

enum class InputType {
    KEYBOARD_MOUSE,
    GAMEPAD,
    NONE
};

#endif /* !INPUTTYPE_HPP_ */
```

7.41 algo.h

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** algo
*/

#ifndef ALGO_H_
    #define ALGO_H_

typedef struct tiles_s {
    int x;
    int y;
} tiles_t;

/* Algo.c */
tiles_t *shuffle_fisher(int width, int heigth);

#endif /* !ALGO_H_ */
```

7.42 game.h

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** game
*/

#include "buffer.h"
#include <time.h>
#include <pthread.h>

#ifndef GAME_H_
    #define GAME_H_

typedef struct action_request_s action_request_t;
```

```c
typedef struct action_queue_s action_queue_t;
typedef struct player_s player_t;

/* Definition of the directions */
typedef enum direction_e {
    NORTH = 1,
    EAST = 2,
    SOUTH = 3,
    WEST = 4
} direction_t;

/* definintion od the different element on the map */
typedef enum crystal_e {
    FOOD,
    LINEMATE,
    DERAUMERE,
    SIBUR,
    MENDIANE,
    PHIRAS,
    THYSTAME
} crystal_t;


/* This enum defines the priority of the action in the queue */
typedef enum action_priority_e {
    PRIORITY_CRITICAL = 0,
    PRIORITY_HIGH = 1,
    PRIORITY_MEDIUM = 2,
    PRIORITY_LOW = 3
} action_priority_t;

/* This strucuture allows use to define a 'queue' of the requests */
typedef struct action_queue_s {
    action_request_t *head;
    action_request_t *tail;
    int count;
} action_queue_t;


typedef struct egg_s {
    int id; /* Id of the egg */
    int posX;
    int posY;
    char *teamName;  /* Name of the team that laid it */
    int idLayer;  /* Id of the player that layed it */
    bool isHatched;
    struct egg_s *next;
} egg_t;

/* Struct that "handles" the network element */
typedef struct network_s {
    int fd;
    buffer_t *buffer;
} network_t;

/* Struct defining the inventory of tiles and players */
typedef struct inventory_s {
    int nbFood;
    int nbLinemate;
    int nbDeraumere;
    int nbSibur;
    int nbMendiane;
    int nbPhiras;
    int nbThystame;
} inventory_t;

/* Definition of the incantation structure */
typedef struct incantation_s {
    int levelt_to_reach;
    int nb_players;
    inventory_t required_inventory;
} incantation_t;


/* Player struct */
typedef struct player_s {
    int id;
    network_t *network;
    int level;
    int posX;
    int posY;
    direction_t direction;
    inventory_t *inventory;
    char *team;
    /* New aditions for the smart pollin */
    action_queue_t *pending_actions;
    time_t last_action_time;
    bool is_busy;
    int remaining_cooldown;
    char *current_action;
    /* Food timer for health system */
    int food_timer;  /* Time units until next food consumption */
    time_t last_food_check;  /* Last time food was checked */

    struct player_s *next;
} player_t;

/* This structure define the request strut */
typedef struct action_request_s {
    char *command;
    time_t timestamp;
    float time_limit;  // in game ticks (7/f, 42/f, etc.)
    action_priority_t priority;
    player_t *player;
    struct action_request_s *next;
} action_request_t;

/* Team Strcut */
typedef struct team_s {
    char *name;
    int nbPlayers;
    int nbPlayerAlive;
    player_t *players;
    struct team_s *next;
} team_t;


/* Structure that holds the size and array of tiles */
typedef struct map_t {
    int width;
    int height;
    egg_t *currentEggs;  /* List of current eggs */
    inventory_t **tiles;  /* Here we call inv for the tile*/
} map_t;


/* Map struct */
typedef struct game_s {
    team_t *teams;
    map_t *map;
} game_t;

#endif /* !GAME_H_ */
```

7.43 my.h

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** my
*/

#ifndef MY_H_
    #define MY_H_

int int_str_len(int value);
char *my_itoa(unsigned int nb);
int is_only_digits(const char *str);
int my_unsignedlen(unsigned int nb);

#endif /* !MY_H_ */
```

7.44 my.h

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** my
*/

#ifndef MY_H_
    #define MY_H_

int int_str_len(int value);
char *my_itoa(unsigned int nb);
int is_only_digits(const char *str);
int my_unsignedlen(unsigned int nb);
```

```

#endif /* !MY_H_ */
```

7.45 zappy.h

```
/*
** EPITECH PROJECT, 2025
** Zappy
** File description:
** Server :: Zappy header
*/

#include <stdbool.h>
#include <poll.h>
#include "game.h"
#include "my.h"

#ifndef ZAPPY_H_
    #define ZAPPY_H_

/* items handler */
typedef struct {
    char *name;
    void (*add_func)(inventory_t *);
} item_handler_t;


/* Cli parameter of the server */
typedef struct params_s {
    int port;
    int x;
    int y;
    int nb_team;
    char **teams;
    int nb_client;
    int freq;
    bool is_debug;
} params_t;

/* Structure to handle the network side of the gui*/
typedef struct graph_net_s {
    int fd;
    bool mapSent;
    struct graph_net_s *next;
} graph_net_t;

/* Unified polling structure for all clients */
typedef struct unified_poll_s {
    struct pollfd *fds;
    int count;
    int capacity;
} unified_poll_t;

/* Server part of the network */
typedef struct server_s {
    int sockfd;
    struct pollfd pollserver;
} server_t;

typedef struct zappy_s {
    server_t *network;
    game_t *game;
    graph_net_t *graph;
    params_t *params;
    unified_poll_t *unified_poll;
} zappy_t;

typedef struct command_pf_s {
    char const *flag;
    bool (*checker)(const char *, const char *, params_t *);
} command_pf_t;

typedef struct {
    char *command;
    float base_time;
    action_priority_t priority;
    int (*handler)(player_t *, char *, zappy_t *);
} command_info_t;

typedef struct graphic_pf_s {
    char *command;
    int (*handler)(zappy_t *zappy, graph_net_t *graphic, char *message);
} graphic_pf_t;

/* messages.c */
```

```
int helper(void);
void error_message(const char *message);
void valid_message(char const *message);

/* checkers.c */
bool check_port(char const *flag, char const *value, params_t *params);
bool check_width(char const *flag, char const *value, params_t *params);
bool check_height(char const *flag, char const *value, params_t *params);
bool check_client(char const *flag, char const *value, params_t *params);
bool check_freq(char const *flag, char const *value, params_t *params);

/* unified_poll.c */
unified_poll_t *init_unified_poll(void);
void free_unified_poll(unified_poll_t *poll_struct);
int add_fd_to_poll(unified_poll_t *poll_struct, int fd, short events);
int remove_fd_from_poll(unified_poll_t *poll_struct, int fd);
void rebuild_poll_fds(zappy_t *zappy);
void poll_all_clients(zappy_t *zappy);


/* signal.c */
void setup_signal(void);
int *get_running_state(void);

/* params.c */
params_t *check_args(int argc, char **argv);
void *free_params(params_t *params);

/* params_cherckers.c */
bool validate_no_extra_args(int argc, char **argv);

/* server.c */
zappy_t *init_server(int argc, char **argv);
void *free_zappy(zappy_t *server);

/* protocol.c */
int start_protocol(zappy_t *server);

/* client.c */
bool process_new_client(const char *team_name, int fd, zappy_t *server);
team_t *add_client_to_team(const char *team_name, int fd, zappy_t *server);
void check_player_status(zappy_t *zappy);
void remove_player_by_fd(zappy_t *zappy, int fd);

/* init_map.c */
void init_game(zappy_t *server);
int distribute_resources(zappy_t *z);

/* init_team.c */
void init_teams(zappy_t *server);

/* accept.c */
int accept_client(zappy_t *server);

/* refill_food.c */
void count_current_resources(zappy_t *z, int current_count[7]);
void refill_food(zappy_t *zappy);

/* free server  */
void *free_zappy(zappy_t *server);
void *free_params(params_t *params);
void *free_player(player_t *player);
void free_map(map_t *map);

/* Function to send info to the gui */
int send_map_size(zappy_t *server);
int send_entrie_map(zappy_t *server);
int send_map_tile(inventory_t **tiles, zappy_t *server,
    int posX, int posY);
int send_team_name(zappy_t *server);
int send_egg(zappy_t *zappy, egg_t *egg);
int send_entire_egg_list(zappy_t *zappy);
int send_time_message(zappy_t *zappy);
int send_egg_death(zappy_t *zappy, egg_t *egg);
int send_egg_connect(zappy_t *zappy, egg_t *currentEgg);
int send_player_connect(zappy_t *zappy, player_t *player);
int send_player_pos(zappy_t *zappy, player_t *player);
int send_player_level(zappy_t *zappy, player_t *player);
int send_player_connect_to_specific_gui(graph_net_t *fd, player_t *p);
int send_player_inventory(zappy_t *zappy, player_t *player);
int send_player_expelled(zappy_t *zappy, player_t *player);
int send_broadcast_to_all(zappy_t *zappy, const char *message);
int send_broadcast_to_player(zappy_t *zappy, player_t *player,
    const char *message);
int send_player_laying_egg(zappy_t *zappy, player_t *player);
int send_ressource_droped(zappy_t *zappy, player_t *player,
    int ressourceType);
int send_ressource_collected(zappy_t *zappy, player_t *player,
    int ressourceType);
int send_player_death(zappy_t *zappy, player_t *player);
int send_updated_time(zappy_t *zappy, int time);
int send_end_game(zappy_t *zappy, const char *teamName);
int send_str_message(zappy_t *zappy, const char *message);
int send_unknown_command(zappy_t *zappy);
int send_command_parameter(zappy_t *zappy);
int send_start_incantation(zappy_t *zappy, player_t *player, int *player_list,
    int nb_player);
int send_end_incantation(zappy_t *zappy, player_t *player, char *result);

/* init_egg.c */
void init_egg(zappy_t *zappy);
egg_t *add_egg_node(int id, int *pos, char *team_name, int id_layer);
egg_t *kil_egg_node(egg_t **head, int egg_id);

/* AI messages */
int forward_message(player_t *player, params_t *params);

/* Pollin handler */
void process_player_actions(player_t *player, zappy_t *zappy);
void process_player_actions_tick(zappy_t *zappy);
void execute_action(player_t *player, action_request_t *action,
    zappy_t *zappy);
void queue_action(player_t *player, char *command, zappy_t *zappy);
action_queue_t *init_action_queue(void);
void free_action_queue(action_queue_t *queue);
action_request_t *create_action_request(char *command, player_t *player,
    int frequency);
const command_info_t *find_command_info(char *command);
action_request_t *dequeue_highest_priority_action(action_queue_t *queue);
void free_action_request(action_request_t *action);
void insert_action_by_priority(action_queue_t *queue,
    action_request_t *action);

/* Unified polling functions */
unified_poll_t *init_unified_poll(void);
void free_unified_poll(unified_poll_t *poll_struct);
int add_fd_to_poll(unified_poll_t *poll_struct, int fd, short events);
int remove_fd_from_poll(unified_poll_t *poll_struct, int fd);
void poll_all_clients(zappy_t *zappy);
void rebuild_poll_fds(zappy_t *zappy);

/* This is the definition of the array function of the commands */
int handle_forward(player_t *player, char *command, zappy_t *zappy);

int handle_left(player_t *player, char *command, zappy_t *zappy);
int left_message(player_t *player);
int print_left_server(player_t *player);

int handle_right(player_t *player, char *command, zappy_t *zappy);
int print_right_server(player_t *player);
int right_message(player_t *player);

int handle_connect_nbr(player_t *player, char *command, zappy_t *zappy);
int handle_eject(player_t *player, char *command, zappy_t *zappy);

/* fork */
int handle_fork(player_t *player, char *command, zappy_t *zappy);
int handle_fork_end(player_t *player, zappy_t *zappy);

int print_look_server(player_t *player);

/* Incantation handler */
int handle_incantation(player_t *player, char *command, zappy_t *zappy);
int check_player_on_tile(player_t *player, zappy_t *zappy);
void increase_level_player(int *player_list, int nb_players, zappy_t *zappy);
int *get_player_on_tile_id(int posX, int posY, zappy_t *zappy, int nb_players);
int handle_end_incantation(player_t *player, zappy_t *zappy);
int get_nb_player_on_tile(int posX, int posY, zappy_t *zappy, int level);
void mark_players_incanting(int *player_list, int nb_players, zappy_t *zappy);
void remove_crystal_from_tiles(int posX, int posY, int level, zappy_t *zappy);
int validate_and_get_players(player_t *player, zappy_t *zappy,
    int **player_list);


int handle_inventory(player_t *player, char *command, zappy_t *zappy);
int inventory_message(player_t *player);
int print_inventory_server(player_t *player, int len);

int handle_broadcast(player_t *player, char *command, zappy_t *zappy);
int broadcast_text(player_t *source, player_t *dest, char *text,
    zappy_t *zappy);

int handle_look(player_t *player, char *command, zappy_t *zappy);
int handle_set(player_t *player, char *command, zappy_t *zappy);
int handle_take(player_t *player, char *command, zappy_t *zappy);

/* graphic_clinet.c */
graph_net_t *add_graph_node(graph_net_t **head, int fd);
graph_net_t *remove_graph_node(graph_net_t **head, int fd);
int poll_graphic_commands(zappy_t *zappy, graph_net_t *current,
    char *buffer);


/* Element hander.c */
void add_food(inventory_t *inventory);
void add_linemate(inventory_t *inventory);
void add_deraumere(inventory_t *inventory);
void add_sibur(inventory_t *inventory);
void add_mendiane(inventory_t *inventory);
void add_phiras(inventory_t *inventory);
void add_thystame(inventory_t *inventory);

void rm_food(inventory_t *inventory);
void rm_linemate(inventory_t *inventory);
void rm_deraumere(inventory_t *inventory);
void rm_sibur(inventory_t *inventory);
void rm_mendiane(inventory_t *inventory);
void rm_phiras(inventory_t *inventory);
void rm_thystame(inventory_t *inventory);

/* Element handler.c */
int msz(zappy_t *zappy, graph_net_t *graphic, char *message);
int bct(zappy_t *zappy, graph_net_t *graphic, char *message);
int mct(zappy_t *zappy, graph_net_t *graphic, char *message);
int tna(zappy_t *zappy, graph_net_t *graphic, char *message);
int ppo(zappy_t *zappy, graph_net_t *graphic, char *message);
int plv(zappy_t *zappy, graph_net_t *graphic, char *message);
int plu(zappy_t *zappy, graph_net_t *graphic, char *message);
int pld(zappy_t *zappy, graph_net_t *graphic, char *message);
int pin(zappy_t *zappy, graph_net_t *graphic, char *message);
int sgt(zappy_t *zappy, graph_net_t *graphic, char *message);
int sst(zappy_t *zappy, graph_net_t *graphic, char *message);
int kil(zappy_t *zappy, graph_net_t *graphic, char *message);
int tar(zappy_t *zappy, graph_net_t *graphic, char *message);
int tsr(zappy_t *zappy, graph_net_t *graphic, char *message);
int pia(zappy_t *zappy, graph_net_t *graphic, char *message);
int pis(zappy_t *zappy, graph_net_t *graphic, char *message);
int send_bct_message(graph_net_t *graphic, int x, int y,
    inventory_t *inventory);
int send_pin_message(graph_net_t *graphic, player_t *player);

/* player_id.c */
player_t *get_player_by_id(game_t *game, int player_id);
int get_next_free_id(zappy_t *server);
void verify_player_id(zappy_t *zappy, player_t *player);
#endif /* !ZAPPY_H_ */
```

7.46 buffer.h

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** buffer
*/

#include <stddef.h>

#ifndef BUFFER_H_
    #define BUFFER_H_

    #define BUFFER_SIZE 1024


typedef struct buffer_s {
    char data[BUFFER_SIZE];
    int head;
    int tail;
    int full;
} buffer_t;

/* buffer.c */
int advance(int idx);
void cb_write(buffer_t *cb, char c);
int cb_getline(buffer_t *cb, char *line, int max_len);

#endif /* !BUFFER_H_ */
```

7.47 buffer.h

```c
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** buffer
*/

#include <stddef.h>

#ifndef BUFFER_H_
    #define BUFFER_H_

    #define BUFFER_SIZE 1024


typedef struct buffer_s {
    char data[BUFFER_SIZE];
    int head;
    int tail;
    int full;
} buffer_t;

/* buffer.c */
int advance(int idx);
void cb_write(buffer_t *cb, char c);
int cb_getline(buffer_t *cb, char *line, int max_len);

#endif /* !BUFFER_H_ */
```

7.48 network.h

```c
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** network
*/

#ifndef NETWORK_H_
    #define NETWORK_H_

/* Write an errro message */
void error_print(char const *message);
/* Set the socket of the file descriptor */
int set_socket(void);
/* Bind the file decriptor to the port */
int bind_socket(int fd, int port);
/* Specify the queue the fd will use */
int listen_socket(int fd, int backlog);

/* Close the server */
void close_fd(int fd);

/* Accept new connetion */
int accept_connection(int server_fd);
/* Handle Message input */
char *get_message(int fd, int timeout);
/* Hello */
int write_message(int fd, const char *message);
#endif /* !NETWORK_H_ */
```

7.49 network.h

```c
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** network
*/

#ifndef NETWORK_H_
    #define NETWORK_H_

/* Write an errro message */
void error_print(char const *message);
/* Set the socket of the file descriptor */
int set_socket(void);
/* Bind the file decriptor to the port */
int bind_socket(int fd, int port);
/* Specify the queue the fd will use */
```

```
int listen_socket(int fd, int backlog);

/* Close the server */
void close_fd(int fd);

/* Accept new connetion */
int accept_connection(int server_fd);
/* Handle Message input */
char *get_message(int fd, int timeout);
/* Hello */
int write_message(int fd, const char *message);
#endif /* !NETWORK_H_ */
```

7.50 fake_malloc.h

```
/*
** EPITECH PROJECT, 2025
** B-YEP-400-NAN-4-1-zappy-albane.merian
** File description:
** fake_malloc
*/

#ifndef FAKE_MALLOC_H_
    #define FAKE_MALLOC_H_


void enable_malloc_failure(int after_calls);
void disable_malloc_failure(void);
void reset_malloc_counter(void);
void *malloc(size_t size);
void *calloc(size_t nmemb, size_t size);

#endif /* !FAKE_MALLOC_H_ */
```