

# PROGRAMAÇÃO EM JAVA

## MÓDULO 3

### MÉTODO CONSTRUTOR

Na linguagem Java, todas as vezes que instanciamos uma classe, isto é, criamos um objeto, precisamos usar a palavra reservada “new” para informar ao compilador que estamos criando um NOVO (new) objeto baseado naquela classe. Mas, toda classe possui um método construtor, que fica escondido, você não é obrigado a usá-lo, mas ele pode ajudar muito no controle do programa. O método construtor SEMPRE tem o mesmo nome da classe e SEMPRE é executado quando um novo objeto é instanciado.

- Mas se eu não escrevi nada minha classe nem declarou o método construtor, o que ele realmente executa?

**Resposta:** Se você não declarou o método construtor, o compilador Java sabe que ele existe e sabe que ele está vazio, por isso, neste caso não executa nada.

- Como então eu posso usar o método construtor para me ajudar a controlar melhor meu código?

**Resposta:** Declarando este método na sua classe!

Veja como você pode declarar e utilizar o método construtor, conforme os seguintes passos:

- Abra o código da sua classe Missao4\_Conta\_Corrente. Ele deve estar como mostrado na figura 9. Vou repeti-la aqui abaixo para facilitar a explicação:

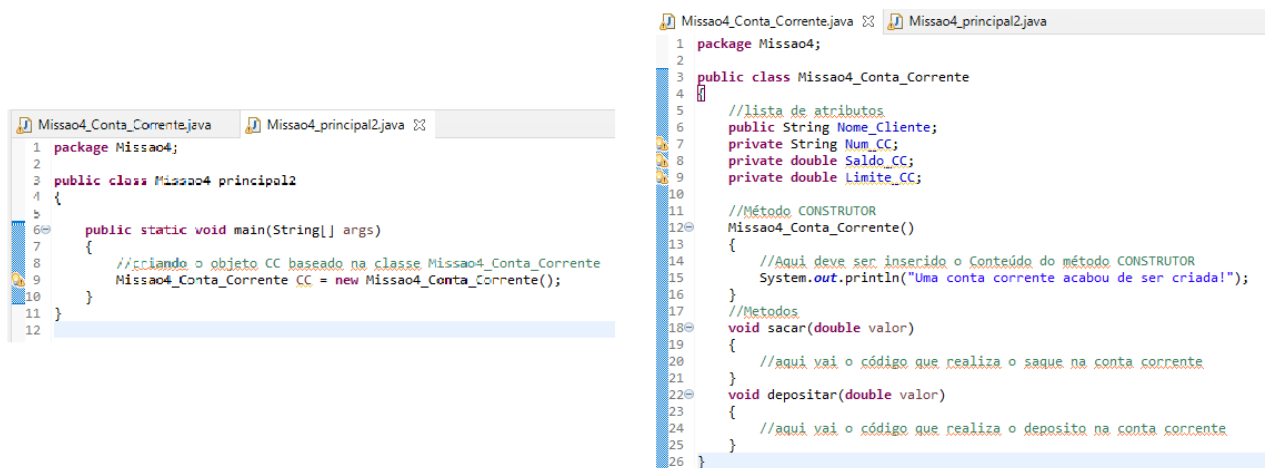
```
*Missao4_Conta_Corrente.java
1 package Missao4;
2
3 public class Missao4_Conta_Corrente
4 {
5     //lista de atributos
6     private String Nome_Cliente;
7     private String Num_CC;
8     private double Saldo_CC;
9     private double Limite_CC;
10
11     //Metodos
12     void sacar(double valor)
13     {
14         //aqui vai o código que realiza o saque na conta corrente
15     }
16     void depositar(double valor)
17     {
18         //aqui vai o código que realiza o deposito na conta corrente
19     }
20 }
```

Figura 14 - Cópia da figura 9

- Clique no início da linha 10 e pressione ENTER;
- Digite as seguintes linhas:

```
//Método CONSTRUTOR
Missao4_Conta_Corrente()
{
    //Aqui deve ser inserido o Conteúdo do método CONSTRUTOR
    System.out.println("Uma conta corrente acabou de ser criada!");
}
```

- Perceba que o método Construtor tem o mesmo nome da classe, como já foi dito e precisa ter o abre e fecha parênteses no final. Como não tem nada dentro dos parênteses você já sabe que este método não está recebendo parâmetro algum, diferente do que está a acontecer com os métodos sacar e depositar.
- Suas classes Missao4\_principal2 (à esquerda) e Missao4\_Conta\_Corrente (à direita) devem estar como mostrado conforme a figura abaixo:



The figure shows two side-by-side code editors. The left editor displays the `Missao4_principal2.java` file with the following code:

```
1 package Missao4;
2
3 public class Missao4_principal2
4 {
5
6     public static void main(String[] args)
7     {
8         //criando o objeto CC baseado na classe Missao4_Conta_Corrente
9         Missao4_Conta_Corrente CC = new Missao4_Conta_Corrente();
10    }
11 }
12
```

The right editor displays the `Missao4_Conta_Corrente.java` file with the following code:

```
1 package Missao4;
2
3 public class Missao4_Conta_Corrente
4 {
5     //lista de atributos
6     public String Nome_Cliente;
7     private String Num_CC;
8     private double Saldo_CC;
9     private double Limite_CC;
10
11     //Método CONSTRUTOR
12     Missao4_Conta_Corrente()
13     {
14         //Aqui deve ser inserido o Conteúdo do método CONSTRUTOR
15         System.out.println("Uma conta corrente acabou de ser criada!");
16     }
17     //Metodos
18     void sacar(double valor)
19     {
20         //aqui vai o código que realiza o saque na conta corrente
21     }
22     void depositar(double valor)
23     {
24         //aqui vai o código que realiza o deposito na conta corrente
25     }
26 }
```

Figura 15 - Classes Missão4-Princila2 e Missao4\_Conta\_Corrente

- Perceba que quando você executar este projeto, o compilador vai começar a executar pela classe que possui o método MAIN (`Missao4_principal2`). Quando chegar na linha 9 desta classe, o compilador vai instanciar a classe `Missao4_Conta_Corrente`, executar o método construtor desta classe `Missao4_Conta_Corrente` e criar o objeto CC.
- E, para provar que o método construtor é executado automaticamente, sem precisar ser chamado, é que o resultado que será apresentado no Console do Ide vai ser: "Uma conta corrente acabou de ser criada! ".
- Salve, execute o projeto para verificar o resultado.

## Palavra reservada “This”

Este termo é uma palavra reservada do Java. Ela é usada para informar ao compilador que os atributos que vem depois do “ponto”, por exemplo: **this.Nome\_Cliente** são atributos da própria classe em que você está.

Mas, por que é importante usar a palavra reservada “this”?

Quando se desenvolve sistemas mais complexos e grandes, pode acontecer que duas classes tenham o mesmo nome de atributo.

### CHAMADA DESAFIO EXTRA

Caso queira aprimorar mais seus conhecimentos para passar para o próximo conteúdo, acesse a atividade **Desafio extra - This**, no PDF de Desafios extras ao final dessa mídia.

## Métodos get e set

- **get** significa **obter** ou **pegar**, isto é, retorna o valor do atributo;
- **set** significa **definir**, isto é, alterar o valor do atributo;

Essas funções são muito utilizadas em programação orientada a objeto, pois, como você já sabe, a maioria das classes declara seus atributos como private.

Então:

- Para termos acesso ao saldo de uma conta, temos que criar o método **getSaldo\_CC()** dentro da classe **Missao4\_Conta\_Corrente** e depois chamar este método dentro da classe **Missao4\_principal2**;
- Para termos acesso ao Num\_CC de uma conta, temos que criar o método **getNum\_CC()** dentro da classe **Missao4\_Conta\_Corrente** e depois chamar este método dentro da classe **Missao4\_principal2**;
- E assim por diante.

Na prática, siga os seguintes passos:

- Abra o código da classe **Missao4\_Conta\_Corrente** e insira o seguinte método logo abaixo da linha 10:

```
public double getSaldo_CC()
{
    return this.Saldo_CC;
}
```

- Salve o código;
- Abra o código da classe **Missao4\_principal2** e faça as seguintes alterações:

```

public static void main(String[] args)
{
    double Saldo_Atual;
    //criando o objeto CC baseado na classe Missao4_Conta_Corrente
    Missao4_Conta_Corrente CC = new Missao4_Conta_Corrente("Paula", 1250);
    Saldo_Atual = CC.getSaldo_CC();
    System.out.println("O Saldo atual da conta é:R$ "+ Saldo_Atual);
    CC.depositar(200.50);
    Saldo_Atual = CC.getSaldo_CC();
    System.out.println("O Saldo atual da conta é:R$ "+ Saldo_Atual);
    CC.sacar(50.25);
    Saldo_Atual = CC.getSaldo_CC();
    System.out.println("O Saldo atual da conta é:R$ "+ Saldo_Atual);
}

```

- OBS: As linhas a serem inseridas estão destacadas em amarelo
- Salve, execute o projeto e verifique o resultado.
- Apague as linhas de código (que mostram o saldo no console) dos métodos sacar e depositar.
- O resultado deve ser o seguinte:

Uma conta corrente acabou de ser criada para o(a) Sr(a) Paula  
 O valor do Limite desta conta foi definido como 1250.0 reais  
 O Saldo atual da conta é:R\$ 0.0  
 O Saldo atual da conta é:R\$ 200.5  
 O Saldo atual da conta é:R\$ 150.25

## CHAMADA DESAFIO EXTRA

Caso queira aprimorar mais seus conhecimentos para passar para o próximo conteúdo, acesse a atividade **Desafio extra – Get e Set**, no PDF de Desafios extras ao final dessa mídia.