Usage

```
./arcade <graphical_libraries.so>
```

Key Binding

- Menu
  - **↑↓** : *Navigate*
  - **Enter** : *Validate*

- Esc : *Quit*

- Game
  - **↕↔** : *Navigate*
  - **Esc** : *Quit*
  - **P** : *Pause*
  - **Del** : *Menu*
  - **R** : *Restart*

Implementation

Graphical

General Information

Graphical lib should be use to

- Display Game / Menu / Game Over
- Get Key strokes
- Get some Information such as the Pseudonyme

Some function are not**essential** to make it works, but could be use to**ease** the usage of libs.

Method

- `IGraphicalLib *creatorGraph()`
  - Init Graphical Lib
    - Return New Object
  - `DisplayMenu(vector\<string>, vector\<string> MenuData, bool)`
  - Display Menu
    - Params :

        - Graphical Libs list
        - Game libs list
        - Struct containing information about Menu (*see structinfo.hpp*)
        - Boolean that could be use to Refresh the game `true` = refresh)
    - Return 0. 1 if the game start.

- `DisplayBoard(vector\<vector\>, map\<string, string>, int)`
  - Display Game
    - Params:
      - Board, create by the Game (*See Game parts*)
      - map with general information(pseudo, game)
      - Actual Score
    - Return 1, -1 on quit

- `DisplayGameOver(string, int)`
  - Display Game Over Screen
    - Params:
      - Pseudonyme
      - Score

- `Quit()`
  - Delete the graphical Class

- `GetData(MenuData)`
  - Get Menu Informations
    - Params:
      - struct containing information about Menu
    - Return the struct updates

- `GetKey()`
  - Return key pressed.

- `InitAsset(string path)`
  - Load Assets stock in `path`
    - Params:
      - path string

- `GetMapping()`
  - Return Key Binding use by the lib
    - Return map of `Interaction::Bind` and key associate by Graphicals libs

- `displayPauseScreen()`

    - Display Pause screen in Game

- `setStart(bool)`

    - Set Start status on `bool` value

Game

General Information

Each Game need to have a assets/ folder where assets will be stocked using the following method : - Each Types of cells are linked to a folder path (ex: *type 1 = /player folder, which will contains assets for player*) - Folder should containing at least 2 types of files : - ***sprite.png*** for each graphicals libs which will use this format - ***ascii*** which contains only ascii information, use for certain graphicals libs

Method

- `IGraphicalLib *creatorGame()`

    - Init Game Lib
        - Return New Object

- `Refresh()`

    - Main function, which is called every frame rates to update the Game.
        - Return `vector\<vector\>`
            - This **board** could have any size you need.
            - This **board** is divide by ***cells*** containing `AssetInfo` struct
            - `AssetInfo` is a struct containing the type of the cells
            - a path link to the Asset to display
            - the orientation of the asset

- `Move(Interaction::Bind)`

    - Use to interact with the game according to your**bind**

- `Quit()`

    - Delete the Game Instance

- `GameOver()`

    - Return a Boolean to check if the game is over.

- `GetScore()`

    - Return the actual Score.

- `GetName()`

  - Return the name of the **game**
- `GetAsset()`
  - Return the path where the Assets are stocked.