

VIDA DE PROGRAMADOR

.COM.BR

/" HISTÓRIA REAL
ENVIADA POR
BETO RAPOSA "/



#134

NÃO ENTENDO O QUE VOCÊ
ESTÁ FAZENDO COM TODO ESSE
CÓDIGO NA TELA. PRA MIM É
GREGO. ENTÃO QUERO QUE
VOCÊ ME EXPLIQUE TUDO O
QUE VOCÊ ESTÁ
FAZENDO SEMPRE...



PARA QUE EU SAIBA QUE VOCÊ
NÃO ESTÁ TRABALHANDO EM
OUTRA COISA FORA
O NOSSO PROJETO

OK. ESTOU
IMPLEMENTANDO
UMA CLASSE
PARA GERENCIAR O...



NÃO, NÃO... NÃO PRECISA DE
TANTOS DETALHES... SÓ DIGA
QUE ESTÁ TRABALHANDO NO
PROJETO SEMPRE QUE ESTIVER
TRABALHANDO
NO PROJETO.





Aula 07

Classes e Construtores

Classes

A declaração da classe consiste no nome da classe, no cabeçalho da classe podemos especificar seus parâmetros e tipo no construtor principal e etc. E no corpo da classe, cercado por chaves podemos declarar funções.

O cabeçalho e o corpo são opcionais; se a classe não tiver corpo, os chavetas podem ser omitidas.

Classes

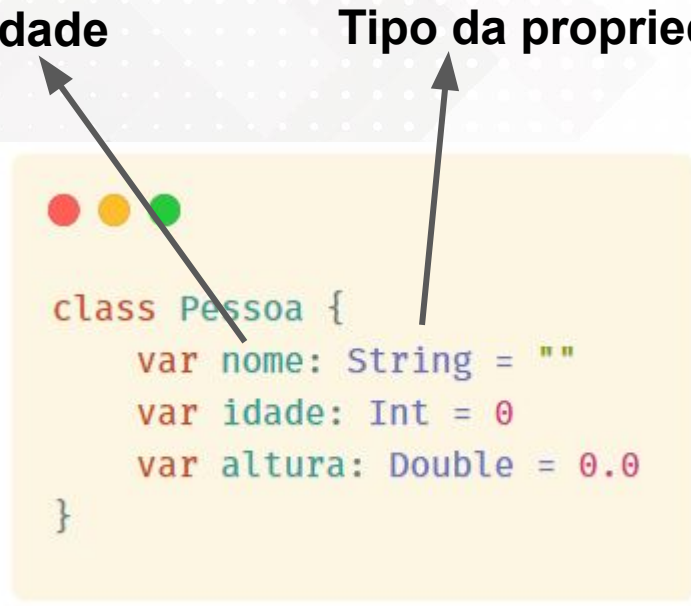


```
class Pessoa { /* ... */ }
```

Propriedades

Nome da propriedade

Tipo da propriedade



```
class Pessoa {  
    var nome: String = ""  
    var idade: Int = 0  
    var altura: Double = 0.0  
}
```

The diagram illustrates the components of a class property declaration. An arrow points from the text 'Nome da propriedade' to the variable name 'nome' in the code. Another arrow points from the text 'Tipo da propriedade' to the type 'String' in the same line. The code is presented in a light yellow box with a window-like header (red, yellow, green circles).

Construtores

Uma classe no Kotlin pode ter um construtor primário e um ou mais construtores secundários . O construtor principal faz parte do cabeçalho da classe: segue o nome da classe (e os parâmetros de tipo opcionais).



```
class Pessoa constructor(var nome: String) { /* .. */ }
```


Construtores

Se o construtor principal não tiver anotações ou modificadores de visibilidade, a palavra-chave do construtor poderá ser omitida:



```
class Pessoa (var nome: String) { /* .. */ }
```

Construtores

Os parâmetros do construtor primário podem ser usados nos blocos inicializadores. Eles também podem ser usados em inicializadores de propriedades declarados no corpo da classe:

```
class Pessoa(val nome: String, val sobrenome: String) {  
    val nomeCompleto = nome + sobrenome  
}
```


Bloco de inicialização

O construtor primário não pode conter nenhum código. O código de inicialização pode ser colocado em blocos inicializadores , que são prefixados com a palavra-chave **init** .

```
class Pessoa(val nome: String, val sobrenome: String) {  
    val nomeCompleto: String  
  
    init {  
        nomeCompleto = nome + sobrenome  
    }  
}
```

Construtores

Além disso as propriedades declaradas no construtor primário podem ser mutáveis (var) ou somente leitura (val).




```
class Pessoa(val nome: String, val idade: Int) { /* .. */ }
```


Construtor secundário

O construtor secundário é declarado a partir da palavra **constructor** o construtor secundário precisará delegar ao construtor principal, direta ou indiretamente, por meio de outro (s) construtor (es) secundário (s). A delegação para outro construtor da mesma classe é feita usando a palavra-chave **this**:

```
class Pessoa(val nome: String, val sobrenome: String) {  
    val nomeCompleto: String  
    var idade: Int = 0  
  
    init {  
        nomeCompleto = nome + sobrenome  
    }  
  
    constructor(nome: String, sobrenome: String, idade: Int): this (nome, sobrenome) {  
        this.idade = idade  
    }  
}
```



Para criar uma instância de uma classe, chamamos o construtor como se fosse uma função regular:



```
var pessoa = Pessoa("Felipe", "Medeiros")
```

Exemplo prático...



Exercícios

