Débuter sous Godot 4

Ce guide de démarrage vous accompagnera dans vos premiers pas avec Godot 4, le moteur de jeu open source, libre et gratuit. Nous allons créer un mini projet 2D pour acquérir les bases.

Qui est l'auteur de ce guide ?

Je suis Anthony Cardinale, ingénieur en informatique et développeur de jeux certifié. J'ai publié plusieurs livres sur la création de jeux dont un sur Godot aux éditions d-booker. J'ai également réalisé des formations vidéo sur Godot disponibles sur Udemy. Je suis également actif au sein de la communauté Godot notamment sur le développement d'extensions pour Godot. Vous pouvez retrouver mes tutoriels sur YouTube ou sur mon site web principal. Vous pouvez également me retrouver sur LinkedIn.

Pourquoi Godot ?

Comme indiqué ci-dessus, Godot est un logiciel gratuit et open source. Godot est au développement de jeux ce qu'est Blender pour la modélisation 3D. Avec Godot vous pouvez librement créer vos projets commerciaux sans aucunes restrictions. Godot est un logiciel puissant qui permet de créer n'importe quel type de projet pour à peu près n'importe quelle plateforme. Les langages de programmation utilisables sont GD Script, C#, C++ et un langage visuel.

Quel langage de programmation utiliser ?

La plupart des tutoriels que vous trouverez sur le net vous feront utiliser GD Script. La raison est simple : C'est le langage historique de Godot, celui qui est le mieux documenté et le plus facile à apprendre.

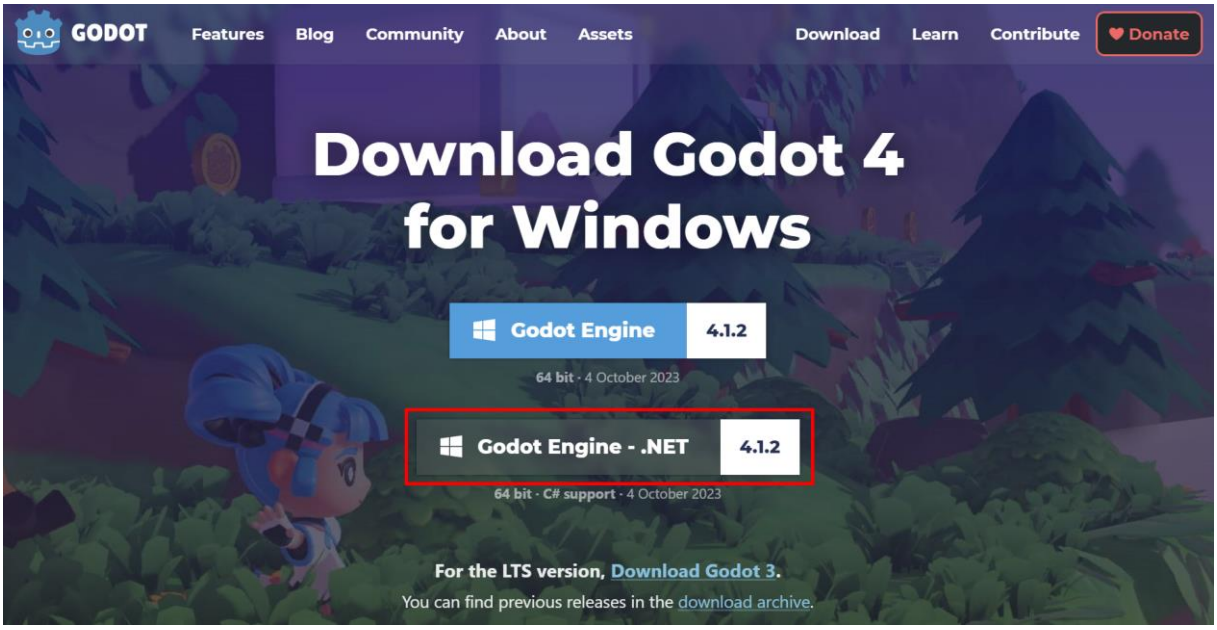
Pour ma part j'ai fait le choix de m'orienter vers le langage C# et d'utiliser ce langage de programmation au travers de ce guide.

Pourquoi C# ? Car je pense que ce langage va devenir le langage principal de Godot. En 2009, 90% des utilisateurs d'Unity utilisaient Javascript. Aujourd'hui presque 100% des développeurs Unity utilisent C#. Je pense que nous allons voir le même mouvement avec Godot. C# est massivement utilisé dans le monde du jeu vidéo. Pour moi, migrer vers Godot est plus simple en passant par C#. De plus, les benchmarks ont montré que C# est beaucoup plus performant que GD Script.
Enfin, je trouve que C# est un langage plus adapté au travail d'équipe ou simplement pour écrire un « beau » programme.

Vous proposer un guide basé sur C# me permet aussi de combler le vide qui existe en termes de ressources pédagogiques pour Godot axées C#.

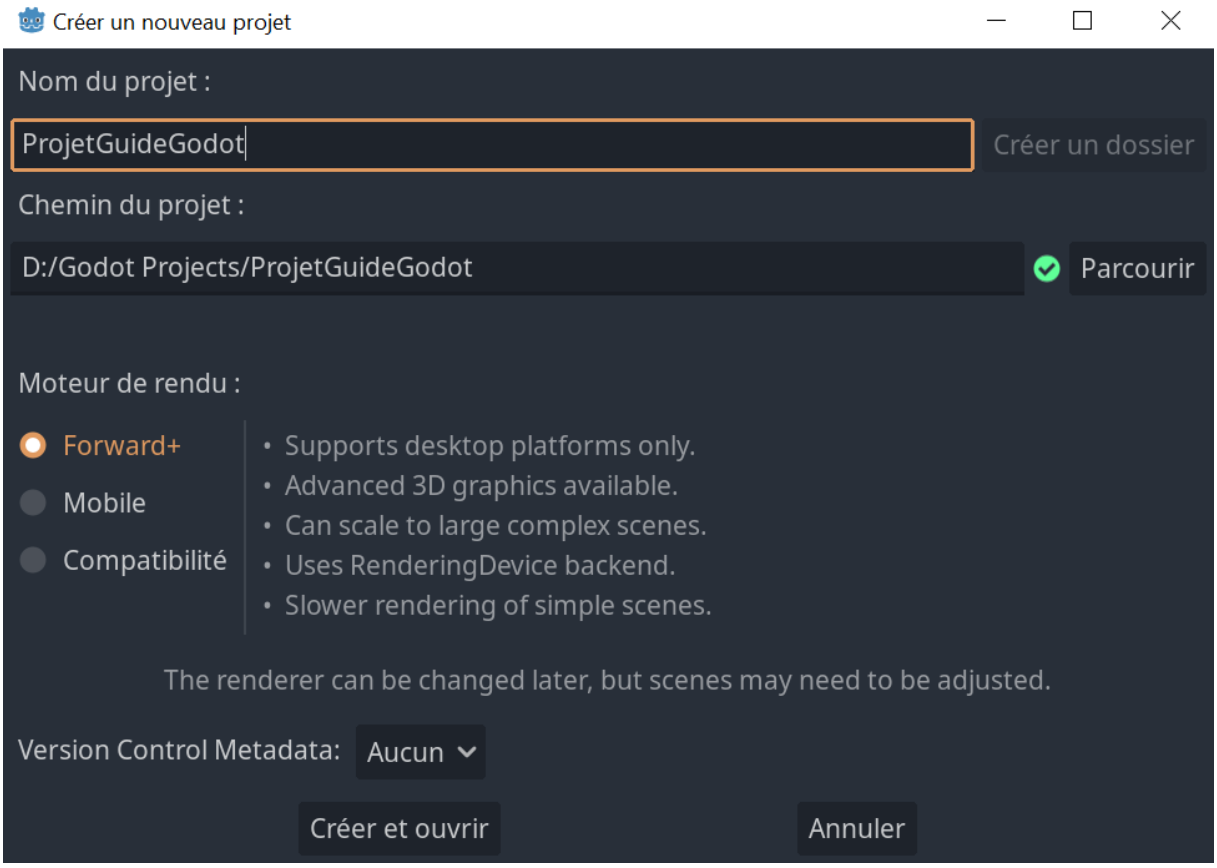
Télécharger Godot

Rendez-vous sur la page de téléchargement du site officiel : https://godotengine.org/download puis récupérez la version .NET de Godot :



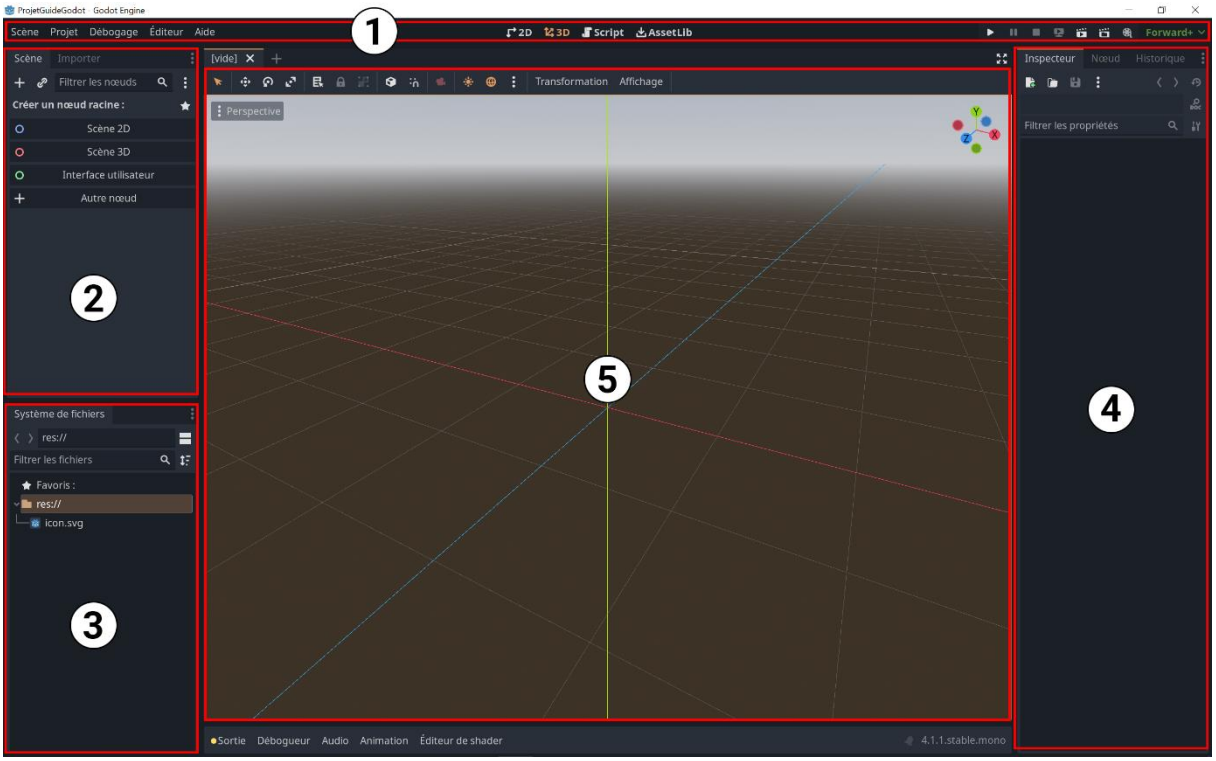
Créer un projet

Ouvrez Godot afin d'afficher le gestionnaire de projets. Ici, créez un nouveau projet standard :



L'interface de Godot 4

J'ai découpé l'interface de Godot en 5 zones principales :



1. Les **menus** : Permettent d'accéder aux principales fonctionnalités de l'éditeur, de changer de vue 2D / 3D / Script et de tester son jeu.
2. La **scène** : Il s'agit de l'arborescence de scène actuelle. C'est la liste des nœuds créés sur cette scène.
3. Le **système de fichiers** : Ce sont toutes les ressources que vous avez importés dans votre projet.
4. **L'inspecteur** : Affiche les caractéristiques et paramètres du nœud sélectionné.
5. Le **viewport** : C'est ici que vous pourrez construire vos niveaux.

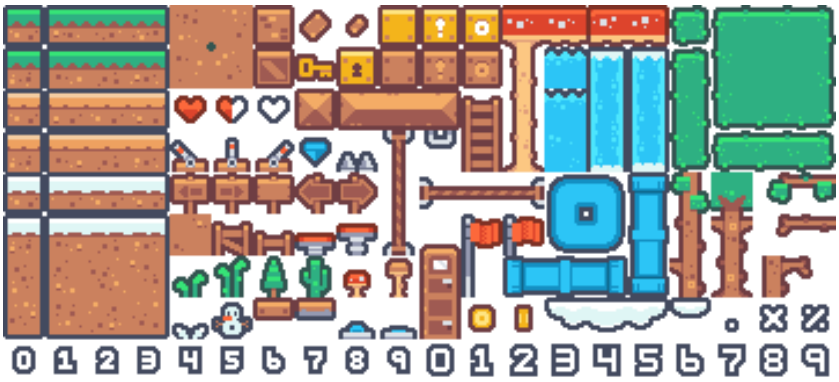
J'ai parlé de nœuds à plusieurs reprises. Dans Godot, chaque élément, chaque composant est un nœud. Un personnage 2D sera par exemple composé d'un nœud pour gérer son affichage et d'un nœud pour gérer les collisions. Plusieurs nœuds reliés forment une branche et plusieurs branches forment un arbre de scène.

Importer des ressources

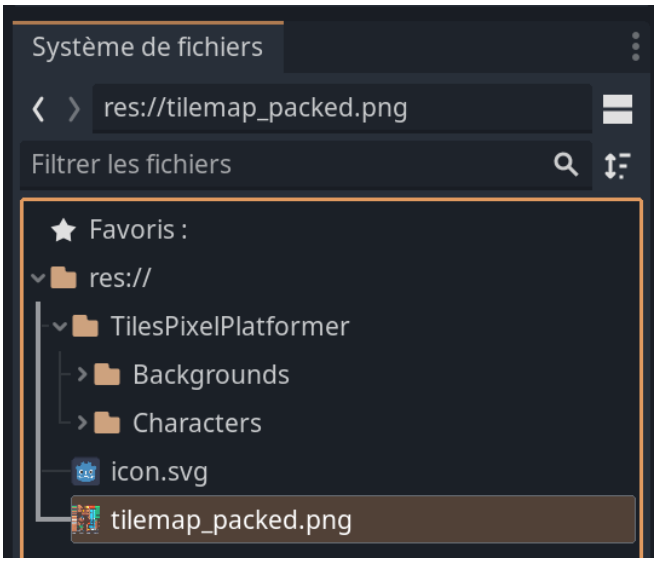
Pour importer des assets il suffit de les faire glisser/déposer dans le système de fichiers.

Rendez-vous sur un site web proposant des assets comme le site de Kenney (https://www.kenney.nl/assets). Et récupérez un pack d'assets (Exemple : https://www.kenney.nl/assets/pixel-platformer). Une fois le zip téléchargé et extrait, vous avez accès aux ressources. Les packs trouvés en lignent contiennent souvent plusieurs versions des assets pour différents moteurs de jeux. Vous n'aurez donc pas besoin de tout importer dans Godot mais juste le nécessaire. Pour cet exemple nous allons créer un décor avec des tuiles ainsi qu'un personnage qui marche. Nous aurons donc besoin d'un TileSet et de quelques Sprites.

Dans mon exemple, je vais récupérer le TileMap contenu dans un sous dossier. (Le tilemap contient toutes les tuiles collées en une seule image contrairement aux tuiles qui sont indépendantes). Ce TileMap existe en deux versions, soit avec les tuiles collées (ce que je vais utiliser) soit les tuiles espacées de quelques pixels. Je vais donc faire glisser cette image dans Godot :



Je vais aussi faire glisser/déposer les personnages (tuiles indépendantes) et les arrière-plans. Dans mon système de fichiers j'aurai donc :



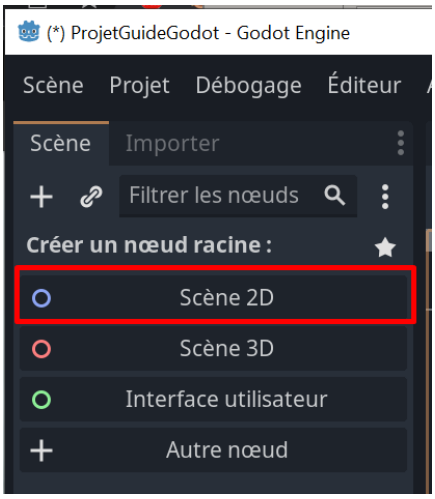
Vous devez vous organiser en créant des dossiers afin que ce soit plus facile de vous y retrouver par la suite.

Créer des nœuds sous Godot et ajouter un TileMap

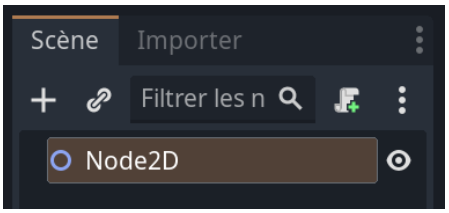
Maintenant que les images sont importées, nous allons pouvoir les configurer dans Godot afin de créer un TileSet utilisable.

Pour cela, nous allons avoir besoin de créer un nœud. Dans Godot, les scènes sont composées de nœuds. Nous allons avoir besoin d'un nœud racine de type Node2D puis d'un nœud de type TileMap.

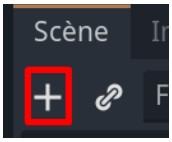
Pour créer un nœud 2D de base, cliquez simplement sur « Scène 2D » dans la zone « Scène » de Godot :



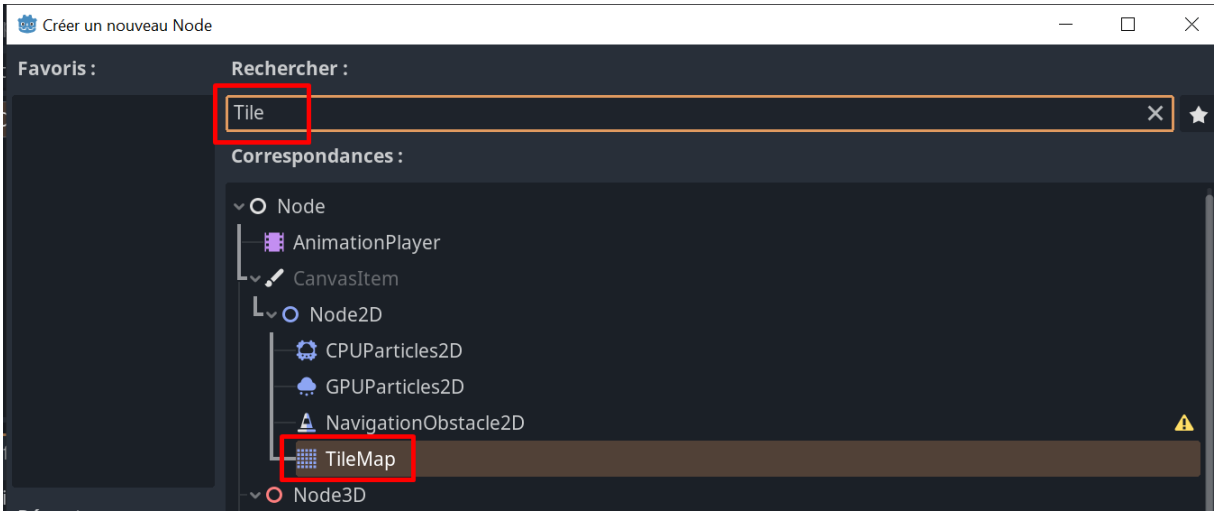
Cela créera le premier nœud de votre scène :



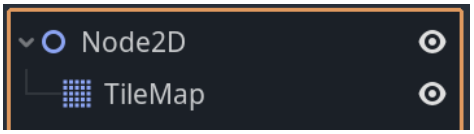
Pour créer un nœud enfant du nœud sélectionné, cliquez sur le « + » :



Puis recherchez le nœud « TileMap » afin de le créer :



Votre arbre de scène devrait ressembler à cela :

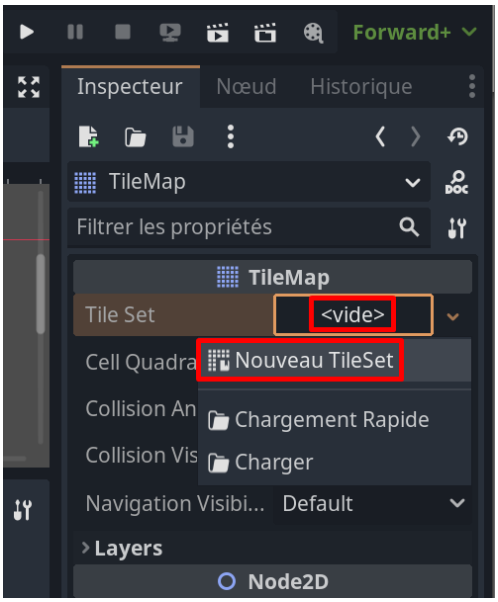


Vous venez de créer vos premiers nœuds sout Godot.

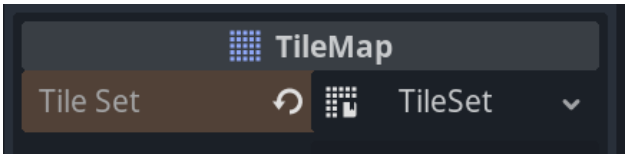
Configurer le TileSet

Une TileMap c'est un niveau de jeu créé à partir de tuiles. Un TileSet c'est un ensemble de tuiles utilisables pour créer notre TileMap. Nous avons créé un nœud de type TileMap qui est actuellement vide. Nous devons créer un TileSet afin de pouvoir créer notre map (niveau de jeu).

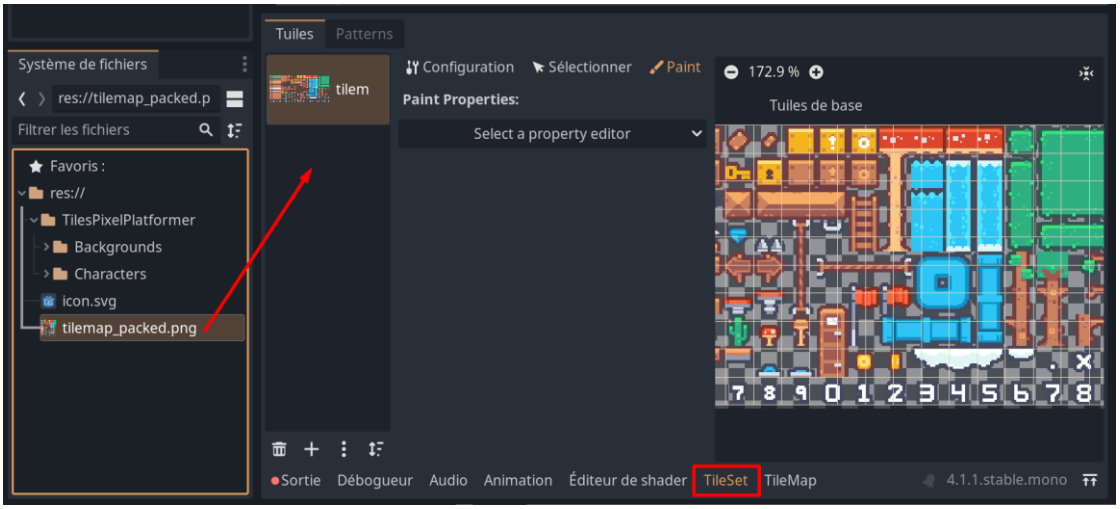
Sélectionnez le nœud TileMap que vous avez créé précédemment afin de faire apparaître ses propriétés dans l'inspecteur. Parmi les propriétés du TileMap, celle qui va nous intéresser c'est la propriété « TileSet ». Cliquez sur le menu déroulant à droite de cette propriété afin de créer un TileSet :



Une fois ceci fait, le TileSet est créé :

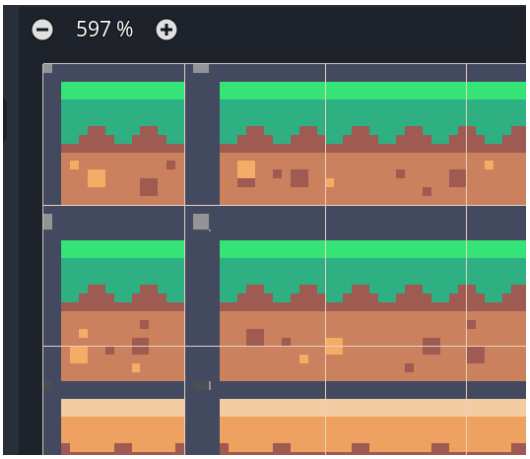


Cliquez dessus et regardez en bas de votre écran, vous verrez qu'une fenêtre apparaît. Cliquez sur l'onglet « TileSet » tout en bas afin d'être en mesure de créer votre Set. Faites alors glisser l'image de votre tileset (l'image téléchargée en ligne) dans la zone sombre de la fenêtre TileSet :

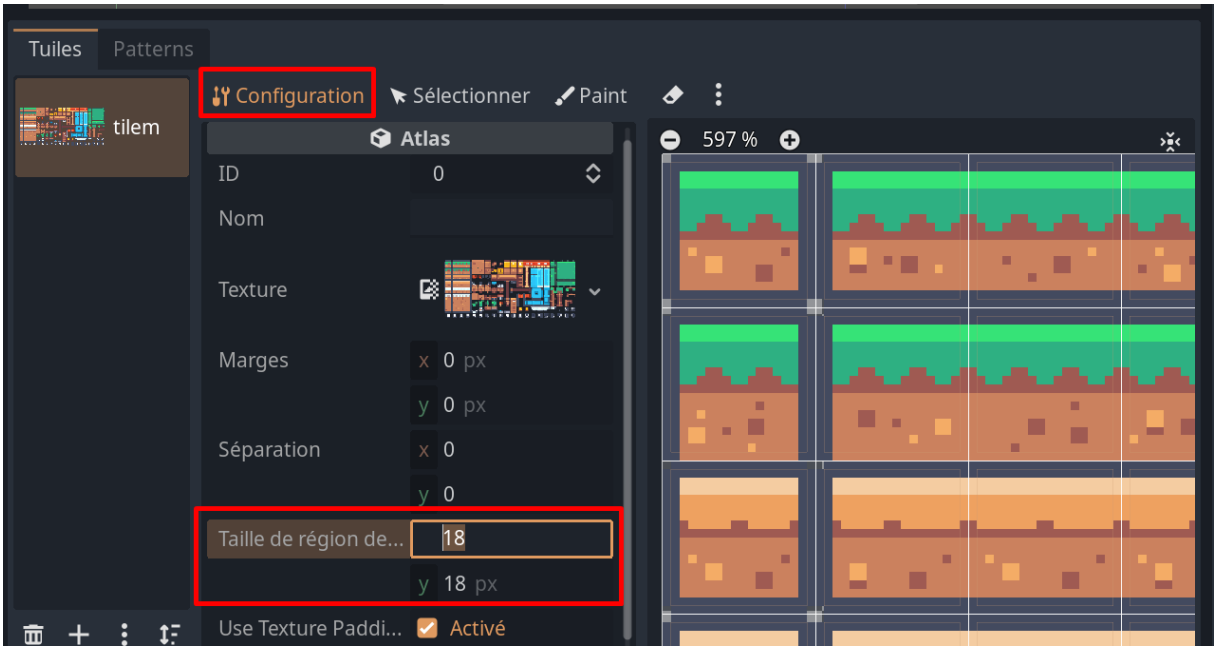


Cliquez sur « Oui » si un message apparaît.

Lorsque votre image est importée, elle apparaît sur la droite et Godot découpe automatiquement votre image en cases de 16 par 16 pixels. En général, les TileSet sont en 8x8, 16x16, 32x32 ou 64x64 pixels par tuiles. Ce sont les tailles les plus courantes. Pour mon exemple, je suis tombé sur un cas très particulier. Mes tuiles sont en 18 par 18. On peut d'ailleurs voir (si je zoom) que la découpe n'est pas bonne :

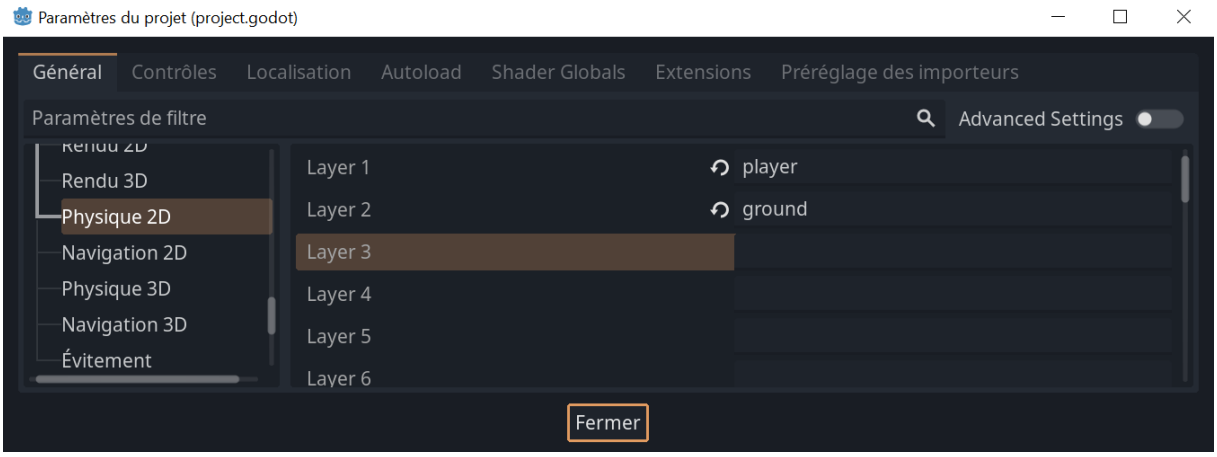


Dans ce cas particulier, je dois aller dans la zone de configuration afin de paramétrer la découpe pour qu'elle soit en 18x18 :

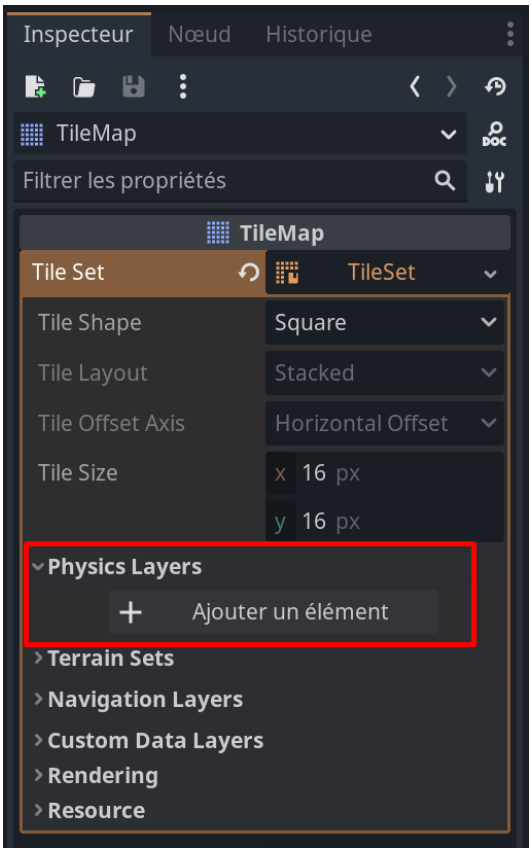


Une fois configuré, la découpe est bien faite. Lorsque vous téléchargez un TileSet en ligne, vous aurez toujours dans la description la taille des tuiles. Vous pourrez facilement configurer vos tuiles dans Godot grâce à cette information.

Nous devons maintenant ajouter de la collision à nos tuiles afin de les rendre solides. Le personnage doit pouvoir marcher sur ces tuiles. Pour cela, commencez par créer des calques de collision en passant par le menu « Projet / Paramètres du projet » puis en vous rendant sur le menu « Physique 2D ». Créez alors deux calques. Un pour le personnage et un pour le sol :



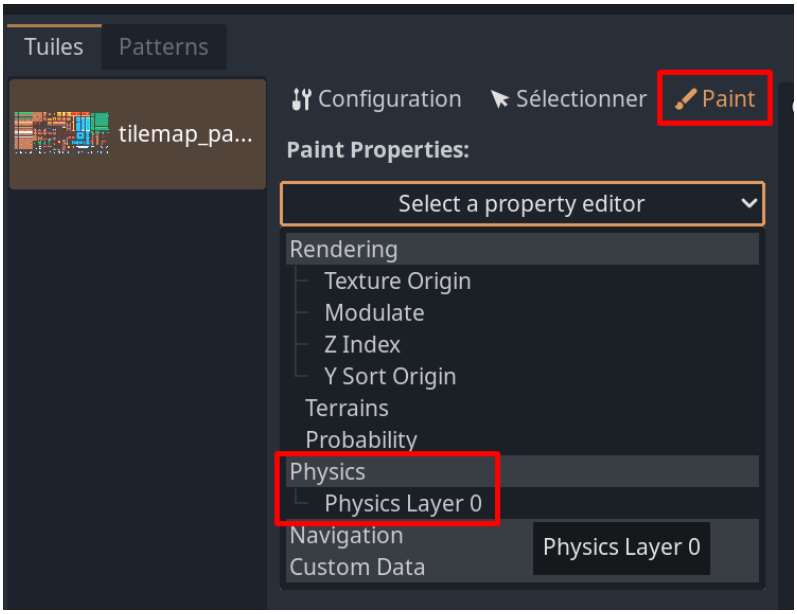
Créez ensuite un calque de collision en passant par l'inspecteur :



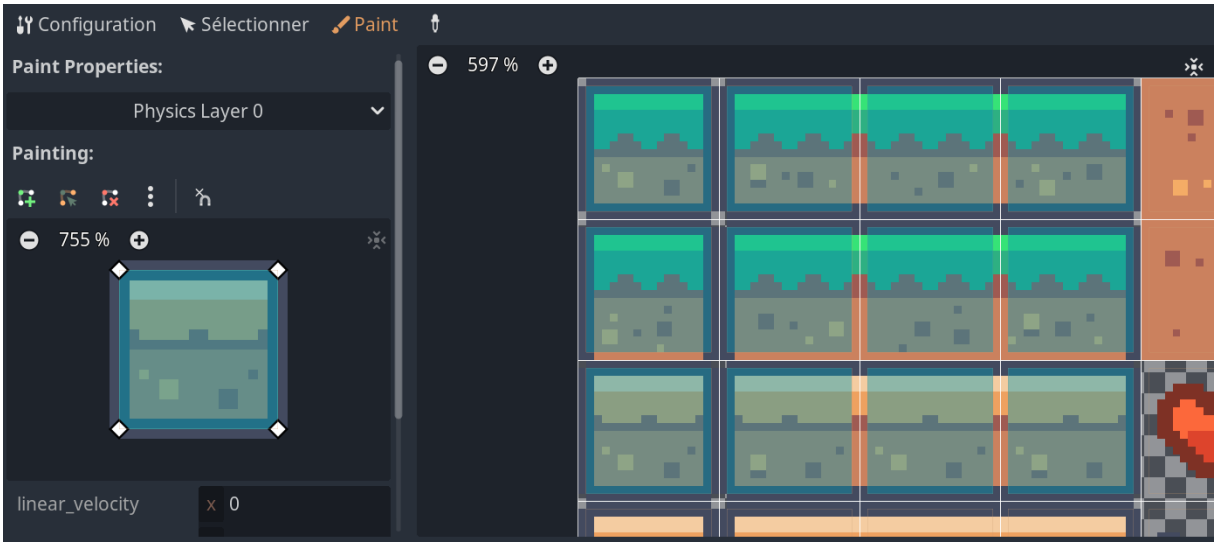
Une fois créé, vous allez pouvoir indiquer que les tuiles peuvent interagir avec le personnage en cochant les chiffres associés à chaque calque :

Ici j'indique que 2 (sol) peut interagir avec 1 (personnage).

Retournez sur la configuration du TileSet, sur l'onglet « Paint » et sélectionnez la propriété physique afin de « peindre » les collisions :



Vous pouvez alors simplement cliquer sur les tuiles que vous voulez rendre « solide » pour ajouter un collider :

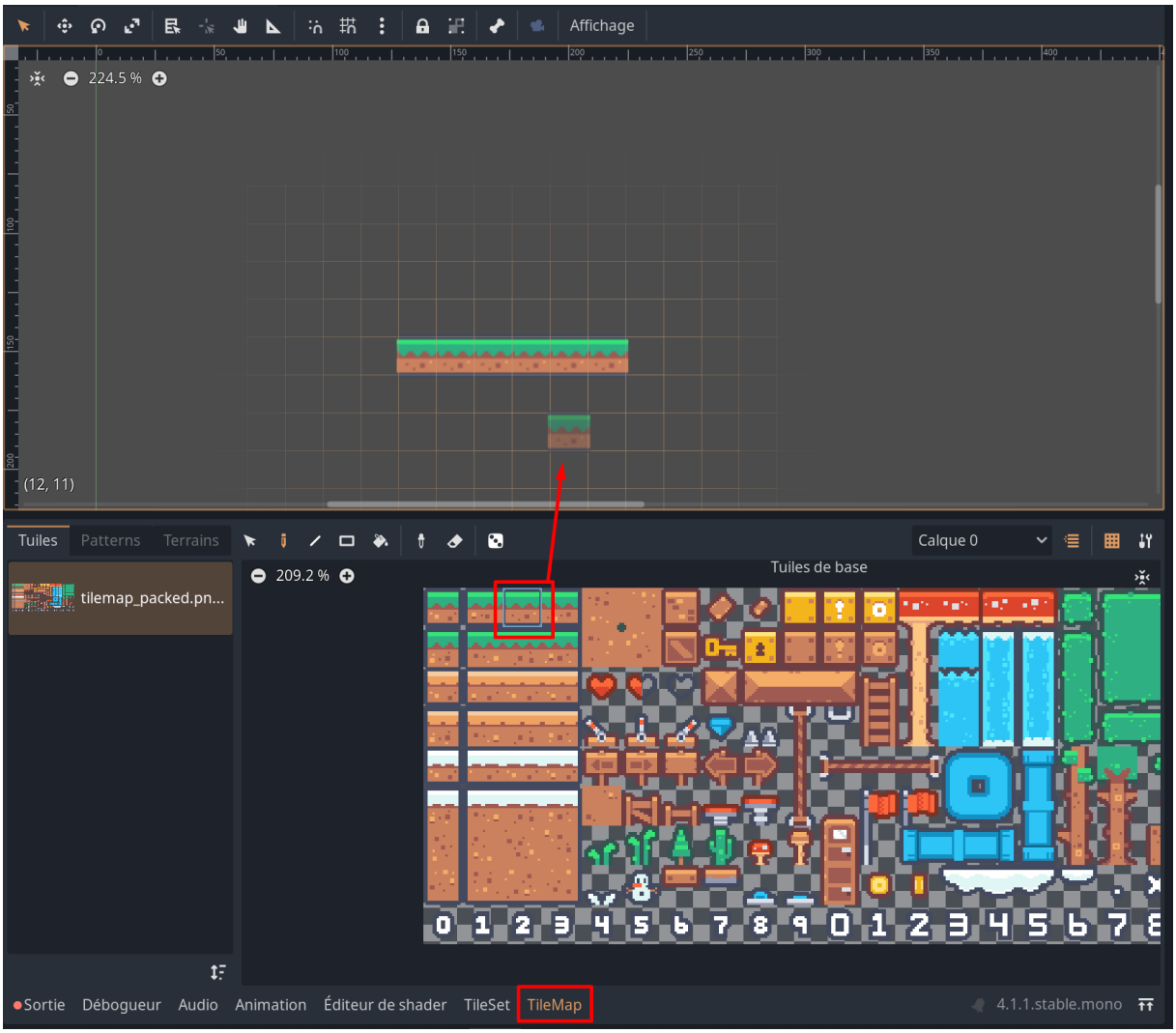


Votre TileSet est prêt.

Dessiner un niveau de jeu

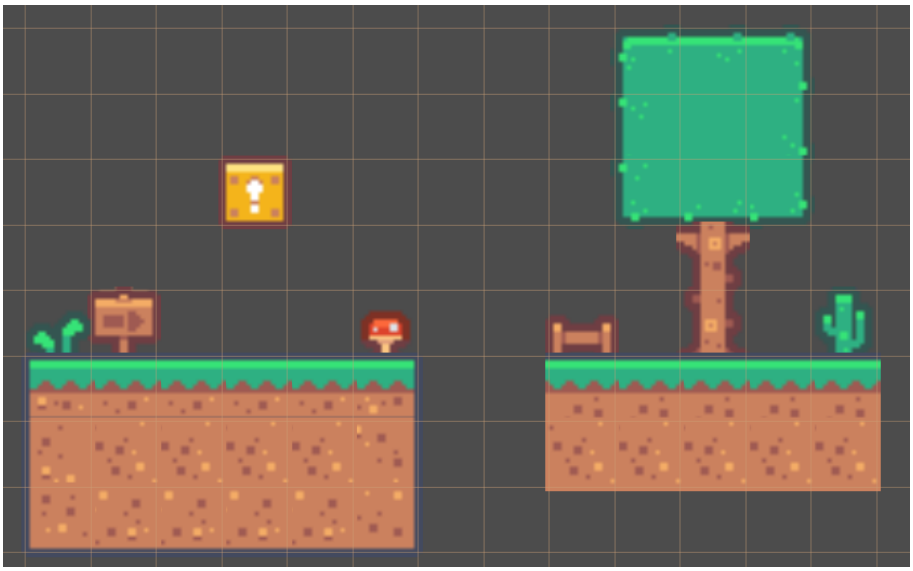
Maintenant que votre TileSet est prêt, vous allez pouvoir peindre vos tuiles sur votre TileMap.

Vous devez sélectionner le nœud TileMap, vous rendre sur l'onglet « TileMap », sélectionner la tuile à peindre puis vous devez cliquer sur la grille pour dessiner la tuile :



La grille n'apparaîtra que si le nœud TileMap est bien sélectionné.

Vous pouvez sélectionner différentes tuiles pour créer votre niveau, vos plateformes etc. :

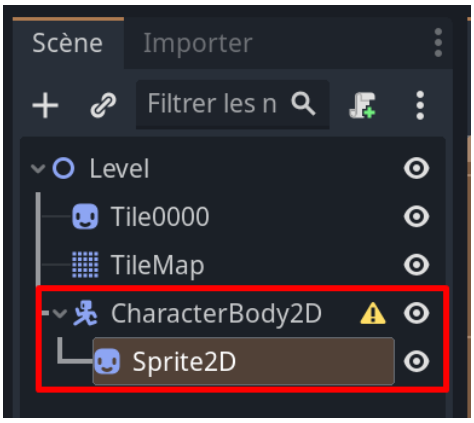


Vous pouvez aussi utiliser des images du packages et les faire glisser/déposer pour ajouter de la décoration comme un ciel ou des nuages.

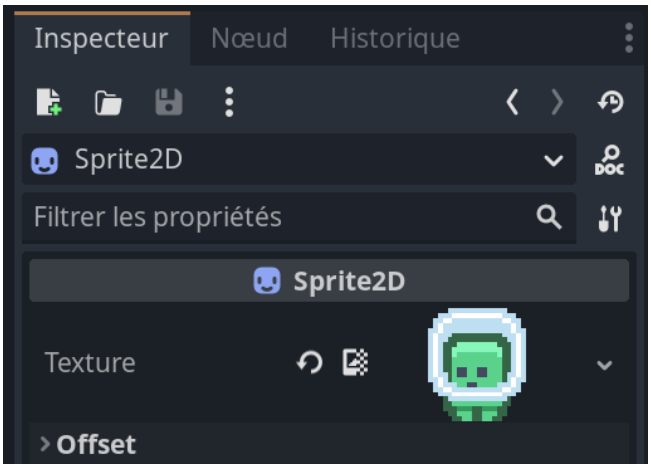
Créer un personnage

Nous allons maintenant créer un personnage afin de pouvoir se déplacer sur le niveau. Je vais simplement survoler le sujet et vous proposer quelque chose de très simple. Nous n'allons ni animer le personnage, ni animer la caméra.

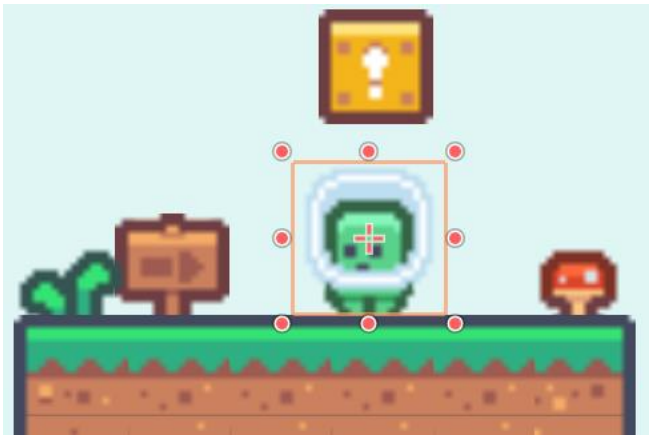
Pour créer un personnage, ajoutez un nœud de type « CharacterBody2D ». Ce nœud permet de créer un personnage capable de se déplacer. Cependant, pour le moment rien n'apparaît à l'écran. Ajoutez un nœud enfant de type Sprite2D afin de pouvoir ajouter un visuel au personnage :



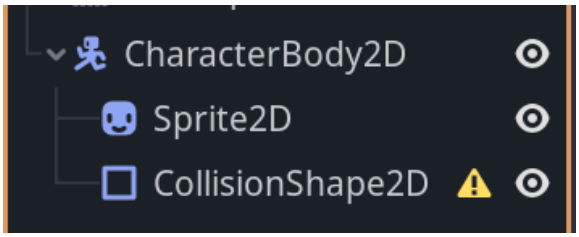
Sélectionnez votre Sprite2D et via l'inspecteur, ajoutez une image pour représenter votre personnage :



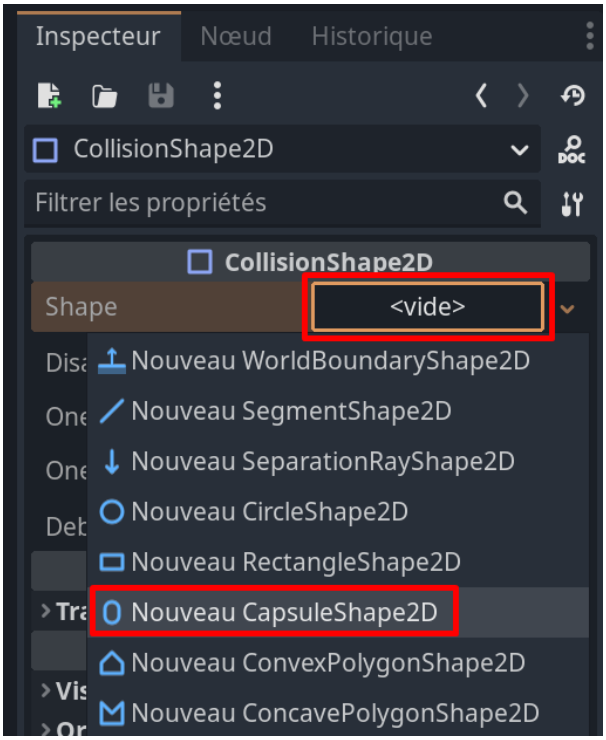
Ce qui nous donne :



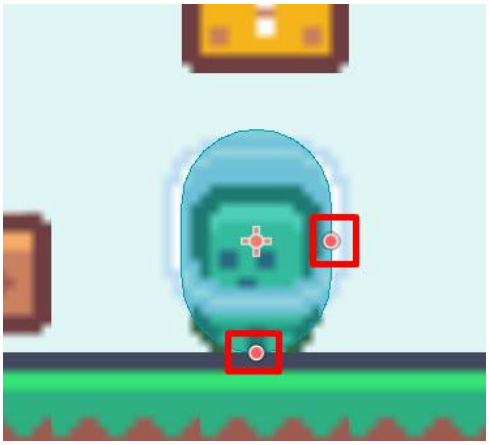
Notre personnage n'est toujours pas prêt. Nous devons lui ajouter un collider. Créez un nœud de type CollisionShape2D en tant qu'enfant du CharacterBody2D :



Sélectionnez ce nœud et via l'inspecteur et créez un Shape de type capsule :

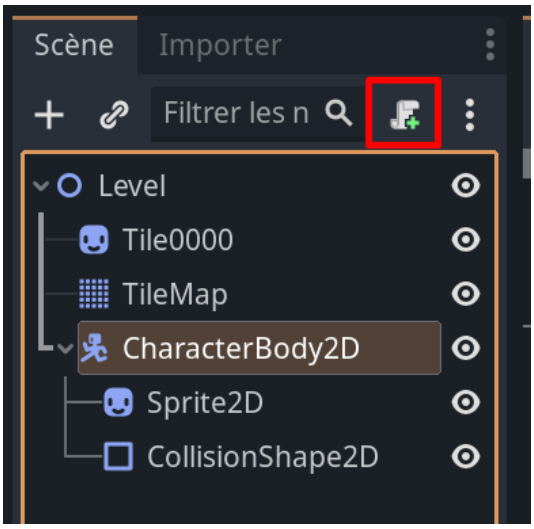


Puis utilisez les poignées afin d'ajuster le collider au personnage :

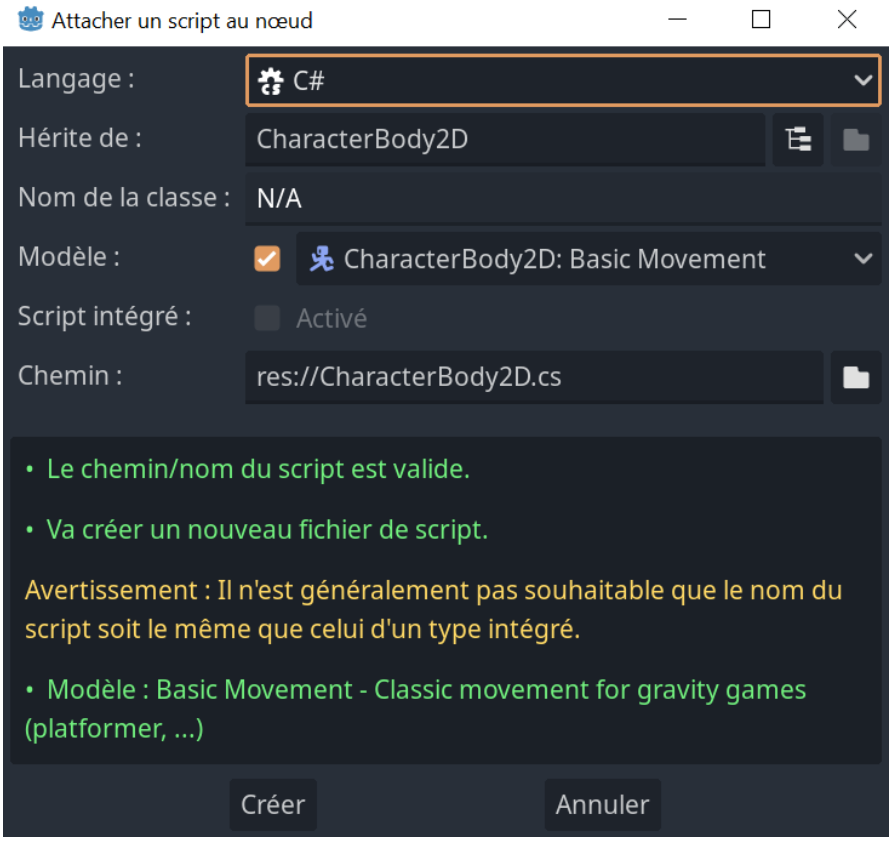


Ajouter un script au personnage

Nous devons maintenant ajouter un script au personnage afin de gérer le mouvement de celui-ci. Sélectionnez le CharacterBody2D et cliquez sur l'icône de création de scripts :

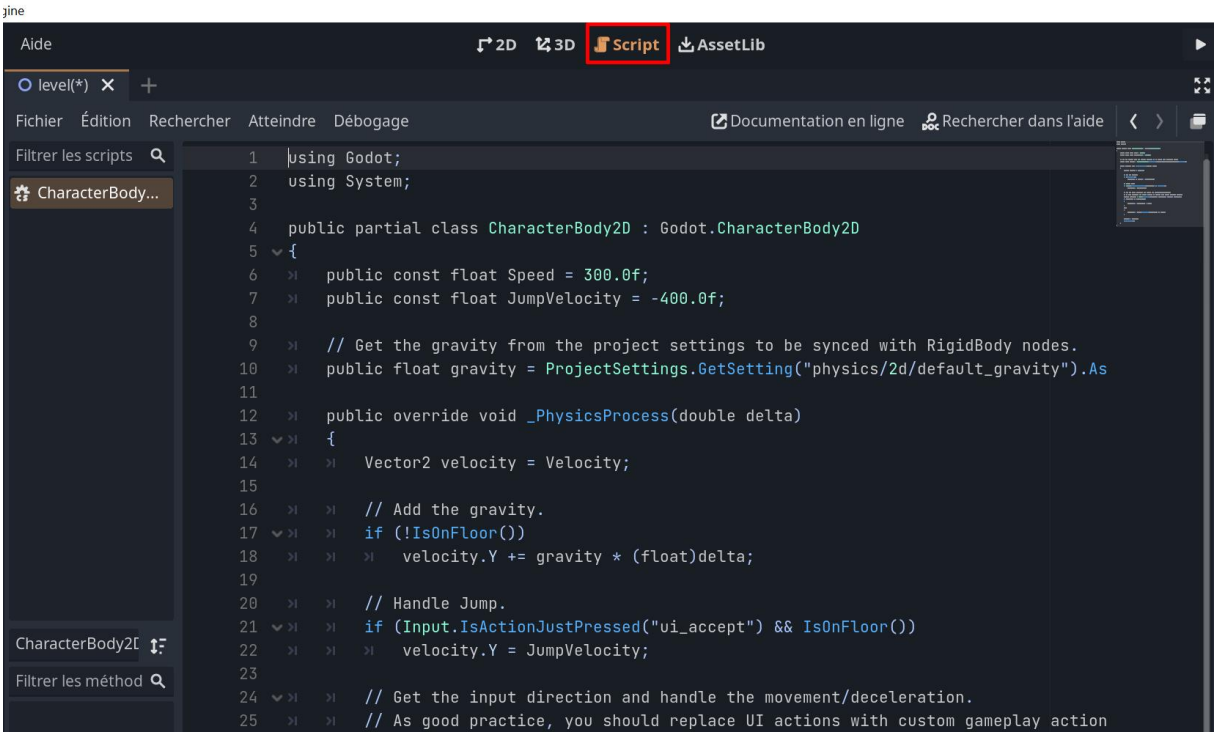


Choisissez le langage C# et activez le modèle afin de choisir un script de type CharacterBody2D :



Cliquez sur créer afin de créer le script et de l'attacher au nœud sélectionné.

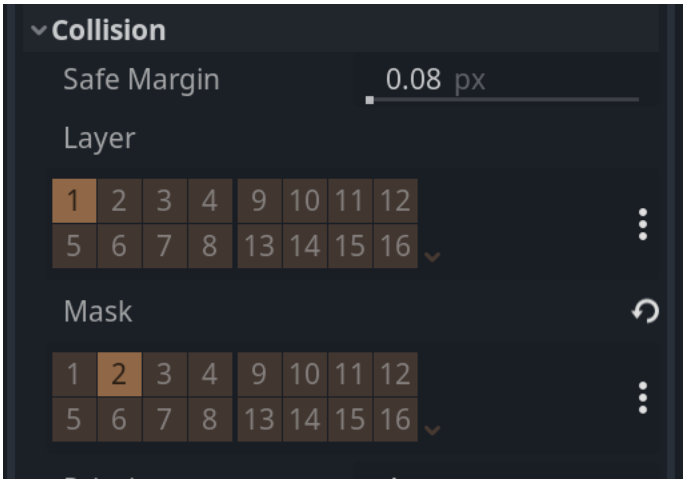
Vous pouvez analyser le script en vous rendant sur l'onglet « Script » :



Ce script gère le mouvement du personnage. Vous pouvez vous déplacer avec les flèches du clavier et sauter avec la barre d'espace. Il s'agit d'un modèle de script spécifique au CharacterBody2D.

Retournez sur l'onglet « 2D » afin de vous rendre sur votre scène.

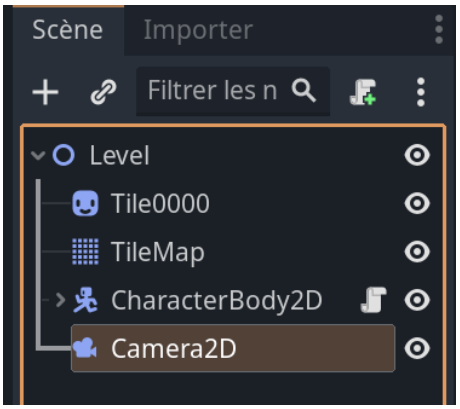
Vous devez impérativement configurer les collisions de votre personnage. Sélectionnez le CharacterBody2D et via l'inspecteur, paramétrez les collisions pour que le calque 1 (player) interagisse avec 2 (ground) :



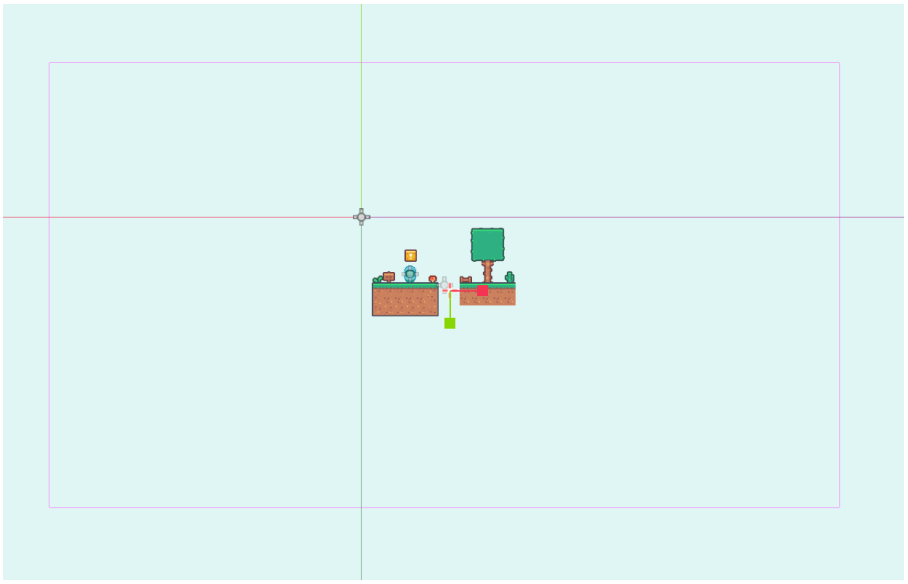
De cette façon les collisions entre le personnage et le sol seront prises en compte.

Ajout d'une caméra

Afin de pouvoir tester votre jeu, vous devez ajouter une caméra. Créez donc un nœud de type Camera2D :

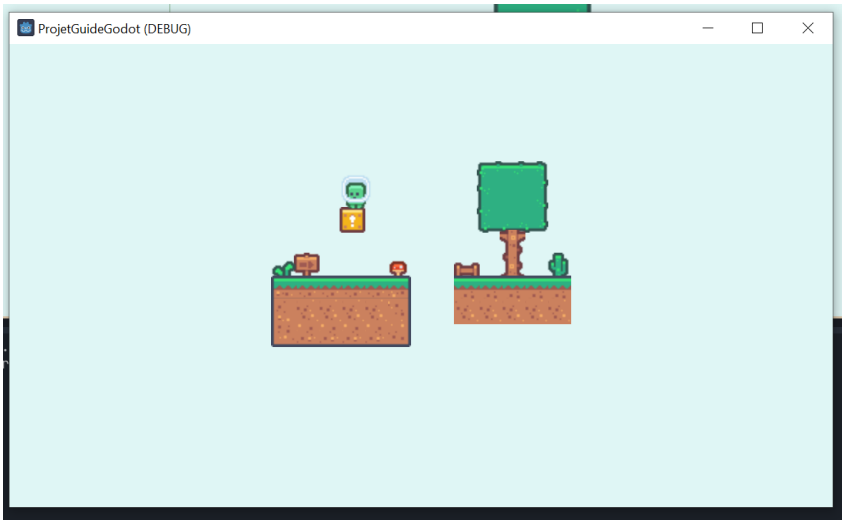


Positionnez la caméra afin de centrer la vue sur votre niveau. Vous pouvez repérer le champ de vision de la caméra grâce au contour mauve :



Tester le jeu

Nous pouvons maintenant tester le niveau. Pour cela, appuyez sur la touche « F6 » afin de lancer la scène actuelle :



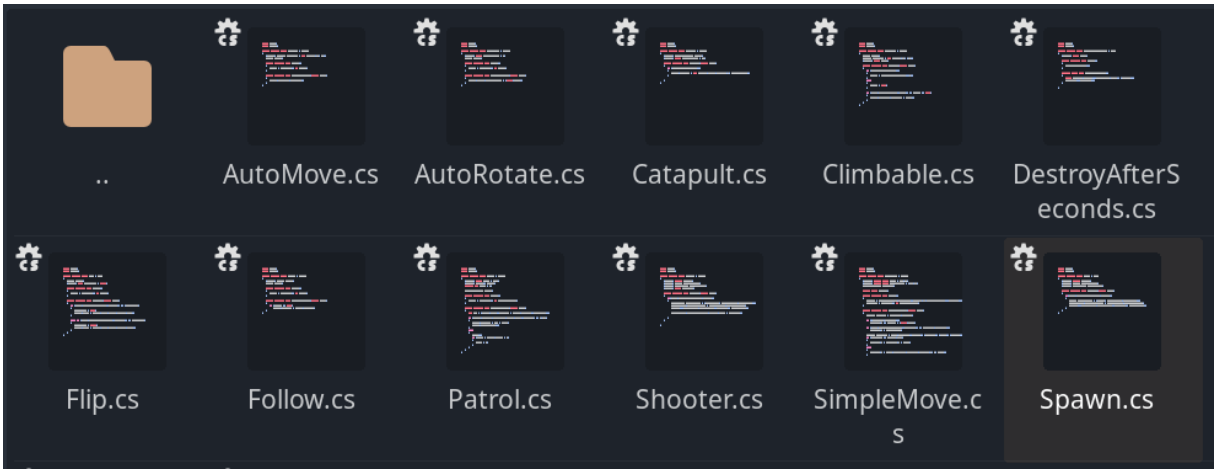
Vous pouvez vous déplacer et sauter.

Conclusion

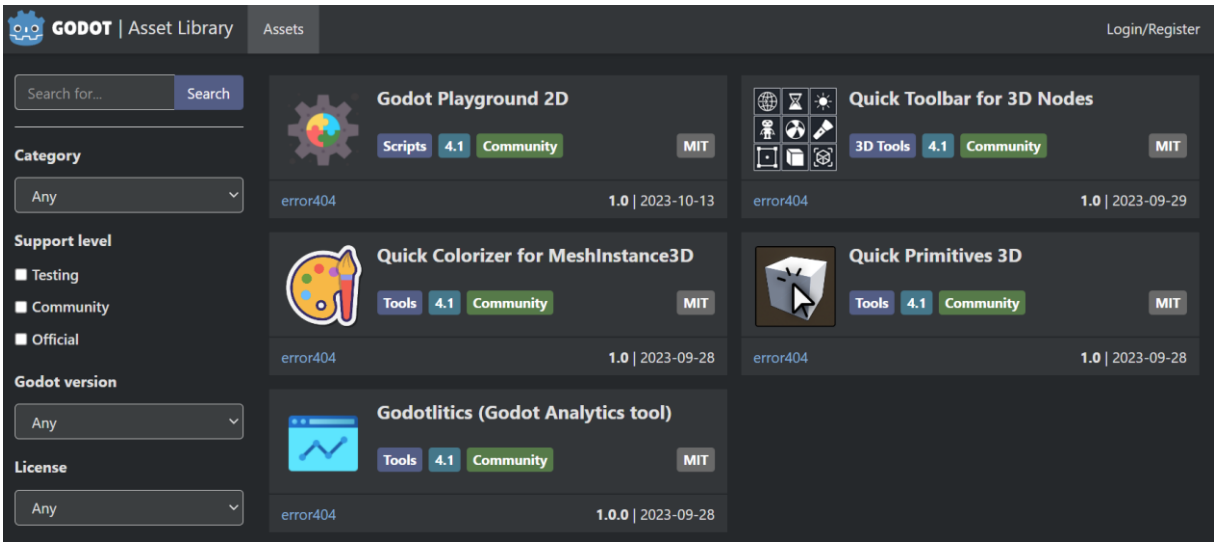
Voilà ce que je voulais vous présenter dans ce guide de démarrage. Vous savez maintenant comment créer une scène, un personnage et utiliser des tuiles. C'est le strict minimum pour pouvoir se lancer dans la création d'un petit jeu 2D. Vous avez encore beaucoup de choses à apprendre comme animer les objets, écrire vos scripts, gérer les collisions, créer une interface utilisateur etc.

Pour aller un peu plus loin, je vous invite à télécharger mon package « Godot Playground » sur l'asset library : https://godotengine.org/asset-library/asset/2243

Il s'agit d'une extension pour Godot qui ajoute de nouveaux nœuds préconfigurés. Des scripts C# sont déjà attachés à ces nœuds et cela vous permettra de rapidement prototyper vos projets. Vous pourrez surtout regarder les scripts du kit afin de vous les approprier et afin d'apprendre à les réécrire vous-même pour prendre en main C# :



Tous mes packages sont disponibles sur ma page créateur : https://godotengine.org/asset-library/asset?user=error404



Pour aller plus loin

J'ai créé une formation vidéo complète de près de 30 heures sur Godot. Vous pouvez la retrouver sur Udemy à cette adresse : https://www.udemy.com/course/godot-engine-creation-de-jeux-3d-avec-le-moteur-libre-et-gdscript/?referralCode=0A8D82AD8A8C775FEA77

J'ai également publié un livre sur Godot 4 que vous pouvez retrouver ici : https://www.d-booker.fr/developpement-de-jeux-video/799-1334-developper-des-jeux-avec-godot-4-et-le-langage-csharp.html#/21-options-consultation_en_ligne

Télécharger le projet

Vous pouvez récupérer le projet à cette adresse : https://anthony-cardinale.fr/_public/links/godot/ProjetGuideGodot.zip