C++

Compte rendu Projet

Zoran **Krivokuca**
Mohamed **Ghaibouche**

Introduction

L'objectif principal de ce projet était de concevoir une application graphique en C++ axée sur le thème des Jeux Olympiques. Ce projet offrait une grande liberté, notamment en termes de choix du sujet et du choix de la bibliothèque graphique. Cependant, il était encadré par plusieurs exigences techniques précises :

- La création d'au moins 8 classes organisées sur au moins 3 niveaux de hiérarchie.
- L'implémentation de 2 fonctions virtuelles.
- L'usage de 2 surcharges d'opérateurs.
- L'utilisation de 2 conteneurs STL.

Pour mener à bien ce projet, nous avons opté pour la bibliothèque graphique SFML, reconnue pour son contrôle avancé des interfaces utilisateurs.

Nous avons choisi de développer un générateur aléatoire de pays participants aux Jeux Olympiques. Cette fonctionnalité, activée par un simple clic sur un bouton, permet d'afficher un pays tiré au sort, accompagné de données telles que son nombre de participations et son palmarès de médailles d'or, d'argent et de bronze.

Les Contraintes

Parmi les contraintes, certaines ne concernent pas vraiment le code en soi et ne seront pas traitées ici. Nous allons cependant traiter toutes les autres contraintes avec des explications et leur respect dans notre projet.

8 classes et 3 niveaux de hiérarchies

Les contraintes de classe et de niveau de hiérarchie ont été facilement remplies par l'utilisation de SFML qui permet la création de classes pour chaque objet que l'on souhaite afficher. Ci-dessous, le diagramme UML montrant la structure du code et le respect de cette contrainte.

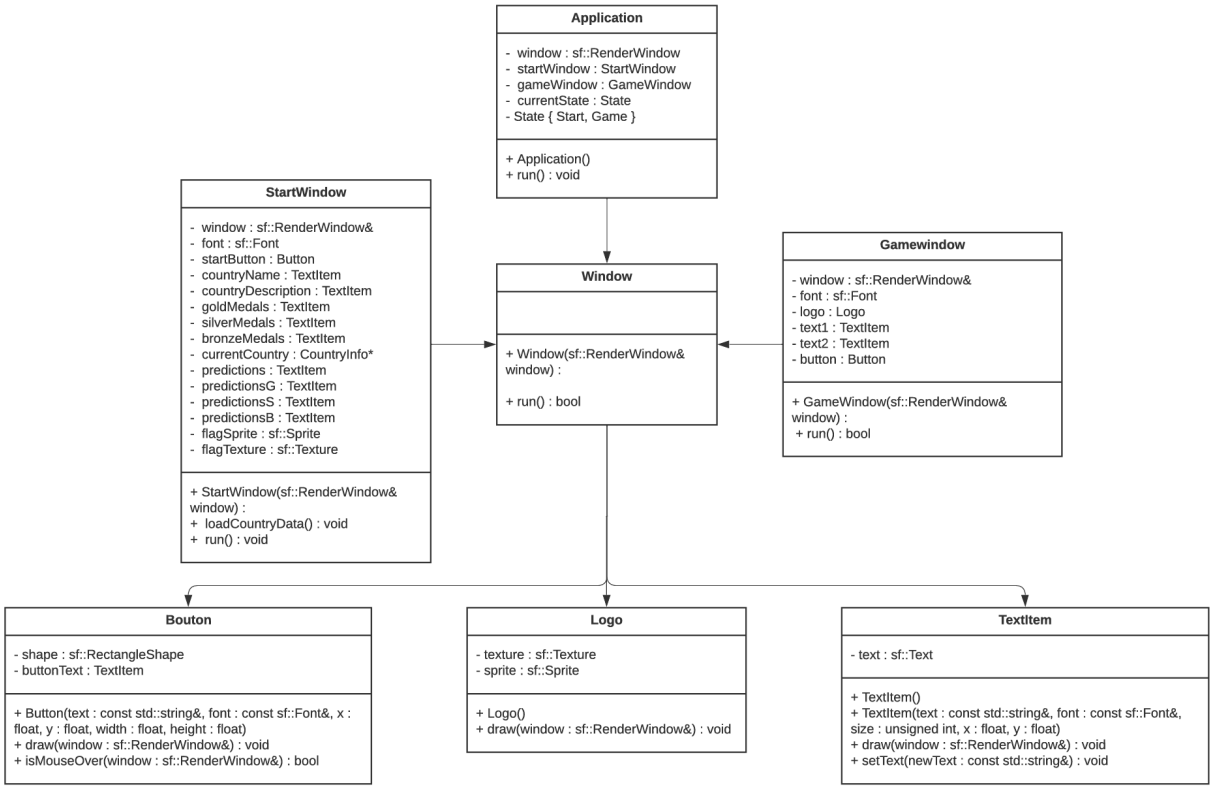


Diagramme UML théorique du projet

En pratique, notre diagramme UML diffère légèrement de celui ci, notamment par l'absence de l'interface Window qui n'était pas nécessaire lors de l'implémentation.

2 fonctions virtuelles

Les fonctions virtuelles que nous avons implémentées sont les fonctions Window() et run() de la classe Window. En effet, celles-ci sont remplacées par la suite par leurs équivalents respectifs des sous classes startWindow et gameWindow.

2 conteneurs STL

std::vector<CountryInfo> countries_list et std::map<std::string, sf::Texture> flagTextures sont les deux conteneurs STL que nous avons utilisés dans notre code. Le premier stock les informations des différents pays que nous extrayons d'un fichier texte que nous avons préparé et formaté au préalable. Le second quant à lui sert à stocker les textures des différents drapeaux pour éviter de les charger à chaque fois.

2 surcharges d'opérateurs

Les surcharges d'opérateurs sont utilisées surtout pour print des informations sur le terminal afin de s'assurer que tout s'est bien passé. Les deux surcharges sont dans la classe CountryInfo et servent respectivements à print les informations du pays (opérateur <<) et à vérifier si les codes de deux CountryInfo sont identiques (opérateur ==) afin de s'assurer qu'on affiche pas le même pays plusieurs fois d'affilée.

Autres contraintes

Les autres contraintes, telles que l'absence d'erreurs ou de warning lors de la compilation et les commentaires du code ont aussi tous été respectés. Aussi, hors lignes vides et commentaires, aucune méthode ne fait plus de 30 lignes, comme demandé.

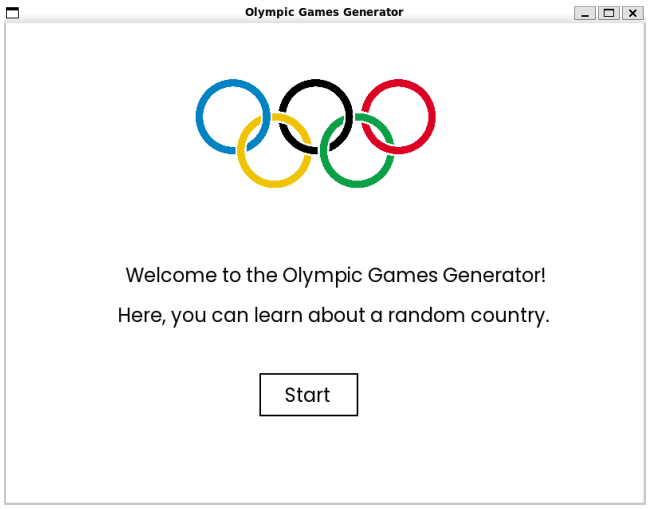
Processus d'installation et comment exécuter le code

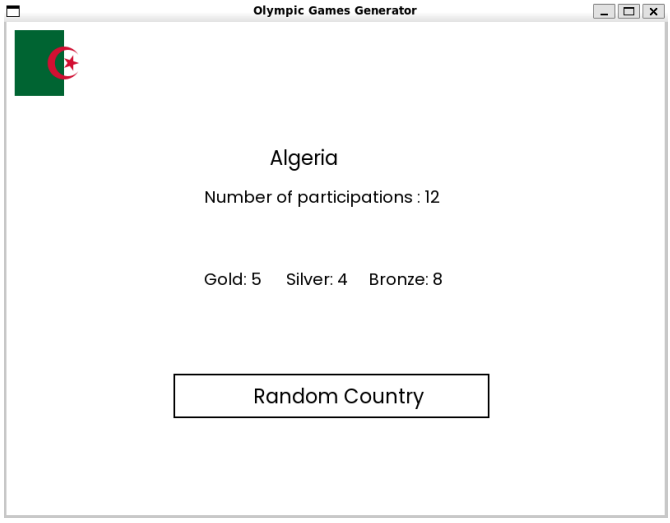
Processus d'installation

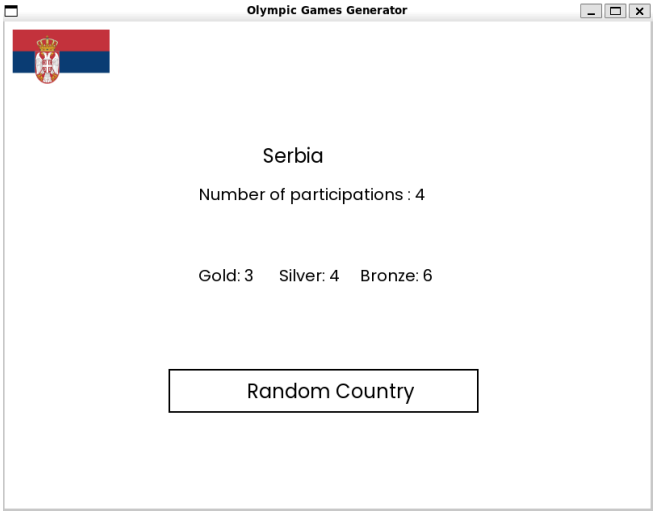
Nous avons décidé pour ce projet d'utiliser la bibliothèque SFML car cela nous semblait être la bibliothèque graphique la plus adaptée pour notre projet. De ce fait, pour exécuter notre projet, il suffit simplement d'installer la librairie SFML.

Exécution du code

Pour exécuter notre code il suffit simplement de se placer dans le directoire du projet et de taper dans terminal “make” puis de lancer lancer l’application avec la commande suivante “./OlympicGameGenerator”







Interface de l’application

Notre plus grande fierté

Notre plus grande fierté dans ce code est l'utilisation du conteneur map afin de stocker les textures des drapeaux. En effet, ça a permis de rendre notre code plus rapide lorsqu'on retombe sur le même pays, on évite de charger plusieurs fois le même drapeau pour économiser des ressources.