

Criando um jogo básico em Godot com C# (parte2)

Prof. Thiago Felski Pereira, MSc.

Visão Geral

- Na primeira parte do tutorial nós aprendemos a colocar um personagem estático em um ambiente 2D e movê-lo nas direções desejadas utilizando as setas do teclado.
- Conseguimos escolher a aparência do nosso jogador carregando uma imagem pronta para ele.
- Além disso garantimos que os movimentos diagonais não seriam feitos mais rápido que os movimentos horizontais e verticais.



[2]



UNIVALI

Visão Geral

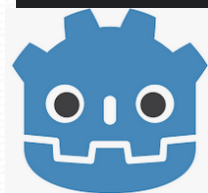
- Provavelmente já sentimos falta de algumas funcionalidades
 - Nosso mapa está vazio.
 - Não há inimigos ou paredes para interagirmos.
 - Nosso personagem não anda ou pula.
 - Ele é apenas uma imagem que mudamos de posição.
- Vamos tentar solucionar alguns desses problemas agora.



[3]

Estrutura de um jogo no Godot

- JOGO
 - Cena1
 - Cena2
 - Nodo1 ← (um script)
 - Nodo1a ← (outro script)
 - Nodo2
 - Cena3
- Quando o pai se move/rotaciona
 - O filho também move/rotaciona
- Quando o pai é removido da cena
 - O filho também é removido
- Quando o pai é deletado (*free*)
 - O filho também é deletado (*free*)



Nodos

- Os jogos no Godot são uma coleção de nodos que interagem entre si.
 - Esses nodos podem ter diversas funcionalidades.
- Podemos estender as funcionalidades de um novo adicionando um script a ele.
- Podemos interagir com outros nodos carregando as informações deles nos nossos scripts.
 - `GetNode<>(caminho)`
 - `GetParentNode<>()` para pegar o nodo pai na hierarquia

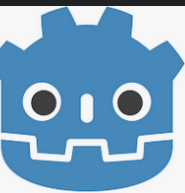


Nodos

- Já vimos também que os nossos scripts tem 2 processos principais

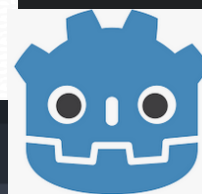
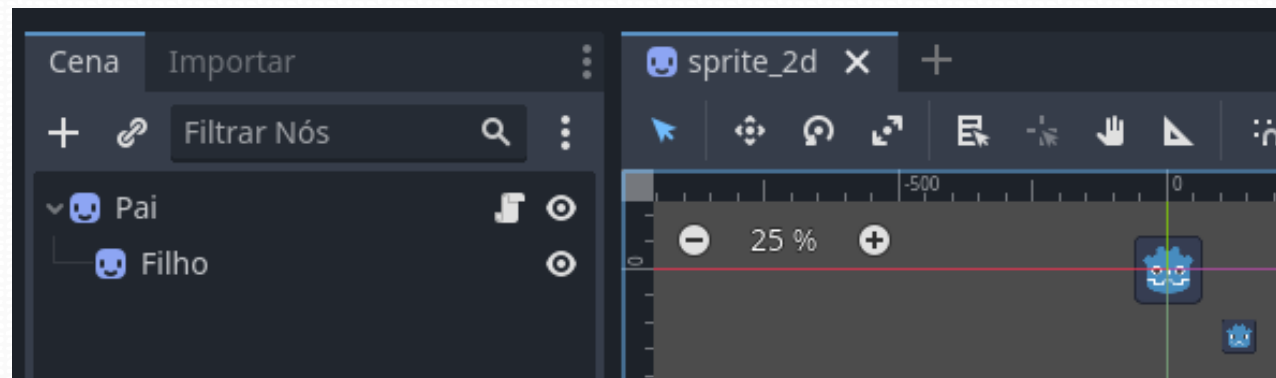
```
// Chamado quando o Nodo entra na Cena pela primeira vez.  
1 reference  
public override void _Ready() {    }  
  
// Chamado a cada frame. 'delta' é o tempo aproximado de um frame  
1 reference  
public override void _Process(double delta) {    }
```

- Esses processos serão muito importantes quando quisermos adicionar novos nodos a uma cena em andamento



Testando o conceito de pai e filho

- Crie um nodo do tipo Sprite2D e chame de Pai.
 - Adicione uma imagem ao Pai.
 - Adicione um script de movimento ao Pai.
 - Vai ficar parecido com o projeto anterior.
- Crie um nodo filho, também do tipo Sprite2D e chame de Filho.
 - Adicione uma imagem ao Filho e desloque ela um pouco.
- Execute a cena do Pai
 - Note que a posição do filho muda junto com o Pai.
 - Se rotacionarmos o Pai o Filho também deverá rotacionar.
 - E assim por diante.



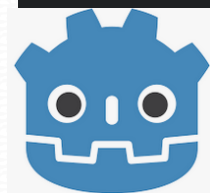
[7]

Testando o conceito de pai e filho

- Agora vamos tentar incluir o Filho no script do Pai e alterar sua posição

```
public override void _Process(double delta)
{
    Godot.Sprite2D filho = this.GetNode<Godot.Sprite2D>("Filho");
    if(Input.IsActionPressed("cima")) {
        GlobalPosition = GlobalPosition + new Vector2(0,-2);
        filho.GlobalPosition = new Vector2(0,0);
    }
}
```

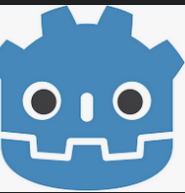
- No processo nós pegamos o nodo do Filho e definimos que quando apertarmos para cima a nova posição do filho será na posição (0,0) que é o canto superior esquerdo.
- Tente testar esse código.



Testando o conceito de pai e filho

- **DESAFIO**

- Tente utilizar tudo o que você aprendeu para fazer o Filho ficar sempre atrás do Pai.
 - Por exemplo, quando o Pai estiver indo para cima, o Filho vai ficar logo atrás dele.
 - Mais ou menos como naquele jogo snake.



[9]



UNIVALI

Cena

- Cenas consistem de nodos e outras cenas.
- A cena possui **apenas um** nodo raiz.
- A cena é salva em um arquivo e quando queremos usar essa cena devemos:
 - Utilizar a função `GD.Load("caminho/para/arquivo_cena")` para criar um objeto do tipo `PackedScene`.
 - Criar uma instância desse objeto com método `packedScene.Instance()`
 - O "tipo" da cena é do mesmo tipo do nodo raiz.



Testando o conceito de instância

- Crie um novo projeto: `CAU_tutorial03`
- Crie um nodo do tipo `Sprite2D`.
 - Adicione uma textura: `icon.svg`
 - Adicione um script `c#` a esse nodo.
 - Vamos mudar um pouco nosso script em relação ao que temos feito.
 - ...



Testando o conceito de instância

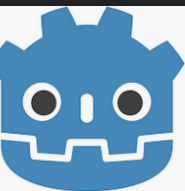
- Usamos a função `GD.Randi() % 4`
 - Gerando um número aleatório entre 0 e 3.
- Também utilizamos o `switch case`
 - mas pode-se continuar utilizando o `if` e obter o mesmo resultado.
- Teste essa cena

```
public override void _Process(double delta)
{
    uint aleatorio = GD.Randi() % 4; // 0 ~ 3
    switch(aleatorio) {
        case 0: //cima
            GlobalPosition = GlobalPosition + new Vector2(0,-4);
            break;

        case 1: //baixo
            GlobalPosition = GlobalPosition + new Vector2(0,4);
            break;

        case 2: //esquerda
            GlobalPosition = GlobalPosition + new Vector2(-4,0);
            break;

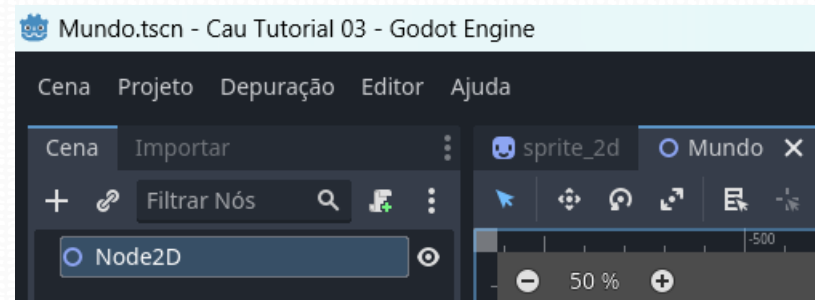
        case 3: //direita
            GlobalPosition = GlobalPosition + new Vector2(4,0);
            break;
    }
}
```



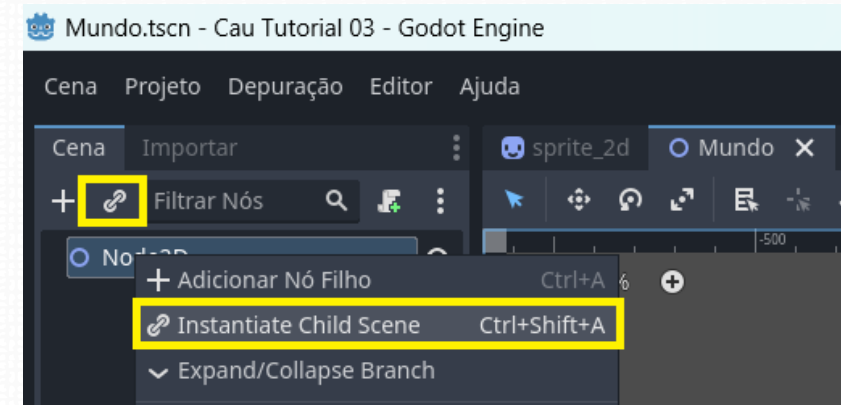
Testando o conceito de instância

- Vamos criar uma nova cena e colocar várias instâncias da cena que criamos anteriormente
 - Cena -> Nova Cena ou CTRL+N

- Adicione um **Node2D** a essa Cena
- Salve com o nome de **Mundo.tscn**

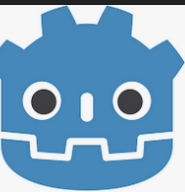
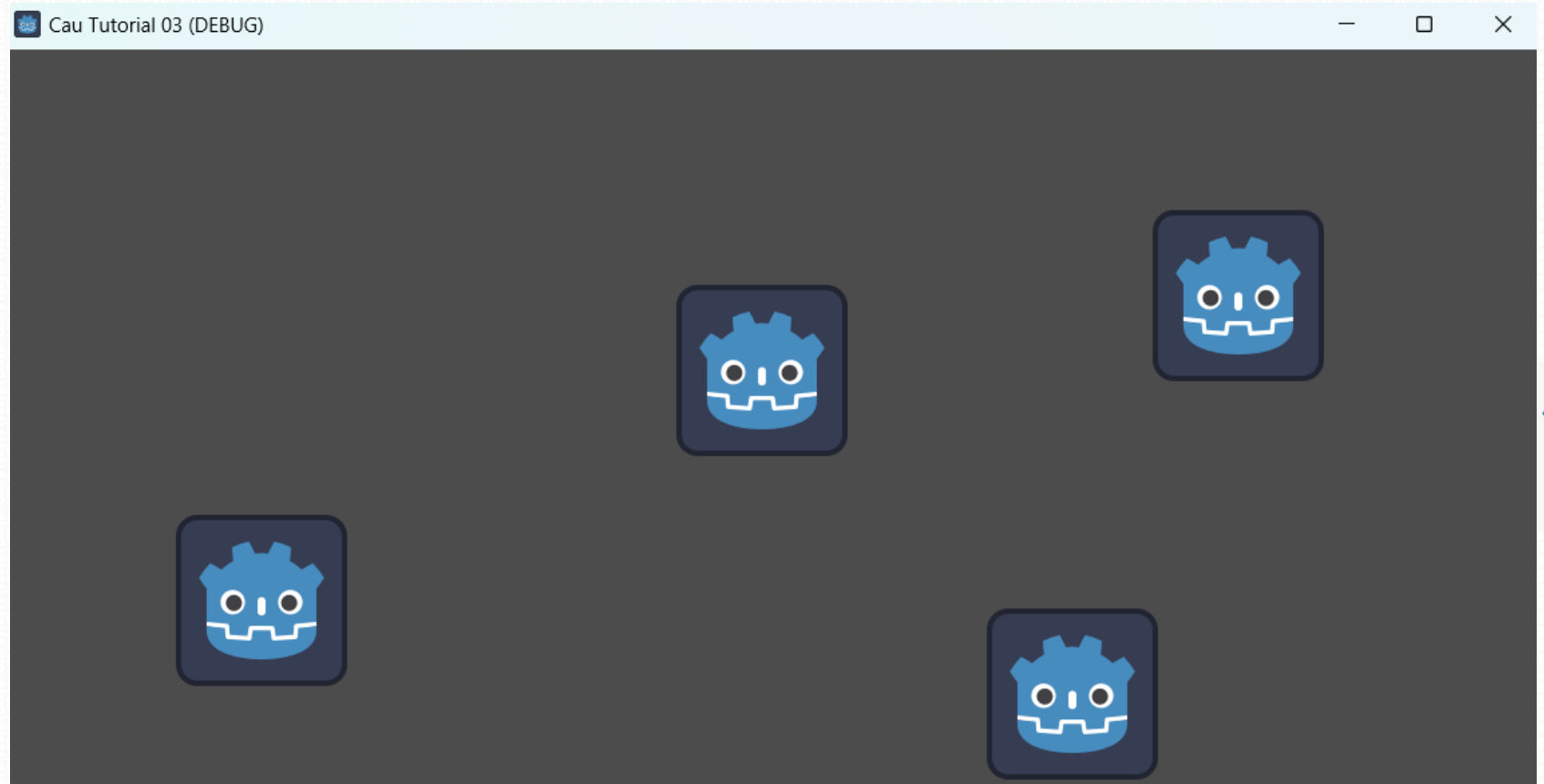


- Adicione a Cena anterior a essa Cena chamada Mundo
 - Botão direito no Node2D -> Instanciate Child Scene
 - Adicione sprite_2d.tscn
- Teste essa cena



Testando o conceito de instância

- A cena mundo não ficou diferente da cena sprite_2d, mas a vantagem é que você pode incluir várias instâncias da cena sprite_2d ao seu mundo e cada uma dessas cenas vai herdar todas as suas funcionalidades



(14)

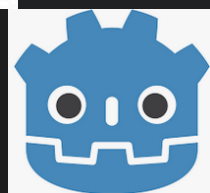


UNIVALI

Criando Instâncias pelo código

- Para criar instâncias da Cena pelo código, nós precisaremos de um **script** para nosso mundo
 - Crie um **script** para o nodo raiz da cena **Mundo**
 - Lembre-se de criar um script c#
- Note que criamos um novo método, além de `_Ready()` e `_Process()` temos o `UnhandledInput()`
 - Esse processo é chamado sempre que temos uma entrada não tratada

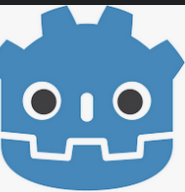
```
public override void _UnhandledInput(InputEvent @event)
{
    if (@event is InputEventMouseButton mouseEvent){
        var sprite = GD.Load<PackedScene>("res://sprite_2d.tscn");
        var inst = sprite.Instantiate();
        AddChild(inst);
    }
}
```



Criando Instâncias pelo código

- ...
- Note também que empacotamos a Cena na variável sprite
- Instanciamos ela
- Em seguida adicionamos uma cena filha que é uma cópia dela

```
public override void _UnhandledInput(InputEvent @event)
{
    if (@event is InputEventMouseButton mouseEvent){
        var sprite = GD.Load<PackedScene>("res://sprite_2d.tscn");
        var inst = sprite.Instantiate();
        AddChild(inst);
    }
}
```



(16)



UNIVALI

