TP

Concevoir et développer une maquette de jeu by Baptiste CLOCHARD

Objectifs du TP

L'objectif de ce TP est de créer un prototype de jeu en 2D avec le moteur Godot. À la fin du TP, vous aurez réalisé :

- Un personnage animé capable de se déplacer.
- Une carte de fond réalisée avec une TileMap, avec des propriétés comme les collisions et le Y-Sort (le personnage passe devant/derrière le décors).
- L'installation et l'utilisation d'outils externes (*Character Generator* de Limezu et le plugin Creator Tool Importer de BananaHolograma).
- L'utilisation de GDScript pour programmer le déplacement.

1 Création et configuration du projet

- Téléchargez les fichiers de map et le character generator sur Moodle (les assets viennent de `https://limezu.itch.io/modernexteriors` si vous volez voir à quoi ils ressemblent une fois assemblé, les tilesets c'est assez abstrait).
- Téléchargez Godot : `https://godotengine.org/download/`
- Lancez Godot (le .exe que vous avez téléchargé, pas besoins de l'installer et ça va 50x plus vite qu'Unreal).
- Sélectionnez le mode compatibility (utile pour faire de l'export web)
- Téléchargez le plugin Creator Tool Importer sur l'*Asset Library* de Godot (directement depuis le moteur, barre d'en haut "assetLib", il y en a deux, prenez celui de BananaHolograma, si il y a des warnings, comme d'habitude en dev on les ignore).
- Activez-le dans *Project Settings* > *Plugins*.

2 Création et animation du personnage

- Créez votre personnage avec le character creator (.exe sur Moodle), à la fin vous devriez obtenir une image semblable à celle-ci (si ça ne marche pas, j'ai pas pu tester sur Linux, prenez le fichier de secours BeauGosse.png sur Moodle) :

- Créez une scène intitulée Player.tscn (clic droit dans l'explorateur à gauche) et cliquez sur 2D en haut pour en afficher l'apparence.
- Suivez la documentation de `https://github.com/ninetailsrabbit/character-generato` `tab=readme-ov-file#getting-started` pour ajouter votre personnage et vos animations au projet (dans la liste des animations à conserver, vous pouvez garder uniquement "idle" et "walk").

3 Ajouter le déplacement au personage

- Créez une nouvelle scène intitulée `Game.tscn`.
- Ajoutez votre joueur à l'arborescence.
- Mon personnage est tout flou ? Project settings -> Rendering -> Textures -> Default Texture Filter -> Nearest.

- Ajoutez le déplacement au personnage :
  - Ajoutez votre script à CharacterBody2D et une variable pour accéder à l'animatedSprite2D.
  - https://docs.godotengine.org/en/stable/tutorials/physics/using_character_body_2d.html pour le code de vélocité etc.
  - https://docs.godotengine.org/en/stable/classes/class_inputmap.html pour ajouter les inputmaps (à ajouter dans Project Settings -> Input Map, identique aux actions d'Unreal, oui je fais de la pub).

4 Création et animation du TileMap

Comme vu en cours, nous allons créer un tileset avant de créer la TileMap.

- Ajouter un noeud TileMapLayer à votre scène (n'hésitez pas à le mettre en fils d'un noeud 2D Tilemap pour tout organiser, attention le noeud de type TileMap est déprécié).
- la documentation sera très utile pour la suite (ajout de collisions notamment) : https://docs.godotengine.org/en/4.0/tutorials/2d/using_tilesets.html, prenez connaissance des outils offerts sur les tilesets (comment ajouter des collisions notemment).
- En bas de votre écran, appuyez sur l'onglet TileSet et créez un nouveau TileSet à partir de l'image "Modern_Exteriors_Complete_Tileset.png".
- Ajoutez ensuite ce set à votre layer (dans l'inspecteur à droite), créez un layer pour le sol et un layer pour les obstacles. Pour réutiliser le tileset, vous devez le sauvegarder avant : Sur le bouton TileSet il y a un menu descendant -> save as -> tileset.tres. Vous pourrez importer ce fichier dans vos autres layers.
- Ensuite ajoutez le Y sorting et les collisions grâce à ce tutoriel : https://www.youtube.com/watch?v=60rpO_0CJII

5 Bonus !

- Animez votre map : https://www.youtube.com/watch?v=AO-pqAvzowk (les fichiers d'animations sont dans "Animated sheets" et "Animated Terrains").
- Ajoutez des collisions à votre tileset (fin du tuto y sorting ou documentation sur l'utilisation des atlas).
- Sauvegardez la dernière position du joueur et chargez celle-ci au demarrage du jeu (voir https://docs.godotengine.org/en/stable/tutorials/io/saving_games.html)

TP

Concevoir et développer une maquette de jeu by Baptiste CLOCHARD

Objectifs du TP

Nous allons reprendre notre TP précédent et allons augmenter le jeu. Nous allons :

- Ajouter un menu principal trop beau (si si).
- C'est tout (comme ça vous pourrez terminer les 2 tps)

1 Changement de l'interface

Pour cette partie, aidez vous de la documentation : `https://docs.godotengine.org/en/stable/tutorials/rendering/multiple_resolutions.html` Pour l'instant notre fenêtre s'adapte très mal au grand écran. Pour cela nous allons faire en sorte que le jeu s'agrandisse quand nous augmentons la taille de la fenêtre. Nous allons donc modifier plusieurs choses dans Project Settings -> Display -> Window :

- La taille de la fenêtre par défaut en 1920x1080.
- Un élément permettant lorsque l'on agrandit la fenêtre du jeu d'agrandir le jeu. Comme ça on a toujours autant d'éléments visibles, même si ils sont plus petits (notre pixel art doit rester net).
- Un élément permettant que le contenu de notre jeu remplisse la fenêtre sans l'ajout de bandes noires. (Comme dans stardew valley, les gens avec un plus grand écran voient une partie plus importante de la map et contrairement à Valorant où on se demande pourquoi on a acheté un écran ultra large pour avoir des bandes noires comme à l'époque du 4/3).
- Observez l'impact de "stretch scale mode integer/fractionnal" recommandé par la documentation.

Maintenant notre jeu parait bien petit, ajouter un noeud Camera à notre Player et ajustez le zoom (la caméra doit suivre le joueur).

2 Ajout du menu principal

Créez une nouvelle scène pour notre interface graphique, contrairement aux éléments du jeu (player, tilemap) qui sont des Node2D, notre interface utilisera des éléments de type "Control". Choisissez donc "Control" comme noeud racine de la scène.

Nous allons maintenant créer un menu de base avec deux boutons : "Return to game" et "Quit"

Voilà à quoi devra ressembler votre interface 1 :

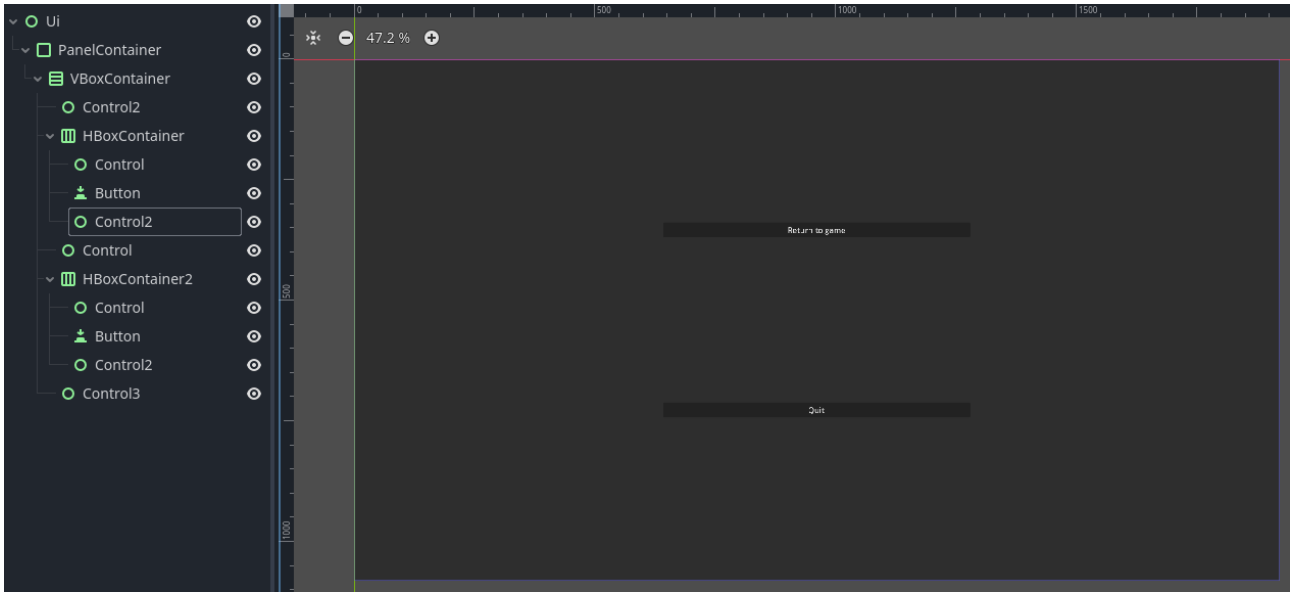

FIGURE 1 – Les controls composants votre menu

Pourquoi il y a autant de control vide ? Excellente question ! Dans Godot pour créer des espaces entre les éléments on ajoute des éléments vide "Control" et on leur donne une taille spéciale. Ici pour spécifier leur taille on utilise la propriété Layout -> Contener sizing -> Expand = true. (Il existe d'autres moyens d'ajouter de l'espace, par exemple le Control MarginContainer, mais le combo Control/Expand reste le mieux).

Vous pouvez régler le ratio espace/boutons avec la propriété StrechRatio, juste en dessous de Expand.

3 Ajout des signaux

Grâce aux InputMap vu dans le tp précédent faites en sortes que la fenêtre du menu s'affiche quand on appuie sur échap (elle se cache au lancement du jeu).

Je ne comprends pas ma fenêtre est zoomée à fond et se déplace avec ma caméra !. C'est normal, il faut rendre l'UI indépendante du reste du jeu, et pour ça vous pouvez utiliser le noeuds CanvasItem (description au survol : A node used for independent rendering of objects within a 2D scene.).

Pour pouvoir interragir avec la fenêtre, nous allons utiliser les signaux. Avec les noeuds et les Scènes, Godot est très fortement POO et il est difficile parfois de connaitre l'état des autres objets et l'envie d'utiliser des autoloads pour tout est forte. Godot propose donc les signaux pour pouvoir communiquer.

Ici nous allons utiliser les signaux proposés par le signal "pressed" que l'on peut éditer dans l'inspecteur à droite, tout en haut onglet "node".

Pour les deux boutons, créez une fonction permettant de quitter le jeu et de revenir au jeu.

Bonus : Mettez le jeu en pause lorsque vous êtes dans le menu principal.

4 Ajoutons du style à notre menu !

Nous allons ajouter un style pixel art à notre menu. Pour ça nous allons utiliser les thèmes.

- Créez un nouveau thème : dans l'explorateur clic droit -> New -> Ressource -> Theme
- Appliquez le thème à notre jeu : dans l'explorateur Project settings -> GUI -> Theme -> Custom
- Téléchargez la police `https://fonts.google.com/specimen/Pixelify+Sans` depuis Google (note : les polices googles sont gratuites et vous pouvez les utiliser dans les projets commerciaux. En plus, elles gèrent les majuscules, les accents etc ce que ne font que peu de police ailleurs sur le web (coucou dafont).)
- Ajoutez le fichier "variable_font" dans votre dossier de projet.
- Ajouter un style de bouton à votre thème (en haut à droite de la fenêtre d'édition de thème, puis ajouter un style "Button"). Ajouter la police comme police pour les bouttons et augmenter la taille du texte par défaut à 32.
- Ajoutons une bordure pixel art à nos boutons : Créez une "StyleBoxTexture" pour la texture de bouton "normal". Ajoutez la texture "empty1.png" sur Moodle et utilisez les "textures margins" dans l'inspecteur pour séparer le centre du bouton de ses marges.

Bonus ! : Ajoutez une animation lorsqu'on survol et qu'on clic sur le bouton avec empty2 et empty3.