

COMPTE RENDU projet CPP

Bouchichit Walid

Bonjour, je vais rapidement vous présenter mon projet en c++.
Ce dernier a été réalisé sur Visual Studio Windows. Une fois installé et la librairie SFML mise en place, il n'y a pas besoin de makefile ou autre pour obtenir les erreurs. Le debugger et le linker sont intégrés à Visual Studio.

Pourquoi ce choix : Après avoir passé plus de 6h à installer la librairie SMFL sur visual studio code Windows avec plusieurs camarades, nous avons tout simplement abandonné et opter pour Visual studio (installation complète en 10 minutes)

Étape d'installation de visual Studio + SFML :

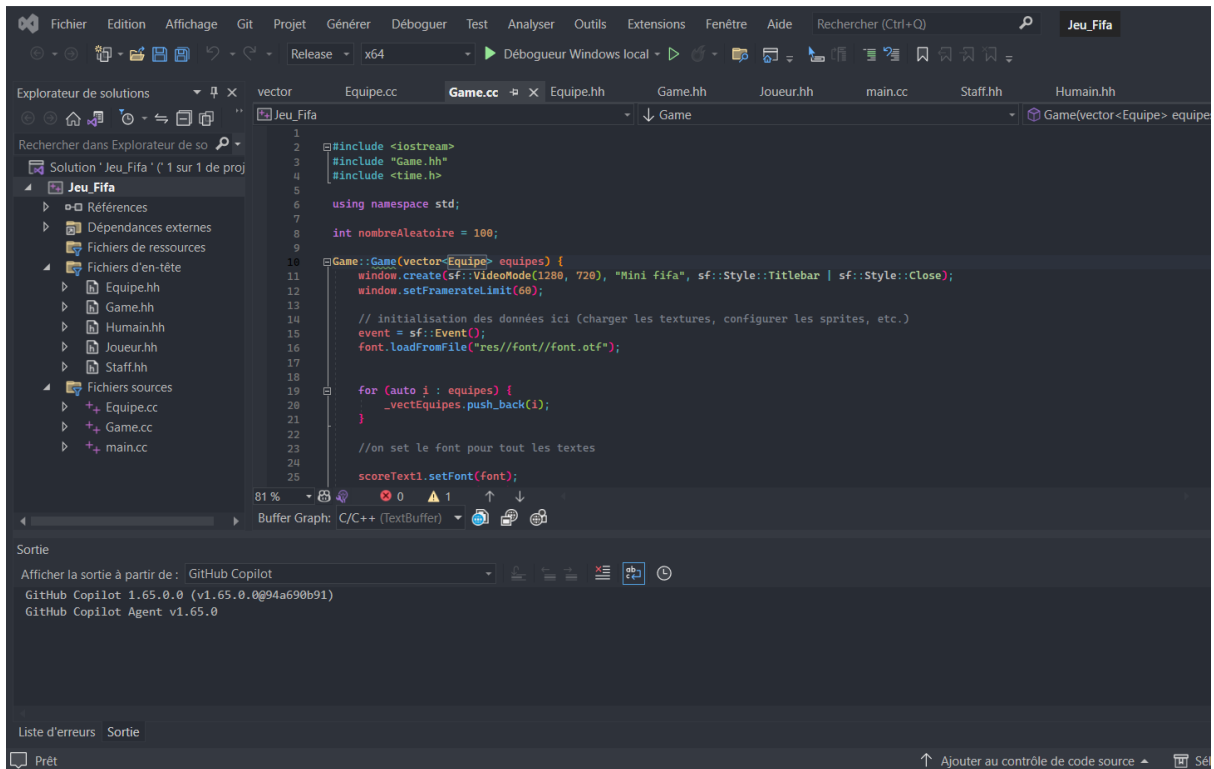
Télécharger VISUAL STUDIO ici : <https://visualstudio.microsoft.com/fr/downloads/>

Télécharger la librairie SFML : <https://www.sfml-dev.org/download/sfml/2.5.1/index-fr.php>

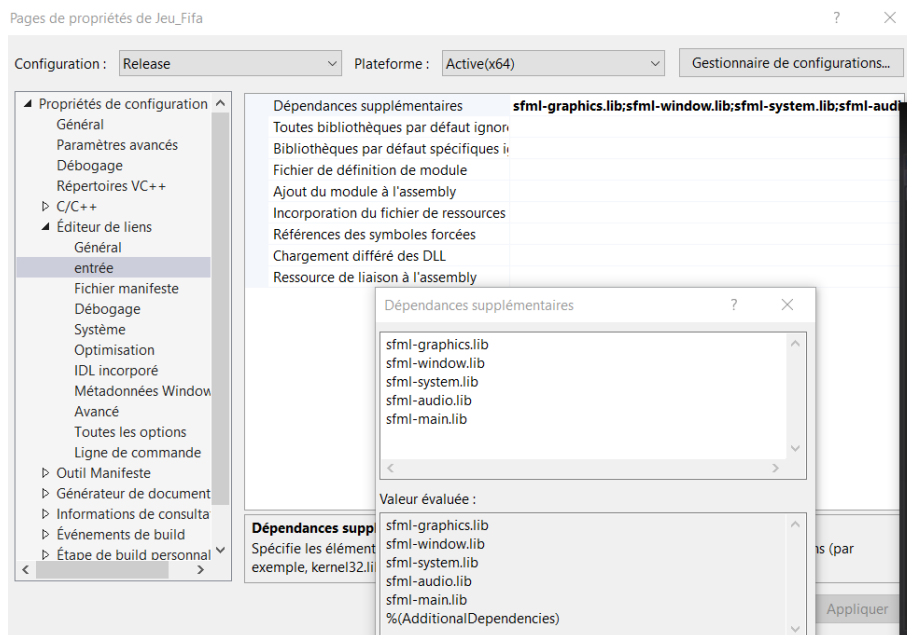
Créez un projet en c++ vide sur Visual Studio.
Il faut dire à visual studio où se trouve les librairies SMFL comme dans les screens ci dessous.

COPIEZ COLLEZ les fichiers includes et lib de SFML dans le dossier source de votre projet

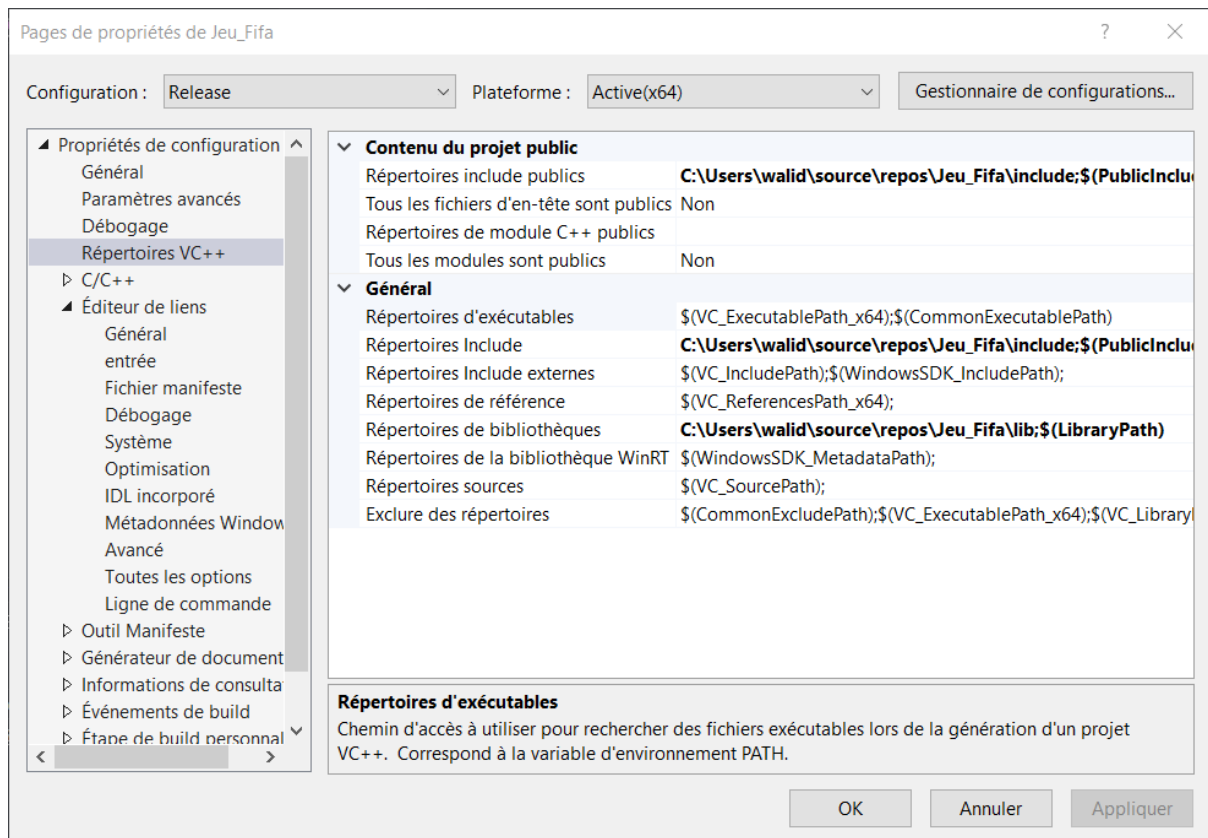
Nom	Modifié le	Type	Taille
.vs	09/01/2023 02:06	Dossier de fichiers	
include	09/01/2023 02:07	Dossier de fichiers	
lib	09/01/2023 02:07	Dossier de fichiers	
res	27/01/2023 15:21	Dossier de fichiers	
x64	09/01/2023 02:23	Dossier de fichiers	
Equipe.cc	27/01/2023 05:16	C++ Source	11 Ko
Equipe.hh	27/01/2023 22:26	C/C++ Header	3 Ko
Game.cc	27/01/2023 23:03	C++ Source	46 Ko
Game.hh	27/01/2023 21:09	C/C++ Header	5 Ko
Humain.hh	27/01/2023 04:12	C/C++ Header	1 Ko
Jeu_Fifa.sln	09/01/2023 02:06	Visual Studio Solut...	2 Ko
Jeu_Fifa.vcxproj	09/01/2023 02:28	VC++ Project	8 Ko
Jeu_Fifa.vcxproj.filters	09/01/2023 02:20	VC++ Project Filte...	2 Ko
Jeu_Fifa.vcxproj.user	09/01/2023 02:06	Per-User Project O...	1 Ko
Joueur.hh	27/01/2023 18:31	C/C++ Header	7 Ko
main.cc	27/01/2023 22:44	C++ Source	3 Ko
Staff.hh	27/01/2023 22:00	C/C++ Header	4 Ko



Cliquez sur le nom de votre projet avec click droit, descendez et cherchez propriété Configuration : mettez en mode debug et en x64
 Ensuite allez sur editeurs de liens puis sur entrée :
 collez le texte suivant sur dependances supplémentaire :
 sfml-graphics-d.lib;sfml-window-d.lib;sfml-system-d.lib;sfml-audio-d.lib;sfml-main-d.lib;%(AdditionalDependencies)



Allez dans Répertoires VC++, c'est ici qu'on va lui dire quel libraryPath utiliser



Dans Répertoire publics, sélectionnez le dossier include de SFML qui est dans votre dossier source.

Ajoutez le aussi dans General : repertoire includes

Dans Répertoire des bibliothèques, ajoutez le fichier Lib

Ajoutez les dll à votre fichier, copiez les dll depuis le fichier SFML vers votre fichier source.

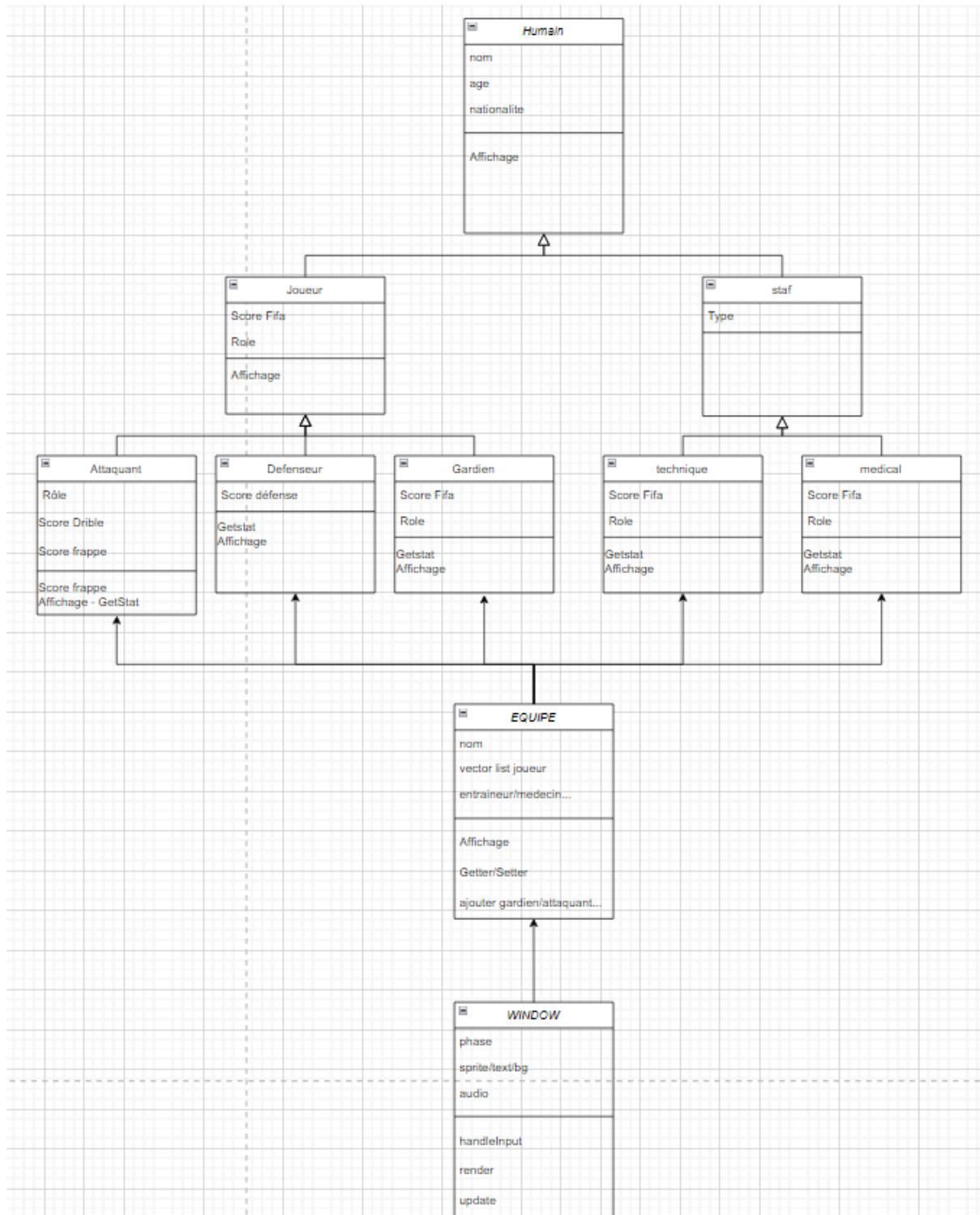
Voici une video tuto au cas où vous n'avez tout bien compris :

<https://www.youtube.com/watch?v=fBbTtnPjy8s>

Ajoutez les fichiers .cc et .cc ainsi que res dans le dossier source de votre projet et vous pourrez le compiler !

Vous êtes prêts à programmer avec votre librairie SFML :

DIAGRAMME



Explication du Diagramme et des choix faits :

Humain:

- J'ai créé des humains pour avoir des données de base qui seront les mêmes pour tous les personnages
- la fonction afficher est en virtual, elle est redéfinis pour les classes suivantes

les joueurs et le staff.

- Pour les Joueurs, nous avons quelques stats à ajouter tel que le score Fifa !
- Pour le staff, ce n'est qu'une classe abstraite qui nous permettra de tester si telle ou telle personne fait bien partie du staff.

Enfin nous avons nos Attaquants, Défenseur, Gardien et notre staff Technique (le coach) et le staff médical (notre médecin)

Utilité du staff :

- je voulais que le staff technique puisse être utilisé 1 fois dans le match pour rendre sa défense impénétrable !
- Le staff médical lui pourrait booster l'attaquant pour marquer un but à coup sûr !

Hélas je n'ai pas eu assez de temps pour tout gérer tout seul mais je suis très fière de ce qui a été accompli.

Classe ÉQUIPE :

- Elle possède un nom ("l'Équipe des Bleus" par exemple),
- un vecteur de joueurs (avec différents rôle),
- un vecteur pour chaque catégorie de joueurs.
- Le staff médical et le coach !

Il y a beaucoup de setter/getter que je n'ai pas utilisé mais que j'ai mis en place au cas où je voulais modifier/améliorer le jeu.

De plus, j'ai supprimé tous les destructeurs car ils empêchaient à chaque fois l'affichage de ma fenêtre graphique (je n'ai pas trouvé la raison mais si vous l'avez, je suis preneur).

Fenêtre Graphique:

- Elle reçoit un vecteur dans lequel j'ai placé nos 4 équipes initialisées.
- Dans le fichier res (ressources) se situe la musique utilisée en fond ainsi que toutes les images affichées dans la fenêtre. Tout cela est classé et ordonné en sous-dossiers.

Fonctionnement en 3 fonctions principales :

- Handle Input() : Cette fonction permet de gérer toutes les interactions entre l'humain et ses périphériques (clavier, souris) et d'agir en conséquence !
- render() : cette dernière va gérer l'affichage de toutes les images, tous les textes, les background et autres en fonction de divers variables !
- update() : On va mettre à jour les images affichées, les sons, certaines variables de position, certaines variables d'état...

A l'aide de ses 3 fonctions, nous pouvons faire à peu près ce que l'on veut.

Fonctionnement du jeu:

- Je ne savais pas comment me lancer dans le développement de ce jeu, surtout avec l'affichage graphique SFML qui est une première ! J'ai donc décidé de faire simple, couper mon jeu en différentes phases et avancer petit à petit
- je ne rentrerais pas dans le détail des phases mais voici un rapide résumé :
- On a d'abord l'accueil puis la mise en place du contexte de jeu (les équipes à choisir, les joueurs à choisir...)
- Ensuite nous avons le déroulement du jeu qui est dans notre cas assez simple

Le JEU :

L'équipe 1 a le ballon en premier : il est possible de faire 2 actions:

- 1) Tirer directement, le pourcentage de chance que le tir pass le défenseur puis le gardien est diminué
- 2) dribbler le défenseur et finir derrière lui afin de tirer sur le gardien, les chances de but sont plus élevée

Si le défenseur ou le gardien intercepte la balle, nous passons au tour suivant et l'attaquant adverse obtient le ballon !

Le jeu se déroule en 60 TOURS.

FIERTÉ :

Je n'ai pas vraiment de partie de code dont je suis fière mais c'est plutôt l'ensemble du code que j'ai beaucoup apprécié faire et le rendu final.

Si je devais parler de 2 parties de codes ce serait les suivantes :

- 1) En effet j'ai mis une petite musique de fond, ce qui n'était vraiment pas facile car le fonctionnement de l'audio n'était pas implémenté, il fallait rajouter certaines directives dans les propriétés du projet. Beaucoup ont abandonné l'idée de mettre un son en arrière plan mais je n'ai pas lâché et j'ai essayé de l'implémenter à tout prix. Je suis aussi très content du fait que l'on peut baisser ou augmenter le son et que l'affichage graphique vous montre bien ce changement !

En bas à gauche, on voit bien cette partie audio dont je parle



- 2) En seconde partie je parlais d'une astuce que j'ai utilisé et qui m'a aidé à sortir d'une interaction assez compliqué à gérer, voici le problème de base : Lorsque vous appuyez sur votre souris, l'information est instantanément traitée par le code, le problème qui en découle et qu'en appuyant sur un bouton play, si il y a un bouton au même endroit dans la phase suivante, alors vous appuierez automatiquement dessus aussi, de plus en appuyant, tout est fait instantanément, vous ne voyez donc pas certaines modifications qui sont changées trop rapidement! J'ai utilisé 4 astuces:

la première est de temporiser grâce à du code et la bibliothèque time.h (beaucoup d'inconvénients puisque le code n'avance plus, mais au moins il ne détecte plus que vous appuyez sur la souris)

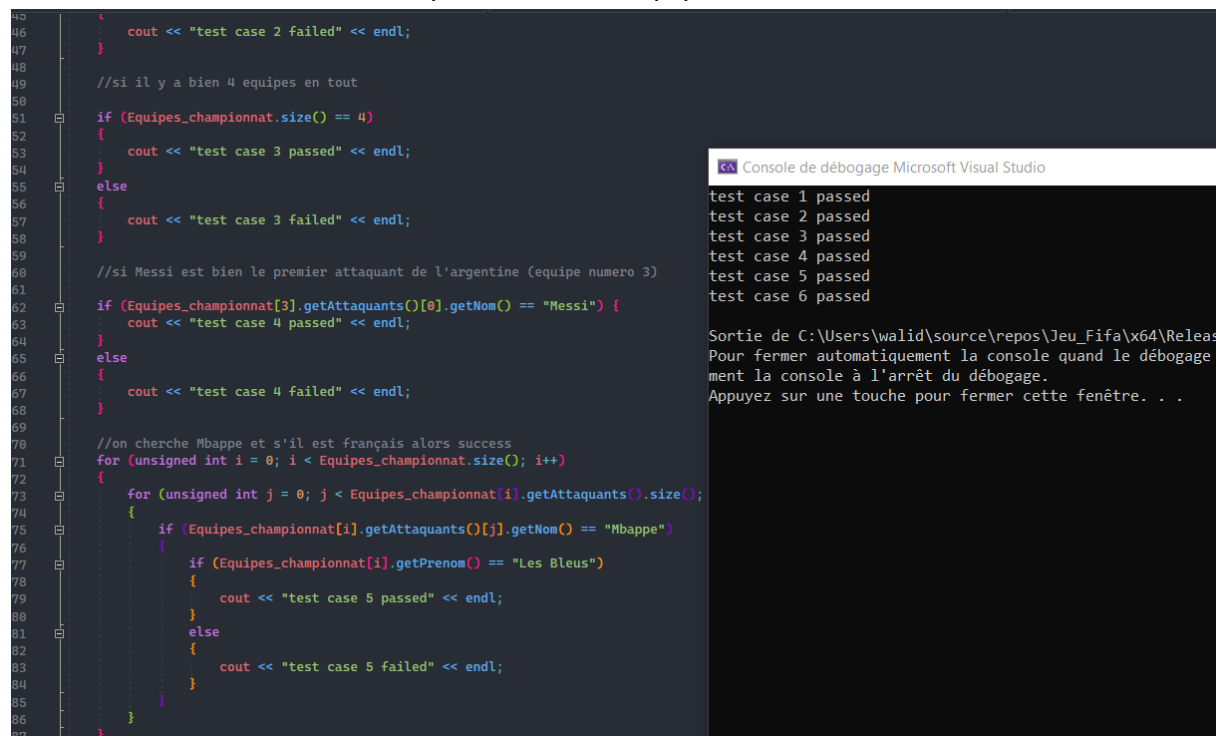
la seconde est de déplacer des boutons afin qu'aucun bouton d'une phase i soit au même endroit qu'un bouton à une phase i+1

la troisième est de rajouter des conditions sur le saut à une prochaine phase, cela entraîne que tout le code est lu et appliqué sans que le prochain bouton ne soit fonctionnel, une fois tout le code lu, je lève la condition sur une variable si vous appuyez de nouveau avec votre souris. Ainsi, vous voyez bien les joueurs se déplacer !

la 4ème astuce est de ne prendre en compte vos actions que lorsque vous relâchez le clic de la souris, ainsi, on a aucun problème d'appuie trop long !

TESTCASE :

Nous faisons différents testcase pour voir si les équipes ont bien été créé et initialisé:



```
45     cout << "test case 2 failed" << endl;
46 }
47
48 //si il y a bien 4 equipes en tout
49
50 if (Equipes_championnat.size() == 4)
51 {
52     cout << "test case 3 passed" << endl;
53 }
54 else
55 {
56     cout << "test case 3 failed" << endl;
57 }
58
59 //si Messi est bien le premier attaquant de l'argentine (equipe numero 3)
60
61 if (Equipes_championnat[3].getAttaquants()[0].getNom() == "Messi") {
62     cout << "test case 4 passed" << endl;
63 }
64 else
65 {
66     cout << "test case 4 failed" << endl;
67 }
68
69 //on cherche Mbappe et s'il est français alors success
70 for (unsigned int i = 0; i < Equipes_championnat.size(); i++)
71 {
72     for (unsigned int j = 0; j < Equipes_championnat[i].getAttaquants().size();
73         {
74             if (Equipes_championnat[i].getAttaquants()[j].getNom() == "Mbappe")
75             {
76                 if (Equipes_championnat[i].getPrenom() == "Les Bleus")
77                 {
78                     cout << "test case 5 passed" << endl;
79                 }
80                 else
81                 {
82                     cout << "test case 5 failed" << endl;
83                 }
84             }
85         }
86     }
87 }
```

Console de débogage Microsoft Visual Studio

```
test case 1 passed
test case 2 passed
test case 3 passed
test case 4 passed
test case 5 passed
test case 6 passed
```

Sortie de C:\Users\walid\source\repos\Jeu_Fifa\x64\Release
Pour fermer automatiquement la console quand le débogage
ment la console à l'arrêt du débogage.
Appuyez sur une touche pour fermer cette fenêtre. . .

Dernières info :

Quasiment tout le code est commenté !

Il est jouable et assez beau !

Petite musique de fond en loop, vous pouvez baisser le son si vous n'aimez pas !

Il y a plus de 8 classes et 3 niveaux de hiérarchies.

Le code n'est pas le plus complexe que l'on a fait mais les petites erreurs de pointeurs ou de redéfinition prennent 80% du temps à debugger !

J'ai beaucoup appris et beaucoup aimé ce projet (~45h), je compte un peu l'améliorer et le faire jouer à mes potes !

Merci beaucoup pour votre lecture et bon courage avec tous les projets
BOUCHICHIT WALID
POLYTECH SORBONNE