Zappy architecture

Generated by Doxygen 1.12.0

Chapter 1

README

1.1 ZAPPY

A multiplayer network strategy game where teams compete for supremacy!

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

1.1.1 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**.

1.1.1.1 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

1.1.2 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
```

1.1.2.1 Technologies Used

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

1.1.3 Quick Start

1.1.3.1 Prerequisites

Before running Zappy, ensure you have:

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

1.1.3.2 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>
   ```

1.1.4 Documentation

1.1.4.1 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
> ```

1.1.4.2 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.

1.1.5 Contributing

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

1.1.5.1 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

1.1.5.2 Gitmoji Reference

1.1.5.2.1 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

1.1.5.2.2 Testing

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

1.1.5.2.3 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

1.1.5.2.4 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!**

1.1.6 Git Commands Reference

1.1.6.1 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
```

1.1.6.2 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)"
```

1.1.7 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/`.

1.1.8 Team

Project developed by EPITECH students

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

Chapter 2

Zappy Server

A server, created in C, that generates the inhabitants' world.

2.1 Usage

```
USAGE: ./zappy_server -p port -x width -y height -n name1 name2 ... -c clientsNb -f freq --auto-start on|off
      --display-eggs true|false [-v | --verbose]
    port        is the port number
    width       is the width of the world
    height      is the height of the world
    nameX       is the name of the team X
    clientsNb   is the number of authorized clients per team
    freq        is the reciprocal of time unit for execution of actions
    auto-start  does the greeting is send automaticly #(see bonus part)
    display-eggs    eggs are visible and destructible
```

The server is executed in the form of one, single process and one, single thread. It must use select to handle socket multiplexing; the select must unlock only if something happen on a socket or if an event is ready to be executed.

> The team name GRAPHIC is reserved for the GUI to authenticate itself as such to the server.

2.2 AI protocol

Each player responds to the following actions and only to these ones, with the following syntax :

Action	Command	Time limit	Response
move up one tile	**Forward**	7/f	ok
turn 90° right	**Right**	7/f	ok
turn 90° left	**Left**	7/f	ok
look around	**Look**	7/f	[tile1, tile2,...]
inventory	**Inventory**	1/f	[linemate n, sibur n, ...]
broadcast text	**Broadcast text**	7/f	ok
number of team unused slots	**Connect_nbr**	-	value
fork a player	**Fork**	42/f	ok
eject players from this tile	**Eject**	7/f	ok/ko
death of a player	-	-	dead
take object	**Take object**	7/f	ok/ko
set object down	**Set object**	7/f	ok/ko

| start incantation | **Incantation** | 300/f | Elevation underway | Current level: k/ko |

> In case of a bad/unknown command, the server must answer "ko".

The AI client's connection to the server happens as follows:

```
1. the client opens a socket on the server's port,

2. the server and the client communicate the following way:
    Server --> WELCOME\n
           <-- TEAM-NAME\n
           --> game informations (see tha above array)
```

X and Y indicate the world's dimensions.
CLIENT-NUM indicates the number of slots available on the server for the TEAM-NAME team. If this number is greater than or equal to 1, a new client can connect.

> The client can send up to 10 requests in a row without any response from the server. Over 10, the server will drop the incomming commands.

The server executes the client's requests in the order they were received.
The requests are buffered and a command's execution time only blocks the player in question.
Trantorians have adopted an international time unit. The time unit is seconds.
An action's execution time is calculated with the following formula:
action / f
Where f is an integer representing the reciprocal (multiplicative inverse) of time unit.
For instance, if f=1, "forward" takes 7 / 1 = 7 seconds.
By default f=100.

2.3 GUI protocol

SYMBOL	MEANING
X	width or horizontal position
Y	height or vertical position
q0	resource 0 (food) quantity
q1	resource 1 (linemate) quantity
q2	resource 2 (deraumere) quantity
q3	resource 3 (sibur) quantity
q4	resource 4 (mendiane) quantity
q5	resource 5 (phiras) quantity
q6	resource 6 (thystame) quantity
n	player number
O	orientation: 1(N), 2(E), 3(S), 4(W)
L	player or incantation level
e	egg number
T	time unit
N	name of the team
R	incantation result
M	message
i	resource number

SERVER	CLIENT	DETAILS	TO A GUI client	TO ALL GUI client
msz X Y	msz	map size	new GUI client connection or msz command	
bct X Y q0 q1 q2 q3 q4 q5 q6	bct X Y	content of a tile	bct command	

SERVER	CLIENT	DETAILS	TO A GUI client	TO ALL GUI client
bct X Y q0 q1 q2 q3 q4 q5 q6 ∗ nbr_tiles	mct	content of the map (all the tiles)	new GUI client connection or mct command or map refill	
tna N ∗ nbr_teams	tna	name of all the teams	new GUI client connection	
pnw #n X Y O L N		connection of a new player	new GUI client connection	new AI client connection
ppo #n X Y O	ppo #n	player's position	ppo command	AI left, right forward action or AI is ejected
plv #n L	plv #n	player's level	new GUI client connection or plv command	AI sucessfully incantate
pin #n X Y q0 q1 q2 q3 q4 q5 q6	pin #n	player's inventory	new GUI client connection or pin command	new AI client connection or AI set, take action or AI lost food
pex #n		expulsion		AI eject action
pbc #n M		broadcast		AI broadcast action
pic X Y L #n #n ...		start of an incantation (by the first player)		AI incantation action
pie X Y R		end of an incantation		AI incatation end
pfk #n		egg laying by the player		AI fork action
pdr #n i		resource dropping		AI set action
pgt #n i		resource collecting		AI take action
pdi #n		death of a player		AI client disconnection or AI lost all it's food
enw #e #n X Y		an egg was laid by a player	new GUI client connection	AI fork action end (after 42/f)
ebo #e		player connection for an egg		new AI client connection
edi #e		death of an egg		egg is ejected by an AI
sgt T	sgt	time unit request	new GUI client connection or sgt	sst command
sst T	sst T	time unit modification		
seg N		end of game		an AI team reach the victory conditions
smg M		message from the server		server send a message
suc		unknown command		empty or unknown command
sbp		command parameter		invalid command (wrong parameter.s)

The GUI client's connection to the server happens as follows:

```
1. the client opens a socket on the server's port,
```

```
2. the server and the client communicate the following way:
    Server --> WELCOME\n
           <-- GRAPHIC\n
           --> game informations (see the above array)
```

2.4 Informations

2.4.1 Incantations

This ritual, which augments physical and mental capacities, must be done according to a particular rite: they must gather the following on the same unit of terrain:

- At least a certain number of each stones
- At least a certain number of players with the same level

The elevation begins as soon as a player initiates the incantation. The player who starts an incantation will receive ko if all the requirements are not satisfied and the incantation will be canceled, the player will receive the ko instantly after the initial server check (not at the end of the incantation duration).
It is not necessary for the players to be on the same team; they only need to be of the same level. Every player with the corresponding level and present at the beginning and at the end of the incantation attain the higher level.
During the incantation, the participants can not make any action until the end of the rite.
At the end of the incantation, the exact quantity of resources needed by the rite are consumed.

2.5 Bonus

2.5.1 Server commands

The server accepts command in its standard input.

Command	Effect
/clients	list all connected clients
/quit	stop the server
/send_ais msg	send messages to all AI
/send_guis msg	send messages to all GUI
/map	display map informations
/clear	clear the shell
/pause	pause the AI's actions
/start	start the server
/setTile ressource quantity x y	set the given ressource quantity of a tile
/tile x y	get the inventory of a tile
/tp id x y	tp an AI by it's id
/kill id	kill an AI by it's id
/noFood true or false	disable the food management
/broadcast "message" x y	simulate a broadcast
/setLevel id level	set the level of an AI by it's id
/setInventory id ressource quantity	set the given ressource quantity inside an AI inventory by it's id
/setClientsNb nb	set the minimum number of AI per team
/setFreq freq	set the frequency of the server
/noRefill true or false	disable the map refill
/fork team x y	simulate a fork for the given team at the given position
/incantate x y	simulate an incantation of the given level at the given position

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

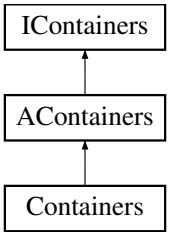
Here is a list of all documented files with brief descriptions:

Chapter 6

Class Documentation

6.1 AContainers Class Reference

Abstract base class for containers.
`#include <AContainers.hpp>`
Inheritance diagram for AContainers:



Public Member Functions

- AContainers (std::shared_ptr< RayLib > raylib, float x, float y, float width, float height)

  Construct a new AContainers object.
- virtual ~**AContainers** ()=default

  Destroy the AContainers object.
- void setPosition (float x, float y) override

  Set the position of the container.
- void setSize (float width, float height) override

  Set the size of the container.
- Rectangle getBounds () const override

  Get the current position of the container.
- bool contains (float x, float y) const override

  Check if a point is within the container.
- void setVisible (bool visible) override

  Set the visibility of the container.
- bool isVisible () const override

  Check if the container is visible.
- void setRelativePosition (float xPercent, float yPercent, float widthPercent, float heightPercent)

  Set position and size as percentages of screen dimensions.
- RelativePosition getRelativePosition () const

  Get the container's relative position.
- void **updatePositionFromRelative** ()

  Update the container's absolute position from relative position.

Public Member Functions inherited from IContainers

- virtual void draw ()=0

  Draw the container and its contents.
- virtual void update ()=0

  Update the container's state.

Protected Attributes

- std::shared_ptr< RayLib > **_raylib**
- Rectangle **_bounds**
- RelativePosition **_relativePos**
- Color **_backgroundColor**
- bool **_visible**
- bool **_hasBackground**

6.1.1 Detailed Description

Abstract base class for containers.
Provides common functionality for all container types

6.1.2 Constructor & Destructor Documentation

6.1.2.1 AContainers()

```
AContainers::AContainers (
            std::shared_ptr< RayLib > raylib,
            float x,
            float y,
            float width,
            float height)
```

Construct a new AContainers object.

Parameters

x	X coordinate
y	Y coordinate
width	Container width
height	Container height

6.1.3 Member Function Documentation

6.1.3.1 contains()

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

Check if a point is within the container.

Parameters

x	X coordinate
y	Y coordinate

Returns

true If point is within container

false Otherwise

Implements IContainers.

6.1.3.2 getBounds()

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

Get the current position of the container.

Returns

Rectangle Containing position and size

Implements IContainers.

6.1.3.3 getRelativePosition()

```
RelativePosition AContainers::getRelativePosition () const
```

Get the container's relative position.

Returns

RelativePosition The relative position and size

6.1.3.4 isVisible()

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

Check if the container is visible.

Returns

true If visible

false Otherwise

Implements IContainers.

6.1.3.5 setPosition()

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

Set the position of the container.

Parameters

x	X coordinate
y	Y coordinate

Implements IContainers.

6.1.3.6 setRelativePosition()

```
void AContainers::setRelativePosition (
            float xPercent,
            float yPercent,
            float widthPercent,
            float heightPercent)
```

Set position and size as percentages of screen dimensions.

Parameters

xPercent	X position as percentage of screen width (0-100)
yPercent	Y position as percentage of screen height (0-100)
widthPercent	Width as percentage of screen width (0-100)
heightPercent	Height as percentage of screen height (0-100)

6.1.3.7 setSize()

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

Set the size of the container.

Parameters

width	Container width
height	Container height

Implements IContainers.

6.1.3.8 setVisible()

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

Set the visibility of the container.

Parameters

visible	Visibility state

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**
- pthread_mutex_t **mutex**

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**

- int **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** = config["port"]
- **name** = config["name"]
- **ip** = config["machine"]
- list **childs** = [ ]

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

- ai/src/App/App.py

6.5 Audio Class Reference

Public Member Functions

- bool **loadSound** (const std::string &id, const std::string &filepath)
- void **playSound** (const std::string &id, float volume=1.0f)
- 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

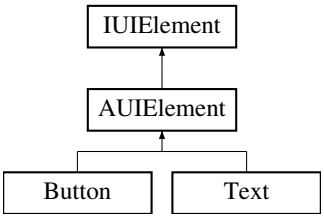
- std::map< std::string, std::unique_ptr< sf::Music > > **_sounds**

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

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

6.6 AUIElement Class Reference

Abstract base class for UI elements.
`#include <AUIElement.hpp>`
Inheritance diagram for AUIElement:



Public Member Functions

- AUIElement (std::shared_ptr< RayLib > raylib, float x, float y, float width, float height)

  Construct a new AUIElement object.
- virtual ~**AUIElement** ()=default

  Destroy the AUIElement object.
- void setPosition (float x, float y) override

  Set the position of the UI element.
- Rectangle getBounds () const override

  Get the bounds of the UI element.
- bool contains (float x, float y) const override

  Check if the UI element contains a point.
- void setVisible (bool visible) override

  Set the visibility of the UI element.
- bool isVisible () const override

  Check if the UI element is visible.
- virtual void setSize (float width, float height)

  Set the element size.
- void setRelativePosition (float xPercent, float yPercent, float widthPercent, float heightPercent)

  Set position and size as percentages of parent container.
- UIRelativePosition getRelativePosition () const

  Get the relative position.

Public Member Functions inherited from IUIElement

- virtual void draw ()=0

  Draw the UI element.
- virtual void update ()=0

  Update the UI element's state.

Protected Attributes

- std::shared_ptr< RayLib > **_raylib**
- Rectangle **_bounds**
- UIRelativePosition **_relativePos**
- bool **_visible**

6.6.1 Detailed Description

Abstract base class for UI elements.
Provides common functionality for all UI elements

6.6.2 Constructor & Destructor Documentation

6.6.2.1 AUIElement()

```
AUIElement::AUIElement (
            std::shared_ptr< RayLib > raylib,
            float x,
            float y,
            float width,
            float height)
```

Construct a new AUIElement object.

Parameters

x	X coordinate
y	Y coordinate
width	Element width
height	Element height

6.6.3 Member Function Documentation

6.6.3.1 contains()

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

Check if the UI element contains a point.

Parameters

x	X coordinate
y	Y coordinate

Returns

true If the point is inside the element

false Otherwise

Implements IUIElement.

6.6.3.2 getBounds()

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

Get the bounds of the UI element.

Returns

Rectangle The bounds of the element

Implements IUIElement.

6.6.3.3 getRelativePosition()

```
UIRelativePosition AUIElement::getRelativePosition () const
```

Get the relative position.

Returns

UIRelativePosition The relative position and size

6.6.3.4 isVisible()

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

Check if the UI element is visible.

Returns

true If visible

false Otherwise

Implements IUIElement.

6.6.3.5 setPosition()

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

Set the position of the UI element.

Parameters

x	X coordinate
y	Y coordinate

Implements IUIElement.

6.6.3.6 setRelativePosition()

```
void AUIElement::setRelativePosition (
            float xPercent,
            float yPercent,
            float widthPercent,
            float heightPercent)
```

Set position and size as percentages of parent container.

Parameters

xPercent	X position as percentage of container width (0-100)
yPercent	Y position as percentage of container height (0-100)
widthPercent	Width as percentage of container width (0-100)
heightPercent	Height as percentage of container height (0-100)

6.6.3.7 setSize()

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

Set the element size.

Parameters

width	New width
height	New height

Implements IUIElement.
Reimplemented in Button, and Text.

6.6.3.8 setVisible()

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

Set the visibility of the UI element.

Parameters

visible	Visibility state

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 buffer_s Struct Reference

Public Attributes

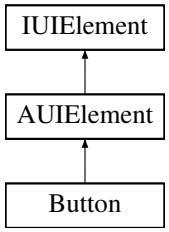
- char **data** [BUFFER_SIZE]
- int **head**
- int **tail**
- int **full**

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

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

6.8 Button Class Reference

Button UI element.
`#include <Button.hpp>`
Inheritance diagram for Button:



Public Member Functions

- Button (std::shared_ptr< RayLib > raylib, std::shared_ptr< Audio > audio, float x, float y, float width, float height, const std::string &text, std::function< void()> callback)

  Construct a new Button.
- ~**Button** () override=default

  Destroy the Button.
- void draw () override

  Draw the button.
- void update () override

  Update the button state.
- void setText (const std::string &text)

Set the text of the button.

- std::string getText () const

  Get the text of the button.

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

  Set the callback function.

- void setColors (Color normal, Color hover, Color pressed, Color textColor)

  Set the colors of the button.

- void setSize (float width, float height) override

  Set the size of the button.

Public Member Functions inherited from AUIElement

- AUIElement (std::shared_ptr< RayLib > raylib, float x, float y, float width, float height)

  Construct a new AUIElement object.

- virtual ~**AUIElement** ()=default

  Destroy the AUIElement object.

- void setPosition (float x, float y) override

  Set the position of the UI element.

- Rectangle getBounds () const override

  Get the bounds of the UI element.

- bool contains (float x, float y) const override

  Check if the UI element contains a point.

- void setVisible (bool visible) override

  Set the visibility of the UI element.

- bool isVisible () const override

  Check if the UI element is visible.

- void setRelativePosition (float xPercent, float yPercent, float widthPercent, float heightPercent)

  Set position and size as percentages of parent container.

- UIRelativePosition getRelativePosition () const

  Get the relative position.

Public Member Functions inherited from IUIElement

Private Attributes

- std::string **_text**
- std::function< void()> **_callback**
- Color **_normalColor**
- Color **_hoverColor**
- Color **_pressedColor**
- Color **_textColor**
- bool **_isHovered**
- bool **_isPressed**
- std::shared_ptr< RayLib > **_raylib**
- std::shared_ptr< Audio > **_audio**

Additional Inherited Members

Protected Attributes inherited from AUIElement

- std::shared_ptr< RayLib > **_raylib**
- Rectangle **_bounds**
- UIRelativePosition **_relativePos**
- bool **_visible**

6.8.1 Detailed Description

Button UI element.
A clickable button with text that can trigger a callback when clicked

6.8.2 Constructor & Destructor Documentation

6.8.2.1 Button()

```
Button::Button (
            std::shared_ptr< RayLib > raylib,
            std::shared_ptr< Audio > audio,
            float x,
            float y,
            float width,
            float height,
            const std::string & text,
            std::function< void()> callback)
```

Construct a new Button.

Parameters

x	X coordinate
y	Y coordinate
width	Button width
height	Button height
text	Button text
callback	Function to call when button is clicked

6.8.3 Member Function Documentation

6.8.3.1 draw()

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

Draw the button.
Implements IUIElement.

6.8.3.2 getText()

```
std::string Button::getText () const
```

Get the text of the button.

Returns

std::string Button text

6.8.3.3 setCallback()

```
void Button::setCallback (
            std::function< void()> callback)
```

Set the callback function.

Parameters

callback	Function to call when button is clicked

6.8.3.4 setColors()

```
void Button::setColors (
            Color normal,
            Color hover,
            Color pressed,
            Color textColor)
```

Set the colors of the button.

Parameters

normal	Normal color
hover	Hover color
pressed	Pressed color
textColor	Text color

6.8.3.5 setSize()

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

Set the size of the button.

Parameters

width	New button width
height	New button height

Reimplemented from AUIElement.

6.8.3.6 setText()

```
void Button::setText (
            const std::string & text)
```

Set the text of the button.

Parameters

text	New text

6.8.3.7 update()

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

Update the button state.
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

6.9 CameraManager Class Reference

Public Member Functions

- **CameraManager** (std::shared_ptr< RayLib > raylib)
- void **updateCamera** (zappy::gui::CameraMode mode)

- void **updateCameraFreeMode** ()
- void **updateCameraTargetMode** ()
- void **updateCameraPlayerMode** ()
- void **setMapCenter** (const Vector3 &center)
- void **setMapSize** (int width, int height)
- float **getCurrentCameraDistance** () const
- 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)

Private Member Functions

- void **handlePlayerCameraMouseInput** ()
- Vector3 **calculatePlayerPosition** (const zappy::structs::Player &player)
- Vector3 **calculateCameraPosition** (const Vector3 &playerPos, float angleXZ)

Private Attributes

- std::shared_ptr< RayLib > **_raylib**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< Map > **_map**
- Vector3 **_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

6.10 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

6.11 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** = None
- **name** = None
- str **machine** = "127.0.0.1"
- bool **port** = True
- bool **name** = True
- int **machine** = 2

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

- ai/src/CLI/CLI.py

6.12 Client Class Reference

Public Member Functions

- **Client** (int ac, const char ∗const ∗av)

Private Member Functions

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

Private Attributes

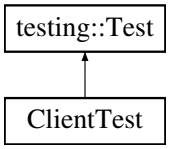
- zappy::structs::Config **_config**
- std::shared_ptr< ICommunication > **_communication**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::unique_ptr< MsgHandler > **_msgHandler**
- std::unique_ptr< GUI > **_gui**

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

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

6.13 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

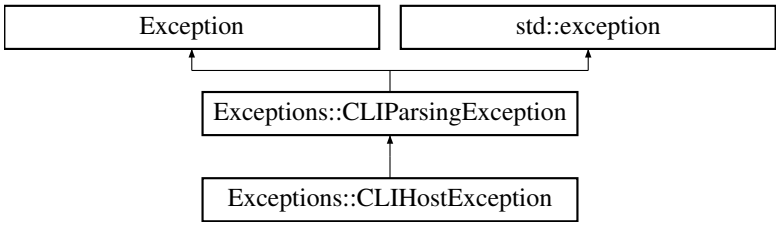
- std::stringstream **buffer**
- std::streambuf ∗ **originalCout**

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

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

6.14 Exceptions::CLIHostException Class Reference

Inheritance diagram for Exceptions::CLIHostException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::CLIParsingException

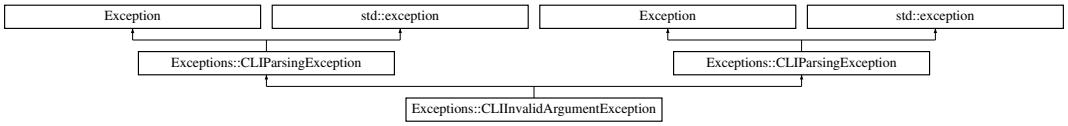
- **__init__** (self, str message)
- **CLIParsingException** (const std::string &message)
- const char ∗ **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.15 Exceptions::CLIInvalidArgumentException Class Reference

Inheritance diagram for Exceptions::CLIInvalidArgumentException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::CLIParsingException

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

6.15.1 Constructor & Destructor Documentation

6.15.1.1 __init__()

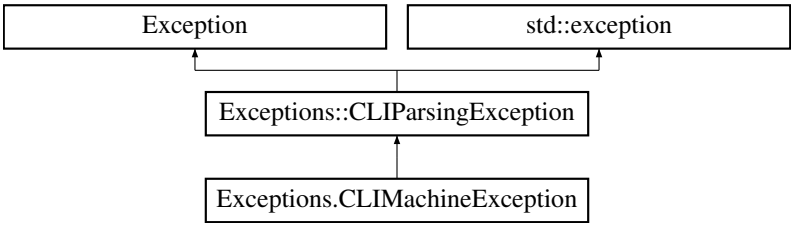
```
Exceptions.CLIInvalidArgumentException.__init__ (
              self,
             str message)
```

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

6.16 Exceptions.CLIMachineException Class Reference

Inheritance diagram for 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

6.16.1 Constructor & Destructor Documentation

6.16.1.1 __init__()

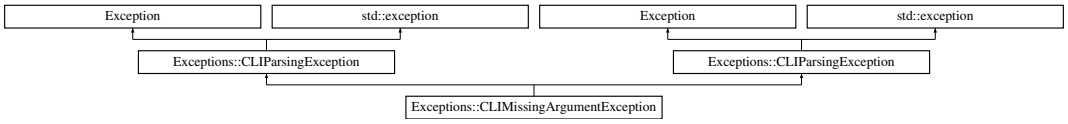
```
Exceptions.CLIMachineException.__init__ (
             self,
            str message)
```

Reimplemented from Exceptions::CLIParsingException.
The documentation for this class was generated from the following file:

- ai/src/Exceptions/Exceptions.py

6.17 Exceptions::CLIMissingArgumentException Class Reference

Inheritance diagram for Exceptions::CLIMissingArgumentException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::CLIParsingException

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

6.17.1 Constructor & Destructor Documentation

6.17.1.1 __init__()

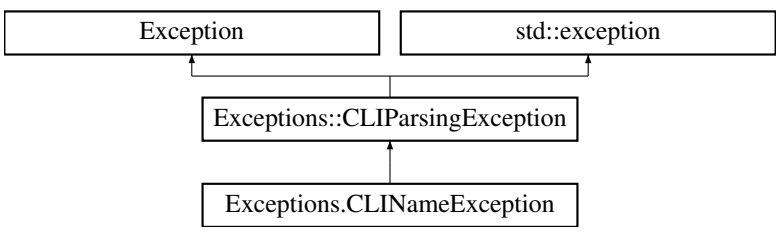
```
Exceptions.CLIMissingArgumentException.__init__ (
             self,
            str message)
```

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

6.18 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

6.18.1 Constructor & Destructor Documentation

6.18.1.1 __init__()

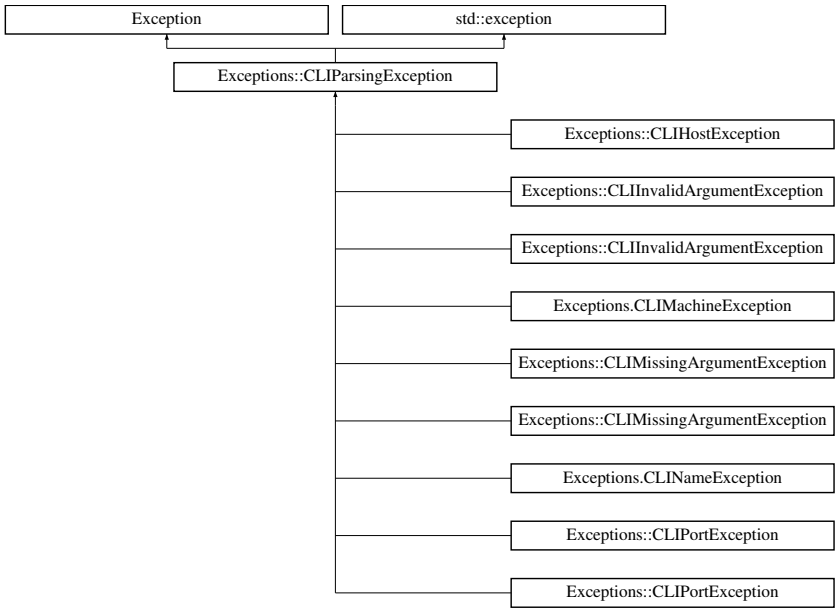
```
Exceptions.CLINameException.__init__ (
              self,
             str message)
```

Reimplemented from Exceptions::CLIParsingException.
The documentation for this class was generated from the following file:

- ai/src/Exceptions/Exceptions.py

6.19 Exceptions::CLIParsingException Class Reference

EPITECH PROJECT, 2025 zappy File description: Exceptions.
Inheritance diagram for Exceptions::CLIParsingException:



Public Member Functions

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

Private Attributes

- std::string **_message**

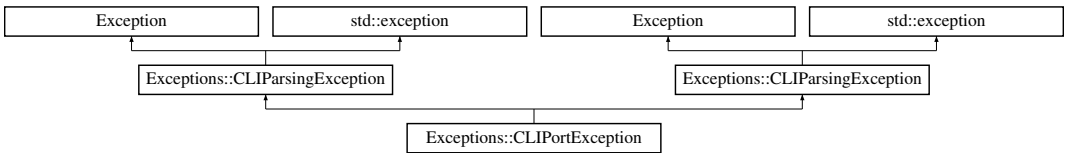
6.19.1 Detailed Description

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

6.20 Exceptions::CLIPortException Class Reference

Inheritance diagram for Exceptions::CLIPortException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::CLIParsingException

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

6.20.1 Constructor & Destructor Documentation

6.20.1.1 __init__()

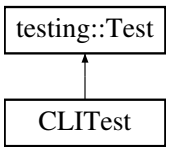
```
Exceptions.CLIPortException.__init__ (
              self,
             str message)
```

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

6.21 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

6.22 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

6.23 command_info_t Struct Reference

Public Attributes

- char * **command**
- int **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

6.24 command_pf_s Struct Reference

Public Attributes

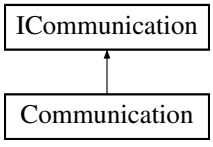
- char const * **flag**
- bool(* **checker**)(const char *, const char *, params_t *)

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

- server/include/zappy.h

6.25 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

Public Member Functions inherited from ICommunication

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'

6.25.1 Member Function Documentation

6.25.1.1 disconnect()

```
void Communication::disconnect ()  [override], [virtual]
```

Implements ICommunication.

6.25.1.2 hasMessages()

```
bool Communication::hasMessages () const  [override], [virtual]
```

Implements ICommunication.

6.25.1.3 isConnected()

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

6.25.1.4 popMessage()

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

6.25.1.5 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

6.26 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)
- None **addResponse** (self, str response)
- bool **hasResponses** (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)
- str **getLastResponse** (self)
- bool **playerIsDead** (self)
- bool **hasPendingCommands** (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

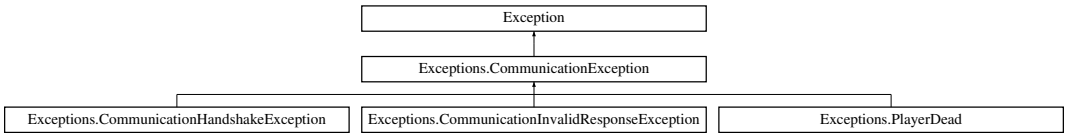
- list **requestQueue** = [ ]
- list **pendingQueue** = [ ]
- list **responseQueue** = [ ]
- str **responseBuffer** = ""
- list **messageQueue** = [ ]
- bool **playerDead** = False
- dict **lastInventory** = {}
- list **lastLook** = [ ]
- **name** = name
- **host** = host
- **port** = port
- **socket** = Socket(host, port)
- **mutex** = threading.Lock()
- list[dict[str, int]]|None **lastInventory** = self.tryGetLook(response)
- None **responseBuffer** = self.socket.receive()

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

- ai/src/Communication/Communication.py

6.27 Exceptions.CommunicationException Class Reference

Inheritance diagram for Exceptions.CommunicationException:



Public Member Functions

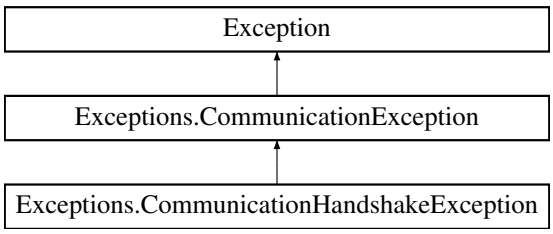
- **__init__** (self, str message)

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

- ai/src/Exceptions/Exceptions.py

6.28 Exceptions.CommunicationHandshakeException Class Reference

Inheritance diagram for Exceptions.CommunicationHandshakeException:



Public Member Functions

- __init__ (self, str message)

Public Member Functions inherited from Exceptions.CommunicationException

6.28.1 Constructor & Destructor Documentation

6.28.1.1 __init__()

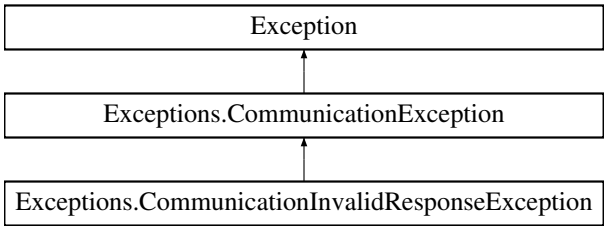
```
Exceptions.CommunicationHandshakeException.__init__ (
             self,
            str message)
```

Reimplemented from Exceptions.CommunicationException.
The documentation for this class was generated from the following file:

- ai/src/Exceptions/Exceptions.py

6.29 Exceptions.CommunicationInvalidResponseException Class Reference

Inheritance diagram for Exceptions.CommunicationInvalidResponseException:



Public Member Functions

- __init__ (self, str message)

Public Member Functions inherited from Exceptions.CommunicationException

6.29.1 Constructor & Destructor Documentation

6.29.1.1 __init__()

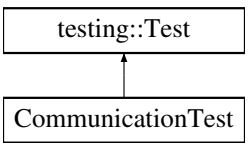
```
Exceptions.CommunicationInvalidResponseException.__init__ (
             self,
            str message)
```

Reimplemented from Exceptions.CommunicationException.
The documentation for this class was generated from the following file:

- ai/src/Exceptions/Exceptions.py

6.30 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

6.31 zappy::structs::Config Struct Reference

Public Attributes

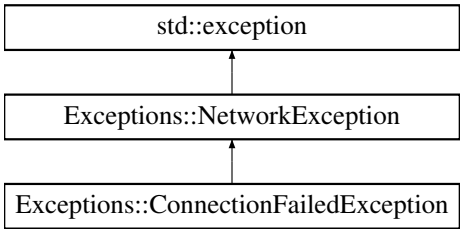
- int **port**
- std::string **hostname**

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

- gui/src/Utils/Constants.hpp

6.32 Exceptions::ConnectionFailedException Class Reference

Inheritance diagram for Exceptions::ConnectionFailedException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

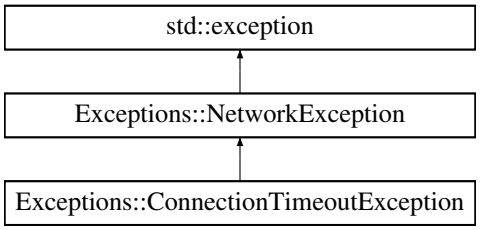
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.33 Exceptions::ConnectionTimeoutException Class Reference

Inheritance diagram for Exceptions::ConnectionTimeoutException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

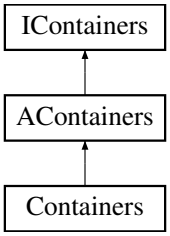
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.34 Containers Class Reference

Container class for organizing UI elements.
`#include <Containers.hpp>`
Inheritance diagram for Containers:



Public Member Functions

- Containers (std::shared_ptr< RayLib > raylib, std::shared_ptr< Audio > audio, float x, float y, float width, float height, Color backgroundColor={40, 40, 40, 200})

  Construct a new Container.
- ~**Containers** () override

  Destroy the Container.
- void draw () override

  Draw the container and its contents.
- void update () override

  Update the container state and its contents.
- void setBackgroundColor (Color color)

  Set the background color.
- void setHasBackground (bool hasBackground)

  Set whether to draw the background.

- void setBackgroundTexture (Texture2D texture)

  Set background texture for the container.
- bool hasBackgroundTexture () const

  Check if the container has a background texture.
- bool addElement (const std::string &id, std::shared_ptr< IUIElement > element)

  Add a UI element to the container.
- std::shared_ptr< IUIElement > getElement (const std::string &id) const

  Get a UI element by its ID.
- bool removeElement (const std::string &id)

  Remove a UI element.
- 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)

  Create and add a button to the container.
- 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, Color normalColor, Color hoverColor, Color pressedColor, Color textColor)

  Create and add a button to the container with custom colors.
- std::shared_ptr< Text > addText (const std::string &id, float x, float y, const std::string &text, float fontSize=20.0f, Color color=BLACK)

  Create and add a text element to the container.
- void **clearElements** ()

  Clear all UI elements from the container.
- void handleResize (int oldWidth, int oldHeight, int newWidth, int newHeight)

  Handle window resize event.
- 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)

  Create and add a button to the container using relative percentages.
- 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, Color normalColor, Color hoverColor, Color pressedColor, Color textColor)

  Create and add a button to the container with custom colors using relative percentages.
- std::shared_ptr< Text > addTextPercent (const std::string &id, float xPercent, float yPercent, const std::string &text, float fontSizePercent=5.0f, Color color=BLACK)

  Create and add a text element to the container using relative percentages.

Public Member Functions inherited from AContainers

- AContainers (std::shared_ptr< RayLib > raylib, float x, float y, float width, float height)

  Construct a new AContainers object.
- virtual ~**AContainers** ()=default

  Destroy the AContainers object.
- void setPosition (float x, float y) override

  Set the position of the container.
- void setSize (float width, float height) override

  Set the size of the container.
- Rectangle getBounds () const override

  Get the current position of the container.
- bool contains (float x, float y) const override

  Check if a point is within the container.
- void setVisible (bool visible) override

  Set the visibility of the container.
- bool isVisible () const override

Check if the container is visible.

- void setRelativePosition (float xPercent, float yPercent, float widthPercent, float heightPercent)

  Set position and size as percentages of screen dimensions.
- RelativePosition getRelativePosition () const

  Get the container's relative position.
- void **updatePositionFromRelative** ()

  Update the container's absolute position from relative position.

Public Member Functions inherited from IContainers

Private Attributes

- std::shared_ptr< RayLib > **_raylib**
- std::shared_ptr< Audio > **_audio**
- Texture2D **_backgroundTexture**
- bool **_hasBackgroundTexture**
- std::unordered_map< std::string, std::shared_ptr< IUIElement > > **_elements**

Additional Inherited Members

Protected Attributes inherited from AContainers

- std::shared_ptr< RayLib > **_raylib**
- Rectangle **_bounds**
- RelativePosition **_relativePos**
- Color **_backgroundColor**
- bool **_visible**
- bool **_hasBackground**

6.34.1 Detailed Description

Container class for organizing UI elements.
Containers can hold UI elements like buttons, text, and scrollbars. They can have a background color or texture.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 Containers()

```
Containers::Containers (
            std::shared_ptr< RayLib > raylib,
            std::shared_ptr< Audio > audio,
            float x,
            float y,
            float width,
            float height,
            Color backgroundColor = {40, 40, 40, 200})
```

Construct a new Container.

Parameters

raylib	Reference to the RayLib instance
audio	Reference to the Audio instance
x	X coordinate
y	Y coordinate
width	Container width
height	Container height
backgroundColor	Background color (default: semi-transparent dark gray)

6.34.3 Member Function Documentation

6.34.3.1 addButton() [1/2]

```
std::shared_ptr< Button > Containers::addButton (
            const std::string & id,
            float x,
            float y,
            float width,
            float height,
            const std::string & text,
            std::function< void()> callback)
```

Create and add a button to the container.

Parameters

id	Unique identifier for the button
x	X coordinate relative to container
y	Y coordinate relative to container
width	Button width
height	Button height
text	Button text
callback	Function to call when button is clicked

Returns

std::shared_ptr<Button> Pointer to the created button, or nullptr if ID already exists

6.34.3.2 addButton() [2/2]

```
std::shared_ptr< Button > Containers::addButton (
            const std::string & id,
            float x,
            float y,
            float width,
            float height,
            const std::string & text,
            std::function< void()> callback,
            Color normalColor,
            Color hoverColor,
            Color pressedColor,
            Color textColor)
```

Create and add a button to the container with custom colors.

Parameters

id	Unique identifier for the button
x	X coordinate relative to container
y	Y coordinate relative to container
width	Button width
height	Button height
text	Button text
callback	Function to call when button is clicked
normalColor	Color when not interacting
hoverColor	Color when mouse is hovering over button
pressedColor	Color when button is pressed
textColor	Color of the button text

Returns

std::shared_ptr<Button> Pointer to the created button, or nullptr if ID already exists

6.34.3.3 addButtonPercent() [1/2]

```
std::shared_ptr< Button > Containers::addButtonPercent (
            const std::string & id,
            float xPercent,
            float yPercent,
            float widthPercent,
            float heightPercent,
            const std::string & text,
            std::function< void()> callback)
```

Create and add a button to the container using relative percentages.

Parameters

id	Unique identifier for the button
xPercent	X position as percentage of container width (0-100)
yPercent	Y position as percentage of container height (0-100)
widthPercent	Width as percentage of container width (0-100)
heightPercent	Height as percentage of container height (0-100)
text	Button text
callback	Function to call when button is clicked

Returns

std::shared_ptr<Button> Pointer to the created button, or nullptr if ID already exists

6.34.3.4 addButtonPercent() [2/2]

```
std::shared_ptr< Button > Containers::addButtonPercent (
            const std::string & id,
            float xPercent,
            float yPercent,
            float widthPercent,
            float heightPercent,
            const std::string & text,
            std::function< void()> callback,
            Color normalColor,
            Color hoverColor,
            Color pressedColor,
            Color textColor)
```

Create and add a button to the container with custom colors using relative percentages.

Parameters

id	Unique identifier for the button
xPercent	X position as percentage of container width (0-100)
yPercent	Y position as percentage of container height (0-100)
widthPercent	Width as percentage of container width (0-100)
heightPercent	Height as percentage of container height (0-100)

Parameters

text	Button text
callback	Function to call when button is clicked
normalColor	Color when not interacting
hoverColor	Color when mouse is hovering over button
pressedColor	Color when button is pressed
textColor	Color of the button text

Returns

std::shared_ptr<Button> Pointer to the created button, or nullptr if ID already exists

6.34.3.5 addElement()

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

Add a UI element to the container.

Parameters

id	Unique identifier for the element
element	UI element to add

Returns

true If element was added successfully

false If element with same ID already exists

6.34.3.6 addText()

```
std::shared_ptr< Text > Containers::addText (
            const std::string & id,
            float x,
            float y,
            const std::string & text,
            float fontSize = 20.0f,
            Color color = BLACK)
```

Create and add a text element to the container.

Parameters

id	Unique identifier for the text element
x	X coordinate relative to container
y	Y coordinate relative to container
text	Text content
fontSize	Font size
color	Text color

Returns

std::shared_ptr<Text> Pointer to the created text element, or nullptr if ID already exists

6.34.3.7 addTextPercent()

```
std::shared_ptr< Text > Containers::addTextPercent (
            const std::string & id,
            float xPercent,
            float yPercent,
            const std::string & text,
            float fontSizePercent = 5.0f,
            Color color = BLACK)
```

Create and add a text element to the container using relative percentages.

Parameters

id	Unique identifier for the text element
xPercent	X position as percentage of container width (0-100)
yPercent	Y position as percentage of container height (0-100)
text	Text content
fontSizePercent	Font size as percentage of container height (0-100)
color	Text color

Returns

std::shared_ptr<Text> Pointer to the created text element, or nullptr if ID already exists

6.34.3.8 draw()

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

Draw the container and its contents.
Implements IContainers.

6.34.3.9 getElement()

```
std::shared_ptr< IUIElement > Containers::getElement (
            const std::string & id) const
```

Get a UI element by its ID.

Parameters

id	Element identifier

Returns

std::shared_ptr<IUIElement> Pointer to the element, or nullptr if not found

6.34.3.10 handleResize()

```
void Containers::handleResize (
            int oldWidth,
            int oldHeight,
            int newWidth,
            int newHeight)
```

Handle window resize event.

Parameters

oldWidth	Previous window width
oldHeight	Previous window height

newWidth	New window width
newHeight	New window height

6.34.3.11 hasBackgroundTexture()

```
bool Containers::hasBackgroundTexture () const
```

Check if the container has a background texture.

Returns

true If the container has a background texture

false Otherwise

6.34.3.12 removeElement()

```
bool Containers::removeElement (
            const std::string & id)
```

Remove a UI element.

Parameters

id	Element identifier

Returns

true If element was found and removed

false If element was not found

6.34.3.13 setBackgroundColor()

```
void Containers::setBackgroundColor (
            Color color)
```

Set the background color.

Parameters

color	New background color

6.34.3.14 setBackgroundTexture()

```
void Containers::setBackgroundTexture (
            Texture2D texture)
```

Set background texture for the container.

Parameters

texture	Texture to use as background

6.34.3.15 setHasBackground()

```
void Containers::setHasBackground (
            bool hasBackground)
```

Set whether to draw the background.

Parameters

hasBackground	True to draw background, false otherwise

6.34.3.16 update()

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

Update the container state and its contents.
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

6.35 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

6.36 egg_s Struct Reference

Public Attributes

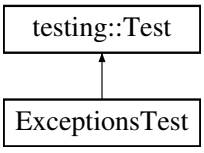
- int **id**
- int **posX**
- int **posY**
- char * **teamName**
- int **idLayer**
- bool **isHatched**
- struct egg_s * **next**

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

- server/include/game.h

6.37 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

6.38 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

6.39 GameInfos Class Reference

Public Member Functions

- void **setMapSize** (int width, int height)
- std::pair< int, int > **getMapSize** () const
- void **setTimeUnit** (int timeUnit)
- int **getTimeUnit** () const
- void **updateTile** (const zappy::structs::Tile tile)
- const std::vector< zappy::structs::Tile > **getTiles** () const
- const zappy::structs::Tile **getTile** (int x, int y) const
- void **updateTeamName** (const std::string &teamName)
- const std::vector< std::string > **getTeamNames** () const
- void **addPlayer** (const zappy::structs::Player player)
- 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
- void **addPlayerBroadcast** (int playerNumber, const std::string &message)
- std::vector< std::pair< int, std::string > > **getPlayersBroadcasting** () const
- void **addIncantation** (const zappy::structs::Incantation incantation)
- void **removeIncantation** (int x, int y, int result)
- 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)
- std::pair< bool, std::string > **isGameOver** () const

Private Attributes

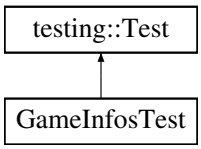
- int **_mapWidth**
- int **_mapHeight**
- int **_timeUnit**
- std::vector< zappy::structs::Tile > **_tiles**
- std::vector< std::string > **_teamNames**
- std::vector< zappy::structs::Player > **_players**
- std::vector< std::pair< int, bool > > **_playersExpulsing**
- std::vector< std::pair< int, std::string > > **_playersBroadcasting**
- std::vector< zappy::structs::Incantation > **_incantations**
- std::vector< zappy::structs::Egg > **_eggs**
- bool **_gameOver**
- std::string **_winningTeam**
- std::mutex **_dataMutex**

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

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

6.40 GameInfosTest Class Reference

Inheritance diagram for GameInfosTest:



Protected Member Functions

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

Protected Attributes

- std::unique_ptr< GameInfos > **gameInfos**

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

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

6.41 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

6.42 GUI Class Reference

Public Member Functions

- **GUI** (std::shared_ptr< GameInfos > gameInfos)
- void **run** ()
- 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** ()
- void **update** ()
- void **draw** ()
- bool **playerExists** (int playerId) const
- void **initModels** ()

Private Attributes

- bool **_isRunning**
- std::shared_ptr< RayLib > **_raylib**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::unique_ptr< Map > **_map**
- std::unique_ptr< HUD > **_hud**
- std::shared_ptr< Audio > **_audio**
- std::unique_ptr< CameraManager > **_cameraManager**
- int **_windowWidth**
- int **_windowHeight**
- zappy::gui::CameraMode **_cameraMode**

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

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

6.43 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

- int **key** = sum((i + 1) * ord(c) for i, c in enumerate(hash_key)) % 256

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

- ai/src/Hash/Hash.py

6.44 HUD Class Reference

Main HUD class to manage all UI elements.
`#include <HUD.hpp>`

Public Member Functions

- HUD (std::shared_ptr< RayLib > raylib, std::shared_ptr< GameInfos > gameInfos, std::shared_ptr< Audio > audio)

  Construct a new HUD object.
- ~**HUD** ()

  Destroy the HUD object.
- void **draw** ()

  Draw all visible HUD elements.
- void **update** ()

  Update all HUD elements.
- std::shared_ptr< Containers > addContainer (const std::string &id, float x, float y, float width, float height, Color backgroundColor={40, 40, 40, 200})

  Add a new container to the HUD.
- std::shared_ptr< Containers > getContainer (const std::string &id) const

  Get a container by its ID.
- bool removeContainer (const std::string &id)

  Remove a container and all its child elements.
- void handleResize (int oldWidth, int oldHeight, int newWidth, int newHeight)

  Handle window resize event.
- void **clearAllContainers** ()

  Clear all containers from the HUD.
- void initDefaultLayout (float sideWidthPercent=15.0f, float bottomHeightPercent=20.0f)

  Initialize default layout with side and bottom containers.
- std::shared_ptr< Containers > getSideContainer () const

  Get the side container.
- std::shared_ptr< Containers > getBottomContainer () const

  Get the bottom container.
- std::shared_ptr< Containers > getSquareContainer () const

  Get the square container in the top-left corner.
- void initExitButton ()

  Initialize an exit button in the square container.
- void initSettingsButton ()

  Initialize a settings button in the square container.
- void initHelpButton ()

  Initialize a help button in the square container.
- void initCameraResetButton ()

  Initialize a camera reset button in the square container.
- void initTeamPlayersDisplay (std::shared_ptr< GameInfos > gameInfos)

  Initialize team and player display in the side container.
- void updateTeamPlayersDisplay (std::shared_ptr< GameInfos > gameInfos)

  Update team and player display in the side container.

Private Member Functions

- std::shared_ptr< Containers > createSquareContainer (float squareSize, float sideWidthPercent)

  Create the square container in the top-left corner.
- std::shared_ptr< Containers > createSideContainer (float sideYStart, float sideWidth, float sideHeight, float sideWidthPercent, float bottomHeightPercent)

  Create the side container for team information.
- std::shared_ptr< Containers > createBottomContainer (int screenWidth, int screenHeight, float bottom↩ Height, float bottomHeightPercent)

  Create the bottom container.
- void recordElementPositions (std::shared_ptr< Containers > container, std::unordered_map< std::string, float > &initialYPositions, float &lastContainerHeight)

  Record element positions for scrolling.
- void updateElementPositions (std::shared_ptr< Containers > container, const std::unordered_map< std↩ ::string, float > &initialYPositions, float offset)

  Update elements positions based on scroll value.
- std::pair< float, float > calculateContentMetrics (std::shared_ptr< Containers > container, const std↩ ::unordered_map< std::string, float > &initialYPositions)

  Calculate content height and scroll distance.
- void clearTeamDisplayElements (std::shared_ptr< Containers > container)

  Clear all team display elements from the container.
- std::vector< int > getTeamPlayerNumbers (const std::string &teamName, const std::vector< zappy::structs::Player > &players)

  Get player numbers for a specific team.
- std::string createPlayerListText (const std::vector< int > &playerNumbers)

  Create player list text representation.
- void addPlayerListText (std::shared_ptr< Containers > container, const std::string &teamId, float yPos, const std::vector< int > &playerNumbers)

  Add player list text to the container.

Private Attributes

- std::unordered_map< std::string, std::shared_ptr< Containers > > **_containers**
- std::shared_ptr< RayLib > **_raylib**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< Audio > **_audio**

6.44.1 Detailed Description

Main HUD class to manage all UI elements.
This class handles the creation, management, and rendering of all UI containers and their elements like buttons, text, scrollbars, etc.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 HUD()

```
HUD::HUD (
            std::shared_ptr< RayLib > raylib,
            std::shared_ptr< GameInfos > gameInfos,
            std::shared_ptr< Audio > audio)
```

Construct a new HUD object.

Parameters

raylib	Reference to the RayLib instance

6.44.3 Member Function Documentation

6.44.3.1 addContainer()

```
std::shared_ptr< Containers > HUD::addContainer (
            const std::string & id,
            float x,
            float y,
            float width,
            float height,
            Color backgroundColor = {40, 40, 40, 200})
```

Add a new container to the HUD.

Parameters

id	Unique identifier for the container
x	X coordinate
y	Y coordinate
width	Container width
height	Container height
backgroundColor	Background color (optional)

Returns

std::shared_ptr<Containers> Pointer to the created container

6.44.3.2 addPlayerListText()

```
void HUD::addPlayerListText (
            std::shared_ptr< Containers > container,
            const std::string & teamId,
            float yPos,
            const std::vector< int > & playerNumbers)  [private]
```

Add player list text to the container.

Parameters

container	Container to add text to
teamId	Team identifier
yPos	Y position percentage
playerNumbers	List of player numbers

6.44.3.3 calculateContentMetrics()

```
std::pair< float, float > HUD::calculateContentMetrics (
            std::shared_ptr< Containers > container,
            const std::unordered_map< std::string, float > & initialYPositions)  [private]
```

Calculate content height and scroll distance.

Parameters

container	The container
initialYPositions	Map with initial positions

Returns

std::pair<float, float> First: content height, Second: team count

6.44.3.4 clearTeamDisplayElements()

```
void HUD::clearTeamDisplayElements (
            std::shared_ptr< Containers > container)  [private]
```

Clear all team display elements from the container.

Parameters

container	Container to clear elements from

6.44.3.5 createBottomContainer()

```
std::shared_ptr< Containers > HUD::createBottomContainer (
            int screenWidth,
            int screenHeight,
            float bottomHeight,
            float bottomHeightPercent)  [private]
```

Create the bottom container.

Parameters

screenWidth	Width of the screen
screenHeight	Height of the screen
bottomHeight	Height of the bottom container
bottomHeightPercent	Height as percentage of screen height

Returns

std::shared_ptr<Containers> The created container

6.44.3.6 createPlayerListText()

```
std::string HUD::createPlayerListText (
            const std::vector< int > & playerNumbers)  [private]
```

Create player list text representation.

Parameters

playerNumbers	List of player numbers

Returns

std::string Formatted string representation of players

6.44.3.7 createSideContainer()

```
std::shared_ptr< Containers > HUD::createSideContainer (
            float sideYStart,
            float sideWidth,
            float sideHeight,
            float sideWidthPercent,
            float bottomHeightPercent)  [private]
```

Create the side container for team information.

Parameters

sideYStart	Y coordinate start position
sideWidth	Width of the side container
sideHeight	Height of the side container
sideWidthPercent	Width as percentage of screen width
bottomHeightPercent	Height of bottom as percentage of screen height

Returns

std::shared_ptr<Containers> The created container

6.44.3.8 createSquareContainer()

```
std::shared_ptr< Containers > HUD::createSquareContainer (
            float squareSize,
            float sideWidthPercent)  [private]
```

Create the square container in the top-left corner.

Parameters

squareSize	Size of the square
sideWidthPercent	Width as percentage of screen width

Returns

std::shared_ptr<Containers> The created container

6.44.3.9 getBottomContainer()

```
std::shared_ptr< Containers > HUD::getBottomContainer () const
```

Get the bottom container.

Returns

std::shared_ptr<Containers> Pointer to the bottom container

6.44.3.10 getContainer()

```
std::shared_ptr< Containers > HUD::getContainer (
            const std::string & id) const
```

Get a container by its ID.

Parameters

id	Container identifier

Returns

std::shared_ptr<Containers> Pointer to the container, or nullptr if not found

6.44.3.11 getSideContainer()

```
std::shared_ptr< Containers > HUD::getSideContainer () const
```

Get the side container.

Returns

std::shared_ptr<Containers> Pointer to the side container

6.44.3.12 getSquareContainer()

```
std::shared_ptr< Containers > HUD::getSquareContainer () const
```

Get the square container in the top-left corner.

Returns

std::shared_ptr<Containers> Pointer to the square container

6.44.3.13 getTeamPlayerNumbers()

```
std::vector< int > HUD::getTeamPlayerNumbers (
            const std::string & teamName,
            const std::vector< zappy::structs::Player > & players)  [private]
```

Get player numbers for a specific team.

Parameters

teamName	Team name to filter players
players	List of all players

Returns

std::vector<int> List of player numbers belonging to the team

6.44.3.14 handleResize()

```
void HUD::handleResize (
            int oldWidth,
            int oldHeight,
            int newWidth,
            int newHeight)
```

Handle window resize event.
Updates all containers and UI elements to adjust to the new window size

Parameters

oldWidth	Previous window width
oldHeight	Previous window height
newWidth	New window width
newHeight	New window height

6.44.3.15 initCameraResetButton()

```
void HUD::initCameraResetButton ()
```

Initialize a camera reset button in the square container.
Creates a button that resets the camera position when clicked

6.44.3.16 initDefaultLayout()

```
void HUD::initDefaultLayout (
            float sideWidthPercent = 15.0f,
            float bottomHeightPercent = 20.0f)
```

Initialize default layout with side and bottom containers.
Creates and adds default containers for the left side and bottom of the screen

Parameters

sideWidth	Width of the side container (default: 250 pixels)
bottomHeight	Height of the bottom container (default: 200 pixels)

Initialize default layout with side and bottom containers

Parameters

sideWidthPercent	Width of side container as percentage of screen width (default: 15%)
bottomHeightPercent	Height of bottom container as percentage of screen height (default: 20%)

6.44.3.17 initExitButton()

```
void HUD::initExitButton ()
```

Initialize an exit button in the square container.
Creates a button that closes the application when clicked

6.44.3.18 initHelpButton()

```
void HUD::initHelpButton ()
```

Initialize a help button in the square container.
Creates a button that opens the help menu when clicked

6.44.3.19 initSettingsButton()

```
void HUD::initSettingsButton ()
```

Initialize a settings button in the square container.
Creates a button that opens the settings menu when clicked

6.44.3.20 initTeamPlayersDisplay()

```
void HUD::initTeamPlayersDisplay (
            std::shared_ptr< GameInfos > gameInfos)
```

Initialize team and player display in the side container.
Creates text elements to show teams and their players

Parameters

gameInfos	The game information containing teams and players

6.44.3.21 recordElementPositions()

```
void HUD::recordElementPositions (
            std::shared_ptr< Containers > container,
            std::unordered_map< std::string, float > & initialYPositions,
            float & lastContainerHeight)  [private]
```

Record element positions for scrolling.

Parameters

container	The container with elements
initialYPositions	Map to store initial positions
lastContainerHeight	Last container height for comparison

6.44.3.22 removeContainer()

```
bool HUD::removeContainer (
            const std::string & id)
```

Remove a container and all its child elements.

Parameters

id	Container identifier

Returns

true If container was found and removed

false If container was not found

6.44.3.23 updateElementPositions()

```
void HUD::updateElementPositions (
            std::shared_ptr< Containers > container,
            const std::unordered_map< std::string, float > & initialYPositions,
            float offset)  [private]
```

Update elements positions based on scroll value.

Parameters

container	The container with elements
initialYPositions	Map with initial positions
offset	Scroll offset to apply

6.44.3.24 updateTeamPlayersDisplay()

```
void HUD::updateTeamPlayersDisplay (
            std::shared_ptr< GameInfos > gameInfos)
```

Update team and player display in the side container.
Updates the text elements showing teams and their players

Parameters

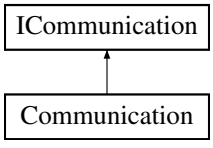
gameInfos	The game information containing teams and players

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

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

6.45 ICommunication Class Reference

Inheritance diagram for ICommunication:



Public Member Functions

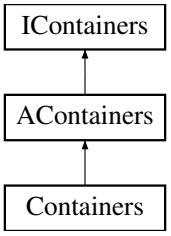
- virtual void **sendMessage** (const std::string &message)=0
- virtual bool **hasMessages** () const =0
- virtual std::string **popMessage** ()=0
- virtual bool **isConnected** () const =0
- virtual void **disconnect** ()=0

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

- gui/src/Communication/ICommunication.hpp

6.46 IContainers Class Reference

Interface for HUD containers.
`#include <IContainers.hpp>`
Inheritance diagram for IContainers:



Public Member Functions

- virtual void draw ()=0

  Draw the container and its contents.
- virtual void update ()=0

  Update the container's state.
- virtual void setPosition (float x, float y)=0

  Set the position of the container.
- virtual void setSize (float width, float height)=0

  Set the size of the container.
- virtual Rectangle getBounds () const =0

  Get the current position of the container.
- virtual bool contains (float x, float y) const =0

  Check if a point is within the container.
- virtual void setVisible (bool visible)=0

  Set the visibility of the container.
- virtual bool isVisible () const =0

  Check if the container is visible.

6.46.1 Detailed Description

Interface for HUD containers.
Containers are UI elements that can hold and organize other UI elements like buttons, text, scrollbars, etc.

6.46.2 Member Function Documentation

6.46.2.1 contains()

```
virtual bool IContainers::contains (
            float x,
            float y) const  [pure virtual]
```

Check if a point is within the container.

Parameters

x	X coordinate
y	Y coordinate

Returns

true If point is within container

false Otherwise

Implemented in AContainers.

6.46.2.2 draw()

```
virtual void IContainers::draw ()  [pure virtual]
```

Draw the container and its contents.
Implemented in Containers.

6.46.2.3 getBounds()

```
virtual Rectangle IContainers::getBounds () const  [pure virtual]
```

Get the current position of the container.

Returns

Rectangle Containing position and size

Implemented in AContainers.

6.46.2.4 isVisible()

```
virtual bool IContainers::isVisible () const  [pure virtual]
```

Check if the container is visible.

Returns

true If visible

false Otherwise

Implemented in AContainers.

6.46.2.5 setPosition()

```
virtual void IContainers::setPosition (
            float x,
            float y)  [pure virtual]
```

Set the position of the container.

Parameters

x	X coordinate
y	Y coordinate

Implemented in AContainers.

6.46.2.6 setSize()

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

Set the size of the container.

Parameters

width	Container width
height	Container height

Implemented in AContainers.

6.46.2.7 setVisible()

```
virtual void IContainers::setVisible (
            bool visible)  [pure virtual]
```

Set the visibility of the container.

Parameters

visible	Visibility state

Implemented in AContainers.

6.46.2.8 update()

```
virtual void IContainers::update ()  [pure virtual]
```

Update the container's state.
Implemented in Containers.
The documentation for this class was generated from the following file:

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

6.47 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

6.48 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

6.49 inventory_s Struct Reference

Public Attributes

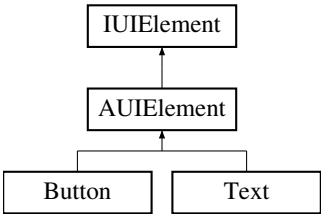
- int **nbFood**
- int **nbLinemate**
- int **nbDeraumere**
- int **nbSibur**
- int **nbMendiane**
- int **nbPhiras**
- int **nbThystame**

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

- server/include/game.h

6.50 IUIElement Class Reference

Interface for all UI elements.
`#include <IUIElement.hpp>`
Inheritance diagram for IUIElement:



Public Member Functions

- virtual void draw ()=0

  Draw the UI element.
- virtual void update ()=0

  Update the UI element's state.
- virtual void setPosition (float x, float y)=0

  Set the position of the UI element.
- virtual void setSize (float width, float height)=0

  Set the size of the UI element.
- virtual Rectangle getBounds () const =0

  Get the bounds of the UI element.
- virtual bool contains (float x, float y) const =0

  Check if the UI element contains a point.
- virtual void setVisible (bool visible)=0

  Set the visibility of the UI element.
- virtual bool isVisible () const =0

  Check if the UI element is visible.

6.50.1 Detailed Description

Interface for all UI elements.
Base interface that all UI elements (buttons, text, scrollbars, etc.) must implement

6.50.2 Member Function Documentation

6.50.2.1 contains()

```
virtual bool IUIElement::contains (
            float x,
            float y) const  [pure virtual]
```

Check if the UI element contains a point.

Parameters

x	X coordinate
y	Y coordinate

Returns

true If the point is inside the element

false Otherwise

Implemented in AUIElement.

6.50.2.2 draw()

```
virtual void IUIElement::draw ()  [pure virtual]
```

Draw the UI element.
Implemented in Button, and Text.

6.50.2.3 getBounds()

```
virtual Rectangle IUIElement::getBounds () const  [pure virtual]
```

Get the bounds of the UI element.

Returns

Rectangle The bounds of the element

Implemented in AUIElement.

6.50.2.4 isVisible()

```
virtual bool IUIElement::isVisible () const  [pure virtual]
```

Check if the UI element is visible.

Returns

true If visible

false Otherwise

Implemented in AUIElement.

6.50.2.5 setPosition()

```
virtual void IUIElement::setPosition (
            float x,
            float y)  [pure virtual]
```

Set the position of the UI element.

Parameters

x	X coordinate
y	Y coordinate

Implemented in AUIElement.

6.50.2.6 setSize()

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

Set the size of the UI element.

Parameters

width	New width
height	New height

Implemented in AUIElement, Button, and Text.

6.50.2.7 setVisible()

```
virtual void IUIElement::setVisible (
            bool visible)  [pure virtual]
```

Set the visibility of the UI element.

Parameters

visible	Visibility state

Implemented in AUIElement.

6.50.2.8 update()

```
virtual void IUIElement::update ()  [pure virtual]
```

Update the UI element's state.
Implemented in Button, and Text.
The documentation for this class was generated from the following file:

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

6.51 Map Class Reference

Public Member Functions

- **Map** (std::shared_ptr< GameInfos > gameInfos, std::shared_ptr< RayLib > raylib)
- void **draw** ()
- void **drawTile** (int x, int y, const zappy::structs::Tile &tile)
- void **drawRock** (int x, int y, const zappy::structs::Tile &tile)
- void **drawFood** (int x, int y, const zappy::structs::Tile &tile)
- void **drawPlayers** (int x, int y)
- void **drawEggs** (int x, int y)
- Color **getTeamColor** (const std::string &teamName)
- float **getOffset** (DisplayPriority priority, int x, int y, size_t stackIndex=0)

Private Member Functions

- void **drawOrientationArrow** (const Vector3 &position, int orientation, float playerHeight)

Private Attributes

- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< RayLib > **_raylib**
- std::unordered_map< std::string, Color > **_teamColors**

Static Private Attributes

- static constexpr float **BASE_HEIGHT_TILE** = 0.0f
- static constexpr float **BASE_HEIGHT_FOOD** = 0.2f
- static constexpr float **BASE_HEIGHT_ROCK** = 0.2f
- static constexpr float **BASE_HEIGHT_EGG** = 0.2f
- static constexpr float **BASE_HEIGHT_PLAYER** = 0.2f
- static constexpr float **FOOD_HEIGHT** = 0.3f
- static constexpr float **ROCK_HEIGHT** = 0.3f
- static constexpr float **EGG_HEIGHT** = 0.3f
- static constexpr float **PLAYER_HEIGHT** = 1.1f

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

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

6.52 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

6.53 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

6.54 RayLib::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/Graphic/RayLib/RayLib.hpp

6.55 MsgHandler Class Reference

Public Member Functions

- **MsgHandler** (std::shared_ptr< GameInfos > gameInfos, std::shared_ptr< ICommunication > communication)
- void **start** ()
- void **stop** ()

Protected Member Functions

- void **messageLoop** ()
- void **handleMessage** (const std::string &message)
- 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

6.56 network_s Struct Reference

Public Attributes

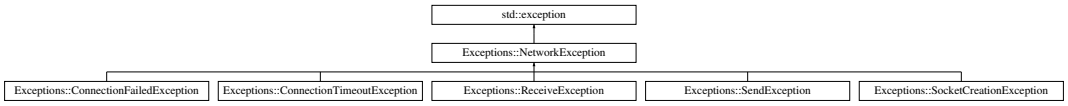
- int **fd**
- buffer_t * **buffer**

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

- server/include/game.h

6.57 Exceptions::NetworkException Class Reference

Inheritance diagram for Exceptions::NetworkException:



Public Member Functions

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

Private Attributes

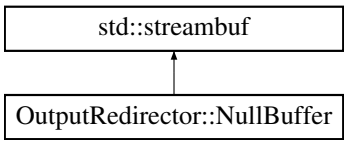
- std::string **_message**

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

- gui/src/Exceptions/Exceptions.hpp

6.58 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

6.59 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

6.60 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

6.61 Parser.Parser Class Reference

Public Member Functions

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

Public Attributes

- str **tests_folder** = ""
- list **tests_files_names** = [ ]
- list **tests_files** = [ ]
- str **output_folder** = ""
- list **testsObjects** = [ ]
- str **tests_files_names** = self.tests_folder + f + ".json"

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

- tests/functional/Parser.py

6.62 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] **getNeededStonesByPriority** (self)
- None **roombaAction** (self)
- None **handleCommandResponse** (self, str response)
- None **loop** (self)

Public Attributes

- Communication **communication** = Communication(name, ip, port)
- list **childs** = [ ]
- str **teamName** = name
- str **ip** = ip
- int **port** = port
- int **level** = 1
- Hash **hash** = Hash(name)
- dict inventory
- list **look** = [ ]
- bool **inIncantation** = False
- int **x** = 0
- int **y** = 0
- dict roombaState
- list[str] **look** = self.getNeededStonesByPriority()
- Communication **inventory** = "look":
- Communication **look** = "ko":

Protected Attributes

- Thread _commThread

6.62.1 Member Data Documentation

6.62.1.1 _commThread

```
Thread Player.Player._commThread  [protected]
```

Initial value:

```
=  Thread(
            target=self.communication.loop,
            name=f"CommunicationThread-{name}"
        )
```

6.62.1.2 inventory

```
dict Player.Player.inventory
```

Initial value:

```
=  {
            "food": 10,
            "linemate": 0,
            "deraumere": 0,
            "sibur": 0,
            "mendiane": 0,
            "phiras": 0,
            "thystame": 0
        }
```

6.62.1.3 roombaState

```
dict Player.Player.roombaState
```

Initial value:

```
=  {
            "forwardCount": 0,
            "targetForward": 10,
            "phase": "forward",
            "lastCommand": None
        }
```

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

- ai/src/Player/Player.py

6.63 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

6.64 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**

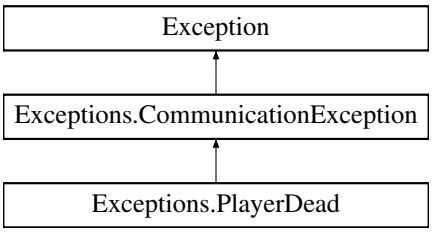
- bool **is_busy**
- int **remaining_cooldown**
- struct player_s * **next**

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

- server/include/game.h

6.65 Exceptions.PlayerDead Class Reference

Inheritance diagram for Exceptions.PlayerDead:



Public Member Functions

- __init__ (self)

Public Member Functions inherited from Exceptions.CommunicationException

6.65.1 Constructor & Destructor Documentation

6.65.1.1 __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

6.66 RayLib 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
- void **drawTextureRec** (Texture2D texture, Rectangle source, Vector2 position, Color tint)
- void **unloadTexture** (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
- 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** ()
- 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
- void **drawRectangleRec** (Rectangle rec, Color color)
- void **drawText** (const std::string &text, float x, float y, float fontSize, Color color)
- float **measureText** (const std::string &text, float fontSize) const
- void **initAudioDevice** ()
- void **closeAudioDevice** ()
- bool **isAudioDeviceReady** () const

Private Attributes

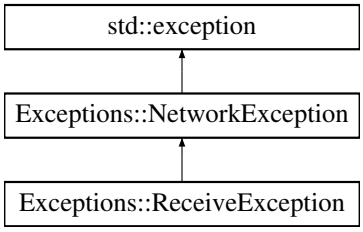
- bool **_isInitialized**
- Camera3D **_camera**
- Vector2 **_previousMousePosition**
- bool **_isCursorLocked**
- std::map< std::string, ModelData > **_models**
- std::map< std::string, Sound > **_sounds**
- std::map< std::string, Music > **_musics**

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

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

6.67 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

6.68 RelativePosition Struct Reference

Structure to store relative positions and sizes as percentages.
`#include <AContainers.hpp>`

Public Attributes

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

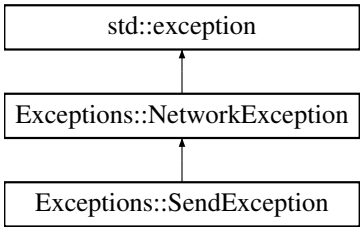
6.68.1 Detailed Description

Structure to store relative positions and sizes as percentages.
The documentation for this struct was generated from the following file:

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

6.69 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

6.70 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

6.71 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

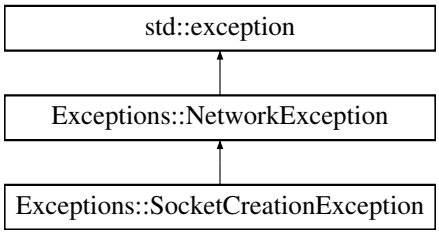
- **_host** = host
- **_port** = port
- tuple **_address** = (host, port)
- **_socket** = None

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

- ai/src/Communication/Socket.py

6.72 Exceptions::SocketCreationException Class Reference

Inheritance diagram for Exceptions::SocketCreationException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

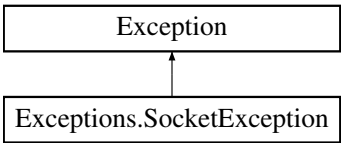
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

6.73 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

6.74 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

6.75 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** = name
- **desc** = desc
- **input** = input
- **output** = output
- **value** = value
- bool **tty_mode** = False
- list **tty_input** = [ ]
- **succeed_after** = None
- bool **succeed_forced** = False
- **real_output** = None
- int **real_value** = None
- **raw_output** = None

Protected Member Functions

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

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

- tests/functional/TestCase.py

6.76 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)

6.76.1 Member Function Documentation

6.76.1.1 test_parse_args_invalid_option()

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

Test parsing invalid option

6.76.1.2 test_parse_args_missing_value()

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

Test parsing missing value for option

6.76.1.3 test_parse_args_not_enough_args()

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

Test parsing not enough arguments

6.76.1.4 test_parse_args_valid()

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

Test parsing valid command line arguments

6.76.1.5 test_parse_args_valid_ip()

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

Test parsing valid IP address

6.76.1.6 test_parse_machine_empty()

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

Test parsing empty machine name

6.76.1.7 test_parse_machine_invalid_ip_chars()

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

Test parsing IP with invalid characters

6.76.1.8 test_parse_machine_invalid_ip_format()

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

Test parsing invalid IP format

6.76.1.9 test_parse_machine_invalid_ip_value()

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

Test parsing invalid IP value

6.76.1.10 test_parse_name_empty()

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

Test parsing empty team name

6.76.1.11 test_parse_name_whitespace()

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

Test parsing whitespace team name

6.76.1.12 test_parse_port_invalid()

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

Test parsing invalid port

6.76.1.13 test_parse_port_negative()

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

Test parsing negative port

6.76.1.14 test_parse_port_too_large()

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

Test parsing port that is too large

6.76.1.15 test_validate_config_missing_name()

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

Test validating config with missing name

6.76.1.16 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

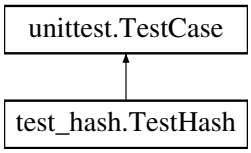
6.77 test_com.TestCommunication Class Reference

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

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

6.78 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** = Hash("test_key")

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

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

6.79 test_player.TestPlayer Class Reference

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

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

6.80 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)

6.80.1 Member Function Documentation

6.80.1.1 test_socket_close()

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

Test socket close

6.80.1.2 test_socket_connect_failure()

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

Test socket connection failure

6.80.1.3 test_socket_connect_success()

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

Test successful socket connection

6.80.1.4 test_socket_different_hosts_and_ports()

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

Test socket creation with different hosts and ports

6.80.1.5 test_socket_init()

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

Test socket initialization

6.80.1.6 test_socket_receive_connection_closed()

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

Test handling closed connection during receive

6.80.1.7 test_socket_receive_unicode()

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

Test receiving unicode messages

6.80.1.8 test_socket_send_success()

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

Test successful message sending

6.80.1.9 test_socket_send_unicode()

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

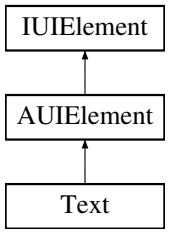
Test sending unicode messages

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

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

6.81 Text Class Reference

Text UI element.
`#include <Text.hpp>`
Inheritance diagram for Text:



Public Member Functions

- Text (std::shared_ptr< RayLib > raylib, float x, float y, const std::string &text, float fontSize=20.0f, Color color=BLACK)

  Construct a new Text element.
- ~**Text** () override=default

  Destroy the Text element.

- void draw () override

  Draw the text.
- void update () override

  Update the text (does nothing for text elements)
- void setText (const std::string &text)

  Set the text content.
- std::string getText () const

  Get the text content.
- void setFontSize (float fontSize)

  Set the font size.
- float getFontSize () const

  Get the font size.
- void setColor (Color color)

  Set the text color.
- Color getColor () const

  Get the text color.
- void setSize (float width, float height) override

  Set the size of the text element For text elements, height determines font size and width is calculated based on text content.

Public Member Functions inherited from AUIElement

- AUIElement (std::shared_ptr< RayLib > raylib, float x, float y, float width, float height)

  Construct a new AUIElement object.
- virtual ~**AUIElement** ()=default

  Destroy the AUIElement object.
- void setPosition (float x, float y) override

  Set the position of the UI element.
- Rectangle getBounds () const override

  Get the bounds of the UI element.
- bool contains (float x, float y) const override

  Check if the UI element contains a point.
- void setVisible (bool visible) override

  Set the visibility of the UI element.
- bool isVisible () const override

  Check if the UI element is visible.
- void setRelativePosition (float xPercent, float yPercent, float widthPercent, float heightPercent)

  Set position and size as percentages of parent container.
- UIRelativePosition getRelativePosition () const

  Get the relative position.

Public Member Functions inherited from IUIElement

Private Attributes

- std::string **_text**
- float **_fontSize**
- Color **_color**
- std::shared_ptr< RayLib > **_raylib**

Additional Inherited Members

Protected Attributes inherited from AUIElement

- std::shared_ptr< RayLib > **_raylib**
- Rectangle **_bounds**
- UIRelativePosition **_relativePos**
- bool **_visible**

6.81.1 Detailed Description

Text UI element.
A UI element for rendering text

6.81.2 Constructor & Destructor Documentation

6.81.2.1 Text()

```
Text::Text (
            std::shared_ptr< RayLib > raylib,
            float x,
            float y,
            const std::string & text,
            float fontSize = 20.0f,
            Color color = BLACK)
```

Construct a new Text element.

Parameters

x	X coordinate
y	Y coordinate
text	Text content
fontSize	Font size
color	Text color

6.81.3 Member Function Documentation

6.81.3.1 draw()

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

Draw the text.
Implements IUIElement.

6.81.3.2 getColor()

```
Color Text::getColor () const
```

Get the text color.

Returns

Color Text color

6.81.3.3 getFontSize()

```
float Text::getFontSize () const
```

Get the font size.

Returns

float Font size

6.81.3.4 getText()

```
std::string Text::getText () const
```

Get the text content.

Returns

std::string Text content

6.81.3.5 setColor()

```
void Text::setColor (
            Color color)
```

Set the text color.

Parameters

color	New text color

6.81.3.6 setFontSize()

```
void Text::setFontSize (
            float fontSize)
```

Set the font size.

Parameters

fontSize	New font size

6.81.3.7 setSize()

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

Set the size of the text element For text elements, height determines font size and width is calculated based on text content.

Parameters

width	Desired width (may be adjusted based on text content)
height	Desired height (used as font size)

Reimplemented from AUIElement.

6.81.3.8 setText()

```
void Text::setText (
            const std::string & text)
```

Set the text content.

Parameters

text	New text content

6.81.3.9 update()

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

Update the text (does nothing for text elements)
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

6.82 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

6.83 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

6.84 UIRelativePosition Struct Reference

Structure to store relative positions and sizes as percentages.
`#include <AUIElement.hpp>`

Public Attributes

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

6.84.1 Detailed Description

Structure to store relative positions and sizes as percentages.
The documentation for this struct was generated from the following file:

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

6.85 zappy_s Struct Reference

Public Attributes

- server_t * **network**
- game_t * **game**
- graph_net_t * **graph**
- params_t * **params**

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

- server/include/zappy.h

Chapter 7

File Documentation

7.1 CLI.hpp

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

7.2 Client.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Client
00006 */
00007
00008 #ifndef CLIENT_HPP_
00009 #define CLIENT_HPP_
00010
00011 #include <memory>
00012
00013 #include "../Utils/Constants.hpp"
00014 #include "../Communication/ICommunication.hpp"
00015 #include "../Game/GameInfos.hpp"
00016 #include "../Graphic/GUI.hpp"
00017 #include "MsgHandler.hpp"
00018
00019 class Client {
00020     public:
00021         Client(int ac, const char *const *av);
00022         ~Client();
00023
00024     private:
00025         zappy::structs::Config _config;
```

```
00026         void initialize(int ac, const char * const *av);
00027
00028         std::shared_ptr<ICommunication> _communication;
00029         std::shared_ptr<GameInfos> _gameInfos;
00030         std::unique_ptr<MsgHandler> _msgHandler;
00031         std::unique_ptr<GUI> _gui;
00032 };
00033
00034 #endif /* !CLIENT_HPP_ */
```

7.3 MsgHandler.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** MsgHandler
00006 */
00007
00008 #ifndef MSGHANDLER_HPP_
00009 #define MSGHANDLER_HPP_
00010
00011 #include <memory>
00012 #include <map>
00013 #include <functional>
00014 #include <thread>
00015 #include <mutex>
00016 #include <atomic>
00017 #include <queue>
00018 #include <condition_variable>
00019 #include <string>
00020
00021 #include "../Game/GameInfos.hpp"
00022 #include "../Communication/ICommunication.hpp"
00023 #include "../Utils/Constants.hpp"
00024
00025 class MsgHandler {
00026     public:
00027         MsgHandler(std::shared_ptr<GameInfos> gameInfos,
00028             std::shared_ptr<ICommunication> communication);
00029         ~MsgHandler();
00030
00031         void start();
00032         void stop();
00033
00034     protected:
00035         void messageLoop();
00036
00037         void handleMessage(const std::string& message);
00038         bool handleWelcomeMessage(const std::string& message);
00039         bool handleMszMessage(const std::string& message);
00040         bool handleBctMessage(const std::string& message);
00041         bool handleTnaMessage(const std::string& message);
00042         bool handlePnwMessage(const std::string& message);
00043         bool handlePpoMessage(const std::string& message);
00044         bool handlePlvMessage(const std::string& message);
00045         bool handlePinMessage(const std::string& message);
00046         bool handlePexMessage(const std::string& message);
00047         bool handlePbcMessage(const std::string& message);
00048         bool handlePicMessage(const std::string& message);
00049         bool handlePieMessage(const std::string& message);
00050         bool handlePfkMessage(const std::string& message);
00051         bool handlePdrMessage(const std::string& message);
00052         bool handlePgtMessage(const std::string& message);
00053         bool handlePdiMessage(const std::string& message);
00054         bool handleEnwMessage(const std::string& message);
00055         bool handleEboMessage(const std::string& message);
00056         bool handleEdiMessage(const std::string& message);
00057         bool handleSgtMessage(const std::string& message);
00058         bool handleSstMessage(const std::string& message);
00059         bool handleSegMessage(const std::string& message);
00060         bool handleSmgMessage(const std::string& message);
00061         bool handleSucMessage(const std::string& message);
00062         bool handleSbpMessage(const std::string& message);
00063
00064     private:
00065         std::thread _thread;
00066         std::atomic<bool> _running;
00067         std::mutex _mutex;
00068         std::condition_variable _condition;
00069
00070         std::shared_ptr<GameInfos> _gameInfos;
00071         std::shared_ptr<ICommunication> _communication;
00072         std::mutex _gameInfosMutex;
00073
```

```
00074         std::map<std::string, std::function<bool(const std::string&)» _messageHandlers;
00075 };
00076
00077 #endif /* !MSGHANDLER_HPP_ */
```

7.4 Communication.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Communication
00006 */
00007
00008 #ifndef COMMUNICATION_HPP_
00009 #define COMMUNICATION_HPP_
00010
00011 #include <sys/socket.h>
00012 #include <netinet/in.h>
00013 #include <arpa/inet.h>
00014 #include <unistd.h>
00015 #include <fcntl.h>
00016 #include <poll.h>
00017 #include <netdb.h>
00018 #include <thread>
00019 #include <mutex>
00020 #include <atomic>
00021 #include <condition_variable>
00022 #include <queue>
00023 #include <string>
00024 #include <vector>
00025
00026 #include "../Utils/Constants.hpp"
00027 #include "../Exceptions/Exceptions.hpp"
00028 #include "ICommunication.hpp"
00029
00030 class Communication : public ICommunication {
00031     public:
00032         explicit Communication(zappy::structs::Config config);
00033         ~Communication();
00034
00035         void sendMessage(const std::string &message) override;
00036         bool hasMessages() const override;
00037         std::string popMessage() override;
00038         bool isConnected() const override;
00039         void disconnect() override;
00040
00041     private:
00042         void setupConnection();
00043         void createSocket();
00044         void connectToServer();
00045         void setupNonBlocking();
00046
00047         void startCommunicationThread();
00048         void communicationLoop();
00049         bool handlePoll();
00050         void processWrite();
00051         void processRead();
00052
00053         void parseReceivedData();
00054
00055         zappy::structs::Config _config;
00056         std::thread _thread;
00057         std::mutex _mutex;
00058         std::condition_variable _cv;
00059         std::atomic<bool> _running;
00060         std::atomic<bool> _connected;
00061
00062         std::queue<std::string> _outgoingMessages;
00063         std::queue<std::string> _incomingMessages;
00064
00065         std::string _receiveBuffer;
00066         std::string _sendBuffer;
00067
00068         int _socket;
00069         struct pollfd _pollfd;
00070         static const int BUFFER_SIZE = 4096;
00071         static const int POLL_TIMEOUT = 100;
00072         static const char MESSAGE_DELIMITER = '\n';
00073 };
00074
00075 #endif /* !COMMUNICATION_HPP_ */
```

7.5 ICommunication.hpp

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

7.6 Exceptions.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Exceptions
00006 */
00007
00008 #ifndef EXCEPTIONS_HPP_
00009 #define EXCEPTIONS_HPP_
00010
00011 #include <exception>
00012 #include <string>
00013 #include "../Utils/Constants.hpp"
00014
00015 namespace Exceptions {
00016
00017     // CLI Exceptions
00018     class CLIParsingException : public std::exception {
00019         public:
00020             explicit CLIParsingException(const std::string &message)
00021                 : _message(std::string(colors::T_RED) +
00022                           "CLI Parsing Error: " + message +
00023                           colors::RESET) {}
00024
00025             const char *what() const noexcept override {
00026                 return _message.c_str();
00027             }
00028
00029         private:
00030             std::string _message;
00031     };
00032
00033     class CLIPortException : public CLIParsingException {
00034         public:
00035             explicit CLIPortException(const std::string &message)
00036                 : CLIParsingException(std::string(colors::T_CYAN) +
00037                                       "Port Error: " + message +
00038                                       colors::RESET) {}
00039     };
00040
00041     class CLIHostException : public CLIParsingException {
00042         public:
00043             explicit CLIHostException(const std::string &message)
00044                 : CLIParsingException(std::string(colors::T_CYAN) +
00045                                       "Hostname Error: " + message +
00046                                       colors::RESET) {}
00047     };
00048
00049     class CLIMissingArgumentException : public CLIParsingException {
00050         public:
00051             explicit CLIMissingArgumentException(const std::string &message)
00052                 : CLIParsingException(std::string(colors::T_CYAN) +
00053                                       "Missing Argument: " + message +
00054                                       colors::RESET) {}
00055     };
```

```
00056
00057     class CLIInvalidArgumentException : public CLIParsingException {
00058         public:
00059             explicit CLIInvalidArgumentException(const std::string &message)
00060                 : CLIParsingException(std::string(colors::T_CYAN) +
00061                                       "Invalid Argument: " + message +
00062                                       colors::RESET) {}
00063     };
00064
00065     class NetworkException : public std::exception {
00066         public:
00067             explicit NetworkException(const std::string &message)
00068                 : _message(std::string(colors::T_RED) +
00069                           "Network Error: " + message +
00070                           colors::RESET) {}
00071
00072             const char *what() const noexcept override {
00073                 return _message.c_str();
00074             }
00075
00076         private:
00077             std::string _message;
00078     };
00079
00080     class ConnectionFailedException : public NetworkException {
00081         public:
00082             explicit ConnectionFailedException(const std::string &message)
00083                 : NetworkException(std::string(colors::T_CYAN) +
00084                                    "Connection Failed: " + message +
00085                                    colors::RESET) {}
00086     };
00087
00088     class SocketCreationException : public NetworkException {
00089         public:
00090             explicit SocketCreationException(const std::string &message)
00091                 : NetworkException(std::string(colors::T_CYAN) +
00092                                    "Socket Creation Failed: " + message +
00093                                    colors::RESET) {}
00094     };
00095
00096     class ConnectionTimeoutException : public NetworkException {
00097         public:
00098             explicit ConnectionTimeoutException(const std::string &message)
00099                 : NetworkException(std::string(colors::T_CYAN) +
00100                                    "Connection Timeout: " + message +
00101                                    colors::RESET) {}
00102     };
00103
00104     class SendException : public NetworkException {
00105         public:
00106             explicit SendException(const std::string &message)
00107                 : NetworkException(std::string(colors::T_CYAN) +
00108                                    "Send Error: " + message +
00109                                    colors::RESET) {}
00110     };
00111
00112     class ReceiveException : public NetworkException {
00113         public:
00114             explicit ReceiveException(const std::string &message)
00115                 : NetworkException(std::string(colors::T_CYAN) +
00116                                    "Receive Error: " + message +
00117                                    colors::RESET) {}
00118     };
00119 }
00120
00121 #endif /* !EXCEPTIONS_HPP_ */
```

7.7 GameInfos.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GameInfos
00006 */
00007
00008 #ifndef GAMEINFOS_HPP_
00009 #define GAMEINFOS_HPP_
00010
00011 #include <utility>
00012 #include <vector>
00013 #include <memory>
00014 #include <mutex>
00015 #include <string>
00016
```

```
00017 #include "../Utils/Constants.hpp"
00018
00019 class GameInfos {
00020     public:
00021         GameInfos();
00022         ~GameInfos();
00023
00024         void setMapSize(int width, int height);
00025         std::pair<int, int> getMapSize() const;
00026
00027         void setTimeUnit(int timeUnit);
00028         int getTimeUnit() const;
00029
00030         void updateTile(const zappy::structs::Tile tile);
00031         const std::vector<zappy::structs::Tile> getTiles() const;
00032         const zappy::structs::Tile getTile(int x, int y) const;
00033
00034         void updateTeamName(const std::string &teamName);
00035         const std::vector<std::string> getTeamNames() const;
00036
00037         void addPlayer(const zappy::structs::Player player);
00038         void updatePlayerPosition(int playerNumber, int x, int y);
00039         void updatePlayerOrientation(int playerNumber, int orientation);
00040         void updatePlayerLevel(int playerNumber, int level);
00041         void updatePlayerInventory(int playerNumber,
00042             const zappy::structs::Inventory inventory);
00043         void updatePlayerExpulsion(int playerNumber);
00044         void updatePlayerDeath(int playerNumber);
00045         void updatePlayerResourceAction(int playerNumber, int resourceId, bool isCollecting);
00046         void updatePlayerFork(int playerNumber);
00047         const std::vector<zappy::structs::Player> getPlayers() const;
00048
00049         void addPlayerBroadcast(int playerNumber, const std::string &message);
00050         std::vector<std::pair<int, std::string» getPlayersBroadcasting() const;
00051
00052         void addIncantation(const zappy::structs::Incantation incantation);
00053         void removeIncantation(int x, int y, int result);
00054
00055         void addEgg(const zappy::structs::Egg egg);
00056         void updateEggHatched(int eggNumber);
00057         void updateEggDeath(int eggNumber);
00058         const std::vector<zappy::structs::Egg> getEggs() const;
00059
00060         void setGameOver(const std::string &winningTeam);
00061         std::pair<bool, std::string> isGameOver() const;
00062
00063     private:
00064         int _mapWidth;
00065         int _mapHeight;
00066         int _timeUnit;
00067
00068         std::vector<zappy::structs::Tile> _tiles;
00069         std::vector<std::string> _teamNames;
00070         std::vector<zappy::structs::Player> _players;
00071         std::vector<std::pair<int, bool» _playersExpulsing;
00072         std::vector<std::pair<int, std::string» _playersBroadcasting;
00073         std::vector<zappy::structs::Incantation> _incantations;
00074         std::vector<zappy::structs::Egg> _eggs;
00075
00076         bool _gameOver;
00077         std::string _winningTeam;
00078
00079         mutable std::mutex _dataMutex;
00080 };
00081
00082 #endif /* !GAMEINFOS_HPP_ */
```

7.8 Audio.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Audio
00006 */
00007
00008 #ifndef AUDIO_HPP_
00009 #define AUDIO_HPP_
00010
00011 #include <string>
00012 #include <map>
00013 #include <memory>
00014 #include <SFML/Audio.hpp>
00015
00016 class Audio {
```

```
00017     public:
00018         Audio();
00019         ~Audio();
00020
00021         bool loadSound(const std::string& id, const std::string& filepath);
00022
00023         void playSound(const std::string& id, float volume = 1.0f);
00024         void stopSound(const std::string& id);
00025         bool isSoundPlaying(const std::string& id) const;
00026
00027         void setSoundLooping(const std::string& id, bool looping);
00028         void setSoundVolume(const std::string& id, float volume);
00029
00030     private:
00031         std::map<std::string, std::unique_ptr<sf::Music» _sounds;
00032 };
00033
00034 #endif /* !AUDIO_HPP_ */
```

7.9 CameraManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** CameraManager
00006 */
00007
00008 #ifndef CAMERA_MANAGER_HPP_
00009 #define CAMERA_MANAGER_HPP_
00010
00011 #include <memory>
00012 #include "../RayLib/RayLib.hpp"
00013 #include "../../Utils/Constants.hpp"
00014 #include "../../Game/GameInfos.hpp"
00015 #include "../Map.hpp"
00016
00017 class CameraManager {
00018     public:
00019         explicit CameraManager(std::shared_ptr<RayLib> raylib);
00020         ~CameraManager();
00021
00022         void updateCamera(zappy::gui::CameraMode mode);
00023         void updateCameraFreeMode();
00024         void updateCameraTargetMode();
00025         void updateCameraPlayerMode();
00026
00027         void setMapCenter(const Vector3& center);
00028         void setMapSize(int width, int height);
00029
00030         float getCurrentCameraDistance() const;
00031         void setTargetDistance(float distance);
00032         void initTargetPositionFromCurrentCamera();
00033
00034         void setPlayerId(int playerId);
00035         int getPlayerId() const;
00036         void setGameInfos(std::shared_ptr<GameInfos> gameInfos);
00037         void setMapInstance(std::shared_ptr<Map> map);
00038
00039     private:
00040         std::shared_ptr<RayLib> _raylib;
00041         std::shared_ptr<GameInfos> _gameInfos;
00042         std::shared_ptr<Map> _map;
00043         Vector3 _mapCenter;
00044         int _mapWidth;
00045         int _mapHeight;
00046
00047         float _targetDistance;
00048         float _targetAngleXZ;
00049         float _targetAngleY;
00050         bool _isDragging;
00051         int _playerId;
00052
00053         float _playerAngleXZ;
00054         bool _isPlayerViewDragging;
00055
00056         void handlePlayerCameraMouseInput();
00057         Vector3 calculatePlayerPosition(const zappy::structs::Player& player);
00058         Vector3 calculateCameraPosition(const Vector3& playerPos, float angleXZ);
00059 };
00060
00061 #endif /* !CAMERA_MANAGER_HPP_ */
```

7.10 GUI.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GUI
00006 */
00007
00008 #ifndef GUI_HPP_
00009 #define GUI_HPP_
00010
00011 #include <memory>
00012
00013 #include "RayLib/RayLib.hpp"
00014 #include "../Game/GameInfos.hpp"
00015 #include "Map.hpp"
00016 #include "HUD/HUD.hpp"
00017 #include "Audio/Audio.hpp"
00018 #include "../Utils/Constants.hpp"
00019 #include "Camera/CameraManager.hpp"
00020
00021 class GUI {
00022     public:
00023         explicit GUI(std::shared_ptr<GameInfos> gameInfos);
00024         ~GUI();
00025
00026         void run();
00027
00028         int getWindowWidth() const;
00029         int getWindowHeight() const;
00030         void setWindowWidth(int width);
00031         void setWindowHeight(int height);
00032
00033         void switchCameraMode(zappy::gui::CameraMode mode);
00034         void switchCameraModeNext();
00035         void setPlayerToFollow(int playerId);
00036         int getPlayerToFollow() const;
00037         bool selectFirstAvailablePlayer();
00038         void switchToNextPlayer();
00039         void switchToPreviousPlayer();
00040
00041     private:
00042         void updateCamera();
00043         void update();
00044         void draw();
00045         bool playerExists(int playerId) const;
00046
00047         void initModels();
00048
00049         bool _isRunning;
00050         std::shared_ptr<RayLib> _raylib;
00051         std::shared_ptr<GameInfos> _gameInfos;
00052         std::unique_ptr<Map> _map;
00053         std::unique_ptr<HUD> _hud;
00054         std::shared_ptr<Audio> _audio;
00055         std::unique_ptr<CameraManager> _cameraManager;
00056
00057         int _windowWidth;
00058         int _windowHeight;
00059
00060         zappy::gui::CameraMode _cameraMode;
00061 };
00062
00063 #endif /* !GUI_HPP_ */
```

7.11 Button.hpp

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

```
00017
00023 class Button : public AUIElement {
00024     public:
00035         Button(
00036             std::shared_ptr<RayLib> raylib,
00037             std::shared_ptr<Audio> audio,
00038             float x, float y,
00039             float width, float height,
00040             const std::string& text,
00041             std::function<void()> callback
00042         );
00043
00047         ~Button() override = default;
00048
00052         void draw() override;
00053
00057         void update() override;
00058
00064         void setText(const std::string& text);
00065
00071         std::string getText() const;
00072
00078         void setCallback(std::function<void()> callback);
00079
00088         void setColors(
00089             Color normal,
00090             Color hover,
00091             Color pressed,
00092             Color textColor
00093         );
00094
00101         void setSize(float width, float height) override;
00102
00103     private:
00104         std::string _text;
00105         std::function<void()> _callback;
00106
00107         Color _normalColor;
00108         Color _hoverColor;
00109         Color _pressedColor;
00110         Color _textColor;
00111
00112         bool _isHovered;
00113         bool _isPressed;
00114
00115         std::shared_ptr<RayLib> _raylib;
00116         std::shared_ptr<Audio> _audio;
00117 };
```

7.12 AContainers.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** AContainers
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <vector>
00012 #include <memory>
00013
00014 #include "IContainers.hpp"
00015 #include "../../RayLib/RayLib.hpp"
00016
00020 struct RelativePosition {
00021     float xPercent;
00022     float yPercent;
00023     float widthPercent;
00024     float heightPercent;
00025 };
00026
00032 class AContainers : public IContainers {
00033     public:
00042         AContainers(std::shared_ptr<RayLib> raylib, float x, float y, float width,
00043             float height);
00044
00048         virtual ~AContainers() = default;
00049
00050         void setPosition(float x, float y) override;
00051         void setSize(float width, float height) override;
00052         Rectangle getBounds() const override;
00053         bool contains(float x, float y) const override;
```

```
00054         void setVisible(bool visible) override;
00055         bool isVisible() const override;
00056
00065         void setRelativePosition(float xPercent, float yPercent, float widthPercent,
00066             float heightPercent);
00067
00073         RelativePosition getRelativePosition() const;
00074
00078         void updatePositionFromRelative();
00079
00080     protected:
00081         std::shared_ptr<RayLib> _raylib;
00082         Rectangle _bounds;
00083         RelativePosition _relativePos;
00084         Color _backgroundColor;
00085         bool _visible;
00086         bool _hasBackground;
00087 };
```

7.13 Containers.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Containers
00006 */
00007
00008 #pragma once
00009
00010 #include <vector>
00011 #include <functional>
00012 #include <unordered_map>
00013 #include <memory>
00014 #include <string>
00015
00016 #include "AContainers.hpp"
00017 #include "../UIElement/IUIElement.hpp"
00018 #include "../Button/Button.hpp"
00019 #include "../Text/Text.hpp"
00020 #include "../../RayLib/RayLib.hpp"
00021 #include "../../Audio/Audio.hpp"
00022
00029 class Containers : public AContainers {
00030     public:
00042         Containers(std::shared_ptr<RayLib> raylib, std::shared_ptr<Audio> audio,
00043             float x, float y, float width, float height,
00044             Color backgroundColor = {40, 40, 40, 200});
00045
00049         ~Containers() override;
00050
00054         void draw() override;
00055
00059         void update() override;
00060
00066         void setBackgroundColor(Color color);
00067
00073         void setHasBackground(bool hasBackground);
00074
00080         void setBackgroundTexture(Texture2D texture);
00081
00088         bool hasBackgroundTexture() const;
00089
00099         bool addElement(const std::string& id, std::shared_ptr<IUIElement> element);
00100
00108         std::shared_ptr<IUIElement> getElement(const std::string& id) const;
00109
00118         bool removeElement(const std::string& id);
00119
00133         std::shared_ptr<Button> addButton(
00134             const std::string& id,
00135             float x, float y,
00136             float width, float height,
00137             const std::string& text,
00138             std::function<void()> callback
00139         );
00140
00158         std::shared_ptr<Button> addButton(
00159             const std::string& id,
00160             float x, float y,
00161             float width, float height,
00162             const std::string& text,
00163             std::function<void()> callback,
00164             Color normalColor,
00165             Color hoverColor,
```

```
00166             Color pressedColor,
00167             Color textColor
00168         );
00169
00182         std::shared_ptr<Text> addText(
00183             const std::string& id,
00184             float x, float y,
00185             const std::string& text,
00186             float fontSize = 20.0f,
00187             Color color = BLACK
00188         );
00189
00193         void clearElements();
00194
00203         void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00204
00218         std::shared_ptr<Button> addButtonPercent(
00219             const std::string& id,
00220             float xPercent, float yPercent,
00221             float widthPercent, float heightPercent,
00222             const std::string& text,
00223             std::function<void()> callback
00224         );
00225
00243         std::shared_ptr<Button> addButtonPercent(
00244             const std::string& id,
00245             float xPercent, float yPercent,
00246             float widthPercent, float heightPercent,
00247             const std::string& text,
00248             std::function<void()> callback,
00249             Color normalColor,
00250             Color hoverColor,
00251             Color pressedColor,
00252             Color textColor
00253         );
00254
00267         std::shared_ptr<Text> addTextPercent(
00268             const std::string& id,
00269             float xPercent, float yPercent,
00270             const std::string& text,
00271             float fontSizePercent = 5.0f,
00272             Color color = BLACK
00273         );
00274
00275     private:
00276         std::shared_ptr<RayLib> _raylib;
00277         std::shared_ptr<Audio> _audio;
00278         Texture2D _backgroundTexture;
00279         bool _hasBackgroundTexture;
00280         std::unordered_map<std::string, std::shared_ptr<IUIElement» _elements;
00281 };
```

7.14 IContainers.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** IContainers
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <memory>
00012 #include <vector>
00013 #include "../../RayLib/RayLib.hpp"
00014
00021 class IContainers {
00022     public:
00023         virtual ~IContainers() = default;
00024
00028         virtual void draw() = 0;
00029
00033         virtual void update() = 0;
00034
00041         virtual void setPosition(float x, float y) = 0;
00042
00049         virtual void setSize(float width, float height) = 0;
00050
00056         virtual Rectangle getBounds() const = 0;
00057
00067         virtual bool contains(float x, float y) const = 0;
00068
00074         virtual void setVisible(bool visible) = 0;
```

```
00075
00082         virtual bool isVisible() const = 0;
00083 };
```

7.15 HUD.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** HUD
00006 */
00007
00008 #pragma once
00009
00010 #include <vector>
00011 #include <unordered_map>
00012 #include <memory>
00013 #include <string>
00014 #include <utility>
00015 #include "Containers/Containers.hpp"
00016 #include "../RayLib/RayLib.hpp"
00017 #include "../../Game/GameInfos.hpp"
00018 #include "../Audio/Audio.hpp"
00019
00026 class HUD {
00027     public:
00033         HUD(std::shared_ptr<RayLib> raylib, std::shared_ptr<GameInfos> gameInfos,
00034             std::shared_ptr<Audio> audio);
00035
00039         ~HUD();
00040
00044         void draw();
00045
00049         void update();
00050
00063         std::shared_ptr<Containers> addContainer(
00064             const std::string& id,
00065             float x, float y,
00066             float width, float height,
00067             Color backgroundColor = {40, 40, 40, 200}
00068         );
00069
00077         std::shared_ptr<Containers> getContainer(const std::string& id) const;
00078
00087         bool removeContainer(const std::string& id);
00088
00099         void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00100
00104         void clearAllContainers();
00105
00120         void initDefaultLayout(float sideWidthPercent = 15.0f,
00121             float bottomHeightPercent = 20.0f);
00122
00128         std::shared_ptr<Containers> getSideContainer() const;
00129
00135         std::shared_ptr<Containers> getBottomContainer() const;
00136
00142         std::shared_ptr<Containers> getSquareContainer() const;
00143
00149         void initExitButton();
00150
00156         void initSettingsButton();
00157
00163         void initHelpButton();
00164
00170         void initCameraResetButton();
00171
00179         void initTeamPlayersDisplay(std::shared_ptr<GameInfos> gameInfos);
00180
00188         void updateTeamPlayersDisplay(std::shared_ptr<GameInfos> gameInfos);
00189
00190     private:
00199         std::shared_ptr<Containers> createSquareContainer(float squareSize,
00200             float sideWidthPercent);
00201
00213         std::shared_ptr<Containers> createSideContainer(
00214             float sideYStart,
00215             float sideWidth,
00216             float sideHeight,
00217             float sideWidthPercent,
00218             float bottomHeightPercent);
00219
00230         std::shared_ptr<Containers> createBottomContainer(
00231             int screenWidth,
```

```
00232             int screenHeight,
00233             float bottomHeight,
00234             float bottomHeightPercent);
00235
00243         void recordElementPositions(
00244             std::shared_ptr<Containers> container,
00245             std::unordered_map<std::string, float>& initialYPositions,
00246             float& lastContainerHeight);
00247
00255         void updateElementPositions(
00256             std::shared_ptr<Containers> container,
00257             const std::unordered_map<std::string, float>& initialYPositions,
00258             float offset);
00259
00268         std::pair<float, float> calculateContentMetrics(
00269             std::shared_ptr<Containers> container,
00270             const std::unordered_map<std::string, float>& initialYPositions);
00271
00277         void clearTeamDisplayElements(std::shared_ptr<Containers> container);
00278
00287         std::vector<int> getTeamPlayerNumbers(const std::string& teamName,
00288             const std::vector<zappy::structs::Player>& players);
00289
00297         std::string createPlayerListText(const std::vector<int>& playerNumbers);
00298
00307         void addPlayerListText(std::shared_ptr<Containers> container,
00308                                const std::string& teamId,
00309                                float yPos, const std::vector<int>& playerNumbers);
00310
00311         std::unordered_map<std::string, std::shared_ptr<Containers>> _containers;
00312         std::shared_ptr<RayLib> _raylib;
00313         std::shared_ptr<GameInfos> _gameInfos;
00314         std::shared_ptr<Audio> _audio;
00315 };
```

7.16 Text.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Text
00006 */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include <string>
00012
00013 #include "../UIElement/AUIElement.hpp"
00014 #include "../../RayLib/RayLib.hpp"
00015
00021 class Text : public AUIElement {
00022     public:
00032         Text(
00033             std::shared_ptr<RayLib> raylib,
00034             float x, float y,
00035             const std::string& text,
00036             float fontSize = 20.0f,
00037             Color color = BLACK
00038         );
00039
00043         ~Text() override = default;
00044
00048         void draw() override;
00049
00053         void update() override;
00054
00060         void setText(const std::string& text);
00061
00067         std::string getText() const;
00068
00074         void setFontSize(float fontSize);
00075
00081         float getFontSize() const;
00082
00088         void setColor(Color color);
00089
00095         Color getColor() const;
00096
00104         void setSize(float width, float height) override;
00105
00106     private:
00107         std::string _text;
00108         float _fontSize;
```

```
00109         Color _color;
00110         std::shared_ptr<RayLib> _raylib;
00111 };
```

7.17 AUIElement.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** AUIElement
00006 */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include "IUIElement.hpp"
00012 #include "../../RayLib/RayLib.hpp"
00013
00017 struct UIRelativePosition {
00018     float xPercent;
00019     float yPercent;
00020     float widthPercent;
00021     float heightPercent;
00022 };
00023
00029 class AUIElement : public IUIElement {
00030     public:
00039         AUIElement(std::shared_ptr<RayLib> raylib, float x, float y, float width,
00040             float height);
00041
00045         virtual ~AUIElement() = default;
00046
00047         // IUIElement implementation
00048         void setPosition(float x, float y) override;
00049         Rectangle getBounds() const override;
00050         bool contains(float x, float y) const override;
00051         void setVisible(bool visible) override;
00052         bool isVisible() const override;
00053
00060         virtual void setSize(float width, float height);
00061
00070         void setRelativePosition(float xPercent, float yPercent, float widthPercent,
00071             float heightPercent);
00072
00078         UIRelativePosition getRelativePosition() const;
00079
00080     protected:
00081         std::shared_ptr<RayLib> _raylib;
00082         Rectangle _bounds;
00083         UIRelativePosition _relativePos;
00084         bool _visible;
00085 };
```

7.18 IUIElement.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** IUIElement
00006 */
00007
00008 #pragma once
00009
00010 #include "../../RayLib/RayLib.hpp"
00011
00017 class IUIElement {
00018     public:
00019         virtual ~IUIElement() = default;
00020
00024         virtual void draw() = 0;
00025
00029         virtual void update() = 0;
00030
00037         virtual void setPosition(float x, float y) = 0;
00038
00045         virtual void setSize(float width, float height) = 0;
00046
00052         virtual Rectangle getBounds() const = 0;
00053
00063         virtual bool contains(float x, float y) const = 0;
```

```
00064
00070         virtual void setVisible(bool visible) = 0;
00071
00078         virtual bool isVisible() const = 0;
00079 };
```

7.19 Map.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Map
00006 */
00007
00008 #ifndef MAP_HPP_
00009 #define MAP_HPP_
00010
00011 #include <memory>
00012 #include <unordered_map>
00013 #include <string>
00014 #include "../Game/GameInfos.hpp"
00015 #include "RayLib/RayLib.hpp"
00016
00017 enum class DisplayPriority {
00018     TILE = 0,
00019     EGG = 1,
00020     PLAYER = 2,
00021     FOOD = 3,
00022     ROCK = 4,
00023 };
00024
00025 class Map {
00026     public:
00027         Map(std::shared_ptr<GameInfos> gameInfos, std::shared_ptr<RayLib> raylib);
00028         ~Map();
00029
00030         void draw();
00031         void drawTile(int x, int y, const zappy::structs::Tile &tile);
00032         void drawRock(int x, int y, const zappy::structs::Tile &tile);
00033         void drawFood(int x, int y, const zappy::structs::Tile &tile);
00034         void drawPlayers(int x, int y);
00035         void drawEggs(int x, int y);
00036         Color getTeamColor(const std::string &teamName);
00037
00038         float getOffset(DisplayPriority priority, int x, int y, size_t stackIndex = 0);
00039
00040     private:
00041         std::shared_ptr<GameInfos> _gameInfos;
00042         std::shared_ptr<RayLib> _raylib;
00043         std::unordered_map<std::string, Color> _teamColors;
00044
00045         static constexpr float BASE_HEIGHT_TILE = 0.0f;
00046         static constexpr float BASE_HEIGHT_FOOD = 0.2f;
00047         static constexpr float BASE_HEIGHT_ROCK = 0.2f;
00048         static constexpr float BASE_HEIGHT_EGG = 0.2f;
00049         static constexpr float BASE_HEIGHT_PLAYER = 0.2f;
00050         static constexpr float FOOD_HEIGHT = 0.3f;
00051         static constexpr float ROCK_HEIGHT = 0.3f;
00052         static constexpr float EGG_HEIGHT = 0.3f;
00053         static constexpr float PLAYER_HEIGHT = 1.1f;
00054
00055         void drawOrientationArrow(const Vector3 &position, int orientation,
00056             float playerHeight);
00057 };
00058
00059 #endif /* !MAP_HPP_ */
```

7.20 RayLib.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** RayLib
00006 */
00007
00008 #ifndef RAYLIB_HPP_
00009 #define RAYLIB_HPP_
00010
00011 #include <string>
00012 #include <map>
```

```
00013 #include <memory>
00014 #include "raylib.h"
00015
00016 class RayLib {
00017     public:
00018         RayLib();
00019         ~RayLib();
00020
00021         // Window management methods
00022         void initWindow(int width, int height, const std::string &title);
00023         void closeWindow();
00024         bool windowShouldClose() const;
00025         void beginDrawing();
00026         void endDrawing();
00027         void clearBackground(Color color = WHITE);
00028         bool isWindowReady() const;
00029         int getMonitorWidth(int monitor) const;
00030         int getMonitorHeight(int monitor) const;
00031         void waitTime(float seconds) const;
00032         void setTargetFPS(int fps) const;
00033         int getFPS() const;
00034         float getFrameTime() const;
00035
00036         // Collision methods
00037         bool checkCollisionPointRec(Vector2 point, Rectangle rec) const;
00038
00039         // Texture methods
00040         void drawTextureRec(Texture2D texture, Rectangle source, Vector2 position, Color tint);
00041         void unloadTexture(Texture2D texture);
00042
00043         // Input methods
00044         bool isMouseButtonDown(int button) const;
00045         bool isMouseButtonPressed(int button) const;
00046         bool isMouseButtonReleased(int button) const;
00047         bool isKeyDown(int key) const;
00048         bool isKeyPressed(int key) const;
00049         bool isKeyReleased(int key) const;
00050         Vector2 getMouseDelta();
00051         Vector2 getMousePosition() const;
00052         void setMousePosition(int x, int y);
00053         void disableCursor();
00054         void enableCursor();
00055         int getScreenWidth() const;
00056         int getScreenHeight() const;
00057         float getMouseWheelMove() const;
00058
00059         // Scissor mode methods for clipping
00060         void beginScissorMode(int x, int y, int width, int height);
00061         void endScissorMode();
00062
00063         // 3D Environment methods
00064         void begin3DMode();
00065         void end3DMode();
00066         float vector3Distance(Vector3 v1, Vector3 v2) const;
00067         Vector3 vector3Normalize(Vector3 v) const;
00068         Vector3 vector3Subtract(Vector3 v1, Vector3 v2) const;
00069         Vector3 vector3Add(Vector3 v1, Vector3 v2) const;
00070
00071         // Camera methods
00072         void initCamera();
00073         void setCameraPosition(Vector3 position);
00074         void setCameraTarget(Vector3 target);
00075         void setCameraUp(Vector3 up);
00076         void setCameraFovy(float fovy);
00077         void setCameraProjection(int projection);
00078         void updateCamera(int mode = CAMERA_FREE);
00079         void updateCameraFreeMode();
00080         Camera3D getCamera() const;
00081
00082         // 3D Drawing methods
00083         void drawGrid(int slices, float spacing);
00084         void drawCube(Vector3 position, float width, float height, float length, Color color);
00085         void drawCubeWires(Vector3 position, float width, float height, float length,
00086             Color color);
00087         void drawSphere(Vector3 position, float radius, Color color);
00088         void drawSphereWires(Vector3 position, float radius, int rings, int slices,
00089             Color color);
00090         void drawCylinder(Vector3 position, float radiusTop, float radiusBottom,
00091             float height, int slices, Color color);
00092         void drawCylinderWires(Vector3 position, float radiusTop, float radiusBottom,
00093             float height, int slices, Color color);
00094         void drawCylinderEx(Vector3 startPos, Vector3 endPos, float startRadius,
00095             float endRadius, int sides, Color color);
00096         void drawPlane(Vector3 position, Vector2 size, Color color);
00097         void drawLine3D(Vector3 startPos, Vector3 endPos, Color color);
00098
00099         // 3D Model methods
```

```
00100         bool loadModel(const std::string& id, const std::string& filepath,
00101             Vector3 center = {0.0f, 0.0f, 0.0f});
00102         void drawModel(const std::string& id, Vector3 position, float scale,
00103             Color tint = WHITE);
00104         void drawModelEx(const std::string& id, Vector3 position, Vector3 rotationAxis,
00105                          float rotationAngle, Vector3 scale, Color tint = WHITE);
00106         void drawModelWires(const std::string& id, Vector3 position, float scale,
00107             Color tint = WHITE);
00108         void drawModelWiresEx(const std::string& id, Vector3 position, Vector3 rotationAxis,
00109                               float rotationAngle, Vector3 scale, Color tint = WHITE);
00110         void unloadModel(const std::string& id);
00111         void unloadAllModels();
00112         bool modelExists(const std::string& id) const;
00113
00114         // 2D Drawing methods
00115         void drawRectangleRec(Rectangle rec, Color color);
00116         void drawText(const std::string& text, float x, float y, float fontSize, Color color);
00117         float measureText(const std::string& text, float fontSize) const;
00118
00119         // Audio methods
00120         void initAudioDevice();
00121         void closeAudioDevice();
00122         bool isAudioDeviceReady() const;
00123
00124     private:
00125         bool _isInitialized;
00126         Camera3D _camera;
00127         Vector2 _previousMousePosition;
00128         bool _isCursorLocked;
00129
00130         struct ModelData {
00131             Model model;
00132             unsigned int animationCount;
00133             Vector3 center;
00134         };
00135
00136         std::map<std::string, ModelData> _models;
00137         std::map<std::string, Sound> _sounds;
00138         std::map<std::string, Music> _musics;
00139 };
00140
00141 #endif /* !RAYLIB_HPP_ */
```

7.21 Constants.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Constants
00006 */
00007
00008 #ifndef CONSTANTS_HPP_
00009 #define CONSTANTS_HPP_
00010
00011 #include <string>
00012 #include <vector>
00013
00014 namespace zappy::constants {
00015
00016     inline const char *USAGE_STRING = "USAGE: ./zappy_gui -p port -h machine\n"
00017                                 "option\t\tdescription\n"
00018                                 "-p port\t\tport number\n"
00019                                 "-h machine\thostname of the server";
00020
00021     inline const int FAILURE_EXIT_CODE = 84;
00022     inline const int SUCCESS_EXIT_CODE = 0;
00023 };
00024
00025 namespace colors {
00026
00027     inline const char *T_BOLD = "\033[1m";
00028     inline const char *T_RED = "\033[1m\033[31m";
00029     inline const char *T_GREEN = "\033[1m\033[32m";
00030     inline const char *T_YELLOW = "\033[1m\033[33m";
00031     inline const char *T_BLUE = "\033[1m\033[34m";
00032     inline const char *T_MAGENTA = "\033[1m\033[35m";
00033     inline const char *T_CYAN = "\033[1m\033[36m";
00034     inline const char *T_WHITE = "\033[1m\033[37m";
00035     inline const char *RESET = "\033[0m";
00036
00037 };
00038
00039 namespace zappy::structs {
00040
```

```
00041     struct Config {
00042         int port;
00043         std::string hostname;
00044     };
00045
00046     struct Tile {
00047         int x;
00048         int y;
00049         int food;
00050         int linemate;
00051         int deraumere;
00052         int sibur;
00053         int mendiane;
00054         int phiras;
00055         int thystame;
00056
00057         Tile(int _x = 0, int _y = 0, int _food = 0, int _linemate = 0,
00058              int _deraumere = 0, int _sibur = 0, int _mendiane = 0,
00059              int _phiras = 0, int _thystame = 0)
00060             : x(_x), y(_y), food(_food), linemate(_linemate),
00061               deraumere(_deraumere), sibur(_sibur),
00062               mendiane(_mendiane), phiras(_phiras), thystame(_thystame) {}
00063     };
00064
00065     struct Inventory {
00066         int food;
00067         int linemate;
00068         int deraumere;
00069         int sibur;
00070         int mendiane;
00071         int phiras;
00072         int thystame;
00073
00074         Inventory(int _food = 0, int _linemate = 0, int _deraumere = 0,
00075                   int _sibur = 0, int _mendiane = 0, int _phiras = 0,
00076                   int _thystame = 0)
00077             : food(_food), linemate(_linemate), deraumere(_deraumere),
00078               sibur(_sibur), mendiane(_mendiane), phiras(_phiras),
00079               thystame(_thystame) {}
00080     };
00081     struct Player {
00082         int number;
00083         int x;
00084         int y;
00085         int orientation;
00086         int level;
00087         std::string teamName;
00088         struct Inventory inventory;
00089
00090         Player(int _number = 0, int _x = 0, int _y = 0, int _orientation = 0,
00091                int _level = 1, const std::string &_teamName = "",
00092                struct Inventory _inventory = Inventory())
00093             : number(_number), x(_x), y(_y), orientation(_orientation),
00094               level(_level), teamName(_teamName), inventory(_inventory) {}
00095     };
00096
00097     struct Incantation {
00098         int x;
00099         int y;
00100         int level;
00101         std::vector<int> players;
00102
00103         Incantation(int _x = 0, int _y = 0, int _level = 1,
00104                     const std::vector<int> &_players = {})
00105             : x(_x), y(_y), level(_level), players(_players) {}
00106     };
00107
00108     struct Egg {
00109         int eggNumber;
00110         int playerNumber;
00111         int x;
00112         int y;
00113         bool hatched;
00114         std::string teamName;
00115
00116         Egg(int _eggNumber = 0, int _playerNumber = 0, int _x = 0, int _y = 0,
00117             bool _hatched = false, const std::string &_teamName = "")
00118             : eggNumber(_eggNumber), playerNumber(_playerNumber), x(_x), y(_y),
00119               hatched(_hatched), teamName(_teamName) {}
00120     };
00121 };
00122
00123 namespace zappy::gui {
00124
00125     inline const std::string WINDOW_TITLE = "Zappy GUI";
00126     inline const int FPS = 120;
00127     inline const float CAMERA_SPEED = 7.5f;
```

```
00128     inline const float CAMERA_SENSITIVITY = 0.001f;
00129
00130     inline const float PLAYER_SCALE = 0.005f;
00131
00132     enum class CameraMode {
00133         FREE = 0,
00134         TARGETED = 1,
00135         PLAYER = 2,
00136         NB_MODES = 3,
00137     };
00138 }
00139
00140 #endif /* !CONSTANTS_HPP_ */
```

7.22 algo.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** algo
00006 */
00007
00008 #ifndef ALGO_H_
00009     #define ALGO_H_
00010
00011 typedef struct tiles_s {
00012     int x;
00013     int y;
00014 } tiles_t;
00015
00016 /* Algo.c */
00017 tiles_t *shuffle_fisher(int width, int heigth);
00018
00019 #endif /* !ALGO_H_ */
```

7.23 game.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** game
00006 */
00007
00008 #include "buffer.h"
00009 #include <time.h>
00010 #include <pthread.h>
00011
00012 #ifndef GAME_H_
00013     #define GAME_H_
00014
00015 typedef struct action_request_s action_request_t;
00016 typedef struct action_queue_s action_queue_t;
00017 typedef struct player_s player_t;
00018
00019 /* Definition of the directions */
00020 typedef enum direction_e {
00021     NORTH = 1,
00022     EAST = 2,
00023     SOUTH = 3,
00024     WEST = 4
00025 } direction_t;
00026
00027 /* definintion od the different element on the map */
00028 typedef enum crystal_e {
00029     FOOD,
00030     LINEMATE,
00031     DERAUMERE,
00032     SIBUR,
00033     MENDIANE,
00034     PHIRAS,
00035     THYSTAME
00036 } crystal_t;
00037
00038
00039 /* This enum defines the priority of the action in the queue */
00040 typedef enum action_priority_e {
00041     PRIORITY_CRITICAL = 0,
00042     PRIORITY_HIGH = 1,
00043     PRIORITY_MEDIUM = 2,
00044     PRIORITY_LOW = 3
```

```
00045 } action_priority_t;
00046
00047 /* This strucuture allows use to define a 'queue' of the requests */
00048 typedef struct action_queue_s {
00049     action_request_t *head;
00050     action_request_t *tail;
00051     int count;
00052     pthread_mutex_t mutex;
00053 } action_queue_t;
00054
00055
00056 typedef struct egg_s {
00057     int id; /* Id of the egg */
00058     int posX;
00059     int posY;
00060     char *teamName;  /* Name of the team that laid it */
00061     int idLayer;  /* Id of the player that layed it */
00062     bool isHatched;
00063     struct egg_s *next;
00064 } egg_t;
00065
00066 /* Struct that "handles" the network element */
00067 typedef struct network_s {
00068     int fd;
00069     buffer_t *buffer;
00070 } network_t;
00071
00072 /* Struct defining the inventory of tiles and players */
00073 typedef struct inventory_s {
00074     int nbFood;
00075     int nbLinemate;
00076     int nbDeraumere;
00077     int nbSibur;
00078     int nbMendiane;
00079     int nbPhiras;
00080     int nbThystame;
00081 } inventory_t;
00082
00083
00084 /* Player struct */
00085 typedef struct player_s {
00086     int id;
00087     network_t *network;
00088     int level;
00089     int posX;
00090     int posY;
00091     direction_t direction;
00092     inventory_t *inventory;
00093     char *team;
00094     /* New aditions for the smart pollin */
00095     action_queue_t *pending_actions;
00096     time_t last_action_time;
00097     bool is_busy;
00098     int remaining_cooldown;
00099
00100     struct player_s *next;
00101 } player_t;
00102
00103 /* This structure define the request strut */
00104 typedef struct action_request_s {
00105     char *command;
00106     time_t timestamp;
00107     int time_limit;  // in game ticks (7/f, 42/f, etc.)
00108     action_priority_t priority;
00109     player_t *player;
00110     struct action_request_s *next;
00111 } action_request_t;
00112
00113 /* Team Strcut */
00114 typedef struct team_s {
00115     char *name;
00116     int nbPlayers;
00117     int nbPlayerAlive;
00118     player_t *players;
00119     struct team_s *next;
00120 } team_t;
00121
00122
00123 /* Structure that holds the size and array of tiles */
00124 typedef struct map_t {
00125     int width;
00126     int height;
00127     egg_t *currentEggs;  /* List of current eggs */
00128     inventory_t **tiles;  /* Here we call inv for the tile*/
00129 } map_t;
00130
00131
```

```
00132 /* Map struct */
00133 typedef struct game_s {
00134     team_t *teams;
00135     map_t *map;
00136 } game_t;
00137
00138 #endif /* !GAME_H_ */
```

7.24 my.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** my
00006 */
00007
00008 #ifndef MY_H_
00009     #define MY_H_
00010
00011 int int_str_len(int value);
00012 char *my_itoa(unsigned int nb);
00013 int is_only_digits(const char *str);
00014 int my_unsignedlen(unsigned int nb);
00015
00016 #endif /* !MY_H_ */
```

7.25 my.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** my
00006 */
00007
00008 #ifndef MY_H_
00009     #define MY_H_
00010
00011 int int_str_len(int value);
00012 char *my_itoa(unsigned int nb);
00013 int is_only_digits(const char *str);
00014 int my_unsignedlen(unsigned int nb);
00015
00016 #endif /* !MY_H_ */
```

7.26 zappy.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** Zappy
00004 ** File description:
00005 ** Server :: Zappy header
00006 */
00007
00008 #include <stdbool.h>
00009 #include <poll.h>
00010 #include "game.h"
00011 #include "my.h"
00012
00013 #ifndef ZAPPY_H_
00014     #define ZAPPY_H_
00015
00016 /* Cli parameter of the server */
00017 typedef struct params_s {
00018     int port;
00019     int x;
00020     int y;
00021     int nb_team;
00022     char **teams;
00023     int nb_client;
00024     int freq;
00025     bool is_debug;
00026 } params_t;
00027
00028 /* Structure to handle the network side of the gui*/
00029 typedef struct graph_net_s {
00030     int fd;
00031     bool mapSent;
```

```
00032     struct graph_net_s *next;
00033 } graph_net_t;
00034
00035 /* Server part of the network */
00036 typedef struct server_s {
00037     int sockfd;
00038     struct pollfd pollserver;
00039 } server_t;
00040
00041 typedef struct zappy_s {
00042     server_t *network;
00043     game_t *game;
00044     graph_net_t *graph;
00045     params_t *params;
00046 } zappy_t;
00047
00048 typedef struct command_pf_s {
00049     char const *flag;
00050     bool (*checker)(const char *, const char *, params_t *);
00051 } command_pf_t;
00052
00053 typedef struct {
00054     char *command;
00055     int base_time;
00056     action_priority_t priority;
00057     int (*handler)(player_t *, char *, zappy_t *);
00058 } command_info_t;
00059
00060 /* messages.c */
00061 int helper(void);
00062 void error_message(const char *message);
00063 void valid_message(char const *message);
00064
00065 /* checkers.c */
00066 bool check_port(char const *flag, char const *value, params_t *params);
00067 bool check_width(char const *flag, char const *value, params_t *params);
00068 bool check_height(char const *flag, char const *value, params_t *params);
00069 bool check_client(char const *flag, char const *value, params_t *params);
00070 bool check_freq(char const *flag, char const *value, params_t *params);
00071
00072 /* signal.c */
00073 void setup_signal(void);
00074 int *get_running_state(void);
00075
00076 /* params.c */
00077 params_t *check_args(int argc, char **argv);
00078 void *free_params(params_t *params);
00079
00080 /* params_cherckers.c */
00081 bool validate_no_extra_args(int argc, char **argv);
00082
00083 /* server.c */
00084 zappy_t *init_server(int argc, char **argv);
00085 void *free_zappy(zappy_t *server);
00086
00087 /* protocol.c */
00088 int start_protocol(zappy_t *server);
00089
00090 /* client.c */
00091 bool process_new_client(const char *team_name, int fd, zappy_t *server);
00092 team_t *add_client_to_team(const char *team_name, int fd, zappy_t *server);
00093 int get_next_free_id(zappy_t *server);
00094 void check_player_status(zappy_t *zappy);
00095
00096 /* init_map.c */
00097 void init_game(zappy_t *server);
00098
00099 /* accept.c */
00100 int accept_client(zappy_t *server);
00101
00102 /* free server  */
00103 void *free_zappy(zappy_t *server);
00104 void *free_params(params_t *params);
00105 void *free_player(player_t *player);
00106 void free_map(map_t *map);
00107
00108 /* Function to send info to the gui */
00109 int send_map_size(zappy_t *server);
00110 int send_entrie_map(zappy_t *server);
00111 int send_map_tile(inventory_t **tiles, zappy_t *server,
00112     int posX, int posY);
00113 int send_team_name(zappy_t *server);
00114 int send_egg(zappy_t *zappy, egg_t *egg);
00115 int send_entire_egg_list(zappy_t *zappy);
00116 int send_time_message(zappy_t *zappy);
00117 int send_egg_death(zappy_t *zappy, egg_t *egg);
00118 int send_egg_connect(zappy_t *zappy, egg_t *currentEgg);
```

```
00119 int send_player_connect(zappy_t *zappy, player_t *player);
00120 int send_player_pos(zappy_t *zappy, player_t *player);
00121 int send_player_level(zappy_t *zappy, player_t *player);
00122 int send_player_inventory(zappy_t *zappy, player_t *player);
00123 int send_player_expelled(zappy_t *zappy, player_t *player);
00124 int send_broadcast_to_all(zappy_t *zappy, const char *message);
00125 int send_broadcast_to_player(zappy_t *zappy, player_t *player,
00126     const char *message);
00127 int send_player_laying_egg(zappy_t *zappy, player_t *player);
00128 int send_ressource_droped(zappy_t *zappy, player_t *player,
00129     int ressourceType);
00130 int send_ressource_collected(zappy_t *zappy, player_t *player,
00131     int ressourceType);
00132 int send_player_death(zappy_t *zappy, player_t *player);
00133 int send_updated_time(zappy_t *zappy, int time);
00134 int send_end_game(zappy_t *zappy, const char *teamName);
00135 int send_str_message(zappy_t *zappy, const char *message);
00136 int send_unknown_command(zappy_t *zappy);
00137 int send_command_parameter(zappy_t *zappy);
00138
00139 /* init_egg.c */
00140 void init_egg(zappy_t *zappy);
00141 egg_t *kil_egg_node(egg_t **head, int egg_id);
00142
00143 /* AI messages */
00144 int forward_message(player_t *player, params_t *params);
00145
00146 /* Pollin handler */
00147 void smart_poll_players(zappy_t *zappy);
00148 void execute_action(player_t *player, action_request_t *action,
00149     zappy_t *zappy);
00150 void queue_action(player_t *player, char *command, zappy_t *zappy);
00151 action_queue_t *init_action_queue(void);
00152 void free_action_queue(action_queue_t *queue);
00153 action_request_t *create_action_request(char *command, player_t *player);
00154 const command_info_t *find_command_info(char *command);
00155 action_request_t *dequeue_highest_priority_action(action_queue_t *queue);
00156 void free_action_request(action_request_t *action);
00157 void insert_action_by_priority(action_queue_t *queue,
00158     action_request_t *action);
00159
00160 /* This is the definition of the array function of the commands */
00161 int handle_forward(player_t *player, char *command, zappy_t *zappy);
00162 int handle_left(player_t *player, char *command, zappy_t *zappy);
00163 int handle_right(player_t *player, char *command, zappy_t *zappy);
00164 int handle_connect_nbr(player_t *player, char *command, zappy_t *zappy);
00165 int handle_eject(player_t *player, char *command, zappy_t *zappy);
00166 int handle_fork(player_t *player, char *command, zappy_t *zappy);
00167 int handle_incantation(player_t *player, char *command, zappy_t *zappy);
00168 int handle_inventory(player_t *player, char *command, zappy_t *zappy);
00169 int handle_broadcast(player_t *player, char *command, zappy_t *zappy);
00170 int handle_look(player_t *player, char *command, zappy_t *zappy);
00171 int handle_set(player_t *player, char *command, zappy_t *zappy);
00172 int handle_take(player_t *player, char *command, zappy_t *zappy);
00173
00174 /* graphic_clinet.c */
00175 graph_net_t *add_graph_node(graph_net_t **head, int fd);
00176 graph_net_t *remove_graph_node(graph_net_t **head, int fd);
00177 void poll_graphic_clients(zappy_t *zappy);
00178
00179 #endif /* !ZAPPY_H_ */
```

7.27 buffer.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** buffer
00006 */
00007
00008 #include <stddef.h>
00009
00010 #ifndef BUFFER_H_
00011     #define BUFFER_H_
00012
00013     #define BUFFER_SIZE 1024
00014
00015
00016 typedef struct buffer_s {
00017     char data[BUFFER_SIZE];
00018     int head;
00019     int tail;
00020     int full;
00021 } buffer_t;
```

```
00022
00023 /* buffer.c */
00024 int advance(int idx);
00025 void cb_write(buffer_t *cb, char c);
00026 int cb_getline(buffer_t *cb, char *line, int max_len);
00027
00028 #endif /* !BUFFER_H_ */
```

7.28 buffer.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** buffer
00006 */
00007
00008 #include <stddef.h>
00009
00010 #ifndef BUFFER_H_
00011     #define BUFFER_H_
00012
00013     #define BUFFER_SIZE 1024
00014
00015
00016 typedef struct buffer_s {
00017     char data[BUFFER_SIZE];
00018     int head;
00019     int tail;
00020     int full;
00021 } buffer_t;
00022
00023 /* buffer.c */
00024 int advance(int idx);
00025 void cb_write(buffer_t *cb, char c);
00026 int cb_getline(buffer_t *cb, char *line, int max_len);
00027
00028 #endif /* !BUFFER_H_ */
```

7.29 network.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** network
00006 */
00007
00008 #ifndef NETWORK_H_
00009     #define NETWORK_H_
00010
00011 /* Write an errro message */
00012 void error_print(char const *message);
00013 /* Set the socket of the file descriptor */
00014 int set_socket(void);
00015 /* Bind the file decriptor to the port */
00016 int bind_socket(int fd, int port);
00017 /* Specify the queue the fd will use */
00018 int listen_socket(int fd, int backlog);
00019
00020 /* Close the server */
00021 void close_fd(int fd);
00022
00023 /* Accept new connetion */
00024 int accept_connection(int server_fd);
00025 /* Handle Message input */
00026 char *get_message(int fd, int timeout);
00027 /* Hello */
00028 int write_message(int fd, const char *message);
00029 #endif /* !NETWORK_H_ */
```

7.30 network.h

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

```
00007
00008 #ifndef NETWORK_H_
00009     #define NETWORK_H_
00010
00011 /* Write an errro message */
00012 void error_print(char const *message);
00013 /* Set the socket of the file descriptor */
00014 int set_socket(void);
00015 /* Bind the file decriptor to the port */
00016 int bind_socket(int fd, int port);
00017 /* Specify the queue the fd will use */
00018 int listen_socket(int fd, int backlog);
00019
00020 /* Close the server */
00021 void close_fd(int fd);
00022
00023 /* Accept new connetion */
00024 int accept_connection(int server_fd);
00025 /* Handle Message input */
00026 char *get_message(int fd, int timeout);
00027 /* Hello */
00028 int write_message(int fd, const char *message);
00029 #endif /* !NETWORK_H_ */
```