Arcade Documentation

1) What is Arcade?

Arcade is a video game project. It consists of three main parts. Game library, Graphical library and a Core that links and uses all the parts together. Game library and graphical library must be totally independent and be linked thanks to the core.

2) How to use the Arcade

First of all, you must download the project and store it locally on your machine. You need to be able to compile with Makefiles as the project works with them. You also need to install C++, Ncurses, SDL2 and SFML libraries as the project needs them to work properly. Once you possess every requirement you are able to build the project thanks to the command “make”. To use it, you need to launch the arcade binary with a graphic library :

```
./arcade [GRAPHICAL]
        => ./arcade lib/arcade_ncurses.so
```

3) How to add a new library?

You need to create a Makefile for your library. You also need to add it to the main Makefile in order to make it compile with every other library. Then, you need to implement the library you want to build with the interface created by Gabriel Ravien and Jeffrey Winkler. Everything you need to know is well explained in this Github:

https://github.com/GabrielRavier/arcade-interface

a) Game library

In order to implement a game library you need to implement every function needed by the IGameModule. There are only three without the Constructor and Deconstructor:

- An init() function that will create all the textures, open the window, and set all the needed settings.
- An update() function that will get every information that the Graphical library collected and use it to update his own information.
- And a draw() function that will decide of what to draw on the screen

b) Graphical library

For the graphical library you need to implement the IDisplayModule interface. Much more functions but a lot of them are easier. Here we're just going to focus on the most important:

- openWindow() is a function that open a window in the graphical library of your choice
- loadTexture() will create a texture in the library of your choice and return a unique_ptr, so it can be used without caring of the type of texture
- drawSprite() will draw sprite on the opened window
- displaySprite() will display on the window every sprites that have been called with drawSprite()

If you need more information on how to implement the interfaces, you should consult the Github where everything is from linked above.

If you want to implement a Core part. You need to implement an ICore interface also linked in the Github above