Documentation : Arcade project

Summary:

- How to use this program?
- How to implement a graphical library?
- How to implement a game library?
- Class diagram of this project

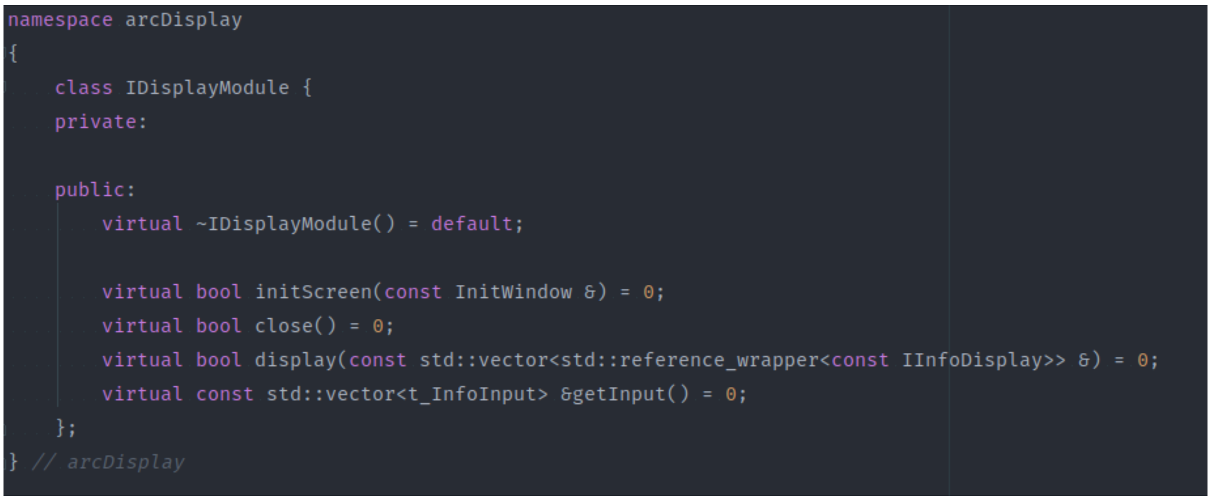
1/ How to use this program?

This program consist to link a core with two libraries (a graphical library and a game library) by using this sentence: ./arcade “Path_To_Your_Graphical_Library”
After that, you’ll be able to choose your game and switch graphical library. During your game, you’re able to switch graphical and game library.

2/ How to implement a graphical library?

To implement a new graphical library, you have to create four mandatory methods following this interface :

- initScreen: Initlialize your screen from the initWindow data and returning a boolean if this one success or not
- close: close your window and returning a boolean if this one success or not
- display: display all objects (IInfoDisplay) from the core that we provided. Returning a boolean if the window has been closed
- getInput: returning all inputs pressed by the user.

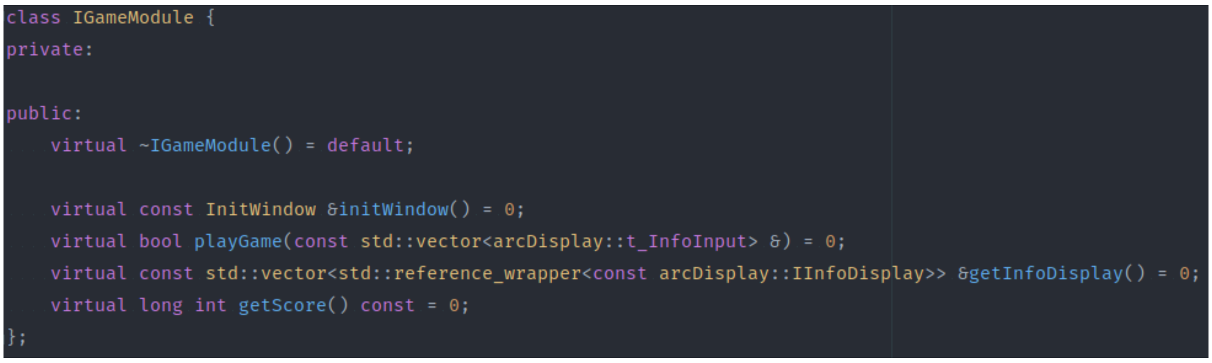


```cpp
namespace arcDisplay
{
    class IDisplayModule {
    private:

    public:
        virtual ~IDisplayModule() = default;

        virtual bool initScreen(const InitWindow &) = 0;
        virtual bool close() = 0;
        virtual bool display(const std::vector<std::reference_wrapper<const IInfoDisplay>> &) = 0;
        virtual const std::vector<t_InfoInput> &getInput() = 0;
    };
} // arcDisplay
```

3/ How to implement a game library?

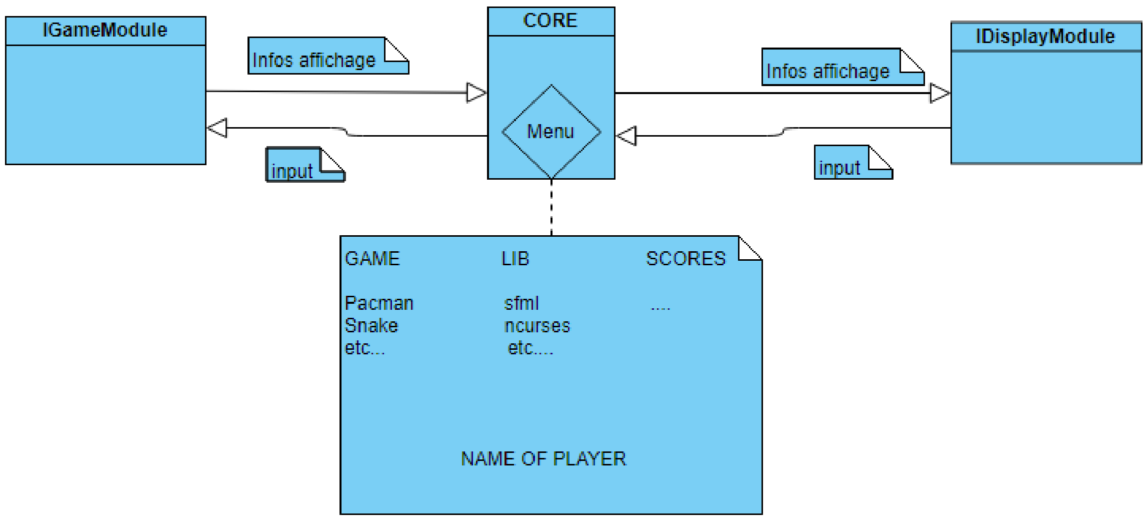
To implement a new game library, you have to create 4 mandatory methods following this interface :

- InitWindow: send every necessary informations inside a InitWindow class to initialize the window in your graphical lib
- playGame: manage all actions and inputs according to the game
- getInfoDisplay: return all objects to be displayed by the graphic library
- getScore: return the score



```
class IGameModule {
private:

public:
    virtual ~IGameModule() = default;

    virtual const InitWindow &initWindow() = 0;
    virtual bool playGame(const std::vector<arcDisplay::t_InfoInput> &) = 0;
    virtual const std::vector<std::reference_wrapper<const arcDisplay::IInfoDisplay>> &getInfoDisplay() = 0;
    virtual long int getScore() const = 0;
};
```

4/ Class diagram


IGameModule
Infos affichage
input
CORE
Menu
Infos affichage
input
IDisplayModule
GAME
Pacman
Snake
etc...
LIB
sfml
ncurses
etc....
SCORES
....
NAME OF PLAYER