DOCUMENTATION

Projet Arcade

Projet réalisé par :

Chloé CHAUVIN
Quentin TREHEUX
Emma RULLIERE

Table des matières

Introduction

Ce programme consiste à permettre à l'utilisateur de jouer aux jeux compatibles avec la librairie graphique compatibles qu'il souhaite. Pour lancer le programme, il suffit de taper cette commande sur votre terminal :

- ./arcade ./lib/arcade_[« la librairie graphique de votre choix »].so

Pour plus d'informations, veuillez entrer cette commande :

- ./arcade -h

Pour que le programme puisse marcher, vous devez impérativement avoir sur votre ordinateur la bibliothèque Simple DirectMedia Layer (SDL2). Si ce n'est pas le cas, vous pouvez l'installer avec ces commandes :

- sudo yum install SDL2_ttf-devel
- sudo yum install SDL2_image-devel

La suite de cette documentation vous explique comment créer une librairie graphique ou de jeu compatible au programme.

Ajouter une librairie de jeu au programme

Pour qu'une librairie de jeu soit compatible avec le programme, votre librairie de jeu devra hériter de l'interface IGames (Game/include/IGame.hpp) :

```
14  namespace game {
15      class IGame {
16          public:
17              ~IGame() = default;
18              virtual void play(std::string direction) noexcept = 0;
19              virtual bool checkDirection(std::string direction, int i, int y) noexcept = 0;
20              virtual void moveF(std::string direction, int i, int y) noexcept = 0;
21              virtual void enemy() noexcept = 0;
22              virtual std::vector<std::string> getMap(void) const noexcept = 0;
23              virtual int getScore(void) const noexcept = 0;
24              virtual int getOver(void) const noexcept = 0;
25              virtual void reset(void) noexcept = 0;
26
27          protected:
28          private:
29      };
30  }
```

Dans notre système, la seule information transmise par le Core de la librairie du jeu à la librairie graphique est la matrice contenue dans un vector de strings (std::vector<std::string>).

Cette matrice une fois transmise à la librairie graphique sera affiché.

Votre librairie de jeu pour être compatible devra recevoir dans la fonction play une direction représentant celle choisi par le joueur, soit « right », « left », « up » ou encore « down ». Avec cette unique information, votre librairie de jeu doit pouvoir déplacer les ennemies ou le joueur dans la matrice. La fonction play sera donc continuellement appelé par le Core avec une des quatres directions possibles pour faire avancer la partie peu importe la librairie graphique choisie.

Ajouter une librairie graphique au programme

Pour qu'une librairie graphique soit compatible avec le programme, elle devra hériter de l'interface ILibGraph (GraphicLib/include/ILibGraphic.hpp) :

```cpp
namespace lib {
    enum key
    {
        space = 32,
        escape = 27,
        backspace = 8,
        arrowUp = -1,
        arrowDown = -2,
        arrowLeft = -3,
        arrowRight = -4,
        Z = 122,
        Q = 113,
        S = 115,
        D = 100,
        null = 0
    };

    class ILibGraph {
        public:
            ~ILibGraph() = default;
            virtual int Menu(std::vector<std::string> game, std::vector<std::string> lib, std::vector<std::string> score) noexcept = 0;
            virtual void Game(std::vector<std::string> map) noexcept = 0;
            virtual void DisplayScore(int score) noexcept = 0;
            virtual int GetKey() noexcept = 0;
            virtual std::string GetName() const noexcept = 0;
            virtual bool GetStatus() const noexcept = 0;

        protected:
        private:
    };
}
```

La fonction Menu reçoit en paramètre la liste des jeux et des librairies graphique ainsi que les scores dans des vector de strings (std::vector<std::string>) cette fonction doit donc display toutes ces informations et retourner au Core la position du jeu choisi dans le vector.

La fonction Game quant à elle reçoit la matrice et se charge de l'afficher avec le score à l'aide de la fonction DisplayScore.

Pour finir GetKey surveille les touches de jeux et si elles sont enfoncées par le joueur et return un int avec une valeur correspondant pour permettre au Core de transmettre l'information à la librairie de jeu (dans notre cas la fonction play de l'interface IGame).

Nous contacter

Si vous souhaitez obtenir des informations supplémentaires, vous pouvez nous contactez via ces adresses électroniques :

emma.rulliere@epitech.eu

chloe.chauvin@epitech.eu

quentin.treheux@epitech.eu