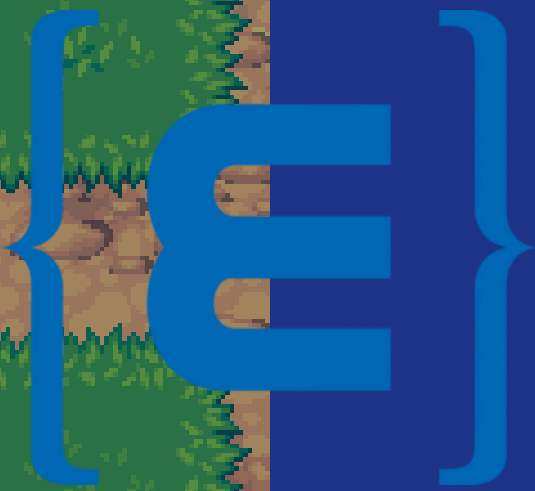




AVRIL 2022



My first RPG

THÈMES

Initiation à l'utilisation d'un moteur de jeu (Godot 3.4.4)
Découverte des spritesheets et du tilemapping

ORGANISÉ PAR

Alexandre VIGEANT (Étudiant en 2ème année PGE)
Robin LANDRAUD (Étudiant en 2ème année PGE)
Corentin BEYRIES (Étudiant en 1ère année PGE)

INTRODUCTION

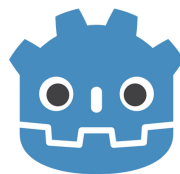
LES MOTEURS DE JEUX VIDÉOS

Il existe aujourd'hui plusieurs façons pour développer son propre jeu vidéo, la première méthode (et la plus difficile) est de développer son propre moteur graphique pour son jeu sans que celui ci soit réutilisable pour un autre jeu. (C'est le genre de chose que l'on peut apprendre à Epitech).

La deuxième méthode consiste à utiliser un moteur graphique déjà existant qui contient tous les outils nécessaires au développement d'un jeu et qui est maintenu par des développeurs qui sont potentiellement meilleurs que nous dans ce domaine. (Exemple : Unity ou Unreal Engine)

Dans ce court atelier nous allons découvrir comment développer son propre jeu vidéo à l'aide d'un moteur de jeu du nom de Godot.

Godot est un moteur de jeu multi-plateforme, c'est-à-dire un logiciel permettant de créer des jeux vidéos qui sont compatibles avec différents systèmes d'exploitation. Il comporte entre autre un moteur 2D, un moteur 3D, un moteur physique, un gestionnaire d'animations, et des langages de script pour programmer des comportements.



QUELQUES CONSIGNES AVANT DE COMMENCER

- En cas de question, pensez à demander de l'aide à votre voisin de droite. Puis de gauche. Si jamais vous êtes toujours bloqué, n'hésitez pas à nous demander à nous. (On est gentils)
- Vous avez tout à fait le droit d'utiliser internet pour vous renseigner et pour chercher des ressources.
- N'hésitez pas à être créatifs, même si nous vous donnons tout ce que vous avez besoin pour faire votre propre jeu, vous pouvez utiliser vos propres sprites si vous êtes à l'aise avec l'idée et personnaliser votre jeu comme vous le souhaitez.

INTRODUCTION

LES ÉTAPES DE CET ATELIER

01

DÉCOUVERTE

Découverte du moteur de jeu et des différents concepts

03

DÉVELOPPEMENT

Mise en place du joueur et de la carte de jeu

02

MISE EN PLACE

Mise en place des ressources

04

BONUS

Système de combat, monstres, dialogues, quêtes

RESSOURCES

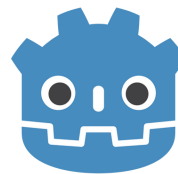
Pour cet atelier, nous vous fournissons diverses ressources afin de bien débiter le développement du jeu (et de ne pas perdre du temps à les chercher).

Tous les liens dont vous allez avoir besoin vous seront fournis au fur et à mesure de l'atelier.

Vous pouvez utiliser vos propres ressources graphiques si vous vous sentez à l'aise mais nous vous conseillons d'utiliser les ressources fournis dans cet atelier.

N'hésitez pas à venir nous voir en cas de problème avec un lien fourni.

PREMIERS PAS



INSTALLATION DE GODOT

Pour cet atelier nous allons utiliser la version 3.4.4 de Godot.

Le lien de téléchargement est celui-ci :

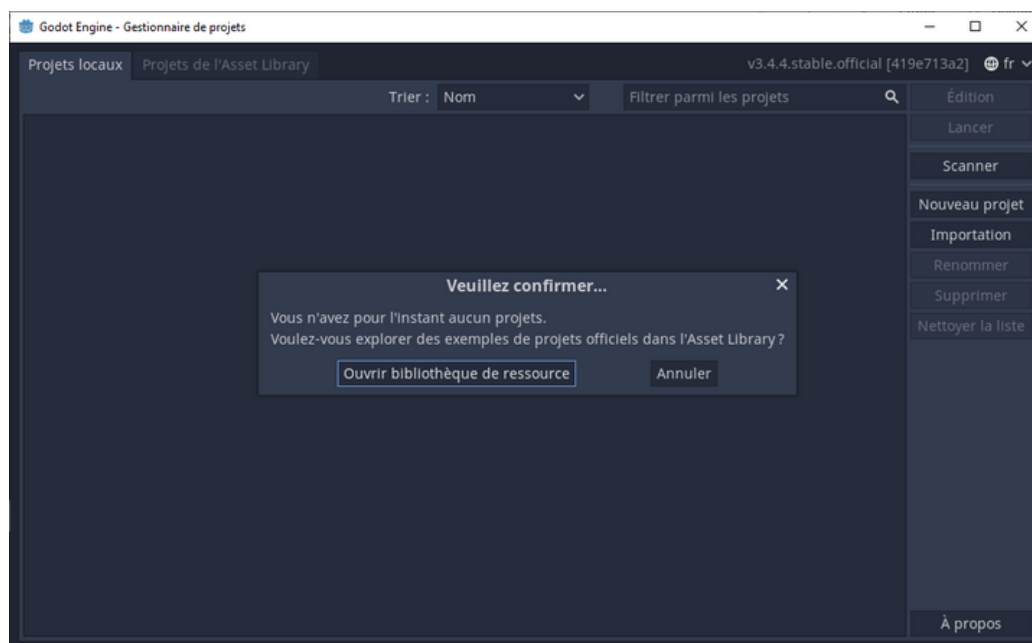
https://downloads.tuxfamily.org/godotengine/3.4.4/Godot_v3.4.4-stable_win64.exe.zip

Télécharger le fichier et décompresser le fichier exécutable (.exe) où vous voulez.

Godot est un exécutable indépendant et ne nécessite pas d'installation particulière.

LANCEMENT DE GODOT

Lors du lancement de Godot, vous verrez la fenêtre de **Gestionnaire de projets**, c'est ici que vous pouvez créer et charger vos projets de jeu.



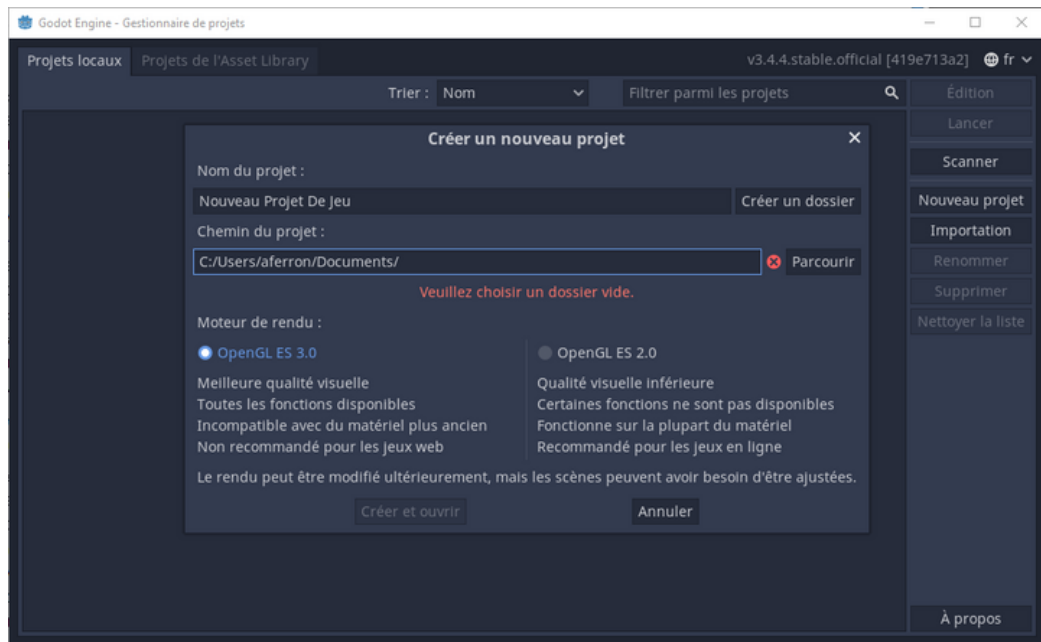
Lors du premier lancement de Godot, une fenêtre vous demandera si vous voulez explorer "l'**Asset Library**", cliquez sur Annuler pour l'instant.

Si la langue n'est pas bonne, vous pouvez la changer en haut à droite.

PREMIERS PAS


CRÉATION DE VOTRE PROJET

Pour créer votre nouveau projet, cliquez sur le bouton "Nouveau Projet" :



La procédure pour créer un nouveau projet peut être peu intuitive parce que Godot ne crée pas automatiquement les dossiers pour celui-ci.

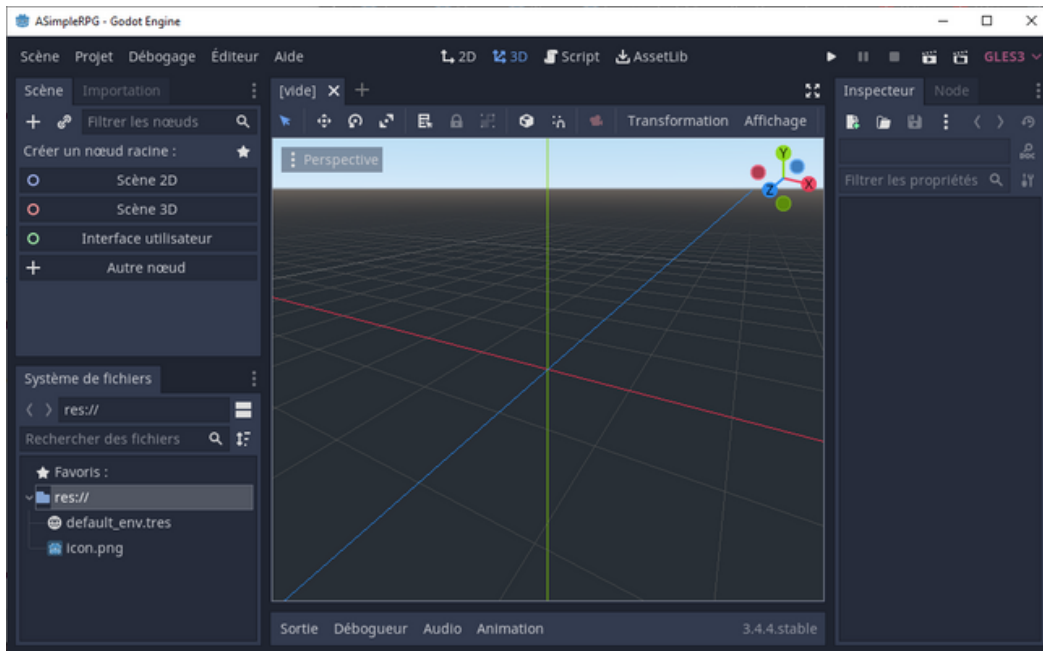
La façon la plus simple est la suivante :

- Dans "Chemin du projet", choisissez le dossier qui contiendra tous vos projets Godot (Le plus simple est de laisser celui par défaut)
- Ecrivez le nom de votre projet (soyez créatif) et cliquez sur le bouton "Créer un dossier". Godot créera automatiquement le dossier pour votre projet et modifiera le chemin.
- Si tout se passe bien, vous devriez voir un petit icône vert  à côté du chemin du projet.
- Cliquez sur le bouton créer et ouvrir et l'éditeur devrait s'ouvrir.

PREMIERS PAS

L'EDITEUR

Une fois votre projet créé, vous devrez voir une interface avec une vue en 3D :



Faisons un rapide tour de l'interface utilisateur de Godot. Nous nous pencherons plus en détails dans les prochaines étapes de l'atelier sur chaque partie du moteur.

Au centre de l'éditeur, occupant la plus grande partie de l'écran, se trouve l'espace de travail (**workspace**), c'est ici que le plus gros du travail sera fait.

En haut de la fenêtre de Godot :



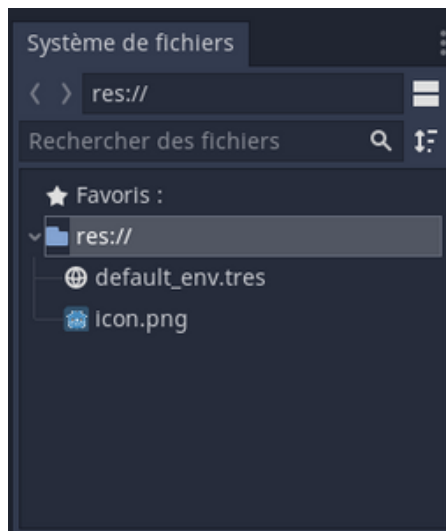
- A gauche se trouve le **menu principal**
- Au centre les boutons pour **changer de workspace**
- A droite les boutons pour **lancer et debugger le jeu**.

PREMIERS PAS

L'ÉDITEUR

Sur la gauche, vous trouverez le dock de Scène, c'est ici que sera listé le contenu de votre scène actuelle et que vous pourrez rajouter des nodes (Nous parlerons des nodes juste après)

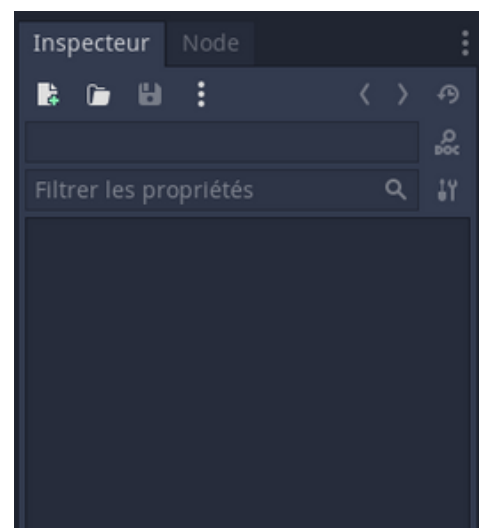
Pour l'autre onglet "Importation", c'est là que seront modifiés les propriétés des différentes ressources graphique qui seront importés dans votre projet.



Juste en dessous se trouve le dock du système de fichiers, c'est ici que seront affichés les différents assets et scripts qui seront utilisés pour votre jeu.

Du côté droit, vous trouverez l'Inspecteur où vous pourrez voir les différentes propriétés des nodes et le panneau des Nodes où seront affichés les différents signaux/événements utilisés pour les nodes afin de pouvoir développer des scripts.

Retenez bien ce nom, car nous l'utiliserons énormément dans l'atelier !



PREMIERS PAS

PARAMÉTRAGES DU PROJET

Godot nous offre de nombreuses options dédiées à ceux qui font des jeux en pixel art.

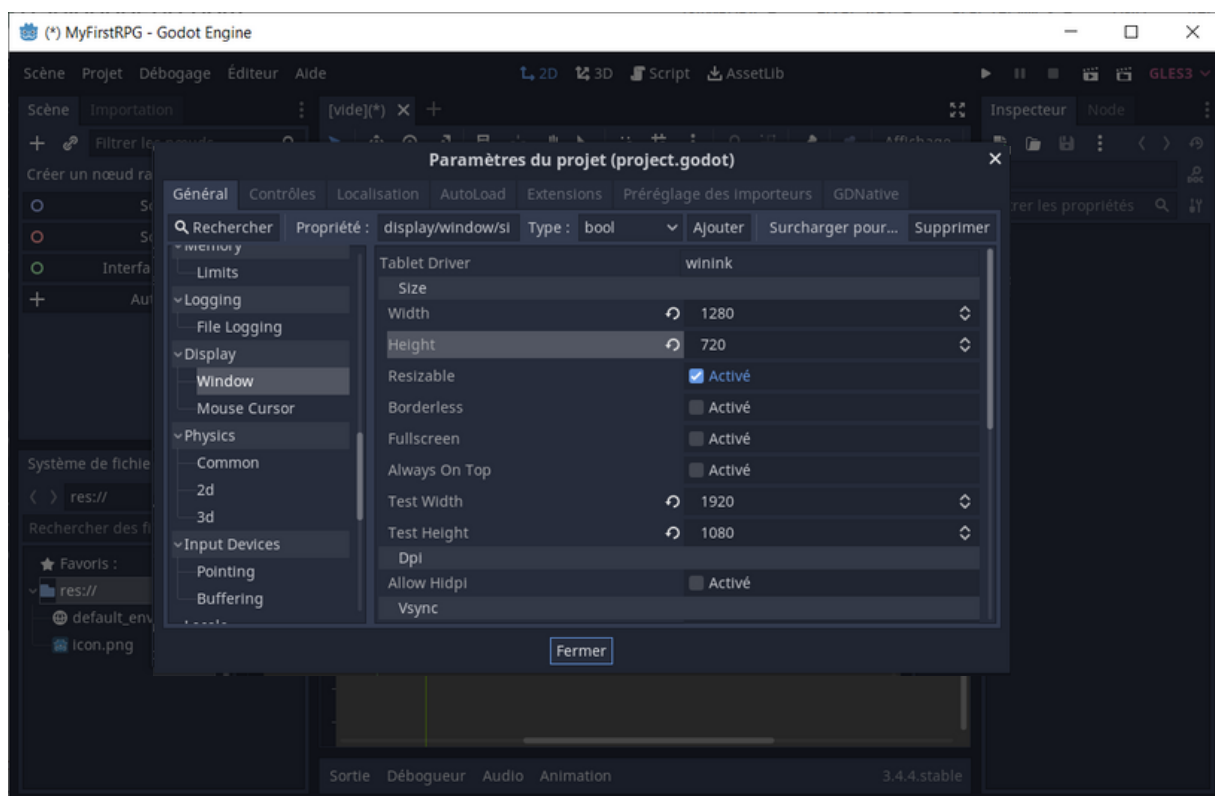
Dans ce type de jeux, nous essayons de recréer l'apparence des jeux de l'époque 8 bits et 16 bits, lorsque la résolution disponible était bien inférieure à celle des moniteurs actuels.

Nous devons donc configurer Godot pour utiliser une résolution inférieure et rendre les images avec une précision au pixel près. Par défaut, Godot rendra à une résolution inférieure au pixel, mais dans ce cas, les images pixel art pourraient être floues.

Pour commencer nous allons modifier quelques paramètres du projet, pour cela, vous pouvez accéder au paramètre du projet dans **Projet → Paramètres du projet...**

Allez dans **Display → Window** et modifiez les paramètres comme ci dessous (En l'occurrence Width, Height, Test Width, Test Height)

Si vous avez besoin d'aide, n'hésitez pas à venir nous demander.

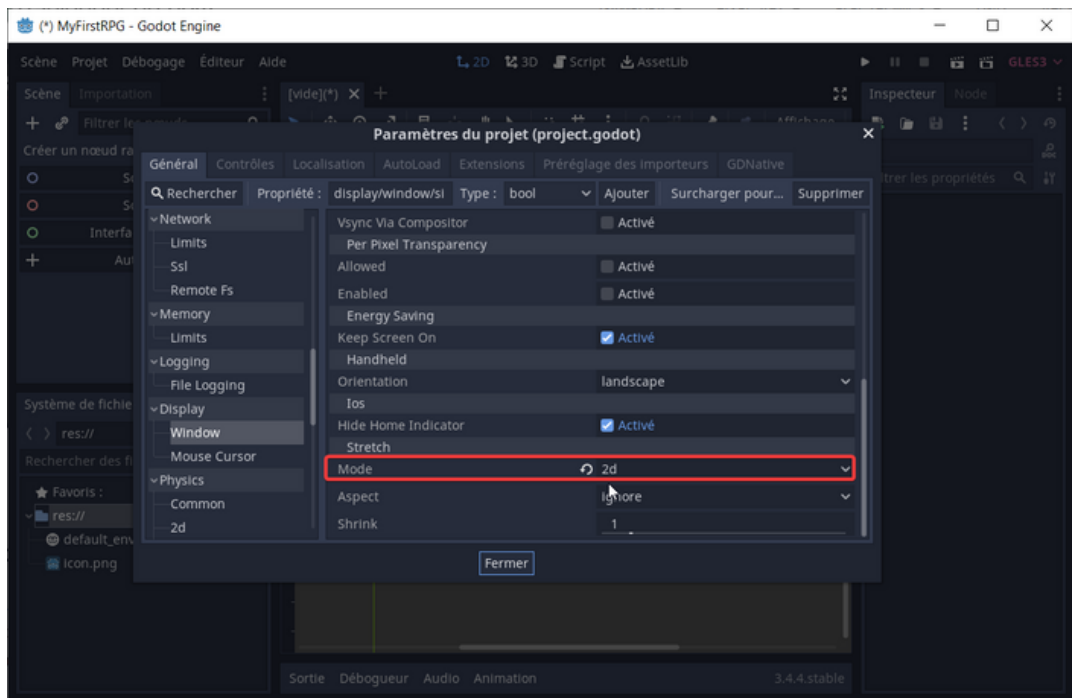


PREMIERS PAS

PARAMÉTRAGES DU PROJET

Toujours au même endroit, modifiez le **Stretch Mode** en **2D**.

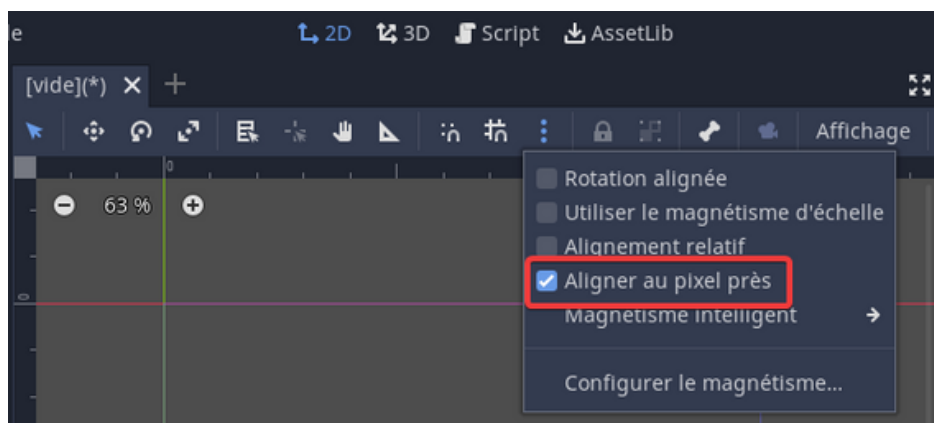
Cela permet de dire à Godot d'utiliser toute la fenêtre pour afficher le jeu tout en préservant la pixellisation d'origine.



Vous pouvez maintenant fermer cette fenêtre et passer votre espace de travail en **2D** en haut au milieu :



Dernière étape, vérifiez que ce paramètre est bien actif :



PREMIERS PAS

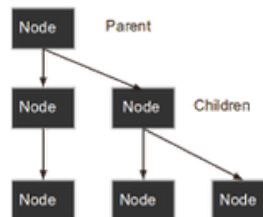
LES NODES

Avant de commencer à poser notre premier sprite, nous allons rapidement voir ce que sont les nodes et comment ça fonctionne.

Les nodes sont les blocs de construction fondamentaux pour créer un jeu. Godot propose de nombreux types de nodes différentes, chacune ayant un objectif spécifique. Cependant, une node donnée a toujours les caractéristiques suivantes :

- Chaque node possède un nom
- Chaque node a des propriétés modifiables
- Chaque node peut recevoir des événements à chaque frames
- Chaque node peut contenir d'autres nodes

Pour mieux comprendre, il faut voir les nodes comme un arbre :



Si vous avez un peu de mal à manipuler les nodes par moment, n'hésitez pas à venir nous demander de l'aide !

LES SCÈNES

Les scène dans Godot peuvent être un niveau, un personnage, une arme ou un item. Dans Godot, **lancer un jeu** se traduit par **lancer une scène**.

Une scène est composé d'un groupe de nodes organisé comme un arbre.
Basiquement, l'éditeur Godot est un éditeur de scène.

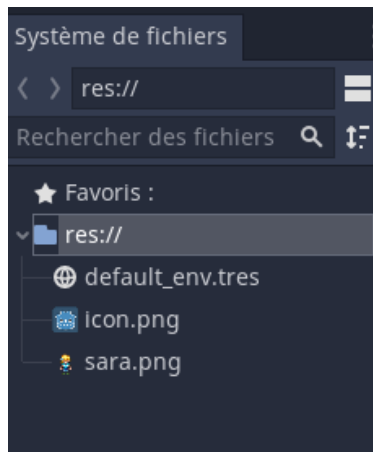
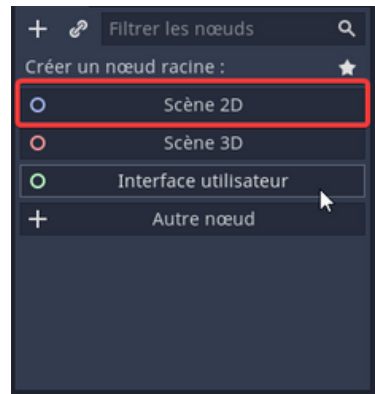
(Je vais m'arrêter ici pour les explications, si vous avez des questions, n'hésitez pas à aller voir sur internet ou nous demander directement (on est toujours gentils))

DÉVELOPPEMENT

AJOUTER UN SPRITE POUR LE JOUEUR

Ouf ! On va enfin passer à quelque chose de plus concret !

Pour commencer, nous allons tout simplement ajouter une scène 2D à la scène vide actuelle :

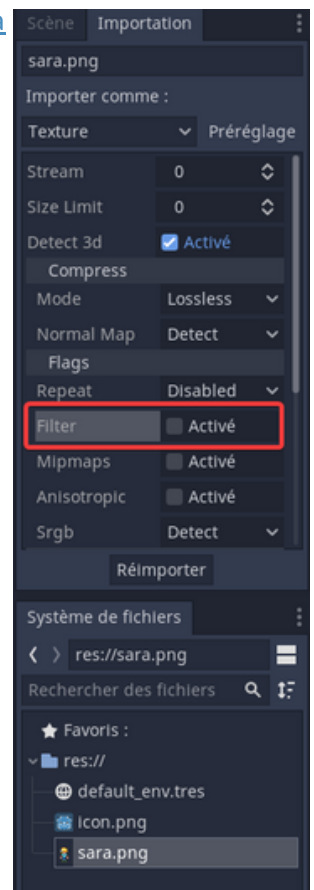


Nous allons ensuite télécharger notre premier sprite et l'ajouter aux fichiers de notre projet (vous pouvez simplement glisser-déposer) :

<https://raw.githubusercontent.com/KatsuKumi/SimpleRPG/main/sara.png>

Afin d'éviter que le sprite devienne flou lorsque nous l'utiliserons dans la scène, il est nécessaire de le réimporter en désactivant le filtre de Godot.

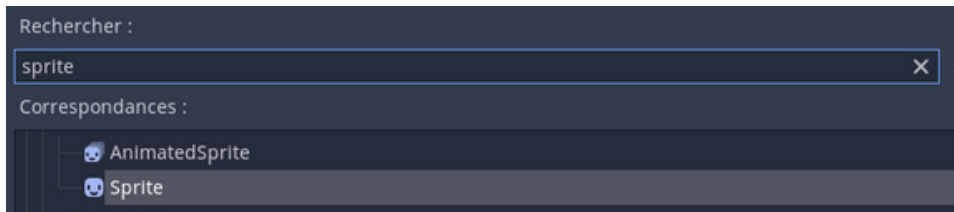
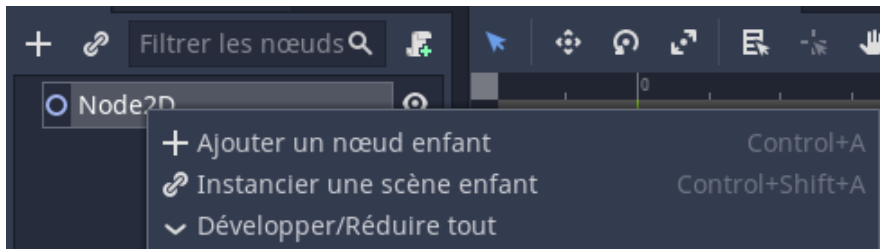
Pour cela, cliquez sur le sprite en question dans le système de fichier, allez dans l'onglet importation, désactiver le "Filter" et cliquez sur Réimporter



DÉVELOPPEMENT

AJOUTER UN SPRITE POUR LE JOUEUR

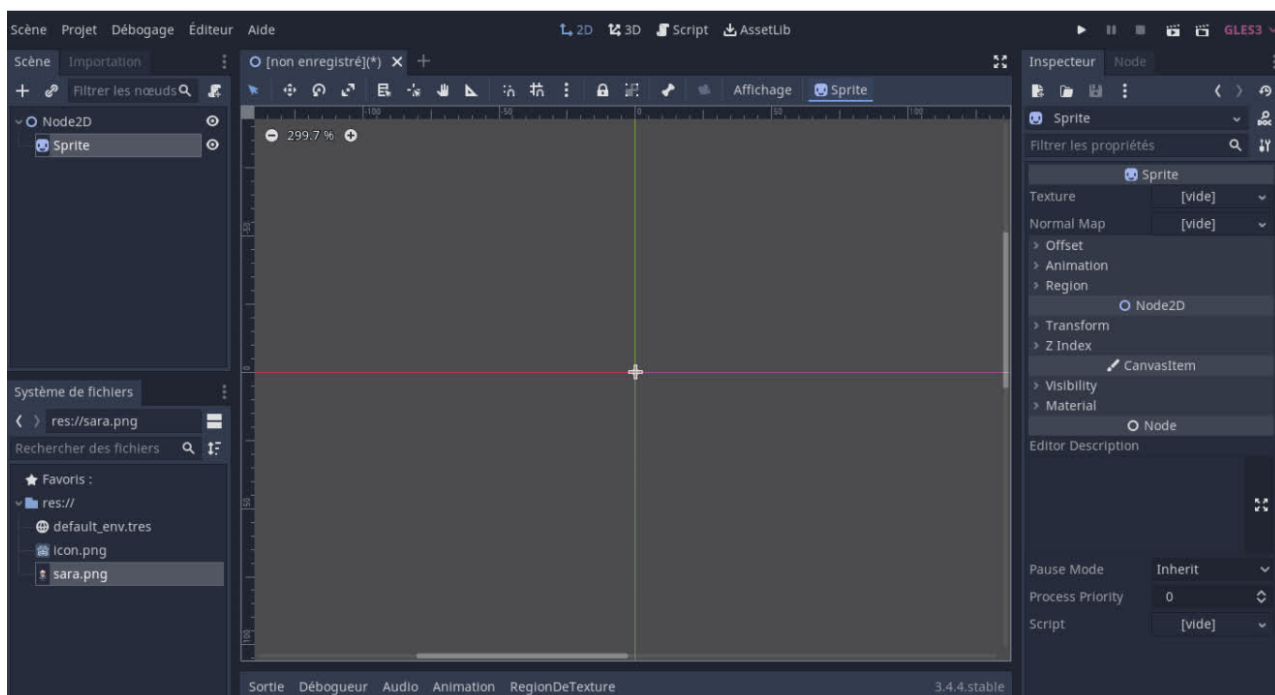
Maintenant que nous avons notre sprite, nous allons devoir le rajouter dans la scène. Pour cela, il suffit de faire un clic droit sur la Node2D et d'ajouter un nœud enfant de type "Sprite" :



Comme vous pouvez le constater, rien de spécial n'est apparu sur la scène, c'est normal car nous n'avons pas encore précisé quel sprite nous devons afficher.

Si vous regardez dans l'inspecteur, vous verrez apparaître les propriétés de notre node de type Sprite et aussi voir que la Texture est vide par défaut.

Afin d'ajouter le sprite il suffit de faire glisser notre image dans le carré vide :



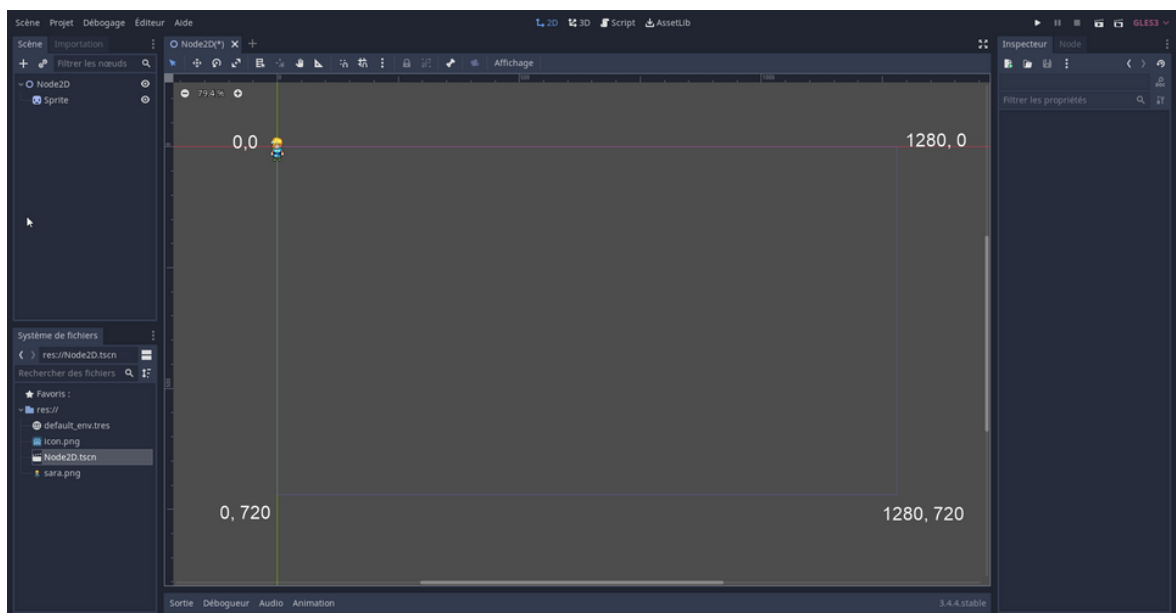
DÉVELOPPEMENT

AJOUTER UN SPRITE POUR LE JOUEUR

Dans l'inspecteur à droite, dans la section Node2D, vous pouvez voir une section "Transform", ouvrez là et vous y trouverez la position, la rotation et la taille (Scale) du sprite.

La première chose que nous pouvons voir sont les coordonnées (0, 0) de la node sélectionnée, qui correspond au coin haut gauche de notre fenêtre de jeu.

Si vous essayez de modifier les positions x et y, vous pouvez constater que la valeur x bouge le sprite de gauche à droite et la valeur y de bas en haut.



Nous voulons que notre joueur soit au centre de l'écran, nous allons donc modifier x et y de notre node pour centrer le joueur. Nous vous laissons faire le calcul ;)

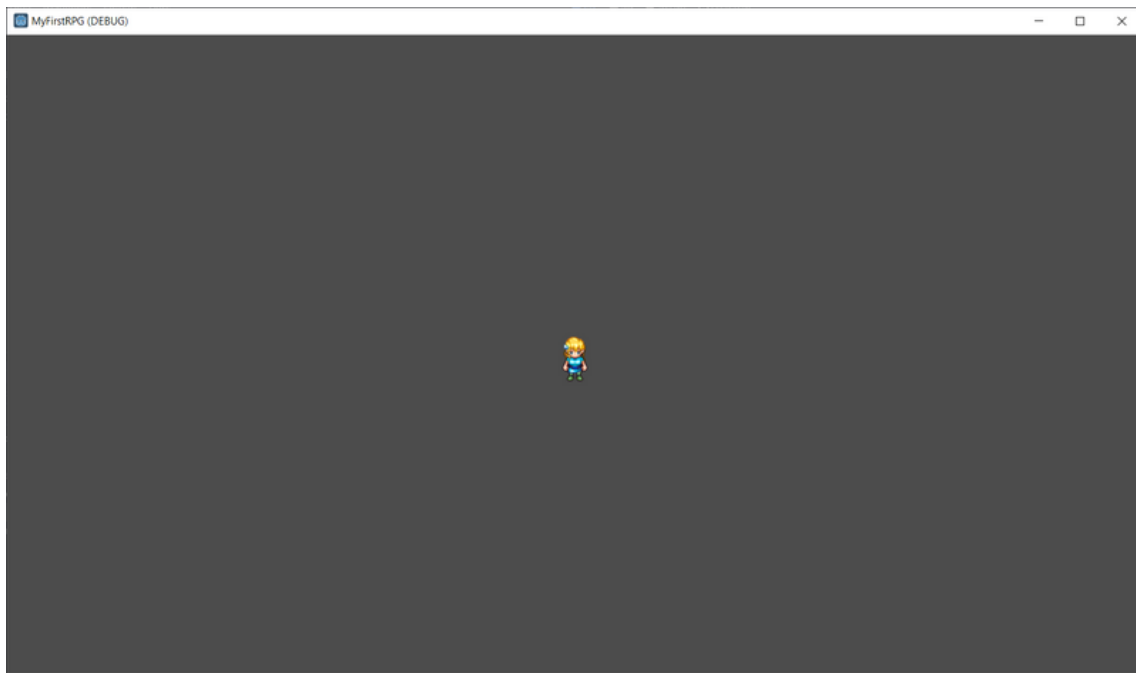
DÉVELOPPEMENT

LANCER LE JEU

Avant de pouvoir tester ce que nous avons fait jusqu'à maintenant, il est nécessaire de sauvegarder la scène actuelle et de l'assigner comme scène principale.

Cliquez sur **Scène → Enregistrer la scène** et appelée la "**Main.tscn**".
Maintenant ouvrez vos paramètres de projet et dans **Application → Run** modifiez le paramètre "Main Scene" en utilisant la scène que vous venez juste de créer.

Vous pouvez maintenant cliquer sur le bouton "Play" en haut à droite de votre éditeur pour lancer votre jeu !



Si votre fenêtre est trop grande, n'hésitez pas à modifier la Test Width et la Test Height dans les paramètres du projet pour une autre résolution en 16:9 :

- 1280x720
- 854x480
- 640x360

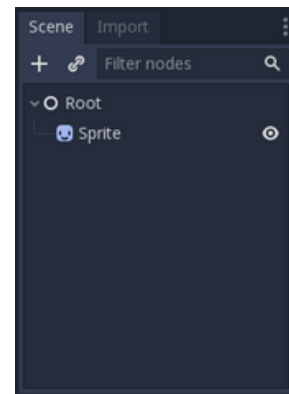
DÉVELOPPEMENT

ORGANISATION DES FICHIERS ET DES NODES

Nous vous conseillons d'organiser vos assets et vos scènes dans différents dossiers afin de pouvoir correctement gérer vos différentes scènes et assets. Cette partie n'est pas obligatoire mais vous pouvez le faire si vous voulez vous organiser.

Vous pouvez aussi supprimer les fichiers `icon.png` et `default_env.tres` qui ne seront pas utiles pour l'instant.

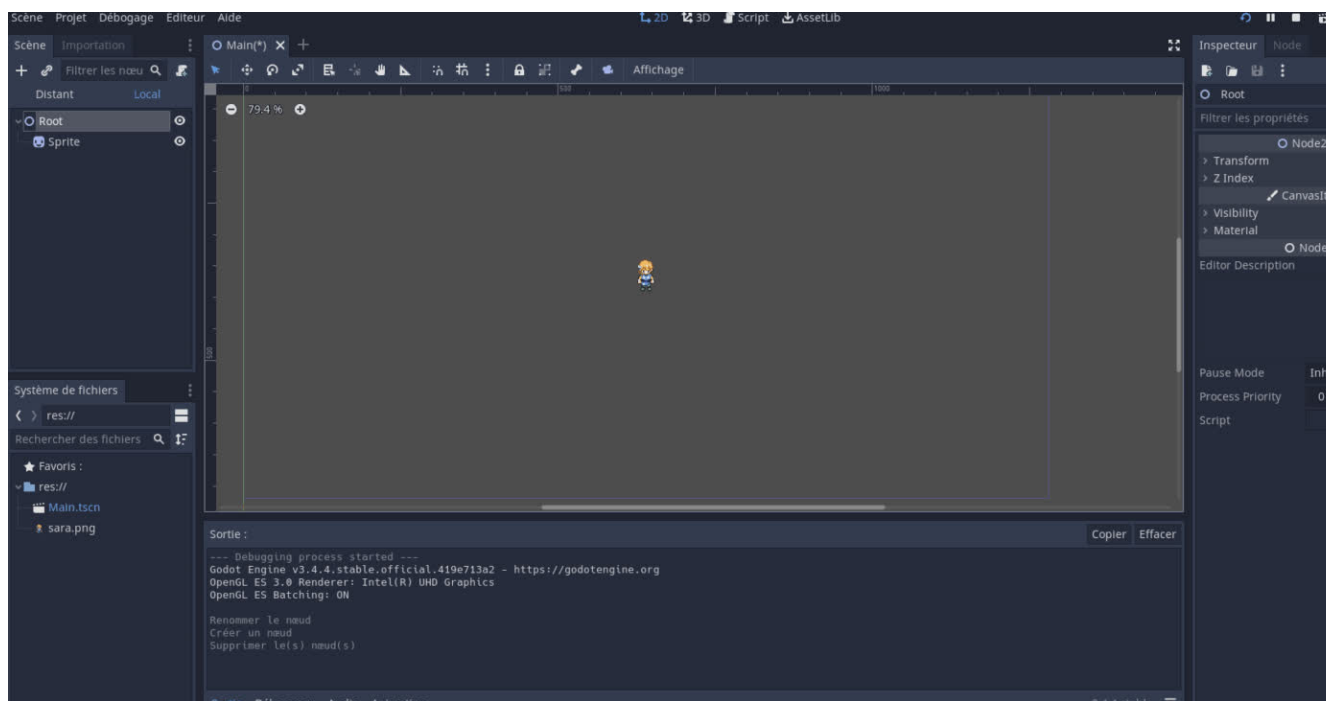
Afin de simplifier la compréhension, renommez la Node2D dans votre scène "Root" :



MOUVEMENT DU JOUEUR

Pour gérer le mouvement des joueurs, nous allons utiliser une node de type "KinematicBody2D". Les nodes KinematicBody2D sont des nodes spéciales correspondants à des éléments physiques pouvant être contrôlé par le joueur.

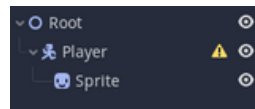
Pour commencer, ajouter une node enfant à Root de type KinematicBody2D. Déplacez ensuite votre sprite dans cette nouvelle node :



DÉVELOPPEMENT

MOUVEMENT DU JOUEUR

Afin de simplifier la compréhension, renommez la KinematicBody2D en "Player" :



Une chose importante à comprendre sur les nodes est que leurs positions est toujours relative à celle de leurs parents.

Maintenant que nous avons bougé le Sprite dans Player, la position du Sprite est relative à celle du Player.

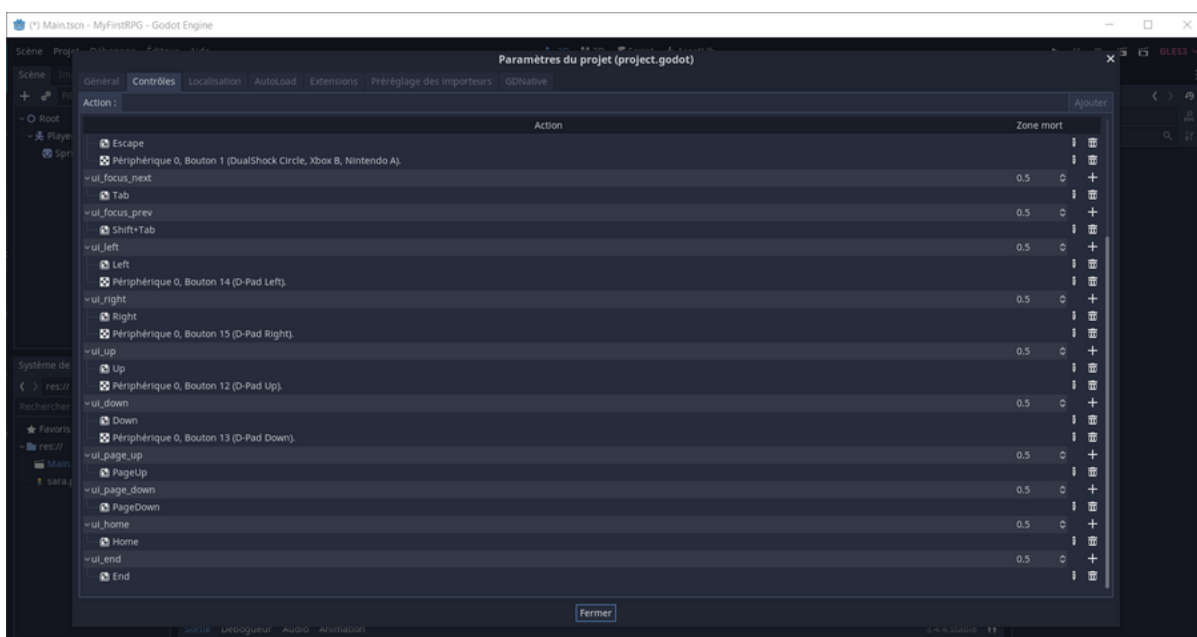
Étant donné que la node Player est celle que sera bougée par la suite, il est nécessaire d'inverser les positions des deux nodes depuis l'inspecteur :

- Player en 640, 360
- Sprite en 0, 0

INPUT DU JOUEUR

Par défaut Godot gère ses propres types d'input, vous pouvez y accéder dans **Projet** → **Paramètre** du projet dans l'onglet **Contrôles**.

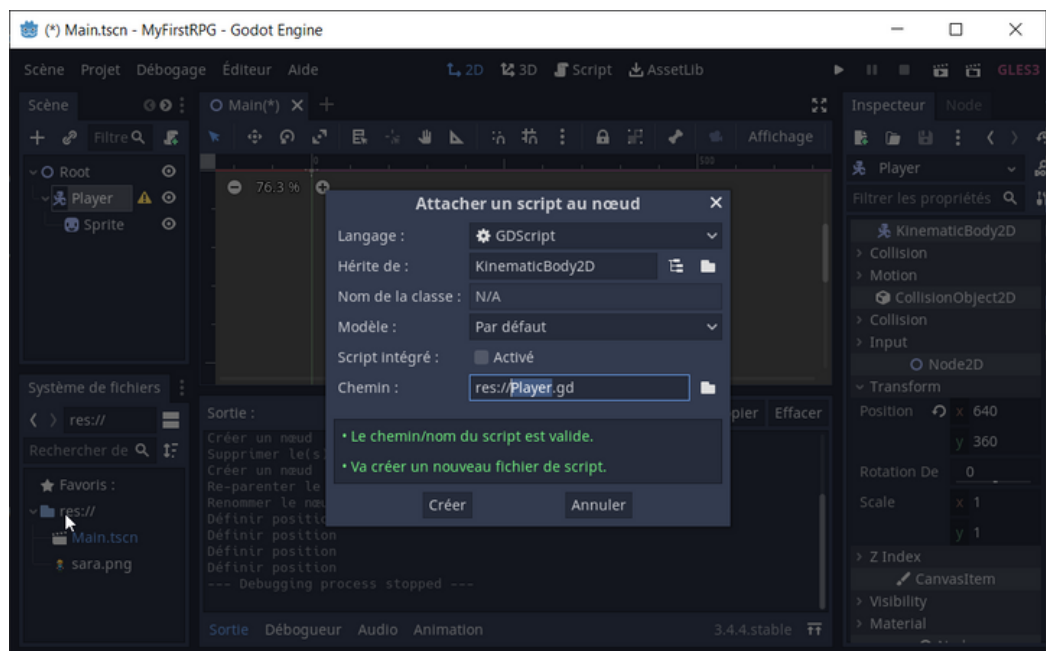
Nous utiliserons ici *ui_left*, *ui_right*, *ui_up* et *ui_down*, vous pouvez rajouter des contrôles si vous voulez (Mettre ZQSD par exemple).



DÉVELOPPEMENT

SCRIPT DE MOUVEMENT

Afin de rajouter un script pour gérer les mouvements du joueurs, faites clique droit sur la node "**Player**" et cliquez sur "**Attacher un script**"



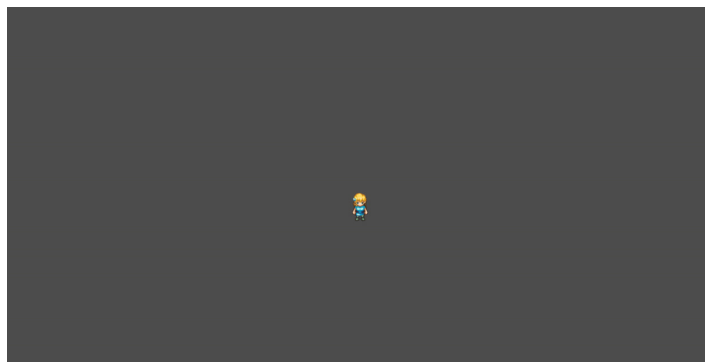
Vous n'avez rien à changer ici sauf si vous voulez ranger vos fichiers dans des dossiers. Vous pouvez ensuite cliquer sur **Créer**.

Maintenant que nous sommes dans l'espace de script, remplacez le code actuel par le code ci-dessous :

<https://raw.githubusercontent.com/KatsuKumi/SimpleRPG/main/Player1.gd>

Je n'expliquerais pas comment fonctionne le script en détails car cela pourrait prendre un peu trop de temps, si vous avez des questions, n'hésitez pas à venir nous voir !

Vous pouvez maintenant lancer votre jeu et vous devriez pouvoir bouger votre personnage avec les flèches directionnelles :

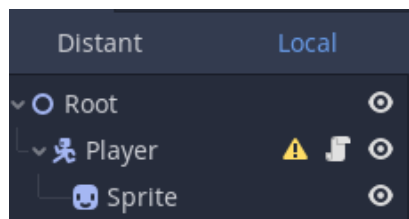


DÉVELOPPEMENT

LES COLLISIONS

Afin de pouvoir retourner sur votre scène, vous pouvez réouvrir le fichier Main.tscn en double cliquant dessus.

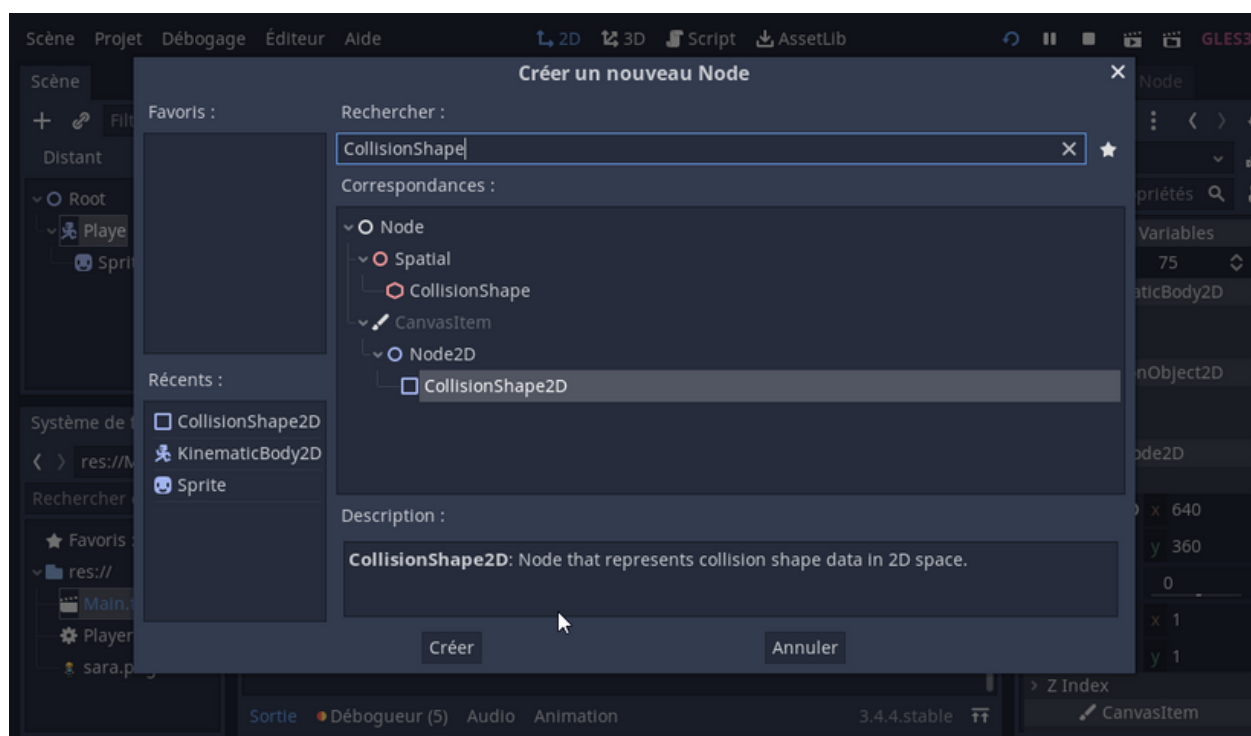
Je sais pas si vous remarquez (mais les profs...) mais lors des étapes précédentes un petit logo jaune était affiché à droite de votre node Player



Cet avertissement est dû au manque de formes de collision pour le Player. Les "Collision shapes" sont utilisés pour définir la forme des bordures et permettre la détection des collisions entre les objets.

La façon la plus commune est de rajouter une node enfant de type CollisionShape2D ou CollisionPolygon2D. Ces nodes vous permettent de dessiner vous même la forme dans l'éditeur.

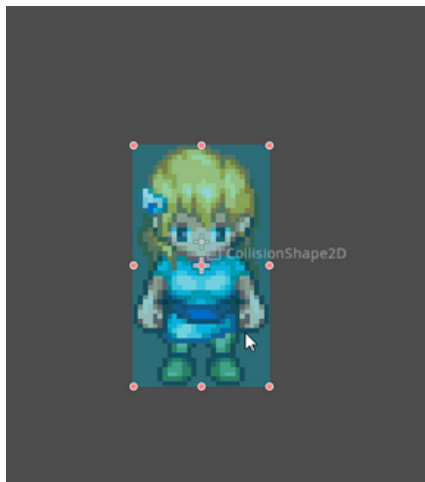
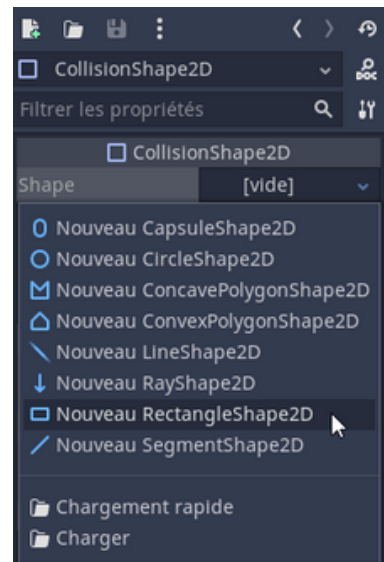
Pour commencer nous allons rajouter une node enfant à Player de type CollisionShape2D :



DÉVELOPPEMENT

LES COLLISIONS

Sélectionnez la node CollisionShape2D et dans l'inspecteur à droite, cliquez sur la shape [vide] et cliquez sur RectangleShape2D pour créer une nouvelle shape.



Modifiez la shape dans l'éditeur pour qu'elle corresponde au Sprite

Aucune différence n'est visible pour l'instant, mais cette étape aura son importance pour la suite de l'atelier.

DÉVELOPPEMENT

LES TILESETS

Nous allons maintenant voir comment mettre en place et manipuler des tilesets, littéralement pile de tuile. Nous utiliserons le mot "tile" pour le reste de l'atelier.

Une tile ou tuile en français est un morceau de terrain dans certains jeux 2D. Dans ceux-ci, la carte est divisée en un quadrillage, comportant des carrés ou rectangles de même taille, qui permettent de simplifier la gestion du terrain en utilisant simplement un tableau.

Godot contient plusieurs systèmes pour gérer proprement les tilesets et pour faire en sorte de simplifier la création d'une carte. Et c'est ça que nous allons maintenant faire.

Il existe 3 types de tiles différentes :

- **Single tiles** : Ce sont des tiles qui peuvent être utilisés de façon individuelle
- **Autotiles** : C'est un système utilisé dans plusieurs éditeurs de carte dont celui de Godot qui permet de créer rapidement des grosses parties de la carte sans avoir à placer toutes les tiles de direction une par une. Cet algorithme va détecter automatiquement les bordures et la partie interne et sélectionnera automatiquement la bonne tile. (Exemple : Un lac, une île)
- **Tile Atlases** : Un atlas est simplement une collection de tile du même type regroupé ensemble, cela permet notamment de dessiner de la végétation comme des fleurs de façon aléatoire.

Pour commencer, téléchargez les tilesets pour les terrains :

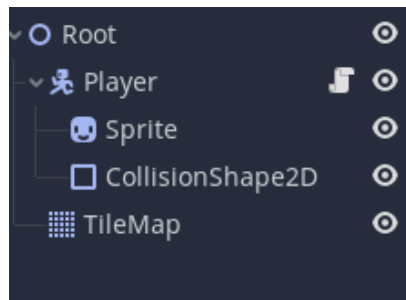
<https://raw.githubusercontent.com/KatsuKumi/SimpleRPG/main/tiles/Terrains.png>

Déplacez ensuite l'image dans les fichiers du projet et n'oubliez pas de désactiver le "Filter" dans l'importation du tile. (Même chose qu'à la page 11)

DÉVELOPPEMENT

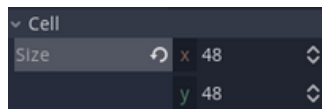
LES TILESETS

Ensuite, créez une node de type "Tilemap" enfant de la node "Root" (Comme les fois d'avant mais avec Root comme parent)

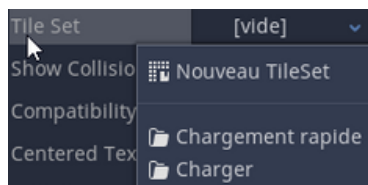


Une tilemap est une sorte de grille pour les tiles utilisée pour créer une carte de jeu. Il y a pleins d'avantages à utiliser une tilemap. Pour commencer, vous pouvez simplement peindre la carte avec des tiles sur la grille, ensuite vous pouvez dessiner de très très grandes maps car la tilemap est optimisée pour gérer cela.

Les tiles que nous allons utiliser dans notre jeu font une taille de 48x48 pixels, donc dans l'inspecteur, nous avons besoin de modifier les sizes x et y des Cell :



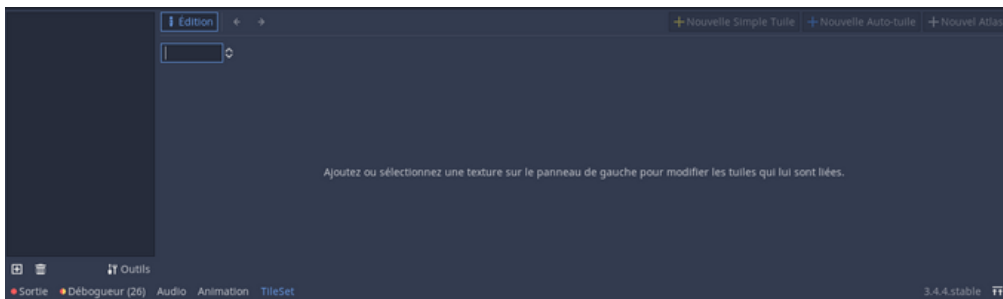
Nous allons ensuite créer un nouveau tileset qui sera utilisé par la TileMap. Dans l'inspecteur, cliquez sur [vide] à côté de TileSet et cliquez sur "Nouveau TileSet"



DÉVELOPPEMENT

LES TILESETS

Cliquez maintenant sur la TileSet que vous venez de créer dans l'Inspecteur et une nouvelle fenêtre devrait s'ouvrir :

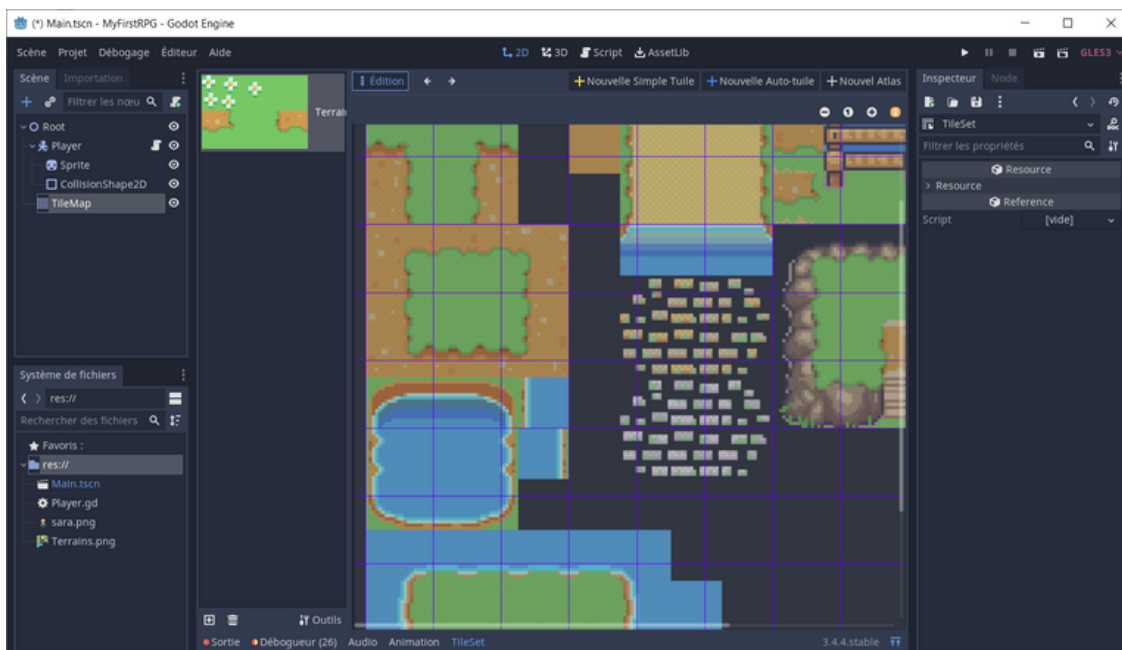


Si le souhaitez, vous pouvez agrandir la fenêtre en faisant Shift + F12

AUTOTILE DE L'HERBE

Pour commencer, nous allons créer une autotile qui nous permettra de dessiner de l'herbe qui aura une texture plus ou moins aléatoire.

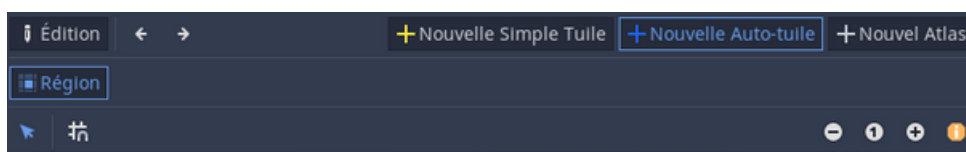
Ajouter l'image du tileset (Terrains.png) en cliquant sur le bouton [+] en bas à gauche de la fenêtre et sélectionnez le tileset :



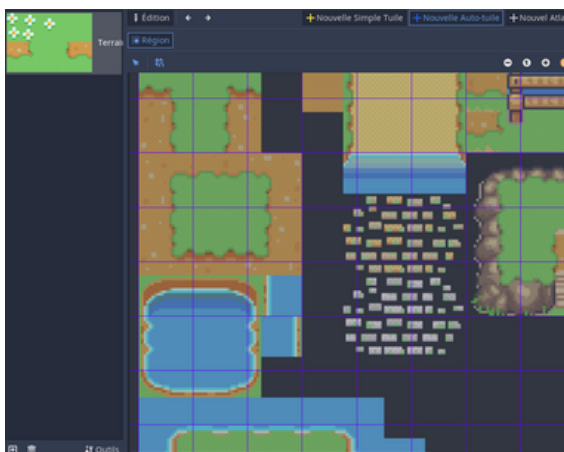
DÉVELOPPEMENT

AUTOTILE DE L'HERBE

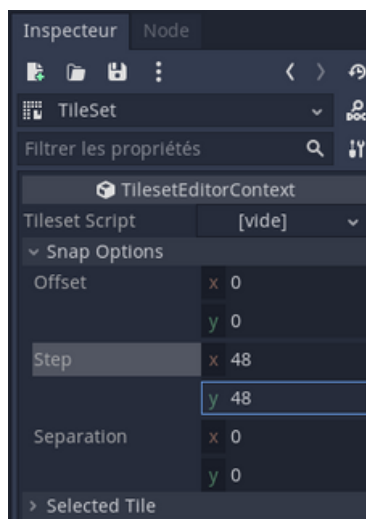
Cliquez sur le bouton "Nouvelle Auto-Tuile" et une nouvelle bar d'outils devrait apparaître :



Cliquez sur le deuxième bouton de cette bar à outils avec l'aimant pour activer l'aimantation, il est très probable que la grille apparaisse de la mauvaise taille :



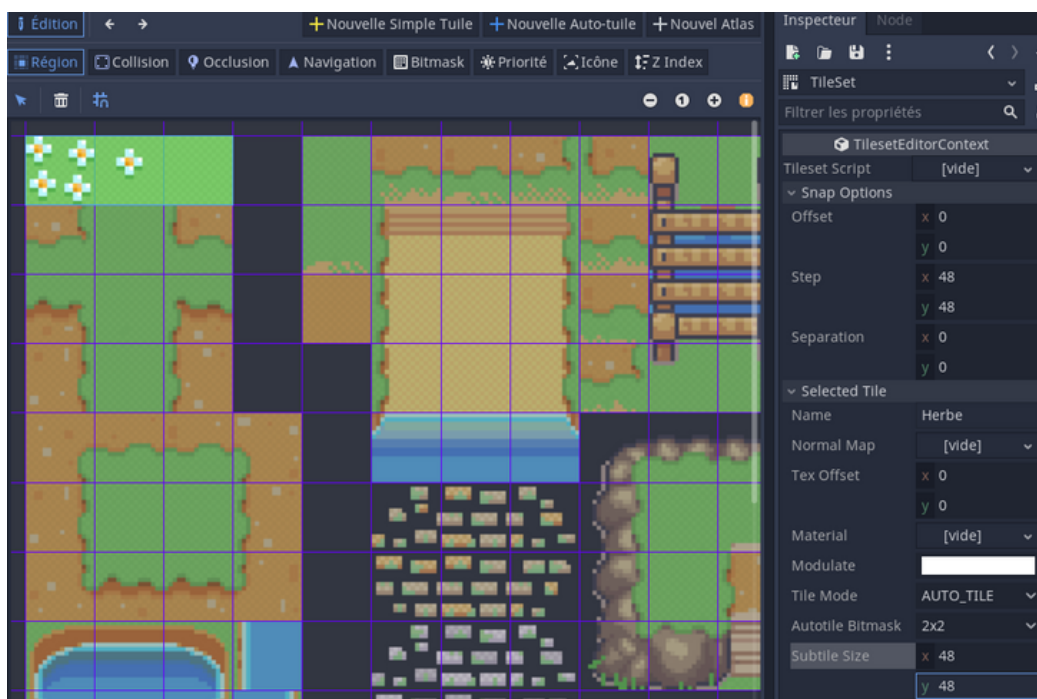
Changer la taille de la grille n'est pas du tout intuitif. Pour faire cela, sélectionnez un endroit aléatoire de la grille et la propriété "Snap Options" devrait apparaître dans l'inspecteur. A partir d'ici, vous pourriez changer la valeur x et y de Step par 48x48 :



DÉVELOPPEMENT

AUTOTILE DE L'HERBE

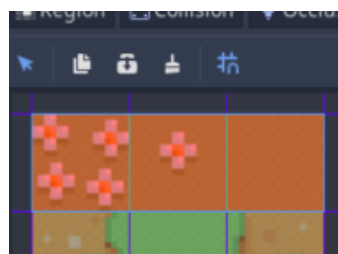
Maintenant, sélectionnez les 3 tuiles d'herbes en haut à gauche (avec les fleurs et sans les fleurs), ensuite à droite dans l'inspecteur, ouvrez la section "Selected Tile" et modifiez Subtile Size à 48x48 et modifier le Name pour que celui ci soit plus compréhensible :



Après avoir sélectionné vos tuiles, plusieurs onglets apparaîtrons :

- Collision : C'est avec ça que nous dessinerons les collisions pour le personnage
- Occlusion : Utilisé pour la gestion de lumière
- Navigation : (Je sais plus mais ce n'est pas utile pour nous)
- Bitmask : Permet de configurer le système d'autotile
- Priorité : Permet de gérer l'aléatoire des tiles en utilisant des priorités
- Icône : Permet de spécifier quelle tuile utiliser comme icône
- Z Index : Permet de préciser Z Index pour chaque tile

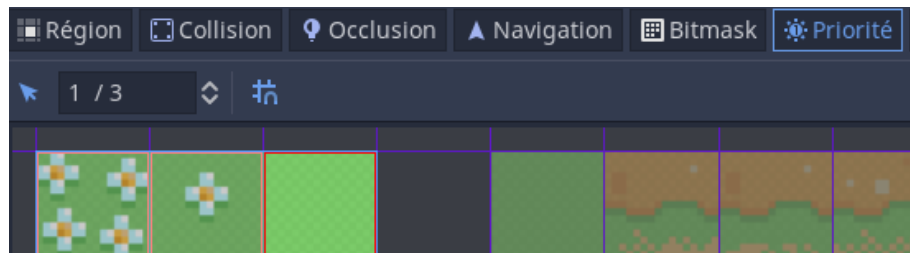
Pour commencer allez dans Bitmask et avec la souris, sélectionnez les 3 tuiles d'herbe en entier. (Vous verrez que chaque tuile est divisé en 2x2, sélectionnez chaque carrés) :



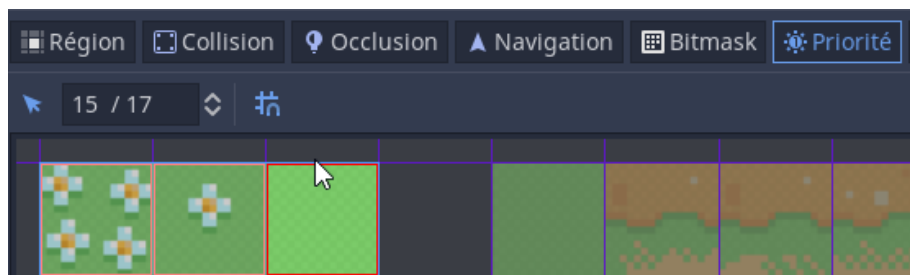
DÉVELOPPEMENT

AUTOTILE DE L'HERBE

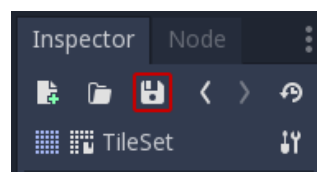
Maintenant allez dans l'onglet Priorité et cliquez sur la tuile sans les fleurs à droite :



En cliquant sur chaque tuile, vous pourrez voir que la priorité de chaque tuile est de 1/3 par défaut. Sélectionnez la tuile sans les fleurs et augmentez (de au moins 15) la priorité pour en faire la tuile la plus utilisée.



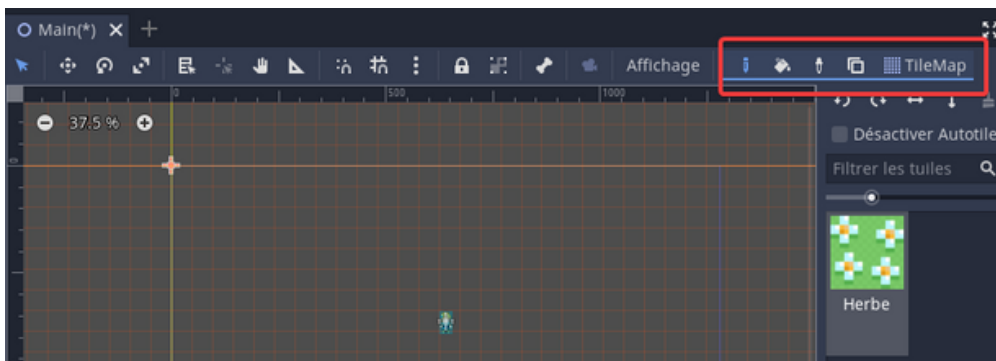
Maintenant que cette tileset est prête, sauvegardez la à l'aide du bouton sauvegarder dans l'inspecteur :



DÉVELOPPEMENT

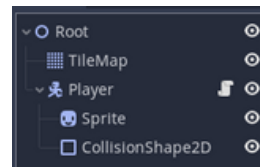
AUTOTILE DE L'HERBE

Retournez sur votre scène et cliquez sur votre TileMap, votre tile devrait apparaître à droite de votre espace de travail et un nouveau menu sera affiché :

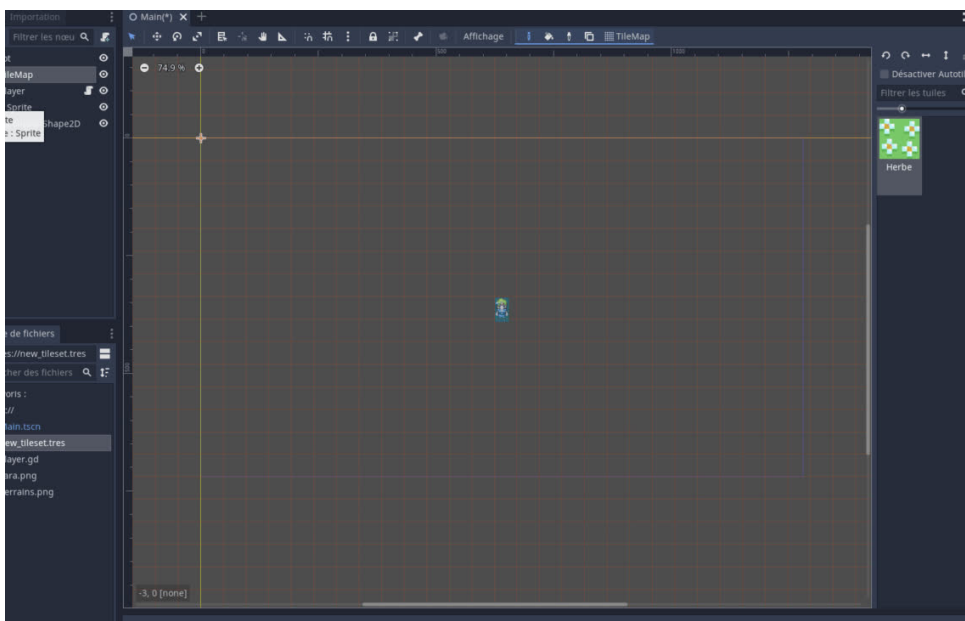


Maintenant vous pouvez vous amuser à dessiner votre première carte, mais avant deux petites chose à faire:

Pour commencer, décochez la case "Désactiver Autotile" si elle est cochée et ensuite mettez votre la TileMap au dessus du joueur dans vos Nodes :



Vous pouvez maintenant dessiner votre carte (Amusez vous à tester les différents outils pour comprendre comme ils fonctionnent, pour la prochaine étape, faites une carte plus grande que votre taille de jeu (définis par les bordure bleus)



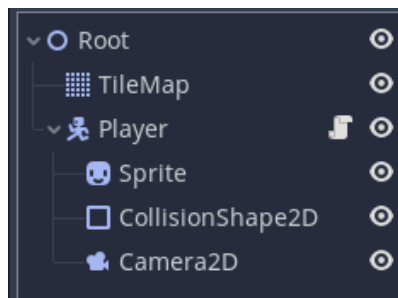
DÉVELOPPEMENT

LA CAMÉRA

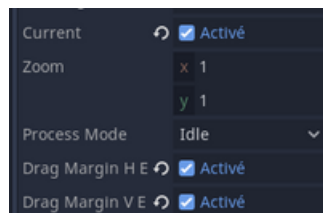
Après avoir lancé le jeu sur un map plus grande que le jeu, vous pouvez constater que la caméra ne bouge pas et que le joueur peut quitter l'écran.

Ce n'est évidemment pas ce qu'on veut ici, nous voulons une caméra qui puisse suivre le joueur.

Nous allons donc créer une Camera2D (encore des nodes) comme une node enfant de Player :



Cliquez sur la node Camera2D et activez Current et les deux options "Drag Margin" :



Lancez votre jeu et vous devriez pouvoir voir votre caméra qui se déplace quand vous déplacez le joueur.

Avertissement : Il est possible que la carte clignote quand elle se déplace, pour cela plusieurs choses sont à vérifier :

- Dans les paramètres du projet dans **Rendering** → **2D**, activez l'option "Use Gpu Pixel Snap"
- Dans votre TileMap dans l'inspecteur, activez l'option "Y Sort"
- Dans les paramètres du projet dans **Rendering** → **Quality** désactivez l'option "Hdr" dans "Depth" tous en bas

DÉVELOPPEMENT

AUTOTILE DU SABLE

L'herbe c'est bien mais ça manque un peu de diversité non ?

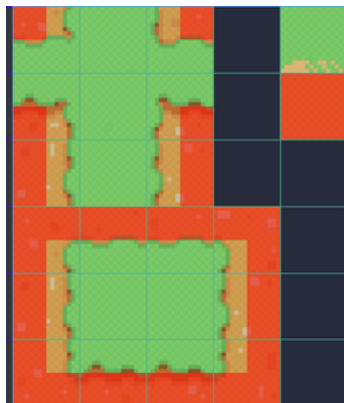
Ouvrez votre tileset (qui est normalement disponible dans le système de fichier) et cliquez sur "Nouvelle Auto-tuile", sélectionnez ensuite toute cette partie de votre tileset :



Allez ensuite dans l'onglet "Bitmask".

Le bitmasking est une méthode qui permet de sélectionner automatiquement les sprites appropriés d'une autotile. Cela fonctionne en paramétrant quelle partie sont les bordures de votre sprite ou non.

Maintenant sélectionner les bordures de votre sable comme ceci :



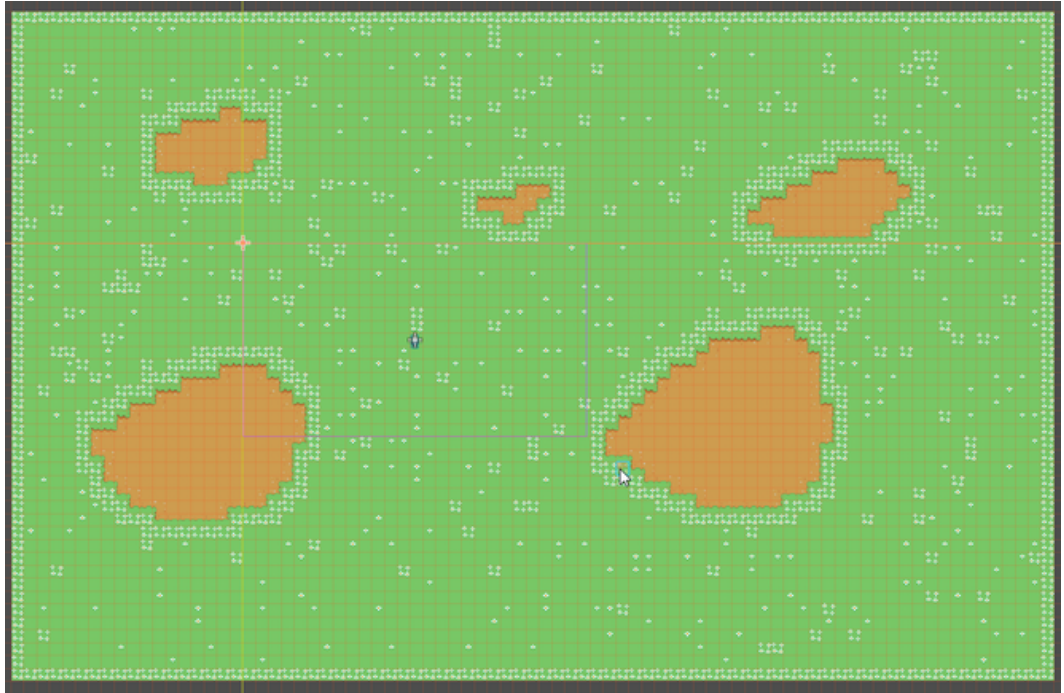
Si vous sélectionnez un carré par erreur, vous pouvez le désélectionner en faisant clique droit.

Une fois votre bitmask terminé, sélectionnez l'icône que vous souhaitez, changer le nom dans l'Inspecteur en "Sable" et sauvegardez.

DÉVELOPPEMENT

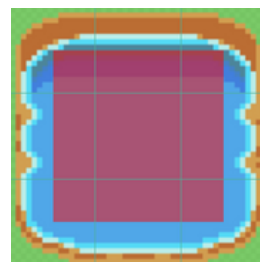
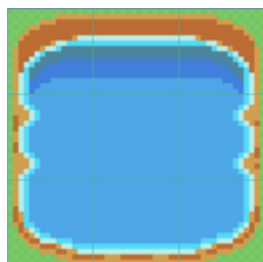
AUTOTILE DU SABLE

Maintenant que vous avez paramétré votre autotile, vous pouvez l'utiliser sur votre carte pour rajouter quelques portions de sable un peu partout :



AUTOTILE DU LAC

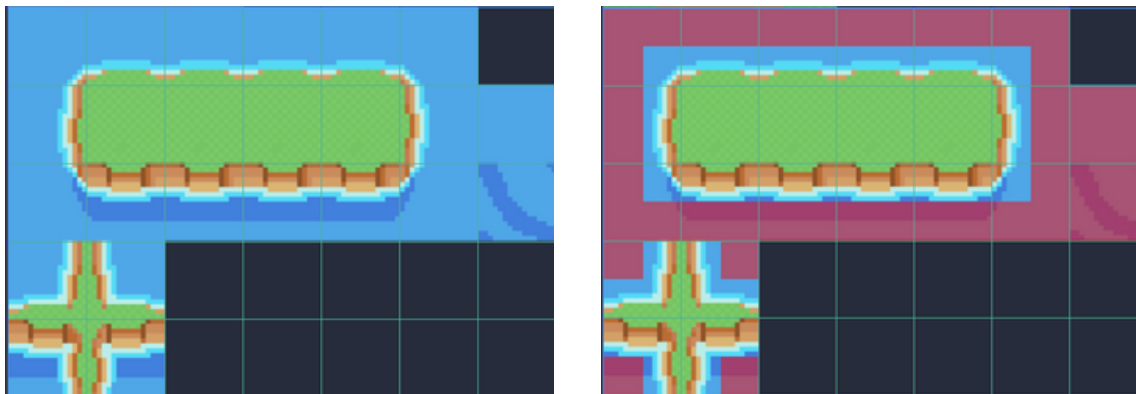
Vous avez un peu la main maintenant, voici donc la prochaine autotile en deux images :



DÉVELOPPEMENT

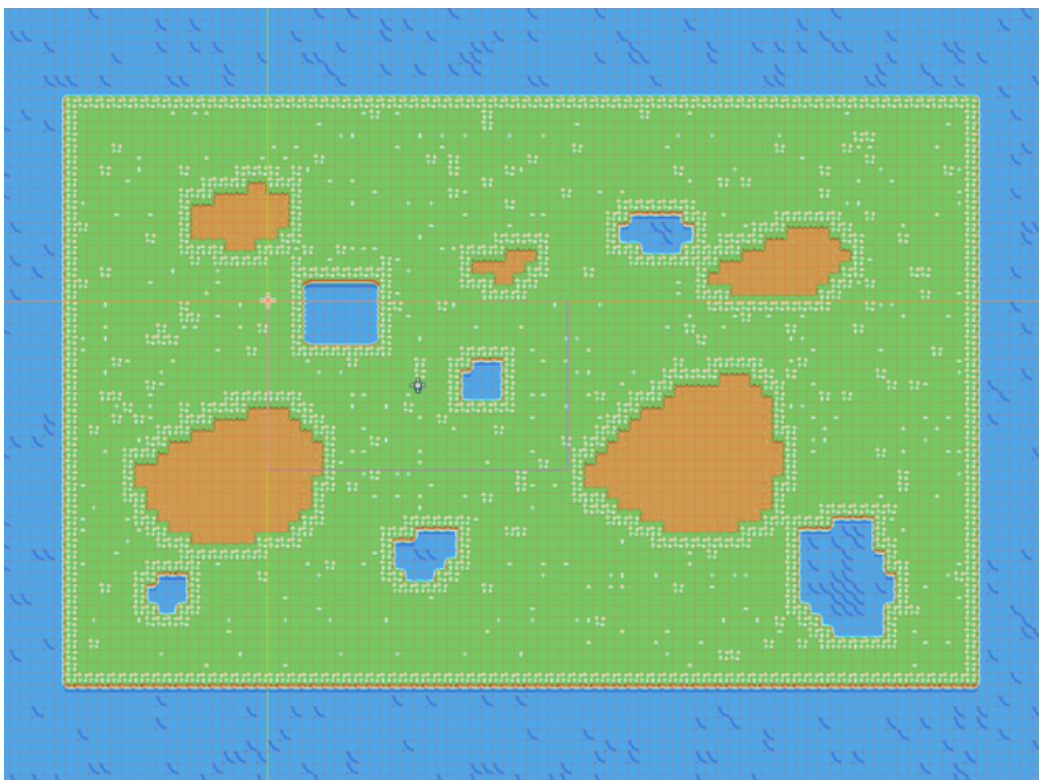
AUTOTILE DE L'EAU

Un peu similaire au lac mais avec plus de possibilité :



Lorsque vous ajoutez votre tile sur la map, vous pouvez désactiver l'autotile pour pouvoir utiliser individuellement, dans notre cas, nous pouvons le faire pour créer une bordure d'eau autour de la carte.

Amusez vous à faire votre carte, voici la mienne (J'ai fait une carte carré mais vous pouvez faire une carte plus développée ;)) :



DÉVELOPPEMENT


LES COLLISIONS

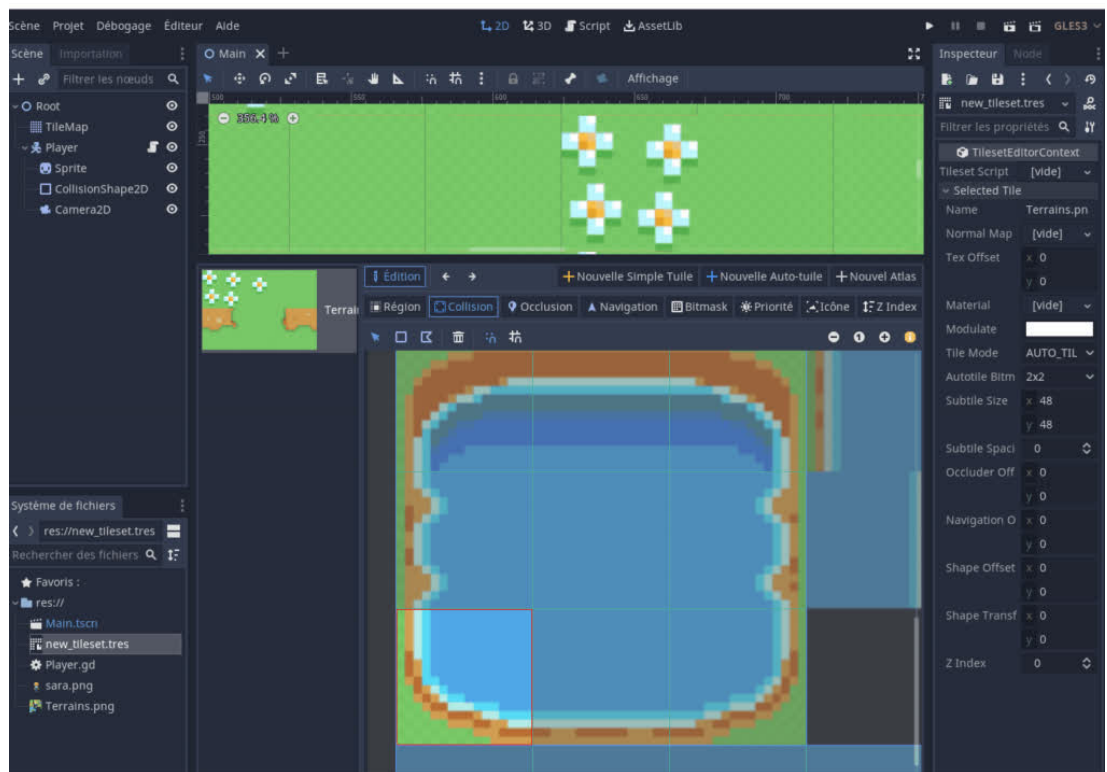
Maintenant que votre map est toute belle et pour terminer cet atelier, nous allons mettre en place les collisions avec le décors.

Cette tâche peut être assez compliqué à prendre en main et est assez longue car il faut définir chaque collision pour chaque sprite.

Nous allons retourner sur notre tileset et sélectionner notre lac. Ensuite nous allons aller dans l'onglet "Collision" :



Comme vous pouvez le voir, une nouvelle barre d'outils est apparue, pour commencer désactiver (comme sur l'image) l'aimant de droite et activer celui de gauche. Ensuite cliquez sur la deuxième forme  et commencez à dessiner votre forme de collision :



Faites ceci pour toutes les tiles où vous voulez des collisions et lancez votre jeu pour tester si les collisions fonctionnent correctement !

N'hésitez pas à changer la box collider du personnage pour que celle-ci ne bloque pas au niveau de la tête.

CONCLUSION

MERCI BEAUCOUP !

Merci beaucoup d'avoir participé à cet atelier, nous espérons que cela vous à plu !

La prise en main d'un moteur comme Godot peut être assez complexe mais globalement, ce sont des outils qui permettent de faire quasiment tout ce qu'on a envie de faire et de rendre certaines parties du développement beaucoup plus simple.

Si vous voulez continuer à travailler sur votre RPG, n'hésitez pas à suivre le tutoriel de **Davide Pesce** qui m'a grandement aidé à écrire cet atelier :

<https://www.davidepesce.com/godot-tutorials/>

D'autres sprites sont disponibles sur mon Github si vous avez besoin d'autres ressources pour continuer votre jeu :

<https://github.com/KatsuKumi/SimpleRPG>

Il est possible que d'autres ateliers soit mis en place dans la continuité de celui-ci, si vous avez des questions vis à vis de ça, n'hésitez pas à contacter les représentants d'Epitech !

Et si vous avez d'autres questions concernant cet atelier, n'hésitez pas à me contacter par mail : **alexandre.vigeant@epitech.eu**