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.

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.

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, const std::string &playerModelPath, const std::string &eggModelPath, 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.

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,
             const std::string & playerModelPath,
             const std::string & eggModelPath,
             Color playerColor )
```

Add a team to the game.

Parameters

name	Name of the team.
playerModelPath	Path to the asset of the team for players.
eggModelPath	Path to the asset of the team for eggs.
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 getServerTick()

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

Get the Server Tick object.

Returns

std::size_t - Server Tick.

5.11.2.16 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.

5.11.2.17 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.

5.11.2.18 getTeams()

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

Get the Teams object.

Returns

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

5.11.2.19 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.

5.11.2.20 getTimeUnitFromServer()

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

Get the Time Unit From Server object.

Returns

true - The time unit has changed.

5.11.2.21 removeServerEgg()

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

Remove an egg from the server ones.

Parameters

id	Id of the egg.

5.11.2.22 restartLastTick()

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

Restart the last tick clock.

5.11.2.23 restartLastTickMctCommand()

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

Restart the last tick mct command clock.

5.11.2.24 setEndMessage()

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

Set the End Message object.

Parameters

endMessage	End message of the game.

5.11.2.25 setIsEndGame()

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

Set the IsEnd Game object.

Parameters

isEndGame	EndGame state.

5.11.2.26 setLastError()

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

Set the Last Error object.

Parameters

error	Error message.

5.11.2.27 setMap()

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

Set the Map object.

Parameters

map	Map of the game.

5.11.2.28 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.

5.11.2.29 setNbBCTCommandReceived()

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

Set the number of bct command received.

Parameters

nb	Number of bct command received.

5.11.2.30 setServerTick()

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

Set the Server Tick object.

Parameters

tick	Tick of the server.

5.11.2.31 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.

5.11.2.32 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

player↩ Id	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, const std::string &playerModelPath, const std::string &eggModelPath, 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,
            const std::string & playerModelPath,
            const std::string & eggModelPath,
            Color playerColor )
```

Construct a new Team object.

Parameters

name	Name of the team.
playerModelPath	Path to the team model asset for players.
eggSkinPath	Path to the skin of the team.
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

player↩ Id	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

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Assets
*/

#pragma once

#include "Config.hpp"

#include <vector>
#include <string>

#define PATH_ASSETS                     "gui/assets/"

#define PATH_RESOURCES                  "resources/"
#define PATH_PLAYER                     "player/"
#define PATH_TILE                       "tile/"
#define PATH_DECORATION                 "decoration/"
#define PATH_HUD                        "hud/"
#define PATH_HELP                       "help/"
#define PATH_END                        "end/"

#define PNG_CURSOR                      PATH_ASSETS PATH_HUD "cursor.png"
#define PNG_END_HUD                     PATH_ASSETS PATH_END "end_hud.png"
#define PNG_HUD                         PATH_ASSETS PATH_HUD "hud.png"
#define PNG_FOOD                        PATH_ASSETS PATH_HUD "food.png"
#define PNG_LINEMATE                    PATH_ASSETS PATH_HUD "linemate.png"
#define PNG_DERAUMERE                   PATH_ASSETS PATH_HUD "deraumere.png"
#define PNG_MENDIANE                    PATH_ASSETS PATH_HUD "mendiane.png"
#define PNG_PHIRAS                      PATH_ASSETS PATH_HUD "phiras.png"
#define PNG_SIBUR                       PATH_ASSETS PATH_HUD "sibur.png"
#define PNG_THYSTAME                    PATH_ASSETS PATH_HUD "thystame.png"
#define PNG_PLAYER                      PATH_ASSETS PATH_HUD "player.png"
#define PNG_TILE                        PATH_ASSETS PATH_HUD "tile.png"
#define PNG_EGG                         PATH_ASSETS PATH_HUD "egg.png"
#define FONT_HUD                        PATH_ASSETS PATH_HUD "SimplyMono-Bold.ttf"

#define PNG_HELP                        PATH_ASSETS PATH_HELP "help_menu.png"
#define PNG_HELP_KEYS                   PATH_ASSETS PATH_HELP "keys.png"
#define PNG_HUD                         PATH_ASSETS PATH_HUD "hud.png"
#define PNG_FOOD                        PATH_ASSETS PATH_HUD "food.png"
#define PNG_LINEMATE                    PATH_ASSETS PATH_HUD "linemate.png"
#define PNG_DERAUMERE                   PATH_ASSETS PATH_HUD "deraumere.png"
#define PNG_MENDIANE                    PATH_ASSETS PATH_HUD "mendiane.png"
#define PNG_PHIRAS                      PATH_ASSETS PATH_HUD "phiras.png"
#define PNG_SIBUR                       PATH_ASSETS PATH_HUD "sibur.png"
#define PNG_THYSTAME                    PATH_ASSETS PATH_HUD "thystame.png"
#define PNG_PLAYER                      PATH_ASSETS PATH_HUD "player.png"
#define PNG_TILE                        PATH_ASSETS PATH_HUD "tile.png"
#define FONT_HUD                        PATH_ASSETS PATH_HUD "SimplyMono-Bold.ttf"

#define MODEL_TILE                      PATH_ASSETS PATH_TILE "tile.glb"
#define MODEL_FOOD                      PATH_ASSETS PATH_RESOURCES "food.glb"
#define MODEL_LINEMATE                  PATH_ASSETS PATH_RESOURCES "linemate.glb"
#define MODEL_MENDIANE                  PATH_ASSETS PATH_RESOURCES "mendiane.glb"
#define MODEL_PHIRAS                    PATH_ASSETS PATH_RESOURCES "phiras.glb"
```

```
#define MODEL_SIBUR                     PATH_ASSETS PATH_RESOURCES "sibur.glb"
#define MODEL_THYSTAME                  PATH_ASSETS PATH_RESOURCES "thystame.glb"
#define MODEL_DERAUMERE                 PATH_ASSETS PATH_RESOURCES "deraumere.glb"
#define MODEL_PLAYER                    PATH_ASSETS PATH_PLAYER "player.glb"
#define MODEL_EGG                       PATH_ASSETS PATH_PLAYER "egg.glb"
#define MODEL_TREE                      PATH_ASSETS PATH_DECORATION "tree.glb"
#define MODEL_LANTERN                   PATH_ASSETS PATH_DECORATION "lantern.glb"

#define SCALE_FOOD                      (Vector3){1, 0.5, 1}
#define SCALE_LINEMATE                  (Vector3){0.1, 0.1, 0.1}
#define SCALE_MENDIANE                  (Vector3){0.1, 0.1, 0.1}
#define SCALE_PHIRAS                    (Vector3){0.001, 0.001, 0.005}
#define SCALE_SIBUR                     (Vector3){0.01, 0.01, 0.01}
#define SCALE_THYSTAME                  (Vector3){2, 2, 2}
#define SCALE_DERAUMERE                 (Vector3){0.5, 0.5, 0.5}
#define SCALE_PLAYER                    (Vector3){0.5, 0.5, 0.5}
#define SCALE_EGG                       (Vector3){0.5, 0.5, 0.5}
#define SCALE_TREE                      (Vector3){1, 1, 1}
#define SCALE_LANTERN                   (Vector3){1, 1, 1}

#define ROTATION_ANGLE_FOOD             45
#define ROTATION_ANGLE_LINEMATE         270
#define ROTATION_ANGLE_MENDIANE         0
#define ROTATION_ANGLE_PHIRAS           270
#define ROTATION_ANGLE_SIBUR            270
#define ROTATION_ANGLE_THYSTAME         270
#define ROTATION_ANGLE_DERAUMERE        90
#define ROTATION_ANGLE_PLAYER           0
#define ROTATION_ANGLE_EGG              270
#define ROTATION_ANGLE_TREE             0
#define ROTATION_ANGLE_LANTERN          0

#define ROTATION_AXIS_FOOD              (Vector3){0, 1, 0}
#define ROTATION_AXIS_LINEMATE          (Vector3){1, 0, 0}
#define ROTATION_AXIS_MENDIANE          (Vector3){1, 0, 0}
#define ROTATION_AXIS_PHIRAS            (Vector3){1, 0, 0}
#define ROTATION_AXIS_SIBUR             (Vector3){1, 0, 0}
#define ROTATION_AXIS_THYSTAME          (Vector3){1, 0, 0}
#define ROTATION_AXIS_DERAUMERE         (Vector3){1, 0, 0}
#define ROTATION_AXIS_PLAYER            (Vector3){0, 1, 0}
#define ROTATION_AXIS_EGG               (Vector3){1, 0, 0}
#define ROTATION_AXIS_TREE              (Vector3){1, 0, 0}
#define ROTATION_AXIS_LANTERN           (Vector3){1, 0, 0}

#define POS_FOOD                        (Vector3){0.5, -0.1, 1.5}
#define POS_LINEMATE                    (Vector3){1, -0.3, -0.5}
#define POS_MENDIANE                    (Vector3){2, -0.25, -0.5}
#define POS_PHIRAS                      (Vector3){0.5, -0.3, -1.5}
#define POS_SIBUR                       (Vector3){1.5, -0.3, -1.5}
#define POS_THYSTAME                    (Vector3){1, 0, -2}
#define POS_DERAUMERE                   (Vector3){2, -0.3, -2}
#define POS_PLAYER                      (Vector3){0, -0.25, 0}
#define POS_EGG                         (Vector3){0.5, 0, 0.5}
#define POS_TREE                        (Vector3){2, -0.3, 2}
#define POS_LANTERN                     (Vector3){1, -0.3, 2}
#define POS_Y_DELIMITATION              -0.27f

#define PLAYER_TEXT_SIZE                40
#define PLAYER_TEXT_SIZE_RATIO          1.5f

#define HELP_BACKGROUND_SCALE           3.5f
#define HELP_BACKGROUND_TEXTURE_SIZE    (Vector2){400 * HELP_BACKGROUND_SCALE, 200 *
HELP_BACKGROUND_SCALE}
#define HELP_BACKGROUND_POSITION        (Vector2){WINDOW_WIDTH / 2 - HELP_BACKGROUND_TEXTURE_SIZE.x /
2, WINDOW_HEIGHT / 2 - HELP_BACKGROUND_TEXTURE_SIZE.y / 2}
#define HELP_TEXT_POSITION_LEFT         (Vector2){HELP_BACKGROUND_POSITION.x + 270,
HELP_BACKGROUND_POSITION.y + 180}
#define HELP_TEXT_POSITION_RIGHT        (Vector2){HELP_TEXT_POSITION_LEFT.x + 610,
HELP_TEXT_POSITION_LEFT.y}
#define HELP_TEXT_SPACING               30
#define HELP_TITLE_OFFSET               90

#define END_HUD_TEXT_SIZE               40
#define END_HUD_ADVICE_TEXT_SIZE        30
#define END_HUD_ADVICE_OFFSET           150

static std::vector<std::string> globalControlsTexts = {
    "Move the camera in the space",
    "Move the camera up",
    "Move the camera down",
    "Switch on/off the debug display",
    "Reset the HUD from tile view to game view",
    "Switch on/off the help HELP",
    "Increase the render distance",
    "Decrease the render distance",
    "Increase the time unit",
    "Decrease the time unit",
    "Select the player point of view",
    "Select the tile to display its HUD",
};

static std::vector<std::string> playerControlsTexts = {
    "Change the player point of view",
    "Get off the player point of view",
    "Switch to the next player",
    "Switch to the previous player",
    "Switch on/off the player vision",
};
```

6.2 Colors.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** Colors
*/

#pragma once

#include "raylib.h"

#include <vector>

#define STR_BLUE "\033[0;34m"
#define STR_GREEN "\033[0;32m"
#define STR_RED "\033[0;31m"
#define STR_YELLOW "\033[0;33m"
#define STR_VIOLET "\033[0;35m"
#define STR_CYAN "\033[0;36m"
#define STR_RESET "\033[0m"

static std::vector<Color> playerColors = {
    PINK,
    LIGHTGRAY,
    GRAY,
    DARKGRAY,
    YELLOW,
    GOLD,
    ORANGE,
    RED,
    MAROON,
    GREEN,
    LIME,
    DARKGREEN,
    SKYBLUE,
    BLUE,
    DARKBLUE,
    PURPLE,
    VIOLET,
    DARKPURPLE,
    BEIGE,
    BROWN,
    DARKBROWN,
    WHITE,
    BLACK,
    MAGENTA,
    RAYWHITE
};
```

6.3 Config.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** Config
*/

#pragma once

#define WINDOW_WIDTH 1920
#define WINDOW_HEIGHT 1080
#define WINDOW_TITLE "Zappy GUI"

#define SERVER_DOWN_MESSAGE         "THE GUI MUST BE CLOSED FOR UNEXPECTED REASONS"
```

```

#define SIZE_TILE                   4.7

#define PLAYER_HEIGHT               2

#define DEFAULT_RENDER_DISTANCE     15
#define MAX_RENDER_DISTANCE         20
#define MIN_RENDER_DISTANCE         1

#define HIGH_CAMERA_INCREASE        0.1
#define LOW_CAMERA_INCREASE         0.1

#define PLAYER_SECOND_PERSON_FOV    4.0f
#define PLAYER_THIRD_PERSON_FOV     5.0f
```

6.4 Engine.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Engine
*/

#pragma once

#include "Event/Event.hpp"
#include "Render/Render.hpp"
#include "Network/INetwork.hpp"
#include "GameDatas/GameData.hpp"
#include "Parsing/ServerParser.hpp"
#include "GUIUpdater/GUIUpdater.hpp"

#include <time.h>

#define TIME_UNIT_MAP_UPDATE 20

namespace Gui {

    class Engine;
};

class Gui::Engine {

    public:

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

        ~Engine() = default;

        void run();

    private:

        std::unique_ptr<IServerParser>      _parser;
        std::shared_ptr<INetwork>           _network;
        std::shared_ptr<Render>             _render;
        std::unique_ptr<IEvent>             _event;
        std::shared_ptr<GameData>           _gameData;
        std::unique_ptr<IGUIUpdater>        _guiUpdater;

        void listenServer();

        void sendMessageUpdate();

        void updateMap();

        void sendUpdateMapMessage();
};
```

6.5 AError.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** AError
*/

```

```

#pragma once

#include "Error/IError.hpp"

#include <string>

namespace Gui {
    namespace Errors {
        class AError;
    };
};

class Gui::Errors::AError : public IError {

    public:

        ~AError() override = default;

        const char *what() const noexcept override;

    protected:

        std::string _message;
};
```

6.6 Error.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Error
*/

#pragma once

#include "Error/AError.hpp"

namespace Gui {
    namespace Errors {

        class Error : public AError {};

        class NetworkException : public Error {

            public:
                NetworkException(std::string message);
        };

        class ServerParserException : public Error {

            public:
                ServerParserException(std::string message);
        };

        class ParseCommandLineException : public Error {

            public:
                ParseCommandLineException(std::string message);
        };

        class GuiGameDataException : public Error {

            public:
                GuiGameDataException(std::string message);
        };

        class GuiUpdaterException : public Error {

            public:
                GuiUpdaterException(std::string message);
        };
    };
};
```

6.7 IError.hpp

```
/*
```

```
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** IError
*/

#pragma once

#include <exception>

namespace Gui {
    namespace Errors {
        class IError;
    };
};

class Gui::Errors::IError : public std::exception {

    public:

        virtual ~IError() = default;

        virtual const char *what() const noexcept = 0;
};
```

6.8 AEvent.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** AEvent
*/

#pragma once

#include "Event/IEvent.hpp"

namespace Gui {

    class AEvent;
}

class Gui::AEvent : public Gui::IEvent {

    public:

        AEvent();

        ~AEvent() = default;

        virtual void listen() = 0;

        void setRender(std::shared_ptr<Render> render);

        void setGameData(std::shared_ptr<GameData> gameData);

    protected:

        std::shared_ptr<Render>         _render;
        std::shared_ptr<GameData>       _gameData;
};
```

6.9 Event.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Event
*/

#pragma once

#include "Config.hpp"
#include "Event/AEvent.hpp"
#include "Render/Render.hpp"

#include <functional>
```

```
#include <unordered_map>

namespace Gui {

    class Event;
};

class Gui::Event : public Gui::AEvent {

    public:

        Event();

        ~Event() = default;

        void listen();

    private:

        std::unordered_map<KeyboardKey, std::function<void()» _eventsKeyDown =
        {
            {KEY_SPACE, [this](){moveUpCamera();}},
            {KEY_LEFT_SHIFT, [this](){moveDownCamera();}},
        };

        std::unordered_map<GamepadButton, std::function<void()» _eventsGamepadButtonDown =
        {
            {GAMEPAD_BUTTON_RIGHT_TRIGGER_2, [this](){handleSpaceGamepad();}},
            {GAMEPAD_BUTTON_LEFT_TRIGGER_2, [this](){moveDownCamera();}},
        };

        std::unordered_map<KeyboardKey, std::function<void()» _eventsKeyPressed =
        {
            {KEY_THREE, [this](){switchDisplayDebug();}},
            {KEY_F3, [this](){switchDisplayDebug();}},
            {KEY_SPACE, [this](){setFreeCam();}},
            {KEY_R, [this](){switchTileHudToGame();}},
            {KEY_J, [this](){increaseRenderDistance();}},
            {KEY_K, [this](){decreaseRenderDistance();}},
            {KEY_F5, [this](){changeActualPlayerPov();}},
            {KEY_FOUR, [this](){changeActualPlayerPov();}},
            {KEY_V, [this](){setPlayerVision();}},
            {KEY_KP_ADD, [this](){increaseTimeUnit();}},
            {KEY_KP_SUBTRACT, [this](){decreaseTimeUnit();}},
            {KEY_H, [this](){displayHelpMenu();}},
        };

        std::unordered_map<GamepadButton, std::function<void()» _eventsGamepadButtonPressed =
        {
            {GAMEPAD_BUTTON_RIGHT_FACE_DOWN, [this](){handleLeftClick();}},
            {GAMEPAD_BUTTON_RIGHT_TRIGGER_1, [this](){handleLeftClick();}},
            {GAMEPAD_BUTTON_LEFT_TRIGGER_1, [this](){handleRightClick();}},
            {GAMEPAD_BUTTON_LEFT_FACE_UP, [this](){increaseRenderDistance();}},
            {GAMEPAD_BUTTON_LEFT_FACE_DOWN, [this](){decreaseRenderDistance();}},
            {GAMEPAD_BUTTON_LEFT_FACE_LEFT, [this](){decreaseTimeUnit();}},
            {GAMEPAD_BUTTON_LEFT_FACE_RIGHT, [this](){increaseTimeUnit();}},
            {GAMEPAD_BUTTON_RIGHT_FACE_LEFT, [this](){switchDisplayDebug();}},
            {GAMEPAD_BUTTON_RIGHT_FACE_RIGHT, [this](){switchTileHudToGame();}},
            {GAMEPAD_BUTTON_RIGHT_FACE_UP, [this](){changeActualPlayerPov();}},
            {GAMEPAD_BUTTON_MIDDLE_RIGHT, [this](){closeWindowGamepad();}},
        };

        std::unordered_map<MouseButton, std::function<void()» _eventsMousePressed =
        {
            {MOUSE_BUTTON_LEFT, [this](){handleLeftClick();}},
            {MOUSE_BUTTON_RIGHT, [this](){handleRightClick();}}
        };

        void moveUpCamera();

        void moveDownCamera();

        void switchDisplayDebug();

        void setFreeCam();

        void handleLeftClick();

        void handleRightClick();

        void selectPlayer();

        void selectTile();

        void changePlayer(bool turn);

        void changeActualPlayerPov();

        void switchTileHudToGame();

        void increaseRenderDistance();

        void decreaseRenderDistance();

        void increaseTimeUnit();

        void decreaseTimeUnit();

        void displayHelpMenu();

        void handleSpaceGamepad();

        void closeWindowGamepad();

        void setPlayerVision();
};
```

6.10 IEvent.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** IEvent
*/

#pragma once

#include "Render/Render.hpp"
#include "GameDatas/GameData.hpp"
#include <memory>

namespace Gui {

    class IEvent;
}

class Gui::IEvent {

    public:

        IEvent() = default;

        virtual ~IEvent() = default;

        virtual void listen() = 0;

        virtual void setRender(std::shared_ptr<Render> render) = 0;

        virtual void setGameData(std::shared_ptr<GameData> gameData) = 0;
};
```

6.11 Egg.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** Egg
*/

#pragma once

#include "Error/Error.hpp"

#include <string>

namespace Gui {

    class Egg;
};

class Gui::Egg {

    public:

        enum EggState {
            IDLE,
            DEAD,
            BORN,
            HATCHING,
        };

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

        ~Egg();

        std::size_t getId() const;

        std::string getTeam() const;

        std::pair<std::size_t, std::size_t> getPosition() const;

        void setId(std::size_t id);

        void setTeam(const std::string &team);

        void setPosition(std::pair<std::size_t, std::size_t> position);

        void setState(EggState state);

        EggState getState() const;

    private:

        std::size_t                             _id;
        std::string                             _team;
        std::pair<std::size_t, std::size_t>     _position;
        EggState                                _state;
};
```

6.12 GameData.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** GameData
*/

#pragma once

#include "Types.hpp"
#include "Error/Error.hpp"
#include "GameDatas/Team.hpp"
#include "GameDatas/Tile.hpp"

#define NO_TICK int(-1)

namespace Gui {

    class GameData;
};

class Gui::GameData {

    public:

        enum TimeUnitState {
            INCREASE,
            DECREASE,
            NONE
        };

        GameData();

        ~GameData() = default;

        std::vector<Gui::Team> &getTeams();

        Gui::Team &getTeam(const std::string &name);

        void addTeam(const Gui::Team &team);

        void addTeam(const std::string &name, const std::string &playerModelPath, const std::string &eggModelPath, Color playerColor);

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

        Gui::Player &getPlayer(size_t id);

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

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

        void setMapSize(size_t x, size_t y);

        std::pair<size_t, size_t> getMapSize() const;

        Gui::Tile &getTile(size_t x, size_t y);

        void setTile(const Gui::Tile &tile);

        void restartLastTick(void);

        void setServerTick(std::size_t tick);

        clock_t getLastTick() const;

        std::size_t getServerTick() const;

        void setIsEndGame(bool isEndGame);

        bool getIsEndGame() const;

        void setLastError(const std::string &error);

        std::string getLastError() const;

        Team &getTeamById(std::size_t id);

        TimeUnitState getTimeUnitFromServer() const;

        void setTimeUnitFromServer(TimeUnitState timeUnitFromServer);

        std::vector<Gui::Egg> &getServerEggs();

        void addServerEgg(const Gui::Egg &egg);

        void removeServerEgg(size_t id);

        void setNbBCTCommandReceived(std::size_t nb);

        std::size_t getNbBCTCommandReceived() const;

        void restartLastTickMctCommand();

        clock_t getLastTickMctCommand() const;

        void setEndMessage(const std::string &endMessage);

        std::string getEndMessage() const;

    private:

        std::vector<Gui::Team>      _teams;
        Map<Gui::Tile>              _map;
        std::size_t                 _serverTick;
        clock_t                     _lastTick;
        bool                        _isEndGame;
        std::size_t                 _nbBCTCommandReceived;
        clock_t                     _lastTickMctCommand;
        std::string                 _lastError;
        TimeUnitState               _timeUnitFromServer;
        std::vector<Gui::Egg>       _serverEggs;
        std::string                _endMessage;
};
```

6.13 Inventory.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Inventory
*/

#pragma once

#include <iostream>

#define RessourcesNumber 6
```

```

namespace Gui {

    class Inventory;
};

class Gui::Inventory {

    public:

        using Ressources = size_t [RessourcesNumber];

        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);

        ~Inventory() = default;

        void setFood(std::size_t food);

        void setLinemate(std::size_t linemate);

        void setDeraumere(std::size_t deraumere);

        void setSibur(std::size_t sibur);

        void setMendiane(std::size_t mendiane);

        void setPhiras(std::size_t phiras);

        void setThystame(std::size_t thytsame);

        void setRessources(Ressources ressources);

        std::size_t getFood(void);

        std::size_t getLinemate(void);

        std::size_t getDeraumere(void);

        std::size_t getSibur(void);

        std::size_t getMendiane(void);

        std::size_t getPhiras(void);

        std::size_t getThystame(void);

        Ressources &getRessources(void);

        void addResource(std::size_t resource, std::size_t quantity);

        void removeResource(std::size_t resource, std::size_t quantity);

    private:

        std::size_t     _food;
        Ressources      _ressources;
};
```

6.14 Player.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Player
*/

#pragma once

#include "raylib.h"
#include "GameDatas/Inventory.hpp"

namespace Gui {

    class Player;
};

class Gui::Player {

    public:

        enum PlayerState {
            IDLE = 2,
            BORN = 8,
            BROADCAST = 12,
            EJECT = 5,
            BEING_EJECTED = 15,
            EJECTED = 7,
            WALK = 6, // or 10
            INCANTATION = 0,
            LAY_EGG = 7,
            DROP = 9,
            COLLECT = 9,
            DEAD = 1,
        };

        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);

        ~Player() = default;

        void setPosition(std::pair<std::size_t, std::size_t> position);

        void setPosition3D(Vector3 position3D);

        void setId(std::size_t id);

        void setLevel(std::size_t level);

        void setOrientation(std::size_t orientation);

        void setTeam(const std::string &team);

        std::pair<std::size_t, std::size_t> getPosition(void) const;

        Vector3 getPosition3D(void) const;

        std::size_t getId(void) const;

        std::size_t getLevel(void) const;

        std::size_t getOrientation(void) const;

        std::string getTeam(void) const;

        void setState(PlayerState state);

        PlayerState getState(void) const;

        void setBroadcast(const std::string &broadcast);

        std::string getBroadcast() const;

        float getRotationFromOrientation() const;

        Vector3 getCenterPosition();

        void setCurrentFrame(int currentFrame);

        int getCurrentFrame() const;

        void restartAnimationTimeEllapsed();

        clock_t getAnimationTimeEllapsed() const;

        Inventory                               inventory;

    private:

        std::size_t                             _id;
        std::string                             _team;
        std::pair<std::size_t, std::size_t>     _position;
        Vector3                                 _position3D;
        std::size_t                             _orientation;
        std::size_t                             _level;
        PlayerState                             _state;
        std::string                             _broadcast;
        int                                     _currentFrame;
        clock_t                                 _animationTimeEllapsed;
};
```

6.15 Team.hpp

```
/*
```

```cpp
** EPITECH PROJECT, 2024
** Zappy
** File description:
** Team
*/

#pragma once

#include "raylib.h"
#include "Types.hpp"
#include "GameDatas/Egg.hpp"
#include "GameDatas/Tile.hpp"
#include "GameDatas/Player.hpp"

#include <vector>
#include <memory>

namespace Gui {

    class Team;
};

class Gui::Team {

    public:

        Team(const std::string &name, const std::string &playerModelPath, const std::string &eggModelPath, Color playerColor);

        ~Team();

        const std::string &getName() const;

        std::vector<Gui::Player> &getPlayers();

        std::vector<Gui::Egg> &getEggs();

        void setName(const std::string &name);

        void addPlayer(const Gui::Player &player);

        void addEgg(const Gui::Egg &egg);

        bool removePlayer(std::size_t id);

        bool removeEgg(std::size_t id);

        std::shared_ptr<Gui::Player> getPlayer(std::size_t id);

        Model getPlayerModel() const;

        ModelAnimation *getPlayerModelAnimation() const;

        void setPlayerModelPath(const std::string &playerModelPath);

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

        Model getEggModel() const;

        void setEggModelPath(const std::string &eggModelPath);

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

        Vector3 getPlayerPositionIn3DSpace(size_t id, Map<Tile> map);

        std::vector<RayCollision> getPlayerModelHitbox(size_t id, Camera camera);

        bool isPlayerHit(size_t id, Camera camera);

        Color getPlayerColor() const;

    private:


        ModelAnimation*             _modelAnimation;
        int                         _animsCount;
        std::string                 _name;
        std::vector<Gui::Player>    _players;
        Model                       _playerModel;
        std::vector<Gui::Egg>       _eggs;
        Model                       _eggModel;
        Color                       _playerColor;

        BoundingBox rotateBoundingBoxByOrientation(BoundingBox bbox, size_t orientation, std::pair<size_t, size_t> pos, Vector3 center);
};
```

6.16 Tile.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Tile
*/

#pragma once

#include "raylib.h"
#include "GameDatas/Inventory.hpp"

#include <vector>

namespace Gui {

    class Tile;
};

class Gui::Tile {

    public:

        Tile(std::pair<std::size_t, std::size_t> position);

        Tile(std::pair<std::size_t, std::size_t> position, Inventory inventory);

        ~Tile() = default;

        void setPosition(std::pair<std::size_t, std::size_t> position);

        std::pair<std::size_t, std::size_t> getPosition() const;

        Vector3 getPositionIn3DSpace();

        std::vector<BoundingBox> getTileBoundingBoxes(Tile tile, Model tileModel);

        std::vector<RayCollision> getTileModelHitbox(Tile tile, Camera camera, Model tileModel);

        bool isTileHit(Camera camera, Model _tileModel);

        Inventory       inventory;

    private:

        std::pair<std::size_t, std::size_t>     _position;
        Vector3                                 _positionIn3DSpace;
};
```

6.17 AGUIUpdater.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** AGUIUpdater
*/

#pragma once

#include "Network/Network.hpp"
#include "GameDatas/GameData.hpp"
#include "GUIUpdater/IGUIUpdater.hpp"

#include <memory>

namespace Gui {

    class AGUIUpdater;
}

class Gui::AGUIUpdater : public Gui::IGUIUpdater {
    public:

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

        ~AGUIUpdater() = default;

        void update(const std::string &command, const std::vector<std::string> &data) override = 0;

    protected:

        std::shared_ptr<GameData>       _gameData;
        std::shared_ptr<INetwork>       _network;
};
```

6.18 GUIUpdater.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** GUIUpdater
*/

#pragma once

#include "GUIUpdater/AGUIUpdater.hpp"

#include <string>
#include <functional>
#include <unordered_map>

namespace Gui {

    class GUIUpdater;
}

class Gui::GUIUpdater : public Gui::AGUIUpdater {

    public:

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

        ~GUIUpdater() = default;

        void update(const std::string &command, const std::vector<std::string> &data);

    private:

        size_t                          _colorIndex;

        std::unordered_map<std::string, std::function<void(std::vector<std::string>)» _updateMap =
        {
            {"msz", [this](std::vector<std::string> data){updateMapSize(data);}},
            {"bct", [this](std::vector<std::string> data){updateMapContent(data);}},
            {"tna", [this](std::vector<std::string> data){updateTeamNames(data);}},
            {"pnw", [this](std::vector<std::string> data){updateTeamMember(data);}},
            {"ppo", [this](std::vector<std::string> data){updatePlayerPosition(data);}},
            {"plv", [this](std::vector<std::string> data){updatePlayerLevel(data);}},
            {"pin", [this](std::vector<std::string> data){updatePlayerInventory(data);}},
            {"pex", [this](std::vector<std::string> data){updatePlayerExpulsion(data);}},
            {"pbc", [this](std::vector<std::string> data){updatePlayerBroadcast(data);}},
            {"pic", [this](std::vector<std::string> data){updatePlayerStartIncantation(data);}},
            {"pie", [this](std::vector<std::string> data){updatePlayerEndIncantation(data);}},
            {"pfk", [this](std::vector<std::string> data){updatePlayerEggLaying(data);}},
            {"pdr", [this](std::vector<std::string> data){updatePlayerRessourceDropping(data);}},
            {"pgt", [this](std::vector<std::string> data){updatePlayerRessourceCollecting(data);}},
            {"pdi", [this](std::vector<std::string> data){updatePlayerDeath(data);}},
            {"enw", [this](std::vector<std::string> data){updateEggLaidByPlayer(data);}},
            {"ebo", [this](std::vector<std::string> data){updatePlayerBorn(data);}},
            {"edi", [this](std::vector<std::string> data){updateEggDeath(data);}},
            {"sgt", [this](std::vector<std::string> data){updateTimeUnitRequest(data);}},
            {"sst", [this](std::vector<std::string> data){updateTimeUnitModification(data);}},
            {"seg", [this](std::vector<std::string> data){updateEndOfGame(data);}},
            {"smg", [this](std::vector<std::string> data){updateMessageFromServer(data);}},
            {"suc", [this](std::vector<std::string> data){updateUnknownMessage(data);}},
            {"sbp", [this](std::vector<std::string> data){updateCommandParameter(data);}},
        }; // The map of commands to update the GUI GameData.

        void updateMapSize(const std::vector<std::string> &data);

        void updateMapContent(const std::vector<std::string> &data);

        void updateTeamNames(const std::vector<std::string> &data);

        void updateTeamMember(const std::vector<std::string> &data);

        void updatePlayerPosition(const std::vector<std::string> &data);

        void updatePlayerLevel(const std::vector<std::string> &data);

        void updatePlayerInventory(const std::vector<std::string> &data);

        void updatePlayerExpulsion(const std::vector<std::string> &data);

        void updatePlayerBroadcast(const std::vector<std::string> &data);

        void updatePlayerStartIncantation(const std::vector<std::string> &data);

        void updatePlayerEndIncantation(const std::vector<std::string> &data);

        void updatePlayerEggLaying(const std::vector<std::string> &data);

        void updatePlayerRessourceDropping(const std::vector<std::string> &data);

        void updatePlayerRessourceCollecting(const std::vector<std::string> &data);

        void updatePlayerDeath(const std::vector<std::string> &data);

        void updateEggLaidByPlayer(const std::vector<std::string> &data);

        void updatePlayerBorn(const std::vector<std::string> &data);

        void updateEggDeath(const std::vector<std::string> &data);

        void updateTimeUnitRequest(const std::vector<std::string> &data);

        void updateTimeUnitModification(const std::vector<std::string> &data);

        void updateEndOfGame(const std::vector<std::string> &data);

        void updateMessageFromServer(const std::vector<std::string> &data);

        void updateUnknownMessage(const std::vector<std::string> &data);

        void updateCommandParameter(const std::vector<std::string> &data);

        void increaseColorIndex();
};
```

6.19 IGUIUpdater.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** IGUIUpdater
*/

#pragma once

#include <string>
#include <vector>

namespace Gui {

    class IGUIUpdater;
}

class Gui::IGUIUpdater {
    public:
        virtual ~IGUIUpdater() = default;

        virtual void update(const std::string &command, const std::vector<std::string> &data) = 0;
};
```

6.20 AHud.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** AHud
*/

#pragma once

#include "Hud/IHud.hpp"
#include "GameDatas/GameData.hpp"

namespace Gui {

    class AHud;
};

class Gui::AHud : public Gui::IHud {

    public:

        ~AHud() = default;

        virtual void display() = 0;

        void setPlayer(std::shared_ptr<Player> player);

        void setTile(std::shared_ptr<Tile> tile);

        TypeScene getType() const;

        void setType(TypeScene type);

    protected:

        TypeScene                       _typeScene;
        std::shared_ptr<GameData>       _gameData;
        std::shared_ptr<Player>         _player;
        std::shared_ptr<Tile>           _tile;
};
```

6.21 HudEnd.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** HudEnd
*/

#pragma once

#include "Hud/AHud.hpp"

namespace Gui {

    class HudEnd;
};

class Gui::HudEnd : public Gui::AHud {

    public:

        HudEnd(std::shared_ptr<GameData> gameData);

        ~HudEnd() = default;

        void display();

    private:

            Texture2D   _texture;
            Font        _font;
};
```

6.22 HudGame.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** HudGame
*/

#pragma once

#include "Hud/AHud.hpp"

#define HUD_GAME_POS Vector2{0, 340}
#define HUD_GAME_TEXT_POS Vector2{25, 420}
#define HUD_GAME_TEXT_TITLE_POS Vector2{50, 380}
#define HUD_GAME_TEXT_MARGING 30

```

```
namespace Gui {

    class HudGame;
};

class Gui::HudGame : public Gui::AHud {

    public:

        HudGame(std::shared_ptr<GameData> gameData);

        ~HudGame() = default;

        void display();

    private:

        Texture2D   _texture;
        Font        _font;
        Texture2D   _playerTexture;
};
```

6.23 HudHelp.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** HudHelp
*/

#pragma once

#include "Hud/AHud.hpp"

namespace Gui {

    class HudHelp;
};

class Gui::HudHelp : public Gui::AHud {

    public:

        HudHelp(std::shared_ptr<GameData> gameData);

        ~HudHelp() = default;

        void display();

    private:

        Texture2D       _texture;           // Texture for Hud Background.
        Texture2D       _textureKeys;       // Texture for Hud keys.
        Font            _font;              // Font for Hud's texts.
        clock_t         _helpMenuClock;     // Clock to display the help menu.

        void displayHelpMenuControls();
};
```

6.24 HudPlayer.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** HudPlayer
*/

#pragma once

#include "Hud/AHud.hpp"

#define HUD_PLAYER_POS Vector2{0, 340}
#define HUD_PLAYER_TEXT_POS Vector2{55, 420}
#define HUD_PLAYER_TEXT_TITLE_POS Vector2{80, 380}
#define HUD_PLAYER_TEXT_MARGING 30
#define HUD_PLAYER_ICONS_MARGING -32
#define HUD_PLAYER_TITLE_ICON_MARGING Vector2{45, 35}
```

```

namespace Gui {

    class HudPlayer;
};

class Gui::HudPlayer : public Gui::AHud {

    public:

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

        ~HudPlayer() = default;

        void display();

    private:

        Texture2D   _texture;
        Font        _font;

        Texture2D   _food;
        Texture2D   _linemate;
        Texture2D   _deraumere;
        Texture2D   _mendiane;
        Texture2D   _phiras;
        Texture2D   _sibur;
        Texture2D   _thystame;
        Texture2D   _playerTexture;
};
```

6.25 HudTile.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** HudTile
*/

#pragma once

#include "Hud/AHud.hpp"

#define HUD_TILE_POS Vector2{0, 340}
#define HUD_TILE_TEXT_POS Vector2{55, 420}
#define HUD_TILE_TEXT_TITLE_POS Vector2{60, 380}
#define HUD_TILE_TEXT_MARGING 30
#define HUD_TILE_ICONS_MARGING -32
#define HUD_TILE_TITLE_ICON_MARGING Vector2{45, 40}

namespace Gui {

    class HudTile;
};

class Gui::HudTile : public Gui::AHud {

    public:

        HudTile(std::shared_ptr<GameData> gameData);

        ~HudTile() = default;

        void display();

        void displayNbPlayers();

        void displayNbEggs();

    private:

        Texture2D   _texture;
        Font        _font;

        Texture2D   _food;
        Texture2D   _linemate;
        Texture2D   _deraumere;
        Texture2D   _mendiane;
        Texture2D   _phiras;
        Texture2D   _sibur;
        Texture2D   _thystame;
        Texture2D   _tileTexture;
        Texture2D   _playerTexture;
        Texture2D   _eggTexture;
};
```

6.26 IHud.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** IHud
*/

#pragma once

#include "GameDatas/Player.hpp"
#include "GameDatas/Tile.hpp"

#include <memory>

namespace Gui {

    class IHud;
};

class Gui::IHud {

    public:

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

        virtual ~IHud() = default;

        virtual void display() = 0;

        virtual void setPlayer(std::shared_ptr<Player> player) = 0;

        virtual void setTile(std::shared_ptr<Tile> tile) = 0;

        virtual TypeScene getType() const = 0;

        virtual void setType(TypeScene type) = 0;
};
```

6.27 ANetwork.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** ANetwork
*/

#pragma once

#include "Network/INetwork.hpp"

#define MIN_PORT 1
#define MAX_PORT 65535

namespace Gui {

    class ANetwork;
};

class Gui::ANetwork : public Gui::INetwork {
    public:

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

        ~ANetwork() = default;

        void setPort(int port) final;

        void setHostName(const std::string &hostName) final;

        int getPort() const final;

        std::string getHostName() const final;

        virtual void connectToServer() = 0;

        virtual BufferState listenServer() = 0;

        virtual void sendMessageServer(const std::string& message) = 0;

        std::string getBuffer();

    protected:
        int             _port;
        std::string     _hostName;
        std::string     _buffer;
};
```

6.28 INetwork.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** INetwork
*/

#pragma once

#include "Error/Error.hpp"

#include <string>

namespace Gui {

    class INetwork;
};

class Gui::INetwork {

    public:

        enum BufferState {
            NONE,
            READY,
            SERVER_ERROR
        };

        virtual ~INetwork() = default;

        virtual void setPort(int port) = 0;

        virtual void setHostName(const std::string &hostName) = 0;

        virtual int getPort() const = 0;

        virtual std::string getHostName() const = 0;

        virtual void connectToServer() = 0;

        virtual BufferState listenServer() = 0;

        virtual void sendMessageServer(const std::string &message) = 0;

        virtual std::string getBuffer() = 0;
};
```

6.29 Network.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Network
```

```
*/

#pragma once

#include "Network/ANetwork.hpp"

#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>

namespace Gui {

    class Network;
};

class Gui::Network : public Gui::ANetwork {

    public:

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

        ~Network();

        void connectToServer();

        BufferState listenServer();

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

    private:

        void selectServer();

        BufferState readInfoServer();

        int             _serverFd;
        fd_set          _writeFd;
        fd_set          _readFd;
        bool            _isConnected;
};
```

6.30 IServerParser.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** IServerParser
*/

#pragma once

#include <string>
#include <vector>

namespace Gui {

    class IServerParser;
}

class Gui::IServerParser {

    public:

        virtual ~IServerParser() = default;

        virtual std::vector<std::string> parse(const std::string& command) = 0;
};
```

6.31 ParseCommandLine.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** ParseCommandLine
*/

#pragma once
```

```

#include <string>

#define GUI_USAGE "USAGE:\t./zappy_gui -p port -h machine"

namespace Gui {

    class ParseCommandLine;
};

class Gui::ParseCommandLine {

    public:

        ParseCommandLine(int argc, char **argv);

        ~ParseCommandLine() = default;

        void parseFlags(int argc, char **argv);

        int getPort(void);

        std::string getHostName(void);

    private:

        int             _port;
        std::string     _hostName;
};
```

6.32 ServerParser.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Parse server command
*/

#pragma once

#include "Error/Error.hpp"
#include "Parsing/IServerParser.hpp"

#include <functional>
#include <unordered_map>

namespace Gui {

    class ServerParser;
};

class Gui::ServerParser : public Gui::IServerParser {

    public:

        ServerParser() = default;

        ~ServerParser() = default;

        std::vector<std::string> parse(const std::string& command);

    private:

        enum ParseType {
            INT,
            STRING,
            MESSAGE,
            LIST_INT
        };

        std::unordered_map<std::string, std::vector<ParseType» _typesCommand =
        {
            {"msz", std::vector<ParseType>{INT, INT}},
            {"bct", std::vector<ParseType>{INT, INT, INT, INT, INT, INT, INT, INT, INT}},
            {"tna", std::vector<ParseType>{STRING}},
            {"pnw", std::vector<ParseType>{INT, INT, INT, INT, INT, STRING}},
            {"ppo", std::vector<ParseType>{INT, INT, INT, INT}},
            {"plv", std::vector<ParseType>{INT, INT}},
            {"pin", std::vector<ParseType>{INT, INT, INT, INT, INT, INT, INT, INT, INT, INT}},
            {"pex", std::vector<ParseType>{INT}},
            {"pbc", std::vector<ParseType>{INT, MESSAGE}},
            {"pic", std::vector<ParseType>{INT, INT, INT, LIST_INT}},
            {"pie", std::vector<ParseType>{INT, INT, INT}},
            {"pfk", std::vector<ParseType>{INT}},
            {"pdr", std::vector<ParseType>{INT, INT}},
            {"pgt", std::vector<ParseType>{INT, INT}},
            {"pdi", std::vector<ParseType>{INT}},
            {"enw", std::vector<ParseType>{INT, INT, INT, INT}},
            {"ebo", std::vector<ParseType>{INT}},
            {"edi", std::vector<ParseType>{INT}},
            {"sgt", std::vector<ParseType>{INT}},
            {"sst", std::vector<ParseType>{INT}},
            {"seg", std::vector<ParseType>{STRING}},
            {"smg", std::vector<ParseType>{MESSAGE}},
            {"suc", std::vector<ParseType>{}},
            {"sbp", std::vector<ParseType>{}}
        };

        std::vector<std::string> parseCommand(const std::string& command, std::vector<ParseType> types);

        std::vector<std::string> parseInt(std::istringstream& stream, std::vector<std::string> arguments);

        std::vector<std::string> parseString(std::istringstream& stream, std::vector<std::string> arguments);

        std::vector<std::string> parseMessage(std::istringstream& stream, std::vector<std::string> arguments, std::string commandName);

        std::vector<std::string> parseListInt(std::istringstream& stream, std::vector<std::string> arguments, std::string commandName);
};
```

6.33 Decoration.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy
** File description:
** Decoration
*/

#pragma once

#include "raylib.h"
#include "Types.hpp"
#include "Assets.hpp"

#include <vector>
#include <iostream>

namespace Gui {

    class Decoration;
}

class Gui::Decoration {

    public:

        Decoration();

        ~Decoration() = default;

        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);

        Map<bool> getGenerationItem(std::size_t ratio);

    private:

        Model                                   _treeModel;
        Map<bool>                               _mapTree;

        std::pair<std::size_t, std::size_t>     _mapSize;

        void displayTree(size_t i, size_t j, Vector3 posTile);

        bool isInArrayPlayerVision(std::pair<size_t, size_t> pos, std::vector<Vector2> _playerVisionPositions);
};
```

6.34 Render.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Render
*/

#pragma once


#include "raylib.h"
#include "Config.hpp"
#include "Hud/HudGame.hpp"
#include "Hud/HudTile.hpp"
#include "Hud/HudPlayer.hpp"
#include "Hud/HudHelp.hpp"
#include "Hud/HudEnd.hpp"
#include "Render/Decoration.hpp"
#include "Render/UserCamera.hpp"
#include "GameDatas/GameData.hpp"

#include <functional>
#include <unordered_map>

namespace Gui {

    class Render;
};

class Gui::Render {

    public:

        Render(std::shared_ptr<GameData> gameData);

        ~Render();

        bool isOpen();

        void draw();

        std::shared_ptr<Camera> getCamera();

        void setIsDebug(bool isDebug);

        bool getIsDebug(void);

        void setCameraType(Gui::UserCamera::CameraType type);

        Gui::UserCamera::CameraType getCameraType() const;

        void setCameraPlayerPov(std::size_t id);

        std::size_t getCameraPlayerPov() const;

        void setCameraTile(std::pair<std::size_t, std::size_t> pos);

        std::pair<std::size_t, std::size_t> getCameraTile() const;

        Model getTileModel() const;

        void setRenderDistance(size_t renderDistance);

        size_t getRenderDistance() const;

        bool isCameraInPlayerPov() const;

        void changePlayerPOV(size_t playerId);

        void setPlayerPov(size_t playerId);

        void changePOVToFirstPerson(size_t id);

        void changePOVToSecondPerson(size_t id);

        void changePOVToThirdPerson(size_t id);

        size_t getTimeUnit() const;

        void setTimeUnit(size_t timeUnit);

        void setPlayerVision(bool isPlayerVision);

        bool getPlayerVision() const;

        void setHelpMenu(bool isHelpMenu);

        bool getHelpMenu() const;

        void drawEnd() const;

    private:

        UserCamera                                  _camera;
        bool                                        _isDebug;
        std::shared_ptr<GameData>                   _gameData;
        std::shared_ptr<Decoration>                 _decoration;
        std::vector<std::shared_ptr<Gui::IHud»     _hudList;
        size_t                                     _renderDistance;
        bool                                        _isHelpMenu;

        Model                                       _tileModel;
        Model                                       _foodModel;
        Model                                       _linemateModel;
        Model                                       _mendianeModel;
        Model                                       _phirasModel;
        Model                                       _siburModel;
        Model                                       _thystameModel;
        Model                                       _deraumereModel;
        Texture2D                                   _cursorTexture;
        std::vector<Vector2>                        _playerVisionPositions;
        bool                                        _endHudSet;

        void LoadModels();

        void displayHUD();

        void displayDebug();

        void displayPlayers();

        void displayPlayerLevel(Player &player, Vector3 position, Team &team);

        void displayPlayerBroadcast(Team &team, Player &player);

        void displayMap();

        void displayTile(Tile tile);

        void displayEggs(Tile tile) const;

        void displayFood(Tile tile) const;

        void displayResources(Tile tile) const;

        void displayLinemate(Tile tile) const;

        void displayMendiane(Tile tile) const;

        void displayPhiras(Tile tile) const;

        void displaySibur(Tile tile) const;

        void displayThystame(Tile tile) const;

        void displayDeraumere(Tile tile) const;

        bool displayAnimations(Team &team, Player &player);

        ModelAnimation displayWalkAnimation(Team &team, Player &player, ModelAnimation anim);

        void displayCursor();

        std::pair<std::size_t, std::size_t> getCameraTile();

        std::vector<Vector2> getPositionsInPlayerVision(size_t playerId);

        bool isInArrayPlayerVision(std::pair<size_t, size_t> pos);

        std::vector<Vector2> getLineOfVision(Vector2 pos, size_t sizeOfHalf, size_t orientation);

        std::vector<Vector2> addVisionPosition(std::vector<Vector2> vision, std::vector<Vector2> pos);

        void displayHelpMenu(std::shared_ptr<IHud> hud);

        void displayHelpMenuControls(Vector2 position);
};
```

6.35 UserCamera.hpp

```
/*
** EPITECH PROJECT, 2024
** Zappy GUI
** File description:
** Camera
*/

#pragma once

#include "raylib.h"

#include <memory>

namespace Gui {

    class UserCamera;
};

class Gui::UserCamera {

    public:

        enum CameraType {
            FREE,
            FIRST_PERSON,
            SECOND_PERSON,
            THIRD_PERSON,
            FREE_TILE
        };

        UserCamera();

        ~UserCamera() = default;

        void setPosition(Vector3 position);

        void setTarget(Vector3 target);

        void setUp(Vector3 up);

        void setFovy(float fovy);

        Vector3 getPosition(void) const;

        Vector3 getTarget(void) const;

        Vector3 getUp(void) const;

        float getFovy(void) const;

        std::shared_ptr<Camera> getCamera();

        void setType(CameraType type);

        CameraType getType() const;

        void setPlayerId(size_t playerId);

        int getPlayerId() const;

        void setTilePos(std::pair<std::size_t, std::size_t> pos);

        std::pair<std::size_t, std::size_t> getTilePos() const;

        bool isPlayerPov() const;

        bool isPlayerVision() const;

        void setPlayerVision(bool isPlayerVision);

    private:

        std::shared_ptr<Camera>                 _camera;
        CameraType                              _type;
        int                                     _playerId;
        std::pair<std::size_t, std::size_t>     _tilePos;
        bool                                    _isPlayerVision;
};
```

6.36 Types.hpp

```
/*
```

```
** EPITECH PROJECT, 2024
** Zappy
** File description:
** Types
*/

#pragma once

#include <vector>

template<typename T>
using Map = std::vector<std::vector<T»;
```

Index