Documentation :

Comment implementer une librairie graphique ?

- Elle doit pourvoir recevoir des events et les renvoyer à travers la méthode “registerEvents” qui renvoi un “vector<strings>”
- Gérer le display de la carte, des joueurs et/ou des différents IA de la fenêtre dans une méthode “display” prenant en paramètre un “vector<vector<strings>>” et un “int”
- Ouvrir une simple fenêtre, à travers la méthode “createWindow”
- Contenir deux méthodes dîtes externes, create prenant en paramètre un “vector<vector<strings>>” contenant le nom de librairie, le nom de l’utilistateur... La seconde méthode, “destroy”, prends en paramètre l’objet précédemment crée afin de le Supprimer

La librairie graphique devra être placée dans un dossier comportant le nom de la librairie, qui se trouvera dans le dossier lib. Voici un exemple: “lib/NCURSES/Ncurses.cpp”.

Interface IModuleGraph

```
class IModuleGraph {
    public:
        virtual ~IModuleGraph() = default;
      Julien Salies, 18 days ago • [ADDED] lib // IModuleGraph Added
        virtual void createWindow() = 0;
        virtual void display(std::vector<std::vector<std::string>>, int) = 0;
        virtual std::string registerEvents() = 0;
        //virtual void destructWindow() = 0;

    protected:
    private:
};
```

Comment ajouter une librairie de jeu ?

- Elle doit gérer le déplacement des différentes entités du jeu
- Vérifier si le joueur à gagner ou non et le renvoyer au programme principal
- Renvoyer différentes informations tel que : le score, l’état du jeu (gagné, en cours ou perdu) et la carte de jeu. Ces méthodes devront s’appeler respectivement : “getScore” (renvoyant un “size_t”), “getState” (renvoyant “game::state”) et “get Map” (renvoyant un

“vector<strings>”)

- Elle doit pouvoir déterminer aussi le statut du jeu (gagné, en cours ou perdu) avec une méthode s’appelant “setState” prenant en paramètre l’état du jeu (“game::state”)
- Elle doit contenir aussi deux méthodes dîtes externes, create et destroy. La méthode externe destroy prend en paramètre l’objet précédemment crée.
- Le jeu devra avoir une carte préalablement crée, au format .txt, et mise dans le dossier (voir ci-dessous) avec les autres fichiers

La librairie de jeu devra être placée dans un dossier comportant le nom du jeu, qui se trouvera dans le dossier “games”. Voici un exemple: “game/Pacman/Pacman.cpp”.

Il faudra aussi ajouter dans le dossier “Score.txt” (situé à la racine du projet) une ligne avec le nom de votre jeu suivi d’une flèche et un zéro (qui correspond au high score) comme ceci : “VOTREJEU -> 0”

Interface IModuleGame:

```cpp
namespace game {

    enum state
        {
            WIN,
            LOOSE,
            RUNNING
        };

    lucas simao, 2 days ago | 1 author (lucas simao)
    class IModuleGame {
        public:
            virtual ~IModuleGame() = default;

            virtual void moveEnemy() = 0;
            virtual void init() = 0;
            virtual void MoveForward() = 0;
            virtual void MoveBackward() = 0;
            virtual void MoveLeft() = 0;
            virtual void MoveRight() = 0;
            virtual void update() = 0;
            virtual size_t getScore() const = 0;
            virtual size_t getHP() const = 0;
            virtual std::string getHighScore() const = 0;
            virtual game::state getState() const = 0;
            virtual void setState(game::state) = 0;
            virtual std::vector<std::string> getMap() const = 0;

        protected:
        private:
    };
}
```

Comment jouer aux jeux ?

Après avoir ajouté votre librairie graphique et votre jeu, il est maintenant possible de jouer à celui-ci.

Pour jouer il faudra connaître quelques touches de commandes :

- Flèche du haut : permet de passer au jeu suivant
- Flèche du bas : permet de passer au jeu précèdent
- Flèche de droite : permet de passer à la librairie graphique suivante
- Flèche du bas : permet de passer à la librairie graphique suivante
- Z : permet d'avancer le personnage en jeu
- S : permet de reculer le personnage en jeu
- Q : permet d'aller à droite
- Z : permet d'aller à gauche
- Echap : permet de revenir au menu si vous êtes en jeu, ou de quitter le programme si vous êtes dans le menu

Vous pourrez ensuite jouez à tous les jeux en essayant d'être meilleur à chaque partie. Vous pourrez voir votre progression grâce à votre meilleur score qui sauvegardé et actualisé après chaque partie.

Documentation écrite par :

Quentin Muratorio

Lucas Simao

Julien Salies