

zappy\_gui

0.1.0

Generated by Doxygen 1.9.7



<b>1 Hierarchical Index</b>	<b>1</b>
1.1 Class Hierarchy	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Class Documentation</b>	<b>7</b>
4.1 gui::Argument Class Reference	7
4.2 myLib::Clock Class Reference	7
4.3 gui::Gui Class Reference	7
4.4 gui::IClient Class Reference	8
4.5 gui::Inventory Class Reference	9
4.6 gui::IPlugin Class Reference	9
4.7 gui::IRenderer Class Reference	10
4.8 gui::KeyBoard Class Reference	11
4.9 gui::Map Class Reference	11
4.10 gui::Mob Class Reference	11
4.11 gui::Parser Class Reference	12
4.12 gui::Parser::ParserException Class Reference	12
4.13 gui::PluginLoader Class Reference	13
4.14 gui::PluginLoader::PluginLoaderException Class Reference	14
4.15 gui::Protocol Class Reference	15
4.16 myLib::Random Class Reference	15
4.17 gui::Resource Class Reference	15
4.18 gui::RunTimeException Class Reference	16
4.19 gui::SFML Class Reference	17
4.19.1 Member Function Documentation	18
4.19.1.1 close()	18
4.19.1.2 getClient()	18
4.19.1.3 getEvents()	18
4.19.1.4 getPluginName()	18
4.19.1.5 init()	19
4.19.1.6 isRunning()	19
4.19.1.7 render()	19
4.19.1.8 setFPS()	19
4.20 gui::SFMLClient Class Reference	19
4.20.1 Member Function Documentation	20
4.20.1.1 connect()	20
4.20.1.2 disconnect()	20
4.20.1.3 getResponse() [1/2]	21

---

4.20.1.4 <a href="#">getResponse()</a> [2/2]	21
4.20.1.5 <a href="#">isConnected()</a>	21
4.20.1.6 <a href="#">sendCommand()</a>	21
4.21 <a href="#">myLib::Time Class Reference</a>	21
<b>5 File Documentation</b>	<b>23</b>
5.1 <a href="#">IClient.hpp</a>	23
5.2 <a href="#">IPlugin.hpp</a>	23
5.3 <a href="#">IRenderer.hpp</a>	24
5.4 <a href="#">Argument.hpp</a>	24
5.5 <a href="#">Constant.hpp</a>	24
5.6 <a href="#">Gui.hpp</a>	25
5.7 <a href="#">Inventory.hpp</a>	25
5.8 <a href="#">Resource.hpp</a>	26
5.9 <a href="#">KeyBoard.hpp</a>	26
5.10 <a href="#">Parser.hpp</a>	27
5.11 <a href="#">PluginLoader.hpp</a>	28
5.12 <a href="#">Protocol.hpp</a>	28
5.13 <a href="#">RunTimeException.hpp</a>	29
5.14 <a href="#">Map.hpp</a>	30
5.15 <a href="#">Mob.hpp</a>	30
5.16 <a href="#">Position.hpp</a>	31
5.17 <a href="#">SFML.hpp</a>	31
5.18 <a href="#">SFMLClient.hpp</a>	32
5.19 <a href="#">Clock.hpp</a>	32
5.20 <a href="#">Time.hpp</a>	33
5.21 <a href="#">Random.hpp</a>	34
<b>Index</b>	<b>37</b>

# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

gui::Argument . . . . .	7
myLib::Clock . . . . .	7
std::exception	
gui::Parser::ParserException . . . . .	12
gui::PluginLoader::PluginLoaderException . . . . .	14
gui::RunTimeException . . . . .	16
gui::Gui . . . . .	7
gui::IClient . . . . .	8
gui::SFMLClient . . . . .	19
gui::Inventory . . . . .	9
gui::IPlugin . . . . .	9
gui::IRenderer . . . . .	10
gui::SFML . . . . .	17
gui::KeyBoard . . . . .	11
gui::Map . . . . .	11
gui::Mob . . . . .	11
gui::Parser . . . . .	12
gui::PluginLoader . . . . .	13
gui::Protocol . . . . .	15
myLib::Random . . . . .	15
gui::Resource . . . . .	15
myLib::Time . . . . .	21



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">gui::Argument</a>	7
<a href="#">myLib::Clock</a>	7
<a href="#">gui::Gui</a>	7
<a href="#">gui::IClient</a>	8
<a href="#">gui::Inventory</a>	9
<a href="#">gui::IPlugin</a>	9
<a href="#">gui::IRenderer</a>	10
<a href="#">gui::KeyBoard</a>	11
<a href="#">gui::Map</a>	11
<a href="#">gui::Mob</a>	11
<a href="#">gui::Parser</a>	12
<a href="#">gui::Parser::ParserException</a>	12
<a href="#">gui::PluginLoader</a>	13
<a href="#">gui::PluginLoader::PluginLoaderException</a>	14
<a href="#">gui::Protocol</a>	15
<a href="#">myLib::Random</a>	15
<a href="#">gui::Resource</a>	15
<a href="#">gui::RunTimeException</a>	16
<a href="#">gui::SFML</a>	17
<a href="#">gui::SFMLClient</a>	19
<a href="#">myLib::Time</a>	21





# Chapter 3

## File Index

### 3.1 File List

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

include/GUI/ <a href="#">Argument.hpp</a>	24
include/GUI/ <a href="#">Constant.hpp</a>	24
include/GUI/ <a href="#">Gui.hpp</a>	25
include/GUI/ <a href="#">KeyBoard.hpp</a>	26
include/GUI/ <a href="#">Parser.hpp</a>	27
include/GUI/ <a href="#">PluginLoader.hpp</a>	28
include/GUI/ <a href="#">Protocol.hpp</a>	28
include/GUI/ <a href="#">RunTimeException.hpp</a>	29
include/GUI/Abstraction/ <a href="#">IClient.hpp</a>	23
include/GUI/Abstraction/ <a href="#">IPlugin.hpp</a>	23
include/GUI/Abstraction/ <a href="#">IRenderer.hpp</a>	24
include/GUI/Inventory/ <a href="#">Inventory.hpp</a>	25
include/GUI/Inventory/ <a href="#">Resource.hpp</a>	26
lib/shared/Renderer/SFML/include/GUI/ <a href="#">Map.hpp</a>	30
lib/shared/Renderer/SFML/include/GUI/ <a href="#">Mob.hpp</a>	30
lib/shared/Renderer/SFML/include/GUI/ <a href="#">Position.hpp</a>	31
lib/shared/Renderer/SFML/include/GUI/ <a href="#">SFML.hpp</a>	31
lib/shared/Renderer/SFML/include/GUI/ <a href="#">SFMLClient.hpp</a>	32
lib/static/myLib/include/myLib/ <a href="#">Random.hpp</a>	34
lib/static/myLib/include/myLib/Clock/ <a href="#">Clock.hpp</a>	32
lib/static/myLib/include/myLib/Clock/ <a href="#">Time.hpp</a>	33



## Chapter 4

# Class Documentation

### 4.1 gui::Argument Class Reference

#### Public Member Functions

- **Argument** (const uint16\_t p, std::string h)

#### Public Attributes

- const uint16\_t **port**
- const std::string **hostName**

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

- include/GUI/Argument.hpp

### 4.2 myLib::Clock Class Reference

#### Public Member Functions

- void **restart** ()
- void **pause** ()
- void **resume** ()
- [Time](#) **getElapsedTime** () const

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

- lib/static/myLib/include/myLib/Clock/Clock.hpp

### 4.3 gui::Gui Class Reference

#### Public Types

- enum class **RendererMode** { **GAME** , **SETTINGS** }

### Public Member Functions

- **Gui** (const [Argument](#) &args)
- std::unique\_ptr< [IRenderer](#) > & **getRenderer** ()
- void **Run** ()

### Static Public Member Functions

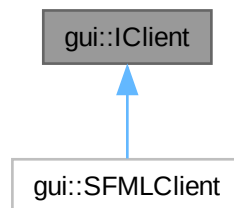
- static std::vector< std::string > **getData** (const std::string &data)

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

- include/GUI/Gui.hpp

## 4.4 gui::IClient Class Reference

Inheritance diagram for gui::IClient:



### Public Member Functions

- virtual bool **connect** (uint16\_t port, const std::string &machineName)=0
- virtual void **disconnect** ()=0
- virtual bool **sendCommand** (const std::string &cmd)=0
- virtual bool **getResponse** (const std::string &cmd)=0
- virtual std::string **getResponse** ()=0
- virtual bool **isConnected** ()=0

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

- include/GUI/Abstraction/IClient.hpp

## 4.5 gui::Inventory Class Reference

### Public Member Functions

- **Inventory** ([Resource](#) food, [Resource](#) linemate, [Resource](#) deraumere, [Resource](#) sibur, [Resource](#) mendiane, [Resource](#) phiras, [Resource](#) thystame)
- **Inventory** (std::vector< [Resource](#) > cresources)

### Public Attributes

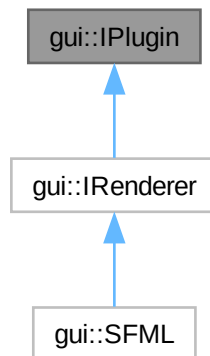
- std::vector< [Resource](#) > **resources**

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

- include/GUI/Inventory/Inventory.hpp

## 4.6 gui::IPlugin Class Reference

Inheritance diagram for gui::IPlugin:



### Public Member Functions

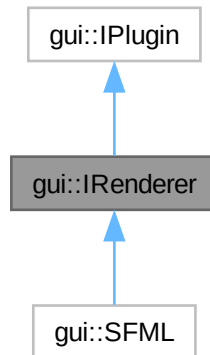
- virtual std::string **getPluginName** () const =0

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

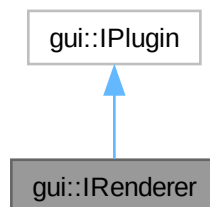
- include/GUI/Abstraction/IPlugin.hpp

## 4.7 gui::IRenderer Class Reference

Inheritance diagram for gui::IRenderer:



Collaboration diagram for gui::IRenderer:



### Public Member Functions

- virtual void **setFPS** (unsigned int FPS)=0
- virtual [IClient](#) & **getClient** ()=0
- virtual bool **isRunning** ()=0
- virtual void **init** (const std::string &name, std::pair< const unsigned int, const unsigned int > resolution, unsigned int bitsPerPixel)=0
- virtual void **render** ()=0
- virtual KeyBoard::Key **getEvents** ()=0
- virtual void **close** ()=0

## Public Member Functions inherited from gui::IPlugin

- virtual std::string **getPluginName** () const =0

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

- include/GUI/Abstraction/IRenderer.hpp

## 4.8 gui::KeyBoard Class Reference

### Public Types

- enum **Key** {  
    **NONE** = -1 , **CLOSE** = 0 , **KEY\_LEFT** = 1 , **KEY\_RIGHT** = 2 ,  
    **KEY\_UP** = 3 , **KEY\_DOWN** = 4 , **KEY\_SPACE** = 5 , **KEY\_ENTER** = 6 ,  
    **KEY\_ESCAPE** = 7 , **COUNT** = 8 }

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

- include/GUI/KeyBoard.hpp

## 4.9 gui::Map Class Reference

### Public Member Functions

- **Map** (unsigned int width, unsigned int height)
- unsigned int **getWidth** () const
- unsigned int **getHeight** () const

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

- lib/shared/Renderer/SFML/include/GUI/Map.hpp

## 4.10 gui::Mob Class Reference

### Public Types

- enum class **Action** { **MOVE** , **FEED** , **ELEVATE** , **NONE** }

### Public Member Functions

- Action **getAction** () const
- [Inventory](#) & **getInventory** ()
- Position & **getPosition** ()
- unsigned int **getLevel** () const
- void **setAction** (const Action &action)
- void **levelUp** ()

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

- lib/shared/Renderer/SFML/include/GUI/Mob.hpp

## 4.11 [gui::Parser](#) Class Reference

### Classes

- class [ParserException](#)

### Static Public Member Functions

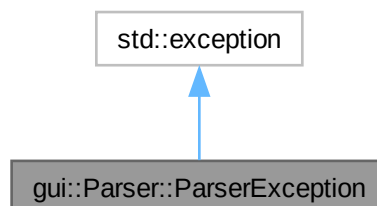
- static [Argument](#) **ParseArgs** (int argc, char \*const argv[])
- static uint16\_t **ParsePort** (const char \*port)
- static std::string **ParseMachineName** (const char \*machineName)
- static void **processData** (std::vector< std::string > data, [Gui](#) &gui)
- static [Inventory](#) **parseTileContent** (std::string tileContent)

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

- include/GUI/Parser.hpp

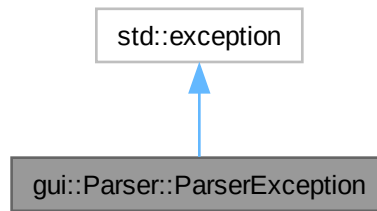
## 4.12 [gui::Parser::ParserException](#) Class Reference

Inheritance diagram for [gui::Parser::ParserException](#):





Collaboration diagram for gui::Parser::ParserException:



### Public Member Functions

- **ParserException** (std::string msg)
- **ParserException** (const [ParserException](#) &)=delete
- [ParserException](#) & **operator=** (const [ParserException](#) &)=delete
- **ParserException** (const [ParserException](#) &&)=delete
- [ParserException](#) & **operator=** (const [ParserException](#) &&)=delete
- const char \* **what** () const noexcept override

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

- include/GUI/Parser.hpp

## 4.13 gui::PluginLoader Class Reference

### Classes

- class [PluginLoaderException](#)

### Public Types

- using **PluginCreator** = std::unique\_ptr< [IPlugin](#) >(\*)()

### Public Member Functions

- template<typename T >  
std::unique\_ptr< T > **getPlugin** (const std::string &pluginName)
- void **closePlugins** ()

### Static Public Member Functions

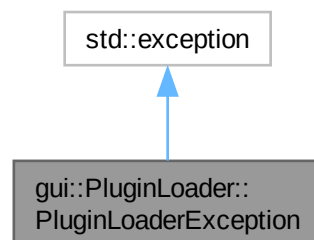
- static [PluginLoader](#) & **getInstance** ()

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

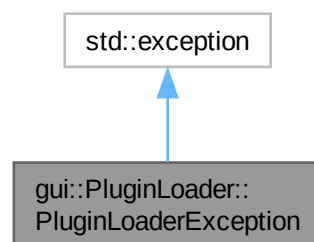
- include/GUI/PluginLoader.hpp

## 4.14 gui::PluginLoader::PluginLoaderException Class Reference

Inheritance diagram for gui::PluginLoader::PluginLoaderException:



Collaboration diagram for gui::PluginLoader::PluginLoaderException:



### Public Member Functions

- **PluginLoaderException** (std::string msg)
- const char \* **what** () const noexcept override

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

- include/GUI/PluginLoader.hpp

## 4.15 gui::Protocol Class Reference

### Static Public Member Functions

- static std::string **getCommand** (ProtocolKey key)
- static ProtocolKey **getKey** (const std::string &command)

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

- include/GUI/Protocol.hpp

## 4.16 myLib::Random Class Reference

### Static Public Member Functions

- static int **randomInt** (int min, int max)
- static int **randomInt** ()
- static float **randomFloat** (float min, float max)
- static float **randomFloat** ()

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

- lib/static/myLib/include/myLib/Random.hpp

## 4.17 gui::Resource Class Reference

### Public Types

- enum class **Type** {  
    **FOOD** , **LINEMATE** , **DERAUMERE** , **SIBUR** ,  
    **MENDIANE** , **PHIRAS** , **THYSTAME** , **NONE** }

### Public Member Functions

- **Resource** (Type type, unsigned int quantity)
- bool **operator==** (const [Resource](#) &resource) const

### Public Attributes

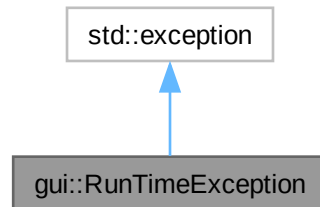
- Type **type**
- double **density**
- unsigned int **quantity**

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

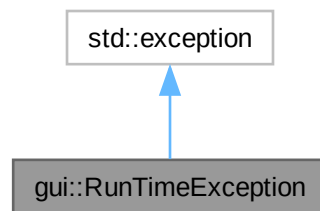
- include/GUI/Inventory/Resource.hpp

## 4.18 gui::RunTimeException Class Reference

Inheritance diagram for gui::RunTimeException:



Collaboration diagram for gui::RunTimeException:



### Public Member Functions

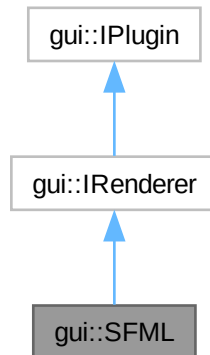
- **RunTimeException** (std::string msg)
- **RunTimeException** (const [RunTimeException](#) &)=delete
- [RunTimeException](#) & **operator=** (const [RunTimeException](#) &)=delete
- **RunTimeException** (const [RunTimeException](#) &&)=delete
- [RunTimeException](#) & **operator=** (const [RunTimeException](#) &&)=delete
- const char \* **what** () const noexcept override

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

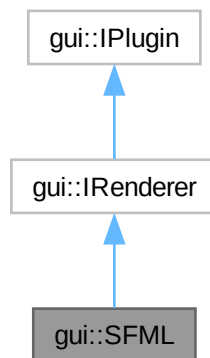
- include/GUI/RunTimeException.hpp

## 4.19 gui::SFML Class Reference

Inheritance diagram for gui::SFML:



Collaboration diagram for gui::SFML:



### Public Member Functions

- void [setFPS](#) (const unsigned int FPS) override
- std::string [getPluginName](#) () const override
- [IClient](#) & [getClient](#) () override
- KeyBoard::Key [getEvents](#) () override
- bool [isRunning](#) () override
- void [init](#) (const std::string &name, std::pair< const unsigned int, const unsigned int > resolution, unsigned int bitsPerPixel) override

- void [close](#) () override
- void [render](#) () override
- bool **checkConnection** (sf::Clock clock)
- virtual void **setFPS** (unsigned int FPS)=0
- virtual [IClient](#) & **getClient** ()=0
- virtual bool **isRunning** ()=0
- virtual void **init** (const std::string &name, std::pair< const unsigned int, const unsigned int > resolution, unsigned int bitsPerPixel)=0
- virtual void **render** ()=0
- virtual KeyBoard::Key **getEvents** ()=0
- virtual void **close** ()=0
- virtual std::string **getPluginName** () const =0

### Static Public Member Functions

- static KeyBoard::Key **getKeyboardEvent** (const sf::Event &event)

## 4.19.1 Member Function Documentation

### 4.19.1.1 close()

```
void gui::SFML::close ( ) [inline], [override], [virtual]
```

Implements [gui::IRenderer](#).

### 4.19.1.2 getClient()

```
IClient & gui::SFML::getClient ( ) [inline], [override], [virtual]
```

Implements [gui::IRenderer](#).

### 4.19.1.3 getEvents()

```
KeyBoard::Key gui::SFML::getEvents ( ) [override], [virtual]
```

Implements [gui::IRenderer](#).

### 4.19.1.4 getPluginName()

```
std::string gui::SFML::getPluginName ( ) const [inline], [override], [virtual]
```

Implements [gui::IPlugin](#).

#### 4.19.1.5 init()

```
void gui::SFML::init (
    const std::string & name,
    std::pair< const unsigned int, const unsigned int > resolution,
    unsigned int bitsPerPixel ) [override], [virtual]
```

Implements [gui::IRenderer](#).

#### 4.19.1.6 isRunning()

```
bool gui::SFML::isRunning ( ) [inline], [override], [virtual]
```

Implements [gui::IRenderer](#).

#### 4.19.1.7 render()

```
void gui::SFML::render ( ) [override], [virtual]
```

Implements [gui::IRenderer](#).

#### 4.19.1.8 setFPS()

```
void gui::SFML::setFPS (
    const unsigned int FPS ) [inline], [override], [virtual]
```

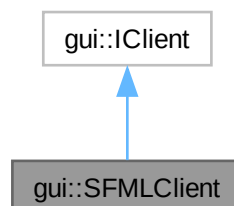
Implements [gui::IRenderer](#).

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

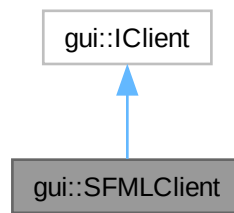
- lib/shared/Renderer/SFML/include/GUI/SFML.hpp

## 4.20 gui::SFMLClient Class Reference

Inheritance diagram for gui::SFMLClient:



Collaboration diagram for `gui::SFMLClient`:



### Public Member Functions

- `bool connect (uint16_t port, const std::string &machineName)` override
  - `void disconnect ()` override
  - `bool sendCommand (const std::string &cmd)` override
  - `bool getResponse (const std::string &cmd)` override
  - `std::string getResponse ()` override
  - `bool isConnected ()` override
- 
- virtual `bool connect (uint16_t port, const std::string &machineName)=0`
  - virtual `void disconnect ()=0`
  - virtual `bool sendCommand (const std::string &cmd)=0`
  - virtual `bool getResponse (const std::string &cmd)=0`
  - virtual `std::string getResponse ()=0`
  - virtual `bool isConnected ()=0`

## 4.20.1 Member Function Documentation

### 4.20.1.1 connect()

```
bool gui::SFMLClient::connect (
    uint16_t port,
    const std::string & machineName ) [override], [virtual]
```

Implements [gui::IClient](#).

### 4.20.1.2 disconnect()

```
void gui::SFMLClient::disconnect ( ) [inline], [override], [virtual]
```

Implements [gui::IClient](#).



#### 4.20.1.3 `getResponse()` [1/2]

```
std::string gui::SFMLClient::getResponse ( ) [override], [virtual]
```

Implements [gui::IClient](#).

#### 4.20.1.4 `getResponse()` [2/2]

```
bool gui::SFMLClient::getResponse (
    const std::string & cmd ) [override], [virtual]
```

Implements [gui::IClient](#).

#### 4.20.1.5 `isConnected()`

```
bool gui::SFMLClient::isConnected ( ) [override], [virtual]
```

Implements [gui::IClient](#).

#### 4.20.1.6 `sendCommand()`

```
bool gui::SFMLClient::sendCommand (
    const std::string & cmd ) [override], [virtual]
```

Implements [gui::IClient](#).

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

- `lib/shared/Renderer/SFML/include/GUI/SFMLClient.hpp`

## 4.21 myLib::Time Class Reference

### Public Member Functions

- **Time** (const double seconds)
- int **asSeconds** () const
- int **asMilliseconds** () const
- int **asMicroseconds** () const

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

- `lib/static/myLib/include/myLib/Clock/Time.hpp`



## Chapter 5

# File Documentation

### 5.1 IClient.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** IClient
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <cstdint>
00012
00013 namespace gui {
00014     class IClient {
00015     public:
00016
00017         virtual ~IClient() = default;
00018
00019         virtual bool connect(uint16_t port, const std::string& machineName) = 0;
00020         virtual void disconnect() = 0;
00021         virtual bool sendCommand(const std::string& cmd) = 0;
00022         virtual bool getResponse(const std::string& cmd) = 0;
00023         virtual std::string getResponse() = 0;
00024         virtual bool isConnected() = 0;
00025
00026     }; // class IClient
00027
00028 } // namespace gui
```

### 5.2 IPlugin.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** IPlugin
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011
00012 namespace gui {
00013     class IPlugin {
00014     public:
00015
00016         virtual ~IPlugin() = default;
00017
00018         [[nodiscard]] virtual std::string getPluginName() const = 0;
00019
00020     }; // class IPlugin
00021
00022 } // namespace gui
```

## 5.3 IRenderer.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** IRenderer
00006 */
00007
00008 #pragma once
00009
00010 #include "GUI/Abstraction/IPlugin.hpp"
00011 #include "GUI/Abstraction/IClient.hpp"
00012 #include "GUI/KeyBoard.hpp"
00013
00014 namespace gui {
00015
00016     class IRenderer : public IPlugin {
00017     public:
00018
00019         virtual void setFPS(unsigned int FPS) = 0;
00020
00021         [[nodiscard]] virtual IClient& getClient() = 0;
00022         [[nodiscard]] virtual bool isRunning() = 0;
00023
00024         virtual void init(const std::string &name, std::pair<const unsigned int, const unsigned
00025 int> resolution, unsigned int bitsPerPixel) = 0;
00026         virtual void render() = 0;
00027         virtual KeyBoard::Key getEvents() = 0;
00028         virtual void close() = 0;
00029
00030     }; // class IRenderer
00031
00032 } // namespace gui

```

## 5.4 Argument.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Argument
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011 #include <cstdint>
00012
00013 namespace gui {
00014
00015     class Argument {
00016     public:
00017
00018         Argument(const uint16_t p, std::string h) : port(p), hostName(std::move(h)) {};
00019         ~Argument() = default;
00020
00021         const uint16_t port;
00022         const std::string hostName;
00023
00024     }; // class Argument
00025
00026 } // namespace gui

```

## 5.5 Constant.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy | GUI
00004 ** File description:
00005 ** Constant.hpp
00006 */
00007
00008 /*
00009 ** @file Constant.hpp
00010 ** @brief Constants for the Zappy GUI
00011 ** @namespace gui

```

```

00012 */
00013
00014 #pragma once
00015
00016 #include <string_view>
00017
00018 namespace gui {
00019
00020     static constexpr const int EPITECH_EXIT_SUCCESS = 0;
00021     static constexpr const int EPITECH_EXIT_ERROR = 84;
00022
00023     static constexpr const std::string_view PLUGIN_RENDERER_SFML = "SFML";
00024
00025     static constexpr const int MAX_OCTETS_READ = 4096;
00026     static constexpr const int TIMEOUT = 20;
00027
00028     static constexpr const int MAX_PORT = 65535;
00029
00030     static constexpr const unsigned int DEFAULT_FPS = 80;
00031     static constexpr const unsigned int DEFAULT_BITS_PER_PIXEL = 64;
00032     static constexpr const std::pair<const unsigned int, const unsigned int> DEFAULT_RESOLUTION {1920,
1080};
00033     static constexpr const std::string_view DEFAULT_NAME = "ZAPPY";
00034
00035 } // namespace gui

```

## 5.6 Gui.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Gui
00006 */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include <vector>
00012
00013 #include "GUI/Abstraction/IRenderer.hpp"
00014 #include "GUI/Argument.hpp"
00015
00016 namespace gui {
00017
00018     class Gui {
00019
00020     public:
00021
00022         enum class RendererMode {
00023             GAME,
00024             SETTINGS
00025         };
00026
00027         explicit Gui(const Argument &args);
00028         ~Gui() = default;
00029
00030         std::unique_ptr<IRenderer>& getRenderer() { return m_renderer; };
00031
00032         void Run();
00033
00034         static std::vector<std::string> getData(const std::string &data);
00035
00036     private:
00037
00038         std::unique_ptr<IRenderer> m_renderer;
00039         std::vector<std::string> m_data;
00040         RendererMode m_mode{RendererMode::GAME};
00041
00042     }; // class Gui
00043
00044 } // namespace gui

```

## 5.7 Inventory.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:

```

```

00005 ** Inventory
00006 */
00007
00008 #pragma once
00009
00010 #include <vector>
00011
00012 #include "GUI/Inventory/Resource.hpp"
00013
00014 namespace gui {
00015
00016     class Inventory {
00017     public:
00018
00019
00020         Inventory(Resource food, Resource linemate, Resource deraumere, Resource sibur, Resource
mendiane, Resource phiras, Resource thystame) :
00021             resources({food, linemate, deraumere, sibur, mendiane, phiras, thystame}) {};
00022         Inventory(std::vector<Resource> cresources): resources(cresources) {};
00023
00024         std::vector<Resource> resources;
00025
00026     }; // class Inventory
00027
00028 } // namespace gui

```

## 5.8 Resource.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Resource.hpp
00006 */
00007
00008 #pragma once
00009
00010 #include "GUI/RunTimeException.hpp"
00011
00012 namespace gui {
00013
00014     class Resource {
00015     public:
00016
00017         enum class Type {
00018             FOOD,
00019             LINEMATE,
00020             DERAUMERE,
00021             SIBUR,
00022             MENDIANE,
00023             PHIRAS,
00024             THYSTAME,
00025             NONE
00026         };
00027
00028         Resource(Type type, unsigned int quantity);
00029
00030         bool operator==(const Resource &resource) const
00031         {
00032             return type == resource.type;
00033         }
00034
00035         Type type;
00036         double density;
00037         unsigned int quantity;
00038
00039     }; // class Resource
00040
00041 } // namespace gui

```

## 5.9 KeyBoard.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** KeyBoard
00006 */

```

```

00007
00008 #pragma once
00009
00010 namespace gui {
00011     class KeyBoard {
00012     public:
00013
00014         enum Key {
00015             NONE = -1, // Keep this at the beginning
00016             CLOSE = 0,
00017             KEY_LEFT = 1,
00018             KEY_RIGHT = 2,
00019             KEY_UP = 3,
00020             KEY_DOWN = 4,
00021             KEY_SPACE = 5,
00022             KEY_ENTER = 6,
00023             KEY_ESCAPE = 7,
00024             COUNT = 8 // corresponding to the size of the enum, keep this at the end
00025         };
00026     }; // class KeyBoard
00027 } // namespace gui

```

## 5.10 Parser.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Parser
00006 */
00007
00008 #pragma once
00009
00010 #include "GUI/Argument.hpp"
00011 #include "GUI/Gui.hpp"
00012 #include "GUI/Inventory/Inventory.hpp"
00013
00014 namespace gui {
00015     class Parser {
00016     public:
00017
00018         Parser() = default;
00019         ~Parser() = default;
00020
00021         static Argument ParseArgs(int argc, char* const argv[]);
00022
00023         static uint16_t ParsePort(const char* port);
00024         static std::string ParseMachineName(const char* machineName);
00025
00026         static void processData(std::vector<std::string> data, Gui &gui);
00027         static Inventory parseTileContent(std::string tileContent);
00028
00029         class ParserException : public std::exception
00030         {
00031         public:
00032
00033             explicit ParserException(std::string msg) : m_msg{std::move(msg)} {};
00034             ~ParserException() override = default;
00035
00036             ParserException(const ParserException &) = delete;
00037             ParserException &operator=(const ParserException &) = delete;
00038             ParserException(const ParserException &&) = delete;
00039             ParserException &operator=(const ParserException &&) = delete;
00040
00041             [[nodiscard]] const char *what() const noexcept override { return m_msg.c_str(); }
00042         };
00043
00044     private:
00045         std::string m_msg{0};
00046     }; // class ParserException
00047 } // namespace gui

```

## 5.11 PluginLoader.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy_gui
00004 ** File description:
00005 ** PluginLoader.hpp
00006 */
00007
00008 #include <unordered_map>
00009 #include <vector>
00010
00011 #include "GUI/Abstraction/IRenderer.hpp"
00012
00013 namespace gui {
00014
00015     class PluginLoader {
00016
00017     public:
00018
00019         using PluginCreator = std::unique_ptr<IPlugin> (*)();
00020
00021         ~PluginLoader() = default;
00022
00023
00024         static PluginLoader &getInstance() {
00025             static PluginLoader instance;
00026             return instance;
00027         }
00028
00029         template <typename T>
00030         std::unique_ptr<T> getPlugin(const std::string &pluginName);
00031
00032         void closePlugins();
00033
00034         class PluginLoaderException : public std::exception{
00035
00036         public:
00037
00038             explicit PluginLoaderException(std::string msg) : m_msg(std::move(msg)) {};
00039             [[nodiscard]] const char* what() const noexcept override { return m_msg.data(); };
00040
00041         private:
00042
00043             std::string m_msg;
00044
00045         }; // class PluginLoaderException
00046
00047     private:
00048
00049         PluginLoader() { loadPlugins(); };
00050
00051         void loadPlugins();
00052
00053         std::unordered_map<std::string, PluginCreator> m_plugins{0};
00054         std::vector<void*> m_handles{nullptr};
00055
00056     }; // class PluginLoader
00057
00058 } // namespace gui

```

## 5.12 Protocol.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Protocol
00006 */
00007
00008 #pragma once
00009
00010 #include <unordered_map>
00011 #include <string>
00012
00013 namespace gui {
00014
00015     enum class ProtocolKey {
00016         COUNT,
00017         MAP_SIZE,
00018         TILE_CONTENT,
00019         MAP_CONTENT,
00020         TEAMS_NAME,

```



```

00021     PLAYER_CONNECTION,
00022     PLAYER_POSITION,
00023     PLAYER_LEVEL,
00024     PLAYER_INVENTORY,
00025     EXPULSION,
00026     BROADCAST,
00027     INCANTATION_START,
00028     INCANTATION_END,
00029     FORK,
00030     RESOURCES_DROP,
00031     RESOURCES_COLLECT,
00032     PLAYER_DEATH,
00033     EGG_LAID,
00034     PLAYER_EGG_CONNECTION,
00035     EGG_DEATH,
00036     TIME_UNIT_REQUEST,
00037     TIME_UNIT_MODIFICATION,
00038     END_GAME,
00039     MESSAGE,
00040     UNKNOWN,
00041     UNKNOWN_PARAMETER,
00042     EGG_MATURE,
00043 };
00044
00045     const std::unordered_map<std::string, gui::ProtocolKey> ProtocolMap {
00046         {"msz", ProtocolKey::MAP_SIZE},
00047         {"bct", ProtocolKey::TILE_CONTENT},
00048         {"mct", ProtocolKey::MAP_CONTENT},
00049         {"tna", ProtocolKey::TEAMS_NAME},
00050         {"pnw", ProtocolKey::PLAYER_CONNECTION},
00051         {"ppo", ProtocolKey::PLAYER_POSITION},
00052         {"plv", ProtocolKey::PLAYER_LEVEL},
00053         {"pin", ProtocolKey::PLAYER_INVENTORY},
00054         {"pex", ProtocolKey::EXPULSION},
00055         {"pbc", ProtocolKey::BROADCAST},
00056         {"pic", ProtocolKey::INCANTATION_START},
00057         {"pie", ProtocolKey::INCANTATION_END},
00058         {"pfk", ProtocolKey::FORK},
00059         {"pdr", ProtocolKey::RESOURCES_DROP},
00060         {"pgt", ProtocolKey::RESOURCES_COLLECT},
00061         {"pdi", ProtocolKey::PLAYER_DEATH},
00062         {"enw", ProtocolKey::EGG_LAID},
00063         {"eht", ProtocolKey::EGG_MATURE},
00064         {"ebo", ProtocolKey::EGG_DEATH},
00065         {"pex", ProtocolKey::PLAYER_EGG_CONNECTION},
00066         {"sgt", ProtocolKey::TIME_UNIT_REQUEST},
00067         {"sst", ProtocolKey::TIME_UNIT_MODIFICATION},
00068         {"seg", ProtocolKey::END_GAME},
00069         {"smg", ProtocolKey::MESSAGE},
00070         {"suc", ProtocolKey::UNKNOWN},
00071         {"sbp", ProtocolKey::UNKNOWN_PARAMETER},
00072     };
00073
00074     class Protocol {
00075     public:
00076
00077         [[nodiscard]] static std::string getCommand(ProtocolKey key);
00078         [[nodiscard]] static ProtocolKey getKey(const std::string &command);
00079
00080     }; // class Protocol
00081
00082 } // namespace gui
00083

```

## 5.13 RunTimeException.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Zappy_gui
00004 ** File description:
00005 ** RunTimeException.hpp
00006 */
00007
00008 #pragma once
00009
00010 #include <string>
00011
00012 namespace gui {
00013
00014     class RunTimeException : public std::exception
00015     {
00016     public:
00017

```

```

00018         explicit RunTimeException(std::string msg) : m_msg{std::move(msg)} {};
00019         ~RunTimeException() override = default;
00020
00021         RunTimeException(const RunTimeException &) = delete;
00022         RunTimeException &operator=(const RunTimeException &) = delete;
00023         RunTimeException(const RunTimeException &&) = delete;
00024         RunTimeException &operator=(const RunTimeException &&) = delete;
00025
00026         [[nodiscard]] const char *what() const noexcept override { return m_msg.c_str(); };
00027
00028     private:
00029
00030         std::string m_msg{0};
00031
00032 }; // class RunTimeException
00033
00034 } // namespace gui

```

## 5.14 Map.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Map.hpp
00006 */
00007
00008 #pragma once
00009
00010 namespace gui {
00011
00012     class Map {
00013
00014     public:
00015
00016         Map(unsigned int width, unsigned int height) : m_width(width), m_height(height) {};
00017         ~Map();
00018
00019         unsigned int getWidth() const { return m_width; };
00020         unsigned int getHeight() const { return m_height; };
00021
00022     private:
00023
00024         unsigned int m_width;
00025         unsigned int m_height;
00026
00027     }; // class Map
00028
00029 } // namespace gui

```

## 5.15 Mob.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Mob
00006 */
00007
00008 #pragma once
00009
00010 #include "GUI/Inventory/Inventory.hpp"
00011 #include "GUI/Position.hpp"
00012
00013 namespace gui {
00014
00015     class Mob {
00016
00017     public:
00018
00019         enum class Action {
00020             MOVE,
00021             FEED,
00022             ELEVATE,
00023             NONE
00024         };
00025
00026         Action getAction() const { return m_action; };
00027         Inventory& getInventory() { return m_inventory; };

```

```

00028         Position& getPosition() { return m_position; };
00029         unsigned int getLevel() const { return m_level; };
00030
00031         void setAction(const Action &action) { m_action = action; };
00032
00033         void levelUp() { m_level++; };
00034
00035         private:
00036
00037         Action m_action{Action::NONE};
00038         Inventory m_inventory;
00039         Position m_position;
00040         unsigned int m_level{1};
00041
00042     }; // class Mob
00043
00044 } // namespace gui

```

## 5.16 Position.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Position
00006 */
00007
00008 #pragma once
00009
00010 #include <utility>
00011
00012 namespace {
00013
00014     class Position {
00015
00016     public:
00017
00018         Position(unsigned int x, unsigned int y) : x(x), y(y) {};
00019         ~Position();
00020
00021         unsigned int x;
00022         unsigned int y;
00023
00024     }; // class Position
00025
00026 } // namespace

```

## 5.17 SFML.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** SFML
00006 */
00007
00008 #pragma once
00009
00010 #include <SFML/Graphics.hpp>
00011 #include <array>
00012
00013 #include "GUI/Abstraction/IRenderer.hpp"
00014 #include "GUI/SFMLClient.hpp"
00015 #include "GUI/KeyBoard.hpp"
00016
00017 namespace gui {
00018
00019     class SFML : public IRenderer {
00020
00021     public:
00022
00023         SFML() = default;
00024         ~SFML() override = default;
00025
00026         void setFPS(const unsigned int FPS) override { m_window.setFramerateLimit(FPS); };
00027
00028         [[nodiscard]] std::string getPluginName() const override { return
00029             PLUGIN_RENDERER_SFML.data(); };
00029         [[nodiscard]] IClient& getClient() override { return m_client; };

```

```

00030         [[nodiscard]] KeyBoard::Key getEvents() override;
00031         [[nodiscard]] bool isRunning() override { return m_window.isOpen() &&
checkConnection(m_timeoutClock); };
00032
00033         void init(const std::string &name, std::pair<const unsigned int, const unsigned int>
resolution,
unsigned int bitsPerPixel) override;
00034         void close() override { m_window.close(); getClient().disconnect(); };
00035         void render() override;
00036
00037         static KeyBoard::Key getKeyboardEvent(const sf::Event &event);
00038         bool checkConnection(sf::Clock clock);
00039
00040     private:
00041
00042         sf::RenderWindow m_window;
00043         SFMLClient m_client;
00044         sf::Clock m_timeoutClock;
00045
00046         static std::array<gui::KeyBoard::Key, sf::Keyboard::KeyCount> KEY_CODE_ARRAY;
00047
00048     }; // class SFML
00049
00050 } // namespace sfml

```

## 5.18 SFMLClient.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** zappy_gui
00004 ** File description:
00005 ** Client.hpp
00006 */
00007
00008 #pragma once
00009
00010 #include <SFML/Network.hpp>
00011
00012 #include "GUI/Abstraction/IClient.hpp"
00013 #include "GUI/Constant.hpp"
00014
00015 namespace gui {
00016
00017     class SFMLClient : public IClient {
00018
00019     public:
00020
00021         ~SFMLClient() override = default;
00022
00023         bool connect(uint16_t port, const std::string &machineName) override;
00024         void disconnect() override { m_socket.disconnect(); };
00025
00026         bool sendCommand(const std::string &cmd) override;
00027         bool getResponse(const std::string &cmd) override;
00028         std::string getResponse() override;
00029
00030         [[nodiscard]] bool isConnected() override;
00031
00032     private:
00033
00034         sf::TcpSocket m_socket{};
00035
00036     }; // class Client
00037
00038 } // namespace gui

```

## 5.19 Clock.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** myLib | Clock
00004 ** File description:
00005 ** Clock.hpp
00006 */
00007
00008 /*
00009 ** @file Clock.hpp
00010 ** @brief Clock class for time management
00011 ** @namespace myLib
00012 */

```

```

00013
00014 #pragma once
00015
00016 #include <chrono>
00017
00018 #include "myLib/Clock/Time.hpp"
00019
00020 /*
00021 ** @brief TimePoint is a type alias for a time point which is a very long and complicated type in the
00022 standard library
00023 */
00024 using TimePoint = std::chrono::time_point<std::chrono::high_resolution_clock>;
00025 namespace myLib {
00026
00027     /*
00028     ** @brief Class for time management
00029     */
00030     class Clock {
00031     public:
00032
00033         Clock() : m_start(std::chrono::high_resolution_clock::now()) {};
00034
00035         ~Clock() = default;
00036
00037         /*
00038         ** @brief Restart the clock
00039         */
00040         void restart() { m_start = std::chrono::high_resolution_clock::now(); };
00041
00042         /*
00043         ** @brief Pause the clock
00044         */
00045         void pause();
00046
00047         /*
00048         ** @brief Resume the clock
00049         */
00050         void resume();
00051
00052         /*
00053         ** @brief Get the elapsed time since the last restart
00054         ** @return Time The elapsed time
00055         */
00056         [[nodiscard]] Time getElapsedTime() const;
00057
00058     private:
00059
00060         /*
00061         ** @property The start time
00062         */
00063         TimePoint m_start;
00064
00065         /*
00066         ** @property The pause time
00067         */
00068         TimePoint m_pause;
00069
00070         /*
00071         ** @property The "is in pause" boolean variable
00072         */
00073         bool m_paused{false};
00074
00075     }; // Clock
00076 } // namespace myLib
00077
00078 } // namespace myLib

```

## 5.20 Time.hpp

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** myLib | Clock
00004 ** File description:
00005 ** Time.hpp
00006 */
00007
00008 /*
00009 ** @file Time.hpp
00010 ** @brief Class for time management
00011 ** @namespace myLib
00012 */
00013

```

```

00014 #pragma once
00015
00016 namespace myLib {
00017
00018     /**
00019     ** @class Time
00020     ** @brief Class used for time management
00021     */
00022     class Time {
00023     public:
00024
00025         /**
00026         ** @brief Construct a new Time object
00027         */
00028         explicit Time(const double seconds) : m_seconds(seconds) {};
00029
00030         /**
00031         ** @brief Transform the time to seconds
00032         ** @return int The time in seconds
00033         */
00034         [[nodiscard]] int asSeconds() const { return static_cast<int>(m_seconds); };
00035
00036         /**
00037         ** @brief Transform the time to milliseconds
00038         ** @return int The time in milliseconds
00039         */
00040         [[nodiscard]] int asMilliseconds() const { return static_cast<int>(m_seconds * 1000); }
00041
00042         /**
00043         ** @brief Transform the time to microseconds
00044         ** @return int The time in microseconds
00045         */
00046         [[nodiscard]] int asMicroseconds() const { return static_cast<int>(m_seconds * 1000000); }
00047     };
00048
00049     private:
00050
00051         /**
00052         ** @property The time in seconds
00053         */
00054         double m_seconds{0.0F};
00055
00056     }; // Time
00057
00058 } // namespace myLib

```

## 5.21 Random.hpp

```

00001 /**
00002 ** EPITECH PROJECT, 2024
00003 ** myLib
00004 ** File description:
00005 ** Random.hpp
00006 */
00007
00008 /**
00009 ** @file Random.hpp
00010 ** @brief Class for random number generation
00011 ** @namespace myLib
00012 */
00013
00014 #pragma once
00015
00016 #include <random>
00017
00018 namespace myLib {
00019
00020     /**
00021     ** @class Random
00022     ** @brief Class for random number generation
00023     */
00024     class Random {
00025     public:
00026
00027         /**
00028         ** @brief Generate a random integer between min and max
00029         ** @param min The minimum value
00030         ** @param max The maximum value
00031         ** @return int The random integer
00032         */
00033         static int randomInt(int min, int max);
00034

```

```
00035         static int randomInt() { return randomInt(-1000, 1000); };
00036
00037         /*
00038         ** @param min The minimum value
00039         ** @param max The maximum value
00040         ** @return float The random float
00041         */
00042         static float randomFloat(float min, float max);
00043         static float randomFloat() { return randomFloat(-1.0f, 1.0f); };
00044
00045     }; // class Random
00046
00047 } // namespace myLib
```





# Index

close  
    gui::SFML, 18

connect  
    gui::SFMLClient, 20

disconnect  
    gui::SFMLClient, 20

getClient  
    gui::SFML, 18

getEvents  
    gui::SFML, 18

getPluginName  
    gui::SFML, 18

getResponse  
    gui::SFMLClient, 20, 21

gui::Argument, 7

gui::Gui, 7

gui::IClient, 8

gui::Inventory, 9

gui::IPlugin, 9

gui::IRenderer, 10

gui::KeyBoard, 11

gui::Map, 11

gui::Mob, 11

gui::Parser, 12

gui::Parser::ParserException, 12

gui::PluginLoader, 13

gui::PluginLoader::PluginLoaderException, 14

gui::Protocol, 15

gui::Resource, 15

gui::RunTimeException, 16

gui::SFML, 17

    close, 18

    getClient, 18

    getEvents, 18

    getPluginName, 18

    init, 18

    isRunning, 19

    render, 19

    setFPS, 19

gui::SFMLClient, 19

    connect, 20

    disconnect, 20

    getResponse, 20, 21

    isConnected, 21

    sendCommand, 21

include/GUI/Abstraction/IClient.hpp, 23

include/GUI/Abstraction/IPlugin.hpp, 23

include/GUI/Abstraction/IRenderer.hpp, 24

include/GUI/Argument.hpp, 24

include/GUI/Constant.hpp, 24

include/GUI/Gui.hpp, 25

include/GUI/Inventory/Inventory.hpp, 25

include/GUI/Inventory/Resource.hpp, 26

include/GUI/KeyBoard.hpp, 26

include/GUI/Parser.hpp, 27

include/GUI/PluginLoader.hpp, 28

include/GUI/Protocol.hpp, 28

include/GUI/RunTimeException.hpp, 29

init  
    gui::SFML, 18

isConnected  
    gui::SFMLClient, 21

isRunning  
    gui::SFML, 19

lib/shared/Renderer/SFML/include/GUI/Map.hpp, 30

lib/shared/Renderer/SFML/include/GUI/Mob.hpp, 30

lib/shared/Renderer/SFML/include/GUI/Position.hpp, 31

lib/shared/Renderer/SFML/include/GUI/SFML.hpp, 31

lib/shared/Renderer/SFML/include/GUI/SFMLClient.hpp, 32

lib/static/myLib/include/myLib/Clock/Clock.hpp, 32

lib/static/myLib/include/myLib/Clock/Time.hpp, 33

lib/static/myLib/include/myLib/Random.hpp, 34

myLib::Clock, 7

myLib::Random, 15

myLib::Time, 21

render  
    gui::SFML, 19

sendCommand  
    gui::SFMLClient, 21

setFPS  
    gui::SFML, 19