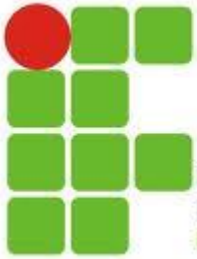


Aula 13
Métodos

Disciplina: Programação Orientada a Objetos

Professora: Alba Lopes

alba.lopes@ifrn.edu.br

Métodos

- **Parâmetros**
  - Em Java, os métodos podem possuir ou não parâmetros
  - Na aula anterior, construímos classes que possuem métodos sem parâmetros
  - Agora, vamos estudar como construir métodos que utilizam parâmetros
- Os parâmetros são utilizados para alterar os valores dos atributos ou o comportamento dos métodos durante sua execução

Métodos

- **No exemplo da classe Automovel abaixo:**

```
public class Automovel {

    //ATRIBUTOS
    String marca;
    String cor;
    int velocidade = 0;

    //MÉTODOS
    void buzinar(){
       System.out.println("BEEEEPPPP...");
    }
    void acelerar(){
        velocidade = velocidade + 1;
    }
    void reduzir(){
        velocidade = velocidade - 1;
    }

}
```

Se nos métodos acelerar e reduzir quiséssemos variar o valor do incremento de acordo com a execução e não deixar amarrado em incrementar apenas em 1 unidade?

Podemos utilizar parâmetros nos métodos!

Parâmetros

- Inserindo parâmetros nos métodos
  - Um método pode possuir um ou mais parâmetros
    - Se o método possuir mais de um parâmetro, deve-se separá-los por vírgulas

```
<tipo_do_param1> <nome_do_param1>, <tipo_do_param2> <nome_do_param2>,...
```

  - Exemplo: se quiséssemos definir em quanto a velocidade aumentou ao acelerar, poderíamos definir o método acelerar da seguinte forma:

```
void acelerar ( int valor ){
        velocidade = velocidade + valor;
}
```

Parâmetros

```
public class Automovel {
    //ATRIBUTOS
    String marca;
    String cor;
    int velocidade;

    void buzinar(){
        System.out.println("BEEEEEP...");
    }
    void acelerar(int valor){
        velocidade = velocidade + valor;
    }

    void reduzir(int valor){
        velocidade = velocidade - valor;
    }

}
```

PARÂMETROS

```
public class Automovel {
    //ATRIBUTOS
    String marca;
    String cor;
    int velocidade;

    void buzinar(){
        System.out.println("BEEEEEP...");
    }
    void acelerar(int valor){
        velocidade = velocidade + valor;
    }

    void reduzir(int valor){
        velocidade = velocidade - valor;
    }

}
```

PARÂMETROS

Parâmetros

- Ao instanciar um objeto da classe e chamar um método que possui parâmetro, a chamada do método deve incluir o valor do parâmetro passado:

```
public static void main(String [] args){
    Automovel meuCarro = new Automovel();
    meuCarro.velocidade = 0;
    meuCarro.marca = "Fiat";
    meuCarro.cor = "Branco";

    meuCarro.acelerar(10);
    meuCarro.acelerar(20);
    System.out.println("Velocidade atual: " + meuCarro.velocidade);
    meuCarro.reduzir(5);
    System.out.println("Velocidade atual: " + meuCarro.velocidade);

}
```

Parâmetros

- Um método pode possuir mais de um parâmetro:

```
public class Calculadora {

    void Soma(int a, int b){
        int resultado = a + b;
        System.out.println("O resultado da soma é: " + resultado );
    }

    void Subtracao(int a, int b){
        int resultado = a - b;
        System.out.println("O resultado da subtração é: " + resultado );
    }

    void Multiplicacao(int a, int b){
        int resultado = a * b;
        System.out.println("O resultado da multiplicação é: " + resultado ) ;
    }

    void Divisao(int a, int b){
        int resultado = a / b;
        System.out.println("O resultado da divisão é: " + resultado );
    }
}
```

Parâmetros

- Um método pode possuir mais de um parâmetro:

```
public class Calculadora {

    void Soma(int a, int b){
        int resultado = a + b;
        System.out.println("O resultado da soma é: " + resultado );
    }

    void Subtracao(int a, int b){
        int resultado = a - b;
        System.out.println("O resultado da subtração é: " + resultado );
    }

    void Multiplicacao(int a, int b){
        int resultado = a * b;
        System.out.println("O resultado da multiplicação é: " + resultado ) ;
    }

    void Divisao(int a, int b){
        int resultado = a / b;
        System.out.println("O resultado da divisão é: " + resultado );
    }
}
```

Parâmetros

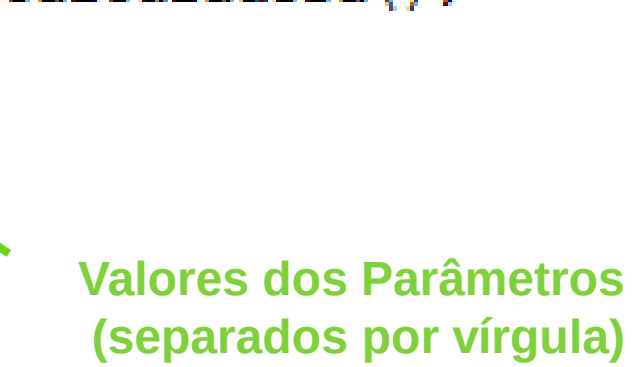
- Chamada de método com mais de um parâmetro:

```
public static void main(String [] args){
     Calculadora calc = new Calculadora();
     calc.Soma(10,5);
}
```

Parâmetros

- Chamada de método com mais de um parâmetro:

```
public static void main(String [] args){
     Calculadora calc = new Calculadora();
     calc.Soma(10,5);
}
```



Retorno de Métodos

- Até o momento, só trabalhamos com métodos do tipo **void**, que não retornam nenhum valor
- Mas os métodos podem retornar algum valor para o trecho que o chamou
  - Nesse caso, deve-se informar o **tipo do retorno** (int, String, double, etc)
- Um método que retorna algum valor tem a seguinte sintaxe:

```
<tipo_de_retorno> <nome_do_método> (<lista_de_parâmetros>){
        <operações>
        return <valor_de_retorno>;
}
```

Retorno de Métodos

- Na classe Calculadora, do exemplo anterior, os métodos podem ser construídos retornando um valor:

```
public class Calculadora {

    int Soma(int a, int b){
        int resultado = a + b;
        return resultado;
    }
```

Retorno de Métodos

- Na classe Calculadora, do exemplo anterior, os métodos podem ser construídos retornando um valor:

```
public class Calculadora {

    int Soma(int a, int b){
        int resultado = a + b;
        return resultado;
    }
```

Tipo de retorno do método

Palavra **return** e valor a ser retornado

Retorno de Métodos

- Utilização de um método que retorna um valor:

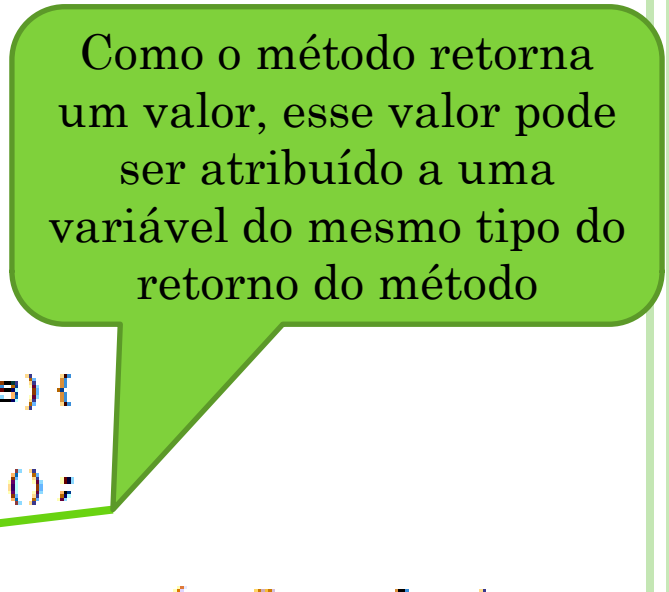
```
public static void main(String [] args){

    Calculadora calc = new Calculadora();
    int valor = calc.Soma(10,5);
    System.out.println("O resultado da soma é: "+ valor);

}
```

Retorno de Métodos

- Utilização de um método que retorna um valor:




```
public static void main(String [] args){

    Calculadora calc = new Calculadora();
    int valor = calc.Soma(10,5);
    System.out.println("O resultado da soma é: "+ valor);

}
```

Retorno de Métodos

- Outro exemplo:
  - Classe Data, do exercício da aula anterior. Novo método retornarNomeDoMes. O método não possui parâmetros, mas retorna um valor do tipo String.
    - **Obs1:** mes *é um atributo da classe Data.*
    - **Obs2:** *o método está incompleto para facilitar a visualização. Deve ser expandido para abranger até o mês de dezembro.*

```
String retornarNomeDoMes(){
    String nomeDoMes = "";
    switch(mes){
        case 1:
            nomeDoMes = "Janeiro";
            break;
        case 2:
            nomeDoMes = "Fevereiro";
            break;
        case 3:
            nomeDoMes = "Março";
            break;
        case 4:
            nomeDoMes = "Abril";
            break;
        case 5:
            nomeDoMes = "Maio";
            break;
        case 6:
            nomeDoMes = "Junho";
            break;
    }
    return nomeDoMes;
}
```

Exercícios

ContaCorrente
+ saldo: float
+ definirSaldoInicial(float) : void + depositar(float) : void + sacar(float): boolean

1. Crie uma classe ContaCorrente que obedeça à descrição abaixo:
   - A classe possui o atributo **saldo** do tipo float e os métodos **definirSaldoInicial**, **depositar** e **sacar**.
   - O método **definirSaldoInicial** deve atribuir o valor passado por parâmetro ao atribuito **saldo**
   - O método **depositar**, deve adicionar o valor passado por parâmetro ao atributo saldo
   - O método **sacar** deve reduzir o valor passado por parâmetro do salto já existente
     - Perceba que é necessário verificar a condição de o valor do saldo ser insuficiente para o saque que se deseja fazer.
     - O valor de retorno deve ser *true* (verdadeiro) quando for possível realizar o saque e *false* (falso) quando não for possível

Exercícios

ContaCorrente
+ saldo: float
+ definirSaldoInicial(float) : void + depositar(float) : void + sacar(float): boolean

1. (continuação)

- Crie um objeto novaConta do tipo ContaCorrente.
- Chame o método definirSaldoInicial passando o valor 1000 como parâmetro.
- Escreva o valor do atributo saldo
- Realize um saque de 500 reais (utilize o método sacar).
- Faça um depósito de 50 reais (utilize o método depositar)
- Escreva o valor do atributo saldo na tela.
- Realize um saque de 600 reais.
- Escreva o valor do atributo saldo na tela.

Exercícios

Funcionario
+ nome: String + sobrenome: String + horasTrabalhadas: int + valorPorHora: float
+ nomeCompleto() : String + calcularSalario() : float + incrementarHoras(int): void

2. Crie uma classe Funcionário que obedeça à descrição abaixo:
   - A classe possui os atributos **nome, sobrenome**, **horasTrabalhadas** e **valorPorHora**.
   - O método **nomeCompleto** deve retornar o atributo **nome** concatenado ao atributo **sobrenome**
   - O método **calcularSalario** faz o cálculo de quanto o funcionário irá receber no mês, multiplicando o atributo **horasTrabalhadas** pelo atributo **valorPorHora** e retornar esse valor.
   - O método de **incrementarHoras** adiciona um valor passado por parâmetro ao valor já existente no atributo valorPorHora.

Exercícios

Funcionario
+ nome: String + sobrenome: String + horasTrabalhadas: int + valorPorHora: float
+ nomeCompleto() : String + calcularSalario() : float + incrementarHoras(int): void

2. (continuação)
   - Crie o método **main** e instancie a classe Funcionário criada, criando um objeto novoFuncionario do tipo Funcionario
     - Atribua o valor “Luis” ao atributo **nome**
     - Atribua o valor “Silva” ao atributo sobrenome
     - Atribua o valor 10 ao atributo **horasTrabalhadas**
     - Atribua o valor 25.50 ao atributo **valorPorHora**
     - Escreva o nome completo do funcionário na tela, utilizando o retorno do método **nomeCompleto**
     - Escreva o valor do salario que o funcionario ira receber, utilizando o retorno do método **calcularSalario**
     - Adicione 8 ao atributo **horasTrabalhadas** utilizando o método **incrementarHoras**
     - Escreva o salario do funcionário (retorno do método **calcularSalario**)

Referências

- http://www.hardware.com.br/artigos/programacao-orientada-objetos/
- http://www.fontes.pro.br/educacional/materialpaginas/java/arquivos/jdbc/jdbc.php
- http://www.dm.ufscar.br/~waldeck/curso/java