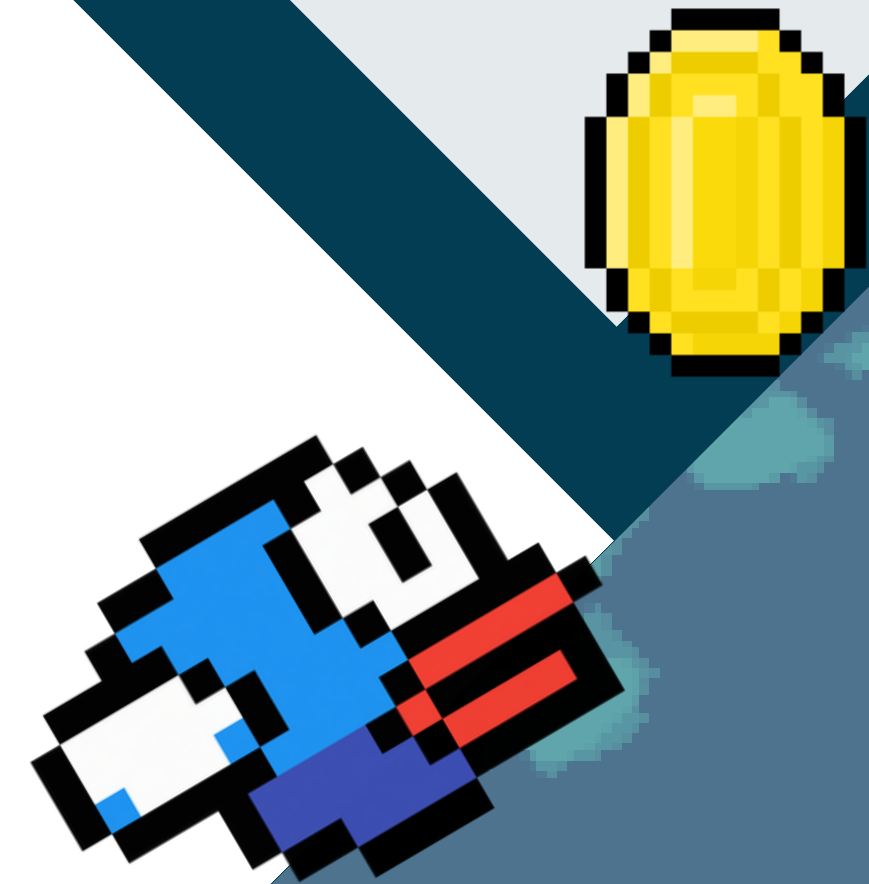


Aula 2

NIVELAMENTO

INTELI GAME LAB



Assuntos do encontro

POO

PROGRAMAÇÃO ORIENTADA A OBJETOS

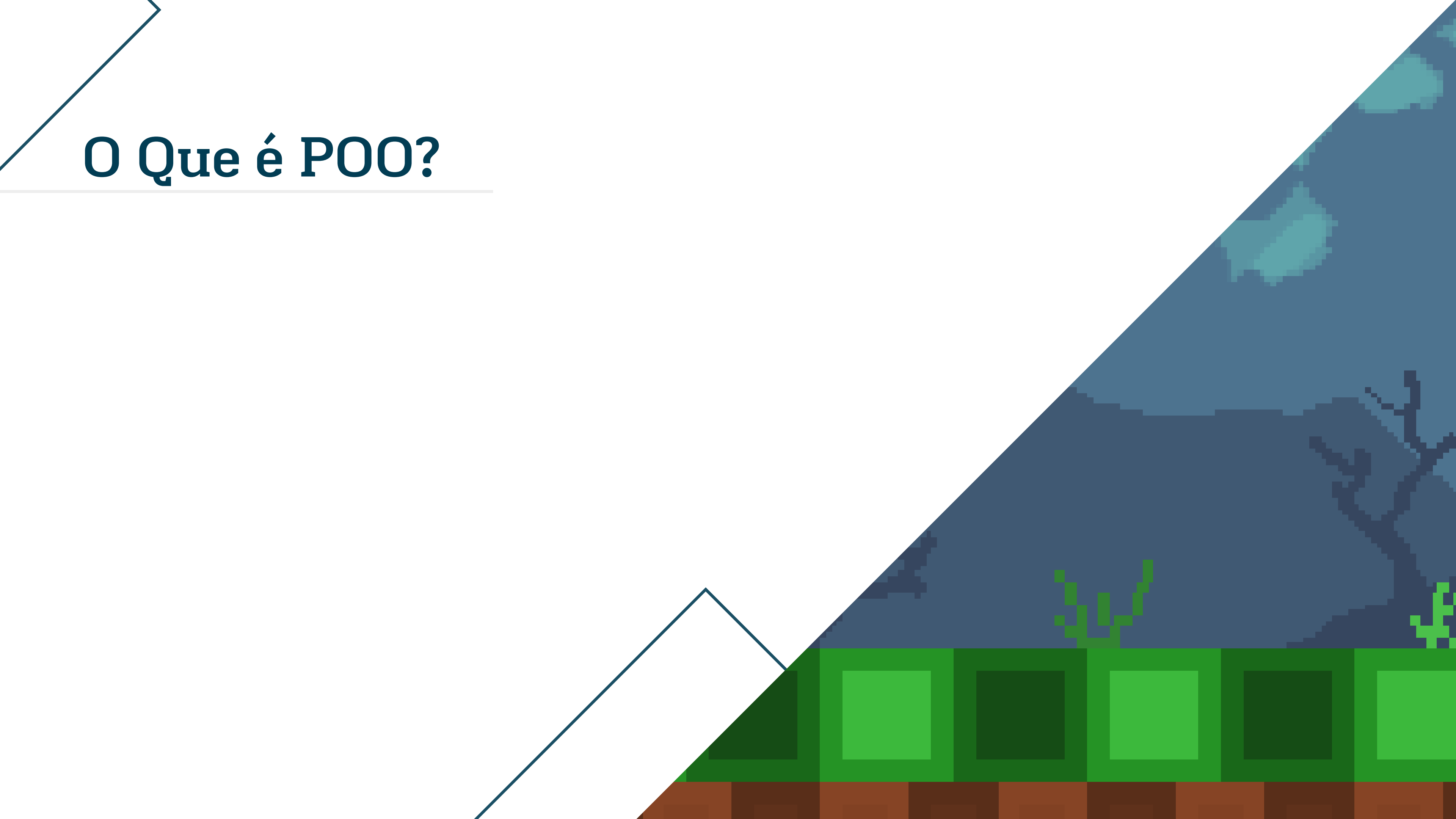
INSTANCIAS

Criando objetos dinamicamente em tempo de código

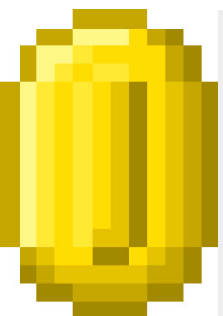
APLICAÇÕES

Exemplos de aplicação destes conceitos

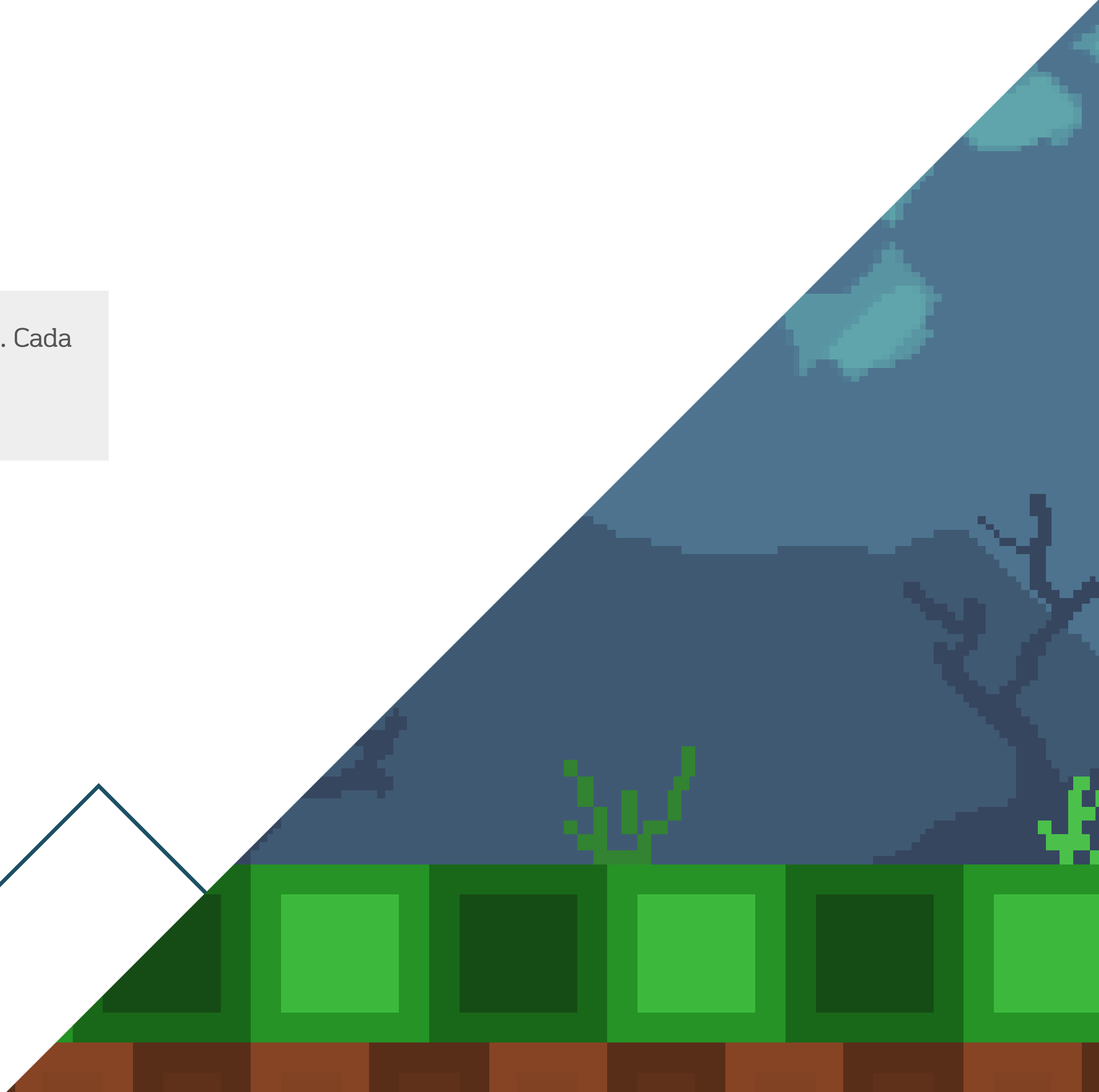
O Que é POO?



O Que é POO?



A POO é um paradigma de programação baseado em objetos. Cada um destes objetos pode possuir atributos ou propriedades



O Que é POO?



A POO é um paradigma de programação baseado em objetos. Cada um destes objetos pode possuir atributos ou propriedades

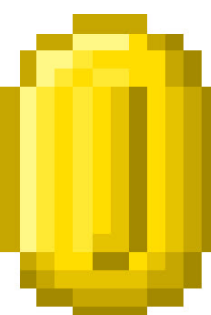


Paradigma?

O Que é POO?



A POO é um paradigma de programação baseado em objetos. Cada um destes objetos pode possuir atributos ou propriedades



Paradigma: modelo de abordagem para o desenvolvimento de softwares.

Características



Características

Classe: estrutura e o comportamento de objetos



Características

Classe: estrutura e o comportamento de objetos

Objeto: instância de uma classe



Características

Classe: estrutura e o comportamento de objetos

Objeto: instância de uma classe

Herança: Capacidade de uma classe herdar características de outra



Características

Classe: estrutura e o comportamento de objetos

Objeto: instância de uma classe

Herança: Capacidade de uma classe herdar características de outra

Classe: estrutura e o comportamento de objetos



```
1  ✓ public class Pessoa {
2
3      // Atributos da classe
4      private String nome;
5      private int idade;
6      private String corFavorita;
7
8      // Construtor da classe
9  ✓ public Pessoa(String nome, int idade, String corFavorita) {
10      this.nome = nome;
11      this.idade = idade;
12      this.corFavorita = corFavorita;
13  }
14
15      // Método para imprimir informações sobre a pessoa
16  ✓ public void imprimirInformacoes() {
17      System.out.println("Nome: " + nome + ", Idade: " + idade + ", cor favorita: " + corFavorita);
18  }
19
20  ✓ public static void main(String[] args) {
21      // Criando objetos da classe Pessoa
22      Pessoa pessoa1 = new Pessoa("Alice", 30, "azul");
23      Pessoa pessoa2 = new Pessoa("Bob", 25, "verde");
24
25      // Chamando o método para imprimir informações sobre as pessoas
26      pessoa1.imprimirInformacoes();
27      pessoa2.imprimirInformacoes();
28  }
29 }
```

Atributos:

Características da classe

Atributos:

Características da classe

Construtor

Função que recebe o mesmo nome da classe, esta função é executada toda vez que o a classe é instanciada.

Atributos:

Características da classe

Construtor

Função que recebe o mesmo nome da classe, esta função é executada toda vez que o a classe é instanciada.

Parâmetros

Valores passados entre os parênteses da função

Atributos:

Características da classe

Construtor

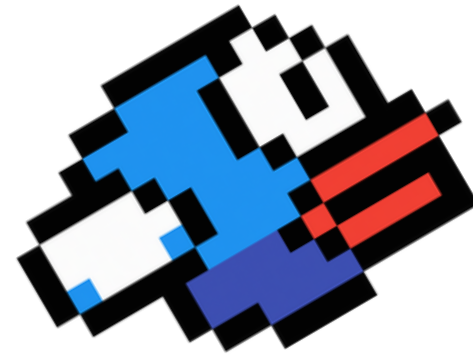
Função que recebe o mesmo nome da classe, esta função é executada toda vez que o a classe é instanciada.

Parâmetros

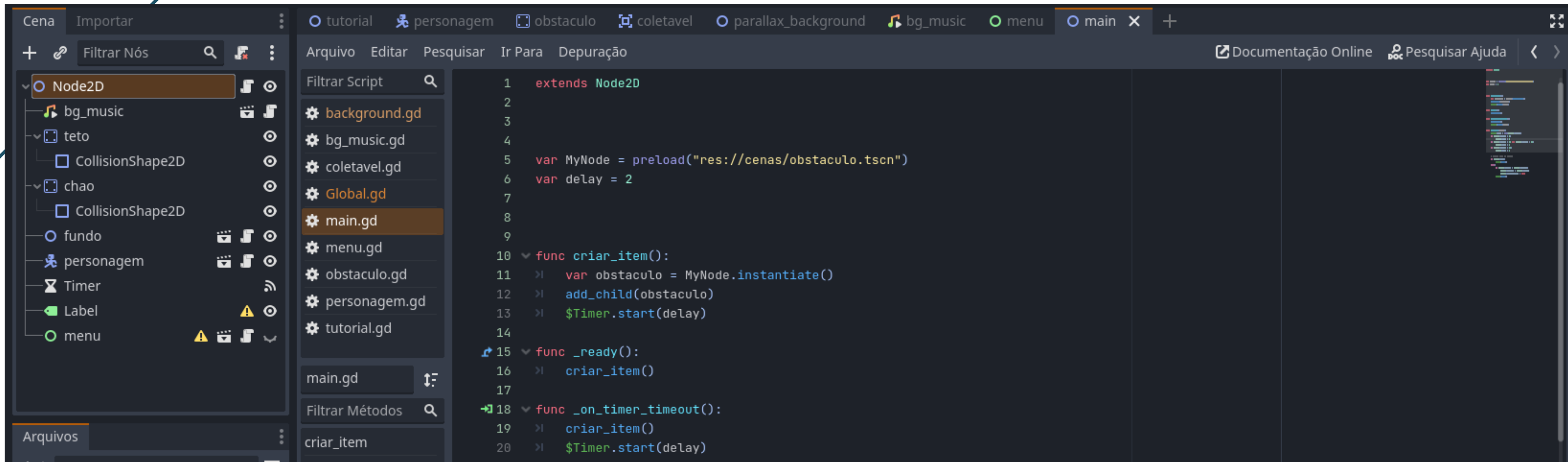
Valores passados entre os parênteses da função

Instânciação

Forma de criar objetos usando a classe escrita como molde.



Instanciando em Godot



Steps

1

A Classe desejada é carregada pelo preload() e recebe o nome de Mynode

2

A função criar_item() é escrita para instanciar o objeto que foi carregado, após isso inicia o objeto Timer.

3

O objeto timer faz uma contagem com base na variável delay, e ao encerrar a contagem, executa novamente a função criar_item()

*

A primeira execução do código é feita na função _ready() da cena atual

```
1  extends Node2D
2
3
4
5  var MyNode = preload("res://cenas/obstaculo.tscn")
6  var delay = 2
7
8
9
10 func criar_item():
11     >| var obstaculo = MyNode.instantiate()
12     >| add_child(obstaculo)
13     >| $Timer.start(delay)
14
15 func _ready():
16     >| criar_item()
17
18 func _on_timer_timeout():
19     >| criar_item()
20     >| $Timer.start(delay)
21     >|
22 func _process(delta):
23     >| $Label.text = str(Global.pontos)
24     >| if Global.pontos > 0:
25         >| Global.fase = 2
26     >| if Global.pontos > 30 and Global.pontos < 60:
27         >| Global.fase = 1
28     >| if Global.pontos > 60:
```



Aplicações e Vantagens

- Criação de objetos em tempo de execução;
- Melhor organização do código;
- Possibilidade de reutilização de código;
- Facilidade de implementar modificações.

OBRIGADO PELA ATENÇÃO!

INTELI GAME LAB

