Curso Técnico

de Programação de Jogos Digitais

Aula 29

índice

[1. Controle de Camadas de um jogo 3](#_Toc169685822)

[2. Layers no Desenvolvimento de Jogos 4](#_Toc169685823)

[3. Utilizando o Sprite Fusion 4](#_Toc169685824)

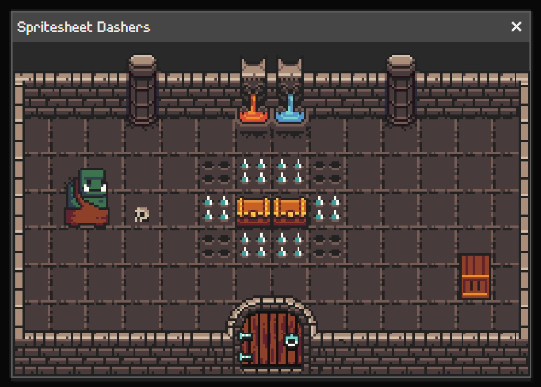
[4. Referencias 8](#_Toc169685825)

# Controle de Camadas de um jogo

As layers no Sprite Fusion são uma funcionalidade essencial para a criação de mapas 2D detalhados e organizados. As layers permitem que os desenvolvedores e designers de jogos construam seus cenários em diferentes níveis, facilitando a adição de profundidade e complexidade ao ambiente do jogo.

Principais Características das Layers no Sprite Fusion

1. Organização Estruturada: As layers ajudam a manter os diferentes elementos do mapa organizados. Por exemplo, é possível ter uma layer para o fundo, outra para os elementos de decoração e uma terceira para objetos interativos. Isso facilita a edição e manutenção do mapa.



2. Facilidade de Edição: Cada layer pode ser editada independentemente das outras. Isso significa que os designers podem modificar elementos específicos sem afetar o resto do mapa, proporcionando um fluxo de trabalho mais eficiente e menos propenso a erros.

3. Controle de Z-Order: As layers permitem controlar a ordem em que os elementos são desenhados na tela. Elementos em layers superiores aparecem acima dos elementos em layers inferiores, permitindo a criação de efeitos de sobreposição e profundidade.

4. Visibilidade Personalizável: As layers podem ser ativadas ou desativadas conforme necessário durante o processo de design. Isso facilita a visualização do mapa sem distrações, permitindo focar em uma camada específica de cada vez.

5. Exportação Integrada: Ao exportar os mapas criados no Sprite Fusion para motores de jogo como Unity ou Godot, as layers são mantidas, preservando a organização e estrutura do mapa original. Isso simplifica a integração e reduz o tempo de ajuste necessário no motor de jogo.

# Layers no Desenvolvimento de Jogos

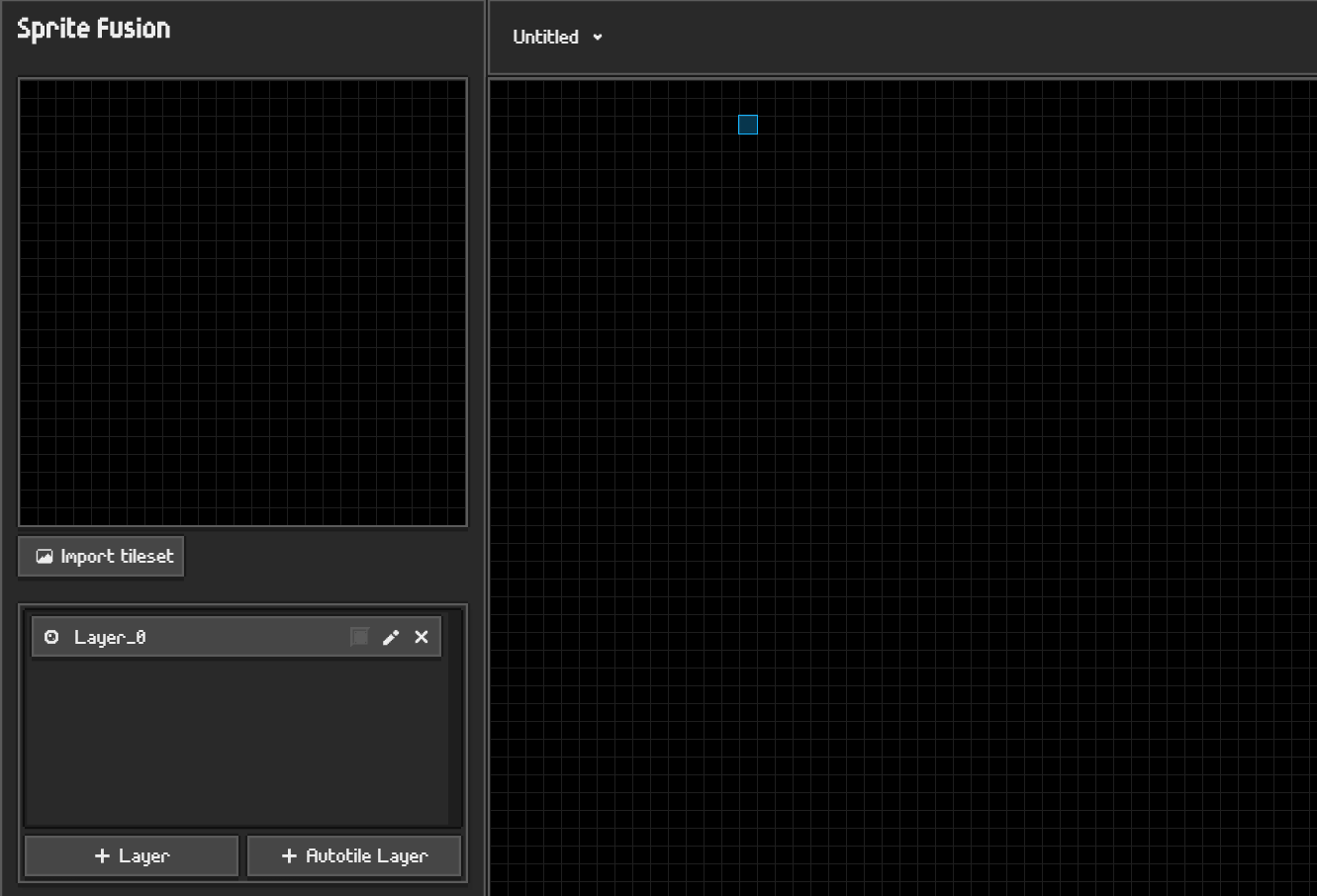
O uso de layers é particularmente benéfico para jogos que exigem ambientes complexos e detalhados, como jogos de plataforma, RPGs e jogos de aventura. A capacidade de gerenciar diferentes aspectos do cenário de forma independente melhora a produtividade e a qualidade do design, permitindo que os desenvolvedores criem mundos de jogo mais imersivos e visualmente atraentes.



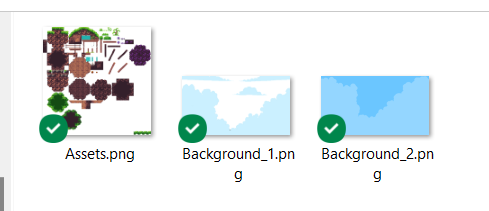
Com as layers, o Sprite Fusion facilita a criação de mapas ricos em detalhes e organizados de maneira lógica e eficiente.

# Utilizando o Sprite Fusion

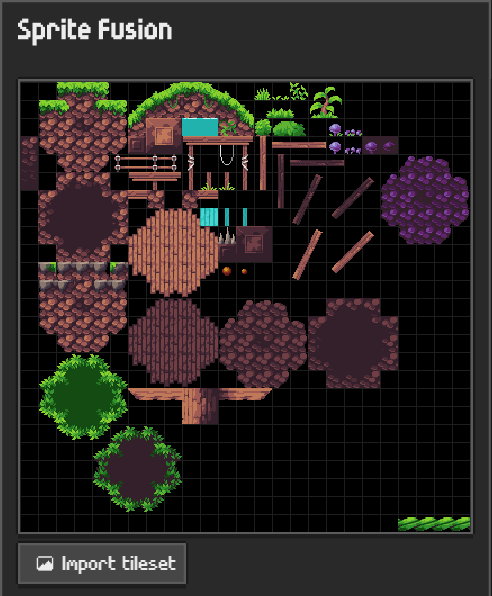
Vamos gerar um novo projeto no sprite fusion



Vamos então utilizar o tiles usados em nossa aula anterior no Unity



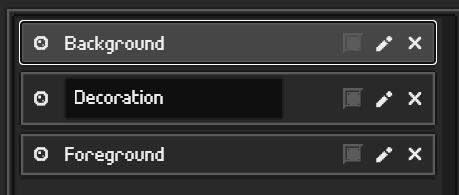
Clique em import tileset e escolha o nosso material.



Agora podemos gerar as nossas layers do projeto. Teremos a Background que manterá todo fundo da cena de nosso game.

O Foreground manterá material que fica entre o player e nos, algo como uma arvore que o personagem passará por detrás desta.

Por fim temos a decoration que deverá manter todos objetos de decoração.

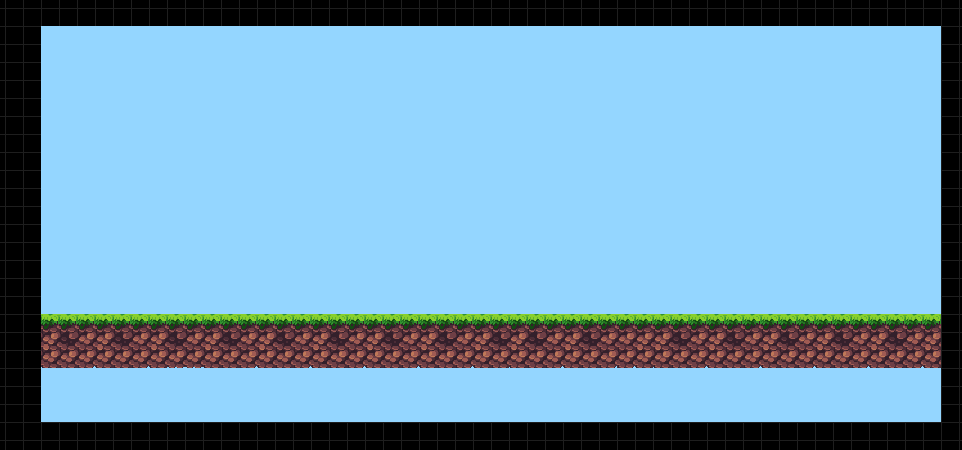


Para projetos com mais detalhes é aconselhável pensar em toda divisão de cenário, como vegetação, céu , águas , pedras, npcs entre outros. A criar uma divisão com muitos detalhes podemos ter um controle sobre qualquer atualização em nosso material.

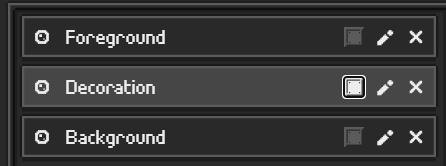
Selecione a Background e vamos iniciar nossa edição.



Crie o solo na camada decoration



Não se esqueça de ativa a colisão nesta camada clicando no icone quadrado ao lado do lápis de edição.



Quando adicionamos mais item na decoration tudo terá efeito de física, ou seja, nosso player poderá tocar nestes itens

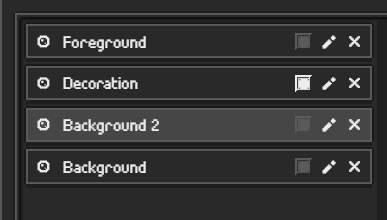


Muito cuidado para não criar barreiras intransponíveis.



Em uma situação onde quero que o player passe em frente de arvores e outras cubram a imagem do personagem, devemos fazer isso através de sobreposição de layers





A arvore 1 e 3 estão no foreground então ficam por cima da imagem do player.

A arvore 2 está na layer background2 logo ela ficara atras da imagem do player.

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.