Curso Técnico

de Programação de Jogos Digitais

Aula 27

índice

[1. Ferramenta Unity 3](#_Toc169681803)

[2. Física no cenário 3](#_Toc169681804)

[3. Referencias 9](#_Toc169681805)

# Ferramenta Unity

No Unity, a montagem de cenários usando layers é uma técnica comum para criar ambientes 2D complexos e visualmente interessantes. Aqui está um resumo de como funciona:

Layers (Camadas): No Unity, você pode organizar os elementos do seu jogo em diferentes camadas.

Isso permite controlar a ordem em que os objetos são renderizados na cena. Cada objeto pode ser atribuído a uma ou mais camadas, e você pode definir a ordem de renderização das camadas no editor.

Background (Fundo): Geralmente, o fundo é colocado em uma camada inferior. Isso pode incluir elementos como céu,

montanhas distantes ou outros elementos que devem aparecer atrás dos objetos principais da cena.

Você pode usar uma imagem de fundo fixa ou criar um fundo parallax para adicionar uma sensação de profundidade ao seu jogo.

Midground (Plano de Fundo): A camada do plano de fundo geralmente contém elementos que estão atrás dos personagens e

objetos principais, mas à frente do fundo. Isso pode incluir árvores, arbustos ou outras características do ambiente que

adicionam detalhes à cena.

Foreground (Primeiro Plano): Esta é a camada mais próxima do jogador e geralmente contém elementos que estão à frente dos

personagens e objetos principais. Isso pode incluir obstáculos, decorações próximas, ou qualquer coisa que ajude a adicionar

profundidade e imersão à cena.

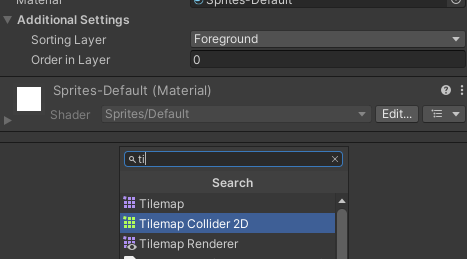


Podemos adicionar efeitos de movimento ao background de um cenário.

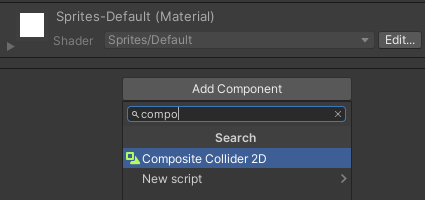
Uma técnica comum em jogos 2D é o efeito de parallax scrolling, onde as camadas se movem em velocidades diferentes para simular a profundidade e criar uma sensação de movimento. Isso pode ser alcançado movendo as camadas de fundo a uma velocidade mais lenta do que as camadas mais próximas do jogador.

# Física no cenário

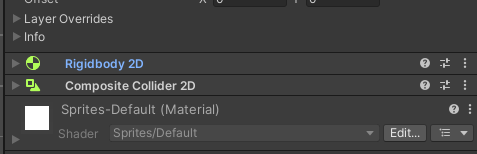
Vamos então trabalhar com cada uma das layers para nosso cenário. O foreground será uma layer que deverá ter alguns colisores.Adicione o Tilemap collider 2d



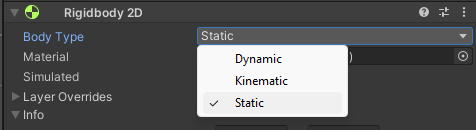
Vamos adicionar o composite collider2d



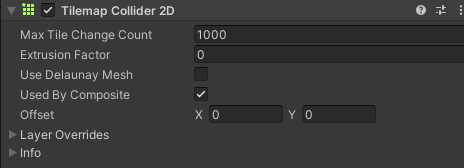
Ele irá criar automaticamente um componente rigidbody2d então por ser um cenário e não personagem devemos fazer algumas modificações



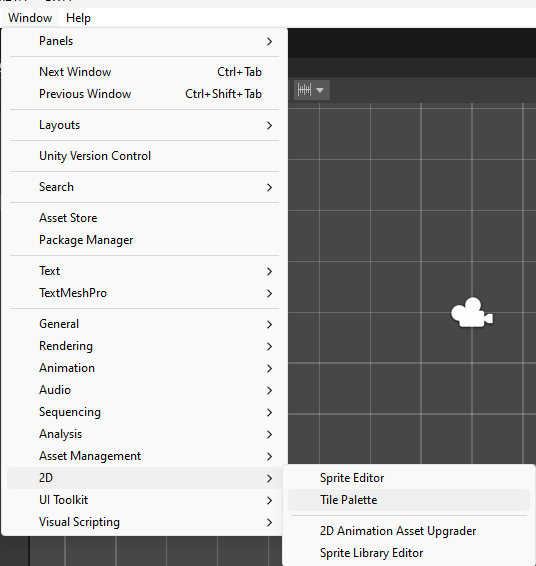
Modifique de dinamic para static no rigidbody2d



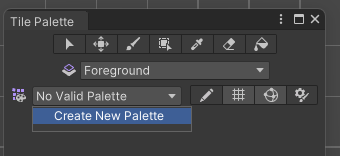
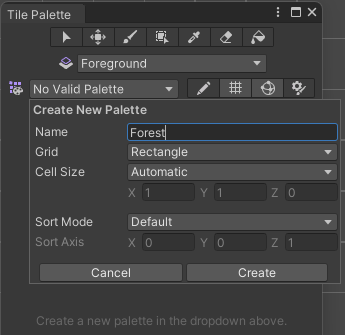
E no Tilemap Collider2d ative o used by composite. Isso irá gerar a colisão por composição.



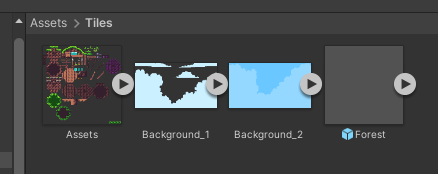
Abra Windows/2D/Tile Palletes



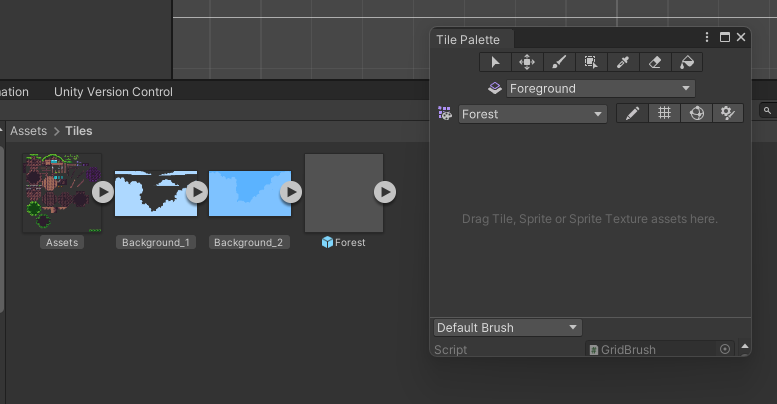
Vamos criar um pallete para essa layer

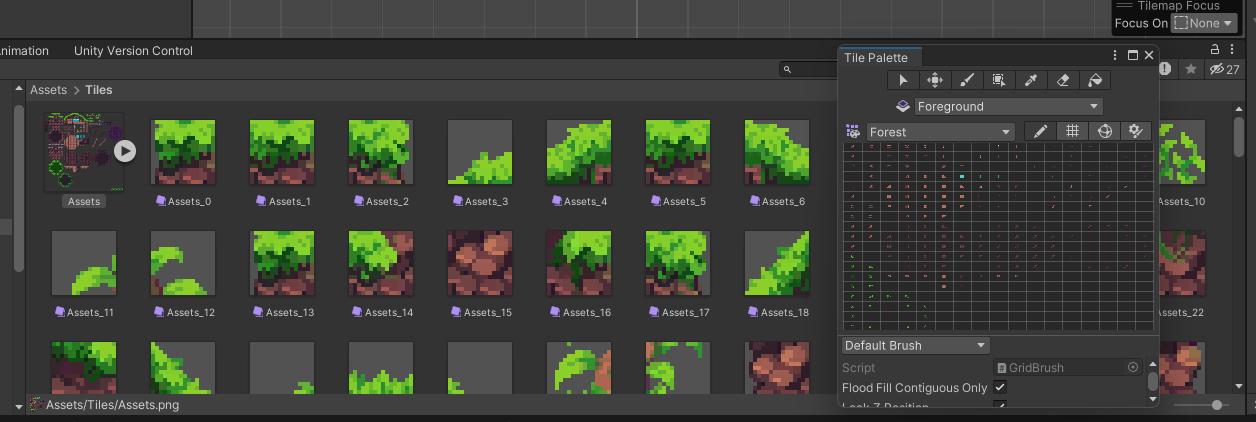
Clique em create e salve na pasta tiles.



Arraste as imagens do tile para o tile pallete



Salve na mesma pasta

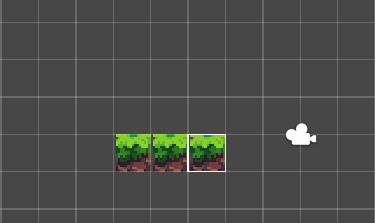




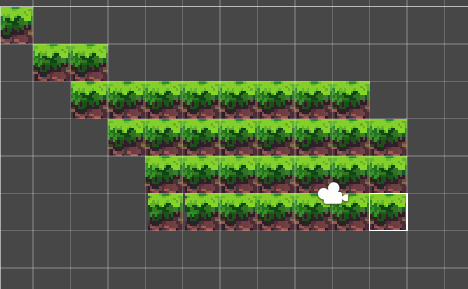
Temos então nesta barra de ferramentas:

* Temo a seleção de um item ou área;
* Seleção de vários blocos ao mesmo tempo;

Temos o pincel para selecionar um item, onde podemos colocar bloco a bloco



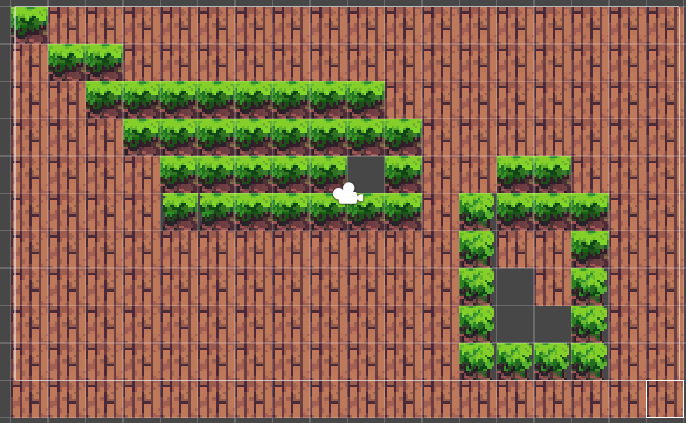
Temos o blush que servirá para preencher grandes áreas.



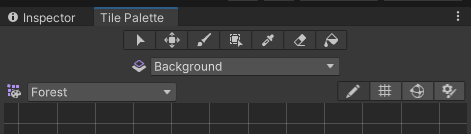
Temos o conta gotas que preenche área fechadas;

Temos a borracha para apagar;

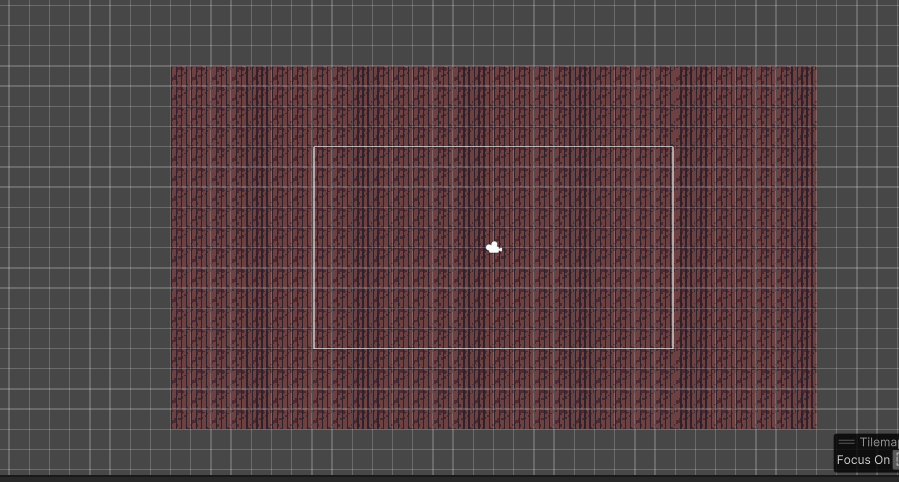
Temos o balde para preencher toda cena;



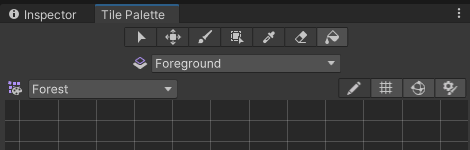
Então podemos criar nosso cenário. Escolha o background e faça o preenchimento.

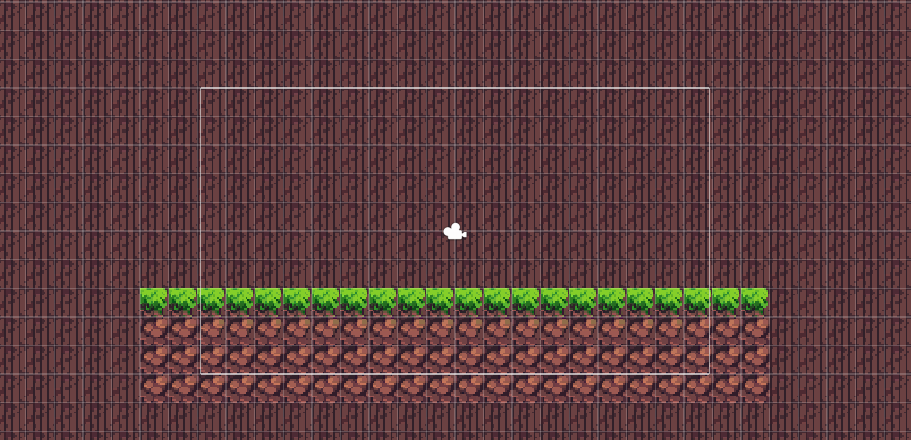


Este será o fundo do cenário então escolha a textura que lhe agrade.

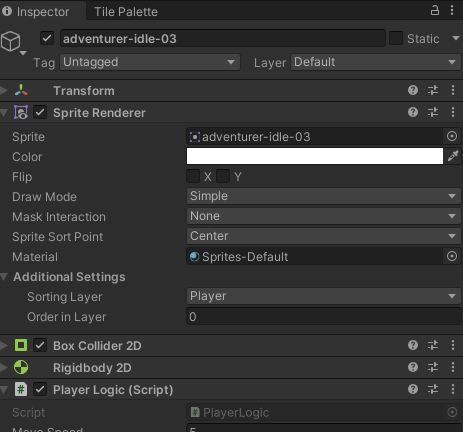


Deixe maior que a área da câmera, isso nos ajudara quando estivermos movendo no cenário. Iremos então criar o piso para o player na foreground.

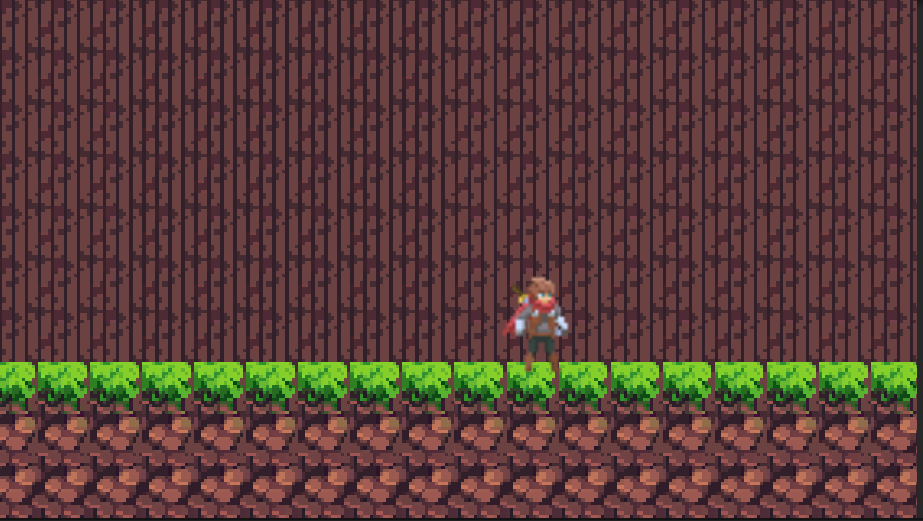




Selecione o player e siga para sprite renderer. Mude sorting layer para player



Nosso personagem agora tem um cenário



Os códigos usados em nossas aulas estão disponíveis em nosso repositório no GitHub.



https://github.com/rildexter/pjd/tree/main/djm

# Referencias

1. Unity Technologies. (2020). **Unity User Manual.** Unity Technologies.
2. [C#, 2020] **Visual C# Developer Center**, Microsoft Docs.

Microsoft Corporation. (2020).

1. Geig, M. (2018). **Unity 2018 Game Development in 24 Hours**, Sams Teach Yourself. Sams Publishing.
2. Hocking, J. (2015). **Unity in Action: Multiplatform Game Development in C#.** Manning Publications.