# Zappy GUI

Generated by Doxygen 1.9.6

# Chapter 1

# Zappy - GUI

Welcome to the GUI component of the Zappy project! This document will guide you through the setup, development, and contribution processes for the GUI part of the project.

## 1.1 Project Overview

The Zappy project is a multiplayer, real-time strategy game developed as part of the Epitech Secondary year curriculum. The GUI component provides a graphical interface for players to interact with the game, visualize game states, and issue commands.

### 1.1.1 Features

- Real-time updates of game state
- Interactive map and player controls
- Visualization of game statistics and resources

## 1.2 Installation

To get started with the GUI, follow these steps:

1. **Clone the repository:**

```
git clone https://github.com/yourusername/zappy.git
cd Zappy/gui
```

1. **Clone the repository:**

Ensure you have the necessary tools and libraries installed, such as `raylib` and a C++ compiler.

### 1.2.1 Installation Steps

Make sure you have an up-to-date version of Fedora and that you have administrator privileges (sudo).

#### 1.2.1.1 Update Your System

Before installing new packages, it is recommended to update your system:
```
sudo dnf update
```

#### 1.2.1.2 Install Necessary Dependencies

Raylib requires certain libraries to function properly. Install them with the following command:
```
sudo dnf install alsa-lib-devel mesa-libGL-devel libX11-devel libXrandr-devel libXi-devel libXcursor-devel
      libXinerama-devel libatomic
```

Fedora offers Raylib directly in its repositories. You can install it using dnf:
```
sudo dnf install raylib-devel
```

**To make it easier, you can install everything in one command:**
```
make install-deps
```

1. **Build the project**

At the root of the Zappy project:
```
make
```

## 1.3 Usage

To run the GUI application:

1. Ensure that the server component of Zappy is running and accessible.
2. Start the GUI application:

```
./zappy_gui -p Server_port -h Server_port
```

1. Connect to the server using the GUI and start interacting with the game.

## 1.4 Development

### 1.4.1 Project Structure

- src/ - Source code for the GUI application
- assets/ - Images, icons, and other graphical assets
- include/ - Header files
- ..tests/gui/tests/ - Unit and integration tests

### 1.4.2 Running Tests

To run the tests at the root of the Zappy project:
```
make tests_run
```

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

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

# Chapter 3

# Class Index

## 3.1 Class List

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

# Chapter 4

# File Index

## 4.1 File List

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

# Chapter 5

# Class Documentation

## 5.1 Gui::Errors::AError Class Reference

Base class for custom error types. This class is derived from the `IError` interface and provides a common base for custom error types. It contains a protected member `_message` to store the error message.

`#include <AError.hpp>`

Inheritance diagram for Gui::Errors::AError:

Collaboration diagram for Gui::Errors::AError:

### Public Member Functions

- ∼**AError** () override=default

  *Destructor.*
- const char ∗ what () const noexcept override

  *Returns the error message.*

#### Public Member Functions inherited from Gui::Errors::IError

- virtual ∼**IError** ()=default

  *Destructor for IError.*
- virtual const char ∗ what () const noexcept=0

  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

### Protected Attributes

- std::string **_message**

  *The error message.*

### 5.1.1 Detailed Description

Base class for custom error types. This class is derived from the `IError` interface and provides a common base for custom error types. It contains a protected member `_message` to store the error message.

### 5.1.2 Member Function Documentation

#### 5.1.2.1 what()

```
const char * Gui::Errors::AError::what ( ) const  [override], [virtual], [noexcept]
```

Returns the error message.

**Returns**

A pointer to a constant character string representing the error message.

Implements Gui::Errors::IError.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/AError.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Error/Error.cpp

## 5.2 Gui::AEvent Class Reference

Inheritance diagram for Gui::AEvent:

## 5.3 Gui::AGUIUpdater Class Reference

Inheritance diagram for Gui::AGUIUpdater:

Collaboration diagram for Gui::AGUIUpdater:

### Public Member Functions

- AGUIUpdater (std::shared_ptr< GameData > gameData, std::shared_ptr< INetwork > network)

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

  *Destroy the AGUIUpdater object.*
- void update (const std::string &command, const std::vector< std::string > &data) override=0

  *Update the GUI GameData.*

**Public Member Functions inherited from Gui::IGUIUpdater**

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

  *Destroy the IGUIUpdater object.*
- virtual void update (const std::string &command, const std::vector< std::string > &data)=0

  *Update the GUI GameData.*

## Protected Attributes

- std::shared_ptr< GameData > **_gameData**

  *The GUI GameData to update.*
- std::shared_ptr< INetwork > **_network**

  *The network to send commands to the server.*

### 5.3.1 Constructor & Destructor Documentation

#### 5.3.1.1 AGUIUpdater()

```
Gui::AGUIUpdater::AGUIUpdater (
            std::shared_ptr< GameData > gameData,
            std::shared_ptr< INetwork > network )
```

Construct a new AGUIUpdater object.

**Parameters**

| | |
|---|---|
| *gameData* | The GUI GameData to update. |
| *network* | The network to send commands to the server. |

### 5.3.2 Member Function Documentation

#### 5.3.2.1 update()

```
void Gui::AGUIUpdater::update (
            const std::string & command,
            const std::vector< std::string > & data )  [override], [pure virtual]
```

Update the GUI GameData.

Implements Gui::IGUIUpdater.

Implemented in Gui::GUIUpdater.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GUIUpdater/AGUIUpdater.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GUIUpdater/AGUIUpdater.cpp

## 5.4 Gui::AHud Class Reference

Inheritance diagram for Gui::AHud:

Collaboration diagram for Gui::AHud:

### Public Member Functions

- ~AHud ()=default
  *Destroy the AHud object.*
- virtual void display ()=0
  *Display Hud.*
- void setPlayer (std::shared_ptr< Player > player)
  *Set the Player object.*
- void setTile (std::shared_ptr< Tile > tile)
  *Set the Tile object.*
- TypeScene getType () const
  *Get the Type object.*
- void setType (TypeScene type)
  *Set the Type object.*

#### Public Member Functions inherited from Gui::IHud

- virtual ~IHud ()=default
  *Destroy the IHud object.*
- virtual void display ()=0
  *Display the Hud.*
- virtual void setPlayer (std::shared_ptr< Player > player)=0
  *Set the Player object.*
- virtual void setTile (std::shared_ptr< Tile > tile)=0
  *Set the Tile object.*
- virtual TypeScene getType () const =0
  *Get the Type object.*
- virtual void setType (TypeScene type)=0
  *Set the Type object.*

### Protected Attributes

- TypeScene **_typeScene**
  *Type of the scene.*
- std::shared_ptr< GameData > **_gameData**
  *GameData class.*
- std::shared_ptr< Player > **_player**
  *Player to display hud.*
- std::shared_ptr< Tile > **_tile**
  *Tile to display hud.*

## Additional Inherited Members

### Public Types inherited from Gui::IHud

- enum TypeScene {
  **GAME** , **POV_PLAYER** , **END_GAME** , **TILE** ,
  **HELP_TEXT** , **HELP_MENU** , **END** }
  *Hud enum for the different scenes.*

## 5.4.1 Constructor & Destructor Documentation

#### 5.4.1.1 ~AHud()

```
Gui::AHud::~AHud ( )  [default]
```

Destroy the AHud object.

## 5.4.2 Member Function Documentation

#### 5.4.2.1 display()

```
virtual void Gui::AHud::display ( )  [pure virtual]
```

Display Hud.

Implements Gui::IHud.

Implemented in Gui::HudEnd, Gui::HudGame, Gui::HudHelp, Gui::HudPlayer, and Gui::HudTile.

#### 5.4.2.2 getType()

```
Gui::AHud::TypeScene Gui::AHud::getType ( ) const  [virtual]
```

Get the Type object.

**Returns**

TypeScene - Type of the scene.

Implements Gui::IHud.

#### 5.4.2.3 setPlayer()

```
void Gui::AHud::setPlayer (
            std::shared_ptr< Player > player )  [virtual]
```

Set the Player object.

**Parameters**

| | |
|---|---|
| *player* | Player to display infos. |

Implements Gui::IHud.

#### 5.4.2.4 setTile()

```
void Gui::AHud::setTile (
            std::shared_ptr< Tile > tile )  [virtual]
```

Set the Tile object.

**Parameters**

| | |
|---|---|
| *tile* | Tile to display infos. |

Implements Gui::IHud.

#### 5.4.2.5 setType()

```
void Gui::AHud::setType (
            TypeScene type )  [virtual]
```

Set the Type object.

**Parameters**

| | |
|---|---|
| *type* | Type of the scene. |

Implements Gui::IHud.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Hud/AHud.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Hud/AHud.cpp

## 5.5 Gui::ANetwork Class Reference

Inheritance diagram for Gui::ANetwork:

Collaboration diagram for Gui::ANetwork:

## Public Member Functions

- ANetwork (int port, const std::string &hostName)

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

  *Destroy the ANetwork object.*
- void setPort (int port) final

  *Set the port object.*
- void setHostName (const std::string &hostName) final

  *Set the host name object.*
- int getPort () const final

  *Get the host name object.*
- std::string getHostName () const final

  *Get the host name object.*
- virtual void connectToServer ()=0

  *Connect to the server.*
- virtual BufferState listenServer ()=0

  *Listen the server and return it message.*
- virtual void sendMessageServer (const std::string &message)=0

  *Send a message to the Server.*
- std::string getBuffer ()

  *Get the Buffer object. Be careful, this method will delete the current buffer.*

### Public Member Functions inherited from Gui::INetwork

- virtual ~INetwork ()=default

  *Destroy the INetwork object.*
- virtual void setPort (int port)=0

  *Set the port object.*
- virtual void setHostName (const std::string &hostName)=0

  *Set the host name object.*
- virtual int getPort () const =0

  *Get the host name object.*
- virtual std::string getHostName () const =0

  *Get the host name object.*
- virtual void connectToServer ()=0

  *Connect to the server.*
- virtual BufferState listenServer ()=0

  *Listen to the server.*
- virtual void sendMessageServer (const std::string &message)=0

  *Send a message to the server.*
- virtual std::string getBuffer ()=0

  *Get the Buffer object. Be careful, this method will delete the current buffer.*

## Protected Attributes

- int **_port**

  *Port of the server.*
- std::string **_hostName**

  *Host name of the server.*
- std::string **_buffer**

  *Buffer to receive server message.*

## Additional Inherited Members

### Public Types inherited from Gui::INetwork

- enum **BufferState** { **NONE** , **READY** , **SERVER_ERROR** }

## 5.5.1 Constructor & Destructor Documentation

### 5.5.1.1 ANetwork()

```
Gui::ANetwork::ANetwork (
            int port,
            const std::string & hostName )
```

Construct a new ANetwork object.

**Parameters**

| | |
|---|---|
| *port* | Port of the server. |
| *hostName* | Host of the server. |

### 5.5.1.2 ~ANetwork()

```
Gui::ANetwork::~ANetwork ( )  [default]
```

Destroy the ANetwork object.

## 5.5.2 Member Function Documentation

### 5.5.2.1 connectToServer()

```
virtual void Gui::ANetwork::connectToServer ( )  [pure virtual]
```

Connect to the server.

**Exceptions**

| | |
|---|---|
| *NetworkException* | If the connection failed. |

Implements Gui::INetwork.

Implemented in Gui::Network.

#### 5.5.2.2 getBuffer()

```
std::string Gui::ANetwork::getBuffer ( )  [virtual]
```

Get the Buffer object. Be careful, this method will delete the current buffer.

**Returns**

std::string - Buffer message.

Implements Gui::INetwork.

#### 5.5.2.3 getHostName()

```
std::string Gui::ANetwork::getHostName ( ) const  [final], [virtual]
```

Get the host name object.

**Returns**

std::string Host of the server.

Implements Gui::INetwork.

#### 5.5.2.4 getPort()

```
int Gui::ANetwork::getPort ( ) const  [final], [virtual]
```

Get the host name object.

**Returns**

std::string Host of the server.

Implements Gui::INetwork.

#### 5.5.2.5 listenServer()

```
virtual BufferState Gui::ANetwork::listenServer ( )  [pure virtual]
```

Listen the server and return it message.

**Returns**

BufferState - Buffer state.

Implements Gui::INetwork.

Implemented in Gui::Network.

#### 5.5.2.6 sendMessageServer()

```
virtual void Gui::ANetwork::sendMessageServer (
            const std::string & message )  [pure virtual]
```

Send a message to the Server.

**Parameters**

| | |
|---|---|
| *message* | Message to send to the server. |

Implements Gui::INetwork.

Implemented in Gui::Network.

#### 5.5.2.7 setHostName()

```
void Gui::ANetwork::setHostName (
            const std::string & hostName )  [final], [virtual]
```

Set the host name object.

**Parameters**

| | |
|---|---|
| *hostName* | Host of the server. |

Implements Gui::INetwork.

#### 5.5.2.8 setPort()

```
void Gui::ANetwork::setPort (
            int port )  [final], [virtual]
```

Set the port object.

**Parameters**

| | |
|---|---|
| *port* | Port of the server. |

**Exceptions**

| | |
|---|---|
| *NetworkException* | If the port is not in range 1 to 65535. |

Implements Gui::INetwork.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Network/ANetwork.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Network/ANetwork.cpp

## 5.6 Gui::Decoration Class Reference

### Public Member Functions

- Decoration ()

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

  *Destroy the Decoration object.*
- void display (std::pair< std::size_t, std::size_t > mapSize, size_t renderDistance, std::pair< std::size_t, std::size_t > camPos, std::vector< Vector2 > displayPos)

  *Display decorations.*
- Map< bool > getGenerationItem (std::size_t ratio)

  *Generate random emplacement for decorations.*

### 5.6.1 Constructor & Destructor Documentation

#### 5.6.1.1 Decoration()

```
Gui::Decoration::Decoration ( )
```

Construct a new Decoration object.

#### 5.6.1.2 ~Decoration()

```
Gui::Decoration::~Decoration ( )  [default]
```

Destroy the Decoration object.

### 5.6.2 Member Function Documentation

#### 5.6.2.1 display()

```
void Gui::Decoration::display (
            std::pair< std::size_t, std::size_t > mapSize,
            size_t renderDistance,
            std::pair< std::size_t, std::size_t > camPos,
            std::vector< Vector2 > displayPos )
```

Display decorations.

**Parameters**

| | |
|---|---|
| *mapSize* | Size of the map. |
| *renderDistance* | Distance to render. |
| *camPos* | Position of the camera. |
| *displayPos* | Positions to know what to display. |

#### 5.6.2.2 getGenerationItem()

```
Map< bool > Gui::Decoration::getGenerationItem (
            std::size_t ratio )
```

Generate random emplacement for decorations.

**Parameters**

| | |
|---|---|
| *ratio* | Ratio fo random emplacement. If ratio = 10 for a tree, there will be one chance at a ten to have a tree on the tile. |

**Returns**

Map<bool> - Boolean list to display item.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Render/Decoration.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Render/Decoration.cpp

## 5.7 Gui::Egg Class Reference

### Public Types

- enum **EggState** { **IDLE** , **DEAD** , **BORN** , **HATCHING** }

### Public Member Functions

- Egg (size_t id, const std::string &team, std::pair< std::size_t, std::size_t > position)
  *Construct a new Egg object.*
- ~Egg ()
  *Destroy the Egg object.*
- std::size_t getId () const
  *Get the Id object.*
- std::string getTeam () const
  *Get the Team object.*
- std::pair< std::size_t, std::size_t > getPosition () const
  *Get the Position object.*
- void setId (std::size_t id)
  *Set the id object.*
- void setTeam (const std::string &team)
  *Set the team object.*
- void setPosition (std::pair< std::size_t, std::size_t > position)
  *Set the position object.*
- void setState (EggState state)
  *Set the state object.*
- EggState getState () const
  *Get the state object.*

### 5.7.1 Constructor & Destructor Documentation

#### 5.7.1.1 Egg()

```
Gui::Egg::Egg (
            size_t id,
            const std::string & team,
            std::pair< std::size_t, std::size_t > position )
```

Construct a new Egg object.

**Parameters**

| | |
|---|---|
| *id* | Id of the egg. |
| *team* | Team name of the egg. |
| *position* | Position of the egg. |

**Note**

The egg is created when a player lays an egg.

The constructor starts the egg animation if implemented.

#### 5.7.1.2 ~Egg()

```
Gui::Egg::~Egg ( )
```

Destroy the Egg object.

**Note**

The destructor starts the egg animation if implemented.

### 5.7.2 Member Function Documentation

#### 5.7.2.1 getId()

```
std::size_t Gui::Egg::getId ( ) const
```

Get the Id object.

**Returns**

std::size_t Id of the egg.

#### 5.7.2.2 getPosition()

```
std::pair< std::size_t, std::size_t > Gui::Egg::getPosition ( ) const
```

Get the Position object.

**Returns**

std::pair<std::size_t, std::size_t> Position of the egg.

#### 5.7.2.3 getState()

```
Gui::Egg::EggState Gui::Egg::getState ( ) const
```

Get the state object.

**Returns**

EggState State of the egg.

#### 5.7.2.4 getTeam()

```
std::string Gui::Egg::getTeam ( ) const
```

Get the Team object.

**Returns**

std::string Team name of the egg.

#### 5.7.2.5 setId()

```
void Gui::Egg::setId (
            std::size_t id )
```

Set the id object.

**Parameters**

| | |
|---|---|
| *id* | Id of the egg. |

#### 5.7.2.6 setPosition()

```
void Gui::Egg::setPosition (
            std::pair< std::size_t, std::size_t > position )
```

Set the position object.

**Parameters**

| | |
|---|---|
| *position* | Position of the egg. |

#### 5.7.2.7 setState()

```
void Gui::Egg::setState (
            EggState state )
```

Set the state object.

**Parameters**

| | |
|---|---|
| *state* | State of the egg. |

#### 5.7.2.8 setTeam()

```
void Gui::Egg::setTeam (
            const std::string & team )
```

Set the team object.

**Parameters**

| | |
|---|---|
| *team* | Team name of the egg. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GameDatas/Egg.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GameDatas/Egg.cpp

## 5.8 Gui::Engine Class Reference

### Public Member Functions

- Engine (std::shared_ptr< INetwork > network)

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

  *Destroy the Engine object.*
- void run ()

  *Run the engine loop.*

### 5.8.1 Constructor & Destructor Documentation

#### 5.8.1.1 Engine()

```
Gui::Engine::Engine (
            std::shared_ptr< INetwork > network )
```

Construct a new Engine object.

**Parameters**

| | |
|---|---|
| *network* | Network class. |

#### 5.8.1.2 ~Engine()

```
Gui::Engine::~Engine ( )  [default]
```

Destroy the Engine object.

### 5.8.2 Member Function Documentation

#### 5.8.2.1 run()

```
void Gui::Engine::run ( )
```

Run the engine loop.

**Note**

This method runs in the main thread.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Engine/Engine.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Engine/Engine.cpp

## 5.9 Gui::Errors::Error Class Reference

Base class for argument-related errors.

```
#include <Error.hpp>
```

Inheritance diagram for Gui::Errors::Error:

Collaboration diagram for Gui::Errors::Error:

### Additional Inherited Members

#### Public Member Functions inherited from Gui::Errors::AError

- ~**AError** () override=default

  *Destructor.*
- const char * what () const noexcept override

  *Returns the error message.*

#### Public Member Functions inherited from Gui::Errors::IError

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

  *Destructor for IError.*
- virtual const char * what () const noexcept=0

  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

#### Protected Attributes inherited from Gui::Errors::AError

- std::string **_message**

  *The error message.*

### 5.9.1 Detailed Description

Base class for argument-related errors.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/Error.hpp

## 5.10 Gui::Event Class Reference

Inheritance diagram for Gui::Event:

Collaboration diagram for Gui::Event:

### Public Member Functions

- Event ()

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

  *Destroy the Event object.*
- void listen ()

  *Listen the user's events.*

#### Public Member Functions inherited from Gui::AEvent

- AEvent ()

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

  *Destroy the AEvent object.*
- virtual void listen ()=0

  *Listen the user's events.*
- void setRender (std::shared_ptr< Render > render)

  *Set the Render object.*
- void setGameData (std::shared_ptr< GameData > gameData)

  *Set the GameData object.*
- std::shared_ptr< Render > getRender ()

  *Get the Render object.*
- std::shared_ptr< GameData > getGameData ()

  *Get the GameData object.*

#### Public Member Functions inherited from Gui::IEvent

- IEvent ()=default

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

  *Destroy the IEvent object.*
- virtual void listen ()=0

  *Listen the user's events.*
- virtual void setRender (std::shared_ptr< Render > render)=0

  *Set the Render object.*
- virtual void setGameData (std::shared_ptr< GameData > gameData)=0

  *Set the GameData object.*

## Additional Inherited Members

#### Protected Attributes inherited from Gui::AEvent

- std::shared_ptr< Render > **_render**

  *Render class to draw scene.*
- std::shared_ptr< GameData > **_gameData**

  *GameData class to contain scene.*

### 5.10.1 Constructor & Destructor Documentation

#### 5.10.1.1 Event()

```
Gui::Event::Event ( )
```

Construct a new Event object.

#### 5.10.1.2 ~Event()

```
Gui::Event::~Event ( )  [default]
```

Destroy the Event object.

### 5.10.2 Member Function Documentation

#### 5.10.2.1 listen()

```
void Gui::Event::listen ( )  [virtual]
```

Listen the user's events.

Implements Gui::AEvent.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Event/Event.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Event/Event.cpp

## 5.11 Gui::GameData Class Reference

### Public Types

- enum **TimeUnitState** { **INCREASE** , **DECREASE** , **NONE** }

### Public Member Functions

- GameData ()
  *Construct a new GameData object.*
- ~GameData ()=default
  *Destroy the GameData object.*
- std::vector< Gui::Team > & getTeams ()
  *Get the Teams object.*
- Gui::Team & getTeam (const std::string &name)
  *Get a Team object.*
- void addTeam (const Gui::Team &team)
  *Add a team to the game.*
- void addTeam (const std::string &name, Color playerColor)
  *Add a team to the game.*
- void addPlayerToTeam (const std::string &teamName, const Gui::Player &player)
  *Add a player to a team.*
- Gui::Player & getPlayer (size_t id)
  *Get a player object.*

- Map< Gui::Tile > & getMap ()

  *Get the Map object.*
- void setMap (const Map< Gui::Tile > &map)

  *Set the Map object.*
- void setMapSize (size_t x, size_t y)

  *Set the Map Size object.*
- std::pair< size_t, size_t > getMapSize () const

  *Get the Map Size object.*
- Gui::Tile & getTile (size_t x, size_t y)

  *Get a Tile object.*
- void setTile (const Gui::Tile &tile)

  *Set the Tile object.*
- void restartLastTick (void)

  *Restart the last tick clock.*
- void setServerTick (std::size_t tick)

  *Set the Server Tick object.*
- clock_t getLastTick () const

  *Get the Last Tick object.*
- std::size_t getServerTick () const

  *Get the Server Tick object.*
- void setIsEndGame (bool isEndGame)

  *Set the IsEnd Game object.*
- bool getIsEndGame () const

  *Get the IsEnd Game object.*
- void setLastError (const std::string &error)

  *Set the Last Error object.*
- std::string getLastError () const

  *Get the Last Error object.*
- Team & getTeamById (std::size_t id)

  *Get the Team From Player object.*
- TimeUnitState getTimeUnitFromServer () const

  *Get the Time Unit From Server object.*
- void setTimeUnitFromServer (TimeUnitState timeUnitFromServer)

  *Set the Time Unit From Server object.*
- std::vector< Gui::Egg > & getServerEggs ()

  *Get the Server Eggs object.*
- void addServerEgg (const Gui::Egg &egg)

  *Add an egg to the server ones.*
- void removeServerEgg (size_t id)

  *Remove an egg from the server ones.*
- void setNbBCTCommandReceived (std::size_t nb)

  *Set the number of bct command received.*
- std::size_t getNbBCTCommandReceived () const

  *Get the number of bct command received.*
- void restartLastTickMctCommand ()

  *Restart the last tick mct command clock.*
- clock_t getLastTickMctCommand () const

  *Get the Last Tick mct command object.*
- void setEndMessage (const std::string &endMessage)

  *Set the End Message object.*
- std::string getEndMessage () const

*Get the End Message object.*

- void setPlayerModel (const Model &playerModel)

  *Set the Player Model object.*
- void setEggModel (const Model &eggModel)

  *Set the Egg Model object.*
- void setPlayerModelAnimation (ModelAnimation *playerModelAnimation)

  *Set the Player Model Animation object.*
- void setAnimsCount (int animsCount)

  *Get the Player Model Animation object.*
- void setServerError (bool isServerError)

  *Get the Anims Count object.*
- bool getServerError () const

  *Get the Server Error object.*

## 5.11.1 Constructor & Destructor Documentation

### 5.11.1.1 GameData()

```
Gui::GameData::GameData ( )
```

Construct a new GameData object.

### 5.11.1.2 ~GameData()

```
Gui::GameData::~GameData ( )  [default]
```

Destroy the GameData object.

## 5.11.2 Member Function Documentation

### 5.11.2.1 addPlayerToTeam()

```
void Gui::GameData::addPlayerToTeam (
            const std::string & teamName,
            const Gui::Player & player )
```

Add a player to a team.

**Parameters**

| | |
|---|---|
| *teamName* | Name of the team. |
| *player* | Player to add. |

#### 5.11.2.2 addServerEgg()

```
void Gui::GameData::addServerEgg (
            const Gui::Egg & egg )
```

Add an egg to the server ones.

**Parameters**

| | |
|---|---|
| *egg* | Egg to add. |

#### 5.11.2.3 addTeam() [1/2]

```
void Gui::GameData::addTeam (
            const Gui::Team & team )
```

Add a team to the game.

**Parameters**

| | |
|---|---|
| *team* | Team to add. |

#### 5.11.2.4 addTeam() [2/2]

```
void Gui::GameData::addTeam (
            const std::string & name,
            Color playerColor )
```

Add a team to the game.

**Parameters**

| | |
|---|---|
| *name* | Name of the team. |
| *playerColor* | Color of the team. |

#### 5.11.2.5 getEndMessage()

```
std::string Gui::GameData::getEndMessage ( ) const
```

Get the End Message object.

**Returns**

std::string - End message of the game.

#### 5.11.2.6 getIsEndGame()

```
bool Gui::GameData::getIsEndGame ( ) const
```

Get the IsEnd Game object.

**Returns**

true - The game is finished.

false - The game continue.

#### 5.11.2.7 getLastError()

```
std::string Gui::GameData::getLastError ( ) const
```

Get the Last Error object.

**Returns**

std::string - Last error message.

#### 5.11.2.8 getLastTick()

```
clock_t Gui::GameData::getLastTick ( ) const
```

Get the Last Tick object.

**Returns**

clock_t - Last Tick.

#### 5.11.2.9 getLastTickMctCommand()

```
clock_t Gui::GameData::getLastTickMctCommand ( ) const
```

Get the Last Tick mct command object.

**Returns**

clock_t - Last Tick Mct command.

#### 5.11.2.10 getMap()

```
Map< Gui::Tile > & Gui::GameData::getMap ( )
```

Get the Map object.

**Returns**

Map<Gui::Tile>& Map of the game.

#### 5.11.2.11 getMapSize()

```
std::pair< size_t, size_t > Gui::GameData::getMapSize ( ) const
```

Get the Map Size object.

**Returns**

std::pair<size_t, size_t> Size of the map.

#### 5.11.2.12 getNbBCTCommandReceived()

```
std::size_t Gui::GameData::getNbBCTCommandReceived ( ) const
```

Get the number of bct command received.

**Returns**

std::size_t - Number of bct command received.

#### 5.11.2.13 getPlayer()

```
Gui::Player & Gui::GameData::getPlayer (
            size_t id )
```

Get a player object.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |

#### 5.11.2.14 getServerEggs()

```
std::vector< Gui::Egg > & Gui::GameData::getServerEggs ( )
```

Get the Server Eggs object.

**Returns**

std::vector<Gui::Egg>& Eggs from the server.

#### 5.11.2.15 getServerError()

```
bool Gui::GameData::getServerError ( ) const
```

Get the Server Error object.

**Returns**

bool - True if the server is in error.

#### 5.11.2.16 getServerTick()

```
std::size_t Gui::GameData::getServerTick ( ) const
```

Get the Server Tick object.

**Returns**

std::size_t - Server Tick.

#### 5.11.2.17 getTeam()

```
Gui::Team & Gui::GameData::getTeam (
            const std::string & name )
```

Get a Team object.

**Parameters**

| | |
|---|---|
| *name* | Name of the team. |

**Returns**

Gui::Team& Team object.

#### 5.11.2.18 getTeamById()

```
Gui::Team & Gui::GameData::getTeamById (
            std::size_t id )
```

Get the Team From Player object.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |

**Returns**

Gui::Team& Team of the player.

#### 5.11.2.19 getTeams()

```
std::vector< Gui::Team > & Gui::GameData::getTeams ( )
```

Get the Teams object.

**Returns**

std::vector<Gui::Team>& Teams of the game.

#### 5.11.2.20 getTile()

```
Gui::Tile & Gui::GameData::getTile (
            size_t x,
            size_t y )
```

Get a Tile object.

**Parameters**

| | |
|---|---|
| *x* | X position of the tile. |
| *y* | Y position of the tile. |

**Returns**

Gui::Tile& Tile object.

#### 5.11.2.21 getTimeUnitFromServer()

```
Gui::GameData::TimeUnitState Gui::GameData::getTimeUnitFromServer ( ) const
```

Get the Time Unit From Server object.

**Returns**

true - The time unit has changed.

#### 5.11.2.22 removeServerEgg()

```
void Gui::GameData::removeServerEgg (
            size_t id )
```

Remove an egg from the server ones.

**Parameters**

| | |
|---|---|
| *id* | Id of the egg. |

#### 5.11.2.23 restartLastTick()

```
void Gui::GameData::restartLastTick (
            void  )
```

Restart the last tick clock.

#### 5.11.2.24 restartLastTickMctCommand()

```
void Gui::GameData::restartLastTickMctCommand ( )
```

Restart the last tick mct command clock.

#### 5.11.2.25 setAnimsCount()

```
void Gui::GameData::setAnimsCount (
            int animsCount )
```

Get the Player Model Animation object.

**Returns**

ModelAnimation* - Model to animate players.

#### 5.11.2.26 setEggModel()

```
void Gui::GameData::setEggModel (
            const Model & eggModel )
```

Set the Egg Model object.

**Parameters**

| | |
|---|---|
| *eggModel* | Model asset of the Team. |

#### 5.11.2.27 setEndMessage()

```
void Gui::GameData::setEndMessage (
            const std::string & endMessage )
```

Set the End Message object.

**Parameters**

| | |
|---|---|
| *endMessage* | End message of the game. |

#### 5.11.2.28 setIsEndGame()

```
void Gui::GameData::setIsEndGame (
            bool isEndGame )
```

Set the IsEnd Game object.

**Parameters**

| | |
|---|---|
| *isEndGame* | EndGame state. |

#### 5.11.2.29 setLastError()

```
void Gui::GameData::setLastError (
            const std::string & error )
```

Set the Last Error object.

**Parameters**

| | |
|---|---|
| *error* | Error message. |

#### 5.11.2.30 setMap()

```
void Gui::GameData::setMap (
            const Map< Gui::Tile > & map )
```

Set the Map object.

**Parameters**

| | |
|---|---|
| *map* | Map of the game. |

#### 5.11.2.31 setMapSize()

```
void Gui::GameData::setMapSize (
            size_t x,
            size_t y )
```

Set the Map Size object.

**Parameters**

| | |
|---|---|
| *x* | X size of the map. |
| *y* | Y size of the map. |

**Note**

This method resizes the map.

#### 5.11.2.32 setNbBCTCommandReceived()

```
void Gui::GameData::setNbBCTCommandReceived (
            std::size_t nb )
```

Set the number of bct command received.

**Parameters**

| | |
|---|---|
| *nb* | Number of bct command received. |

#### 5.11.2.33 setPlayerModel()

```
void Gui::GameData::setPlayerModel (
            const Model & playerModel )
```

Set the Player Model object.

**Parameters**

| | |
|---|---|
| *playerModel* | Model asset of the Team. |

#### 5.11.2.34 setPlayerModelAnimation()

```
void Gui::GameData::setPlayerModelAnimation (
            ModelAnimation * playerModelAnimation )
```

Set the Player Model Animation object.

**Parameters**

| | |
|---|---|
| *playerModelAnimation* | Model to animate players. |

#### 5.11.2.35 setServerError()

```
void Gui::GameData::setServerError (
```

```
            bool isServerError )
```

Get the Anims Count object.

**Returns**

int - Animation number of players.

#### 5.11.2.36 setServerTick()

```
void Gui::GameData::setServerTick (
            std::size_t tick )
```

Set the Server Tick object.

**Parameters**

| | |
|---|---|
| *tick* | Tick of the server. |

#### 5.11.2.37 setTile()

```
void Gui::GameData::setTile (
            const Gui::Tile & tile )
```

Set the Tile object.

**Parameters**

| | |
|---|---|
| *x* | X position of the tile. |
| *y* | Y position of the tile. |
| *tile* | Tile to set. |

#### 5.11.2.38 setTimeUnitFromServer()

```
void Gui::GameData::setTimeUnitFromServer (
            TimeUnitState timeUnitFromServer )
```

Set the Time Unit From Server object.

**Parameters**

| | |
|---|---|
| *timeUnitFromServer* | Time unit state. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GameDatas/GameData.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GameDatas/GameData.cpp

## 5.12 Gui::Errors::GuiGameDataException Class Reference

Error class for GameData errors.

`#include <Error.hpp>`

Inheritance diagram for Gui::Errors::GuiGameDataException:

Collaboration diagram for Gui::Errors::GuiGameDataException:

### Public Member Functions

- GuiGameDataException (std::string message)

  *Constructor for GuiGameDataException.*

#### Public Member Functions inherited from Gui::Errors::AError

- ~**AError** () override=default

  *Destructor.*
- const char * what () const noexcept override

  *Returns the error message.*

#### Public Member Functions inherited from Gui::Errors::IError

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

  *Destructor for IError.*
- virtual const char * what () const noexcept=0

  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

### Additional Inherited Members

#### Protected Attributes inherited from Gui::Errors::AError

- std::string **_message**

  *The error message.*

### 5.12.1 Detailed Description

Error class for GameData errors.

### 5.12.2 Constructor & Destructor Documentation

#### 5.12.2.1 GuiGameDataException()

```
Gui::Errors::GuiGameDataException::GuiGameDataException (
            std::string message )
```

Constructor for GuiGameDataException.

**Parameters**

| | |
|---|---|
| *message* | The error message. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/Error.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Error/Error.cpp

## 5.13 Gui::GUIUpdater Class Reference

Inheritance diagram for Gui::GUIUpdater:

Collaboration diagram for Gui::GUIUpdater:

### Public Member Functions

- GUIUpdater (std::shared_ptr< GameData > gameData, std::shared_ptr< INetwork > network)
  *Construct a new GUIUpdater object.*
- ~**GUIUpdater** ()=default
  *Destroy the GUIUpdater object.*
- void update (const std::string &command, const std::vector< std::string > &data)
  *Update the GUI GameData.*

#### Public Member Functions inherited from Gui::AGUIUpdater

- AGUIUpdater (std::shared_ptr< GameData > gameData, std::shared_ptr< INetwork > network)
  *Construct a new AGUIUpdater object.*
- ~**AGUIUpdater** ()=default
  *Destroy the AGUIUpdater object.*
- void update (const std::string &command, const std::vector< std::string > &data) override=0
  *Update the GUI GameData.*

#### Public Member Functions inherited from Gui::IGUIUpdater

- virtual ~**IGUIUpdater** ()=default
  *Destroy the IGUIUpdater object.*
- virtual void update (const std::string &command, const std::vector< std::string > &data)=0
  *Update the GUI GameData.*

### Additional Inherited Members

#### Protected Attributes inherited from Gui::AGUIUpdater

- std::shared_ptr< GameData > **_gameData**
  *The GUI GameData to update.*
- std::shared_ptr< INetwork > **_network**
  *The network to send commands to the server.*

### 5.13.1 Constructor & Destructor Documentation

#### 5.13.1.1 GUIUpdater()

```
Gui::GUIUpdater::GUIUpdater (
            std::shared_ptr< GameData > gameData,
            std::shared_ptr< INetwork > network )
```

Construct a new GUIUpdater object.

**Parameters**

| | |
|---|---|
| *gameData* | The GUI GameData to update. |
| *network* | The network to send commands to the server. |

### 5.13.2 Member Function Documentation

#### 5.13.2.1 update()

```
void Gui::GUIUpdater::update (
            const std::string & command,
            const std::vector< std::string > & data )  [virtual]
```

Update the GUI GameData.

**Parameters**

| | |
|---|---|
| *command* | The command to update the GUI GameData. |
| *data* | The data to update the GUI GameData. |

Implements Gui::AGUIUpdater.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GUIUpdater/GUIUpdater.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GUIUpdater/GUIUpdater.cpp

## 5.14 Gui::Errors::GuiUpdaterException Class Reference

Error class for GUIUpdater errors.

`#include <Error.hpp>`

Inheritance diagram for Gui::Errors::GuiUpdaterException:

Collaboration diagram for Gui::Errors::GuiUpdaterException:

## Public Member Functions

- GuiUpdaterException (std::string message)

  *Constructor for GuiUpdaterException.*

### Public Member Functions inherited from Gui::Errors::AError

- ~**AError** () override=default

  *Destructor.*
- const char * what () const noexcept override

  *Returns the error message.*

### Public Member Functions inherited from Gui::Errors::IError

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

  *Destructor for IError.*
- virtual const char * what () const noexcept=0

  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

## Additional Inherited Members

### Protected Attributes inherited from Gui::Errors::AError

- std::string **_message**

  *The error message.*

## 5.14.1 Detailed Description

Error class for GUIUpdater errors.

## 5.14.2 Constructor & Destructor Documentation

### 5.14.2.1 GuiUpdaterException()

```
Gui::Errors::GuiUpdaterException::GuiUpdaterException (
            std::string message )
```

Constructor for GuiUpdaterException.

**Parameters**

| | |
|---|---|
| *message* | The error message. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/Error.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Error/Error.cpp

## 5.15 Gui::HudEnd Class Reference

Inheritance diagram for Gui::HudEnd:

Collaboration diagram for Gui::HudEnd:

### Public Member Functions

- HudEnd (std::shared_ptr< GameData > gameData)
  *Construct a new Hud End object.*
- ~HudEnd ()=default
  *Destroy the Hud End object.*
- void display ()
  *Display End Hud.*

#### Public Member Functions inherited from Gui::AHud

- ~AHud ()=default
  *Destroy the AHud object.*
- virtual void display ()=0
  *Display Hud.*
- void setPlayer (std::shared_ptr< Player > player)
  *Set the Player object.*
- void setTile (std::shared_ptr< Tile > tile)
  *Set the Tile object.*
- TypeScene getType () const
  *Get the Type object.*
- void setType (TypeScene type)
  *Set the Type object.*

#### Public Member Functions inherited from Gui::IHud

- virtual ~IHud ()=default
  *Destroy the IHud object.*
- virtual void display ()=0
  *Display the Hud.*
- virtual void setPlayer (std::shared_ptr< Player > player)=0
  *Set the Player object.*
- virtual void setTile (std::shared_ptr< Tile > tile)=0
  *Set the Tile object.*
- virtual TypeScene getType () const =0
  *Get the Type object.*
- virtual void setType (TypeScene type)=0
  *Set the Type object.*

## Additional Inherited Members

### Public Types inherited from Gui::IHud

- enum TypeScene {
  **GAME** , **POV_PLAYER** , **END_GAME** , **TILE** ,
  **HELP_TEXT** , **HELP_MENU** , **END** }
  *Hud enum for the different scenes.*

### Protected Attributes inherited from Gui::AHud

- TypeScene **_typeScene**
  *Type of the scene.*
- std::shared_ptr< GameData > **_gameData**
  *GameData class.*
- std::shared_ptr< Player > **_player**
  *Player to display hud.*
- std::shared_ptr< Tile > **_tile**
  *Tile to display hud.*

## 5.15.1 Constructor & Destructor Documentation

### 5.15.1.1 HudEnd()

```
Gui::HudEnd::HudEnd (
            std::shared_ptr< GameData > gameData )
```

Construct a new Hud End object.

**Parameters**

| | |
|---|---|
| *gameData* | GameData class. |

### 5.15.1.2 ~HudEnd()

```
Gui::HudEnd::~HudEnd ( )  [default]
```

Destroy the Hud End object.

## 5.15.2 Member Function Documentation

#### 5.15.2.1 display()

```
void Gui::HudEnd::display ( )  [virtual]
```

Display End Hud.

Implements Gui::AHud.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Hud/HudEnd.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Hud/HudEnd.cpp

## 5.16 Gui::HudGame Class Reference

Inheritance diagram for Gui::HudGame:

Collaboration diagram for Gui::HudGame:

### Public Member Functions

- HudGame (std::shared_ptr< GameData > gameData)
  *Construct a new Hud Game object.*
- ~HudGame ()=default
  *Destroy the Hud Game object.*
- void display ()
  *Display Game Hud.*

#### Public Member Functions inherited from Gui::AHud

- ~AHud ()=default
  *Destroy the AHud object.*
- virtual void display ()=0
  *Display Hud.*
- void setPlayer (std::shared_ptr< Player > player)
  *Set the Player object.*
- void setTile (std::shared_ptr< Tile > tile)
  *Set the Tile object.*
- TypeScene getType () const
  *Get the Type object.*
- void setType (TypeScene type)
  *Set the Type object.*

#### Public Member Functions inherited from Gui::IHud

- virtual ~IHud ()=default

  *Destroy the IHud object.*
- virtual void display ()=0

  *Display the Hud.*
- virtual void setPlayer (std::shared_ptr< Player > player)=0

  *Set the Player object.*
- virtual void setTile (std::shared_ptr< Tile > tile)=0

  *Set the Tile object.*
- virtual TypeScene getType () const =0

  *Get the Type object.*
- virtual void setType (TypeScene type)=0

  *Set the Type object.*

## Additional Inherited Members

#### Public Types inherited from Gui::IHud

- enum TypeScene {
  **GAME** , **POV_PLAYER** , **END_GAME** , **TILE** ,
  **HELP_TEXT** , **HELP_MENU** , **END** }

  *Hud enum for the different scenes.*

#### Protected Attributes inherited from Gui::AHud

- TypeScene **_typeScene**

  *Type of the scene.*
- std::shared_ptr< GameData > **_gameData**

  *GameData class.*
- std::shared_ptr< Player > **_player**

  *Player to display hud.*
- std::shared_ptr< Tile > **_tile**

  *Tile to display hud.*

### 5.16.1 Constructor & Destructor Documentation

#### 5.16.1.1 HudGame()

```
Gui::HudGame::HudGame (
            std::shared_ptr< GameData > gameData )
```

Construct a new Hud Game object.

**Parameters**

| | |
|---|---|
| *gameData* | GameData class. |

#### 5.16.1.2 ~HudGame()

```
Gui::HudGame::~HudGame ( )  [default]
```

Destroy the Hud Game object.

### 5.16.2 Member Function Documentation

#### 5.16.2.1 display()

```
void Gui::HudGame::display ( )  [virtual]
```

Display Game Hud.

Implements Gui::AHud.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Hud/HudGame.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Hud/HudGame.cpp

## 5.17 Gui::HudHelp Class Reference

Inheritance diagram for Gui::HudHelp:

Collaboration diagram for Gui::HudHelp:

### Public Member Functions

- HudHelp (std::shared_ptr< GameData > gameData)
  *Construct a new Hud Help object.*
- ~HudHelp ()=default
  *Destroy the Hud Help object.*
- void display ()
  *Display Help Hud.*

#### Public Member Functions inherited from Gui::AHud

- ~AHud ()=default

  *Destroy the AHud object.*
- virtual void display ()=0

  *Display Hud.*
- void setPlayer (std::shared_ptr< Player > player)

  *Set the Player object.*
- void setTile (std::shared_ptr< Tile > tile)

  *Set the Tile object.*
- TypeScene getType () const

  *Get the Type object.*
- void setType (TypeScene type)

  *Set the Type object.*

#### Public Member Functions inherited from Gui::IHud

- virtual ~IHud ()=default

  *Destroy the IHud object.*
- virtual void display ()=0

  *Display the Hud.*
- virtual void setPlayer (std::shared_ptr< Player > player)=0

  *Set the Player object.*
- virtual void setTile (std::shared_ptr< Tile > tile)=0

  *Set the Tile object.*
- virtual TypeScene getType () const =0

  *Get the Type object.*
- virtual void setType (TypeScene type)=0

  *Set the Type object.*

## Additional Inherited Members

#### Public Types inherited from Gui::IHud

- enum TypeScene {
  **GAME** , **POV_PLAYER** , **END_GAME** , **TILE** ,
  **HELP_TEXT** , **HELP_MENU** , **END** }

  *Hud enum for the different scenes.*

#### Protected Attributes inherited from Gui::AHud

- TypeScene **_typeScene**

  *Type of the scene.*
- std::shared_ptr< GameData > **_gameData**

  *GameData class.*
- std::shared_ptr< Player > **_player**

  *Player to display hud.*
- std::shared_ptr< Tile > **_tile**

  *Tile to display hud.*

### 5.17.1 Constructor & Destructor Documentation

#### 5.17.1.1 HudHelp()

```
Gui::HudHelp::HudHelp (
            std::shared_ptr< GameData > gameData )
```

Construct a new Hud Help object.

**Parameters**

| | |
|---|---|
| *gameData* | GameData class. |

#### 5.17.1.2 ~HudHelp()

```
Gui::HudHelp::~HudHelp ( )  [default]
```

Destroy the Hud Help object.

### 5.17.2 Member Function Documentation

#### 5.17.2.1 display()

```
void Gui::HudHelp::display ( )  [virtual]
```

Display Help Hud.

Implements Gui::AHud.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Hud/HudHelp.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Hud/HudHelp.cpp

## 5.18 Gui::HudPlayer Class Reference

Inheritance diagram for Gui::HudPlayer:

Collaboration diagram for Gui::HudPlayer:

## Public Member Functions

- HudPlayer (std::shared_ptr< GameData > gameData)

  *Construct a new Hud Player object.*
- ~HudPlayer ()=default

  *Destroy the Hud Player object.*
- void display ()

  *Display Player Hud.*

### Public Member Functions inherited from Gui::AHud

- ~AHud ()=default

  *Destroy the AHud object.*
- virtual void display ()=0

  *Display Hud.*
- void setPlayer (std::shared_ptr< Player > player)

  *Set the Player object.*
- void setTile (std::shared_ptr< Tile > tile)

  *Set the Tile object.*
- TypeScene getType () const

  *Get the Type object.*
- void setType (TypeScene type)

  *Set the Type object.*

### Public Member Functions inherited from Gui::IHud

- virtual ~IHud ()=default

  *Destroy the IHud object.*
- virtual void display ()=0

  *Display the Hud.*
- virtual void setPlayer (std::shared_ptr< Player > player)=0

  *Set the Player object.*
- virtual void setTile (std::shared_ptr< Tile > tile)=0

  *Set the Tile object.*
- virtual TypeScene getType () const =0

  *Get the Type object.*
- virtual void setType (TypeScene type)=0

  *Set the Type object.*

## Additional Inherited Members

### Public Types inherited from Gui::IHud

- enum TypeScene {
  **GAME** , **POV_PLAYER** , **END_GAME** , **TILE** ,
  **HELP_TEXT** , **HELP_MENU** , **END** }

  *Hud enum for the different scenes.*

**Protected Attributes inherited from Gui::AHud**

- TypeScene **_typeScene**

  *Type of the scene.*
- std::shared_ptr< GameData > **_gameData**

  *GameData class.*
- std::shared_ptr< Player > **_player**

  *Player to display hud.*
- std::shared_ptr< Tile > **_tile**

  *Tile to display hud.*

## 5.18.1 Constructor & Destructor Documentation

### 5.18.1.1 HudPlayer()

```
Gui::HudPlayer::HudPlayer (
            std::shared_ptr< GameData > gameData )
```

Construct a new Hud Player object.

**Parameters**

| | |
|---|---|
| *gameData* | GameData class. |

### 5.18.1.2 ~HudPlayer()

```
Gui::HudPlayer::~HudPlayer ( )  [default]
```

Destroy the Hud Player object.

## 5.18.2 Member Function Documentation

### 5.18.2.1 display()

```
void Gui::HudPlayer::display ( )  [virtual]
```

Display Player Hud.

Implements Gui::AHud.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Hud/HudPlayer.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Hud/HudPlayer.cpp

## 5.19 Gui::HudTile Class Reference

Inheritance diagram for Gui::HudTile:

Collaboration diagram for Gui::HudTile:

### Public Member Functions

- HudTile (std::shared_ptr< GameData > gameData)
  *Construct a new Hud Tile object.*
- ~HudTile ()=default
  *Destroy the Hud Tile object.*
- void display ()
  *Display Tile Hud.*
- void displayNbPlayers ()
  *Display number of players.*
- void displayNbEggs ()
  *Display number of eggs.*

#### Public Member Functions inherited from Gui::AHud

- ~AHud ()=default
  *Destroy the AHud object.*
- virtual void display ()=0
  *Display Hud.*
- void setPlayer (std::shared_ptr< Player > player)
  *Set the Player object.*
- void setTile (std::shared_ptr< Tile > tile)
  *Set the Tile object.*
- TypeScene getType () const
  *Get the Type object.*
- void setType (TypeScene type)
  *Set the Type object.*

#### Public Member Functions inherited from Gui::IHud

- virtual ~IHud ()=default
  *Destroy the IHud object.*
- virtual void display ()=0
  *Display the Hud.*
- virtual void setPlayer (std::shared_ptr< Player > player)=0
  *Set the Player object.*
- virtual void setTile (std::shared_ptr< Tile > tile)=0
  *Set the Tile object.*
- virtual TypeScene getType () const =0
  *Get the Type object.*
- virtual void setType (TypeScene type)=0
  *Set the Type object.*

## Additional Inherited Members

### Public Types inherited from Gui::IHud

- enum TypeScene {
  **GAME** , **POV_PLAYER** , **END_GAME** , **TILE** ,
  **HELP_TEXT** , **HELP_MENU** , **END** }
  *Hud enum for the different scenes.*

### Protected Attributes inherited from Gui::AHud

- TypeScene **_typeScene**
  *Type of the scene.*
- std::shared_ptr< GameData > **_gameData**
  *GameData class.*
- std::shared_ptr< Player > **_player**
  *Player to display hud.*
- std::shared_ptr< Tile > **_tile**
  *Tile to display hud.*

## 5.19.1 Constructor & Destructor Documentation

### 5.19.1.1 HudTile()

```
Gui::HudTile::HudTile (
            std::shared_ptr< GameData > gameData )
```

Construct a new Hud Tile object.

**Parameters**

| | |
|---|---|
| *gameData* | GameData class. |

### 5.19.1.2 ~HudTile()

```
Gui::HudTile::~HudTile ( )  [default]
```

Destroy the Hud Tile object.

## 5.19.2 Member Function Documentation

#### 5.19.2.1 display()

```
void Gui::HudTile::display ( )  [virtual]
```

Display Tile Hud.

Implements Gui::AHud.

#### 5.19.2.2 displayNbEggs()

```
void Gui::HudTile::displayNbEggs ( )
```

Display number of eggs.

#### 5.19.2.3 displayNbPlayers()

```
void Gui::HudTile::displayNbPlayers ( )
```

Display number of players.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Hud/HudTile.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Hud/HudTile.cpp

## 5.20 Gui::Errors::IError Class Reference

Inheritance diagram for Gui::Errors::IError:

Collaboration diagram for Gui::Errors::IError:

### Public Member Functions

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

  *Destructor for IError.*
- virtual const char * what () const noexcept=0

  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

### 5.20.1 Member Function Documentation

#### 5.20.1.1 what()

```
virtual const char * Gui::Errors::IError::what ( ) const  [pure virtual], [noexcept]
```

Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.

**Returns**

const char* A C-style string describing the error.

Implemented in Gui::Errors::AError.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/IError.hpp

## 5.21 Gui::IEvent Class Reference

Inheritance diagram for Gui::IEvent:

### Public Member Functions

- IEvent ()=default
  *Construct a new IEvent object.*
- virtual ~IEvent ()=default
  *Destroy the IEvent object.*
- virtual void listen ()=0
  *Listen the user's events.*
- virtual void setRender (std::shared_ptr< Render > render)=0
  *Set the Render object.*
- virtual void setGameData (std::shared_ptr< GameData > gameData)=0
  *Set the GameData object.*

### 5.21.1 Constructor & Destructor Documentation

#### 5.21.1.1 IEvent()

```
Gui::IEvent::IEvent ( )  [default]
```

Construct a new IEvent object.

#### 5.21.1.2 ~IEvent()

```
virtual Gui::IEvent::~IEvent ( )  [virtual], [default]
```

Destroy the IEvent object.

### 5.21.2 Member Function Documentation

#### 5.21.2.1 listen()

```
virtual void Gui::IEvent::listen ( )  [pure virtual]
```

Listen the user's events.

Implemented in Gui::Event, and Gui::AEvent.

#### 5.21.2.2 setGameData()

```
virtual void Gui::IEvent::setGameData (
            std::shared_ptr< GameData > gameData )  [pure virtual]
```

Set the GameData object.

**Parameters**

| | |
|---|---|
| *gameData* | GameData class. |

Implemented in Gui::AEvent.

#### 5.21.2.3 setRender()

```
virtual void Gui::IEvent::setRender (
            std::shared_ptr< Render > render )  [pure virtual]
```

Set the Render object.

**Parameters**

| | |
|---|---|
| *render* | Render class. |

Implemented in Gui::AEvent.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Event/IEvent.hpp

## 5.22 Gui::IGUIUpdater Class Reference

Inheritance diagram for Gui::IGUIUpdater:

### Public Member Functions

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

  *Destroy the IGUIUpdater object.*
- virtual void update (const std::string &command, const std::vector< std::string > &data)=0

  *Update the GUI GameData.*

### 5.22.1 Member Function Documentation

#### 5.22.1.1 update()

```
virtual void Gui::IGUIUpdater::update (
            const std::string & command,
            const std::vector< std::string > & data )  [pure virtual]
```

Update the GUI GameData.

Implemented in Gui::GUIUpdater, and Gui::AGUIUpdater.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GUIUpdater/IGUIUpdater.hpp

## 5.23 Gui::IHud Class Reference

Inheritance diagram for Gui::IHud:

### Public Types

- enum TypeScene {
  **GAME** , **POV_PLAYER** , **END_GAME** , **TILE** ,
  **HELP_TEXT** , **HELP_MENU** , **END** }

  *Hud enum for the different scenes.*

## Public Member Functions

- virtual ~IHud ()=default

  *Destroy the IHud object.*
- virtual void display ()=0

  *Display the Hud.*
- virtual void setPlayer (std::shared_ptr< Player > player)=0

  *Set the Player object.*
- virtual void setTile (std::shared_ptr< Tile > tile)=0

  *Set the Tile object.*
- virtual TypeScene getType () const =0

  *Get the Type object.*
- virtual void setType (TypeScene type)=0

  *Set the Type object.*

### 5.23.1 Member Enumeration Documentation

#### 5.23.1.1 TypeScene

```
enum Gui::IHud::TypeScene
```

Hud enum for the different scenes.

### 5.23.2 Constructor & Destructor Documentation

#### 5.23.2.1 ~IHud()

```
virtual Gui::IHud::~IHud ( )  [virtual], [default]
```

Destroy the IHud object.

### 5.23.3 Member Function Documentation

#### 5.23.3.1 display()

```
virtual void Gui::IHud::display ( )  [pure virtual]
```

Display the Hud.

Implemented in Gui::HudEnd, Gui::HudGame, Gui::HudHelp, Gui::HudPlayer, Gui::HudTile, and Gui::AHud.

#### 5.23.3.2 getType()

```
virtual TypeScene Gui::IHud::getType ( ) const  [pure virtual]
```

Get the Type object.

**Returns**

TypeScene - Type of the scene.

Implemented in Gui::AHud.

#### 5.23.3.3 setPlayer()

```
virtual void Gui::IHud::setPlayer (
            std::shared_ptr< Player > player )  [pure virtual]
```

Set the Player object.

**Parameters**

| | |
|---|---|
| *player* | Player to display infos. |

Implemented in Gui::AHud.

#### 5.23.3.4 setTile()

```
virtual void Gui::IHud::setTile (
            std::shared_ptr< Tile > tile )  [pure virtual]
```

Set the Tile object.

**Parameters**

| | |
|---|---|
| *tile* | Tile to display infos. |

Implemented in Gui::AHud.

#### 5.23.3.5 setType()

```
virtual void Gui::IHud::setType (
            TypeScene type )  [pure virtual]
```

Set the Type object.

**Parameters**

| | |
|---|---|
| *type* | Type of the scene. |

Implemented in Gui::AHud.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Hud/IHud.hpp

## 5.24 Gui::INetwork Class Reference

Inheritance diagram for Gui::INetwork:

### Public Types

- enum **BufferState** { **NONE** , **READY** , **SERVER_ERROR** }

### Public Member Functions

- virtual ~INetwork ()=default

  *Destroy the INetwork object.*
- virtual void setPort (int port)=0

  *Set the port object.*
- virtual void setHostName (const std::string &hostName)=0

  *Set the host name object.*
- virtual int getPort () const =0

  *Get the host name object.*
- virtual std::string getHostName () const =0

  *Get the host name object.*
- virtual void connectToServer ()=0

  *Connect to the server.*
- virtual BufferState listenServer ()=0

  *Listen to the server.*
- virtual void sendMessageServer (const std::string &message)=0

  *Send a message to the server.*
- virtual std::string getBuffer ()=0

  *Get the Buffer object. Be careful, this method will delete the current buffer.*

### 5.24.1 Constructor & Destructor Documentation

#### 5.24.1.1 ~INetwork()

```
virtual Gui::INetwork::~INetwork ( )  [virtual], [default]
```

Destroy the INetwork object.

### 5.24.2 Member Function Documentation

#### 5.24.2.1 connectToServer()

```
virtual void Gui::INetwork::connectToServer ( )  [pure virtual]
```

Connect to the server.

**Exceptions**

| | |
|---|---|
| *Error::NetworkError* | If the connection failed. |

Implemented in Gui::Network, and Gui::ANetwork.

#### 5.24.2.2 getBuffer()

```
virtual std::string Gui::INetwork::getBuffer ( )  [pure virtual]
```

Get the Buffer object. Be careful, this method will delete the current buffer.

**Returns**

std::string - Buffer message.

Implemented in Gui::ANetwork.

#### 5.24.2.3 getHostName()

```
virtual std::string Gui::INetwork::getHostName ( ) const  [pure virtual]
```

Get the host name object.

**Returns**

std::string Host of the server.

Implemented in Gui::ANetwork.

#### 5.24.2.4 getPort()

```
virtual int Gui::INetwork::getPort ( ) const  [pure virtual]
```

Get the host name object.

**Returns**

std::string Host of the server.

Implemented in Gui::ANetwork.

#### 5.24.2.5 listenServer()

```
virtual BufferState Gui::INetwork::listenServer ( )  [pure virtual]
```

Listen to the server.

**Returns**

BufferState - Buffer state.

Implemented in Gui::Network, and Gui::ANetwork.

#### 5.24.2.6 sendMessageServer()

```
virtual void Gui::INetwork::sendMessageServer (
            const std::string & message )  [pure virtual]
```

Send a message to the server.

**Parameters**

| | |
|---|---|
| *message* | Message to send. |

Implemented in Gui::Network, and Gui::ANetwork.

#### 5.24.2.7 setHostName()

```
virtual void Gui::INetwork::setHostName (
            const std::string & hostName )  [pure virtual]
```

Set the host name object.

**Parameters**

| | |
|---|---|
| *hostName* | Host of the server. |

Implemented in Gui::ANetwork.

#### 5.24.2.8 setPort()

```
virtual void Gui::INetwork::setPort (
            int port )  [pure virtual]
```

Set the port object.

**Parameters**

| | |
|---|---|
| *port* | Port of the server. |

**Exceptions**

| | |
|---|---|
| *NetworkException* | If the port is not in range 1 to 65535. |

Implemented in Gui::ANetwork.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Network/INetwork.hpp

## 5.25 Gui::Inventory Class Reference

### Public Types

- using **Ressources** = size_t[RessourcesNumber]

### Public Member Functions

- Inventory ()

  *Construct a new Inventory object.*
- Inventory (std::size_t food, std::size_t linemate, std::size_t deraumere, std::size_t sibur, std::size_t mendiane, std::size_t phiras, std::size_t thystame)

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

  *Destroy the Inventory object.*
- void setFood (std::size_t food)

  *Set the Food object.*

- void setLinemate (std::size_t linemate)

  *Set the Linemate object.*
- void setDeraumere (std::size_t deraumere)

  *Set the Deraumere object.*
- void setSibur (std::size_t sibur)

  *Set the Sibur object.*
- void setMendiane (std::size_t mendiane)

  *Set the Mendiane object.*
- void setPhiras (std::size_t phiras)

  *Set the Phiras object.*
- void setThystame (std::size_t thytsame)

  *Set the Thystame object.*
- void setRessources (Ressources ressources)

  *Set the Ressources object.*
- std::size_t getFood (void)

  *Get the Food object.*
- std::size_t getLinemate (void)

  *Get the Linemate object.*
- std::size_t getDeraumere (void)

  *Get the Deraumere object.*
- std::size_t getSibur (void)

  *Get the Sibur object.*
- std::size_t getMendiane (void)

  *Get the Mendiane object.*
- std::size_t getPhiras (void)

  *Get the Phiras object.*
- std::size_t getThystame (void)

  *Get the Thystame object.*
- Ressources & getRessources (void)

  *Get the Ressources object.*
- void addResource (std::size_t resource, std::size_t quantity)

  *Add resources to inventory.*
- void removeResource (std::size_t resource, std::size_t quantity)

  *Remove resources to inventory.*

### 5.25.1 Constructor & Destructor Documentation

#### 5.25.1.1 Inventory() [1/2]

```
Gui::Inventory::Inventory ( )
```

Construct a new Inventory object.

#### 5.25.1.2 Inventory() [2/2]

```
Gui::Inventory::Inventory (
            std::size_t food,
            std::size_t linemate,
            std::size_t deraumere,
            std::size_t sibur,
            std::size_t mendiane,
            std::size_t phiras,
            std::size_t thystame )
```

Construct a new Inventory object.

**Parameters**

| | |
|---|---|
| *food* | Food to set. |
| *linemate* | Linemate to set. |
| *deraumere* | Deraumere to set. |
| *sibur* | Sibur to set. |
| *mendiane* | Mendiane to set. |
| *phiras* | Phiras to set. |
| *thystame* | Thystame to set. |

#### 5.25.1.3 ~Inventory()

```
Gui::Inventory::~Inventory ( )  [default]
```

Destroy the Inventory object.

### 5.25.2 Member Function Documentation

#### 5.25.2.1 addResource()

```
void Gui::Inventory::addResource (
            std::size_t resource,
            std::size_t quantity )
```

Add resources to inventory.

**Parameters**

| | |
|---|---|
| *resource* | Index resource's. resource 0 (food) resource 1 (linemate) resource 2 (deraumere) resource 3 (sibur) resource 4 (mendiane) resource 5 (phiras) resource 6 (thystame) |
| *quantity* | Quantity to add. |

#### 5.25.2.2 getDeraumere()

```
std::size_t Gui::Inventory::getDeraumere (
            void  )
```

Get the Deraumere object.

**Returns**

std::size_t - deraumere

#### 5.25.2.3 getFood()

```
std::size_t Gui::Inventory::getFood (
            void  )
```

Get the Food object.

**Returns**

std::size_t - food

#### 5.25.2.4 getLinemate()

```
std::size_t Gui::Inventory::getLinemate (
            void  )
```

Get the Linemate object.

**Returns**

std::size_t - linemate

#### 5.25.2.5 getMendiane()

```
std::size_t Gui::Inventory::getMendiane (
            void  )
```

Get the Mendiane object.

**Returns**

std::size_t - mendiane

#### 5.25.2.6 getPhiras()

```
std::size_t Gui::Inventory::getPhiras (
            void  )
```

Get the Phiras object.

**Returns**

std::size_t - phiras

#### 5.25.2.7 getRessources()

```
Gui::Inventory::Ressources & Gui::Inventory::getRessources (
            void  )
```

Get the Ressources object.

**Returns**

Ressources - ressources

#### 5.25.2.8 getSibur()

```
std::size_t Gui::Inventory::getSibur (
            void  )
```

Get the Sibur object.

**Returns**

std::size_t - sibur

#### 5.25.2.9 getThystame()

```
std::size_t Gui::Inventory::getThystame (
            void  )
```

Get the Thystame object.

**Returns**

std::size_t - thystame

#### 5.25.2.10 removeResource()

```
void Gui::Inventory::removeResource (
            std::size_t resource,
            std::size_t quantity )
```

Remove resources to inventory.

**Parameters**

| | |
|---|---|
| *resource* | Index resource's. resource 0 (food) resource 1 (linemate) resource 2 (deraumere) resource 3 (sibur) resource 4 (mendiane) resource 5 (phiras) resource 6 (thystame) |
| *quantity* | Quantity to remove. |

#### 5.25.2.11 setDeraumere()

```
void Gui::Inventory::setDeraumere (
            std::size_t deraumere )
```

Set the Deraumere object.

**Parameters**

| | |
|---|---|
| *deraumere* | Deraumere to set. |

#### 5.25.2.12 setFood()

```
void Gui::Inventory::setFood (
            std::size_t food )
```

Set the Food object.

**Parameters**

| | |
|---|---|
| *food* | Food to set. |

#### 5.25.2.13 setLinemate()

```
void Gui::Inventory::setLinemate (
            std::size_t linemate )
```

Set the Linemate object.

**Parameters**

| | |
|---|---|
| *linemate* | Linemate to set. |

#### 5.25.2.14 setMendiane()

```
void Gui::Inventory::setMendiane (
            std::size_t mendiane )
```

Set the Mendiane object.

**Parameters**

| | |
|---|---|
| *mendiane* | Mendiane to set. |

#### 5.25.2.15 setPhiras()

```
void Gui::Inventory::setPhiras (
            std::size_t phiras )
```

Set the Phiras object.

**Parameters**

| | |
|---|---|
| *phiras* | Phiras to set. |

#### 5.25.2.16 setRessources()

```
void Gui::Inventory::setRessources (
            Ressources ressources )
```

Set the Ressources object.

**Parameters**

| | |
|---|---|
| *ressources* | Ressources to set. |

#### 5.25.2.17 setSibur()

```
void Gui::Inventory::setSibur (
            std::size_t sibur )
```

Set the Sibur object.

**Parameters**

| | |
|---|---|
| *sibur* | Sibur to set. |

#### 5.25.2.18 setThystame()

```
void Gui::Inventory::setThystame (
            std::size_t thytsame )
```

Set the Thystame object.

**Parameters**

| | |
|---|---|
| *thystame* | Thystame to set. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GameDatas/Inventory.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GameDatas/Inventory.cpp

## 5.26 Gui::IServerParser Class Reference

Inheritance diagram for Gui::IServerParser:

### Public Member Functions

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

  *Destroy the IServerParser object.*
- virtual std::vector< std::string > parse (const std::string &command)=0

  *Parse the command server.*

### 5.26.1 Member Function Documentation

#### 5.26.1.1 parse()

```
virtual std::vector< std::string > Gui::IServerParser::parse (
            const std::string & command )  [pure virtual]
```

Parse the command server.

**Parameters**

| | |
|---|---|
| *command* | Command to parse. |

**Returns**

std::vector<std::string> - arguments parsed.

Implemented in Gui::ServerParser.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Parsing/IServerParser.hpp

## 5.27 Gui::Network Class Reference

Inheritance diagram for Gui::Network:

Collaboration diagram for Gui::Network:

### Public Member Functions

- Network (int port, const std::string &hostName)
  *Construct a new Network object.*
- ~Network ()
  *Destroy the Network object.*
- void connectToServer ()
  *Connect the Gui network with the server.*
- BufferState listenServer ()
  *Listen the server and return it message.*
- void sendMessageServer (const std::string &message)
  *Send a message to the Server.*

#### Public Member Functions inherited from Gui::ANetwork

- ANetwork (int port, const std::string &hostName)
  *Construct a new ANetwork object.*
- ~ANetwork ()=default
  *Destroy the ANetwork object.*
- void setPort (int port) final
  *Set the port object.*
- void setHostName (const std::string &hostName) final
  *Set the host name object.*
- int getPort () const final
  *Get the host name object.*
- std::string getHostName () const final
  *Get the host name object.*
- virtual void connectToServer ()=0
  *Connect to the server.*
- virtual BufferState listenServer ()=0
  *Listen the server and return it message.*
- virtual void sendMessageServer (const std::string &message)=0
  *Send a message to the Server.*
- std::string getBuffer ()
  *Get the Buffer object. Be careful, this method will delete the current buffer.*

#### Public Member Functions inherited from Gui::INetwork

- virtual ~INetwork ()=default

  *Destroy the INetwork object.*
- virtual void setPort (int port)=0

  *Set the port object.*
- virtual void setHostName (const std::string &hostName)=0

  *Set the host name object.*
- virtual int getPort () const =0

  *Get the host name object.*
- virtual std::string getHostName () const =0

  *Get the host name object.*
- virtual void connectToServer ()=0

  *Connect to the server.*
- virtual BufferState listenServer ()=0

  *Listen to the server.*
- virtual void sendMessageServer (const std::string &message)=0

  *Send a message to the server.*
- virtual std::string getBuffer ()=0

  *Get the Buffer object. Be careful, this method will delete the current buffer.*

## Additional Inherited Members

#### Public Types inherited from Gui::INetwork

- enum **BufferState** { **NONE** , **READY** , **SERVER_ERROR** }

#### Protected Attributes inherited from Gui::ANetwork

- int **_port**

  *Port of the server.*
- std::string **_hostName**

  *Host name of the server.*
- std::string **_buffer**

  *Buffer to receive server message.*

## 5.27.1 Constructor & Destructor Documentation

### 5.27.1.1 Network()

```
Gui::Network::Network (
            int port,
            const std::string & hostName )
```

Construct a new Network object.

**Parameters**

| | |
|---|---|
| *port* | Port of the server. |
| *hostName* | Host of the server. |

#### 5.27.1.2 ~Network()

```
Gui::Network::~Network ( )
```

Destroy the Network object.

### 5.27.2 Member Function Documentation

#### 5.27.2.1 connectToServer()

```
void Gui::Network::connectToServer ( )  [virtual]
```

Connect the Gui network with the server.

**Exceptions**

| | |
|---|---|
| *NetworkException* | If the connection failed. |

Implements Gui::ANetwork.

#### 5.27.2.2 listenServer()

```
Gui::Network::BufferState Gui::Network::listenServer ( )  [virtual]
```

Listen the server and return it message.

**Returns**

BufferState - Buffer state.

Implements Gui::ANetwork.

#### 5.27.2.3 sendMessageServer()

```
void Gui::Network::sendMessageServer (
            const std::string & message )  [virtual]
```

Send a message to the Server.

**Parameters**

| | |
|---|---|
| *message* | Message to send to the server. |

Implements Gui::ANetwork.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Network/Network.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Network/Network.cpp

## 5.28 Gui::Errors::NetworkException Class Reference

Error class for network errors.

`#include <Error.hpp>`

Inheritance diagram for Gui::Errors::NetworkException:

Collaboration diagram for Gui::Errors::NetworkException:

### Public Member Functions

- NetworkException (std::string message)
  *Constructor for NetworkException.*

#### Public Member Functions inherited from Gui::Errors::AError

- ~**AError** () override=default
  *Destructor.*
- const char * what () const noexcept override
  *Returns the error message.*

#### Public Member Functions inherited from Gui::Errors::IError

- virtual ~**IError** ()=default
  *Destructor for IError.*
- virtual const char * what () const noexcept=0
  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

### Additional Inherited Members

#### Protected Attributes inherited from Gui::Errors::AError

- std::string **_message**
  *The error message.*

### 5.28.1 Detailed Description

Error class for network errors.

### 5.28.2 Constructor & Destructor Documentation

#### 5.28.2.1 NetworkException()

```
Gui::Errors::NetworkException::NetworkException (
            std::string message )
```

Constructor for NetworkException.

**Parameters**

| | |
|---|---|
| *message* | The error message. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/Error.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Error/Error.cpp

## 5.29 Gui::ParseCommandLine Class Reference

### Public Member Functions

- ParseCommandLine (int argc, char **argv)
  *Construct a new Parse Command Line object.*
- ~ParseCommandLine ()=default
  *Destroy the Parse Command Line object.*
- void parseFlags (int argc, char **argv)
  *Parse flags in command line.*
- int getPort (void)
  *Get the port object.*
- std::string getHostName (void)
  *Get the hostName object.*

### 5.29.1 Constructor & Destructor Documentation

#### 5.29.1.1 ParseCommandLine()

```
Gui::ParseCommandLine::ParseCommandLine (
            int argc,
            char ** argv )
```

Construct a new Parse Command Line object.

**Parameters**

| | |
|---|---|
| *argc* | Number of arguments in command line. |
| *argv* | Array with command line arguments. |

#### 5.29.1.2 ~ParseCommandLine()

```
Gui::ParseCommandLine::~ParseCommandLine ( )  [default]
```

Destroy the Parse Command Line object.

### 5.29.2 Member Function Documentation

#### 5.29.2.1 getHostName()

```
std::string Gui::ParseCommandLine::getHostName (
            void  )
```

Get the hostName object.

**Returns**

std::string - hostName

#### 5.29.2.2 getPort()

```
int Gui::ParseCommandLine::getPort (
            void  )
```

Get the port object.

**Returns**

int - port

#### 5.29.2.3 parseFlags()

```
void Gui::ParseCommandLine::parseFlags (
            int argc,
            char ** argv )
```

Parse flags in command line.

**Parameters**

| | |
|---|---|
| *argc* | Number of arguments in command line. |
| *argv* | Array with command line arguments. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Parsing/ParseCommandLine.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Parsing/ParseCommandLine.cpp

## 5.30 Gui::Errors::ParseCommandLineException Class Reference

Error class for parseCommandLine errors.

`#include <Error.hpp>`

Inheritance diagram for Gui::Errors::ParseCommandLineException:

Collaboration diagram for Gui::Errors::ParseCommandLineException:

### Public Member Functions

- ParseCommandLineException (std::string message)

  *Constructor for ParseCommandLineException.*

#### Public Member Functions inherited from Gui::Errors::AError

- ∼**AError** () override=default

  *Destructor.*
- const char ∗ what () const noexcept override

  *Returns the error message.*

#### Public Member Functions inherited from Gui::Errors::IError

- virtual ∼**IError** ()=default

  *Destructor for IError.*
- virtual const char ∗ what () const noexcept=0

  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

### Additional Inherited Members

#### Protected Attributes inherited from Gui::Errors::AError

- std::string **_message**

  *The error message.*

### 5.30.1 Detailed Description

Error class for parseCommandLine errors.

### 5.30.2 Constructor & Destructor Documentation

#### 5.30.2.1 ParseCommandLineException()

```
Gui::Errors::ParseCommandLineException::ParseCommandLineException (
            std::string message )
```

Constructor for ParseCommandLineException.

**Parameters**

| | |
|---|---|
| *message* | The error message. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/Error.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Error/Error.cpp

## 5.31 Gui::Player Class Reference

Collaboration diagram for Gui::Player:

### Public Types

- enum **PlayerState** {
  **IDLE** = 2 , **BORN** = 8 , **BROADCAST** = 12 , **EJECT** = 5 ,
  **BEING_EJECTED** = 15 , **EJECTED** = 7 , **WALK** = 6 , **INCANTATION** = 0 ,
  **LAY_EGG** = 7 , **DROP** = 9 , **COLLECT** = 9 , **DEAD** = 1 }

### Public Member Functions

- Player (std::size_t id, const std::string &team, std::pair< std::size_t, std::size_t > position, std::size_t orientation, std::size_t level=1)

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

  *Destroy the Player object.*
- void setPosition (std::pair< std::size_t, std::size_t > position)

  *Set the Position object.*
- void setPosition3D (Vector3 position3D)

*Set the Position3D object.*

- void setId (std::size_t id)

  *Set the Id object.*
- void setLevel (std::size_t level)

  *Set the Level object.*
- void setOrientation (std::size_t orientation)

  *Set the Orientation object.*
- void setTeam (const std::string &team)

  *Set the Team object.*
- std::pair< std::size_t, std::size_t > getPosition (void) const

  *Get the Position object.*
- Vector3 getPosition3D (void) const

  *Get the Position3D object.*
- std::size_t getId (void) const

  *Get the Id object.*
- std::size_t getLevel (void) const

  *Get the Level object.*
- std::size_t getOrientation (void) const

  *Get the Orientation object.*
- std::string getTeam (void) const

  *Get the Team object.*
- void setState (PlayerState state)

  *Set the State object.*
- PlayerState getState (void) const

  *Get the State object.*
- void setBroadcast (const std::string &broadcast)

  *Set the Broadcast object.*
- std::string getBroadcast () const

  *Get the Broadcast object.*
- float getRotationFromOrientation () const

  *Get the Vector From Orientation object.*
- Vector3 getCenterPosition ()

  *Get the Center Position object.*
- void setCurrentFrame (int currentFrame)

  *Set the Current Frame object.*
- int getCurrentFrame () const

  *Get the Current Frame object.*
- void restartAnimationTimeEllapsed ()

  *Restart the timer animation.*
- clock_t getAnimationTimeEllapsed () const

  *Get the Animation Time Ellapsed object.*

## Public Attributes

- Inventory inventory

  *Inventory of the player.*

### 5.31.1 Constructor & Destructor Documentation

#### 5.31.1.1 Player()

```
Gui::Player::Player (
            std::size_t id,
            const std::string & team,
            std::pair< std::size_t, std::size_t > position,
            std::size_t orientation,
            std::size_t level = 1 )
```

Construct a new Player object.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |
| *team* | Team name of the player. |
| *position* | Position of the Player. |

#### 5.31.1.2 ~Player()

```
Gui::Player::~Player ( )  [default]
```

Destroy the Player object.

### 5.31.2 Member Function Documentation

#### 5.31.2.1 getAnimationTimeEllapsed()

```
clock_t Gui::Player::getAnimationTimeEllapsed ( ) const
```

Get the Animation Time Ellapsed object.

**Returns**

clock_t - Animation time ellapsed.

#### 5.31.2.2 getBroadcast()

```
std::string Gui::Player::getBroadcast ( ) const
```

Get the Broadcast object.

**Returns**

std::string - Broadcast message.

#### 5.31.2.3 getCenterPosition()

```
Vector3 Gui::Player::getCenterPosition ( )
```

Get the Center Position object.

**Returns**

Vector3 - Center position.

#### 5.31.2.4 getCurrentFrame()

```
int Gui::Player::getCurrentFrame ( ) const
```

Get the Current Frame object.

**Returns**

int - Current frame.

#### 5.31.2.5 getId()

```
std::size_t Gui::Player::getId (
            void  ) const
```

Get the Id object.

**Returns**

std::size_t - id

#### 5.31.2.6 getLevel()

```
std::size_t Gui::Player::getLevel (
            void  ) const
```

Get the Level object.

**Returns**

std::size_t - level

#### 5.31.2.7 getOrientation()

```
std::size_t Gui::Player::getOrientation (
            void  ) const
```

Get the Orientation object.

**Returns**

std::size_t - orientation

#### 5.31.2.8 getPosition()

```
std::pair< std::size_t, std::size_t > Gui::Player::getPosition (
            void  ) const
```

Get the Position object.

**Returns**

std::pair<std::size_t, std::size_t> - position

#### 5.31.2.9 getPosition3D()

```
Vector3 Gui::Player::getPosition3D (
            void  ) const
```

Get the Position3D object.

**Returns**

Vector3 - position3D

#### 5.31.2.10 getRotationFromOrientation()

```
float Gui::Player::getRotationFromOrientation ( ) const
```

Get the Vector From Orientation object.

**Returns**

Vector3 - Vector3 from orientation.

#### 5.31.2.11 getState()

```
Gui::Player::PlayerState Gui::Player::getState (
            void  ) const
```

Get the State object.

**Returns**

PlayerState - Player state.

#### 5.31.2.12 getTeam()

```
std::string Gui::Player::getTeam (
            void  ) const
```

Get the Team object.

**Returns**

std::string - team name

#### 5.31.2.13 restartAnimationTimeEllapsed()

```
void Gui::Player::restartAnimationTimeEllapsed ( )
```

Restart the timer animation.

#### 5.31.2.14 setBroadcast()

```
void Gui::Player::setBroadcast (
            const std::string & broadcast )
```

Set the Broadcast object.

**Parameters**

| | |
|---|---|
| *broadcast* | New broadcast message. |

#### 5.31.2.15 setCurrentFrame()

```
void Gui::Player::setCurrentFrame (
            int currentFrame )
```

Set the Current Frame object.

**Parameters**

| | |
|---|---|
| *currentFrame* | Current Frame to set. |

#### 5.31.2.16 setId()

```
void Gui::Player::setId (
            std::size_t id )
```

Set the Id object.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |

#### 5.31.2.17 setLevel()

```
void Gui::Player::setLevel (
            std::size_t level )
```

Set the Level object.

**Parameters**

| | |
|---|---|
| *level* | Level of the player. |

#### 5.31.2.18 setOrientation()

```
void Gui::Player::setOrientation (
            std::size_t orientation )
```

Set the Orientation object.

**Parameters**

| | |
|---|---|
| *orientation* | Orientation of the player. |

#### 5.31.2.19 setPosition()

```
void Gui::Player::setPosition (
            std::pair< std::size_t, std::size_t > position )
```

Set the Position object.

**Parameters**

| | |
|---|---|
| *position* | Position of the player. |

#### 5.31.2.20 setPosition3D()

```
void Gui::Player::setPosition3D (
            Vector3 position3D )
```

Set the Position3D object.

**Parameters**

| | |
|---|---|
| *position3D* | Position of the player. |

#### 5.31.2.21 setState()

```
void Gui::Player::setState (
            PlayerState state )
```

Set the State object.

**Parameters**

| | |
|---|---|
| *state* | New player state. |

#### 5.31.2.22 setTeam()

```
void Gui::Player::setTeam (
            const std::string & team )
```

Set the Team object.

**Parameters**

| | |
|---|---|
| *team* | Team name of the player. |

### 5.31.3 Member Data Documentation

#### 5.31.3.1 inventory

```
Inventory Gui::Player::inventory
```

Inventory of the player.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GameDatas/Player.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GameDatas/Player.cpp

## 5.32 Gui::Render Class Reference

### Public Member Functions

- Render (std::shared_ptr< GameData > gameData)
  *Construct a new Render object.*
- ~Render ()
  *Destroy the Render object.*
- bool isOpen ()
  *Check if the window is open.*
- void draw ()
  *Draw the scene.*
- std::shared_ptr< Camera > getCamera ()
  *Get the Camera object.*
- void setIsDebug (bool isDebug)
  *Set the Is Debug object.*
- bool getIsDebug (void)
  *Get the Is Debug object.*
- void setCameraType (Gui::UserCamera::CameraType type)
  *Set the Type object.*
- Gui::UserCamera::CameraType getCameraType () const

*Get the Type object.*

- void setCameraPlayerPov (std::size_t id)

  *Set the Camera player pov id.*
- std::size_t getCameraPlayerPov () const

  *Get the Camera player pov id.*
- void setCameraTile (std::pair< std::size_t, std::size_t > pos)

  *Set the Camera Tile object.*
- std::pair< std::size_t, std::size_t > getCameraTile () const

  *Get the Camera Tile object.*
- Model getTileModel () const

  *Get the Tile model.*
- void setRenderDistance (size_t renderDistance)

  *Set the Render Distance value.*
- size_t getRenderDistance () const

  *Get the Render Distance value.*
- bool isCameraInPlayerPov () const

  *Check if the camera is in player pov.*
- void changePlayerPOV (size_t playerId)

  *Change the player point of view.*
- void setPlayerPov (size_t playerId)

  *Sets the Pov of the player.*
- void changePOVToFirstPerson (size_t id)

  *Change the camera to the player.*
- void changePOVToSecondPerson (size_t id)

  *Change the camera to the player.*
- void changePOVToThirdPerson (size_t id)

  *Change the camera to the player.*
- size_t getTimeUnit () const

  *Get the Time Unit value.*
- void setTimeUnit (size_t timeUnit)

  *Set the Time Unit value.*
- void setPlayerVision (bool isPlayerVision)

  *Set the Player Vision value.*
- bool getPlayerVision () const

  *Get the Player Vision value.*
- void setHelpMenu (bool isHelpMenu)

  *Set the Help Menu value.*
- bool getHelpMenu () const

  *Get the Help Menu value.*
- void drawEnd () const

  *Draw the end of the game.*

### 5.32.1 Constructor & Destructor Documentation

#### 5.32.1.1 Render()

```
Gui::Render::Render (
            std::shared_ptr< GameData > gameData )
```

Construct a new Render object.

#### 5.32.1.2 ~Render()

```
Gui::Render::~Render ( )
```

Destroy the Render object.

### 5.32.2 Member Function Documentation

#### 5.32.2.1 changePlayerPOV()

```
void Gui::Render::changePlayerPOV (
            size_t playerId )
```

Change the player point of view.

**Parameters**

| | |
|---|---|
| *playerId* | Player id to select. |

**Note**

The player point of view is the first person, second person and third person.

#### 5.32.2.2 changePOVToFirstPerson()

```
void Gui::Render::changePOVToFirstPerson (
            size_t id )
```

Change the camera to the player.

**Parameters**

| | |
|---|---|
| *player* | Player to select. |

#### 5.32.2.3 changePOVToSecondPerson()

```
void Gui::Render::changePOVToSecondPerson (
            size_t id )
```

Change the camera to the player.

**Parameters**

| | |
|---|---|
| *player* | Player to select. |

#### 5.32.2.4 changePOVToThirdPerson()

```
void Gui::Render::changePOVToThirdPerson (
            size_t id )
```

Change the camera to the player.

**Parameters**

| | |
|---|---|
| *player* | Player to select. |

#### 5.32.2.5 draw()

```
void Gui::Render::draw ( )
```

Draw the scene.

#### 5.32.2.6 drawEnd()

```
void Gui::Render::drawEnd ( ) const
```

Draw the end of the game.

#### 5.32.2.7 getCamera()

```
std::shared_ptr< Camera > Gui::Render::getCamera ( )
```

Get the Camera object.

**Returns**

std::shared_ptr<Camera> - camera

#### 5.32.2.8 getCameraPlayerPov()

```
std::size_t Gui::Render::getCameraPlayerPov ( ) const
```

Get the Camera player pov id.

**Returns**

std::size_t - Id of the player.

#### 5.32.2.9 getCameraTile()

```
std::pair< std::size_t, std::size_t > Gui::Render::getCameraTile ( ) const
```

Get the Camera Tile object.

**Returns**

std::pair<std::size_t, std::size_t> - Tile position.

#### 5.32.2.10 getCameraType()

```
Gui::UserCamera::CameraType Gui::Render::getCameraType ( ) const
```

Get the Type object.

**Returns**

CameraType - Camera type.

#### 5.32.2.11 getHelpMenu()

```
bool Gui::Render::getHelpMenu ( ) const
```

Get the Help Menu value.

**Returns**

true - Display the help menu.

false - Do not display the help menu.

#### 5.32.2.12 getIsDebug()

```
bool Gui::Render::getIsDebug (
            void  )
```

Get the Is Debug object.

**Returns**

true - diplay debug

false - do not display debug

#### 5.32.2.13 getPlayerVision()

```
bool Gui::Render::getPlayerVision ( ) const
```

Get the Player Vision value.

**Returns**

true - Display player vision.

false - Do not display player vision.

#### 5.32.2.14 getRenderDistance()

```
size_t Gui::Render::getRenderDistance ( ) const
```

Get the Render Distance value.

#### 5.32.2.15 getTileModel()

```
Model Gui::Render::getTileModel ( ) const
```

Get the Tile model.

#### 5.32.2.16 getTimeUnit()

```
size_t Gui::Render::getTimeUnit ( ) const
```

Get the Time Unit value.

**Returns**

size_t - Time unit value.

#### 5.32.2.17 isCameraInPlayerPov()

```
bool Gui::Render::isCameraInPlayerPov ( ) const
```

Check if the camera is in player pov.

**Returns**

true - Camera is in player pov.

false - Camera is not in player pov.

**Note**

The player pov is the first person, second person and third person.

#### 5.32.2.18 isOpen()

```
bool Gui::Render::isOpen ( )
```

Check if the window is open.

**Returns**

true - the window is open

false - the window is closed

#### 5.32.2.19 setCameraPlayerPov()

```
void Gui::Render::setCameraPlayerPov (
            std::size_t id )
```

Set the Camera player pov id.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |

#### 5.32.2.20 setCameraTile()

```
void Gui::Render::setCameraTile (
            std::pair< std::size_t, std::size_t > pos )
```

Set the Camera Tile object.

**Parameters**

| | |
|---|---|
| *pos* | Tile position. |

#### 5.32.2.21 setCameraType()

```
void Gui::Render::setCameraType (
            Gui::UserCamera::CameraType type )
```

Set the Type object.

**Parameters**

| | |
|---|---|
| *type* | Type to set. |

#### 5.32.2.22 setHelpMenu()

```
void Gui::Render::setHelpMenu (
            bool isHelpMenu )
```

Set the Help Menu value.

**Parameters**

| | |
|---|---|
| *isHelpMenu* | New help menu value. |

#### 5.32.2.23 setIsDebug()

```
void Gui::Render::setIsDebug (
            bool isDebug )
```

Set the Is Debug object.

**Parameters**

| | |
|---|---|
| *isDebug* | New Is Debug value to set. |

#### 5.32.2.24 setPlayerPov()

```
void Gui::Render::setPlayerPov (
            size_t playerId )
```

Sets the Pov of the player.

**Parameters**

| | |
|---|---|
| *playerId* | Player id to select. |

#### 5.32.2.25 setPlayerVision()

```
void Gui::Render::setPlayerVision (
            bool isPlayerVision )
```

Set the Player Vision value.

**Parameters**

| | |
|---|---|
| *isPlayerVision* | New player vision value. |

**Note**

True to display player vision.

False to not display player vision.

#### 5.32.2.26 setRenderDistance()

```
void Gui::Render::setRenderDistance (
            size_t renderDistance )
```

Set the Render Distance value.

**Parameters**

| | |
|---|---|
| *renderDistance* | New render distance value. |

#### 5.32.2.27 setTimeUnit()

```
void Gui::Render::setTimeUnit (
            size_t timeUnit )
```

Set the Time Unit value.

**Parameters**

| | |
|---|---|
| *timeUnit* | New time unit value. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Render/Render.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Render/Render.cpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Render/RenderAnimations.cpp

## 5.33 Gui::ServerParser Class Reference

Inheritance diagram for Gui::ServerParser:

Collaboration diagram for Gui::ServerParser:

### Public Member Functions

- ServerParser ()=default

  *Construct a new Server Parser object.*
- ~ServerParser ()=default

  *Destroy the Server Parser object.*
- std::vector< std::string > parse (const std::string &command)

  *Parse the command server.*

#### Public Member Functions inherited from Gui::IServerParser

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

  *Destroy the IServerParser object.*
- virtual std::vector< std::string > parse (const std::string &command)=0

  *Parse the command server.*

## 5.33.1 Constructor & Destructor Documentation

### 5.33.1.1 ServerParser()

```
Gui::ServerParser::ServerParser ( )  [default]
```

Construct a new Server Parser object.

### 5.33.1.2 ~ServerParser()

```
Gui::ServerParser::~ServerParser ( )  [default]
```

Destroy the Server Parser object.

## 5.33.2 Member Function Documentation

### 5.33.2.1 parse()

```
std::vector< std::string > Gui::ServerParser::parse (
            const std::string & command )  [virtual]
```

Parse the command server.

**Parameters**

| | |
|---|---|
| *command* | Command to parse. |

**Returns**

std::vector<std::string> - arguments parsed

Implements Gui::IServerParser.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Parsing/ServerParser.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Parsing/ServerParser.cpp

## 5.34 Gui::Errors::ServerParserException Class Reference

Error class for network errors.

`#include <Error.hpp>`

Inheritance diagram for Gui::Errors::ServerParserException:

Collaboration diagram for Gui::Errors::ServerParserException:

### Public Member Functions

- ServerParserException (std::string message)
  *Constructor for ServerParserException.*

#### Public Member Functions inherited from Gui::Errors::AError

- ~**AError** () override=default
  *Destructor.*
- const char * what () const noexcept override
  *Returns the error message.*

#### Public Member Functions inherited from Gui::Errors::IError

- virtual ~**IError** ()=default
  *Destructor for IError.*
- virtual const char * what () const noexcept=0
  *Returns a C-style string describing the error. This function must be implemented by derived classes to provide a description of the error. The returned string should be null-terminated.*

### Additional Inherited Members

#### Protected Attributes inherited from Gui::Errors::AError

- std::string **_message**
  *The error message.*

### 5.34.1 Detailed Description

Error class for network errors.

### 5.34.2 Constructor & Destructor Documentation

#### 5.34.2.1 ServerParserException()

```
Gui::Errors::ServerParserException::ServerParserException (
            std::string message )
```

Constructor for ServerParserException.

**Parameters**

| | |
|---|---|
| *message* | The error message. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Error/Error.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Error/Error.cpp

## 5.35 Gui::Team Class Reference

### Public Member Functions

- Team (const std::string &name, Model playerModel, Model eggModel, ModelAnimation *modelAnimation, Color playerColor)

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

  *Destroy the Team object.*
- const std::string & getName () const

  *Get the Name object.*
- std::vector< Gui::Player > & getPlayers ()

  *Get the Players object.*
- std::vector< Gui::Egg > & getEggs ()

  *Get the Eggs object.*
- void setName (const std::string &name)

  *Set the Name object.*
- void addPlayer (const Gui::Player &player)

  *Add a player to the team.*
- void addEgg (const Gui::Egg &egg)

  *Add an egg to the team.*
- bool removePlayer (std::size_t id)

  *Remove a player from the team.*
- bool removeEgg (std::size_t id)

  *Remove an egg from the team.*
- std::shared_ptr< Gui::Player > getPlayer (std::size_t id)

  *Get the Player object.*
- Model getPlayerModel () const

  *Get the Model object.*
- ModelAnimation * getPlayerModelAnimation () const

  *Get the Player Model Animation object.*
- void setPlayerModelPath (const std::string &playerModelPath)

  *Set the Model object.*
- std::shared_ptr< Gui::Egg > getEgg (std::size_t id)

  *Get the Egg object.*
- Model getEggModel () const

  *Get the Egg Model Path object.*
- void setEggModelPath (const std::string &eggModelPath)

  *Set the Egg Model Path object.*

- std::vector< BoundingBox > getPlayerBoundingBoxes (std::pair< size_t, size_t > pos, size_t orientation, Vector3 center)

  *Get the Player Boundig Boxes object.*
- Vector3 getPlayerPositionIn3DSpace (size_t id, Map< Tile > map)

  *Get the Player position in 3D space.*
- std::vector< RayCollision > getPlayerModelHitbox (size_t id, Camera camera)

  *Get the Player Model hitbox.*
- bool isPlayerHit (size_t id, Camera camera)

  *Check if the player is hit.*
- Color getPlayerColor () const

  *Get the Player Color object.*

### 5.35.1 Constructor & Destructor Documentation

#### 5.35.1.1 Team()

```
Gui::Team::Team (
            const std::string & name,
            Model playerModel,
            Model eggModel,
            ModelAnimation * modelAnimation,
            Color playerColor )
```

Construct a new Team object.

**Parameters**

| | |
|---|---|
| *name* | Name of the team. |
| *playerModel* | Model asset of the team. |
| *eggSkin* | Model asset of the eggs. |
| *playerColor* | Color of the players. |

#### 5.35.1.2 ~Team()

```
Gui::Team::~Team ( )
```

Destroy the Team object.

### 5.35.2 Member Function Documentation

#### 5.35.2.1 addEgg()

```
void Gui::Team::addEgg (
            const Gui::Egg & egg )
```

Add an egg to the team.

**Parameters**

| | |
|---|---|
| *egg* | Egg to add. |

#### 5.35.2.2 addPlayer()

```
void Gui::Team::addPlayer (
            const Gui::Player & player )
```

Add a player to the team.

**Parameters**

| | |
|---|---|
| *player* | Player to add. |

#### 5.35.2.3 getEgg()

```
std::shared_ptr< Gui::Egg > Gui::Team::getEgg (
            std::size_t id )
```

Get the Egg object.

**Parameters**

| | |
|---|---|
| *id* | Id of the egg. |

**Returns**

std::shared_ptr<Gui::Egg> Egg.

#### 5.35.2.4 getEggModel()

```
Model Gui::Team::getEggModel ( ) const
```

Get the Egg Model Path object.

**Returns**

const std::string& Path to the eggs Model of the team.

#### 5.35.2.5 getEggs()

```
std::vector< Gui::Egg > & Gui::Team::getEggs ( )
```

Get the Eggs object.

**Returns**

std::vector<Gui::Egg>& Eggs of the team.

#### 5.35.2.6 getName()

```
const std::string & Gui::Team::getName ( ) const
```

Get the Name object.

**Returns**

const std::string& Name of the team.

#### 5.35.2.7 getPlayer()

```
std::shared_ptr< Gui::Player > Gui::Team::getPlayer (
            std::size_t id )
```

Get the Player object.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |

**Returns**

std::shared_ptr<Gui::Player> Player.

#### 5.35.2.8 getPlayerBoundingBoxes()

```
std::vector< BoundingBox > Gui::Team::getPlayerBoundingBoxes (
            std::pair< size_t, size_t > pos,
            size_t orientation,
            Vector3 center )
```

Get the Player Boundig Boxes object.

**Parameters**

| | |
|---|---|
| *pos* | Position of the player. |
| *orientation* | Orientation of the player. |

**Returns**

std::vector<BoundingBox> Bounding boxes of the player.

#### 5.35.2.9 getPlayerColor()

```
Color Gui::Team::getPlayerColor ( ) const
```

Get the Player Color object.

#### 5.35.2.10 getPlayerModel()

```
Model Gui::Team::getPlayerModel ( ) const
```

Get the Model object.

**Returns**

Model - Model asset of the Team.

#### 5.35.2.11 getPlayerModelAnimation()

```
ModelAnimation * Gui::Team::getPlayerModelAnimation ( ) const
```

Get the Player Model Animation object.

**Returns**

ModelAnimation* - Players' animations.

#### 5.35.2.12 getPlayerModelHitbox()

```
std::vector< RayCollision > Gui::Team::getPlayerModelHitbox (
            size_t id,
            Camera camera )
```

Get the Player Model hitbox.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |
| *camera* | Camera of the game. |

**Returns**

std::vector<RayCollision> Hitbox of the player.

#### 5.35.2.13 getPlayerPositionIn3DSpace()

```
Vector3 Gui::Team::getPlayerPositionIn3DSpace (
            size_t id,
            Map< Tile > map )
```

Get the Player position in 3D space.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |
| *map* | Map of the game. |

#### 5.35.2.14 getPlayers()

```
std::vector< Gui::Player > & Gui::Team::getPlayers ( )
```

Get the Players object.

**Returns**

std::vector<Gui::Player>& Players of the team.

#### 5.35.2.15 isPlayerHit()

```
bool Gui::Team::isPlayerHit (
            size_t id,
            Camera camera )
```

Check if the player is hit.

**Parameters**

| | |
|---|---|
| *id* | Id of the player. |
| *camera* | Camera of the game. |

**Returns**

true If the player is hit.

#### 5.35.2.16 removeEgg()

```
bool Gui::Team::removeEgg (
            std::size_t id )
```

Remove an egg from the team.

**Parameters**

| | |
|---|---|
| *id* | Id of the egg to remove. |

**Returns**

true If the egg has been removed.

false If the egg has not been removed.

#### 5.35.2.17 removePlayer()

```
bool Gui::Team::removePlayer (
            std::size_t id )
```

Remove a player from the team.

**Parameters**

| | |
|---|---|
| *id* | Id of the player to remove. |

**Returns**

true If the player has been removed.

false If the player has not been removed.

#### 5.35.2.18 setEggModelPath()

```
void Gui::Team::setEggModelPath (
            const std::string & eggModelPath )
```

Set the Egg Model Path object.

**Parameters**

| | |
|---|---|
| *eggSkinPath* | Path to the eggs Model of the team. |

#### 5.35.2.19 setName()

```
void Gui::Team::setName (
            const std::string & name )
```

Set the Name object.

**Parameters**

| | |
|---|---|
| *name* | Name of the team. |

#### 5.35.2.20 setPlayerModelPath()

```
void Gui::Team::setPlayerModelPath (
            const std::string & playerModelPath )
```

Set the Model object.

**Parameters**

| | |
|---|---|
| *playerModelPath* | Path to the team model asset for players. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GameDatas/Team.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GameDatas/Team.cpp

## 5.36 Gui::Tile Class Reference

Collaboration diagram for Gui::Tile:

## Public Member Functions

- Tile (std::pair< std::size_t, std::size_t > position)

  *Construct a new Tile object.*
- Tile (std::pair< std::size_t, std::size_t > position, Inventory inventory)

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

  *Destroy the Tile object.*
- void setPosition (std::pair< std::size_t, std::size_t > position)

  *Set the Position object.*
- std::pair< std::size_t, std::size_t > getPosition () const

  *Get the Position object.*
- Vector3 getPositionIn3DSpace ()

  *Get the Position In Space object.*
- std::vector< BoundingBox > getTileBoundingBoxes (Tile tile, Model tileModel)

  *Get the Tile Bounding Boxes object.*
- std::vector< RayCollision > getTileModelHitbox (Tile tile, Camera camera, Model tileModel)

  *Get the Tile Model Hitbox object.*
- bool isTileHit (Camera camera, Model _tileModel)

  *Check if the tile is hit.*

## Public Attributes

- Inventory inventory

  *Inventory of the tile.*

### 5.36.1 Constructor & Destructor Documentation

#### 5.36.1.1 Tile() [1/2]

```
Gui::Tile::Tile (
            std::pair< std::size_t, std::size_t > position )
```

Construct a new Tile object.

**Parameters**

| | |
|---|---|
| *position* | Set the position of the tile. |

#### 5.36.1.2 Tile() [2/2]

```
Gui::Tile::Tile (
            std::pair< std::size_t, std::size_t > position,
            Inventory inventory )
```

Construct a new Tile object.

**Parameters**

| | |
|---|---|
| *position* | Set the position of the tile. |
| *inventory* | Set the inventory of the tile. |

#### 5.36.1.3 ~Tile()

```
Gui::Tile::~Tile ( )  [default]
```

Destroy the Tile object.

### 5.36.2 Member Function Documentation

#### 5.36.2.1 getPosition()

```
std::pair< std::size_t, std::size_t > Gui::Tile::getPosition ( ) const
```

Get the Position object.

**Returns**

std::pair<std::size_t, std::size_t> - position x y

#### 5.36.2.2 getPositionIn3DSpace()

```
Vector3 Gui::Tile::getPositionIn3DSpace ( )
```

Get the Position In Space object.

**Returns**

Vector3 - Position in space.

#### 5.36.2.3 getTileBoundingBoxes()

```
std::vector< BoundingBox > Gui::Tile::getTileBoundingBoxes (
            Tile tile,
            Model tileModel )
```

Get the Tile Bounding Boxes object.

**Parameters**

| | |
|---|---|
| *tile* | Tile to get the bounding boxes. |

**Returns**

std::vector<BoundingBox> - Bounding boxes of the tile.

#### 5.36.2.4 getTileModelHitbox()

```
std::vector< RayCollision > Gui::Tile::getTileModelHitbox (
            Tile tile,
            Camera camera,
            Model tileModel )
```

Get the Tile Model Hitbox object.

**Parameters**

| | |
|---|---|
| *tile* | Tile to get the hitbox. |
| *camera* | Camera to get the hitbox. |

**Returns**

std::vector<RayCollision> - Hitbox of the tile.

#### 5.36.2.5 isTileHit()

```
bool Gui::Tile::isTileHit (
            Camera camera,
            Model _tileModel )
```

Check if the tile is hit.

**Parameters**

| | |
|---|---|
| *camera* | Camera to check if the tile is hit. |
| *_tileModel* | Model of the tile. |

**Returns**

true - The tile is hit.

false - The tile is not hit.

#### 5.36.2.6 setPosition()

```
void Gui::Tile::setPosition (
            std::pair< std::size_t, std::size_t > position )
```

Set the Position object.

**Parameters**

| | |
|---|---|
| *position* | New position of the tile. |

### 5.36.3 Member Data Documentation

#### 5.36.3.1 inventory

```
Inventory Gui::Tile::inventory
```

Inventory of the tile.

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/GameDatas/Tile.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/GameDatas/Tile.cpp

## 5.37 Gui::UserCamera Class Reference

### Public Types

- enum **CameraType** {
  **FREE** , **FIRST_PERSON** , **SECOND_PERSON** , **THIRD_PERSON** ,
  **FREE_TILE** }

### Public Member Functions

- UserCamera ()

  *Construct a new User Camera object.*
- ~UserCamera ()=default

  *Destroy the User Camera object.*
- void setPosition (Vector3 position)

  *Set the Position object.*
- void setTarget (Vector3 target)

  *Set the Target object.*
- void setUp (Vector3 up)

  *Set the Up object.*

- void setFovy (float fovy)

  *Set the Fovy object.*
- Vector3 getPosition (void) const

  *Get the Position object.*
- Vector3 getTarget (void) const

  *Get the Target object.*
- Vector3 getUp (void) const

  *Get the Up object.*
- float getFovy (void) const

  *Get the Fovy object.*
- std::shared_ptr< Camera > getCamera ()

  *Get the Camera object.*
- void setType (CameraType type)

  *Set the Type object.*
- CameraType getType () const

  *Get the Type object.*
- void setPlayerId (size_t playerId)

  *Set the Player Id object.*
- int getPlayerId () const

  *Get the Player Id object.*
- void setTilePos (std::pair< std::size_t, std::size_t > pos)

  *Set the Tile Pos object.*
- std::pair< std::size_t, std::size_t > getTilePos () const

  *Get the Tile position object.*
- bool isPlayerPov () const

  *Check if the camera is in player pov.*
- bool isPlayerVision () const

  *Check if the camera is in player vision.*
- void setPlayerVision (bool isPlayerVision)

  *Set the Player Vision object.*

## 5.37.1 Constructor & Destructor Documentation

### 5.37.1.1 UserCamera()

```
Gui::UserCamera::UserCamera ( )
```

Construct a new User Camera object.

### 5.37.1.2 ~UserCamera()

```
Gui::UserCamera::~UserCamera ( )  [default]
```

Destroy the User Camera object.

## 5.37.2 Member Function Documentation

### 5.37.2.1 getCamera()

```
std::shared_ptr< Camera > Gui::UserCamera::getCamera ( )
```

Get the Camera object.

**Returns**

Camera - camera

### 5.37.2.2 getFovy()

```
float Gui::UserCamera::getFovy (
            void  ) const
```

Get the Fovy object.

**Returns**

float - fovy

### 5.37.2.3 getPlayerId()

```
int Gui::UserCamera::getPlayerId ( ) const
```

Get the Player Id object.

**Returns**

int - Player id.

### 5.37.2.4 getPosition()

```
Vector3 Gui::UserCamera::getPosition (
            void  ) const
```

Get the Position object.

**Returns**

Vector3 - position

#### 5.37.2.5 getTarget()

```
Vector3 Gui::UserCamera::getTarget (
            void  ) const
```

Get the Target object.

**Returns**

Vector3 - target

#### 5.37.2.6 getTilePos()

```
std::pair< std::size_t, std::size_t > Gui::UserCamera::getTilePos ( ) const
```

Get the Tile position object.

**Returns**

std::pair<std::size_t, std::size_t> - Position of the tile.

#### 5.37.2.7 getType()

```
Gui::UserCamera::CameraType Gui::UserCamera::getType ( ) const
```

Get the Type object.

**Returns**

CameraType - Camera type.

#### 5.37.2.8 getUp()

```
Vector3 Gui::UserCamera::getUp (
            void  ) const
```

Get the Up object.

**Returns**

Vector3 - up

#### 5.37.2.9 isPlayerPov()

```
bool Gui::UserCamera::isPlayerPov ( ) const
```

Check if the camera is in player pov.

**Returns**

true - Camera is in player pov.

false - Camera is not in player pov.

**Note**

The player pov is the first person, second person and third person.

#### 5.37.2.10 isPlayerVision()

```
bool Gui::UserCamera::isPlayerVision ( ) const
```

Check if the camera is in player vision.

**Returns**

true - Camera is in player vision.

false - Camera is not in player vision.

#### 5.37.2.11 setFovy()

```
void Gui::UserCamera::setFovy (
            float fovy )
```

Set the Fovy object.

**Parameters**

| | |
|---|---|
| *fovy* | New camera fovy. |

#### 5.37.2.12 setPlayerId()

```
void Gui::UserCamera::setPlayerId (
            size_t playerId )
```

Set the Player Id object.

**Parameters**

| | |
|---|---|
| *playerId* | Player id to set. |

#### 5.37.2.13 setPlayerVision()

```
void Gui::UserCamera::setPlayerVision (
            bool isPlayerVision )
```

Set the Player Vision object.

**Parameters**

| | |
|---|---|
| *isPlayerVision* | Is player vision. |

#### 5.37.2.14 setPosition()

```
void Gui::UserCamera::setPosition (
            Vector3 position )
```

Set the Position object.

**Parameters**

| | |
|---|---|
| *position* | New camera position. |

#### 5.37.2.15 setTarget()

```
void Gui::UserCamera::setTarget (
            Vector3 target )
```

Set the Target object.

**Parameters**

| | |
|---|---|
| *target* | New camera target. |

#### 5.37.2.16 setTilePos()

```
void Gui::UserCamera::setTilePos (
            std::pair< std::size_t, std::size_t > pos )
```

Set the Tile Pos object.

**Parameters**

| | |
|---|---|
| *pos* | Position of the tile. |

#### 5.37.2.17 setType()

```
void Gui::UserCamera::setType (
            CameraType type )
```

Set the Type object.

**Parameters**

| | |
|---|---|
| *type* | Type to set. |

#### 5.37.2.18 setUp()

```
void Gui::UserCamera::setUp (
            Vector3 up )
```

Set the Up object.

**Parameters**

| | |
|---|---|
| *up* | New camera up vector. |

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

- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/include/Render/UserCamera.hpp
- /home/tjerome-rocher/Desktop/Tek2/Zappy/gui/src/Render/UserCamera.cpp

# Chapter 6

# File Documentation

## 6.1 Assets.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Assets
00006 */
00007
00008 #pragma once
00009
00010 #include "Config.hpp"
00011
00012 #include <vector>
00013 #include <string>
00014
00015 #define PATH_ASSETS                     "gui/assets/"
00016
00017 #define PATH_RESOURCES                  "resources/"
00018 #define PATH_PLAYER                     "player/"
00019 #define PATH_TILE                       "tile/"
00020 #define PATH_DECORATION                 "decoration/"
00021 #define PATH_HUD                        "hud/"
00022 #define PATH_HELP                       "help/"
00023 #define PATH_END                        "end/"
00024
00025 #define PNG_CURSOR                      PATH_ASSETS PATH_HUD "cursor.png"
00026 #define PNG_END_HUD                     PATH_ASSETS PATH_END "end_hud.png"
00027 #define PNG_HUD                         PATH_ASSETS PATH_HUD "hud.png"
00028 #define PNG_FOOD                        PATH_ASSETS PATH_HUD "food.png"
00029 #define PNG_LINEMATE                    PATH_ASSETS PATH_HUD "linemate.png"
00030 #define PNG_DERAUMERE                   PATH_ASSETS PATH_HUD "deraumere.png"
00031 #define PNG_MENDIANE                    PATH_ASSETS PATH_HUD "mendiane.png"
00032 #define PNG_PHIRAS                      PATH_ASSETS PATH_HUD "phiras.png"
00033 #define PNG_SIBUR                       PATH_ASSETS PATH_HUD "sibur.png"
00034 #define PNG_THYSTAME                    PATH_ASSETS PATH_HUD "thystame.png"
00035 #define PNG_PLAYER                      PATH_ASSETS PATH_HUD "player.png"
00036 #define PNG_TILE                        PATH_ASSETS PATH_HUD "tile.png"
00037 #define PNG_EGG                         PATH_ASSETS PATH_HUD "egg.png"
00038 #define FONT_HUD                        PATH_ASSETS PATH_HUD "SimplyMono-Bold.ttf"
00039
00040 #define PNG_HELP                        PATH_ASSETS PATH_HELP "help_menu.png"
00041 #define PNG_HELP_KEYS                   PATH_ASSETS PATH_HELP "keys.png"
00042 #define PNG_HUD                         PATH_ASSETS PATH_HUD "hud.png"
00043 #define PNG_FOOD                        PATH_ASSETS PATH_HUD "food.png"
00044 #define PNG_LINEMATE                    PATH_ASSETS PATH_HUD "linemate.png"
00045 #define PNG_DERAUMERE                   PATH_ASSETS PATH_HUD "deraumere.png"
00046 #define PNG_MENDIANE                    PATH_ASSETS PATH_HUD "mendiane.png"
00047 #define PNG_PHIRAS                      PATH_ASSETS PATH_HUD "phiras.png"
00048 #define PNG_SIBUR                       PATH_ASSETS PATH_HUD "sibur.png"
00049 #define PNG_THYSTAME                    PATH_ASSETS PATH_HUD "thystame.png"
00050 #define PNG_PLAYER                      PATH_ASSETS PATH_HUD "player.png"
00051 #define PNG_TILE                        PATH_ASSETS PATH_HUD "tile.png"
00052 #define FONT_HUD                        PATH_ASSETS PATH_HUD "SimplyMono-Bold.ttf"
00053
00054 #define MODEL_TILE                      PATH_ASSETS PATH_TILE "tile.glb"
00055 #define MODEL_FOOD                      PATH_ASSETS PATH_RESOURCES "food.glb"
00056 #define MODEL_LINEMATE                  PATH_ASSETS PATH_RESOURCES "linemate.glb"
00057 #define MODEL_MENDIANE                  PATH_ASSETS PATH_RESOURCES "mendiane.glb"
00058 #define MODEL_PHIRAS                    PATH_ASSETS PATH_RESOURCES "phiras.glb"
```

```
00059 #define MODEL_SIBUR                   PATH_ASSETS PATH_RESOURCES "sibur.glb"
00060 #define MODEL_THYSTAME                PATH_ASSETS PATH_RESOURCES "thystame.glb"
00061 #define MODEL_DERAUMERE               PATH_ASSETS PATH_RESOURCES "deraumere.glb"
00062 #define MODEL_PLAYER                  PATH_ASSETS PATH_PLAYER "player.glb"
00063 #define MODEL_EGG                     PATH_ASSETS PATH_PLAYER "egg.glb"
00064 #define MODEL_TREE                    PATH_ASSETS PATH_DECORATION "tree.glb"
00065 #define MODEL_LANTERN                 PATH_ASSETS PATH_DECORATION "lantern.glb"
00066
00067 #define SCALE_FOOD                    (Vector3){1, 0.5, 1}
00068 #define SCALE_LINEMATE                (Vector3){0.1, 0.1, 0.1}
00069 #define SCALE_MENDIANE                (Vector3){0.1, 0.1, 0.1}
00070 #define SCALE_PHIRAS                  (Vector3){0.001, 0.001, 0.005}
00071 #define SCALE_SIBUR                   (Vector3){0.01, 0.01, 0.01}
00072 #define SCALE_THYSTAME                (Vector3){2, 2, 2}
00073 #define SCALE_DERAUMERE               (Vector3){0.5, 0.5, 0.5}
00074 #define SCALE_PLAYER                  (Vector3){0.5, 0.5, 0.5}
00075 #define SCALE_EGG                     (Vector3){0.5, 0.5, 0.5}
00076 #define SCALE_TREE                    (Vector3){1, 1, 1}
00077 #define SCALE_LANTERN                 (Vector3){1, 1, 1}
00078
00079 #define ROTATION_ANGLE_FOOD           45
00080 #define ROTATION_ANGLE_LINEMATE       270
00081 #define ROTATION_ANGLE_MENDIANE       0
00082 #define ROTATION_ANGLE_PHIRAS         270
00083 #define ROTATION_ANGLE_SIBUR          270
00084 #define ROTATION_ANGLE_THYSTAME       270
00085 #define ROTATION_ANGLE_DERAUMERE      90
00086 #define ROTATION_ANGLE_PLAYER         0
00087 #define ROTATION_ANGLE_EGG            270
00088 #define ROTATION_ANGLE_TREE           0
00089 #define ROTATION_ANGLE_LANTERN        0
00090
00091 #define ROTATION_AXIS_FOOD            (Vector3){0, 1, 0}
00092 #define ROTATION_AXIS_LINEMATE        (Vector3){1, 0, 0}
00093 #define ROTATION_AXIS_MENDIANE        (Vector3){1, 0, 0}
00094 #define ROTATION_AXIS_PHIRAS          (Vector3){1, 0, 0}
00095 #define ROTATION_AXIS_SIBUR           (Vector3){1, 0, 0}
00096 #define ROTATION_AXIS_THYSTAME        (Vector3){1, 0, 0}
00097 #define ROTATION_AXIS_DERAUMERE       (Vector3){1, 0, 0}
00098 #define ROTATION_AXIS_PLAYER          (Vector3){0, 1, 0}
00099 #define ROTATION_AXIS_EGG             (Vector3){1, 0, 0}
00100 #define ROTATION_AXIS_TREE            (Vector3){1, 0, 0}
00101 #define ROTATION_AXIS_LANTERN         (Vector3){1, 0, 0}
00102
00103 #define POS_FOOD                      (Vector3){0.5, -0.1, 1.5}
00104 #define POS_LINEMATE                  (Vector3){1, -0.3, -0.5}
00105 #define POS_MENDIANE                  (Vector3){2, -0.25, -0.5}
00106 #define POS_PHIRAS                    (Vector3){0.5, -0.3, -1.5}
00107 #define POS_SIBUR                     (Vector3){1.5, -0.3, -1.5}
00108 #define POS_THYSTAME                  (Vector3){1, 0, -2}
00109 #define POS_DERAUMERE                 (Vector3){2, -0.3, -2}
00110 #define POS_PLAYER                    (Vector3){0, -0.25, 0}
00111 #define POS_EGG                       (Vector3){0.5, 0, 0.5}
00112 #define POS_TREE                      (Vector3){2, -0.3, 2}
00113 #define POS_LANTERN                   (Vector3){1, -0.3, 2}
00114 #define POS_Y_DELIMITATION            -0.27f
00115
00116 #define PLAYER_TEXT_SIZE              40
00117 #define PLAYER_TEXT_SIZE_RATIO        1.5f
00118
00119 #define HELP_BACKGROUND_SCALE         3.5f
00120 #define HELP_BACKGROUND_TEXTURE_SIZE  (Vector2){400 * HELP_BACKGROUND_SCALE, 200 *
      HELP_BACKGROUND_SCALE}
00121 #define HELP_BACKGROUND_POSITION      (Vector2){WINDOW_WIDTH / 2 - HELP_BACKGROUND_TEXTURE_SIZE.x /
      2, WINDOW_HEIGHT / 2 - HELP_BACKGROUND_TEXTURE_SIZE.y / 2}
00122 #define HELP_TEXT_POSITION_LEFT       (Vector2){HELP_BACKGROUND_POSITION.x + 270,
      HELP_BACKGROUND_POSITION.y + 180}
00123 #define HELP_TEXT_POSITION_RIGHT      (Vector2){HELP_TEXT_POSITION_LEFT.x + 610,
      HELP_TEXT_POSITION_LEFT.y}
00124 #define HELP_TEXT_SPACING             30
00125 #define HELP_TITLE_OFFSET             90
00126
00127 #define END_HUD_TEXT_SIZE             40
00128 #define END_HUD_ADVICE_TEXT_SIZE      30
00129 #define END_HUD_ADVICE_OFFSET         150
00130
00131 static std::vector<std::string> globalControlsTexts = {
00132     "Move the camera in the space",
00133     "Move the camera up",
00134     "Move the camera down",
00135     "Switch on/off the debug display",
00136     "Reset the HUD from tile view to game view",
00137     "Switch on/off the help HELP",
00138     "Increase the render distance",
00139     "Decrease the render distance",
00140     "Increase the time unit",
00141     "Decrease the time unit",
```

```
00142     "Select the player point of view",
00143     "Select the tile to display its HUD",
00144 };
00145
00146 static std::vector<std::string> playerControlsTexts = {
00147     "Change the player point of view",
00148     "Get off the player point of view",
00149     "Switch to the next player",
00150     "Switch to the previous player",
00151     "Switch on/off the player vision",
00152 };
```

## 6.2 Colors.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Colors
00006 */
00007
00008 #pragma once
00009
00010 #include "raylib.h"
00011
00012 #include <vector>
00013
00014 #define STR_BLUE "\033[0;34m"
00015 #define STR_GREEN "\033[0;32m"
00016 #define STR_RED "\033[0;31m"
00017 #define STR_YELLOW "\033[0;33m"
00018 #define STR_VIOLET "\033[0;35m"
00019 #define STR_CYAN "\033[0;36m"
00020 #define STR_RESET "\033[0m"
00021
00022 static std::vector<Color> playerColors = {
00023     PINK,
00024     LIGHTGRAY,
00025     GRAY,
00026     DARKGRAY,
00027     YELLOW,
00028     GOLD,
00029     ORANGE,
00030     RED,
00031     MAROON,
00032     GREEN,
00033     LIME,
00034     DARKGREEN,
00035     SKYBLUE,
00036     BLUE,
00037     DARKBLUE,
00038     PURPLE,
00039     VIOLET,
00040     DARKPURPLE,
00041     BEIGE,
00042     BROWN,
00043     DARKBROWN,
00044     WHITE,
00045     BLACK,
00046     MAGENTA,
00047     RAYWHITE
00048 };
```

## 6.3 Config.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Config
00006 */
00007
00008 #pragma once
00009
00010 #define WINDOW_WIDTH 1920
00011 #define WINDOW_HEIGHT 1080
00012 #define WINDOW_TITLE "Zappy GUI"
00013
00014 #define SERVER_DOWN_MESSAGE         "THE GUI MUST BE CLOSED FOR UNEXPECTED REASONS"
```

```
00015
00016 #define SIZE_TILE                   4.7
00017
00018 #define PLAYER_HEIGHT               2
00019
00020 #define DEFAULT_RENDER_DISTANCE     15
00021 #define MAX_RENDER_DISTANCE         20
00022 #define MIN_RENDER_DISTANCE         1
00023
00024 #define HIGH_CAMERA_INCREASE        0.1
00025 #define LOW_CAMERA_INCREASE         0.1
00026
00027 #define PLAYER_SECOND_PERSON_FOV    4.0f
00028 #define PLAYER_THIRD_PERSON_FOV     5.0f
```

## 6.4 Engine.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Engine
00006 */
00007
00008 #pragma once
00009
00010 #include "Event/Event.hpp"
00011 #include "Render/Render.hpp"
00012 #include "Network/INetwork.hpp"
00013 #include "GameDatas/GameData.hpp"
00014 #include "Parsing/ServerParser.hpp"
00015 #include "GUIUpdater/GUIUpdater.hpp"
00016
00017 #include <thread>
00018 #include <time.h>
00019
00020 #define TIME_UNIT_MAP_UPDATE 20
00021
00022 namespace Gui {
00023
00028     class Engine;
00029 };
00030
00031 class Gui::Engine {
00032
00033     public:
00034
00040         Engine(std::shared_ptr<INetwork> network);
00041
00046         ~Engine() = default;
00047
00053         void run();
00054
00055     private:
00056
00057         std::unique_ptr<IServerParser>      _parser;
00058         std::shared_ptr<INetwork>           _network;
00059         std::shared_ptr<Render>             _render;
00060         std::unique_ptr<IEvent>             _event;
00061         std::shared_ptr<GameData>           _gameData;
00062         std::unique_ptr<IGUIUpdater>        _guiUpdater;
00063         std::thread                         _networkThread;
00064
00069         void listenServer();
00070
00078         void sendMessageUpdate();
00079
00084         void updateMap();
00085
00090         void sendUpdateMapMessage();
00091
00097         void threadLoop();
00098 };
```

## 6.5 AError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
```

```
00004 ** File description:
00005 ** AError
00006 */
00007
00008
00009 #pragma once
00010
00011 #include "Error/IError.hpp"
00012
00013 #include <string>
00014
00015 namespace Gui {
00016     namespace Errors {
00023         class AError;
00024     };
00025 };
00026
00027 class Gui::Errors::AError : public IError {
00028
00029     public:
00030
00034         ~AError() override = default;
00035
00040         const char *what() const noexcept override;
00041
00042     protected:
00043
00044         std::string _message;
00045 };
```

## 6.6 Error.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Error
00006 */
00007
00008 #pragma once
00009
00010 #include "Error/AError.hpp"
00011
00012 namespace Gui {
00013     namespace Errors {
00014
00019         class Error : public AError {};
00020
00025         class NetworkException : public Error {
00026
00027             public:
00033                 NetworkException(std::string message);
00034         };
00035
00040         class ServerParserException : public Error {
00041
00042             public:
00048                 ServerParserException(std::string message);
00049         };
00050
00055         class ParseCommandLineException : public Error {
00056
00057             public:
00063                 ParseCommandLineException(std::string message);
00064         };
00065
00070         class GuiGameDataException : public Error {
00071
00072             public:
00078                 GuiGameDataException(std::string message);
00079         };
00080
00085         class GuiUpdaterException : public Error {
00086
00087             public:
00093                 GuiUpdaterException(std::string message);
00094         };
00095     };
00096 };
```

## 6.7 IError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** IError
00006 */
00007
00008 #pragma once
00009
00010 #include <exception>
00011
00012 namespace Gui {
00013     namespace Errors {
00019         class IError;
00020     };
00021 };
00022
00023 class Gui::Errors::IError : public std::exception {
00024
00025     public:
00026
00030         virtual ~IError() = default;
00031
00039         virtual const char *what() const noexcept = 0;
00040 };
```

## 6.8 AEvent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** AEvent
00006 */
00007
00008 #pragma once
00009
00010 #include "Event/IEvent.hpp"
00011
00012 namespace Gui {
00013
00018     class AEvent;
00019 }
00020
00021 class Gui::AEvent : public Gui::IEvent {
00022
00023     public:
00024
00029         AEvent();
00030
00035         ~AEvent() = default;
00036
00041         virtual void listen() = 0;
00042
00048         void setRender(std::shared_ptr<Render> render);
00049
00055         void setGameData(std::shared_ptr<GameData> gameData);
00056
00062         std::shared_ptr<Render> getRender();
00063
00069         std::shared_ptr<GameData> getGameData();
00070
00071     protected:
00072
00073         std::shared_ptr<Render>         _render;
00074         std::shared_ptr<GameData>       _gameData;
00075 };
```

## 6.9 Event.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Event
00006 */
00007
```

```
00008 #pragma once
00009
00010 #include "Config.hpp"
00011 #include "Event/AEvent.hpp"
00012 #include "Render/Render.hpp"
00013
00014 #include <functional>
00015 #include <unordered_map>
00016
00017 namespace Gui {
00018
00023     class Event;
00024 };
00025
00026 class Gui::Event : public Gui::AEvent {
00027
00028     public:
00029
00034         Event();
00035
00040         ~Event() = default;
00041
00046         void listen();
00047
00048     private:
00049
00054         std::unordered_map<KeyboardKey, std::function<void()» _eventsKeyDown =
00055         {
00056             {KEY_SPACE, [this](){moveUpCamera();}},
00057             {KEY_LEFT_SHIFT, [this](){moveDownCamera();}},
00058         };
00059
00060         std::unordered_map<GamepadButton, std::function<void()» _eventsGamepadButtonDown =
00061         {
00062             {GAMEPAD_BUTTON_RIGHT_TRIGGER_2, [this](){handleSpaceGamepad();}},
00063             {GAMEPAD_BUTTON_LEFT_TRIGGER_2, [this](){moveDownCamera();}},
00064         };
00065
00070         std::unordered_map<KeyboardKey, std::function<void()» _eventsKeyPressed =
00071         {
00072             {KEY_THREE, [this](){switchDisplayDebug();}},
00073             {KEY_F3, [this](){switchDisplayDebug();}},
00074             {KEY_SPACE, [this](){setFreeCam();}},
00075             {KEY_R, [this](){switchTileHudToGame();}},
00076             {KEY_J, [this](){increaseRenderDistance();}},
00077             {KEY_K, [this](){decreaseRenderDistance();}},
00078             {KEY_F5, [this](){changeActualPlayerPov();}},
00079             {KEY_FOUR, [this](){changeActualPlayerPov();}},
00080             {KEY_V, [this](){setPlayerVision();}},
00081             {KEY_KP_ADD, [this](){increaseTimeUnit();}},
00082             {KEY_KP_SUBTRACT, [this](){decreaseTimeUnit();}},
00083             {KEY_H, [this](){displayHelpMenu();}},
00084         };
00085
00086         std::unordered_map<GamepadButton, std::function<void()» _eventsGamepadButtonPressed =
00087         {
00088             {GAMEPAD_BUTTON_RIGHT_FACE_DOWN, [this](){handleLeftClick();}},
00089             {GAMEPAD_BUTTON_RIGHT_TRIGGER_1, [this](){handleLeftClick();}},
00090             {GAMEPAD_BUTTON_LEFT_TRIGGER_1, [this](){handleRightClick();}},
00091             {GAMEPAD_BUTTON_LEFT_FACE_UP, [this](){increaseRenderDistance();}},
00092             {GAMEPAD_BUTTON_LEFT_FACE_DOWN, [this](){decreaseRenderDistance();}},
00093             {GAMEPAD_BUTTON_LEFT_FACE_LEFT, [this](){decreaseTimeUnit();}},
00094             {GAMEPAD_BUTTON_LEFT_FACE_RIGHT, [this](){increaseTimeUnit();}},
00095             {GAMEPAD_BUTTON_RIGHT_FACE_LEFT, [this](){switchDisplayDebug();}},
00096             {GAMEPAD_BUTTON_RIGHT_FACE_RIGHT, [this](){switchTileHudToGame();}},
00097             {GAMEPAD_BUTTON_RIGHT_FACE_UP, [this](){changeActualPlayerPov();}},
00098             {GAMEPAD_BUTTON_MIDDLE_RIGHT, [this](){closeWindowGamepad();}},
00099         };
00100
00105         std::unordered_map<MouseButton, std::function<void()» _eventsMousePressed =
00106         {
00107             {MOUSE_BUTTON_LEFT, [this](){handleLeftClick();}},
00108             {MOUSE_BUTTON_RIGHT, [this](){handleRightClick();}}
00109         };
00110
00115         void moveUpCamera();
00116
00121         void moveDownCamera();
00122
00127         void switchDisplayDebug();
00128
00132         void setFreeCam();
00133
00138         void handleLeftClick();
00139
00143         void handleRightClick();
00144
```

```
00149         void selectPlayer();
00150
00154         void selectTile();
00155
00161         void changePlayer(bool turn);
00162
00167         void changeActualPlayerPov();
00168
00173         void switchTileHudToGame();
00174
00179         void increaseRenderDistance();
00180
00185         void decreaseRenderDistance();
00186
00191         void increaseTimeUnit();
00192
00197         void decreaseTimeUnit();
00198
00203         void displayHelpMenu();
00204
00208         void handleSpaceGamepad();
00209
00213         void closeWindowGamepad();
00214
00219         void setPlayerVision();
00220 };
```

## 6.10 IEvent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** IEvent
00006 */
00007
00008 #pragma once
00009
00010 #include "Render/Render.hpp"
00011 #include "GameDatas/GameData.hpp"
00012 #include <memory>
00013
00014 namespace Gui {
00015
00020     class IEvent;
00021 }
00022
00023 class Gui::IEvent {
00024
00025     public:
00026
00031         IEvent() = default;
00032
00037         virtual ~IEvent() = default;
00038
00043         virtual void listen() = 0;
00044
00050         virtual void setRender(std::shared_ptr<Render> render) = 0;
00051
00057         virtual void setGameData(std::shared_ptr<GameData> gameData) = 0;
00058 };
```

## 6.11 Egg.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Egg
00006 */
00007
00008 #pragma once
00009
00010 #include "Error/Error.hpp"
00011
00012 #include <string>
00013
00014 namespace Gui {
00015
```

```
00019     class Egg;
00020 };
00021
00022 class Gui::Egg {
00023
00024     public:
00025
00026         enum EggState {
00027             IDLE,
00028             DEAD,
00029             BORN,
00030             HATCHING,
00031         };
00032
00043         Egg(size_t id, const std::string &team, std::pair<std::size_t, std::size_t> position);
00044
00050         ~Egg();
00051
00057         std::size_t getId() const;
00058
00064         std::string getTeam() const;
00065
00071         std::pair<std::size_t, std::size_t> getPosition() const;
00072
00078         void setId(std::size_t id);
00079
00085         void setTeam(const std::string &team);
00086
00092         void setPosition(std::pair<std::size_t, std::size_t> position);
00093
00099         void setState(EggState state);
00100
00106         EggState getState() const;
00107
00108     private:
00109
00110         std::size_t                             _id;
00111         std::string                             _team;
00112         std::pair<std::size_t, std::size_t>     _position;
00113         EggState                                _state;
00114 };
```

## 6.12 GameData.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** GameData
00006 */
00007
00008 #pragma once
00009
00010 #include "Types.hpp"
00011 #include "Error/Error.hpp"
00012 #include "GameDatas/Team.hpp"
00013 #include "GameDatas/Tile.hpp"
00014
00015 #define NO_TICK int(-1)
00016
00017 namespace Gui {
00018
00023     class GameData;
00024 };
00025
00026 class Gui::GameData {
00027
00028     public:
00029
00030         enum TimeUnitState {
00031             INCREASE,
00032             DECREASE,
00033             NONE
00034         };
00035
00040         GameData();
00041
00046         ~GameData() = default;
00047
00053         std::vector<Gui::Team> &getTeams();
00054
00061         Gui::Team &getTeam(const std::string &name);
00062
```

```
00068         void addTeam(const Gui::Team &team);
00069
00076         void addTeam(const std::string &name, Color playerColor);
00077
00084         void addPlayerToTeam(const std::string &teamName, const Gui::Player &player);
00085
00091         Gui::Player &getPlayer(size_t id);
00092
00098         Map<Gui::Tile> &getMap();
00099
00105         void setMap(const Map<Gui::Tile> &map);
00106
00114         void setMapSize(size_t x, size_t y);
00115
00121         std::pair<size_t, size_t> getMapSize() const;
00122
00130         Gui::Tile &getTile(size_t x, size_t y);
00131
00139         void setTile(const Gui::Tile &tile);
00140
00145         void restartLastTick(void);
00146
00152         void setServerTick(std::size_t tick);
00153
00159         clock_t getLastTick() const;
00160
00166         std::size_t getServerTick() const;
00167
00173         void setIsEndGame(bool isEndGame);
00174
00181         bool getIsEndGame() const;
00182
00188         void setLastError(const std::string &error);
00189
00195         std::string getLastError() const;
00196
00203         Team &getTeamById(std::size_t id);
00204
00210         TimeUnitState getTimeUnitFromServer() const;
00211
00217         void setTimeUnitFromServer(TimeUnitState timeUnitFromServer);
00218
00224         std::vector<Gui::Egg> &getServerEggs();
00225
00231         void addServerEgg(const Gui::Egg &egg);
00232
00238         void removeServerEgg(size_t id);
00239
00245         void setNbBCTCommandReceived(std::size_t nb);
00246
00252         std::size_t getNbBCTCommandReceived() const;
00253
00258         void restartLastTickMctCommand();
00259
00265         clock_t getLastTickMctCommand() const;
00266
00272         void setEndMessage(const std::string &endMessage);
00273
00279         std::string getEndMessage() const;
00280
00286         void setPlayerModel(const Model &playerModel);
00287
00293         void setEggModel(const Model &eggModel);
00294
00300         void setPlayerModelAnimation(ModelAnimation *playerModelAnimation);
00301
00307         void setAnimsCount(int animsCount);
00308
00314         void setServerError(bool isServerError);
00315
00321         bool getServerError() const;
00322
00323     private:
00324
00325         std::vector<Gui::Team>      _teams;
00326         Map<Gui::Tile>              _map;
00327         std::size_t                 _serverTick;
00328         clock_t                     _lastTick;
00329         bool                        _isEndGame;
00330         std::size_t                 _nbBCTCommandReceived;
00331         clock_t                     _lastTickMctCommand;
00332         std::string                 _lastError;
00333         TimeUnitState               _timeUnitFromServer;
00334         std::vector<Gui::Egg>       _serverEggs;
00335         std::string                 _endMessage;
00336         Model                       _playerModel;
00337         Model                       _eggModel;
```

```
00338         ModelAnimation*             _playerModelAnimation;
00339         int                         _animsCount;
00340         bool                        _isServerError;
00341 };
```

## 6.13 Inventory.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Inventory
00006 */
00007
00008 #pragma once
00009
00010 #include <iostream>
00011
00012 #define RessourcesNumber 6
00013
00014 namespace Gui {
00015
00020     class Inventory;
00021 };
00022
00023 class Gui::Inventory {
00024
00025     public:
00026
00027         using Ressources = size_t [RessourcesNumber];
00028
00033         Inventory();
00034
00046         Inventory(std::size_t food, std::size_t linemate, std::size_t deraumere, std::size_t sibur,
      std::size_t mendiane, std::size_t phiras, std::size_t thystame);
00047
00052         ~Inventory() = default;
00053
00059         void setFood(std::size_t food);
00060
00066         void setLinemate(std::size_t linemate);
00067
00073         void setDeraumere(std::size_t deraumere);
00074
00080         void setSibur(std::size_t sibur);
00081
00087         void setMendiane(std::size_t mendiane);
00088
00094         void setPhiras(std::size_t phiras);
00095
00101         void setThystame(std::size_t thytsame);
00102
00108         void setRessources(Ressources ressources);
00109
00115         std::size_t getFood(void);
00116
00122         std::size_t getLinemate(void);
00123
00129         std::size_t getDeraumere(void);
00130
00136         std::size_t getSibur(void);
00137
00143         std::size_t getMendiane(void);
00144
00150         std::size_t getPhiras(void);
00151
00157         std::size_t getThystame(void);
00158
00164         Ressources &getRessources(void);
00165
00179         void addResource(std::size_t resource, std::size_t quantity);
00180
00194         void removeResource(std::size_t resource, std::size_t quantity);
00195
00196     private:
00197
00198         std::size_t     _food;
00199         Ressources      _ressources;
00200 };
```

## 6.14 Player.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Player
00006 */
00007
00008 #pragma once
00009
00010 #include "raylib.h"
00011 #include "GameDatas/Inventory.hpp"
00012
00013 namespace Gui {
00014
00019     class Player;
00020 };
00021
00022 class Gui::Player {
00023
00024     public:
00025
00026         enum PlayerState {
00027             IDLE = 2,
00028             BORN = 8,
00029             BROADCAST = 12,
00030             EJECT = 5,
00031             BEING_EJECTED = 15,
00032             EJECTED = 7,
00033             WALK = 6, // or 10
00034             INCANTATION = 0,
00035             LAY_EGG = 7,
00036             DROP = 9,
00037             COLLECT = 9,
00038             DEAD = 1,
00039         };
00040
00048         Player(std::size_t id, const std::string &team, std::pair<std::size_t, std::size_t> position,
      std::size_t orientation, std::size_t level = 1);
00049
00054         ~Player() = default;
00055
00061         void setPosition(std::pair<std::size_t, std::size_t> position);
00062
00068         void setPosition3D(Vector3 position3D);
00069
00075         void setId(std::size_t id);
00076
00082         void setLevel(std::size_t level);
00083
00089         void setOrientation(std::size_t orientation);
00090
00096         void setTeam(const std::string &team);
00097
00103         std::pair<std::size_t, std::size_t> getPosition(void) const;
00104
00110         Vector3 getPosition3D(void) const;
00111
00117         std::size_t getId(void) const;
00118
00124         std::size_t getLevel(void) const;
00125
00131         std::size_t getOrientation(void) const;
00132
00138         std::string getTeam(void) const;
00139
00145         void setState(PlayerState state);
00146
00152         PlayerState getState(void) const;
00153
00159         void setBroadcast(const std::string &broadcast);
00160
00166         std::string getBroadcast() const;
00167
00173         float getRotationFromOrientation() const;
00174
00180         Vector3 getCenterPosition();
00181
00187         void setCurrentFrame(int currentFrame);
00188
00194         int getCurrentFrame() const;
00195
00200         void restartAnimationTimeEllapsed();
00201
00207         clock_t getAnimationTimeEllapsed() const;
00208
```

```
00213         Inventory                               inventory;
00214
00215     private:
00216
00217         std::size_t                             _id;
00218         std::string                             _team;
00219         std::pair<std::size_t, std::size_t>     _position;
00220         Vector3                                 _position3D;
00221         std::size_t                             _orientation;
00222         std::size_t                             _level;
00223         PlayerState                             _state;
00224         std::string                             _broadcast;
00225         int                                     _currentFrame;
00226         clock_t                                 _animationTimeEllapsed;
00227 };
```

## 6.15 Team.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Team
00006 */
00007
00008 #pragma once
00009
00010 #include "raylib.h"
00011 #include "Types.hpp"
00012 #include "GameDatas/Egg.hpp"
00013 #include "GameDatas/Tile.hpp"
00014 #include "GameDatas/Player.hpp"
00015
00016 #include <vector>
00017 #include <memory>
00018
00019 namespace Gui {
00020
00025     class Team;
00026 };
00027
00028 class Gui::Team {
00029
00030     public:
00031
00040         Team(const std::string &name, Model playerModel, Model eggModel, ModelAnimation
      *modelAnimation, Color playerColor);
00041
00046         ~Team();
00047
00053         const std::string &getName() const;
00054
00060         std::vector<Gui::Player> &getPlayers();
00061
00067         std::vector<Gui::Egg> &getEggs();
00068
00074         void setName(const std::string &name);
00075
00081         void addPlayer(const Gui::Player &player);
00082
00088         void addEgg(const Gui::Egg &egg);
00089
00097         bool removePlayer(std::size_t id);
00098
00106         bool removeEgg(std::size_t id);
00107
00114         std::shared_ptr<Gui::Player> getPlayer(std::size_t id);
00115
00121         Model getPlayerModel() const;
00122
00128         ModelAnimation *getPlayerModelAnimation() const;
00129
00135         void setPlayerModelPath(const std::string &playerModelPath);
00136
00143         std::shared_ptr<Gui::Egg> getEgg(std::size_t id);
00144
00150         Model getEggModel() const;
00151
00157         void setEggModelPath(const std::string &eggModelPath);
00158
00166         std::vector<BoundingBox> getPlayerBoundingBoxes(std::pair<size_t, size_t> pos, size_t
      orientation, Vector3 center);
00167
```

```
00174         Vector3 getPlayerPositionIn3DSpace(size_t id, Map<Tile> map);
00175
00183         std::vector<RayCollision> getPlayerModelHitbox(size_t id, Camera camera);
00184
00192         bool isPlayerHit(size_t id, Camera camera);
00193
00198         Color getPlayerColor() const;
00199
00200     private:
00201
00202         ModelAnimation*             _modelAnimation;
00203         int                         _animsCount;
00204         std::string                 _name;
00205         std::vector<Gui::Player>    _players;
00206         Model                       _playerModel;
00207         std::vector<Gui::Egg>       _eggs;
00208         Model                       _eggModel;
00209         Color                       _playerColor;
00210
00220         BoundingBox rotateBoundingBoxByOrientation(BoundingBox bbox, size_t orientation,
      std::pair<size_t, size_t> pos, Vector3 center);
00221 };
```

## 6.16 Tile.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Tile
00006 */
00007
00008 #pragma once
00009
00010 #include "raylib.h"
00011 #include "GameDatas/Inventory.hpp"
00012
00013 #include <vector>
00014
00015 namespace Gui {
00016
00021     class Tile;
00022 };
00023
00024 class Gui::Tile {
00025
00026     public:
00027
00033         Tile(std::pair<std::size_t, std::size_t> position);
00034
00041         Tile(std::pair<std::size_t, std::size_t> position, Inventory inventory);
00042
00047         ~Tile() = default;
00048
00054         void setPosition(std::pair<std::size_t, std::size_t> position);
00055
00061         std::pair<std::size_t, std::size_t> getPosition() const;
00062
00068         Vector3 getPositionIn3DSpace();
00069
00076         std::vector<BoundingBox> getTileBoundingBoxes(Tile tile, Model tileModel);
00077
00085         std::vector<RayCollision> getTileModelHitbox(Tile tile, Camera camera, Model tileModel);
00086
00095         bool isTileHit(Camera camera, Model _tileModel);
00096
00101         Inventory       inventory;
00102
00103     private:
00104
00105         std::pair<std::size_t, std::size_t>     _position;
00106         Vector3                                 _positionIn3DSpace;
00107 };
```

## 6.17 AGUIUpdater.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
```

```
00004 ** File description:
00005 ** AGUIUpdater
00006 */
00007
00008 #pragma once
00009
00010 #include "Network/Network.hpp"
00011 #include "GameDatas/GameData.hpp"
00012 #include "GUIUpdater/IGUIUpdater.hpp"
00013
00014 #include <memory>
00015
00016 namespace Gui {
00017
00021     class AGUIUpdater;
00022 }
00023
00024 class Gui::AGUIUpdater : public Gui::IGUIUpdater {
00025     public:
00026
00033         AGUIUpdater(std::shared_ptr<GameData> gameData, std::shared_ptr<INetwork> network);
00034
00038         ~AGUIUpdater() = default;
00039
00044         void update(const std::string &command, const std::vector<std::string> &data) override = 0;
00045
00046     protected:
00047
00048         std::shared_ptr<GameData>       _gameData;
00049         std::shared_ptr<INetwork>       _network;
00050 };
```

# 6.18 GUIUpdater.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** GUIUpdater
00006 */
00007
00008 #pragma once
00009
00010 #include "GUIUpdater/AGUIUpdater.hpp"
00011
00012 #include <string>
00013 #include <functional>
00014 #include <unordered_map>
00015
00016 namespace Gui {
00017
00023     class GUIUpdater;
00024 }
00025
00026 class Gui::GUIUpdater : public Gui::AGUIUpdater {
00027
00028     public:
00029
00036         GUIUpdater(std::shared_ptr<GameData> gameData, std::shared_ptr<INetwork> network);
00037
00041         ~GUIUpdater() = default;
00042
00049         void update(const std::string &command, const std::vector<std::string> &data);
00050
00051     private:
00052
00053         size_t                          _colorIndex;
00054
00055         std::unordered_map<std::string, std::function<void(std::vector<std::string>)» _updateMap =
00056         {
00057             {"msz", [this](std::vector<std::string> data){updateMapSize(data);}},
00058             {"bct", [this](std::vector<std::string> data){updateMapContent(data);}},
00059             {"tna", [this](std::vector<std::string> data){updateTeamNames(data);}},
00060             {"pnw", [this](std::vector<std::string> data){updateTeamMember(data);}},
00061             {"ppo", [this](std::vector<std::string> data){updatePlayerPosition(data);}},
00062             {"plv", [this](std::vector<std::string> data){updatePlayerLevel(data);}},
00063             {"pin", [this](std::vector<std::string> data){updatePlayerInventory(data);}},
00064             {"pex", [this](std::vector<std::string> data){updatePlayerExpulsion(data);}},
00065             {"pbc", [this](std::vector<std::string> data){updatePlayerBroadcast(data);}},
00066             {"pic", [this](std::vector<std::string> data){updatePlayerStartIncantation(data);}},
00067             {"pie", [this](std::vector<std::string> data){updatePlayerEndIncantation(data);}},
00068             {"pfk", [this](std::vector<std::string> data){updatePlayerEggLaying(data);}},
00069             {"pdr", [this](std::vector<std::string> data){updatePlayerRessourceDropping(data);}},
```

```
00070             {"pgt", [this](std::vector<std::string> data){updatePlayerRessourceCollecting(data);}},
00071             {"pdi", [this](std::vector<std::string> data){updatePlayerDeath(data);}},
00072             {"enw", [this](std::vector<std::string> data){updateEggLaidByPlayer(data);}},
00073             {"ebo", [this](std::vector<std::string> data){updatePlayerBorn(data);}},
00074             {"edi", [this](std::vector<std::string> data){updateEggDeath(data);}},
00075             {"sgt", [this](std::vector<std::string> data){updateTimeUnitRequest(data);}},
00076             {"sst", [this](std::vector<std::string> data){updateTimeUnitModification(data);}},
00077             {"seg", [this](std::vector<std::string> data){updateEndOfGame(data);}},
00078             {"smg", [this](std::vector<std::string> data){updateMessageFromServer(data);}},
00079             {"suc", [this](std::vector<std::string> data){updateUnknownMessage(data);}},
00080             {"sbp", [this](std::vector<std::string> data){updateCommandParameter(data);}},
00081         }; // The map of commands to update the GUI GameData.
00082
00088         void updateMapSize(const std::vector<std::string> &data);
00089
00095         void updateMapContent(const std::vector<std::string> &data);
00096
00102         void updateTeamNames(const std::vector<std::string> &data);
00103
00109         void updateTeamMember(const std::vector<std::string> &data);
00110
00116         void updatePlayerPosition(const std::vector<std::string> &data);
00117
00123         void updatePlayerLevel(const std::vector<std::string> &data);
00124
00130         void updatePlayerInventory(const std::vector<std::string> &data);
00131
00137         void updatePlayerExpulsion(const std::vector<std::string> &data);
00138
00144         void updatePlayerBroadcast(const std::vector<std::string> &data);
00145
00151         void updatePlayerStartIncantation(const std::vector<std::string> &data);
00152
00158         void updatePlayerEndIncantation(const std::vector<std::string> &data);
00159
00165         void updatePlayerEggLaying(const std::vector<std::string> &data);
00166
00172         void updatePlayerRessourceDropping(const std::vector<std::string> &data);
00173
00179         void updatePlayerRessourceCollecting(const std::vector<std::string> &data);
00180
00186         void updatePlayerDeath(const std::vector<std::string> &data);
00187
00193         void updateEggLaidByPlayer(const std::vector<std::string> &data);
00194
00200         void updatePlayerBorn(const std::vector<std::string> &data);
00201
00207         void updateEggDeath(const std::vector<std::string> &data);
00208
00214         void updateTimeUnitRequest(const std::vector<std::string> &data);
00215
00221         void updateTimeUnitModification(const std::vector<std::string> &data);
00222
00228         void updateEndOfGame(const std::vector<std::string> &data);
00229
00235         void updateMessageFromServer(const std::vector<std::string> &data);
00236
00242         void updateUnknownMessage(const std::vector<std::string> &data);
00243
00249         void updateCommandParameter(const std::vector<std::string> &data);
00250
00255         void increaseColorIndex();
00256 };
```

## 6.19 IGUIUpdater.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** IGUIUpdater
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <vector>
00012
00013 namespace Gui {
00014
00019     class IGUIUpdater;
00020 }
00021
```

```
00022 class Gui::IGUIUpdater {
00023     public:
00027         virtual ~IGUIUpdater() = default;
00028
00033         virtual void update(const std::string &command, const std::vector<std::string> &data) = 0;
00034 };
```

## 6.20 AHud.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** AHud
00006 */
00007
00008 #pragma once
00009
00010 #include "Hud/IHud.hpp"
00011 #include "GameDatas/GameData.hpp"
00012
00013 namespace Gui {
00014
00019     class AHud;
00020 };
00021
00022 class Gui::AHud : public Gui::IHud {
00023
00024     public:
00025
00030         ~AHud() = default;
00031
00036         virtual void display() = 0;
00037
00043         void setPlayer(std::shared_ptr<Player> player);
00044
00050         void setTile(std::shared_ptr<Tile> tile);
00051
00057         TypeScene getType() const;
00058
00064         void setType(TypeScene type);
00065
00066     protected:
00067
00068         TypeScene                       _typeScene;
00069         std::shared_ptr<GameData>       _gameData;
00070         std::shared_ptr<Player>         _player;
00071         std::shared_ptr<Tile>           _tile;
00072 };
```

## 6.21 HudEnd.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** HudEnd
00006 */
00007
00008 #pragma once
00009
00010 #include "Hud/AHud.hpp"
00011
00012 namespace Gui {
00013
00018     class HudEnd;
00019 };
00020
00021 class Gui::HudEnd : public Gui::AHud {
00022
00023     public:
00024
00030         HudEnd(std::shared_ptr<GameData> gameData);
00031
00036         ~HudEnd() = default;
00037
00042         void display();
00043
00044     private:
```

```
00045
00046             Texture2D   _texture;
00047             Font        _font;
00048 };
```

## 6.22 HudGame.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** HudGame
00006 */
00007
00008 #pragma once
00009
00010 #include "Hud/AHud.hpp"
00011
00012 #define HUD_GAME_POS Vector2{0, 340}
00013 #define HUD_GAME_TEXT_POS Vector2{25, 420}
00014 #define HUD_GAME_TEXT_TITLE_POS Vector2{50, 380}
00015 #define HUD_GAME_TEXT_MARGING 30
00016
00017 namespace Gui {
00018
00023     class HudGame;
00024 };
00025
00026 class Gui::HudGame : public Gui::AHud {
00027
00028     public:
00029
00035         HudGame(std::shared_ptr<GameData> gameData);
00036
00041         ~HudGame() = default;
00042
00047         void display();
00048
00049     private:
00050
00051         Texture2D   _texture;
00052         Font        _font;
00053         Texture2D   _playerTexture;
00054 };
```

## 6.23 HudHelp.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** HudHelp
00006 */
00007
00008 #pragma once
00009
00010 #include "Hud/AHud.hpp"
00011
00012 namespace Gui {
00013
00018     class HudHelp;
00019 };
00020
00021 class Gui::HudHelp : public Gui::AHud {
00022
00023     public:
00024
00030         HudHelp(std::shared_ptr<GameData> gameData);
00031
00036         ~HudHelp() = default;
00037
00042         void display();
00043
00044     private:
00045
00046         Texture2D       _texture;           // Texture for Hud Background.
00047         Texture2D       _textureKeys;       // Texture for Hud keys.
00048         Font            _font;              // Font for Hud's texts.
00049         clock_t         _helpMenuClock;     // Clock to display the help menu.
```

```
00050
00056         void displayHelpMenuControls();
00057 };
```

## 6.24 HudPlayer.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** HudPlayer
00006 */
00007
00008 #pragma once
00009
00010 #include "Hud/AHud.hpp"
00011
00012 #define HUD_PLAYER_POS Vector2{0, 340}
00013 #define HUD_PLAYER_TEXT_POS Vector2{55, 420}
00014 #define HUD_PLAYER_TEXT_TITLE_POS Vector2{80, 380}
00015 #define HUD_PLAYER_TEXT_MARGING 30
00016 #define HUD_PLAYER_ICONS_MARGING -32
00017 #define HUD_PLAYER_TITLE_ICON_MARGING Vector2{45, 35}
00018
00019 namespace Gui {
00020
00025     class HudPlayer;
00026 };
00027
00028 class Gui::HudPlayer : public Gui::AHud {
00029
00030     public:
00031
00037         HudPlayer(std::shared_ptr<GameData> gameData);
00038
00043         ~HudPlayer() = default;
00044
00049         void display();
00050
00051     private:
00052
00053         Texture2D   _texture;
00054         Font        _font;
00055
00056         Texture2D   _food;
00057         Texture2D   _linemate;
00058         Texture2D   _deraumere;
00059         Texture2D   _mendiane;
00060         Texture2D   _phiras;
00061         Texture2D   _sibur;
00062         Texture2D   _thystame;
00063         Texture2D   _playerTexture;
00064 };
```

## 6.25 HudTile.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** HudTile
00006 */
00007
00008 #pragma once
00009
00010 #include "Hud/AHud.hpp"
00011
00012 #define HUD_TILE_POS Vector2{0, 340}
00013 #define HUD_TILE_TEXT_POS Vector2{55, 420}
00014 #define HUD_TILE_TEXT_TITLE_POS Vector2{60, 380}
00015 #define HUD_TILE_TEXT_MARGING 30
00016 #define HUD_TILE_ICONS_MARGING -32
00017 #define HUD_TILE_TITLE_ICON_MARGING Vector2{45, 40}
00018
00019 namespace Gui {
00020
00025     class HudTile;
00026 };
00027
```

```
00028 class Gui::HudTile : public Gui::AHud {
00029
00030     public:
00031
00037         HudTile(std::shared_ptr<GameData> gameData);
00038
00043         ~HudTile() = default;
00044
00049         void display();
00050
00055         void displayNbPlayers();
00056
00061         void displayNbEggs();
00062
00063     private:
00064
00065         Texture2D   _texture;
00066         Font        _font;
00067
00068         Texture2D   _food;
00069         Texture2D   _linemate;
00070         Texture2D   _deraumere;
00071         Texture2D   _mendiane;
00072         Texture2D   _phiras;
00073         Texture2D   _sibur;
00074         Texture2D   _thystame;
00075         Texture2D   _tileTexture;
00076         Texture2D   _playerTexture;
00077         Texture2D   _eggTexture;
00078 };
```

## 6.26 IHud.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** IHud
00006 */
00007
00008 #pragma once
00009
00010 #include "GameDatas/Player.hpp"
00011 #include "GameDatas/Tile.hpp"
00012
00013 #include <memory>
00014
00015 namespace Gui {
00016
00021     class IHud;
00022 };
00023
00024 class Gui::IHud {
00025
00026     public:
00027
00032         enum TypeScene {
00033             GAME,
00034             POV_PLAYER,
00035             END_GAME,
00036             TILE,
00037             HELP_TEXT,
00038             HELP_MENU,
00039             END
00040         };
00041
00046         virtual ~IHud() = default;
00047
00052         virtual void display() = 0;
00053
00059         virtual void setPlayer(std::shared_ptr<Player> player) = 0;
00060
00066         virtual void setTile(std::shared_ptr<Tile> tile) = 0;
00067
00073         virtual TypeScene getType() const = 0;
00074
00080         virtual void setType(TypeScene type) = 0;
00081 };
```

## 6.27 ANetwork.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** ANetwork
00006 */
00007
00008 #pragma once
00009
00010 #include "Network/INetwork.hpp"
00011
00012 #define MIN_PORT 1
00013 #define MAX_PORT 65535
00014
00015 namespace Gui {
00016
00021     class ANetwork;
00022 };
00023
00024 class Gui::ANetwork : public Gui::INetwork {
00025     public:
00026
00033         ANetwork(int port, const std::string &hostName);
00034
00039         ~ANetwork() = default;
00040
00047         void setPort(int port) final;
00048
00054         void setHostName(const std::string &hostName) final;
00055
00061         int getPort() const final;
00062
00068         std::string getHostName() const final;
00069
00075         virtual void connectToServer() = 0;
00076
00082         virtual BufferState listenServer() = 0;
00083
00089         virtual void sendMessageServer(const std::string& message) = 0;
00090
00098         std::string getBuffer();
00099
00100     protected:
00101         int             _port;
00102         std::string     _hostName;
00103         std::string     _buffer;
00104 };
```

## 6.28 INetwork.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** INetwork
00006 */
00007
00008 #pragma once
00009
00010 #include "Error/Error.hpp"
00011
00012 #include <string>
00013
00014 namespace Gui {
00015
00020     class INetwork;
00021 };
00022
00023 class Gui::INetwork {
00024
00025     public:
00026
00027         enum BufferState {
00028             NONE,
00029             READY,
00030             SERVER_ERROR
00031         };
00032
00037         virtual ~INetwork() = default;
00038
00045         virtual void setPort(int port) = 0;
```

```
00046
00052         virtual void setHostName(const std::string &hostName) = 0;
00053
00059         virtual int getPort() const = 0;
00060
00066         virtual std::string getHostName() const = 0;
00067
00073         virtual void connectToServer() = 0;
00074
00080         virtual BufferState listenServer() = 0;
00081
00087         virtual void sendMessageServer(const std::string &message) = 0;
00088
00096         virtual std::string getBuffer() = 0;
00097 };
```

## 6.29 Network.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Network
00006 */
00007
00008 #pragma once
00009
00010 #include "Network/ANetwork.hpp"
00011
00012 #include <arpa/inet.h>
00013 #include <sys/socket.h>
00014 #include <netinet/in.h>
00015
00016 namespace Gui {
00017
00022     class Network;
00023 };
00024
00025 class Gui::Network : public Gui::ANetwork {
00026
00027     public:
00028
00035         Network(int port, const std::string &hostName);
00036
00041         ~Network();
00042
00048         void connectToServer();
00049
00055         BufferState listenServer();
00056
00062         void sendMessageServer(const std::string& message);
00063
00064     private:
00065
00070         void selectServer();
00071
00077         BufferState readInfoServer();
00078
00079         int             _serverFd;
00080         fd_set          _writeFd;
00081         fd_set          _readFd;
00082         bool            _isConnected;
00083 };
```

## 6.30 IServerParser.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** IServerParser
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <vector>
00012
00013 namespace Gui {
```

```
00014
00018     class IServerParser;
00019 }
00020
00021 class Gui::IServerParser {
00022
00023     public:
00024
00028         virtual ~IServerParser() = default;
00029
00036         virtual std::vector<std::string> parse(const std::string& command) = 0;
00037 };
```

## 6.31 ParseCommandLine.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** ParseCommandLine
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011
00012 #define GUI_USAGE "USAGE:\t./zappy_gui -p port -h machine"
00013
00014 namespace Gui {
00015
00020     class ParseCommandLine;
00021 };
00022
00023 class Gui::ParseCommandLine {
00024
00025     public:
00026
00033         ParseCommandLine(int argc, char **argv);
00034
00039         ~ParseCommandLine() = default;
00040
00047         void parseFlags(int argc, char **argv);
00048
00054         int getPort(void);
00055
00061         std::string getHostName(void);
00062
00063     private:
00064
00065         int             _port;
00066         std::string     _hostName;
00067 };
```

## 6.32 ServerParser.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Parse server command
00006 */
00007
00008 #pragma once
00009
00010 #include "Error/Error.hpp"
00011 #include "Parsing/IServerParser.hpp"
00012
00013 #include <functional>
00014 #include <unordered_map>
00015
00016 namespace Gui {
00017
00022     class ServerParser;
00023 };
00024
00025 class Gui::ServerParser : public Gui::IServerParser {
00026
00027     public:
00028
```

```
00033         ServerParser() = default;
00034
00039         ~ServerParser() = default;
00040
00047         std::vector<std::string> parse(const std::string& command);
00048
00049     private:
00050
00055         enum ParseType {
00056             INT,
00057             STRING,
00058             MESSAGE,
00059             LIST_INT
00060         };
00061
00066         std::unordered_map<std::string, std::vector<ParseType» _typesCommand =
00067         {
00068             {"msz", std::vector<ParseType>{INT, INT}},
00069             {"bct", std::vector<ParseType>{INT, INT, INT, INT, INT, INT, INT, INT, INT}},
00070             {"tna", std::vector<ParseType>{STRING}},
00071             {"pnw", std::vector<ParseType>{INT, INT, INT, INT, INT, STRING}},
00072             {"ppo", std::vector<ParseType>{INT, INT, INT, INT}},
00073             {"plv", std::vector<ParseType>{INT, INT}},
00074             {"pin", std::vector<ParseType>{INT, INT, INT, INT, INT, INT, INT, INT, INT, INT}},
00075             {"pex", std::vector<ParseType>{INT}},
00076             {"pbc", std::vector<ParseType>{INT, MESSAGE}},
00077             {"pic", std::vector<ParseType>{INT, INT, INT, LIST_INT}},
00078             {"pie", std::vector<ParseType>{INT, INT, INT}},
00079             {"pfk", std::vector<ParseType>{INT}},
00080             {"pdr", std::vector<ParseType>{INT, INT}},
00081             {"pgt", std::vector<ParseType>{INT, INT}},
00082             {"pdi", std::vector<ParseType>{INT}},
00083             {"enw", std::vector<ParseType>{INT, INT, INT, INT}},
00084             {"ebo", std::vector<ParseType>{INT}},
00085             {"edi", std::vector<ParseType>{INT}},
00086             {"sgt", std::vector<ParseType>{INT}},
00087             {"sst", std::vector<ParseType>{INT}},
00088             {"seg", std::vector<ParseType>{STRING}},
00089             {"smg", std::vector<ParseType>{MESSAGE}},
00090             {"suc", std::vector<ParseType>{}},
00091             {"sbp", std::vector<ParseType>{}}
00092         };
00093
00101         std::vector<std::string> parseCommand(const std::string& command, std::vector<ParseType>
      types);
00102
00110         std::vector<std::string> parseInt(std::istringstream& stream, std::vector<std::string>
      arguments);
00111
00119         std::vector<std::string> parseString(std::istringstream& stream, std::vector<std::string>
      arguments);
00120
00129         std::vector<std::string> parseMessage(std::istringstream& stream, std::vector<std::string>
      arguments, std::string commandName);
00130
00139         std::vector<std::string> parseListInt(std::istringstream& stream, std::vector<std::string>
      arguments, std::string commandName);
00140 };
```

## 6.33 Decoration.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Decoration
00006 */
00007
00008 #pragma once
00009
00010 #include "raylib.h"
00011 #include "Types.hpp"
00012 #include "Assets.hpp"
00013
00014 #include <vector>
00015 #include <iostream>
00016
00017 namespace Gui {
00018
00024     class Decoration;
00025 }
00026
00027 class Gui::Decoration {
```

```
00028
00029     public:
00030
00035         Decoration();
00036
00041         ~Decoration() = default;
00042
00051         void display(std::pair<std::size_t, std::size_t> mapSize, size_t renderDistance,
      std::pair<std::size_t, std::size_t> camPos, std::vector<Vector2> displayPos);
00052
00061         Map<bool> getGenerationItem(std::size_t ratio);
00062
00063     private:
00064
00065         Model                                   _treeModel;
00066         Map<bool>                               _mapTree;
00067
00068         std::pair<std::size_t, std::size_t>     _mapSize;
00069
00077         void displayTree(size_t i, size_t j, Vector3 posTile);
00078
00087         bool isInArrayPlayerVision(std::pair<size_t, size_t> pos, std::vector<Vector2>
      _playerVisionPositions);
00088 };
```

## 6.34 Render.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Render
00006 */
00007
00008 #pragma once
00009
00010
00011 #include "raylib.h"
00012 #include "Config.hpp"
00013 #include "Hud/HudGame.hpp"
00014 #include "Hud/HudTile.hpp"
00015 #include "Hud/HudPlayer.hpp"
00016 #include "Hud/HudHelp.hpp"
00017 #include "Hud/HudEnd.hpp"
00018 #include "Render/Decoration.hpp"
00019 #include "Render/UserCamera.hpp"
00020 #include "GameDatas/GameData.hpp"
00021
00022 #include <functional>
00023 #include <unordered_map>
00024
00025 namespace Gui {
00026
00031     class Render;
00032 };
00033
00034 class Gui::Render {
00035
00036     public:
00037
00042         Render(std::shared_ptr<GameData> gameData);
00043
00048         ~Render();
00049
00056         bool isOpen();
00057
00062         void draw();
00063
00069         std::shared_ptr<Camera> getCamera();
00070
00076         void setIsDebug(bool isDebug);
00077
00084         bool getIsDebug(void);
00085
00091         void setCameraType(Gui::UserCamera::CameraType type);
00092
00098         Gui::UserCamera::CameraType getCameraType() const;
00099
00105         void setCameraPlayerPov(std::size_t id);
00106
00112         std::size_t getCameraPlayerPov() const;
00113
00119         void setCameraTile(std::pair<std::size_t, std::size_t> pos);
```

```
00120
00126         std::pair<std::size_t, std::size_t> getCameraTile() const;
00127
00132         Model getTileModel() const;
00133
00139         void setRenderDistance(size_t renderDistance);
00140
00145         size_t getRenderDistance() const;
00146
00154         bool isCameraInPlayerPov() const;
00155
00162         void changePlayerPOV(size_t playerId);
00163
00169         void setPlayerPov(size_t playerId);
00170
00176         void changePOVToFirstPerson(size_t id);
00177
00183         void changePOVToSecondPerson(size_t id);
00184
00190         void changePOVToThirdPerson(size_t id);
00191
00197         size_t getTimeUnit() const;
00198
00204         void setTimeUnit(size_t timeUnit);
00205
00213         void setPlayerVision(bool isPlayerVision);
00214
00221         bool getPlayerVision() const;
00222
00228         void setHelpMenu(bool isHelpMenu);
00229
00236         bool getHelpMenu() const;
00237
00242         void drawEnd() const;
00243
00244     private:
00245
00246         UserCamera                                  _camera;
00247         bool                                        _isDebug;
00248         std::shared_ptr<GameData>                   _gameData;
00249         std::shared_ptr<Decoration>                 _decoration;
00250         std::vector<std::shared_ptr<Gui::IHud»     _hudList;
00251         size_t                                      _renderDistance;
00252         bool                                        _isHelpMenu;
00253
00254         Model                                       _tileModel;
00255         Model                                       _foodModel;
00256         Model                                       _linemateModel;
00257         Model                                       _mendianeModel;
00258         Model                                       _phirasModel;
00259         Model                                       _siburModel;
00260         Model                                       _thystameModel;
00261         Model                                       _deraumereModel;
00262         Texture2D                                   _cursorTexture;
00263         std::vector<Vector2>                        _playerVisionPositions;
00264         bool                                        _endHudSet;
00265
00270         void LoadModels();
00271
00276         void displayHUD();
00277
00282         void displayDebug();
00283
00288         void displayPlayers();
00289
00297         void displayPlayerLevel(Player &player, Vector3 position, Team &team);
00298
00305         void displayPlayerBroadcast(Team &team, Player &player);
00306
00311         void displayMap();
00312
00317         void displayTile(Tile tile);
00318
00324         void displayEggs(Tile tile) const;
00325
00331         void displayFood(Tile tile) const;
00332
00338         void displayResources(Tile tile) const;
00339
00345         void displayLinemate(Tile tile) const;
00346
00352         void displayMendiane(Tile tile) const;
00353
00359         void displayPhiras(Tile tile) const;
00360
00366         void displaySibur(Tile tile) const;
00367
```

```
00373         void displayThystame(Tile tile) const;
00374
00380         void displayDeraumere(Tile tile) const;
00381
00388         bool displayAnimations(Team &team, Player &player);
00389
00398         ModelAnimation displayWalkAnimation(Team &team, Player &player, ModelAnimation anim);
00399
00404         void displayCursor();
00405
00411         std::pair<std::size_t, std::size_t> getCameraTile();
00412
00418         std::vector<Vector2> getPositionsInPlayerVision(size_t playerId);
00419
00427         bool isInArrayPlayerVision(std::pair<size_t, size_t> pos);
00428
00436         std::vector<Vector2> getLineOfVision(Vector2 pos, size_t sizeOfHalf, size_t orientation);
00437
00445         std::vector<Vector2> addVisionPosition(std::vector<Vector2> vision, std::vector<Vector2> pos);
00446
00451         void displayHelpMenu(std::shared_ptr<IHud> hud);
00452
00458         void displayHelpMenuControls(Vector2 position);
00459 };
```

# 6.35 UserCamera.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy GUI
00004 ** File description:
00005 ** Camera
00006 */
00007
00008 #pragma once
00009
00010 #include "raylib.h"
00011
00012 #include <memory>
00013
00014 namespace Gui {
00015
00020     class UserCamera;
00021 };
00022
00023 class Gui::UserCamera {
00024
00025     public:
00026
00027         enum CameraType {
00028             FREE,
00029             FIRST_PERSON,
00030             SECOND_PERSON,
00031             THIRD_PERSON,
00032             FREE_TILE
00033         };
00034
00039         UserCamera();
00040
00045         ~UserCamera() = default;
00046
00052         void setPosition(Vector3 position);
00053
00059         void setTarget(Vector3 target);
00060
00066         void setUp(Vector3 up);
00067
00073         void setFovy(float fovy);
00074
00080         Vector3 getPosition(void) const;
00081
00087         Vector3 getTarget(void) const;
00088
00094         Vector3 getUp(void) const;
00095
00101         float getFovy(void) const;
00102
00108         std::shared_ptr<Camera> getCamera();
00109
00115         void setType(CameraType type);
00116
00122         CameraType getType() const;
00123
```

```
00129         void setPlayerId(size_t playerId);
00130
00136         int getPlayerId() const;
00137
00143         void setTilePos(std::pair<std::size_t, std::size_t> pos);
00144
00150         std::pair<std::size_t, std::size_t> getTilePos() const;
00151
00159         bool isPlayerPov() const;
00160
00167         bool isPlayerVision() const;
00168
00174         void setPlayerVision(bool isPlayerVision);
00175
00176     private:
00177
00178         std::shared_ptr<Camera>                 _camera;
00179         CameraType                              _type;
00180         int                                     _playerId;
00181         std::pair<std::size_t, std::size_t>     _tilePos;
00182         bool                                    _isPlayerVision;
00183 };
```

## 6.36 Types.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy
00004 ** File description:
00005 ** Types
00006 */
00007
00008 #pragma once
00009
00010 #include <vector>
00011
00017 template<typename T>
00018 using Map = std::vector<std::vector<T»;
```

# Index