



INTRODUCTION À LA PROGRAMMATION ORIENTÉE OBJET EN C++

IS_3436

Sujets de Travaux pratiques - 12 h

3^{ème} année IS

Contents

1	TP numéro 1 : Pokemon selector	3
1.1	Mise en place du Framework SFML	3
1.2	Premières classes	4
1.3	Une interface graphique	5



Je vous encourage à utiliser vos PC personnels pour ces TPs. Je ferai personnellement la correction avec CLion, mais vous êtes autorisé à utiliser Visual Studio. Si vous souhaitez utiliser un autre IDE, c'est possible à vos risques et péril.
Les instructions d'installation ont été testé sous Linux...

1 TP numéro 1 : Pokemon selector

1.1 Mise en place du Framework SFML

Nous allons commencer par faire un setup de notre programme.

Tout d'abord, il faut être certain d'avoir installé SFML. Le plus simple :

```
$ sudo apt-get install libsFML-dev
```

Créez un nouveau projet C++ sous CLion (niveau de langage 17 mais nous n'utiliseront pas de fonctionnalité avancée).

Dans le fichier CMake, préciser l'usage de SFML de la manière suivante :

```
cmake_minimum_required(VERSION 3.21)
project(tictactoe)

set(CMAKE_CXX_STANDARD 17)

add_executable(tictactoe main.cpp)

target_link_libraries(tictactoe
    sfml-graphics
    sfml-window
    sfml-system
    sfml-audio
)
```

Enfin, nous allons tester notre setup à l'aide d'un programme simple (main.cpp) :

```
#include <SFML/Graphics.hpp>

int main() {
    sf::RenderWindow window(sf::VideoMode(800, 600), "Hello SFML");
    sf::CircleShape shape(100.f);
    shape.setFillColor(sf::Color::Green);

    while (window.isOpen()) {
        sf::Event event{};
        while (window.pollEvent(event)) {
            if (event.type == sf::Event::Closed)
                window.close();
        }
        window.clear();
        window.draw(shape);
        window.display();
    }
    return 0;
}
```

Téléchargez sur moodle les fichiers de données du projet : Pokedex et ensemble des images associées.



Réalisez le setup et vérifiez la bonne installation de l'ensemble des dépendances.

1.2 Premières classes



Ecrivez ou reprenez du cours une classe `Pokemon` (fichier `.cpp` et `.hpp`) permettant de définir un `Pokemon`. Il faut : un constructeur, un destructeur, une méthode d'attaque. La classe comprend entre autre :

- Le numéro du `pokemon`
- Son nom
- Son évolution (à zéro par défaut)
- Son nombre de point de vie maximum
- Son nombre de point de vie actuel
- Son nombre de point d'attaque
- Son nombre de point de défense

Il va s'agir d'être créatif : il vous faut maintenant définir une règle de jeu pour la méthode d'attaque. A vous de décider.



Ecrivez ou reprenez du cours une classe abstraite `Pokemon_Vector` (fichier `.cpp` et `.hpp`) permettant de définir une liste de `Pokemon`.



Ecrivez une classe `Pokedex` qui hérite de la classe précédente. Cette classe représente l'ensemble des `Pokemon` possible. C'est une référence dans laquelle on piochera des `Pokemon`. Elle est donc sous la forme d'un Singleton. Son constructeur vient lire le fichier `csv` et crée l'ensemble des `Pokemon`.

Une méthode permet d'extraire **le clone** d'un `Pokemon` interne. L'ensemble des données du `Pokedex` sont inaccessible.



Ecrivez une classe `Pokemon_Party` qui hérite de la classe `Pokemon_Vector`. Cette classe représente l'ensemble de vos `Pokemons`. Il n'y a pas de limite de taille.

On peut extraire ou ajouter un `Pokemon` du vecteur.



Ecrivez une classe `Pokemon_Attack` qui hérite de la classe `Pokemon_Vector`. Cette classe est un extrait de votre `Pokemon_Party` limité à 6 `Pokemon`.

Des méthodes permettent de créer la `Pokemon_Attack` à partir de la `Pokemon_Party`, d'autres permettent de réintégrer les `Pokemon`.

1.3 Une interface graphique



A l'aide de SFML, fabriquez une interface graphique qui permette de sélectionner les 6 Pokemons de la `Pokemon_Attack`, et de les positionner là ou le souhaite le joueur dans la `Pokemon_Party`.