Zappy architecture

Generated by Doxygen 1.10.0

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

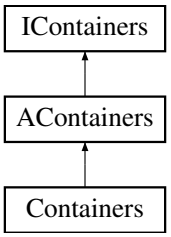
Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 AContainers Class Reference

Inheritance diagram for AContainers:



Public Member Functions

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

Public Member Functions inherited from IContainers

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

Protected Attributes

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

4.1.1 Member Function Documentation

4.1.1.1 contains()

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

Implements IContainers.

4.1.1.2 getBounds()

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

Implements IContainers.

4.1.1.3 isVisible()

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

Implements IContainers.

4.1.1.4 setPosition()

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

Implements IContainers.

4.1.1.5 setSize()

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

Implements IContainers.

4.1.1.6 setVisible()

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

Implements IContainers.

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

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

4.2 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

4.3 action_request_s Struct Reference

Public Attributes

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

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

- server/include/game.h

4.4 App.App Class Reference

Public Member Functions

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

Public Attributes

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

Protected Member Functions

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

Protected Attributes

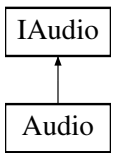
- **_signal_handler**
- **_child_signal_handler**

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

- ai/src/App/App.py

4.5 Audio Class Reference

Inheritance diagram for Audio:



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**

4.5.1 Member Function Documentation

4.5.1.1 isSoundPlaying()

```
bool Audio::isSoundPlaying (
            const std::string & id ) const  [virtual]
```

Implements IAudio.

4.5.1.2 loadSound()

```
bool Audio::loadSound (
            const std::string & id,
            const std::string & filepath )  [virtual]
```

Implements IAudio.

4.5.1.3 playSound()

```
void Audio::playSound (
            const std::string & id,
            float volume = 1.0f )  [virtual]
```

Implements IAudio.

4.5.1.4 setSoundLooping()

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

Implements IAudio.

4.5.1.5 setSoundVolume()

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

Implements IAudio.

4.5.1.6 stopSound()

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

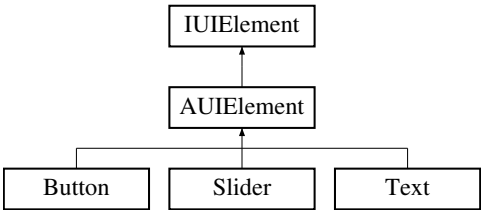
Implements IAudio.

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

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

4.6 AUIElement Class Reference

Inheritance diagram for AUIElement:



Public Member Functions

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

Public Member Functions inherited from IUIElement

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

Protected Attributes

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

4.6.1 Member Function Documentation

4.6.1.1 contains()

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

Implements IUIElement.

4.6.1.2 getBounds()

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

Implements IUIElement.

4.6.1.3 isVisible()

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

Implements IUIElement.

4.6.1.4 setPosition()

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

Implements IUIElement.

4.6.1.5 setSize()

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

Implements IUIElement.

4.6.1.6 setVisible()

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

Implements IUIElement.

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

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

4.7 Broadcaster.Broadcaster Class Reference

Public Member Functions

- None **__init__** (self, Communication com, str team)
- str **revealMessage** (self, str message)
- None **broadcastMessage** (self, str message)

Public Attributes

- **com**
- **hash**
- **lastIndex**

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

- ai/src/Broadcaster/Broadcaster.py

4.8 buffer_s Struct Reference

Public Attributes

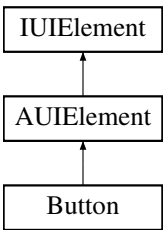
- char **data** [BUFFER_SIZE]
- int **head**
- int **tail**
- int **full**

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

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

4.9 Button Class Reference

Inheritance diagram for Button:



Public Member Functions

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

Public Member Functions inherited from AUIElement

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

Private Attributes

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

Additional Inherited Members

Protected Attributes inherited from AUIElement

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

4.9.1 Member Function Documentation

4.9.1.1 draw()

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

Implements IUIElement.

4.9.1.2 setSize()

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

Reimplemented from AUIElement.

4.9.1.3 update()

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

Implements IUIElement.

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

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

4.10 CameraManager Class Reference

Public Member Functions

- **CameraManager** (std::shared_ptr< IDisplay > display)
- void **updateCamera** (zappy::gui::CameraMode mode)
- void **updateCameraFreeMode** ()
- void **updateCameraTargetMode** ()
- void **updateCameraPlayerMode** ()
- void **setMapCenter** (const Vector3f &center)
- void **setMapSize** (int width, int height)
- void **setTargetDistance** (float distance)
- void **initTargetPositionFromCurrentCamera** ()
- void **setPlayerId** (int playerId)
- int **getPlayerId** () const
- void **setGameInfos** (std::shared_ptr< GameInfos > gameInfos)
- void **setMapInstance** (std::shared_ptr< Map > map)

Private Member Functions

- void **handlePlayerCameraMouseInput** ()
- Vector3f **calculatePlayerPosition** (const zappy::structs::Player &player)
- Vector3f **calculateCameraPosition** (const Vector3f &playerPos, float angleXZ)

Private Attributes

- std::shared_ptr< IDisplay > **_display**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< Map > **_map**
- Vector3f **_mapCenter**
- int **_mapWidth**
- int **_mapHeight**
- float **_targetDistance**
- float **_targetAngleXZ**
- float **_targetAngleY**
- bool **_isDragging**
- int **_playerId**
- float **_playerAngleXZ**
- bool **_isPlayerViewDragging**

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

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

4.11 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

4.12 CLI.CLI Class Reference

Public Member Functions

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

Public Attributes

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

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

- ai/src/CLI/CLI.py

4.13 Client Class Reference

Public Member Functions

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

Private Member Functions

- void **_tryToCreateGuiWithSharedLibInFolder** (const std::string &libPath="./gui/lib/")
- void **initialize** (int ac, const char ∗const ∗av)

Private Attributes

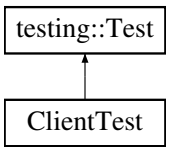
- zappy::structs::Config **_config**
- std::shared_ptr< ICommunication > **_communication**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::unique_ptr< MsgHandler > **_msgHandler**
- std::unique_ptr< GUI > **_gui**
- std::shared_ptr< GuiObserver > **_guiObserver**

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

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

4.14 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

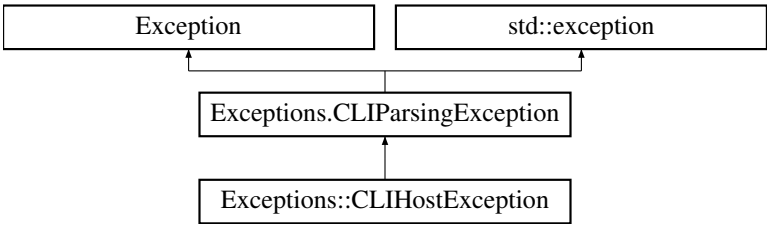
- std::stringstream **buffer**
- std::streambuf ∗ **originalCout**

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

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

4.15 Exceptions::CLIHostException Class Reference

Inheritance diagram for Exceptions::CLIHostException:



Public Member Functions

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

Public Member Functions inherited from Exceptions.CLIParsingException

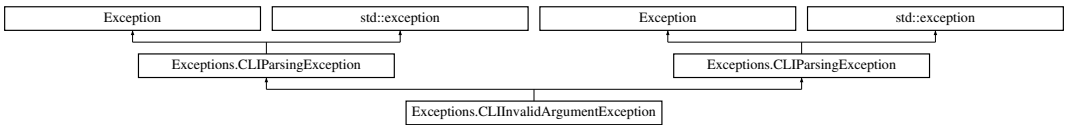
- **__init__** (self, str message)
- **CLIParsingException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

4.16 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

4.16.1 Constructor & Destructor Documentation

4.16.1.1 __init__()

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

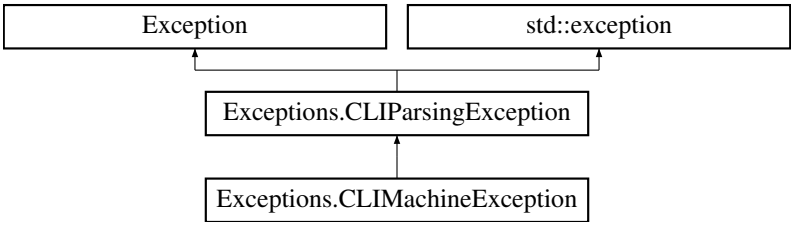
Reimplemented from Exceptions.CLIParsingException.

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

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

4.17 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

4.17.1 Constructor & Destructor Documentation

4.17.1.1 __init__()

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

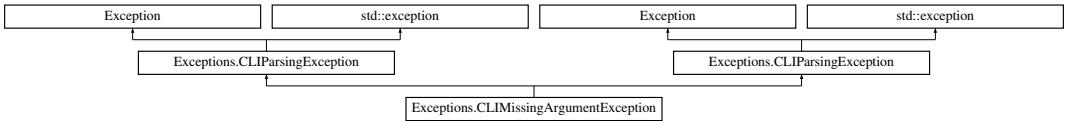
Reimplemented from Exceptions.CLIParsingException.

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

- ai/src/Exceptions/Exceptions.py

4.18 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

4.18.1 Constructor & Destructor Documentation

4.18.1.1 __init__()

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

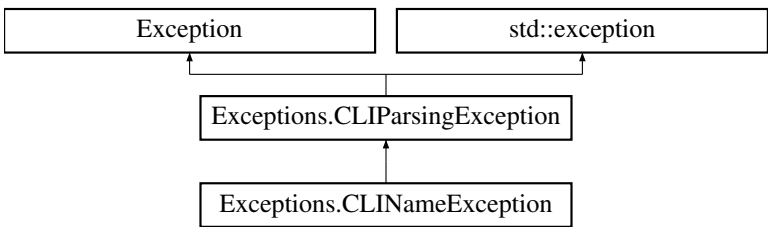
Reimplemented from Exceptions.CLIParsingException.

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

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

4.19 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

4.19.1 Constructor & Destructor Documentation

4.19.1.1 __init__()

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

Reimplemented from Exceptions.CLIParsingException.

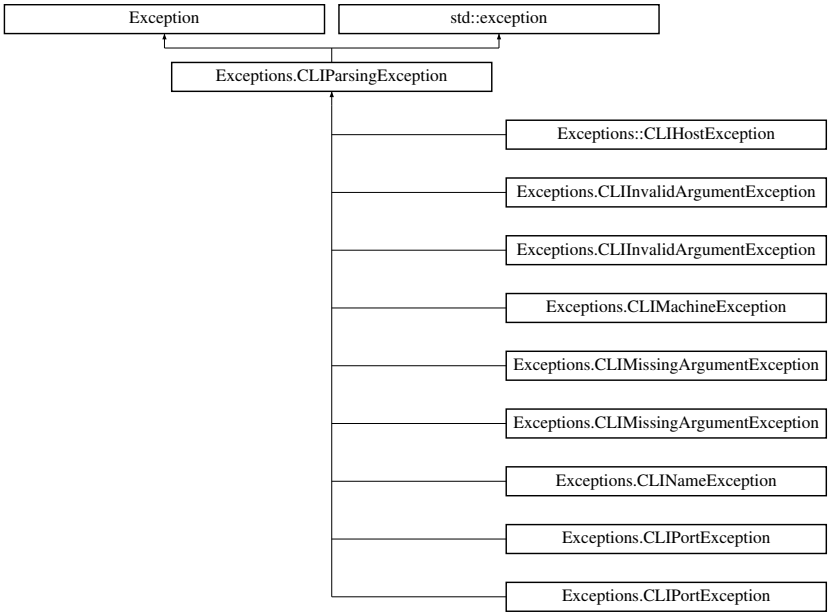
The documentation for this class was generated from the following file:

- ai/src/Exceptions/Exceptions.py

4.20 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**

4.20.1 Detailed Description

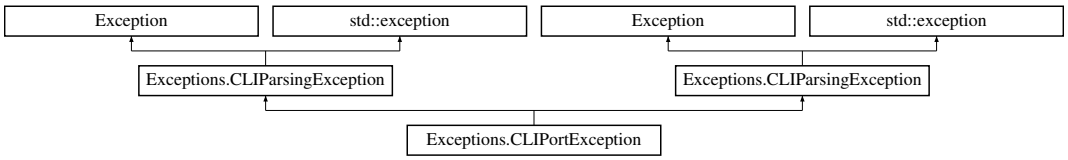
EPITECH PROJECT, 2025 zappy File description: Exceptions.

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

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

4.21 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

4.21.1 Constructor & Destructor Documentation

4.21.1.1 __init__()

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

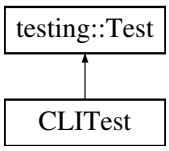
Reimplemented from Exceptions.CLIParsingException.

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

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

4.22 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

4.23 Color32 Struct Reference

Public Attributes

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

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

- gui/src/IDisplay.hpp

4.24 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

4.25 command_info_t Struct Reference

Public Attributes

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

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

- server/include/zappy.h

4.26 command_pf_s Struct Reference

Public Attributes

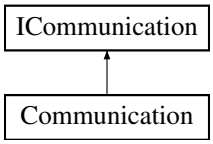
- char const * **flag**
- bool(* **checker**)(const char *, const char *, params_t *)

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

- server/include/zappy.h

4.27 Communication Class Reference

Inheritance diagram for Communication:



Public Member Functions

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

Private Member Functions

- void **setupConnection** ()
- void **createSocket** ()
- void **connectToServer** ()
- void **setupNonBlocking** ()
- void **startCommunicationThread** ()
- void **communicationLoop** ()
- bool **handlePoll** ()
- void **processWrite** ()
- void **processRead** ()
- void **parseReceivedData** ()

Private Attributes

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

Static Private Attributes

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

4.27.1 Member Function Documentation

4.27.1.1 disconnect()

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

Implements ICommunication.

4.27.1.2 hasMessages()

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

Implements ICommunication.

4.27.1.3 isConnected()

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

Implements ICommunication.

4.27.1.4 popMessage()

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

Implements ICommunication.

4.27.1.5 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

4.28 Communication.Communication Class Reference

Public Member Functions

- **__init__** (self, str name, str host, int port)
- **__del__** (self)
- None **stopLoop** (self)
- None **loop** (self)
- dict[str, int]|None **tryGetInventory** (self, str response)
- list[dict[str, int]]|None **tryGetLook** (self, str response)
- str **handleResponse** (self, str response)
- str **receiveData** (self)
- None **receive** (self)
- dict[str, int] **getInventory** (self)
- list[dict[str, int]] **getLook** (self)
- int **lenMessageQueue** (self)
- bool **hasMessages** (self)
- tuple[int, str] **getLastMessage** (self)
- int **lenResponseQueue** (self)
- bool **hasResponses** (self)
- None **addResponse** (self, str response)
- str **getLastResponse** (self)
- int **lenPendingQueue** (self)
- bool **hasPendingCommands** (self)
- int **lenRequestQueue** (self)
- bool **playerIsDead** (self)

- **connectToServer** (self)
- None **sendCommand** (self, str message)
- **sendForward** (self)
- **sendRight** (self)
- **sendLeft** (self)
- None **sendLook** (self)
- None **sendInventory** (self)
- **sendBroadcast** (self, str message)
- None **sendGetConnectNbr** (self)
- **sendFork** (self)
- **sendEject** (self)
- **sendTakeObject** (self, str object_name)
- **sendSetObject** (self, str object_name)
- **sendIncantation** (self)

Public Attributes

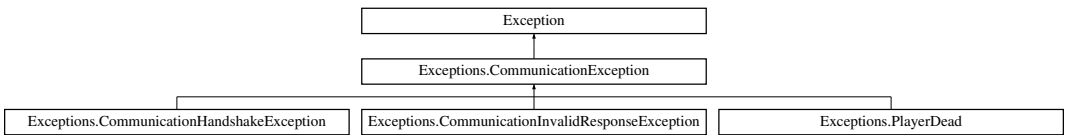
- **name**
- **host**
- **port**
- **socket**
- **mutex**
- **logger**
- **playerDead**
- **lastInventory**
- **lastLook**
- **responseBuffer**
- **messageQueue**
- **responseQueue**
- **pendingQueue**
- **requestQueue**

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

- ai/src/Communication/Communication.py

4.29 Exceptions.CommunicationException Class Reference

Inheritance diagram for Exceptions.CommunicationException:



Public Member Functions

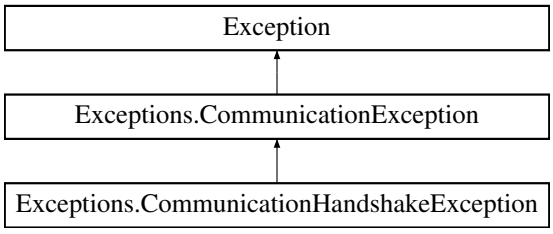
- **__init__** (self, str message)

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

- ai/src/Exceptions/Exceptions.py

4.30 Exceptions.CommunicationHandshakeException Class Reference

Inheritance diagram for Exceptions.CommunicationHandshakeException:



Public Member Functions

- __init__ (self, str message)

4.30.1 Constructor & Destructor Documentation

4.30.1.1 __init__()

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

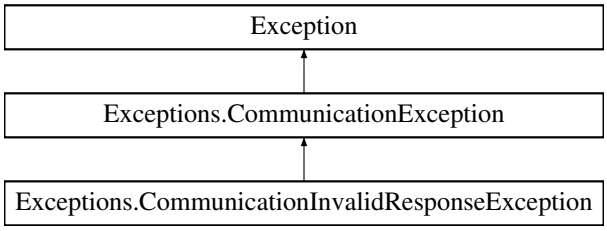
Reimplemented from Exceptions.CommunicationException.

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

- ai/src/Exceptions/Exceptions.py

4.31 Exceptions.CommunicationInvalidResponseException Class Reference

Inheritance diagram for Exceptions.CommunicationInvalidResponseException:



Public Member Functions

- __init__ (self, str message)

4.31.1 Constructor & Destructor Documentation

4.31.1.1 __init__()

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

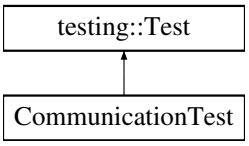
Reimplemented from Exceptions.CommunicationException.

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

- ai/src/Exceptions/Exceptions.py

4.32 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

4.33 zappy::structs::Config Struct Reference

Public Attributes

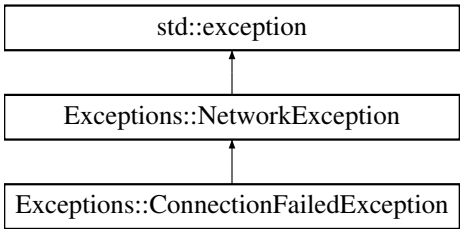
- int **port**
- std::string **hostname**

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

- gui/src/Utils/Constants.hpp

4.34 Exceptions::ConnectionFailedException Class Reference

Inheritance diagram for Exceptions::ConnectionFailedException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

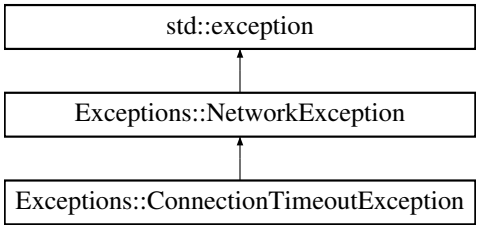
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

4.35 Exceptions::ConnectionTimeoutException Class Reference

Inheritance diagram for Exceptions::ConnectionTimeoutException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

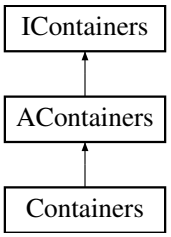
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

4.36 Containers Class Reference

Inheritance diagram for Containers:



Public Member Functions

- **Containers** (std::shared_ptr< IDisplay > display, std::shared_ptr< IAudio > audio, float x, float y, float width, float height, Color32 backgroundColor={40, 40, 40, 200})
- void draw () override
- void update () override
- void **setBackgroundColor** (Color32 color)
- bool **addElement** (const std::string &id, std::shared_ptr< IUIElement > element)
- std::shared_ptr< IUIElement > **getElement** (const std::string &id) const
- bool **removeElement** (const std::string &id)
- std::shared_ptr< Button > **addButton** (const std::string &id, float x, float y, float width, float height, const std::string &text, std::function< void()> callback)
- std::shared_ptr< Button > **addButton** (const std::string &id, float x, float y, float width, float height, const std::string &text, std::function< void()> callback, Color32 normalColor, Color32 hoverColor, Color32 pressedColor, Color32 textColor)
- std::shared_ptr< Text > **addText** (const std::string &id, float x, float y, const std::string &text, float fontSize=20.0f, Color32 color=CBLACK)
- std::shared_ptr< Slider > **addSlider** (const std::string &id, float x, float y, float width, float height, float minValue, float maxValue, float initialValue, const std::string &text, std::function< void(float)> onValueChanged)
- std::shared_ptr< Slider > **addSliderPercent** (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, float minValue, float maxValue, float initialValue, const std::string &text, std::function< void(float)> onValueChanged)
- void **clearElements** ()
- void **handleResize** (int oldWidth, int oldHeight, int newWidth, int newHeight)
- std::shared_ptr< Button > **addButtonPercent** (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &text, std::function< void()> callback)
- std::shared_ptr< Button > **addButtonPercent** (const std::string &id, float xPercent, float yPercent, float widthPercent, float heightPercent, const std::string &text, std::function< void()> callback, Color32 normalColor, Color32 hoverColor, Color32 pressedColor, Color32 textColor)
- std::shared_ptr< Text > **addTextPercent** (const std::string &id, float xPercent, float yPercent, const std::string &text, float fontSizePercent=5.0f, Color32 color=CBLACK)

Public Member Functions inherited from AContainers

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

Private Attributes

- std::shared_ptr< IAudio > **_audio**
- std::unordered_map< std::string, std::shared_ptr< IUIElement > > **_elements**

Additional Inherited Members

Protected Attributes inherited from AContainers

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

4.36.1 Member Function Documentation

4.36.1.1 draw()

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

Implements IContainers.

4.36.1.2 update()

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

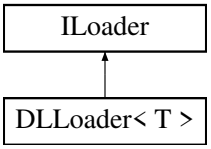
Implements IContainers.

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

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

4.37 DLLoader< T > Class Template Reference

Inheritance diagram for DLLoader< T >:



Public Member Functions

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

Private Attributes

- void * **_handler** = nullptr

4.37.1 Member Function Documentation

4.37.1.1 Close()

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

Implements ILoader.

4.37.1.2 Error()

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

Implements ILoader.

4.37.1.3 getHandler()

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

Implements ILoader.

4.37.1.4 Open()

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

Implements ILoader.

4.37.1.5 Symbol()

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

Implements ILoader.

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

- gui/src/DLLoader/DLLoader.hpp

4.38 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

4.39 egg_s Struct Reference

Public Attributes

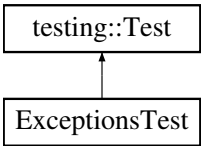
- int **id**
- int **posX**
- int **posY**
- char * **teamName**
- int **idLayer**
- bool **isHatched**
- struct egg_s * **next**

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

- server/include/game.h

4.40 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

4.41 FloatRect Struct Reference

Public Attributes

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

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

- gui/src/IDisplay.hpp

4.42 game_s Struct Reference

Public Attributes

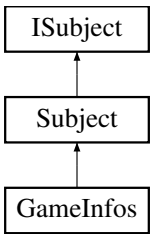
- team_t * **teams**
- map_t * **map**

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

- server/include/game.h

4.43 GameInfos Class Reference

Inheritance diagram for GameInfos:



Public Member Functions

- **GameInfos** (std::shared_ptr< ICommunication > communication)
- void **setMapSize** (int width, int height)
- std::pair< int, int > **getMapSize** () const
- void **setTimeUnit** (int timeUnit, bool sendToServer=false)
- int **getTimeUnit** () const
- void **updateTile** (const zappy::structs::Tile tile)
- const 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
- const zappy::structs::Player **getPlayer** (int playerNumber) const
- void **addPlayerBroadcast** (int playerNumber, const std::string &message)
- const std::vector< std::pair< int, std::string > > **getPlayersBroadcasting** ()
- void **addIncantation** (const zappy::structs::Incantation incantation)
- void **removeIncantation** (int x, int y, int result)
- 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

Public Member Functions inherited from Subject

- void addObserver (std::shared_ptr< IObserver > observer) override
- void removeObserver (std::shared_ptr< IObserver > observer) override
- void notifyObservers () override

Private Member Functions

- void **notifyStateChange** ()

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::tuple< int, std::string, std::chrono::steady_clock::time_point > > **_playersBroadcasting**
- std::vector< zappy::structs::Incantation > **_incantations**
- std::vector< zappy::structs::Egg > **_eggs**
- bool **_gameOver**
- std::string **_winningTeam**
- std::mutex **_dataMutex**
- std::shared_ptr< ICommunication > **_communication**

Additional Inherited Members

Protected Attributes inherited from ISubject

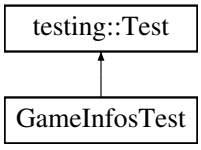
- std::vector< std::weak_ptr< IObserver > > **_observers**

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

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

4.44 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

4.45 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

4.46 GUI Class Reference

Public Member Functions

- **GUI** (std::shared_ptr< GameInfos > gameInfos, const std::string &libPath)
- void **run** ()
- void **refresh** ()
- int **getWindowWidth** () const
- int **getWindowHeight** () const
- void **setWindowWidth** (int width)
- void **setWindowHeight** (int height)
- void **switchCameraMode** (zappy::gui::CameraMode mode)
- void **switchCameraModeNext** ()
- void **setPlayerToFollow** (int playerId)
- int **getPlayerToFollow** () const
- bool **selectFirstAvailablePlayer** ()
- void **switchToNextPlayer** ()
- void **switchToPreviousPlayer** ()

Private Member Functions

- void **updateCamera** ()
- virtual void **update** ()
- virtual void **draw** ()
- virtual bool **isRunning** ()
- bool **playerExists** (int playerId) const
- void **initModels** ()

Private Attributes

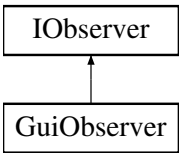
- std::string **_currentLibLoaded**
- bool **_isRunning**
- DLLoader< std::shared_ptr< IDisplay > > **_dlLoader**
- std::shared_ptr< IDisplay > **_display**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::unique_ptr< Map > **_map**
- std::unique_ptr< HUD > **_hud**
- std::shared_ptr< IAudio > **_audio**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

4.47 GuiObserver Class Reference

Inheritance diagram for GuiObserver:



Public Member Functions

- **GuiObserver** (std::shared_ptr< GUI > gui)
- void update () override

Private Attributes

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

4.47.1 Member Function Documentation

4.47.1.1 update()

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

Implements IObserver.

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

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

4.48 Hash.Hash Class Reference

Public Member Functions

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

Public Attributes

- **key**

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

- ai/src/Hash/Hash.py

4.49 Help Class Reference

Public Member Functions

- **Help** (std::shared_ptr< IDisplay > display, std::shared_ptr< IAudio > audio)
- void **show** ()
- void **hide** ()
- bool **isVisible** () const
- void **update** ()
- void **draw** ()
- void **handleResize** (int oldWidth, int oldHeight, int newWidth, int newHeight)

Private Member Functions

- void **initHelpContainer** ()

Private Attributes

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

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

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

4.50 HUD Class Reference

Public Member Functions

- **HUD** (std::shared_ptr< IDisplay > display, std::shared_ptr< GameInfos > gameInfos, std::shared_ptr< IAudio > audio, std::function< void()> resetCameraFunc=nullptr)
- void **draw** ()
- void **update** ()
- std::shared_ptr< Containers > **addContainer** (const std::string &id, float x, float y, float width, float height, Color32 backgroundColor={40, 40, 40, 200})
- std::shared_ptr< Containers > **getContainer** (const std::string &id) const
- bool **removeContainer** (const std::string &id)
- void **handleResize** (int oldWidth, int oldHeight, int newWidth, int newHeight)
- void **clearAllContainers** ()
- void **initDefaultLayout** (float sideWidthPercent=15.0f, float bottomHeightPercent=20.0f)
- std::shared_ptr< Containers > **getSideContainer** () const
- std::shared_ptr< Containers > **getBottomContainer** () const
- std::shared_ptr< Containers > **getSquareContainer** () const
- std::shared_ptr< Containers > **getTpsContainer** () const
- void **initExitButton** ()
- void **initSettingsButton** ()
- void **initHelpButton** ()
- void **initCameraResetButton** ()
- void **initTeamPlayersDisplay** (std::shared_ptr< GameInfos > gameInfos)
- void **updateTeamPlayersDisplay** (std::shared_ptr< GameInfos > gameInfos)
- void **initTpsSlider** (std::shared_ptr< GameInfos > gameInfos, std::shared_ptr< IDisplay > raylib, std::shared_ptr< IAudio > audio)
- void **updateTpsSlider** (std::shared_ptr< GameInfos > gameInfos)
- void **initPlayerInventoryDisplay** (int playerId)
- void **updatePlayerInventoryDisplay** (int playerId, zappy::gui::CameraMode cameraMode)
- void **clearPlayerInventoryElements** ()
- zappy::structs::Player **getPlayerById** (int playerId) const
- void **setResetCameraCallback** (std::function< void()> resetFunc)

Private Member Functions

- std::shared_ptr< Containers > **createSquareContainer** (float squareSize, float sideWidthPercent)
- std::shared_ptr< Containers > **createSideContainer** (float sideYStart, float sideWidth, float sideHeight, float sideWidthPercent, float bottomHeightPercent)
- std::shared_ptr< Containers > **createBottomContainer** (int screenWidth, int screenHeight, float bottom↩ Height, float bottomHeightPercent)
- std::shared_ptr< Containers > **createTpsContainer** (int screenWidth, int screenHeight, float bottomHeight, float bottomHeightPercent)
- void **updateElementPositions** (std::shared_ptr< Containers > container, const std::unordered_map< std↩ ::string, float > &initialYPositions, float offset)
- std::pair< float, float > **calculateContentMetrics** (std::shared_ptr< Containers > container, const std↩ ::unordered_map< std::string, float > &initialYPositions)
- void **clearTeamDisplayElements** (std::shared_ptr< Containers > container)
- std::vector< int > **getTeamPlayerNumbers** (const std::string &teamName, const std::vector< zappy::structs::Player > &players)
- std::string **createPlayerListText** (const std::vector< int > &playerNumbers)
- void **addPlayerListText** (std::shared_ptr< Containers > container, const std::string &teamId, float yPos, const std::vector< int > &playerNumbers)

Private Attributes

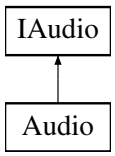
- std::unordered_map< std::string, std::shared_ptr< Containers > > **_containers**
- std::shared_ptr< IDisplay > **_display**
- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< IAudio > **_audio**
- std::shared_ptr< Help > **_help**
- std::function< void()> **_resetCameraFunc**

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

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

4.51 IAudio Class Reference

Inheritance diagram for IAudio:



Public Member Functions

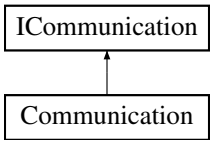
- virtual bool **loadSound** (const std::string &id, const std::string &filepath)=0
- virtual void **playSound** (const std::string &id, float volume=1.0f)=0
- virtual void **stopSound** (const std::string &id)=0
- virtual bool **isSoundPlaying** (const std::string &id) const =0
- virtual void **setSoundLooping** (const std::string &id, bool looping)=0
- virtual void **setSoundVolume** (const std::string &id, float volume)=0

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

- gui/src/Audio/IAudio.hpp

4.52 ICommunication Class Reference

Inheritance diagram for ICommunication:



Public Member Functions

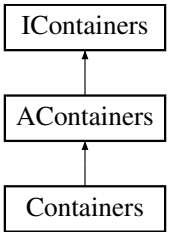
- virtual void **sendMessage** (const std::string &message)=0
- virtual bool **hasMessages** () const =0
- virtual std::string **popMessage** ()=0
- virtual bool **isConnected** () const =0
- virtual void **disconnect** ()=0

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

- gui/src/Communication/ICommunication.hpp

4.53 IContainers Class Reference

Inheritance diagram for IContainers:



Public Member Functions

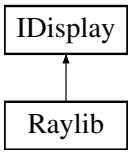
- virtual void **draw** ()=0
- virtual void **update** ()=0
- virtual void **setPosition** (float x, float y)=0
- virtual void **setSize** (float width, float height)=0
- virtual FloatRect **getBounds** () const =0
- virtual bool **contains** (float x, float y) const =0
- virtual void **setVisible**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

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

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

4.54 IDisplay Class Reference

Inheritance diagram for IDisplay:



Public Member Functions

- virtual Vector2i **getMonitorSize** ()=0
- virtual Vector2i **getScreenSize** ()=0
- virtual void **initWindow** (int width, int height, std::string)=0
- virtual void **initCamera** ()=0
- virtual bool **isWindowReady** ()=0
- virtual void **setTargetFPS** (unsigned int FPS)=0
- virtual bool **isOpen** ()=0
- virtual void **closeWindow** ()=0
- virtual int **getKeyId** (enum Key)=0
- virtual bool **isKeyReleased** (int key)=0
- virtual bool **isKeyPressed** (int key)=0
- virtual bool **isKeyDown** (int key)=0
- virtual bool **isGamepadAvailable** ()=0
- virtual bool **isGamepadButtonReleased** (int key)=0
- virtual bool **isGamepadButtonPressed** (int key)=0
- virtual bool **isGamepadButtonDown** (int key)=0
- virtual bool **isMouseButtonDown** (int key)=0
- virtual bool **isMouseButtonReleased** (int key)=0
- virtual bool **isMouseButtonPressed** (int key)=0
- virtual Vector2f **getMousePosition** ()=0
- virtual void **setMousePosition** (Vector2f)=0
- virtual float **getMouseWheelMove** ()=0
- virtual float **getGamepadAxisMovement** (int key)=0
- virtual void **setCameraPosition** (Vector3f)=0
- virtual void **setCameraTarget** (Vector3f)=0
- virtual Vector2f **getMouseDelta** ()=0
- virtual float **vector3DDistanceFromCamera** (Vector3f target)=0
- virtual Vector3f **vector3SubtractFromCamera** (Vector3f target)=0
- virtual Vector3f **vector3Normalize** (Vector3f)=0
- virtual void **enableCursor** ()=0
- virtual void **disableCursor** ()=0
- virtual float **getFrameTime** ()=0
- virtual void **updateCameraFreeMode** ()=0
- virtual float **measureText** (const std::string &text, float fontSize) const =0
- virtual bool **checkCollisionPointRec** (Vector2f point, FloatRect rec)=0
- virtual void **beginDrawing** ()=0
- virtual void **endDrawing** ()=0
- virtual void **clearBackground** (Color32)=0
- virtual void **begin3DMode** ()=0
- virtual void **end3DMode** ()=0
- virtual void **endScissorMode** ()=0

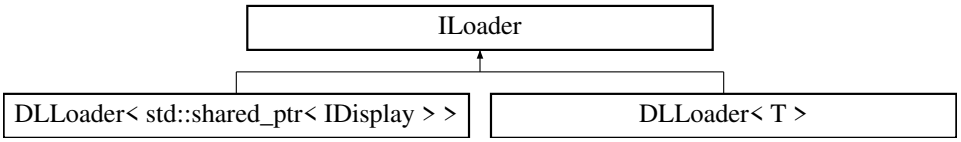
- virtual void **beginScissorMode** (IntRect)=0
- virtual bool **loadModel** (const std::string &id, const std::string &filepath, Vector3f center={0.0f, 0.0f, 0.0f})=0
- virtual void **drawCube** (Vector3f position, float width, float height, float length, Color32 color)=0
- virtual void **drawCubeWires** (Vector3f position, float width, float height, float length, Color32 color)=0
- virtual void **drawSphere** (Vector3f position, float radius, Color32 color)=0
- virtual void **drawSphereWires** (Vector3f position, float radius, int rings, int slices, Color32 color)=0
- virtual void **drawCylinder** (Vector3f position, float radiusTop, float radiusBottom, float height, int slices, Color32 color)=0
- virtual void **drawCylinderWires** (Vector3f position, float radiusTop, float radiusBottom, float height, int slices, Color32 color)=0
- virtual void **drawCylinderEx** (Vector3f startPos, Vector3f endPos, float startRadius, float endRadius, int sides, Color32 color)=0
- virtual void **drawPlane** (Vector3f position, Vector2f size, Color32 color)=0
- virtual void **drawLine3D** (Vector3f startPos, Vector3f endPos, Color32 color)=0
- virtual void **drawModelEx** (const std::string &id, Vector3f position, Vector3f rotationAxis, float rotationAngle, Vector3f scale, Color32 tint=CWHITE)=0
- virtual void **drawCircle** (float centerX, float centerY, float radius, Color32 color)=0
- virtual void **drawCircleLines** (float centerX, float centerY, float radius, Color32 color)=0
- virtual void **drawText** (const std::string &text, float x, float y, float fontSize, Color32 color)=0
- virtual void **drawRectangleRec** (FloatRect rec, Color32 color)=0

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

- gui/src/IDisplay.hpp

4.55 ILoader Class Reference

Inheritance diagram for ILoader:



Public Member Functions

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

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

- gui/src/DLLoader/ILoader.hpp

4.56 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

4.57 incantation_s Struct Reference

Public Attributes

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

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

- server/include/game.h

4.58 IntRect Struct Reference

Public Attributes

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

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

- gui/src/IDisplay.hpp

4.59 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

4.60 inventory_s Struct Reference

Public Attributes

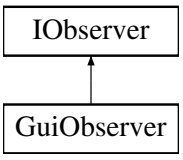
- int **nbFood**
- int **nbLinemate**
- int **nbDeraumere**
- int **nbSibur**
- int **nbMendiane**
- int **nbPhiras**
- int **nbThystame**

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

- server/include/game.h

4.61 IObserver Class Reference

Inheritance diagram for IObserver:



Public Member Functions

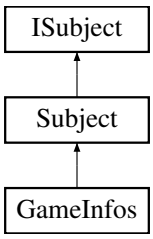
- virtual void **update** ()=0

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

- gui/src/Observer/IObserver.hpp

4.62 ISubject Class Reference

Inheritance diagram for ISubject:



Public Member Functions

- virtual void **addObserver** (std::shared_ptr< IObserver > observer)=0
- virtual void **removeObserver** (std::shared_ptr< IObserver > observer)=0
- virtual void **notifyObservers** ()=0

Protected Attributes

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

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

- gui/src/Observer/ISubject.hpp

4.63 item_handler_t Struct Reference

Public Attributes

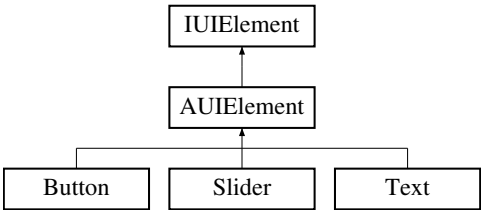
- char * **name**
- void(* **add_func**)(inventory_t *)

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

- server/include/zappy.h

4.64 IUIElement Class Reference

Inheritance diagram for IUIElement:



Public Member Functions

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

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

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

4.65 Logger.Logger Class Reference

Public Member Functions

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

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

- ai/src/Logger/Logger.py

4.66 Map Class Reference

Public Member Functions

- **Map** (std::shared_ptr< GameInfos > gameInfos, std::shared_ptr< IDisplay > display)
- void **draw** ()
- void **drawBroadcastingPlayers** ()
- 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)
- Color32 **getTeamColor** (const std::string &teamName)
- float **getOffset** (DisplayPriority priority, int x, int y, size_t stackIndex=0)

Private Member Functions

- void **drawOrientationArrow** (const Vector3f &position, int orientation, float playerHeight)
- void **drawTorus** (const Vector3f &position, float radius, float thickness, int radialSegments, Color32 color)

Private Attributes

- std::shared_ptr< GameInfos > **_gameInfos**
- std::shared_ptr< IDisplay > **_display**
- std::unordered_map< std::string, Color32 > **_teamColors**
- std::vector< Color32 > **_colors**
- int **_colorIndex** = 0
- std::unordered_map< int, std::chrono::steady_clock::time_point > **_broadcastStartTimes**

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

4.67 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

4.68 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

4.69 RayLibEnc::ModelData Struct Reference

Public Attributes

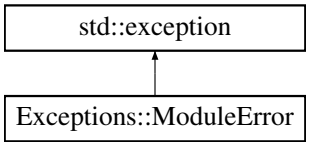
- Model **model**
- unsigned int **animationCount**
- Vector3 **center**

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

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

4.70 Exceptions::ModuleError Class Reference

Inheritance diagram for Exceptions::ModuleError:



Public Member Functions

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

Private Attributes

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

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

- gui/src/Exceptions/Exceptions.hpp

4.71 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

4.72 network_s Struct Reference

Public Attributes

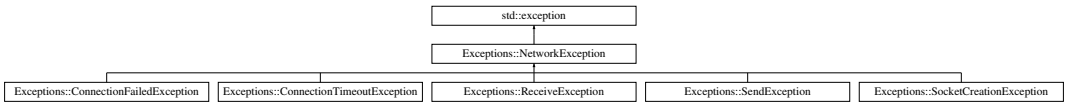
- int **fd**
- buffer_t * **buffer**

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

- server/include/game.h

4.73 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

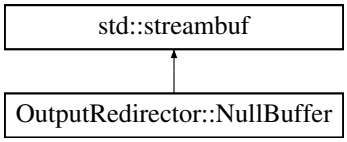
- std::string **_message**

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

- gui/src/Exceptions/Exceptions.hpp

4.74 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

4.75 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

4.76 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

4.77 Parser.Parser Class Reference

Public Member Functions

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

Public Attributes

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

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

- tests/functional/Parser.py

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

Public Attributes

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

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

- ai/src/Player/Player.py

4.79 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

4.80 player_s Struct Reference

Public Attributes

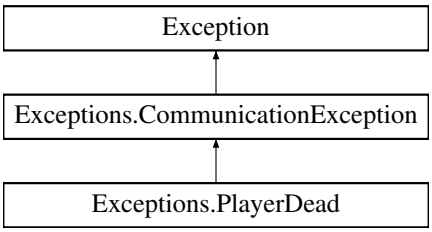
- int **id**
- network_t * **network**
- int **level**
- int **posX**
- int **posY**
- direction_t **direction**
- inventory_t * **inventory**
- char * **team**
- action_queue_t * **pending_actions**
- time_t **last_action_time**
- bool **is_busy**
- int **remaining_cooldown**
- char * **current_action**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

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

- server/include/game.h

4.81 Exceptions.PlayerDead Class Reference

Inheritance diagram for Exceptions.PlayerDead:



Public Member Functions

- __init__ (self)

4.81.1 Constructor & Destructor Documentation

4.81.1.1 __init__()

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

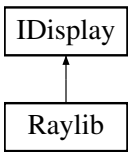
Reimplemented from Exceptions.CommunicationException.

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

- ai/src/Exceptions/Exceptions.py

4.82 Raylib Class Reference

Inheritance diagram for Raylib:



Public Member Functions

- virtual Vector2i getMonitorSize ()
- virtual Vector2i getScreenSize ()
- virtual void initWindow (int width, int height, std::string)
- virtual void initCamera ()
- virtual bool isWindowReady ()
- virtual void setTargetFPS (unsigned int FPS)
- virtual bool isOpen ()
- virtual void closeWindow ()
- virtual int getKeyId (enum Key)
- virtual bool isKeyReleased (int key)
- virtual bool isKeyPressed (int key)
- virtual bool isKeyDown (int key)
- virtual bool isGamepadAvailable ()
- virtual bool isGamepadButtonReleased (int key)
- virtual bool isGamepadButtonPressed (int key)
- virtual bool isGamepadButtonDown (int key)
- virtual bool isMouseButtonDown (int key)
- virtual bool isMouseButtonReleased (int key)
- virtual bool isMouseButtonPressed (int key)
- virtual Vector2f getMousePosition ()
- virtual void setMousePosition (Vector2f)
- virtual float getMouseWheelMove ()
- virtual float getGamepadAxisMovement (int key)
- virtual void setCameraPosition (Vector3f)
- virtual void setCameraTarget (Vector3f)
- virtual Vector2f getMouseDelta ()
- virtual float vector3DDistanceFromCamera (Vector3f target)
- virtual Vector3f vector3SubtractFromCamera (Vector3f target)
- virtual Vector3f vector3Normalize (Vector3f)
- virtual void enableCursor ()
- virtual void disableCursor ()
- virtual float getFrameTime ()
- virtual void updateCameraFreeMode ()
- virtual float measureText (const std::string &text, float fontSize) const
- virtual bool checkCollisionPointRec (Vector2f point, FloatRect rec)
- virtual void beginScissorMode (IntRect)
- virtual void endScissorMode ()
- virtual void beginDrawing ()
- virtual void endDrawing ()
- virtual void clearBackground (Color32)
- virtual void begin3DMode ()
- virtual void end3DMode ()
- virtual bool loadModel (const std::string &id, const std::string &filepath, Vector3f center={0.0f, 0.0f, 0.0f})
- virtual void drawCube (Vector3f position, float width, float height, float length, Color32 color)
- virtual void drawCubeWires (Vector3f position, float width, float height, float length, Color32 color)
- virtual void drawSphere (Vector3f position, float radius, Color32 color)
- virtual void drawSphereWires (Vector3f position, float radius, int rings, int slices, Color32 color)
- virtual void drawCylinder (Vector3f position, float radiusTop, float radiusBottom, float height, int slices, Color32 color)
- virtual void drawCylinderWires (Vector3f position, float radiusTop, float radiusBottom, float height, int slices, Color32 color)
- virtual void drawCylinderEx (Vector3f startPos, Vector3f endPos, float startRadius, float endRadius, int sides, Color32 color)

- virtual void drawPlane (Vector3f position, Vector2f size, Color32 color)
- virtual void drawLine3D (Vector3f startPos, Vector3f endPos, Color32 color)
- virtual void drawModelEx (const std::string &id, Vector3f position, Vector3f rotationAxis, float rotationAngle, Vector3f scale, Color32 tint=CWHITE)
- virtual void drawText (const std::string &text, float x, float y, float fontSize, Color32 color)
- virtual void drawCircle (float centerX, float centerY, float radius, Color32 color)
- virtual void drawCircleLines (float centerX, float centerY, float radius, Color32 color)
- virtual void drawRectangleRec (FloatRect rec, Color32 color)

Private Attributes

- std::unique_ptr< RayLibEnc > **_raylib**

4.82.1 Member Function Documentation

4.82.1.1 begin3DMode()

```
void Raylib::begin3DMode ( )  [virtual]
```

Implements IDisplay.

4.82.1.2 beginDrawing()

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

Implements IDisplay.

4.82.1.3 beginScissorMode()

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

Implements IDisplay.

4.82.1.4 checkCollisionPointRec()

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

Implements IDisplay.

4.82.1.5 clearBackground()

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

Implements IDisplay.

4.82.1.6 closeWindow()

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

Implements IDisplay.

4.82.1.7 disableCursor()

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

Implements IDisplay.

4.82.1.8 drawCircle()

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

Implements IDisplay.

4.82.1.9 drawCircleLines()

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

Implements IDisplay.

4.82.1.10 drawCube()

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

Implements IDisplay.

4.82.1.11 drawCubeWires()

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

Implements IDisplay.

4.82.1.12 drawCylinder()

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

Implements IDisplay.

4.82.1.13 drawCylinderEx()

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

Implements IDisplay.

4.82.1.14 drawCylinderWires()

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

Implements IDisplay.

4.82.1.15 drawLine3D()

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

Implements IDisplay.

4.82.1.16 drawModelEx()

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

Implements IDisplay.

4.82.1.17 drawPlane()

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

Implements IDisplay.

4.82.1.18 drawRectangleRec()

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

Implements IDisplay.

4.82.1.19 drawSphere()

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

Implements IDisplay.

4.82.1.20 drawSphereWires()

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

Implements IDisplay.

4.82.1.21 drawText()

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

Implements IDisplay.

4.82.1.22 enableCursor()

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

Implements IDisplay.

4.82.1.23 end3DMode()

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

Implements IDisplay.

4.82.1.24 endDrawing()

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

Implements IDisplay.

4.82.1.25 endScissorMode()

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

Implements IDisplay.

4.82.1.26 getFrameTime()

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

Implements IDisplay.

4.82.1.27 getGamepadAxisMovement()

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

Implements IDisplay.

4.82.1.28 getKeyId()

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

Implements IDisplay.

4.82.1.29 getMonitorSize()

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

Implements IDisplay.

4.82.1.30 getMouseDelta()

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

Implements IDisplay.

4.82.1.31 getMousePosition()

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

Implements IDisplay.

4.82.1.32 getMouseWheelMove()

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

Implements IDisplay.

4.82.1.33 getScreenSize()

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

Implements IDisplay.

4.82.1.34 initCamera()

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

Implements IDisplay.

4.82.1.35 initWindow()

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

Implements IDisplay.

4.82.1.36 isGamepadAvailable()

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

Implements IDisplay.

4.82.1.37 isGamepadButtonDown()

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

Implements IDisplay.

4.82.1.38 isGamepadButtonPressed()

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

Implements IDisplay.

4.82.1.39 isGamepadButtonReleased()

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

Implements IDisplay.

4.82.1.40 isKeyDown()

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

Implements IDisplay.

4.82.1.41 isKeyPressed()

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

Implements IDisplay.

4.82.1.42 isKeyReleased()

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

Implements IDisplay.

4.82.1.43 isMouseButtonDown()

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

Implements IDisplay.

4.82.1.44 isMouseButtonPressed()

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

Implements IDisplay.

4.82.1.45 isMouseButtonReleased()

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

Implements IDisplay.

4.82.1.46 isOpen()

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

Implements IDisplay.

4.82.1.47 isWindowReady()

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

Implements IDisplay.

4.82.1.48 loadModel()

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

Implements IDisplay.

4.82.1.49 measureText()

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

Implements IDisplay.

4.82.1.50 setCameraPosition()

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

Implements IDisplay.

4.82.1.51 setCameraTarget()

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

Implements IDisplay.

4.82.1.52 setMousePosition()

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

Implements IDisplay.

4.82.1.53 setTargetFPS()

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

Implements IDisplay.

4.82.1.54 updateCameraFreeMode()

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

Implements IDisplay.

4.82.1.55 vector3DDistanceFromCamera()

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

Implements IDisplay.

4.82.1.56 vector3Normalize()

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

Implements IDisplay.

4.82.1.57 vector3SubtractFromCamera()

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

Implements IDisplay.

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

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

4.83 RayLibEnc Class Reference

Classes

- struct ModelData

Public Member Functions

- void **initWindow** (int width, int height, const std::string &title)
- void **closeWindow** ()
- bool **windowShouldClose** () const
- void **beginDrawing** ()
- void **endDrawing** ()
- void **clearBackground** (Color color=WHITE)
- bool **isWindowReady** () const
- int **getMonitorWidth** (int monitor) const
- int **getMonitorHeight** (int monitor) const
- void **waitTime** (float seconds) const
- void **setTargetFPS** (int fps) const
- int **getFPS** () const
- float **getFrameTime** () const
- bool **checkCollisionPointRec** (Vector2 point, Rectangle rec) const
- 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
- bool **isGamepadAvailable** (int gamepad) const

- bool **isGamepadButtonPressed** (int gamepad, int button) const
- bool **isGamepadButtonDown** (int gamepad, int button) const
- bool **isGamepadButtonReleased** (int gamepad, int button) const
- float **getGamepadAxisMovement** (int gamepad, int axis) const
- 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)
- void **drawCircle** (float centerX, float centerY, float radius, Color color)
- void **drawCircleLines** (float centerX, float centerY, float radius, Color color)
- float **measureText** (const std::string &text, float fontSize) const

Private Attributes

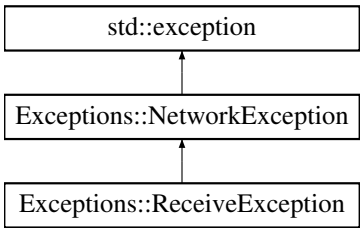
- bool **_isInitialized**
- Camera3D **_camera**
- Vector2 **_previousMousePosition**
- bool **_isCursorLocked**
- std::map< std::string, ModelData > **_models**
- std::map< std::string, Sound > **_sounds**
- std::map< std::string, Music > **_musics**

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

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

4.84 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

4.85 RelativePosition Struct Reference

Public Attributes

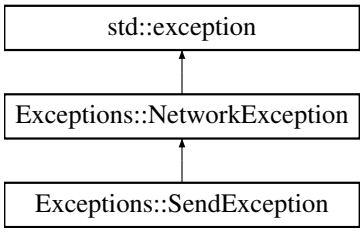
- float **xPercent**
- float **yPercent**
- float **widthPercent**
- float **heightPercent**

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

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

4.86 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

4.87 server_s Struct Reference

Public Attributes

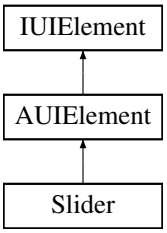
- int **sockfd**
- struct pollfd **pollserver**

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

- server/include/zappy.h

4.88 Slider Class Reference

Inheritance diagram for Slider:



Public Member Functions

- **Slider** (std::shared_ptr< IDisplay > raylib, float x, float y, float width, float height, float minValue, float max↩ Value, float initialValue, const std::string &text, std::function< void(float)> onValueChanged)
- void draw () override
- void update () override
- bool **isDragging** () const
- void **setValue** (float value)
- float **getValue** () const
- void **setMinValue** (float minValue)
- void **setMaxValue** (float maxValue)
- float **getMinValue** () const
- float **getMaxValue** () const
- void **setText** (const std::string &text)
- std::string **getText** () const
- void setSize (float width, float height) override

Public Member Functions inherited from AUIElement

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

Private Member Functions

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

Private Attributes

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

Additional Inherited Members

Protected Attributes inherited from AUIElement

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

4.88.1 Member Function Documentation

4.88.1.1 draw()

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

Implements IUIElement.

4.88.1.2 setSize()

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

Reimplemented from AUIElement.

4.88.1.3 update()

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

Implements IUIElement.

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

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

4.89 Socket.Socket Class Reference

Public Member Functions

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

Protected Attributes

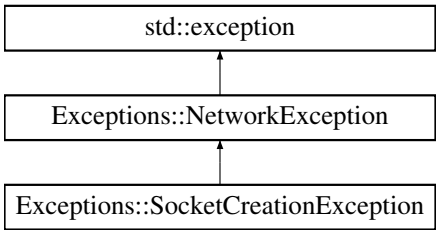
- **_host**
- **_port**
- **_address**
- **_socket**

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

- ai/src/Communication/Socket.py

4.90 Exceptions::SocketCreationException Class Reference

Inheritance diagram for Exceptions::SocketCreationException:



Public Member Functions

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

Public Member Functions inherited from Exceptions::NetworkException

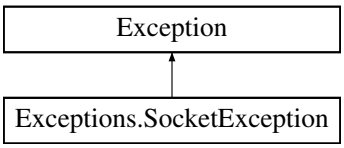
- **NetworkException** (const std::string &message)
- const char * **what** () const noexcept override

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

- gui/src/Exceptions/Exceptions.hpp

4.91 Exceptions.SocketException Class Reference

Inheritance diagram for Exceptions.SocketException:



Public Member Functions

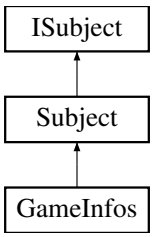
- **__init__** (self, str message)

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

- ai/src/Exceptions/Exceptions.py

4.92 Subject Class Reference

Inheritance diagram for Subject:



Public Member Functions

- void addObserver (std::shared_ptr< IObserver > observer) override
- void removeObserver (std::shared_ptr< IObserver > observer) override
- void notifyObservers () override

Private Attributes

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

Additional Inherited Members

Protected Attributes inherited from ISubject

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

4.92.1 Member Function Documentation

4.92.1.1 addObserver()

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

Implements ISubject.

4.92.1.2 notifyObservers()

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

Implements ISubject.

4.92.1.3 removeObserver()

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

Implements ISubject.

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

- gui/src/Observer/Subject.hpp

4.93 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

4.94 TestCase.TestCase Class Reference

Public Member Functions

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

Public Attributes

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

Protected Member Functions

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

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

- tests/functional/TestCase.py

4.95 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)

4.95.1 Member Function Documentation

4.95.1.1 test_parse_args_invalid_option()

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

Test parsing invalid option

4.95.1.2 test_parse_args_missing_value()

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

Test parsing missing value for option

4.95.1.3 test_parse_args_not_enough_args()

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

Test parsing not enough arguments

4.95.1.4 test_parse_args_valid()

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

Test parsing valid command line arguments

4.95.1.5 test_parse_args_valid_ip()

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

Test parsing valid IP address

4.95.1.6 test_parse_machine_empty()

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

Test parsing empty machine name

4.95.1.7 test_parse_machine_invalid_ip_chars()

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

Test parsing IP with invalid characters

4.95.1.8 test_parse_machine_invalid_ip_format()

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

Test parsing invalid IP format

4.95.1.9 test_parse_machine_invalid_ip_value()

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

Test parsing invalid IP value

4.95.1.10 test_parse_name_empty()

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

Test parsing empty team name

4.95.1.11 test_parse_name_whitespace()

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

Test parsing whitespace team name

4.95.1.12 test_parse_port_invalid()

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

Test parsing invalid port

4.95.1.13 test_parse_port_negative()

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

Test parsing negative port

4.95.1.14 test_parse_port_too_large()

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

Test parsing port that is too large

4.95.1.15 test_validate_config_missing_name()

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

Test validating config with missing name

4.95.1.16 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

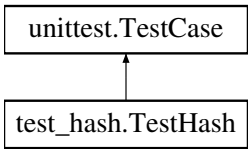
4.96 test_com.TestCommunication Class Reference

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

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

4.97 test_hash.TestHash Class Reference

Inheritance diagram for test_hash.TestHash:



Public Member Functions

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

Public Attributes

- **hash_obj**

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

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

4.98 test_player.TestPlayer Class Reference

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

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

4.99 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)

4.99.1 Member Function Documentation

4.99.1.1 test_socket_close()

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

Test socket close

4.99.1.2 test_socket_connect_failure()

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

Test socket connection failure

4.99.1.3 test_socket_connect_success()

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

Test successful socket connection

4.99.1.4 test_socket_different_hosts_and_ports()

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

Test socket creation with different hosts and ports

4.99.1.5 test_socket_init()

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

Test socket initialization

4.99.1.6 test_socket_receive_connection_closed()

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

Test handling closed connection during receive
```

4.99.1.7 test_socket_receive_unicode()

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

Test receiving unicode messages
```

4.99.1.8 test_socket_send_success()

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

Test successful message sending
```

4.99.1.9 test_socket_send_unicode()

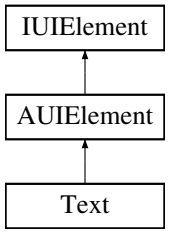
```
test_socket.TestSocket.test_socket_send_unicode (
              self,
              mock_socket )

Test sending unicode messages
```

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

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

4.100 Text Class Reference

Inheritance diagram for Text:



Public Member Functions

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

Public Member Functions inherited from AUIElement

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

Private Attributes

- std::string **_text**
- float **_fontSize**
- Color32 **_color**
- std::shared_ptr< IDisplay > **_display**

Additional Inherited Members

Protected Attributes inherited from AUIElement

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

4.100.1 Member Function Documentation

4.100.1.1 draw()

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

Implements IUIElement.

4.100.1.2 setSize()

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

Reimplemented from AUIElement.

4.100.1.3 update()

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

Implements IUIElement.

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

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

4.101 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

4.102 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

4.103 UIRelativePosition Struct Reference

Public Attributes

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

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

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

4.104 Vector2f Struct Reference

Public Attributes

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

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

- gui/src/IDisplay.hpp

4.105 Vector2i Struct Reference

Public Attributes

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

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

- gui/src/IDisplay.hpp

4.106 Vector3f Struct Reference

Public Attributes

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

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

- gui/src/IDisplay.hpp

4.107 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 5

File Documentation

5.1 Audio.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Audio
00006 */
00007
00008 #ifndef AUDIO_HPP_
00009 #define AUDIO_HPP_
00010
00011 #include <string>
00012 #include <map>
00013 #include <memory>
00014 #include <SFML/Audio.hpp>
00015 #include "IAudio.hpp"
00016
00017 class Audio : public IAudio {
00018     public:
00019         Audio();
00020         ~Audio();
00021
00022         bool loadSound(const std::string& id, const std::string& filepath);
00023
00024         void playSound(const std::string& id, float volume = 1.0f);
00025         void stopSound(const std::string& id);
00026         bool isSoundPlaying(const std::string& id) const;
00027
00028         void setSoundLooping(const std::string& id, bool looping);
00029         void setSoundVolume(const std::string& id, float volume);
00030
00031     private:
00032         std::map<std::string, std::unique_ptr<sf::Music» _sounds;
00033 };
00034
00035 #endif /* !AUDIO_HPP_ */
```

5.2 IAudio.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** IAudio
00006 */
00007
00008 #ifndef IAUDIO_HPP_
00009 #define IAUDIO_HPP_
00010
00011 #include <string>
00012
00013 class IAudio {
00014     public:
00015         virtual ~IAudio() = default;
00016
```

```
00017         virtual bool loadSound(const std::string& id, const std::string& filepath) = 0;
00018
00019         virtual void playSound(const std::string& id, float volume = 1.0f) = 0;
00020         virtual void stopSound(const std::string& id) = 0;
00021         virtual bool isSoundPlaying(const std::string& id) const = 0;
00022
00023         virtual void setSoundLooping(const std::string& id, bool looping) = 0;
00024         virtual void setSoundVolume(const std::string& id, float volume) = 0;
00025 };
00026
00027 #endif /* !IAUDIO_HPP_ */
```

5.3 CLI.hpp

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

5.4 Client.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Client
00006 */
00007
00008 #ifndef CLIENT_HPP_
00009 #define CLIENT_HPP_
00010
00011 #include <memory>
00012 #include <filesystem>
00013 #include <string>
00014 #include "../Utils/Constants.hpp"
00015 #include "../Communication/ICommunication.hpp"
00016 #include "../Game/GameInfos.hpp"
00017 #include "../Graphic/GUI.hpp"
00018 #include "MsgHandler.hpp"
00019 #include "../Observer/GuiObserver.hpp"
00020 #include "../Observer/IObserver.hpp"
00021
00022 class Client {
00023     public:
00024         Client(int ac, const char *const *av);
00025         ~Client();
00026
00027     private:
00028         void _tryToCreateGuiWithSharedLibInFolder(const std::string &libPath = "./gui/lib/");
00029         zappy::structs::Config _config;
00030         void initialize(int ac, const char * const *av);
00031
```

```
00032         std::shared_ptr<ICommunication> _communication;
00033         std::shared_ptr<GameInfos> _gameInfos;
00034         std::unique_ptr<MsgHandler> _msgHandler;
00035         std::unique_ptr<GUI> _gui;
00036         std::shared_ptr<GuiObserver> _guiObserver;
00037 };
00038
00039 #endif /* !CLIENT_HPP_ */
```

5.5 MsgHandler.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** MsgHandler
00006 */
00007
00008 #ifndef MSGHANDLER_HPP_
00009 #define MSGHANDLER_HPP_
00010
00011 #include <memory>
00012 #include <map>
00013 #include <functional>
00014 #include <thread>
00015 #include <mutex>
00016 #include <atomic>
00017 #include <queue>
00018 #include <condition_variable>
00019 #include <string>
00020
00021 #include "../Game/GameInfos.hpp"
00022 #include "../Communication/ICommunication.hpp"
00023 #include "../Utils/Constants.hpp"
00024
00025 class MsgHandler {
00026     public:
00027         MsgHandler(std::shared_ptr<GameInfos> gameInfos,
00028             std::shared_ptr<ICommunication> communication);
00029         ~MsgHandler();
00030
00031         void start();
00032         void stop();
00033
00034     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_ */
```

5.6 Communication.hpp

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

5.7 ICommunication.hpp

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

5.8 DLLoader.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** DLLoader
00006 */
00007
00008 #ifndef DLLOADER_HPP_
00009 #define DLLOADER_HPP_
00010
00011 #include <dlfcn.h>
00012 #include <iostream>
00013 #include <ostream>
00014 #include <memory>
00015 #include "ILoader.hpp"
00016
00017 template <typename T>
00018
00019 class DLLoader : public ILoader {
00020     private:
00021         void *_handler = nullptr;
00022
00023     public:
00024         ~DLLoader() = default;
00025
00026         void *getHandler() const override {
00027             return _handler;
00028         };
00029         void *Open(const char *path, int flag = RTLD_LAZY) override {
00030             _handler = dlopen(path, flag);
00031             return _handler;
00032         };
00033         void *Symbol(const char *symbolName) override {
00034             void *symbol = dlsym(_handler, symbolName);
00035             const char *error = dlerror();
00036             if (error) {
00037                 std::cerr « "dlerror: " « error « std::endl;
00038                 return nullptr;
00039             }
00040             return symbol;
00041         };
00042         T getSymbol(const char *symbolName) {
00043             return reinterpret_cast<T>(dlsym(_handler, symbolName));
00044         };
00045         int Close() override{
00046             if (_handler == nullptr)
00047                 return -1;
00048             return dlclose(_handler);
00049         };
00050         const char *Error() override {
00051             return dlerror();
00052         };
00053 };
00054
00055 #endif /* !DLLOADER_HPP_ */
```

5.9 ILoader.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** ILoader
00006 */
00007
00008 #ifndef ILoader_HPP_
00009 #define ILoader_HPP_
00010
00011
00012 class ILoader {
00013     public:
00014         ~ILoader() = default;
00015
00016         virtual void *Open(const char *path, int flag) = 0;
00017         virtual void *Symbol(const char *symbolName) = 0;
00018         virtual int Close() = 0;
00019         virtual const char *Error() = 0;
00020         virtual void *getHandler() const = 0;
00021
00022     protected:
00023     private:
00024 };
00025
00026 #endif /* !ILoader_HPP_ */
```

5.10 LoaderType.hpp

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

5.11 Exceptions.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Exceptions
00006 */
00007
00008 #ifndef EXCEPTIONS_HPP_
00009 #define EXCEPTIONS_HPP_
00010
00011 #include <exception>
00012 #include <string>
00013 #include "../Utils/Constants.hpp"
00014
00015 namespace Exceptions {
00016
00017     // CLI Exceptions
00018     class CLIParsingException : public std::exception {
00019         public:
00020             explicit CLIParsingException(const std::string &message)
00021                 : _message(std::string(colors::T_RED) +
00022                           "CLI Parsing Error: " + message +
00023                           colors::RESET) {}
00024
00025             const char *what() const noexcept override {
00026                 return _message.c_str();
00027             }
00028
00029         private:
```

```
00030             std::string _message;
00031     };
00032
00033     class CLIPortException : public CLIParsingException {
00034         public:
00035             explicit CLIPortException(const std::string &message)
00036                 : CLIParsingException(std::string(colors::T_CYAN) +
00037                                       "Port Error: " + message +
00038                                       colors::RESET) {}
00039     };
00040
00041     class CLIHostException : public CLIParsingException {
00042         public:
00043             explicit CLIHostException(const std::string &message)
00044                 : CLIParsingException(std::string(colors::T_CYAN) +
00045                                       "Hostname Error: " + message +
00046                                       colors::RESET) {}
00047     };
00048
00049     class CLIMissingArgumentException : public CLIParsingException {
00050         public:
00051             explicit CLIMissingArgumentException(const std::string &message)
00052                 : CLIParsingException(std::string(colors::T_CYAN) +
00053                                       "Missing Argument: " + message +
00054                                       colors::RESET) {}
00055     };
00056
00057     class CLIInvalidArgumentException : public CLIParsingException {
00058         public:
00059             explicit CLIInvalidArgumentException(const std::string &message)
00060                 : CLIParsingException(std::string(colors::T_CYAN) +
00061                                       "Invalid Argument: " + message +
00062                                       colors::RESET) {}
00063     };
00064
00065     class NetworkException : public std::exception {
00066         public:
00067             explicit NetworkException(const std::string &message)
00068                 : _message(std::string(colors::T_RED) +
00069                           "Network Error: " + message +
00070                           colors::RESET) {}
00071
00072             const char *what() const noexcept override {
00073                 return _message.c_str();
00074             }
00075
00076         private:
00077             std::string _message;
00078     };
00079
00080     class ConnectionFailedException : public NetworkException {
00081         public:
00082             explicit ConnectionFailedException(const std::string &message)
00083                 : NetworkException(std::string(colors::T_CYAN) +
00084                                    "Connection Failed: " + message +
00085                                    colors::RESET) {}
00086     };
00087
00088     class SocketCreationException : public NetworkException {
00089         public:
00090             explicit SocketCreationException(const std::string &message)
00091                 : NetworkException(std::string(colors::T_CYAN) +
00092                                    "Socket Creation Failed: " + message +
00093                                    colors::RESET) {}
00094     };
00095
00096     class ConnectionTimeoutException : public NetworkException {
00097         public:
00098             explicit ConnectionTimeoutException(const std::string &message)
00099                 : NetworkException(std::string(colors::T_CYAN) +
00100                                    "Connection Timeout: " + message +
00101                                    colors::RESET) {}
00102     };
00103
00104     class SendException : public NetworkException {
00105         public:
00106             explicit SendException(const std::string &message)
00107                 : NetworkException(std::string(colors::T_CYAN) +
00108                                    "Send Error: " + message +
00109                                    colors::RESET) {}
00110     };
00111
00112     class ReceiveException : public NetworkException {
00113         public:
00114             explicit ReceiveException(const std::string &message)
00115                 : NetworkException(std::string(colors::T_CYAN) +
00116                                    "Receive Error: " + message +
```

```
00117                                   colors::RESET) {}
00118     };
00119
00120     class ModuleError : public std::exception {
00121         private:
00122             std::string _message = "";
00123         public:
00124             explicit ModuleError(const std::string &msg) : _message(msg) {};
00125             const char *what() const noexcept override {
00126                 return this->_message.c_str();
00127             }
00128     };
00129 }
00130
00131 #endif /* !EXCEPTIONS_HPP_ */
```

5.12 GameInfos.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GameInfos
00006 */
00007
00008 #ifndef GAMEINFOS_HPP_
00009 #define GAMEINFOS_HPP_
00010
00011 #include <utility>
00012 #include <vector>
00013 #include <memory>
00014 #include <mutex>
00015 #include <string>
00016 #include <chrono>
00017
00018 #include "../Utils/Constants.hpp"
00019 #include "../Communication/ICommunication.hpp"
00020 #include "../Observer/Subject.hpp"
00021
00022 class GameInfos : public Subject {
00023     public:
00024         explicit GameInfos(std::shared_ptr<ICommunication> communication);
00025         ~GameInfos();
00026
00027         void setMapSize(int width, int height);
00028         std::pair<int, int> getMapSize() const;
00029
00030         void setTimeUnit(int timeUnit, bool sendToServer = false);
00031         int getTimeUnit() const;
00032
00033         void updateTile(const zappy::structs::Tile tile);
00034         const std::vector<zappy::structs::Tile> getTiles() const;
00035         const zappy::structs::Tile getTile(int x, int y) const;
00036
00037         void updateTeamName(const std::string &teamName);
00038         const std::vector<std::string> getTeamNames() const;
00039
00040         void addPlayer(const zappy::structs::Player player);
00041         void updatePlayerPosition(int playerNumber, int x, int y);
00042         void updatePlayerOrientation(int playerNumber, int orientation);
00043         void updatePlayerLevel(int playerNumber, int level);
00044         void updatePlayerInventory(int playerNumber,
00045             const zappy::structs::Inventory inventory);
00046         void updatePlayerExpulsion(int playerNumber);
00047         void updatePlayerDeath(int playerNumber);
00048         void updatePlayerResourceAction(int playerNumber, int resourceId, bool isCollecting);
00049         void updatePlayerFork(int playerNumber);
00050         const std::vector<zappy::structs::Player> getPlayers() const;
00051         const zappy::structs::Player getPlayer(int playerNumber) const;
00052
00053         void addPlayerBroadcast(int playerNumber, const std::string &message);
00054         const std::vector<std::pair<int, std::string>> getPlayersBroadcasting();
00055
00056         void addIncantation(const zappy::structs::Incantation incantation);
00057         void removeIncantation(int x, int y, int result);
00058
00059         void addEgg(const zappy::structs::Egg egg);
00060         void updateEggHatched(int eggNumber);
00061         void updateEggDeath(int eggNumber);
00062         const std::vector<zappy::structs::Egg> getEggs() const;
00063
00064         void setGameOver(const std::string &winningTeam);
00065         std::pair<bool, std::string> isGameOver() const;
```

```
00066
00067     private:
00068         int _mapWidth;
00069         int _mapHeight;
00070         int _timeUnit;
00071
00072         std::vector<zappy::structs::Tile> _tiles;
00073         std::vector<std::string> _teamNames;
00074         std::vector<zappy::structs::Player> _players;
00075         std::vector<std::pair<int, bool» _playersExpulsing;
00076         std::vector<std::tuple<int, std::string, std::chrono::steady_clock::time_point»
00077             _playersBroadcasting;
00078         std::vector<zappy::structs::Incantation> _incantations;
00079         std::vector<zappy::structs::Egg> _eggs;
00080
00081         bool _gameOver;
00082         std::string _winningTeam;
00083
00084         mutable std::mutex _dataMutex;
00085
00086         std::shared_ptr<ICommunication> _communication;
00087
00088         void notifyStateChange() {
00089             notifyObservers();
00090         }
00091 };
00092
00093 #endif /* !GAMEINFOS_HPP_ */
```

5.13 CameraManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** CameraManager
00006 */
00007
00008 #ifndef CAMERA_MANAGER_HPP_
00009 #define CAMERA_MANAGER_HPP_
00010
00011 #include <memory>
00012 #include "../../Utils/Constants.hpp"
00013 #include "../../Game/GameInfos.hpp"
00014 #include "../Map.hpp"
00015
00016 class CameraManager {
00017     public:
00018         explicit CameraManager(std::shared_ptr<IDisplay> display);
00019         ~CameraManager();
00020
00021         void updateCamera(zappy::gui::CameraMode mode);
00022         void updateCameraFreeMode();
00023         void updateCameraTargetMode();
00024         void updateCameraPlayerMode();
00025
00026         void setMapCenter(const Vector3f &center);
00027         void setMapSize(int width, int height);
00028
00029         void setTargetDistance(float distance);
00030         void initTargetPositionFromCurrentCamera();
00031
00032         void setPlayerId(int playerId);
00033         int getPlayerId() const;
00034         void setGameInfos(std::shared_ptr<GameInfos> gameInfos);
00035         void setMapInstance(std::shared_ptr<Map> map);
00036
00037     private:
00038         std::shared_ptr<IDisplay> _display;
00039         std::shared_ptr<GameInfos> _gameInfos;
00040         std::shared_ptr<Map> _map;
00041         Vector3f _mapCenter;
00042         int _mapWidth;
00043         int _mapHeight;
00044
00045         float _targetDistance;
00046         float _targetAngleXZ;
00047         float _targetAngleY;
00048         bool _isDragging;
00049         int _playerId;
00050
00051         float _playerAngleXZ;
00052         bool _isPlayerViewDragging;
```

```
00053
00054         void handlePlayerCameraMouseInput();
00055         Vector3f calculatePlayerPosition(const zappy::structs::Player& player);
00056         Vector3f calculateCameraPosition(const Vector3f& playerPos, float angleXZ);
00057 };
00058
00059 #endif /* !CAMERA_MANAGER_HPP_ */
```

5.14 GUI.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GUI
00006 */
00007
00008 #ifndef GUI_HPP_
00009 #define GUI_HPP_
00010
00011 #include <memory>
00012 #include <string>
00013 #include "../Game/GameInfos.hpp"
00014 #include "Map.hpp"
00015 #include "HUD/HUD.hpp"
00016 #include "../Audio/IAudio.hpp"
00017 #include "../Utils/Constants.hpp"
00018 #include "Camera/CameraManager.hpp"
00019 #include "../IDisplay.hpp"
00020 #include "../DLLoader/DLLoader.hpp"
00021
00022 class GUI {
00023     public:
00024         GUI(std::shared_ptr<GameInfos> gameInfos, const std::string &libPath);
00025         ~GUI();
00026
00027         void run();
00028         void refresh();
00029
00030         int getWindowWidth() const;
00031         int getWindowHeight() const;
00032         void setWindowWidth(int width);
00033         void setWindowHeight(int height);
00034
00035         void switchCameraMode(zappy::gui::CameraMode mode);
00036         void switchCameraModeNext();
00037         void setPlayerToFollow(int playerId);
00038         int getPlayerToFollow() const;
00039         bool selectFirstAvailablePlayer();
00040         void switchToNextPlayer();
00041         void switchToPreviousPlayer();
00042
00043     private:
00044         void updateCamera();
00045         virtual void update();
00046         virtual void draw();
00047         virtual bool isRunning();
00048         bool playerExists(int playerId) const;
00049
00050         void initModels();
00051
00052         std::string _currentLibLoaded;
00053         bool _isRunning;
00054
00055         DLLoader<std::shared_ptr<IDisplay>> _dlLoader;
00056         std::shared_ptr<IDisplay> _display;
00057         std::shared_ptr<GameInfos> _gameInfos;
00058         std::unique_ptr<Map> _map;
00059         std::unique_ptr<HUD> _hud;
00060         std::shared_ptr<IAudio> _audio;
00061         std::unique_ptr<CameraManager> _cameraManager;
00062
00063         int _windowWidth;
00064         int _windowHeight;
00065
00066         zappy::gui::CameraMode _cameraMode;
00067 };
00068
00069 #endif /* !GUI_HPP_ */
```

5.15 Button.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Button
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <functional>
00012 #include <memory>
00013
00014 #include "../UIElement/AUIElement.hpp"
00015 #include "../../../Audio/IAudio.hpp"
00016 #include "../../../IDisplay.hpp"
00017
00018 class Button : public AUIElement {
00019     public:
00020         Button(
00021             std::shared_ptr<IDisplay> display,
00022             std::shared_ptr<IAudio> audio,
00023             float x, float y,
00024             float width, float height,
00025             const std::string& text,
00026             std::function<void()> callback
00027         );
00028
00029         ~Button() override = default;
00030
00031         void draw() override;
00032
00033         void update() override;
00034
00035         void setText(const std::string& text);
00036
00037         std::string getText() const;
00038
00039         void setCallback(std::function<void()> callback);
00040
00041         void setColors(
00042             Color32 normal,
00043             Color32 hover,
00044             Color32 pressed,
00045             Color32 textColor
00046         );
00047
00048         void setSize(float width, float height) override;
00049
00050     private:
00051         std::string _text;
00052         std::function<void()> _callback;
00053
00054         Color32 _normalColor;
00055         Color32 _hoverColor;
00056         Color32 _pressedColor;
00057         Color32 _textColor;
00058
00059         bool _isHovered;
00060         bool _isPressed;
00061
00062         std::shared_ptr<IDisplay> _display;
00063         std::shared_ptr<IAudio> _audio;
00064 };
```

5.16 AContainers.hpp

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

```
00015
00016 struct RelativePosition {
00017     float xPercent;
00018     float yPercent;
00019     float widthPercent;
00020     float heightPercent;
00021 };
00022
00023 class AContainers : public IContainers {
00024     public:
00025         AContainers(std::shared_ptr<IDisplay> display, float x, float y, float width,
00026             float height);
00027
00028         virtual ~AContainers() = default;
00029
00030         void setPosition(float x, float y) override;
00031         void setSize(float width, float height) override;
00032         FloatRect getBounds() const override;
00033         bool contains(float x, float y) const override;
00034         void setVisible(bool visible) override;
00035         bool isVisible() const override;
00036
00037         void setRelativePosition(float xPercent, float yPercent, float widthPercent,
00038             float heightPercent);
00039
00040         RelativePosition getRelativePosition() const;
00041
00042         void updatePositionFromRelative();
00043
00044     protected:
00045         std::shared_ptr<IDisplay> _display;
00046         FloatRect _bounds;
00047         RelativePosition _relativePos;
00048         Color32 _backgroundColor;
00049         bool _visible;
00050         bool _hasBackground;
00051 };
```

5.17 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 "../Slider/Slider.hpp"
00021 #include "../../../Audio/IAudio.hpp"
00022 #include "../../../IDisplay.hpp"
00023
00024 class Containers : public AContainers {
00025     public:
00026         Containers(std::shared_ptr<IDisplay> display, std::shared_ptr<IAudio> audio,
00027             float x, float y, float width, float height,
00028             Color32 backgroundColor = {40, 40, 40, 200});
00029
00030         ~Containers() override = default;
00031
00032         void draw() override;
00033
00034         void update() override;
00035
00036         void setBackgroundColor(Color32 color);
00037
00038         bool addElement(const std::string& id, std::shared_ptr<IUIElement> element);
00039
00040         std::shared_ptr<IUIElement> getElement(const std::string& id) const;
00041
00042         bool removeElement(const std::string& id);
00043
```

```
00044         std::shared_ptr<Button> addButton(
00045             const std::string& id,
00046             float x, float y,
00047             float width, float height,
00048             const std::string& text,
00049             std::function<void()> callback
00050         );
00051 
00052         std::shared_ptr<Button> addButton(
00053             const std::string& id,
00054             float x, float y,
00055             float width, float height,
00056             const std::string& text,
00057             std::function<void()> callback,
00058             Color32 normalColor,
00059             Color32 hoverColor,
00060             Color32 pressedColor,
00061             Color32 textColor
00062         );
00063 
00064         std::shared_ptr<Text> addText(
00065             const std::string& id,
00066             float x, float y,
00067             const std::string& text,
00068             float fontSize = 20.0f,
00069             Color32 color = CBLACK
00070         );
00071 
00072         std::shared_ptr<Slider> addSlider(
00073             const std::string& id,
00074             float x, float y,
00075             float width, float height,
00076             float minValue, float maxValue,
00077             float initialValue,
00078             const std::string& text,
00079             std::function<void(float)> onValueChanged
00080         );
00081 
00082         std::shared_ptr<Slider> addSliderPercent(
00083             const std::string& id,
00084             float xPercent, float yPercent,
00085             float widthPercent, float heightPercent,
00086             float minValue, float maxValue,
00087             float initialValue,
00088             const std::string& text,
00089             std::function<void(float)> onValueChanged
00090         );
00091 
00092         void clearElements();
00093 
00094         void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00095 
00096         std::shared_ptr<Button> addButtonPercent(
00097             const std::string& id,
00098             float xPercent, float yPercent,
00099             float widthPercent, float heightPercent,
00100             const std::string& text,
00101             std::function<void()> callback
00102         );
00103 
00104         std::shared_ptr<Button> addButtonPercent(
00105             const std::string& id,
00106             float xPercent, float yPercent,
00107             float widthPercent, float heightPercent,
00108             const std::string& text,
00109             std::function<void()> callback,
00110             Color32 normalColor,
00111             Color32 hoverColor,
00112             Color32 pressedColor,
00113             Color32 textColor
00114         );
00115 
00116         std::shared_ptr<Text> addTextPercent(
00117             const std::string& id,
00118             float xPercent, float yPercent,
00119             const std::string& text,
00120             float fontSizePercent = 5.0f,
00121             Color32 color = CBLACK
00122         );
00123 
00124     private:
00125         std::shared_ptr<IAudio> _audio;
00126         std::unordered_map<std::string, std::shared_ptr<IUIElement» _elements;
00127 };
```

5.18 IContainers.hpp

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

5.19 Help.hpp

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

5.20 HUD.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** HUD
00006 */
00007
00008 #pragma once
00009
00010 #include <vector>
00011 #include <unordered_map>
00012 #include <memory>
00013 #include <string>
00014 #include <utility>
00015 #include <functional>
00016 #include "Containers/Containers.hpp"
00017 #include "../../Game/GameInfos.hpp"
00018 #include "../../Audio/IAudio.hpp"
00019 #include "../../Utils/Constants.hpp"
00020 #include "Help/Help.hpp"
00021 #include "../../IDisplay.hpp"
00022
00023 class HUD {
00024     public:
00025         HUD(std::shared_ptr<IDisplay> display, std::shared_ptr<GameInfos> gameInfos,
00026             std::shared_ptr<IAudio> audio,
00027             std::function<void()> resetCameraFunc = nullptr);
00028
00029         ~HUD();
00030
00031         void draw();
00032
00033         void update();
00034
00035         std::shared_ptr<Containers> addContainer(
00036             const std::string& id,
00037             float x, float y,
00038             float width, float height,
00039             Color32 backgroundColor = {40, 40, 40, 200}
00040         );
00041
00042         std::shared_ptr<Containers> getContainer(const std::string& id) const;
00043
00044         bool removeContainer(const std::string& id);
00045
00046         void handleResize(int oldWidth, int oldHeight, int newWidth, int newHeight);
00047
00048         void clearAllContainers();
00049
00050         void initDefaultLayout(float sideWidthPercent = 15.0f,
00051             float bottomHeightPercent = 20.0f);
00052
00053         std::shared_ptr<Containers> getSideContainer() const;
00054
00055         std::shared_ptr<Containers> getBottomContainer() const;
00056
00057         std::shared_ptr<Containers> getSquareContainer() const;
00058
00059         std::shared_ptr<Containers> getTpsContainer() const;
00060
00061         void initExitButton();
00062
00063         void initSettingsButton();
00064
00065         void initHelpButton();
00066
00067         void initCameraResetButton();
00068
00069         void initTeamPlayersDisplay(std::shared_ptr<GameInfos> gameInfos);
00070
00071         void updateTeamPlayersDisplay(std::shared_ptr<GameInfos> gameInfos);
00072
00073         void initTpsSlider(std::shared_ptr<GameInfos> gameInfos,
00074             std::shared_ptr<IDisplay> raylib, std::shared_ptr<IAudio> audio);
00075
00076         void updateTpsSlider(std::shared_ptr<GameInfos> gameInfos);
00077
00078         void initPlayerInventoryDisplay(int playerId);
00079
00080         void updatePlayerInventoryDisplay(int playerId, zappy::gui::CameraMode cameraMode);
00081
00082         void clearPlayerInventoryElements();
00083
00084         zappy::structs::Player getPlayerById(int playerId) const;
00085
```

```
00086         void setResetCameraCallback(std::function<void()> resetFunc);
00087
00088     private:
00089         std::shared_ptr<Containers> createSquareContainer(float squareSize,
00090             float sideWidthPercent);
00091
00092         std::shared_ptr<Containers> createSideContainer(
00093             float sideYStart,
00094             float sideWidth,
00095             float sideHeight,
00096             float sideWidthPercent,
00097             float bottomHeightPercent);
00098
00099         std::shared_ptr<Containers> createBottomContainer(
00100             int screenWidth,
00101             int screenHeight,
00102             float bottomHeight,
00103             float bottomHeightPercent);
00104
00105         std::shared_ptr<Containers> createTpsContainer(
00106             int screenWidth,
00107             int screenHeight,
00108             float bottomHeight,
00109             float bottomHeightPercent);
00110
00111         void updateElementPositions(
00112             std::shared_ptr<Containers> container,
00113             const std::unordered_map<std::string, float>& initialYPositions,
00114             float offset);
00115
00116         std::pair<float, float> calculateContentMetrics(
00117             std::shared_ptr<Containers> container,
00118             const std::unordered_map<std::string, float>& initialYPositions);
00119
00120         void clearTeamDisplayElements(std::shared_ptr<Containers> container);
00121
00122         std::vector<int> getTeamPlayerNumbers(const std::string& teamName,
00123             const std::vector<zappy::structs::Player>& players);
00124
00125         std::string createPlayerListText(const std::vector<int>& playerNumbers);
00126
00127         void addPlayerListText(std::shared_ptr<Containers> container,
00128                                const std::string& teamId,
00129                                float yPos, const std::vector<int>& playerNumbers);
00130
00131         std::unordered_map<std::string, std::shared_ptr<Containers» _containers;
00132         std::shared_ptr<IDisplay> _display;
00133         std::shared_ptr<GameInfos> _gameInfos;
00134         std::shared_ptr<IAudio> _audio;
00135         std::shared_ptr<Help> _help;
00136         std::function<void()> _resetCameraFunc;
00137 };
```

5.21 Slider.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Slider
00006 */
00007
00008 #ifndef SLIDER_HPP_
00009 #define SLIDER_HPP_
00010
00011 #include <string>
00012 #include <functional>
00013 #include <memory>
00014
00015 #include "../../../IDisplay.hpp"
00016 #include "../UIElement/AUIElement.hpp"
00017
00018 class Slider : public AUIElement {
00019     public:
00020         Slider(
00021             std::shared_ptr<IDisplay> raylib,
00022             float x, float y,
00023             float width, float height,
00024             float minValue, float maxValue,
00025             float initialValue,
00026             const std::string& text,
00027             std::function<void(float)> onValueChanged
00028         );
```

```
00029
00030         ~Slider() override = default;
00031
00032         void draw() override;
00033         void update() override;
00034         bool isDragging() const;
00035
00036         void setValue(float value);
00037         float getValue() const;
00038         void setMinValue(float minValue);
00039         void setMaxValue(float maxValue);
00040         float getMinValue() const;
00041         float getMaxValue() const;
00042         void setText(const std::string& text);
00043         std::string getText() const;
00044
00045         void setSize(float width, float height) override;
00046
00047     private:
00048         float _value;
00049         float _minValue;
00050         float _maxValue;
00051         std::string _text;
00052         std::function<void(float)> _onValueChanged;
00053
00054         bool _isDragging;
00055         float _sliderTrackWidth;
00056         float _sliderHandleRadius;
00057
00058         Color32 _trackColor;
00059         Color32 _fillColor;
00060         Color32 _handleColor;
00061         Color32 _textColor;
00062
00063         float _lastChangeTime;
00064         bool _hasUnnotifiedChange;
00065         float _lastNotifiedValue;
00066
00067         void updateValueFromMousePosition(float mouseX);
00068         float getHandlePosition() const;
00069         bool isMouseOverHandle(float mouseX, float mouseY) const;
00070 };
00071
00072 #endif /* !SLIDER_HPP_ */
```

5.22 Text.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Text
00006 */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include <string>
00012
00013 #include "../UIElement/AUIElement.hpp"
00014 #include "../../../IDisplay.hpp"
00015
00016 class Text : public AUIElement {
00017     public:
00018         Text(
00019             std::shared_ptr<IDisplay> raylib,
00020             float x, float y,
00021             const std::string& text,
00022             float fontSize = 20.0f,
00023             Color32 color = CBLACK
00024         );
00025
00026         ~Text() override = default;
00027
00028         void draw() override;
00029
00030         void update() override;
00031
00032         void setText(const std::string& text);
00033
00034         std::string getText() const;
00035
00036         void setFontSize(float fontSize);
```

```
00037
00038         float getFontSize() const;
00039
00040         void setColor(Color32 color);
00041
00042         Color32 getColor() const;
00043
00044         void setSize(float width, float height) override;
00045
00046     private:
00047         std::string _text;
00048         float _fontSize;
00049         Color32 _color;
00050         std::shared_ptr<IDisplay> _display;
00051 };
```

5.23 AUIElement.hpp

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

5.24 IUIElement.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** IUIElement
00006 */
00007
00008 #pragma once
00009
00010 #include "../../../IDisplay.hpp"
00011
00012 class IUIElement {
```

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

5.25 Map.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Map
00006 */
00007
00008 #ifndef MAP_HPP_
00009 #define MAP_HPP_
00010
00011 #include <memory>
00012 #include <unordered_map>
00013 #include <vector>
00014 #include <string>
00015 #include <chrono>
00016 #include "../Game/GameInfos.hpp"
00017 #include "../IDisplay.hpp"
00018
00019 enum class DisplayPriority {
00020     TILE = 0,
00021     EGG = 1,
00022     PLAYER = 2,
00023     FOOD = 3,
00024     ROCK = 4,
00025 };
00026
00027 class Map {
00028     public:
00029         Map(std::shared_ptr<GameInfos> gameInfos, std::shared_ptr<IDisplay> display);
00030         ~Map();
00031
00032         void draw();
00033         void drawBroadcastingPlayers();
00034         void drawTile(int x, int y, const zappy::structs::Tile &tile);
00035         void drawRock(int x, int y, const zappy::structs::Tile &tile);
00036         void drawFood(int x, int y, const zappy::structs::Tile &tile);
00037         void drawPlayers(int x, int y);
00038         void drawEggs(int x, int y);
00039         Color32 getTeamColor(const std::string &teamName);
00040
00041         float getOffset(DisplayPriority priority, int x, int y, size_t stackIndex = 0);
00042
00043     private:
00044         std::shared_ptr<GameInfos> _gameInfos;
00045         std::shared_ptr<IDisplay> _display;
00046         std::unordered_map<std::string, Color32> _teamColors;
00047         std::vector<Color32> _colors;
00048         int _colorIndex = 0;
00049
00050         std::unordered_map<int, std::chrono::steady_clock::time_point> _broadcastStartTimes;
00051
00052         static constexpr float BASE_HEIGHT_TILE = 0.0f;
00053         static constexpr float BASE_HEIGHT_FOOD = 0.2f;
00054         static constexpr float BASE_HEIGHT_ROCK = 0.2f;
00055         static constexpr float BASE_HEIGHT_EGG = 0.2f;
00056         static constexpr float BASE_HEIGHT_PLAYER = 0.2f;
00057         static constexpr float FOOD_HEIGHT = 0.3f;
00058         static constexpr float ROCK_HEIGHT = 0.3f;
00059         static constexpr float EGG_HEIGHT = 0.3f;
00060         static constexpr float PLAYER_HEIGHT = 1.1f;
00061
```

```
00062         void drawOrientationArrow(const Vector3f &position, int orientation,
00063             float playerHeight);
00064         void drawTorus(const Vector3f &position, float radius, float thickness,
00065             int radialSegments, Color32 color);
00066 };
00067
00068 #endif /* !MAP_HPP_ */
```

5.26 IDisplay.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** IDisplay
00006 */
00007
00008 #ifndef IDISPLAY_HPP_
00009 #define IDISPLAY_HPP_
00010 #include <utility>
00011 #include <string>
00012
00013 enum Key {
00014     TAB,
00015     UP,
00016     DOWN,
00017     RIGHT,
00018     LEFT,
00019     GM_PD_LEFT_SHOULDER,
00020     GM_PD_RIGHT_SHOULDER,
00021     GM_PD_LEFT_TRIGGER,
00022     GM_PD_RIGHT_TRIGGER,
00023     GM_PD_UP,
00024     GM_PD_DOWN,
00025     GM_PD_AXIS_RIGHT_X,
00026     GM_PD_AXIS_RIGHT_Y,
00027     MOUSE_LEFT,
00028     MOUSE_RIGHT,
00029 };
00030
00031 typedef struct Vector3f {
00032     float x;
00033     float y;
00034     float z;
00035 } Vector3f;
00036
00037 typedef struct Vector2f {
00038     float x;
00039     float y;
00040 } Vector2f;
00041
00042 typedef struct Vector2i {
00043     int x;
00044     int y;
00045 } Vector2i;
00046
00047 typedef struct Color32 {
00048     unsigned char r;
00049     unsigned char g;
00050     unsigned char b;
00051     unsigned char a;
00052 } Color32;
00053
00054 typedef struct FloatRect {
00055     float x;
00056     float y;
00057     float width;
00058     float height;
00059 } FloatRect;
00060
00061 typedef struct IntRect {
00062     int x;
00063     int y;
00064     int width;
00065     int height;
00066 } IntRect;
00067
00068 #define COLOR(r, g, b) Color32{ r, g, b, 255 }
00069 #define CLIGHTGRAY COLOR(200, 200, 200)
00070 #define CBLACK COLOR(0, 0, 0)
00071 #define CRED COLOR(230, 41, 55)
00072 #define CBROWN COLOR(127, 106, 79)
00073 #define CBLUE COLOR(0, 121, 241)
```

```
00074 #define CWHITE COLOR(255, 255, 255)
00075
00076 #define CRAYWHITE COLOR(245, 245, 245)
00077 #define CPINK COLOR(255, 109, 194)
00078 #define CGREEN COLOR(0, 228, 48)
00079 #define CMAROON COLOR(190, 33, 55)
00080 #define CPURPLE COLOR(200, 122, 255)
00081 #define CORANGE COLOR(255, 161, 0)
00082 #define CYELLOW COLOR(253, 249, 0)
00083
00084 class IDisplay {
00085     public:
00086         virtual Vector2i getMonitorSize() = 0;
00087         virtual Vector2i getScreenSize() = 0;
00088
00089         virtual void initWindow(int width, int height, std::string) = 0;
00090         virtual void initCamera() = 0;
00091
00092         virtual bool isWindowReady() = 0;
00093         virtual void setTargetFPS(unsigned int FPS) = 0;
00094
00095         virtual bool isOpen() = 0;
00096         virtual void closeWindow() = 0;
00097
00098         virtual int getKeyId(enum Key) = 0;
00099
00100         virtual bool isKeyReleased(int key) = 0;
00101         virtual bool isKeyPressed(int key) = 0;
00102         virtual bool isKeyDown(int key) = 0;
00103
00104         virtual bool isGamepadAvailable() = 0;
00105
00106         virtual bool isGamepadButtonReleased(int key) = 0;
00107         virtual bool isGamepadButtonPressed(int key) = 0;
00108         virtual bool isGamepadButtonDown(int key) = 0;
00109
00110         virtual bool isMouseButtonDown(int key) = 0;
00111         virtual bool isMouseButtonReleased(int key) = 0;
00112         virtual bool isMouseButtonPressed(int key) = 0;
00113
00114         virtual Vector2f getMousePosition() = 0;
00115         virtual void setMousePosition(Vector2f) = 0;
00116
00117         virtual float getMouseWheelMove() = 0;
00118
00119         virtual float getGamepadAxisMovement(int key) = 0;
00120
00121         virtual void setCameraPosition(Vector3f) = 0;
00122
00123         virtual void setCameraTarget(Vector3f) = 0;
00124
00125         virtual Vector2f getMouseDelta() = 0;
00126
00127         virtual float vector3DDistanceFromCamera(Vector3f target) = 0;
00128         virtual Vector3f vector3SubtractFromCamera(Vector3f target) = 0;
00129
00130         virtual Vector3f vector3Normalize(Vector3f) = 0;
00131
00132
00133         virtual void enableCursor() = 0;
00134         virtual void disableCursor() = 0;
00135
00136         virtual float getFrameTime() = 0;
00137
00138         virtual void updateCameraFreeMode() = 0;
00139
00140         virtual float measureText(const std::string& text, float fontSize) const = 0;
00141
00142         virtual bool checkCollisionPointRec(Vector2f point, FloatRect rec) = 0;
00143
00144         virtual void beginDrawing() = 0;
00145         virtual void endDrawing() = 0;
00146         virtual void clearBackground(Color32) = 0;
00147
00148         virtual void begin3DMode() = 0;
00149         virtual void end3DMode() = 0;
00150
00151         virtual void endScissorMode() = 0;
00152         virtual void beginScissorMode(IntRect) = 0;
00153
00154         virtual bool loadModel(const std::string& id, const std::string& filepath,
00155             Vector3f center = {0.0f, 0.0f, 0.0f}) = 0;
00156
00157         virtual void drawCube(Vector3f position, float width, float height, float length,
00158             Color32 color) = 0;
00159         virtual void drawCubeWires(Vector3f position, float width, float height, float length,
00160             Color32 color) = 0;
```

```
00161
00162         virtual void drawSphere(Vector3f position, float radius, Color32 color) = 0;
00163         virtual void drawSphereWires(Vector3f position, float radius, int rings, int slices,
00164             Color32 color) = 0;
00165
00166         virtual void drawCylinder(Vector3f position, float radiusTop, float radiusBottom,
00167             float height, int slices, Color32 color) = 0;
00168         virtual void drawCylinderWires(Vector3f position, float radiusTop, float radiusBottom,
00169             float height, int slices, Color32 color) = 0;
00170         virtual void drawCylinderEx(Vector3f startPos, Vector3f endPos, float startRadius,
00171             float endRadius, int sides, Color32 color) = 0;
00172
00173         virtual void drawPlane(Vector3f position, Vector2f size, Color32 color) = 0;
00174
00175         virtual void drawLine3D(Vector3f startPos, Vector3f endPos, Color32 color) = 0;
00176
00177         virtual void drawModelEx(const std::string& id, Vector3f position,
00178             Vector3f rotationAxis, float rotationAngle, Vector3f scale,
00179             Color32 tint = CWHITE) = 0;
00180
00181         virtual void drawCircle(float centerX, float centerY, float radius,
00182             Color32 color) = 0;
00183         virtual void drawCircleLines(float centerX, float centerY, float radius,
00184             Color32 color) = 0;
00185
00186         virtual void drawText(const std::string& text, float x, float y, float fontSize,
00187             Color32 color) = 0;
00188
00189         virtual void drawRectangleRec(FloatRect rec, Color32 color) = 0;
00190         ~IDisplay() = default;
00191 };
00192
00193 #endif /* !IDISPLAY_HPP_ */
```

5.27 GuiObserver.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GuiObserver
00006 */
00007
00008 #ifndef GUIOBSERVER_HPP_
00009 #define GUIOBSERVER_HPP_
00010
00011 #include <memory>
00012
00013 #include "IObserver.hpp"
00014
00015 class GUI;
00016
00017 class GuiObserver : public IObserver {
00018     public:
00019         GuiObserver(std::shared_ptr<GUI> gui);
00020         virtual ~GuiObserver() = default;
00021
00022         void update() override;
00023
00024     private:
00025         std::weak_ptr<GUI> _gui;
00026 };
00027
00028 #endif /* !GUIOBSERVER_HPP_ */
```

5.28 IObserver.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** IObserver
00006 */
00007
00008 #ifndef IOBSERVER_HPP_
00009 #define IOBSERVER_HPP_
00010
00011 class IObserver {
00012     public:
```

```
00013         virtual ~IObserver() = default;
00014         virtual void update() = 0;
00015 };
00016
00017 #endif /* !IOBSERVER_HPP_ */
```

5.29 ISubject.hpp

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

5.30 Subject.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** Subject
00006 */
00007
00008 #include <algorithm>
00009 #include <memory>
00010 #include <vector>
00011
00012 #include "ISubject.hpp"
00013
00014 #ifndef SUBJECT_HPP_
00015 #define SUBJECT_HPP_
00016
00017 class Subject : public ISubject {
00018     public:
00019         virtual ~Subject() = default;
00020
00021         void addObserver(std::shared_ptr<IObserver> observer) override {
00022             _observers.push_back(observer);
00023         }
00024
00025         void removeObserver(std::shared_ptr<IObserver> observer) override {
00026             _observers.erase(
00027                 std::remove_if(_observers.begin(), _observers.end(),
00028                     [&observer](const std::weak_ptr<IObserver>& weak_obs) {
00029                         return weak_obs.expired() || weak_obs.lock() == observer;
00030                     }),
00031                 _observers.end());
00032         }
00033
00034         void notifyObservers() override {
00035             _observers.erase(
00036                 std::remove_if(_observers.begin(), _observers.end(),
00037                     [](const std::weak_ptr<IObserver>& weak_obs) {
00038                         if (auto obs = weak_obs.lock()) {
00039                             obs->update();
00040                             return false;
00041                         }
00042                         return true;
```

```
00043                     }),
00044                 _observers.end());
00045         }
00046
00047     private:
00048         std::vector<std::weak_ptr<IObserver>> _observers;
00049 };
00050
00051 #endif /* !SUBJECT_HPP_ */
```

5.31 Raylib.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** zappy
00004 ** File description:
00005 ** Raylib
00006 */
00007
00008 #ifndef RAYLIB_HPP_
00009 #define RAYLIB_HPP_
00010 #include <memory>
00011 #include <string>
00012 #include "../IDisplay.hpp"
00013 #include "RaylibEnc/RayLibEnc.hpp"
00014
00015 class Raylib : public IDisplay {
00016     private:
00017         std::unique_ptr<RayLibEnc> _raylib;
00018
00019     public:
00020         virtual Vector2i getMonitorSize();
00021         virtual Vector2i getScreenSize();
00022
00023         virtual void initWindow(int width, int height, std::string);
00024         virtual void initCamera();
00025
00026         virtual bool isWindowReady();
00027         virtual void setTargetFPS(unsigned int FPS);
00028
00029         virtual bool isOpen();
00030         virtual void closeWindow();
00031
00032         virtual int getKeyId(enum Key);
00033
00034         virtual bool isKeyReleased(int key);
00035         virtual bool isKeyPressed(int key);
00036         virtual bool isKeyDown(int key);
00037
00038         virtual bool isGamepadAvailable();
00039
00040         virtual bool isGamepadButtonReleased(int key);
00041         virtual bool isGamepadButtonPressed(int key);
00042         virtual bool isGamepadButtonDown(int key);
00043
00044
00045         virtual bool isMouseButtonDown(int key);
00046         virtual bool isMouseButtonReleased(int key);
00047         virtual bool isMouseButtonPressed(int key);
00048
00049         virtual Vector2f getMousePosition();
00050         virtual void setMousePosition(Vector2f);
00051
00052         virtual float getMouseWheelMove();
00053
00054         virtual float getGamepadAxisMovement(int key);
00055
00056         virtual void setCameraPosition(Vector3f);
00057
00058         virtual void setCameraTarget(Vector3f);
00059
00060         virtual Vector2f getMouseDelta();
00061
00062         virtual float vector3DDistanceFromCamera(Vector3f target);
00063         virtual Vector3f vector3SubtractFromCamera(Vector3f target);
00064
00065         virtual Vector3f vector3Normalize(Vector3f);
00066
00067         virtual void enableCursor();
00068         virtual void disableCursor();
00069
00070         virtual float getFrameTime();
00071
```

```
00072         virtual void updateCameraFreeMode();
00073
00074         virtual float measureText(const std::string& text, float fontSize) const;
00075
00076         virtual bool checkCollisionPointRec(Vector2f point, FloatRect rec);
00077
00078         virtual void beginScissorMode(IntRect);
00079         virtual void endScissorMode();
00080
00081         virtual void beginDrawing();
00082         virtual void endDrawing();
00083
00084         virtual void clearBackground(Color32);
00085
00086         virtual void begin3DMode();
00087         virtual void end3DMode();
00088
00089         virtual bool loadModel(const std::string& id, const std::string& filepath,
00090             Vector3f center = {0.0f, 0.0f, 0.0f});
00091
00092         virtual void drawCube(Vector3f position, float width, float height, float length,
00093             Color32 color);
00094         virtual void drawCubeWires(Vector3f position, float width, float height, float length,
00095             Color32 color);
00096
00097         virtual void drawSphere(Vector3f position, float radius, Color32 color);
00098         virtual void drawSphereWires(Vector3f position, float radius, int rings, int slices,
00099             Color32 color);
00100
00101         virtual void drawCylinder(Vector3f position, float radiusTop, float radiusBottom,
00102             float height, int slices, Color32 color);
00103         virtual void drawCylinderWires(Vector3f position, float radiusTop, float radiusBottom,
00104             float height, int slices, Color32 color);
00105         virtual void drawCylinderEx(Vector3f startPos, Vector3f endPos, float startRadius,
00106             float endRadius, int sides, Color32 color);
00107
00108         virtual void drawPlane(Vector3f position, Vector2f size, Color32 color);
00109
00110         virtual void drawLine3D(Vector3f startPos, Vector3f endPos, Color32 color);
00111
00112         virtual void drawModelEx(const std::string& id, Vector3f position,
00113             Vector3f rotationAxis, float rotationAngle, Vector3f scale,
00114             Color32 tint = CWHITE);
00115
00116         virtual void drawText(const std::string& text, float x, float y, float fontSize,
00117             Color32 color);
00118
00119         virtual void drawCircle(float centerX, float centerY, float radius,
00120             Color32 color);
00121         virtual void drawCircleLines(float centerX, float centerY,
00122             float radius, Color32 color);
00123
00124         virtual void drawRectangleRec(FloatRect rec, Color32 color);
00125
00126         Raylib();
00127         ~Raylib() = default;
00128 };
00129
00130 #endif /* !RAYLIB_HPP_ */
```

5.32 RayLibEnc.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** RayLibEnc
00006 */
00007
00008 #ifndef RAYLIBENC_HPP_
00009 #define RAYLIBENC_HPP_
00010
00011 #include <string>
00012 #include <map>
00013 #include <memory>
00014 #include "raylib.h"
00015
00016 class RayLibEnc {
00017     public:
00018         RayLibEnc();
00019         ~RayLibEnc();
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         // Gamepad methods
00060         bool isGamepadAvailable(int gamepad) const;
00061         bool isGamepadButtonPressed(int gamepad, int button) const;
00062         bool isGamepadButtonDown(int gamepad, int button) const;
00063         bool isGamepadButtonReleased(int gamepad, int button) const;
00064         float getGamepadAxisMovement(int gamepad, int axis) const;
00065 
00066         // Scissor mode methods for clipping
00067         void beginScissorMode(int x, int y, int width, int height);
00068         void endScissorMode();
00069 
00070         // 3D Environment methods
00071         void begin3DMode();
00072         void end3DMode();
00073         float vector3Distance(Vector3 v1, Vector3 v2) const;
00074         Vector3 vector3Normalize(Vector3 v) const;
00075         Vector3 vector3Subtract(Vector3 v1, Vector3 v2) const;
00076         Vector3 vector3Add(Vector3 v1, Vector3 v2) const;
00077 
00078         // Camera methods
00079         void initCamera();
00080         void setCameraPosition(Vector3 position);
00081         void setCameraTarget(Vector3 target);
00082         void setCameraUp(Vector3 up);
00083         void setCameraFovy(float fovy);
00084         void setCameraProjection(int projection);
00085         void updateCamera(int mode = CAMERA_FREE);
00086         void updateCameraFreeMode();
00087         Camera3D getCamera() const;
00088 
00089         // 3D Drawing methods
00090         void drawGrid(int slices, float spacing);
00091         void drawCube(Vector3 position, float width, float height, float length, Color color);
00092         void drawCubeWires(Vector3 position, float width, float height, float length,
00093             Color color);
00094         void drawSphere(Vector3 position, float radius, Color color);
00095         void drawSphereWires(Vector3 position, float radius, int rings, int slices,
00096             Color color);
00097         void drawCylinder(Vector3 position, float radiusTop, float radiusBottom,
00098             float height, int slices, Color color);
00099         void drawCylinderWires(Vector3 position, float radiusTop, float radiusBottom,
00100             float height, int slices, Color color);
00101         void drawCylinderEx(Vector3 startPos, Vector3 endPos, float startRadius,
00102             float endRadius, int sides, Color color);
00103         void drawPlane(Vector3 position, Vector2 size, Color color);
00104         void drawLine3D(Vector3 startPos, Vector3 endPos, Color color);
00105 
00106         // 3D Model methods
00107         bool loadModel(const std::string& id, const std::string& filepath,
00108             Vector3 center = {0.0f, 0.0f, 0.0f});
```

```
00109         void drawModel(const std::string& id, Vector3 position, float scale,
00110             Color tint = WHITE);
00111         void drawModelEx(const std::string& id, Vector3 position, Vector3 rotationAxis,
00112                          float rotationAngle, Vector3 scale, Color tint = WHITE);
00113         void drawModelWires(const std::string& id, Vector3 position, float scale,
00114             Color tint = WHITE);
00115         void drawModelWiresEx(const std::string& id, Vector3 position, Vector3 rotationAxis,
00116                               float rotationAngle, Vector3 scale, Color tint = WHITE);
00117         void unloadModel(const std::string& id);
00118         void unloadAllModels();
00119         bool modelExists(const std::string& id) const;
00120
00121         // 2D Drawing methods
00122         void drawRectangleRec(Rectangle rec, Color color);
00123         void drawText(const std::string& text, float x, float y, float fontSize, Color color);
00124         void drawCircle(float centerX, float centerY, float radius, Color color);
00125         void drawCircleLines(float centerX, float centerY, float radius, Color color);
00126         float measureText(const std::string& text, float fontSize) const;
00127
00128     private:
00129         bool _isInitialized;
00130         Camera3D _camera;
00131         Vector2 _previousMousePosition;
00132         bool _isCursorLocked;
00133
00134         struct ModelData {
00135             Model model;
00136             unsigned int animationCount;
00137             Vector3 center;
00138         };
00139
00140         std::map<std::string, ModelData> _models;
00141         std::map<std::string, Sound> _sounds;
00142         std::map<std::string, Music> _musics;
00143 };
00144
00145 #endif /* !RAYLIBEnc_HPP_ */
```

5.33 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 #include "HelpText.hpp"
00014
00015 namespace zappy::constants {
00016
00017     inline const char *USAGE_STRING = "USAGE: ./zappy_gui -p port -h machine\n"
00018                                 "option\t\tdescription\n"
00019                                 "-p port\t\tport number\n"
00020                                 "-h machine\thostname of the server";
00021
00022     inline const int FAILURE_EXIT_CODE = 84;
00023     inline const int SUCCESS_EXIT_CODE = 0;
00024 };
00025
00026 namespace colors {
00027
00028     inline const char *T_BOLD = "\033[1m";
00029     inline const char *T_RED = "\033[1m\033[31m";
00030     inline const char *T_GREEN = "\033[1m\033[32m";
00031     inline const char *T_YELLOW = "\033[1m\033[33m";
00032     inline const char *T_BLUE = "\033[1m\033[34m";
00033     inline const char *T_MAGENTA = "\033[1m\033[35m";
00034     inline const char *T_CYAN = "\033[1m\033[36m";
00035     inline const char *T_WHITE = "\033[1m\033[37m";
00036     inline const char *RESET = "\033[0m";
00037
00038 };
00039
00040 namespace zappy::structs {
00041
00042     struct Config {
00043         int port;
```

```
00044         std::string hostname;
00045     };
00046
00047     struct Tile {
00048         int x;
00049         int y;
00050         int food;
00051         int linemate;
00052         int deraumere;
00053         int sibur;
00054         int mendiane;
00055         int phiras;
00056         int thystame;
00057
00058         Tile(int _x = 0, int _y = 0, int _food = 0, int _linemate = 0,
00059              int _deraumere = 0, int _sibur = 0, int _mendiane = 0,
00060              int _phiras = 0, int _thystame = 0)
00061             : x(_x), y(_y), food(_food), linemate(_linemate),
00062               deraumere(_deraumere), sibur(_sibur),
00063               mendiane(_mendiane), phiras(_phiras), thystame(_thystame) {}
00064     };
00065
00066     struct Inventory {
00067         int food;
00068         int linemate;
00069         int deraumere;
00070         int sibur;
00071         int mendiane;
00072         int phiras;
00073         int thystame;
00074
00075         Inventory(int _food = 0, int _linemate = 0, int _deraumere = 0,
00076                   int _sibur = 0, int _mendiane = 0, int _phiras = 0,
00077                   int _thystame = 0)
00078             : food(_food), linemate(_linemate), deraumere(_deraumere),
00079               sibur(_sibur), mendiane(_mendiane), phiras(_phiras),
00080               thystame(_thystame) {}
00081     };
00082     struct Player {
00083         int number;
00084         int x;
00085         int y;
00086         int orientation;
00087         int level;
00088         std::string teamName;
00089         struct Inventory inventory;
00090
00091         Player(int _number = 0, int _x = 0, int _y = 0, int _orientation = 0,
00092                int _level = 1, const std::string &_teamName = "",
00093                struct Inventory _inventory = Inventory())
00094             : number(_number), x(_x), y(_y), orientation(_orientation),
00095               level(_level), teamName(_teamName), inventory(_inventory) {}
00096     };
00097
00098     struct Incantation {
00099         int x;
00100         int y;
00101         int level;
00102         std::vector<int> players;
00103
00104         Incantation(int _x = 0, int _y = 0, int _level = 1,
00105                     const std::vector<int> &_players = {})
00106             : x(_x), y(_y), level(_level), players(_players) {}
00107     };
00108
00109     struct Egg {
00110         int eggNumber;
00111         int playerNumber;
00112         int x;
00113         int y;
00114         bool hatched;
00115         std::string teamName;
00116
00117         Egg(int _eggNumber = 0, int _playerNumber = 0, int _x = 0, int _y = 0,
00118             bool _hatched = false, const std::string &_teamName = "")
00119             : eggNumber(_eggNumber), playerNumber(_playerNumber), x(_x), y(_y),
00120               hatched(_hatched), teamName(_teamName) {}
00121     };
00122 };
00123
00124 namespace zappy::gui {
00125
00126     inline const std::string WINDOW_TITLE = "Zappy GUI";
00127     inline const int FPS = 120;
00128     inline const float CAMERA_SPEED = 7.5f;
00129     inline const float CAMERA_SENSITIVITY = 0.001f;
00130     inline const float CAMERA_ROTATE_SPEED_KEY = 2.0f;
```

```
00131     inline const float GAMEPAD_STICK_SENSITIVITY = 3.0f;
00132     inline const float GAMEPAD_DEADZONE = 0.2f;
00133     inline const float POSITION_MULTIPLIER = 2.2f;
00134
00135     inline const float PLAYER_SCALE = 0.005f;
00136
00137     enum class CameraMode {
00138         FREE = 0,
00139         TARGETED = 1,
00140         PLAYER = 2,
00141         NB_MODES = 3,
00142     };
00143 }
00144
00145 #endif /* !CONSTANTS_HPP_ */
```

5.34 GamepadConstants.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** GamepadConstants
00006 */
00007
00008 #ifndef GAMEPAD_CONSTANTS_HPP_
00009 #define GAMEPAD_CONSTANTS_HPP_
00010
00011 #ifndef GAMEPAD_AXIS_LEFT_X
00012     #define GAMEPAD_AXIS_LEFT_X       0
00013     #define GAMEPAD_AXIS_LEFT_Y       1
00014     #define GAMEPAD_AXIS_RIGHT_X      2
00015     #define GAMEPAD_AXIS_RIGHT_Y      3
00016     #define GAMEPAD_AXIS_LEFT_TRIGGER 4
00017     #define GAMEPAD_AXIS_RIGHT_TRIGGER 5
00018 #endif
00019
00020 #ifndef GAMEPAD_BUTTON_A
00021     #define GAMEPAD_BUTTON_A          6
00022     #define GAMEPAD_BUTTON_B          5
00023     #define GAMEPAD_BUTTON_X          9
00024     #define GAMEPAD_BUTTON_Y          8
00025     #define GAMEPAD_BUTTON_START      17
00026     #define GAMEPAD_BUTTON_SELECT     16
00027     #define GAMEPAD_BUTTON_UP         1
00028     #define GAMEPAD_BUTTON_RIGHT      2
00029     #define GAMEPAD_BUTTON_DOWN       3
00030     #define GAMEPAD_BUTTON_LEFT       4
00031     #define GAMEPAD_BUTTON_LEFT_SHOULDER  10
00032     #define GAMEPAD_BUTTON_RIGHT_SHOULDER 12
00033     #define GAMEPAD_BUTTON_LEFT_TRIGGER   13
00034     #define GAMEPAD_BUTTON_RIGHT_TRIGGER  15
00035 #endif
00036
00037 #endif /* !GAMEPAD_CONSTANTS_HPP_ */
```

5.35 HelpText.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** HelpText
00006 */
00007
00008 #ifndef HELP_TEXT_HPP_
00009 #define HELP_TEXT_HPP_
00010
00011 namespace zappy::constants {
00012
00013     inline const char *HELP_TITLE =
00014         "HELP";
00015
00016     inline const char *HELP_SECTION_1 =
00017         "Game Overview";
00018
00019     inline const char *HELP_SECTION_1_CONTENT =
00020         "Zappy is a game where AI-controlled players compete to collect resources\n"
00021         "and level up on a dynamically changing map. The GUI allows you to visualize\n"
```

```
00022         "the game state, players, and resources in real-time.";
00023
00024     inline const char *HELP_SECTION_2 =
00025         "Controls";
00026
00027     inline const char *HELP_SECTION_2_CONTENT =
00028         "Camera Movement:\n"
00029         "  - Arrow keys or ZQSD: Move camera\n"
00030         "  - Controller: Use left stick to move camera\n"
00031         "  - Right mouse button + drag: Rotate camera\n\n"
00032         "Interface:\n"
00033         "  - Click on players to see their stats\n"
00034         "  - Use the RESET CAMERA button to return to default view\n"
00035         "  - Use the Settings button to adjust game settings";
00036
00037     inline const char *HELP_SECTION_3 =
00038         "Teams and Players";
00039
00040     inline const char *HELP_SECTION_3_CONTENT =
00041         "The left panel shows all teams and their player IDs.\n"
00042         "Players have different levels based on collected resources.\n"
00043         "The team that first gets a player to level 8 wins the game.";
00044
00045     inline const char *HELP_SECTION_4 =
00046         "Resources";
00047
00048     inline const char *HELP_SECTION_4_CONTENT =
00049         "Resources on the map are represented by different colored objects.\n"
00050         "Players collect these resources to perform rituals and level up.";
00051
00052 }  // namespace zappy::constants
00053
00054 #endif /* !HELP_TEXT_HPP_ */
```

5.36 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_ */
```

5.37 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 /* Definition of the incantation structure */
00084 typedef struct incantation_s {
00085     int levelt_to_reach;
00086     int nb_players;
00087     inventory_t required_inventory;
00088 } incantation_t;
00089
00090
00091 /* Player struct */
00092 typedef struct player_s {
00093     int id;
00094     network_t *network;
00095     int level;
00096     int posX;
00097     int posY;
00098     direction_t direction;
00099     inventory_t *inventory;
00100     char *team;
00101     /* New aditions for the smart pollin */
00102     action_queue_t *pending_actions;
00103     time_t last_action_time;
00104     bool is_busy;
00105     int remaining_cooldown;
00106     char *current_action;
00107
00108     struct player_s *next;
```

```
00109 } player_t;
00110
00111 /* This structure define the request strut */
00112 typedef struct action_request_s {
00113     char *command;
00114     time_t timestamp;
00115     float time_limit;  // in game ticks (7/f, 42/f, etc.)
00116     action_priority_t priority;
00117     player_t *player;
00118     struct action_request_s *next;
00119 } action_request_t;
00120
00121 /* Team Strcut */
00122 typedef struct team_s {
00123     char *name;
00124     int nbPlayers;
00125     int nbPlayerAlive;
00126     player_t *players;
00127     struct team_s *next;
00128 } team_t;
00129
00130
00131 /* Structure that holds the size and array of tiles */
00132 typedef struct map_t {
00133     int width;
00134     int height;
00135     egg_t *currentEggs;  /* List of current eggs */
00136     inventory_t **tiles;  /* Here we call inv for the tile*/
00137 } map_t;
00138
00139
00140 /* Map struct */
00141 typedef struct game_s {
00142     team_t *teams;
00143     map_t *map;
00144 } game_t;
00145
00146 #endif /* !GAME_H_ */
```

5.38 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_ */
```

5.39 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_ */
```

5.40 zappy.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** Zappy
00004 ** File description:
00005 ** Server :: Zappy header
00006 */
00007
00008 #include <stdbool.h>
00009 #include <poll.h>
00010 #include "game.h"
00011 #include "my.h"
00012
00013 #ifndef ZAPPY_H_
00014     #define ZAPPY_H_
00015
00016 /* items handler */
00017 typedef struct {
00018     char *name;
00019     void (*add_func)(inventory_t *);
00020 } item_handler_t;
00021
00022
00023 /* Cli parameter of the server */
00024 typedef struct params_s {
00025     int port;
00026     int x;
00027     int y;
00028     int nb_team;
00029     char **teams;
00030     int nb_client;
00031     int freq;
00032     bool is_debug;
00033 } params_t;
00034
00035 /* Structure to handle the network side of the gui*/
00036 typedef struct graph_net_s {
00037     int fd;
00038     bool mapSent;
00039     struct graph_net_s *next;
00040 } graph_net_t;
00041
00042 /* Server part of the network */
00043 typedef struct server_s {
00044     int sockfd;
00045     struct pollfd pollserver;
00046 } server_t;
00047
00048 typedef struct zappy_s {
00049     server_t *network;
00050     game_t *game;
00051     graph_net_t *graph;
00052     params_t *params;
00053 } zappy_t;
00054
00055 typedef struct command_pf_s {
00056     char const *flag;
00057     bool (*checker)(const char *, const char *, params_t *);
00058 } command_pf_t;
00059
00060 typedef struct {
00061     char *command;
00062     float base_time;
00063     action_priority_t priority;
00064     int (*handler)(player_t *, char *, zappy_t *);
00065 } command_info_t;
00066
00067 /* messages.c */
00068 int helper(void);
00069 void error_message(const char *message);
00070 void valid_message(char const *message);
00071
00072 /* checkers.c */
00073 bool check_port(char const *flag, char const *value, params_t *params);
00074 bool check_width(char const *flag, char const *value, params_t *params);
00075 bool check_height(char const *flag, char const *value, params_t *params);
00076 bool check_client(char const *flag, char const *value, params_t *params);
00077 bool check_freq(char const *flag, char const *value, params_t *params);
00078
00079 /* signal.c */
00080 void setup_signal(void);
00081 int *get_running_state(void);
00082
00083 /* params.c */
00084 params_t *check_args(int argc, char **argv);
00085 void *free_params(params_t *params);
```

```
00086
00087 /* params_cherckers.c */
00088 bool validate_no_extra_args(int argc, char **argv);
00089
00090 /* server.c */
00091 zappy_t *init_server(int argc, char **argv);
00092 void *free_zappy(zappy_t *server);
00093
00094 /* protocol.c */
00095 int start_protocol(zappy_t *server);
00096
00097 /* client.c */
00098 bool process_new_client(const char *team_name, int fd, zappy_t *server);
00099 team_t *add_client_to_team(const char *team_name, int fd, zappy_t *server);
00100 int get_next_free_id(zappy_t *server);
00101 void check_player_status(zappy_t *zappy);
00102
00103 /* init_map.c */
00104 void init_game(zappy_t *server);
00105
00106
00107 /* accept.c */
00108 int accept_client(zappy_t *server);
00109
00110 /* free server  */
00111 void *free_zappy(zappy_t *server);
00112 void *free_params(params_t *params);
00113 void *free_player(player_t *player);
00114 void free_map(map_t *map);
00115
00116 /* Function to send info to the gui */
00117 int send_map_size(zappy_t *server);
00118 int send_entrie_map(zappy_t *server);
00119 int send_map_tile(inventory_t **tiles, zappy_t *server,
00120     int posX, int posY);
00121 int send_team_name(zappy_t *server);
00122 int send_egg(zappy_t *zappy, egg_t *egg);
00123 int send_entire_egg_list(zappy_t *zappy);
00124 int send_time_message(zappy_t *zappy);
00125 int send_egg_death(zappy_t *zappy, egg_t *egg);
00126 int send_egg_connect(zappy_t *zappy, egg_t *currentEgg);
00127 int send_player_connect(zappy_t *zappy, player_t *player);
00128 int send_player_pos(zappy_t *zappy, player_t *player);
00129 int send_player_level(zappy_t *zappy, player_t *player);
00130 int send_player_inventory(zappy_t *zappy, player_t *player);
00131 int send_player_expelled(zappy_t *zappy, player_t *player);
00132 int send_broadcast_to_all(zappy_t *zappy, const char *message);
00133 int send_broadcast_to_player(zappy_t *zappy, player_t *player,
00134     const char *message);
00135 int send_player_laying_egg(zappy_t *zappy, player_t *player);
00136 int send_ressource_droped(zappy_t *zappy, player_t *player,
00137     int ressourceType);
00138 int send_ressource_collected(zappy_t *zappy, player_t *player,
00139     int ressourceType);
00140 int send_player_death(zappy_t *zappy, player_t *player);
00141 int send_updated_time(zappy_t *zappy, int time);
00142 int send_end_game(zappy_t *zappy, const char *teamName);
00143 int send_str_message(zappy_t *zappy, const char *message);
00144 int send_unknown_command(zappy_t *zappy);
00145 int send_command_parameter(zappy_t *zappy);
00146 int send_start_incantation(zappy_t *zappy, player_t *player, int *player_list,
00147     int nb_player);
00148 int send_end_incantation(zappy_t *zappy, player_t *player, char *result);
00149
00150 /* init_egg.c */
00151 void init_egg(zappy_t *zappy);
00152 egg_t *add_egg_node(int id, int *pos, char *team_name, int id_layer);
00153 egg_t *kil_egg_node(egg_t **head, int egg_id);
00154
00155 /* AI messages */
00156 int forward_message(player_t *player, params_t *params);
00157
00158 /* Pollin handler */
00159 void smart_poll_players(zappy_t *zappy);
00160 void execute_action(player_t *player, action_request_t *action,
00161     zappy_t *zappy);
00162 void queue_action(player_t *player, char *command, zappy_t *zappy);
00163 action_queue_t *init_action_queue(void);
00164 void free_action_queue(action_queue_t *queue);
00165 action_request_t *create_action_request(char *command, player_t *player,
00166     int frequency);
00167 const command_info_t *find_command_info(char *command);
00168 action_request_t *dequeue_highest_priority_action(action_queue_t *queue);
00169 void free_action_request(action_request_t *action);
00170 void insert_action_by_priority(action_queue_t *queue,
00171     action_request_t *action);
00172
```

```
00173 /* This is the definition of the array function of the commands */
00174 int handle_forward(player_t *player, char *command, zappy_t *zappy);
00175
00176 int handle_left(player_t *player, char *command, zappy_t *zappy);
00177 int left_message(player_t *player);
00178 int print_left_server(player_t *player);
00179
00180 int handle_right(player_t *player, char *command, zappy_t *zappy);
00181 int print_right_server(player_t *player);
00182 int right_message(player_t *player);
00183
00184 int handle_connect_nbr(player_t *player, char *command, zappy_t *zappy);
00185 int handle_eject(player_t *player, char *command, zappy_t *zappy);
00186
00187 int handle_fork(player_t *player, char *command, zappy_t *zappy);
00188 int print_look_server(player_t *player);
00189
00190 /* Incantation handler */
00191 int handle_incantation(player_t *player, char *command, zappy_t *zappy);
00192 int check_player_on_tile(player_t *player, zappy_t *zappy);
00193 void increase_level_player(int *player_list, int nb_players, zappy_t *zappy);
00194 int *get_player_on_tile_id(int posX, int posY, zappy_t *zappy, int nb_players);
00195 int handle_end_incantation(player_t *player, zappy_t *zappy);
00196 int get_nb_player_on_tile(int posX, int posY, zappy_t *zappy, int level);
00197 void mark_players_incanting(int *player_list, int nb_players, zappy_t *zappy);
00198 void remove_crystal_from_tiles(int posX, int posY, int level, zappy_t *zappy);
00199 int validate_and_get_players(player_t *player, zappy_t *zappy,
00200     int **player_list);
00201
00202
00203 int handle_inventory(player_t *player, char *command, zappy_t *zappy);
00204 int inventory_message(player_t *player);
00205 int print_inventory_server(player_t *player, int len);
00206
00207 int handle_broadcast(player_t *player, char *command, zappy_t *zappy);
00208 int handle_look(player_t *player, char *command, zappy_t *zappy);
00209 int handle_set(player_t *player, char *command, zappy_t *zappy);
00210 int handle_take(player_t *player, char *command, zappy_t *zappy);
00211
00212 /* graphic_clinet.c */
00213 graph_net_t *add_graph_node(graph_net_t **head, int fd);
00214 graph_net_t *remove_graph_node(graph_net_t **head, int fd);
00215 void poll_graphic_clients(zappy_t *zappy);
00216
00217
00218 /* Element hander.c */
00219 void add_food(inventory_t *inventory);
00220 void add_linemate(inventory_t *inventory);
00221 void add_deraumere(inventory_t *inventory);
00222 void add_sibur(inventory_t *inventory);
00223 void add_mendiane(inventory_t *inventory);
00224 void add_phiras(inventory_t *inventory);
00225 void add_thystame(inventory_t *inventory);
00226
00227 void rm_food(inventory_t *inventory);
00228 void rm_linemate(inventory_t *inventory);
00229 void rm_deraumere(inventory_t *inventory);
00230 void rm_sibur(inventory_t *inventory);
00231 void rm_mendiane(inventory_t *inventory);
00232 void rm_phiras(inventory_t *inventory);
00233 void rm_thystame(inventory_t *inventory);
00234 #endif /* !ZAPPY_H_ */
```

5.41 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_ */
```

5.42 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_ */
```

5.43 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_ */
```

5.44 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_ */
```

5.45 fake_malloc.h

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** B-YEP-400-NAN-4-1-zappy-albane.merian
00004 ** File description:
00005 ** fake_malloc
00006 */
00007
00008 #ifndef FAKE_MALLOC_H_
00009     #define FAKE_MALLOC_H_
00010
00011
00012 void enable_malloc_failure(int after_calls);
00013 void disable_malloc_failure(void);
00014 void reset_malloc_counter(void);
00015 void *malloc(size_t size);
00016 void *calloc(size_t nmemb, size_t size);
00017
00018 #endif /* !FAKE_MALLOC_H_ */
```

Index