

Criando um jogo básico em Godot com C#

Prof. Thiago Felski Pereira, MSc.

Visão Geral

- Porque C# se a linguagem principal é o GDScript?
 - Porque o C# é utilizado em todo lugar, enquanto o GDScript é utilizado apenas no Godot.
 - Você pode utilizar todo “arsenal” do .Net para seu jogo, como por exemplo utilizar bibliotecas de aprendizado de máquina para criar inimigos que aprendem com seus próprios erros.
 - Utilizada em outros ambientes de desenvolvimento de jogos populares: Unity.

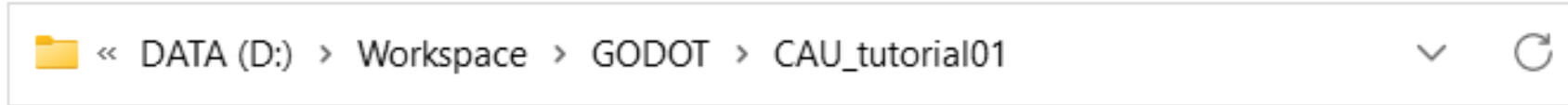


[2]

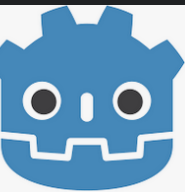


Preparando o ambiente

- Crie uma pasta para seu jogo.



- Eu chamei a minha de **CAU_tutorial01**, mas vocês podem chamar como quiserem.



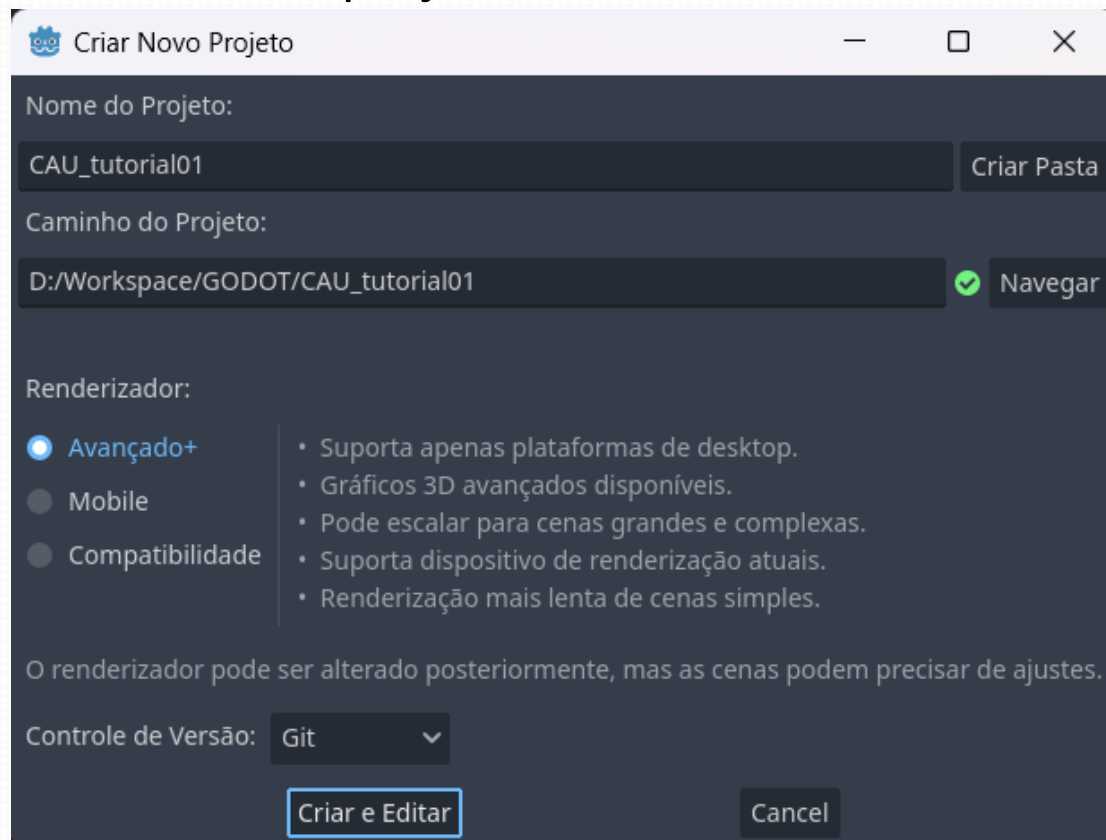
[3]



UNIVALI

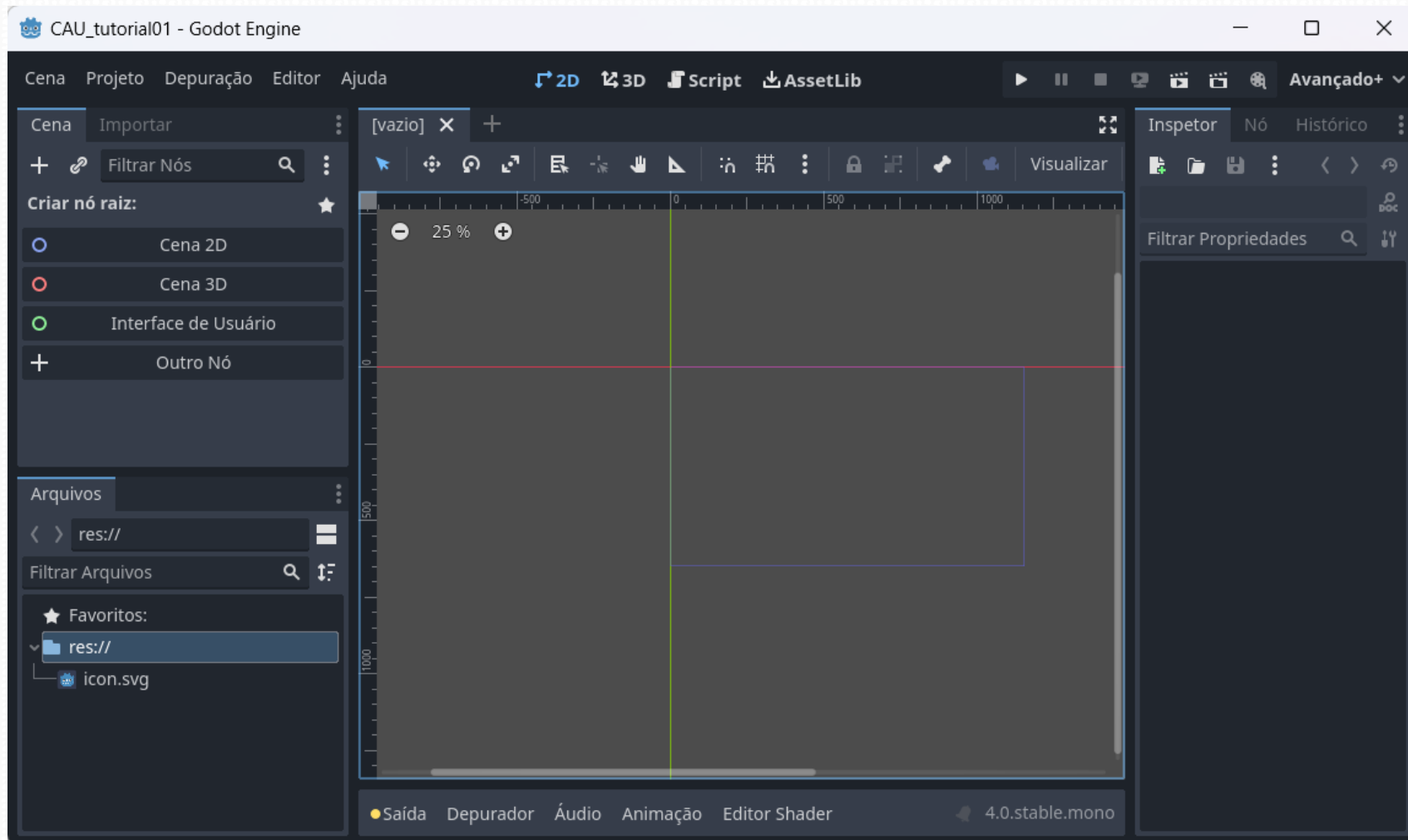
Preparando o ambiente

- Abra o aplicativo do Godot
- Crie um novo projeto, escolhendo um nome e a pasta que você criou para ele



Preparando o ambiente

- Configure o ambiente para 2D, pois assim será nosso primeiro projeto

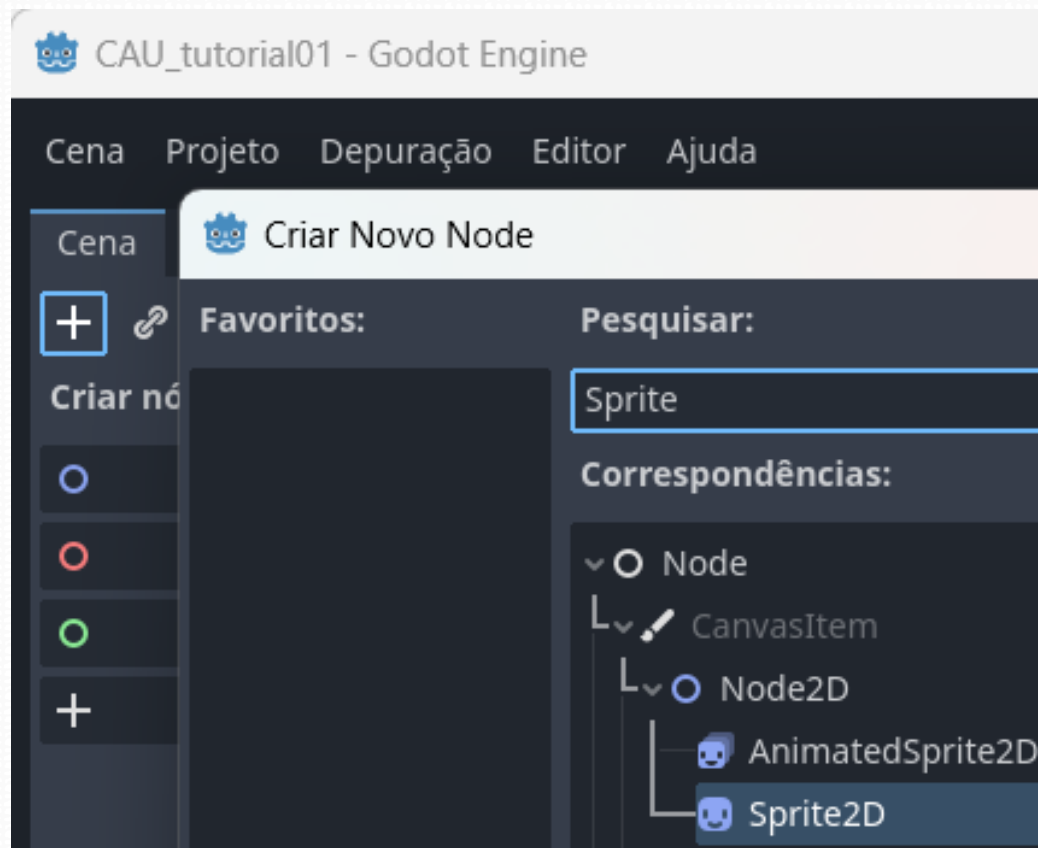


5



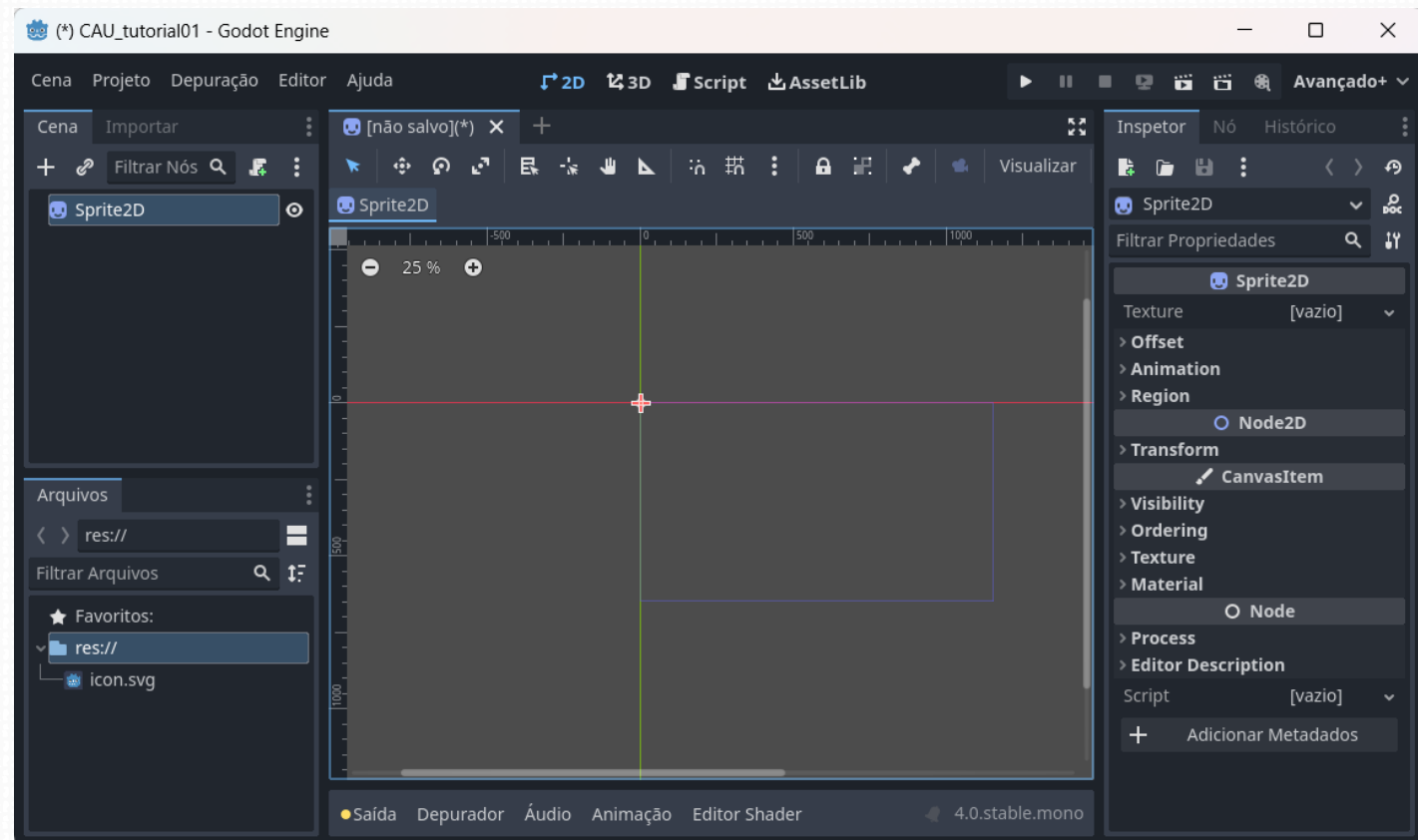
Desenvolvendo o jogo

- O Godot é composto de
 - **Nodos**/Nodes: cada objeto individual dentro do nosso jogo
 - **Cenas**/Scenes: um agrupamento de Nodos.
- Crie, seu primeiro Nodo
 - + -> Sprite2D



Desenvolvendo o jogo

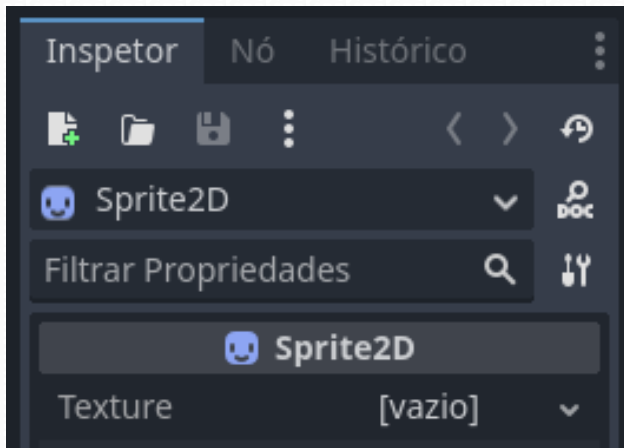
- É possível observar os objetos criados em 2 locais
 - Na Hierarquia
 - Na Cena (não aparece nada lá ainda, pois ainda não associamos nenhuma imagem a nosso Sprite2D)
- Note que na direita temos o **Inspetor**, que nos permite visualizar as características dos objetos selecionados



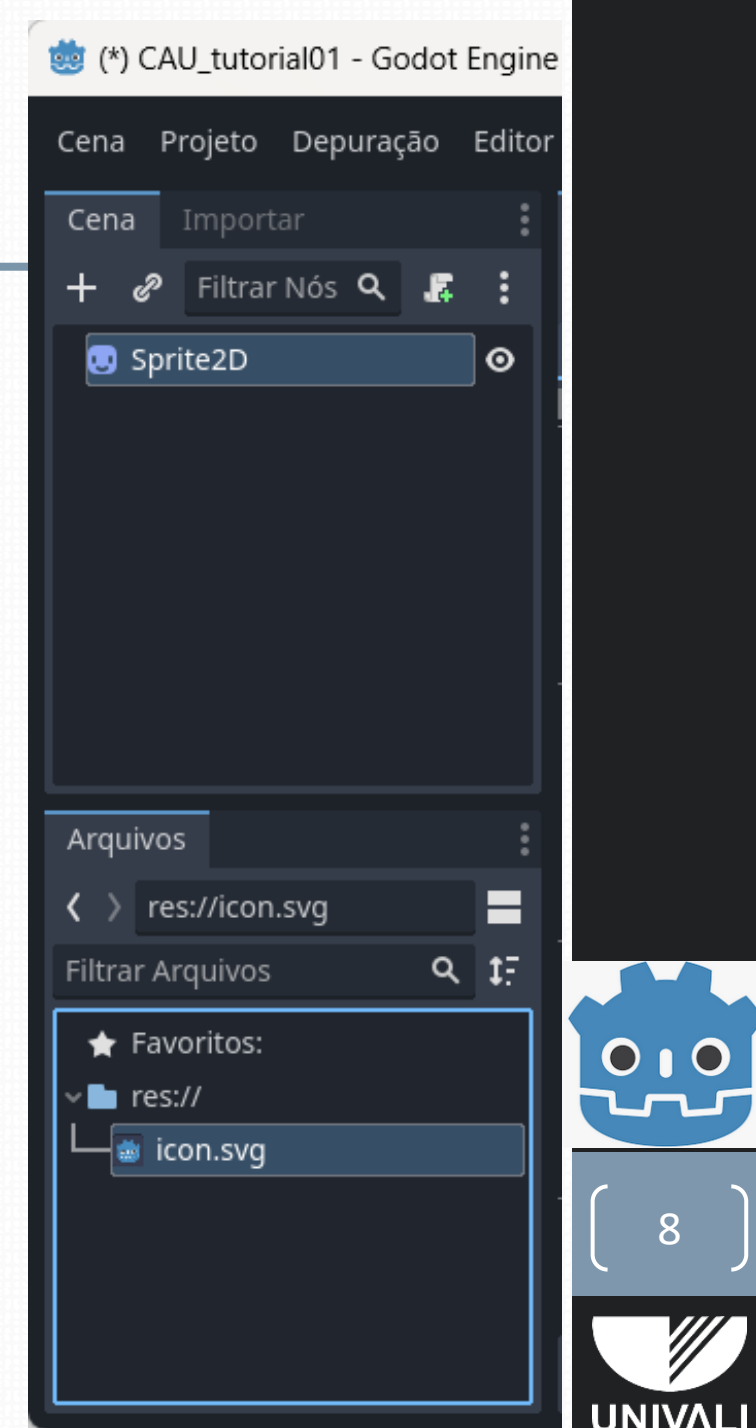
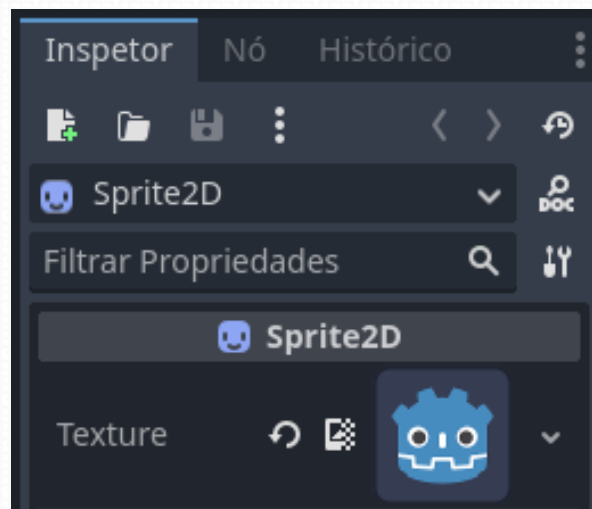
Desenvolvendo o jogo

- Se você é bom observador deve ter notado um ícone na pasta raiz do projeto icon.svg
 - Arraste ele para a textura do Sprite2D

- Antes



- Depois



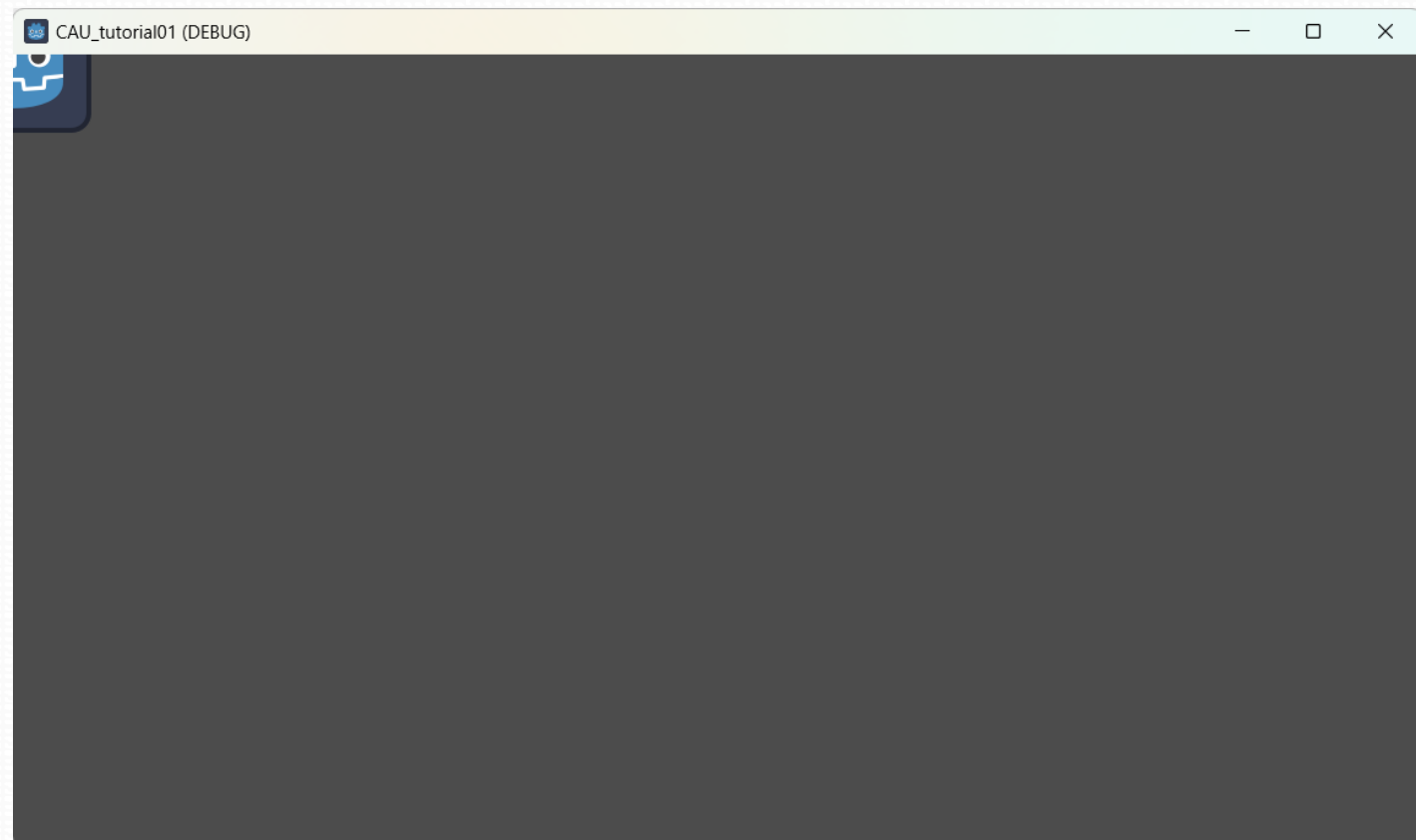
Desenvolvendo o jogo

- Salve seu Nodo: **CTRL+S**

Arquivo: `sprite_2d.tscn`

- Note que mesmo tendo um único Nodo ele salva esse nodo como uma Cena.

- Execute a cena atual salva: **F6**



Desenvolvendo o jogo

- Nosso personagem ficou fora da área visível.
- Podemos corrigir isso manualmente.
 - Clicando e arrastando mais para o centro.
- Podemos corrigir isso nos atributos do Inspetor.
 - Inspetor -> Node2D -> Transform -> Position
 - E ajustar ele no eixo (X,Y) do nosso plano cartesiano.
 - Esse é o atributo que iremos alterar quando formos movimentar o personagem no jogo.
- Faça alguns testes e observe esses atributos com cuidado.
 - Pode ir executando a Cena para observar essas mudanças lá também.

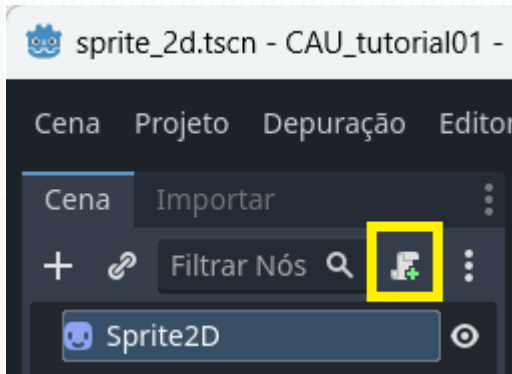


(10)

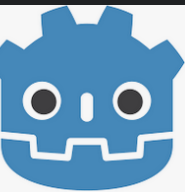


Programando o jogo

- Programando a movimentação do personagem.
- Crie um **Script** para o nodo, ou seja, um programa.



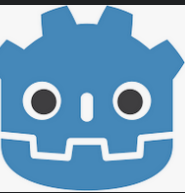
- Troque a linguagem para C# e deixe o resto como está.



Programando o jogo

- Estrutura normal de um Script C# para um Nodo

```
1  using Godot;
2  using System;
3
4  public partial class sprite_2d : Sprite2D
5  {
6
7      // Chamado quando o Nodo entra na Cena pela primeira vez.
8      public override void _Ready()
9      {
10     }
11
12     // Chamado a cada frame. 'delta' é o tempo aproximado de um frame
13     public override void _Process(double delta)
14     {
15     }
16 }
17 }
```

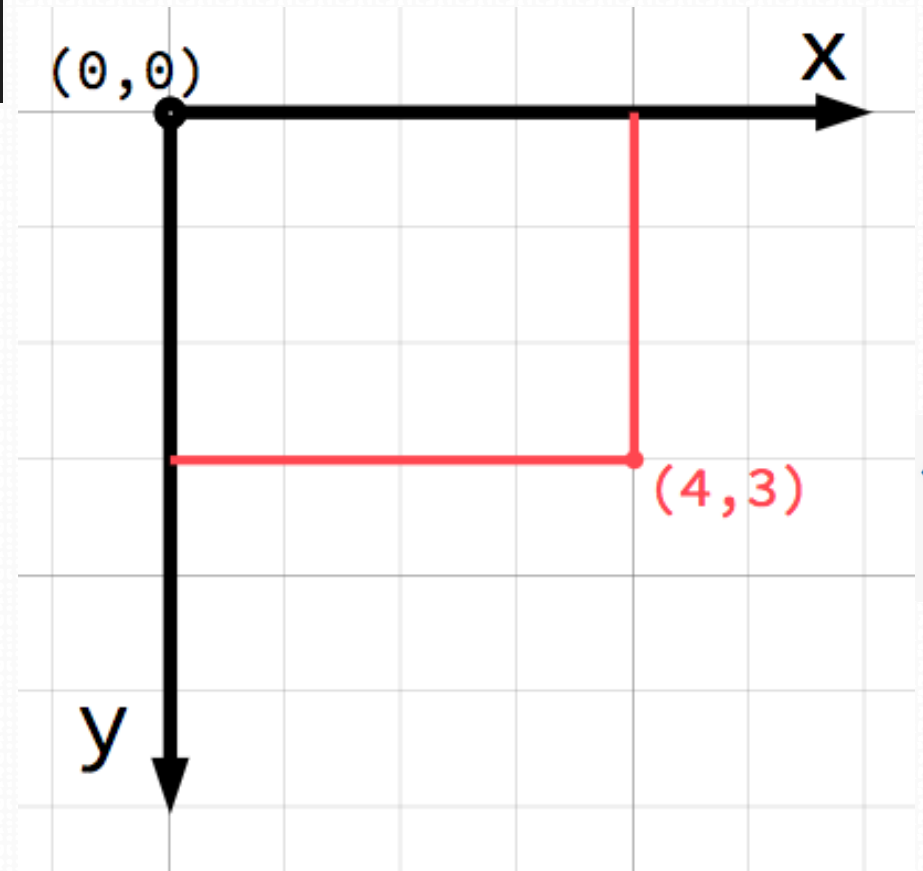


Programando o jogo

- O código a seguir permite mudar a posição do

```
public override void _Process(double delta)
{
    GlobalPosition = GlobalPosition + new Vector2(1,0);
}
```

- GlobalPosition é uma variável que diz qual a posição global do nosso objeto no cenário.
- Essa posição é definida por um plano cartesiano como o apresentado na imagem.
- Note também que o ponto 0,0 está posicionado no canto superior esquerdo da nossa tela visível.



Programando o jogo

- O código a seguir permite mudar a posição do

```
public override void _Process(double delta)
{
    GlobalPosition = GlobalPosition + new Vector2(1,0);
}
```

- $\text{Vector2}(x,y)$ é uma variável com os valores x,y ideais para operarmos no plano cartesiano

