# **L**ógica de **P**rogramação

**Unidade 3** - Métodos


**QI ESCOLAS E FACULDADES**
Curso Técnico em Informática

## SUMÁRIO

# MÉTODOS

Métodos são blocos de código que pertencem a uma classe e tem por finalidade realizar uma tarefa. Ou seja, são as ações, comportamentos que nossos objetos poderão ter. Métodos podem ou não alterar o estado (dados) de um objeto.

## *Tipos de Métodos*

### *Métodos de ação (sem retorno) - void*

Apenas realizam a ação sem dar nenhum resultado. A palavra ***void*** significa ausência de retorno. **Exemplo:** sentar(), levantar(), etc.

### *Métodos de retorno – double, int, byte, String, boolean*

Realizam a ação e ao final retornam um valor de resposta. **Exemplo:** verificarStatus(), calcularMedia(), etc.

*Os métodos podem possuir ou não argumentos.*

## *Argumentos*

### *Métodos com ou sem argumento*

São os dados adicionais que o método requer para realizar a sua tarefa. Por exemplo, para uma pessoa realizar a ação “andar”, é necessário informar a quantidade de passos e a direção. Para uma pessoa girar, é necessário indicar a direção e os graus. Em alguns casos o método pode não ter argumento. Nesse caso, apenas utilizamos um conjunto vazio de parênteses.

Para cada argumento devemos especificar o tipo de dado, assim como nos atributos. Ex: andar(passos:int, direcao:String).

**Exemplo:** Imagine o seguinte: Se eu falar para uma pessoa sentar, não preciso falar mais nada, isso basta, ela sabe que é para sentar-se. Agora se eu falar para andar, a pessoa provavelmente perguntará para onde. A direção seria o argumento, para que a pessoa execute a ação, ela precisa de mais informações além da própria ação imposta.

### *Passagem por Parâmetro*

A passagem por parâmetro, nada mais é, do que o argumento do método, ou seja, o que deverá ser informado ao método para que o mesmo seja executado.

**Exemplo: andar(direcao:String,passos:int):void**

O exemplo acima é um método sem retorno e com argumento, temos dois parâmetros: direcao e passos.

## *Exemplos de Método no Diagrama*

**Exemplo 1:**

| ContaBancaria |
|---|
| numero:int<br>agencia:int<br>conta:String<br>titular:String<br>saldo:double |
| sacar(valor:double):void<br>depositar(valor:double):void<br>obterSaldo(): double |

Observando a classe acima, percebemos que temos dois métodos sem retorno e com argumentos e um método com retorno e sem argumentos.

**sacar(valor:double):void** → Método sem retorno, pois o mesmo irá apenas subtrair do saldo da conta o valor pedido, não retornará nenhum resultado. O argumento é o valor que o usuário deverá informar para que seja feito o saque. Este método modifica o estado do objeto, pois irá alterar o valor do atributo "saldo".

**depositar(valor:double):void** → Método sem retorno, pois o mesmo irá apenas adicionar no saldo da conta o valor pedido para depositar, não retornará nenhum resultado. O argumento é: valor. Este método modifica o estado do objeto, pois irá alterar o valor do atributo "saldo".

**obterSaldo():double** → Método com retorno, pois o mesmo buscará na memória o valor que temos em saldo. E sem argumento, pois não precisamos informar nada além dos atributos que já temos na classe.

**Exemplo 2:**

| Numero |
|---|
| valor: int |
| calcularDobroDoNumero():int |

| somar2AoNumero():int<br>multiplicarNumeroPor3():int<br>somarOutroValorAoNumero(valor:double):int |
|---|

Observando a classe acima, vamos responder.

1. Qual o nome da classe?
2. Quantos atributos a classe possui? E quais são?
3. Quantos métodos a classe possui?
4. Quantos métodos possuem retorno? E quais são?
5. Quantos métodos não possuem retorno? E quais são?
6. Cite um método que possua argumento.
7. Qual o parâmetro passado pelo método que possui argumento?
8. Temos algum método sem retorno e com argumento? Qual?

Respostas:

1. Numero
2. Um atributo: valor.
3. Quatro métodos.
4. Quatro métodos com retorno: calcularDobroDoNumero():int; somar2AoNumero():int; multiplicarNumeroPor3():int; somarOutroValorAoNumero(valor:double):int
5. Nenhum
6. somarOutroValorAoNumero(valor:double):int
7. valor
8. Não, pois nenhum termina com void

## *Como os métodos afetam os objetos*

Os métodos ficam à disposição para uso na classe; porém, eles só executam quando são acionados, invocados. Vamos observar o exemplo.

| ContaBancaria |
|---|
| numeroDaConta:String<br>saldo:double |
| sacar(valor:double):void<br>depositar(valor:double):void<br>obterSaldo(): double |

Considere que temos duas contas correntes em determinado banco, portanto, dois objetos da classe ContaBancaria.

***conta1****(numeroDaConta=123-0, saldo=100)*
***conta2****(numeroDaConta=321-1, saldo=500)*

Vamos imaginar que durante o dia, ocorreram as seguintes operações bancárias:

```
conta1.sacar(50)
conta2.depositar(200)
conta1.depositar(400)
conta2.sacar(150)
conta1.sacar(20)
conta2.depositar(120)
```

Quando utilizamos métodos como sacar ou depositar, sabemos que o saldo da conta é alterado, pois quando sacamos estamos retirando dinheiro da conta, e quando depositamos estamos acrescentando dinheiro na conta. Se inicialmente a conta1 possuía um saldo de 100, e posteriormente teve um saque de 50, um depósito de 400 e outro saque de 20, seu novo saldo será:

*100 – 50 + 400 – 20 =* ***430***

E a conta2, que tinha inicialmente um saldo de 500, teve um depósito de 200, depois um saque de 150 e outro depósito de 120, seu novo saldo será:

*500 + 200 – 150 + 120 =* ***670***

Assim, ao final do dia, o estado de cada objeto seria:

***conta1****(numeroDaConta=123-0, saldo=430)*
***conta2****(numeroDaConta=321-1, saldo=670)*