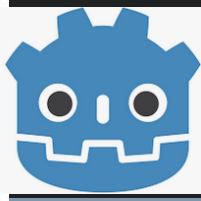
Detectando Colisões: Area2D

Prof. Thiago Felski Pereira, MSc.

Visão Geral

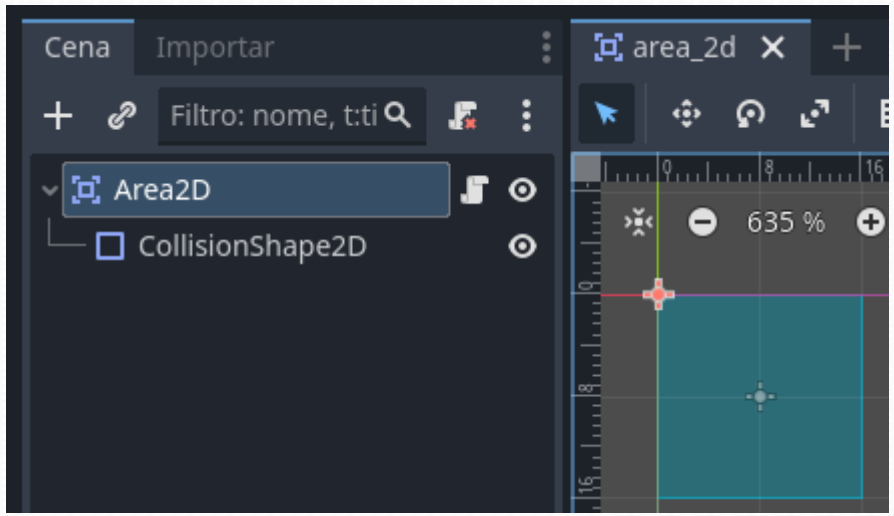
- Em um jogo com movimento podem acontecer diferentes tipos de colisões.
  - Dois objetos se movendo podem colidir;
  - Um objeto pode colidir com objetos estáticos como paredes e espinhos;
  - Algumas vezes queremos “colidir” com alguns objetos para poder interagir com ele;
    - um botão, alavanca, escada, porta, ...
- Essas colisões enriquecem a experiência de jogo, tornando-o mais interativo e desafiador; e
- Em um jogo temos a cooperação de vários desenvolvedores (artistas, roteiristas, programadores) e não queremos que o trabalho de um dificulte o de outro.
  - Por isso iremos detectar colisões de forma modular, ou seja, podendo reaproveitar em diferentes ambientes e situações.



UNIVALI

Area2D com detecção de sinais

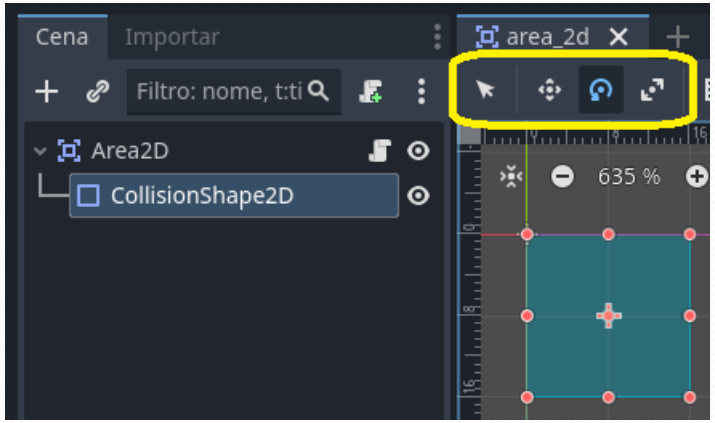
- O **Godot** possui um nó chamado `Area2D`, ele nos permite utilizar **sinais** para, facilmente, detectar quando um objeto entrou ou saiu de sua área.
- Crie uma cena, como a da imagem, a seguir:



- Note algumas informações importantes:
  - Não há imagens, pois essa área será colocada por cima das imagens do nosso jogo.
  - Há um script para programarmos as ações quando incluirmos sinais.

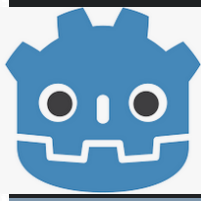
Area2D com detecção de sinais

- Mas se a área for maior ou menor do que a área inicial que foi definida?
  - Da mesma forma que a seta é utilizada para posicionar a Area2D no local desejado, existem outras ferramentas para redimensionar/rotacionar a Area2D de acordo com a necessidade.



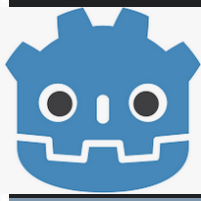
Area2D com detecção de sinais

- Mas se o objeto se mover?
  - O próprio Godot cuida disso, basta que esta seja uma cena filha do objeto que irá se mover.



Area2D com detecção de sinais

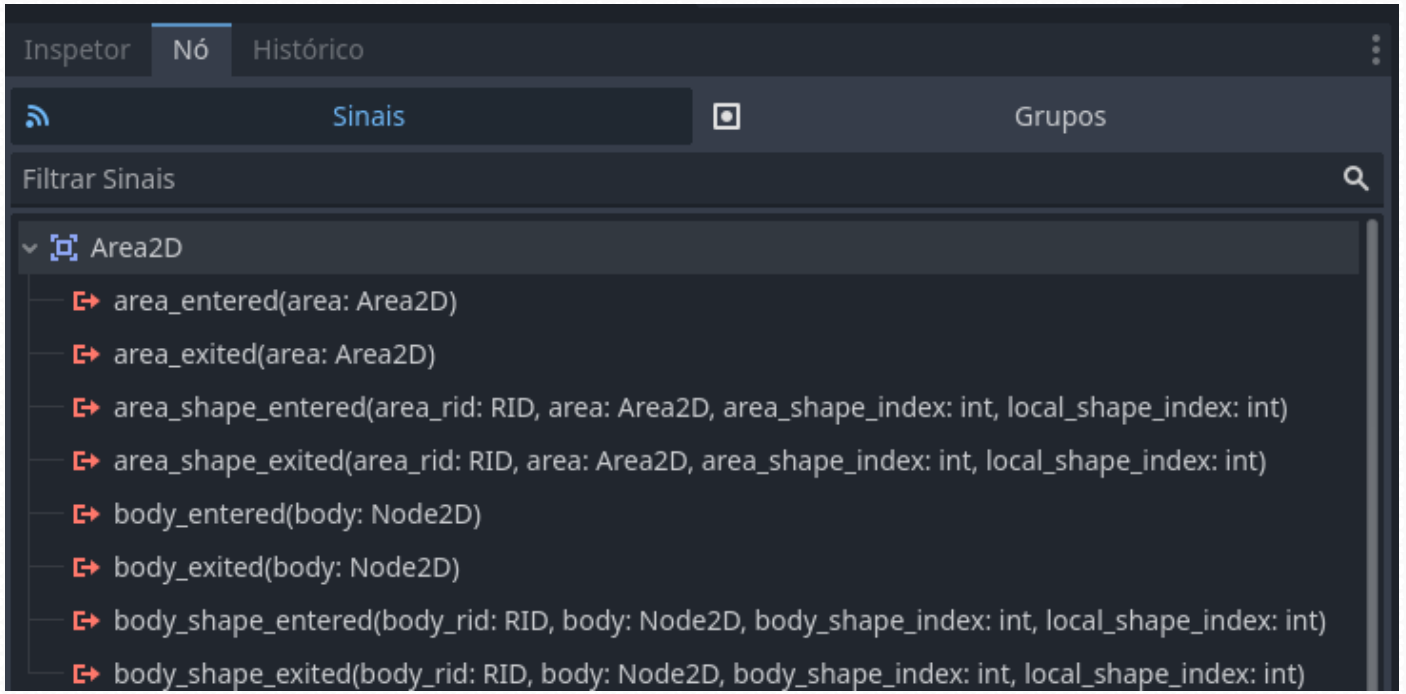
- Mas se as áreas tiverem funções diferentes?
  - Isso pode ser resolvido por código.
  - O comando `[Export]` nos permitirá indicar o tipo de Area2D que estamos criando. Assim uma variável `[Export] private string` tipo, poderia diferenciar uma área de "`coração`" de uma área de "`espinho`".



UNIVALI

Area2D com detecção de sinais

- Com a **Area2D** selecionada, mude para aba **Nó** (ao lado do **Inspetor**)



- A **Area2D** já possui uma série de **sinais** úteis, basta conectá-los e programa-los.
- Observe, que ao conectar um sinal, precisaremos indicar o **Script** que implementará, suas regras.

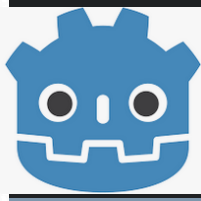
UNIVALI

Area2D com detecção de sinais

- O sinal `body_entered(body: Node2D)`
  - É ativado quando um body entrar na **Area2D**.
  - Exemplos de body: `CharacterBody2D`, `StaticBody2D`, `RigidBody2D`, ...

- O sinal `body_exited(body: Node2D)`
  - É ativado quando um body sair da Area2D.

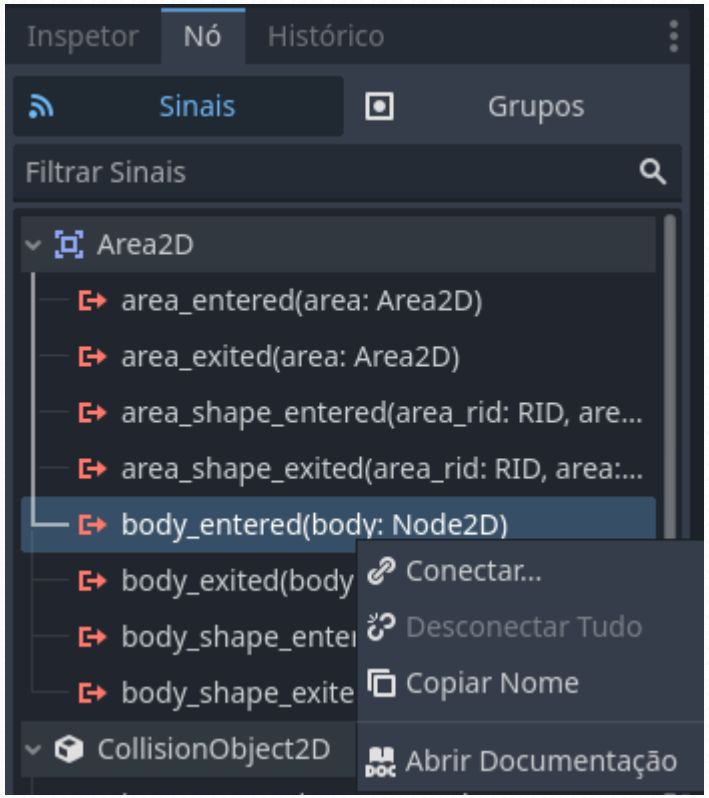
- Com esses dois sinais podemos identificar:
  - Se o Jogador entrou, ou seja, na Area2D;
  - Se o Jogador está na Area2D, ou seja, detectou que entrou e não detectou que saiu;
  - ...



UNIVALI

EXEMPLO: Area2D

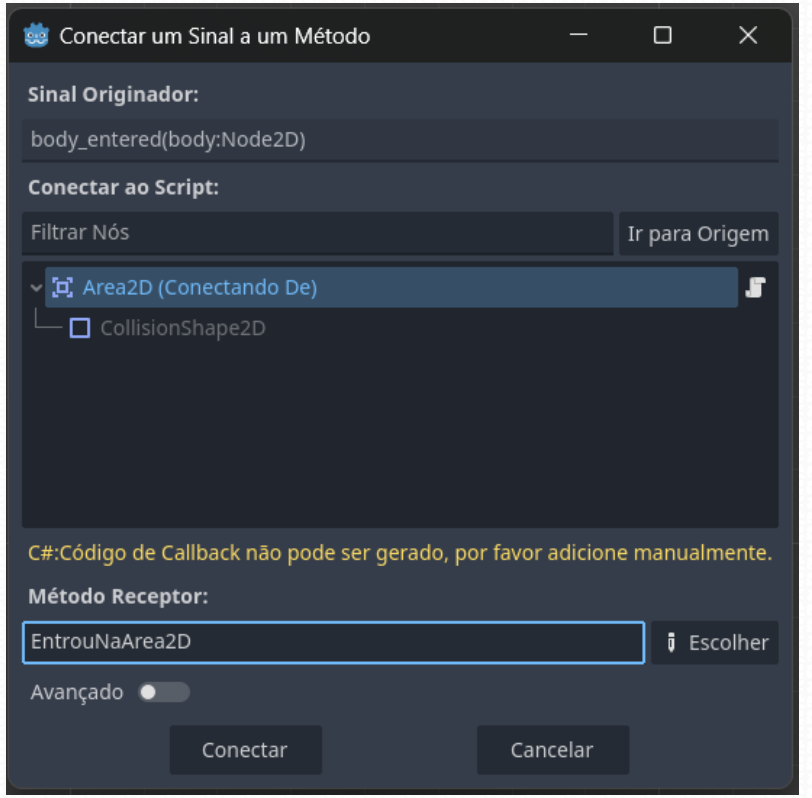
- Conecte o sinal `body_entered(body: Node2D)`



UNIVALI

EXEMPLO: Area2D

- Conecte o sinal `body_entered(body: Node2D)`
  - Escolha o Script; e
    - Conectar ao Script: **Area2D**
  - Escolha o nome da função que será chamada.
    - Método receptor: **EntrouNaArea2D**



UNIVALI

EXEMPLO: Area2D

- Crie/Edite a função `EntrouNaArea2D()`

```
public void EntrouNaArea2D(Node2D body) {

}
```

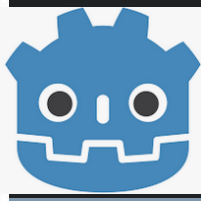
- Observe que essa função será chamada sempre que qualquer body colidir/entrar na **Area2D**, mas podemos diferenciar esses body com testes simples.

```
public void EntrouNaArea2D(Node2D body) {
    if (body is CharacterBody2D) { //entra se for um personagem

    }

    if (body is Raposa) { //entra se for especificamente a raposa

    }
}
```

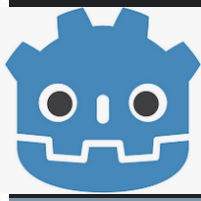


UNIVALI

EXEMPLO: Area2D

- O código, a seguir, chama uma função no script da **Raposa** quando a **Raposa** entrar na **Area2D**.

```
public void EntrouNaArea2D(Node2D body) {
    if (body is Raposa) { //entra se for especificamente a raposa
        ((Raposa)body).FuncaoNoScriptDaRaposa();
    }
}
```



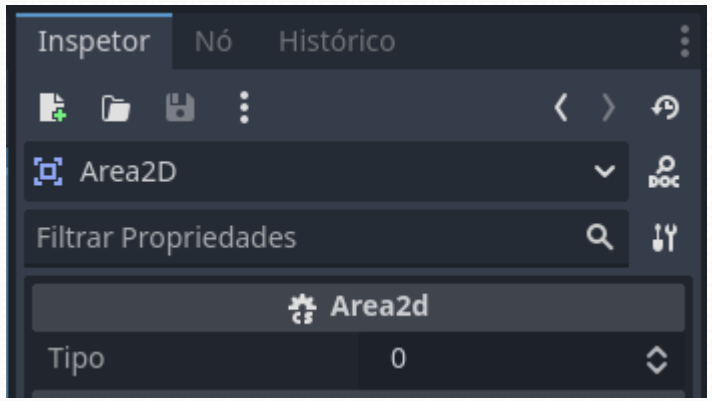
UNIVALI

EXEMPLO: Area2D

- Falta apenas diferenciarmos o tipo de **Area2D** que estamos inserindo no **Mapa**.
  - O comando `[Export]` serve para deixar uma variável visível na interface do Godot. Com isso, é possível criar uma variável e definir seu valor no momento que incluímos a **Area2D**.

```
[Export] int tipo = 0;
```

- Essa variável aparecerá no **Inspetor** e com isso poderemos selecionar sua funcionalidade.
- Feito isso, é só utilizar o parâmetro no código com um teste simples.



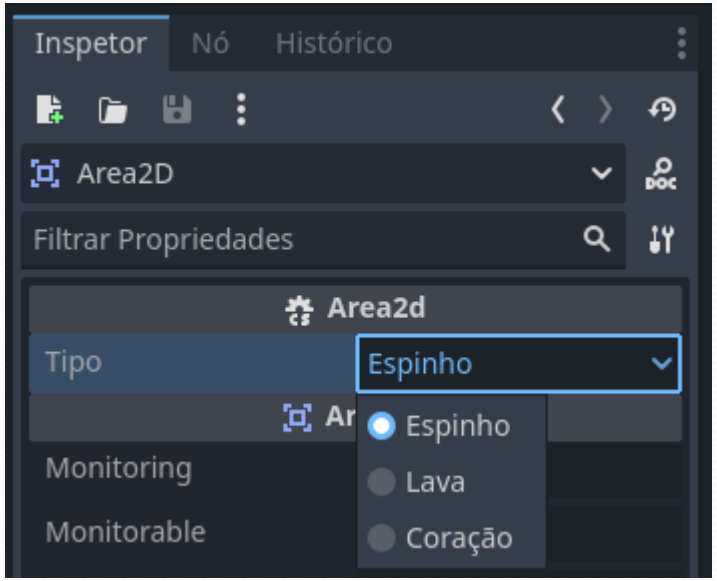

```
if (tipo == 1) {
    ((Raposa)body).FuncaoCoracaoDaRaposa();
}
```


UNIVALI

EXEMPLO: Area2D

- Também é possível utilizar o `[Export]`, no exemplo, a seguir o `[Export]`, irá gerar uma lista de opções para o usuário selecionar.

```
[Export(PropertyHint.Enum, "Espinho,Lava,Coração")]
5 references
public string tipo { get; set; } = "Espinho";
```




```
public void EntrouNaArea2D(Node2D body) {
    if (body is Raposa) { //entra se for especificamente a raposa
        if (tipo == "Coração") {
            ((Raposa)body).FuncaoCoracaoDaRaposa();
        }
    }
}
```


UNIVALI

Obrigado pela atenção!