A CRONICA DO GDSCRIPT

DESBRAVANDO A TERRA DO CODIGO

HANDERSON DIAS

Desbravando o GDScript 4.3

Domine os Comandos com Exemplos Reais

O GDScript 4.3 é a linguagem de programação principal para o motor de jogo Godot, e é uma excelente ferramenta para criar jogos de maneira rápida e eficiente. Neste ebook, vamos explorar os comandos mais importantes e como utilizá-los de forma prática para você começar a desenvolver seus próprios projetos.

1 - Declaração de Variáveis

Declaração de Variáveis

Em GDScript, declaramos variáveis de forma simples. Veja o exemplo:

```
# Declaração de variáveis
var vida = 100
var nome = "Heroi"
var inimigo = null
```

Explicação:

- var é a palavra-chave para declarar uma variável.
- vida e nome são variáveis simples, armazenando um número e uma string, respectivamente.
- inimigo é uma variável inicializada com null, o que significa que ainda não aponta para nenhum objeto.

2 - Funções

Funções

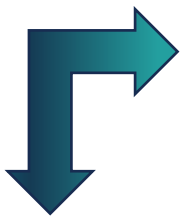
As funções são blocos de código reutilizáveis. Veja um exemplo de uma função que calcula a vida restante de um personagem após um dano:

```
# Função para aplicar dano
func aplicar_dano(dano):
    vida -= dano
    print("Vida restante: ", vida)
```

Explicação:

- func é a palavra-chave para declarar uma função.
- aplicar_dano(dano) recebe um parâmetro chamado dano e subtrai esse valor da variável vida.
- print() imprime a mensagem no console, mostrando a vida restante.

3 - Estruturas Condicionais (if/else)



Estruturas Condicionais (if/else)

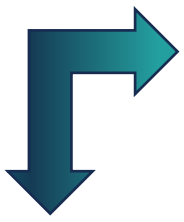
Com o comando if, podemos criar condições para que o código execute ações específicas dependendo do que acontecer no jogo.

```
# Verificando se a vida é maior que 0
if vida > 0:
    print("O herói está vivo!")
else:
    print("O herói morreu!")
```

Explicação:

- if verifica se a condição é verdadeira (vida maior que 0).
- Caso contrário, o código entra no bloco else e executa a ação correspondente.

4 - Loops (for/while)



Loops (for/while)

Loops são usados para repetir um bloco de código várias vezes. O for é útil quando você sabe quantas vezes quer repetir.

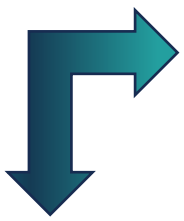
```
# Loop for para criar 5 inimigos
for i in range(5):
    print("Inimigo ", i + 1, " apareceu!")
```

Explicação:

- for i in range(5) faz o loop repetir 5 vezes, começando de 0 até 4.
- A cada iteração, um inimigo é criado, e o número do inimigo é exibido no console.

5 -

Controle de Entrada do Usuário



Controle de Entrada do Usuário

Você pode capturar entradas do usuário no Godot com o GDScript. Aqui, vamos verificar se o jogador apertou a tecla "espaço" para pular:

```
# Verificando a entrada do jogador
if Input.is_action_just_pressed("ui_accept"):
    print("O herói pulou!")
```

Explicação:

- Input.is_action_just_pressed("ui_accept") verifica se a tecla "espaço" (ou outra definida no projeto) foi pressionada.
- Se o jogador apertar a tecla, o código imprime "O herói pulou!".

6 -

Atribuição de Objetos (Instanciando)

Atribuição de Objetos (Instanciando)

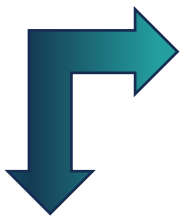
Em Godot, podemos criar instâncias de objetos para adicionar à cena, como um inimigo ou uma plataforma.

```
# Criando um inimigo
var inimigo = load("res://Inimigo.tscn").instance()
add_child(inimigo)
```

Explicação:

- load("res://Inimigo.tscn") carrega um arquivo de cena do tipo .tscn (no caso, um inimigo).
- .instance() cria uma instância da cena carregada.
- add_child(inimigo) adiciona o inimigo à cena atual.

7 - Sinais (Signals)



Sinais (Signals)

Os sinais permitem que um objeto envie mensagens para outros objetos quando algo acontecer. Um exemplo simples de sinal:

```
# Emitindo um sinal quando o herói colide com um inimigo
signal heroi_colidiu

func _on_Heroi_body_entered(body):
    if body.is_in_group("inimigos"):
        emit_signal("heroi_colidiu")
```

Explicação:

- signal heroi_colidiu define um novo sinal.
- O método _on_Heroi_body_entered(body) é chamado quando o corpo do herói entra em contato com outro corpo. Se for um inimigo, o sinal heroi_colidiu é emitido.

8 - Acessando Propriedades de Objetos

Acessando Propriedades de Objetos

Em Godot, podemos acessar e modificar propriedades de outros objetos diretamente.

```
# Acessando e alterando a posição de um inimigo
inimigo.position = Vector2(300, 200)
```

Explicação:

- inimigo.position acessa a propriedade de posição do objeto.
- Vector2(300, 200) define uma nova posição para o inimigo.

AGRADECIMENTOS

OBRIGADO POR LER ATÉ AQUI

Esse Ebook foi gerado utilizando IA, e diagramado por humano.

Esse conteúdo foi gerado com fins didáticos de construção, não foi realizado uma validação aprofundada no conteúdo e pode conter erros gerados por uma IA.

https://github.com/handersongodias/prompts-recipe-to-create-a-ebook