Documentation

IDisplayModule - Methods

void open() :

Open the window where the program will draw

void close() :

Close the window

bool isOpen() :

Tell if the window is currently open or close

void putRectFill(Color color, Coord size, Coord pos) :

Draw a full rectangle in the window

void putRectOutline(Color color, Coord size, Coord pos) :

Draw the outline of a rectangle in the window

void putCircle(Color color, Coord pos, size_t radius) :

Draw a full circle in the window

void putText(Color color, Coord size, std::string const &value) :

Draw a text in the window

void displayScreen() :

Display in the window everything that has been draw

void refreshScreen() :

Refresh the window

void clearScreen() :

Clear the window

bool isKeyPress(const KeyList key) :

Tell if a certain key is currently pressed

bool isMouseClicked() :

Tell if a click is currently made with the mouse

Coord getMousePos() :

Give the currently coordinate of the cursor in the window

IDisplayModule - Enums

Color :

List of all the color, which can be used for drawing

KeyList :

List of all the keys

Special :

- NEXT_GAME : Change the game for the next one in the list (Key Z)
- PREV_GAME : Change the game for the previous one in the list (Key A)
- NEXT_LIB : Change the graphic library for the next one in the list (Key X)
- PREV_LIB : Change the graphic library for the previous one in the list (Key W)
- RESTART_GAME : Restart the current played game (Key R)
- MENU : Send to the menu (Key Q)
- EXIT : Leave the program (Key Escape)
- PAUSE : Pause the game (Key P)

IDisplayModule - Implementation

Inheritance

First of all, your module must inherit from the interface IDisplayModule, which is in the arcade namespace.

After that, you must implement all the declared function, for your library to work.

Entrypoint

To be able to use your graphic library, you must declare a function named entryPoint, like this :

```
extern "C"
{
	std::shared_ptr<IDisplayModule> entryPoint()
	{
		return std::make_shared<YOUR_MODULE_NAME>();
	}
}
```

Enums

To use the enums, Color and KeyList, you must link them with the corresponding element in the graphic library you want to implement.

Compilation

When you have finished everything, or for testing, you have to compile your module into a shared file (.so). Then, give it a name like "arcade_YOUR_MODULE_NAME.so", and put it in the lib folder, which is at the root. And, finally, add its name in the deque GRAPHIC_LIB_NAMES, in CoreConfig.cpp (./src/CoreConfig/CoreConfig.cpp)

IGameModule - Methods

GameStatus getStatus() :

Get the current status of the game

void refresh() :

Update the game

void reset() :

Reset the game

size_t getScore() :

Get the current score of the player for the currently played game

size_t getScoreHigh() :

Get the current high score of the player for the currently played game

void setUsername(std::string const &username) :

Set the username of the player

const IDisplayModule &getDisplayModule() :

Get the display module, which is currently linked with the game

void setDisplayModule(IDisplayModule &displayModule) :

Set the display module linked with the game

IGameModule - Enums

GameStatus :

List of all the possible status that the game can have

Special :

- SUCCESS : The game is played and everything is fine
- ERROR : The game is played and there is an error
- GAMEOVER : The game is over (Victory and defeat include)
- PAUSE : The game is in pause

IGameModule - Implementation

Inheritance

First of all, your module must inherit from the interface IGameModule, which is in the arcade namespace.

After that, you must implement all the declared function, for your library to work.

For your game to be functional, you probably will have to add other methods, but don't forget that only the methods in the interface will be called.

Entrypoint

To be able to use your game library, you must declare a function named entryPoint, like this :

```
extern "C"
{
	std::shared_ptr<IGameModule> entryPoint()
	{
		return std::make_shared<YOUR_MODULE_NAME>();
	}
}
```

Compilation

When you have finished everything, or for testing, you have to compile your module into a shared file (.so). Then, give it a name like "arcade_YOUR_MODULE_NAME.so", and put it in the lib folder, which is at the root. And, finally, add its name in the deque GAME_LIB_NAMES, in CoreConfig.cpp (./src/CoreConfig/CoreConfig.cpp)