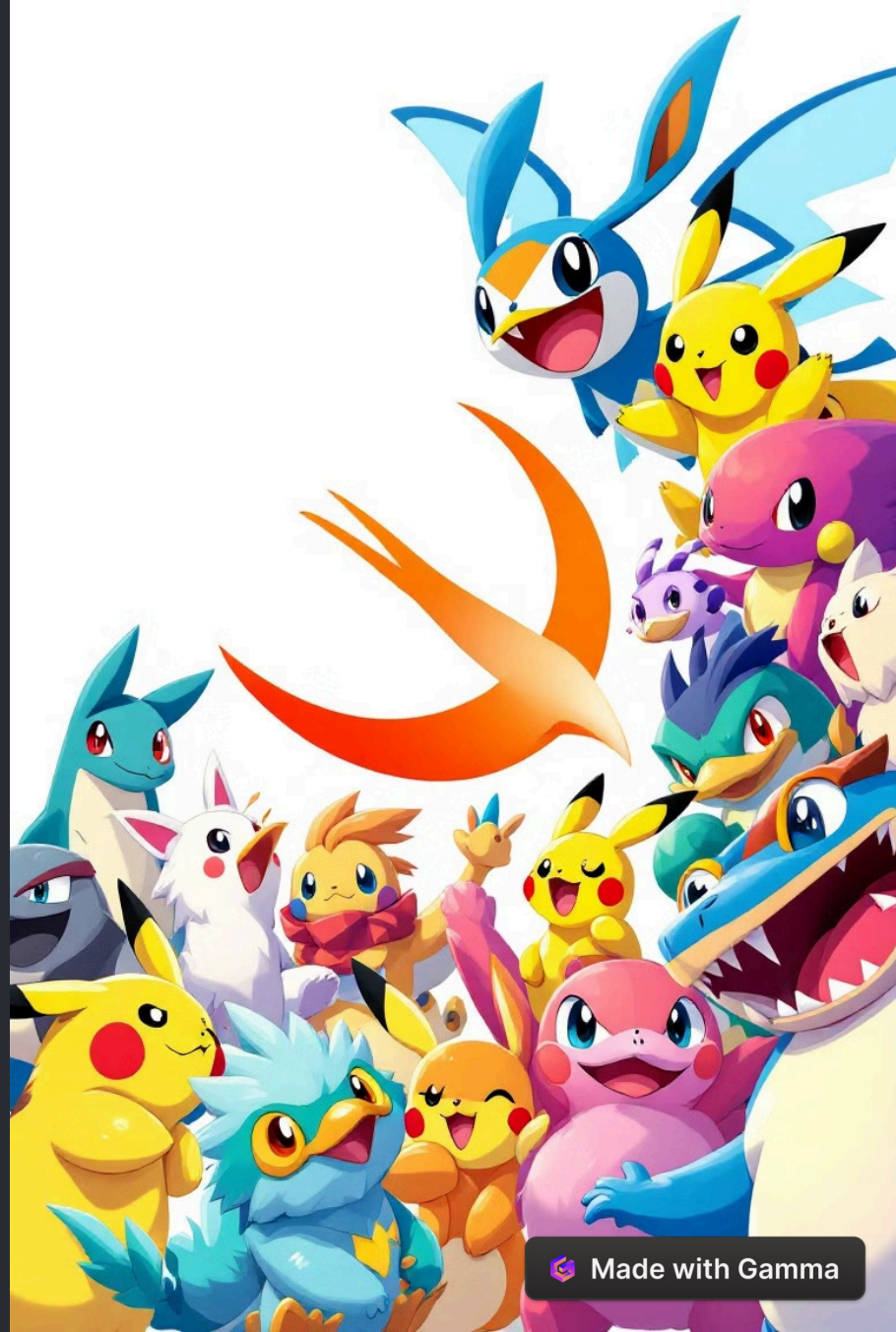
Classes e Objetos Swift com Pokémon

Swift é uma linguagem de programação orientada a objetos. Ela suporta o conceito de objetos e classes.

Um objeto é uma coleção de dados (variáveis) e métodos (funções). Uma classe é um blueprint para esse objeto.

por Isaac Mesquita

RGM: 38211521



Made with Gamma

Classes Swift

O que é uma classe?

Uma classe é um projeto de objetos. É como a planta baixa de uma casa.

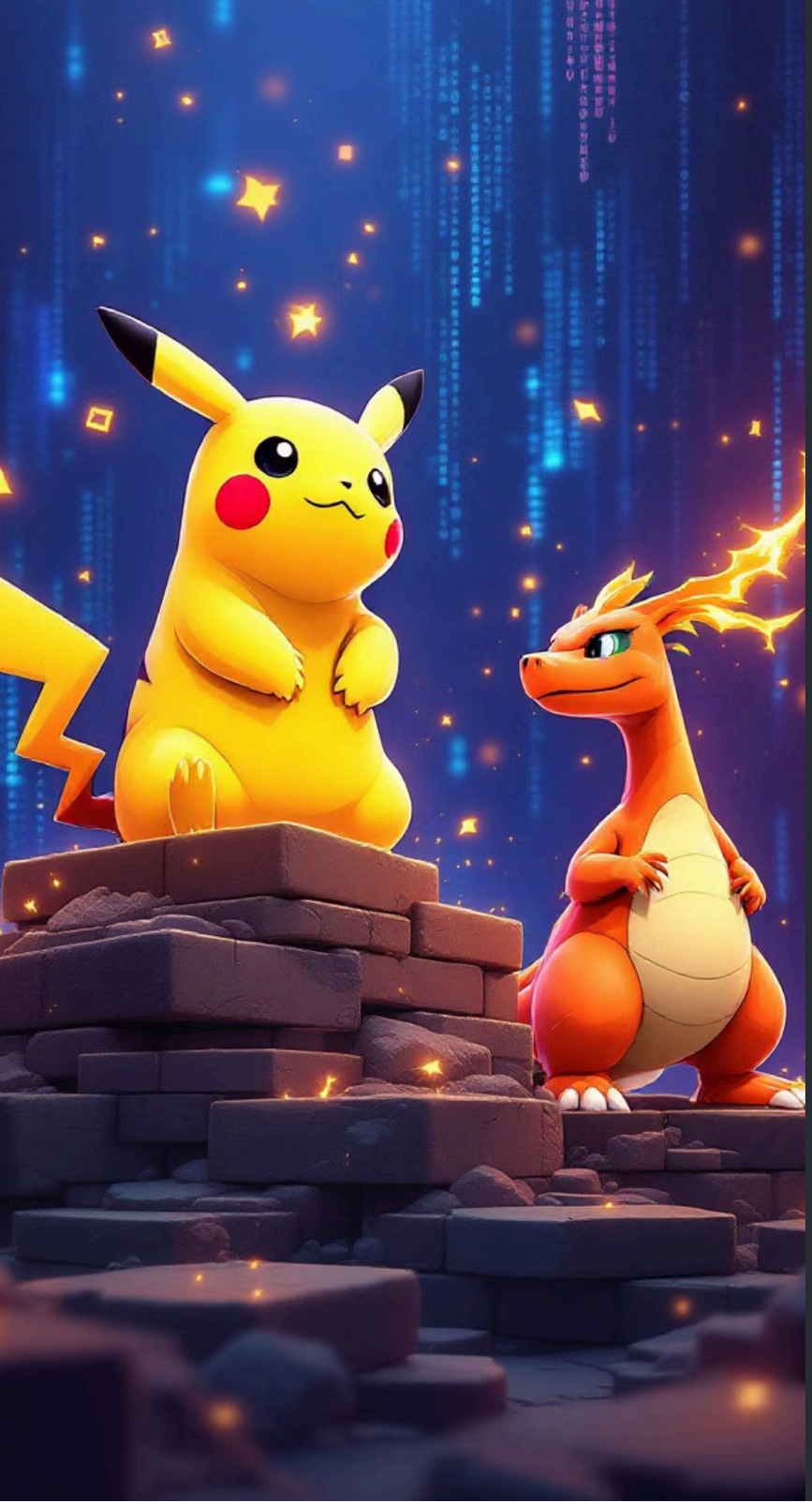
Objetos

Com base na classe, construímos objetos. Como casas feitas do mesmo projeto.

Múltiplos objetos

Podemos criar muitos objetos a partir de uma única classe.

Made with Gamma



Definir Classe Swift

1 Sintaxe

Usamos a palavra-chave "class" para criar uma classe em Swift.

```
class Pokemon{
    //Definição da classe
}
```

Vamos utilizar "Pokemon" como a nossa classe.

2 Componentes

Pokemon é o nome da classe, nome e hp são variáveis com valores padrão.

```
// Criar a classe
class Pokemon{

// Define os atributos
    var nome = ""
    var hp = 0
}
```

As variáveis e constantes dentro de uma classe são chamadas de propriedades.

Made with Gamma

Objetos Swift

Definição

Um objeto é uma instância de uma classe. Por exemplo, pikachu é um objeto da classe Pokemon.

Esta é a sintaxe para criar um objeto.

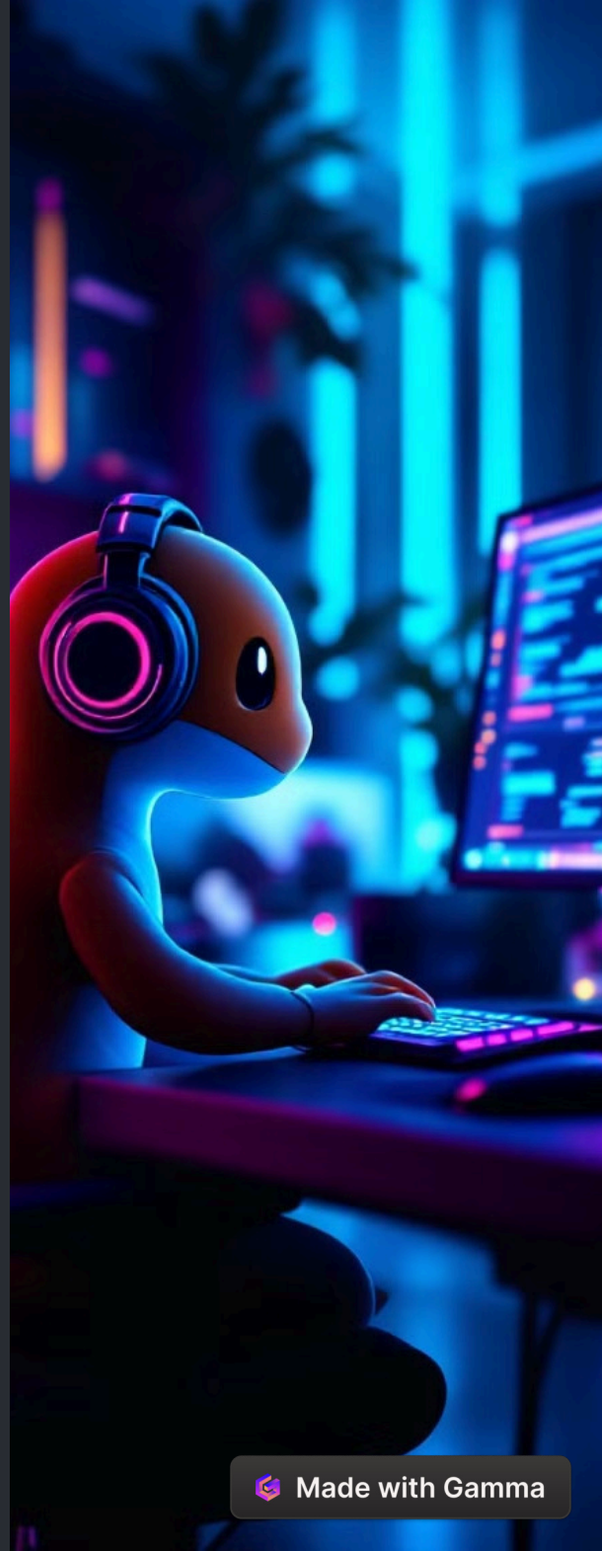
var nomeObjeto = NomeClasse()

Exemplo

var pikachu = Pokemon()

```
1  // Criar a classe
2  class Pokemon{
3
4  // Define os atributos
5      var nome = ""
6      var hp = 0
7  }
8  // Cria o objeto da classe
9  var pikachu = Pokemon()
```

Usamos o objeto para acessar as propriedades da classe.



Made with Gamma

Acessar Propriedades

1

Notação de Ponto

Usamos a notação de ponto (.) para acessar propriedades de uma classe.

2

Modificar Propriedades

pikachu.name = "Pikachu"

3

Acessar Valores

pikachu.hp para obter o valor da propriedade hp.

Made with Gamma

Exemplo: Classe e Objetos

```
// Criar a classe
class Pokemon{

// Define os atributos
    var nome = ""
    var hp = 0
}
// Cria o objeto (chamado de intância da classe)
var pikachu = Pokemon()

// Acessa os atributos da classe e atribui novos valores
pikachu.nome = "Pikachu"
pikachu.hp = 200

print("O nome do pokemon é \(pikachu.nome) e ele tem \
    (pikachu.hp) pontos de vida.")
```

Definir Classe

class Pokemon{

var name = ""

var gears = 0}

Criar Objeto

var pikachu = Pokemon()

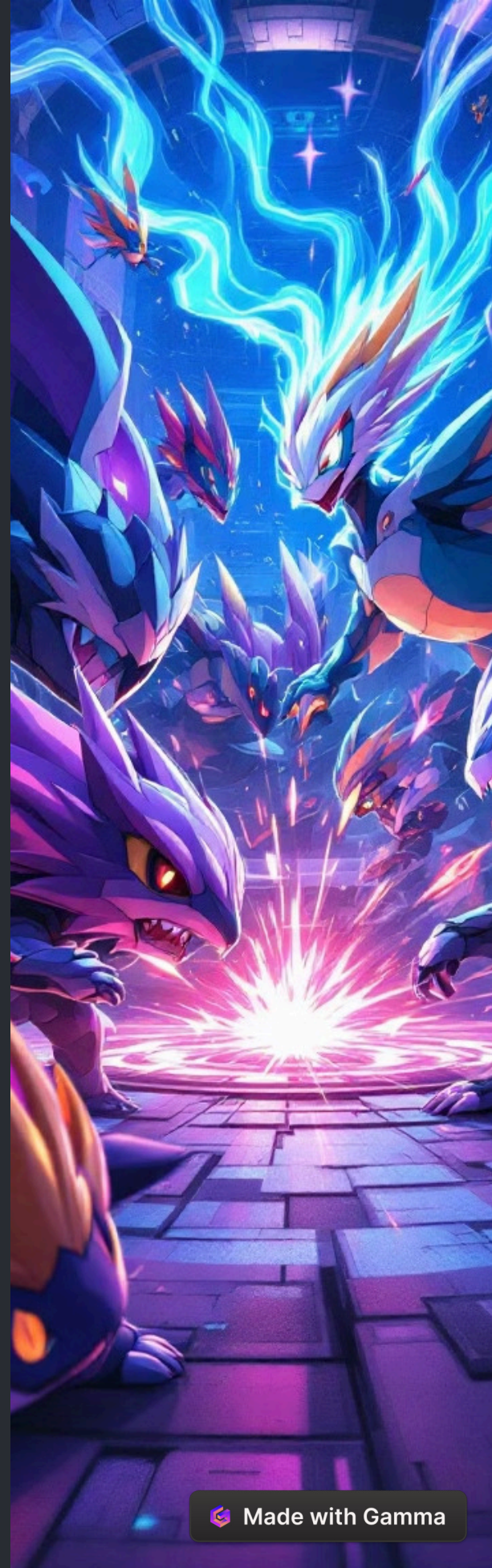
Atribuir Valores

pikachu.nome = "pikachu"

pikachu.hp = 200

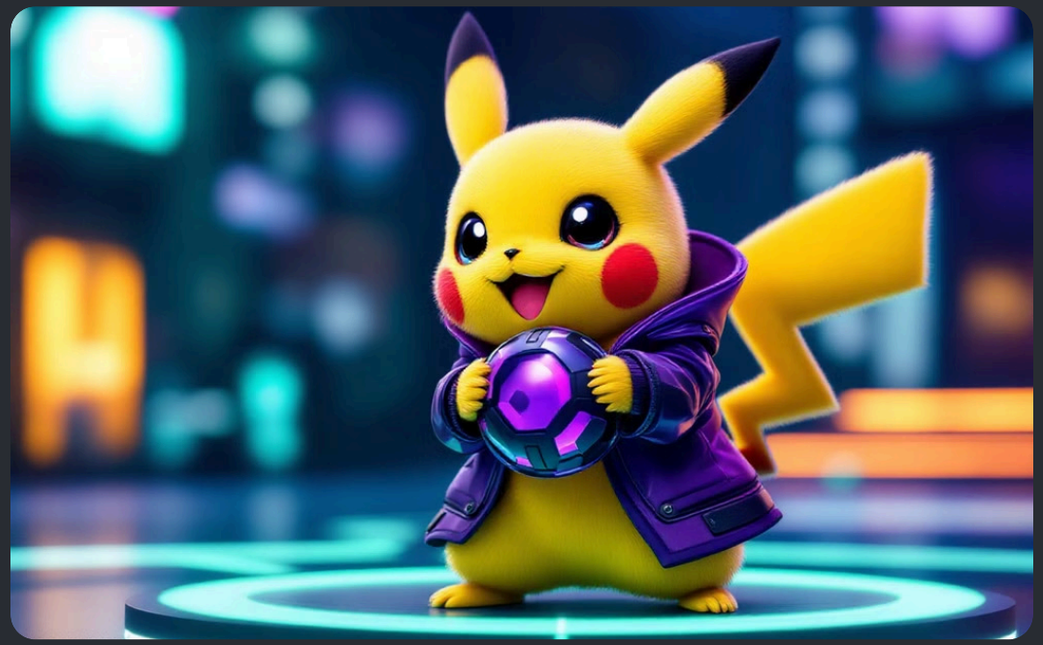
Resultado

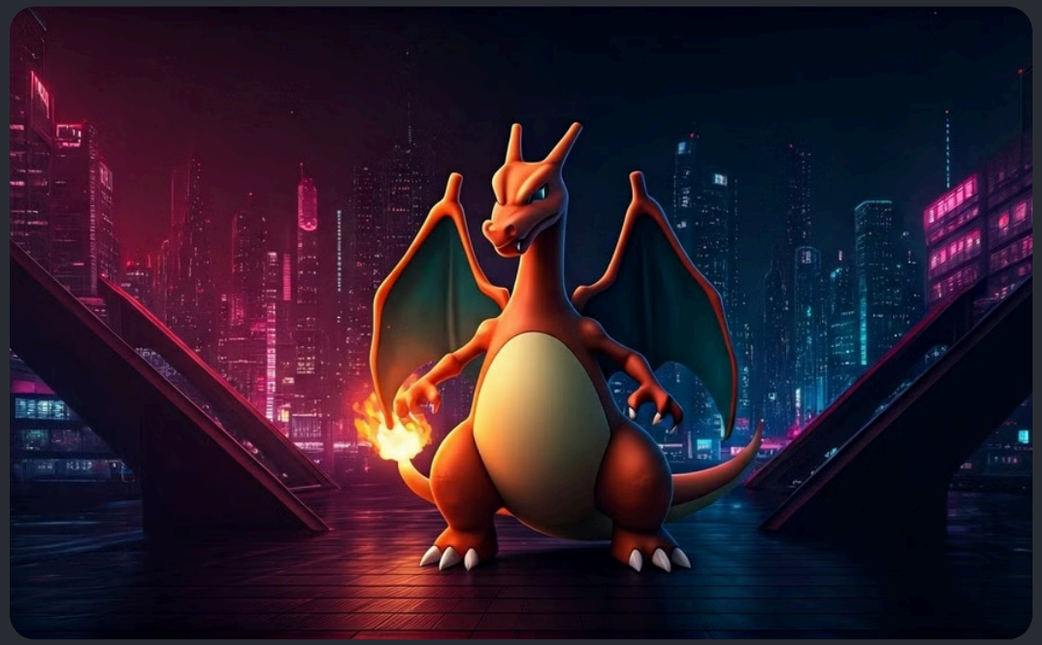
O nome do pokemon é Pikachu e ele tem 200 pontos de vida.



Múltiplos Objetos Pokémon

Criando diferentes instâncias da classe Pokemon:





Pikachu

```
var pikachu = Pokemon()
pikachu.nome = "Pikachu"
pikachu.hp = 200
```

O Pokémon Pikachu foi criado com 200 pontos de vida.

Charizard

```
var charizard = Pokemon()
charizard.nome = "Charizard"
charizard.hp = 250
```

O Pokémon Charizard foi criado com 250 pontos de vida.

Podemos criar múltiplas instâncias da classe Pokemon, cada uma com suas próprias propriedades.

Made with Gamma

Função Dentro da Classe

```swift
// Criando a classe Pokemon
class Pokemon {
    var nome: String
    var tipo: String
    var ataque: Int

    // Inicializador
    init(nome: String, tipo: String, ataque: Int) {
        self.nome = nome
        self.tipo = tipo
        self.ataque = ataque
    }

    // Método para atacar outro Pokémon
    func atacar(alvo: Pokemon) {
        print("\(nome) usou um ataque do tipo \(tipo) contra \(alvo.nome)
            causando \(ataque) de dano!")
    }
}

// Criando instâncias de Pokémon
let pikachu = Pokemon(nome: "Pikachu", tipo: "Elétrico", ataque: 50)
let charmander = Pokemon(nome: "Charmander", tipo: "Fogo", ataque: 40)

// Pikachu atacando Charmander
pikachu.atacar(alvo: charmander)
```

O que acontece aqui?

- Criamos uma classe `Pokemon` com os atributos `nome`, `tipo` e `ataque`.
- Criamos um **método** `atacar(alvo:)` que imprime uma mensagem simulando um ataque.
- Instanciamos dois Pokémon (`pikachu` e `charmander`).
- Chamamos o método `atacar()` para simular o ataque de Pikachu contra Charmander.

1 Funções dentro de Classes

Encapsulam comportamentos específicos

2 Acesso a Propriedades

Permitem que objetos acessem e modifiquem dados internos

3 Interação com Dados

Objetos podem realizar ações usando suas propriedades

Made with Gamma

Inicializador Swift

1

Inicialização

Inicializa valores da classe

2

Sintaxe

init(parâmetros) { código }

3

Palavra-chave self

Refere-se ao objeto atual

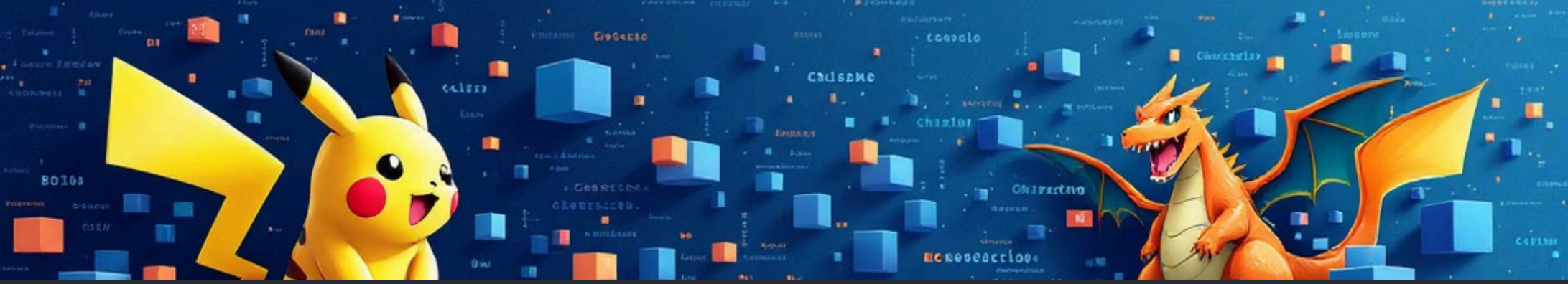
4

Uso

Atribui valores às propriedades

Se usarmos um inicializador, precisamos passar os valores correspondentes durante a criação do objeto.

Made with Gamma



Exemplo: Inicializador Swift

```swift
// Criando a classe Pokemon
class Pokemon {

    // Propriedades sem valores padrão
    var nome: String
    var tipo: String
    var nivel: Int

    // Inicializador personalizado
    init(nome: String, tipo: String, nivel: Int) {
        self.nome = nome
        self.tipo = tipo
        self.nivel = nivel
    }
}

// Criando um objeto da classe Pokemon
var pikachu = Pokemon(nome: "Pikachu", tipo: "Elétrico", nivel: 5)

// Exibindo os valores do objeto
print("Nome: \(pikachu.nome), Tipo: \(pikachu.tipo), Nível: \(pikachu.nivel)"
    )
```

O que acontece aqui?

1. Criamos a classe `Pokemon` com as propriedades `nome`, `tipo` e `nivel`.
2. Definimos um **inicializador personalizado (`init`)** para atribuir valores a essas propriedades.
3. Criamos um objeto `pikachu` passando os valores ao inicializador.
4. Exibimos os valores do objeto com `print()`.

Saída esperada:

Nome: Pikachu, Tipo: Elétrico, Nível: 5

Made with Gamma

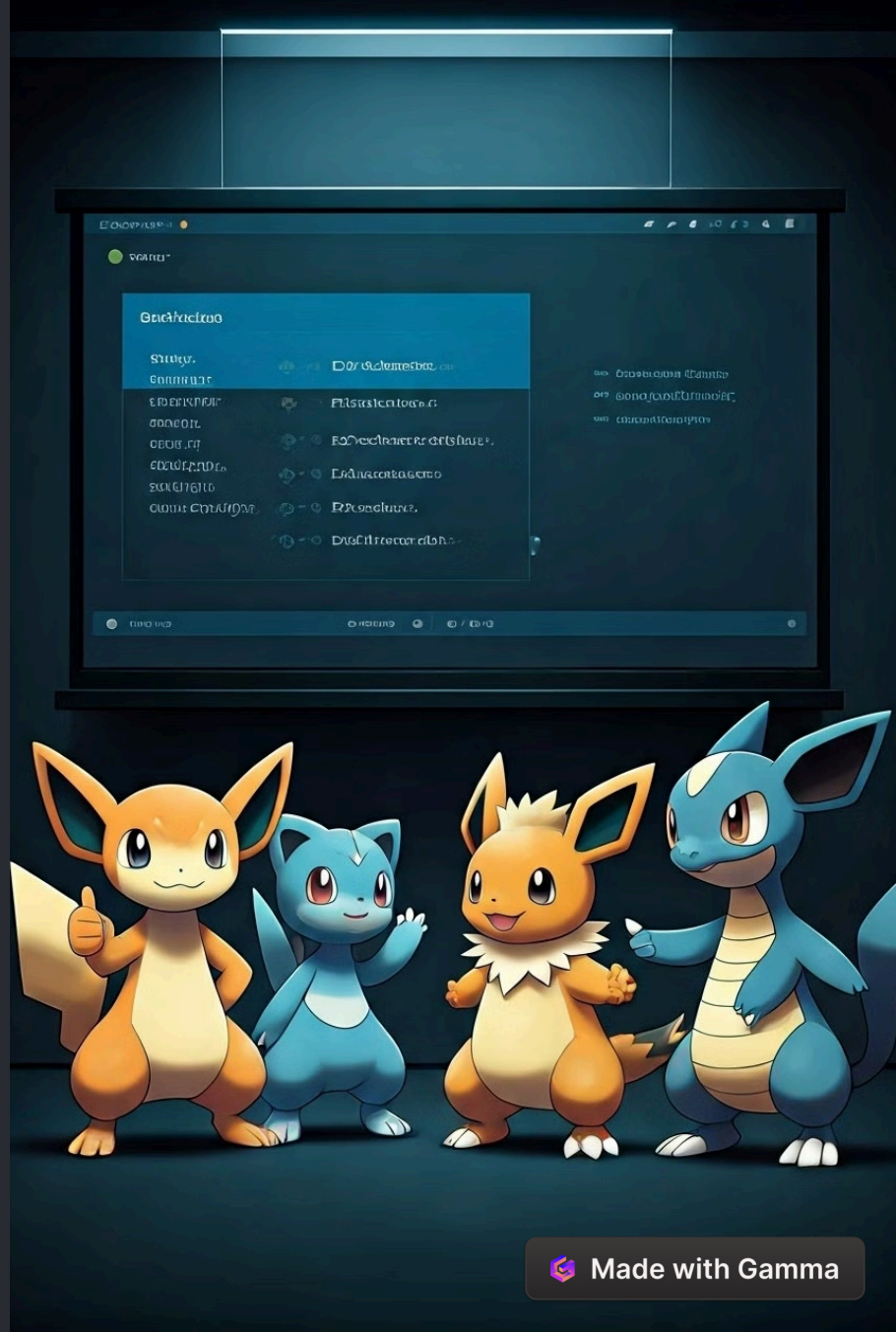
Obrigado!

Obrigado por acompanhar esta apresentação sobre Classes e Objetos Swift com Pokémon!

Esperamos que tenha sido útil e informativo.

Agora, capture seus próprios Pokémon e comece a programar!

Referência: https://www-programiz-com.translate.goog/swift-programming/classes-objects?_x_tr_sl=en&_x_tr_tl=pt&_x_tr_hl=pt&_x_tr_pto=tc



Made with Gamma