DOCUMENTATION ARCADE

I – Faire une librairie graphique

Pour créer une librairie graphique compatible avec l'Arcade, créez une classe héritant de l'interface ***IDisplayModule*** dans un .hpp et surchargez dans un fichier .cpp les méthodes suivantes :

```
bool createAsset(const std::string &path, const std::string &assetName);
Méthode qui permet de créer un asset et de le stocker dans la map sprites
      path : le chemin du fichier contenant l'asset
      assetName : le nom de l'asset
      return true si l'asset a bien été créé, false sinon

bool drawAsset(const std::string &assetName, int x, int y);
Méthode qui permet d'afficher un asset aux coordonnées fournies
       assetName : le nom de l'asset
      x : coordonnée abscisse
      y : coordonnée ordonnée
      return true si l'asset a bien été affiché, false sinon

bool drawText(const std::string &textName, int x, int y);
Méthode qui permet d'afficher du texte aux coordonnées fournies
      textName : le nom du texte
      x : coordonnée abscisse
      y : coordonnée ordonnée
      return true si le texte a bien été affiché, false sinon

void refreshWindow();
Méthode qui permet de rafraichir l'affichage

void clearScreen();
Méthode qui permet de nettoyer l'affichage
```

```
   bool createText(const std::string &text, const std::string &assetName);
Méthode qui permet de créer du texte et de le stocker dans la map sprites
      text : le texte à stocker
      assetName : le nom du texte
      return true si le texte a bien été créé, false sinon

   e_event catchEvent();
Méthode qui permet de catch des events dans le programme
      return un displayModule::e_event

   void createSound(const std::string &path, const std::string &soundKey);
Créé un son

   void startSound(const std::string &soundKey);
Lance la lecture d'un son

   void stopSound(const std::string &soundKey);
Arrête le son joué
```

D’autres méthodes et propriétés peuvent être ajoutées en private uniquement si celles-ci sont jugées nécessaires. De plus, ajoutez à la fin de votre .cpp :

```
extern "C"
{
      std::shared_ptr<IDisplayModule> allocator()
      {
            return std::make_shared<Ncurses>();
      }
}
```

Une fois créés, compilez vos fichiers via g++ avec les flags de votre librairie ainsi que les flags suivants :

```
g++ -shared –o {NAME.so} -I{PATH_TO_IDISPLAYMODULE} -I./ -shared -fPIC –g
```

Puis, déposer l'archive dans le dossier lib/ du répertoire de l'arcade.

Il ne vous reste plus qu'à lancer l'arcade avec la nouvelle librairie :

```
./arcade lib/{NAME.so}
```

II – Faire une librairie de jeu

Pour créer une librairie de jeu compatible avec l'Arcade, créez une classe héritant de l'interface ***IGameModule*** dans un .hpp et surchargez dans un fichier .cpp les méthodes suivantes :

```
displayModule::e_event game();
```

```
Méthode qui permet de lancer le jeu depuis le Core

    return displayModule::e_event si le joueur souhaite revenir à la
sélection des jeux ou des librairies
```

```
bool initGame(const std::shared_ptr<displayModule::IDisplayModule>
&asset);
```

```
Méthode qui permet d'initialiser le jeu au lancement de celui-ci

    asset : la librairie à utiliser

    return true si le jeu est bien initialisé, false sinon
```

```
bool setLib(const std::shared_ptr<displayModule::IDisplayModule> &asset);
```

```
Méthode qui permet de stocker ou de changer de librairie

    asset : la librairie à stocker

    return true si la librairie est bien stockée, false sinon
```

D'autres méthodes et propriétés peuvent être ajoutées en private uniquement si celles-ci sont jugées nécessaires. De plus, ajoutez à la fin de votre .cpp :

Une fois créés, compilez vos fichiers via g++ avec les flags de votre librairie ainsi que les flags suivants :

```
g++ -shared –o {NAME.so} -I{PATH_TO_GRAPHICAL_LIBS} -I{PATH_TO_IGAMEMODULE} -
I./ -shared -fPIC
```

Puis, déposer l’archive dans le dossier games/ du répertoire de l’arcade.

Il ne vous reste plus qu’à lancer l’arcade avec une librairie graphique:

```
./arcade lib/{NAME.so}
```

Et à sélectionner votre jeu via les flèches directionnelles HAUT et BAS.