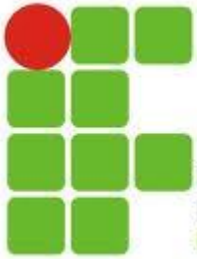


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

AULA 15
CONSTRUTORES

Disciplina: Programação Orientada a Objetos

Professora: Alba Lopes

alba.lopes@ifrn.edu.br

Construtores

- Quando usamos a palavra chave **new**, estamos construindo um objeto.

```
ContaCorrente conta = new ContaCorrente();
```

- Sempre quando o new é chamado, executa o construtor da classe.
- O construtor da classe é um bloco declarado com o mesmo nome que a classe
- Ele não possui tipo de retorno, mas pode possuir parâmetros

CONSTRUTORES

```
public class ContaCorrente {
    private float saldo;
    private Cliente clienteConta;

    //Exemplo de método construtor
    public ContaCorrente(){


    }

    public void setSaldo(float novoSaldo){
        saldo = novoSaldo;
    }
    ...
```

Construtores

- Até agora, as nossas classes não possuíam nenhum construtor. Então como é que era possível usar **new**, se todo **new** chama um construtor obrigatoriamente?
  - Quando você não declara nenhum construtor na sua classe, o Java cria um para você.
  - Esse construtor é o **construtor default**, ele não recebe nenhum parâmetro e o corpo dele é vazio
  - A partir do momento que o construtor é definido, não é possível utilizar mais o construtor **default**

Métodos Construtores

- Exemplo:

```
3   public class ContaCorrente {
4       private float saldo;
5       private Cliente clienteConta;
6
7
8       //Exemplo de método construtor
9       public ContaCorrente(){
10          System.out.println("Nova conta corrente criada.");
11
12      }
13
14      public void setSaldo(float novoSaldo){
15          saldo = novoSaldo;
16      }
17
18      public float getSaldo(){
19          return saldo;
20      }
        ...
```

Métodos Construtores

- Com o construtor definido, todo objeto criado irá executar os comandos que se encontram no corpo do método construtor.
- No caso do exemplo anterior, a partir de agora, ao instanciarmos um novo objeto, a mensagem “Nova conta corrente criada” será exibida, pois esse é o comando que definimos no nosso construtor.

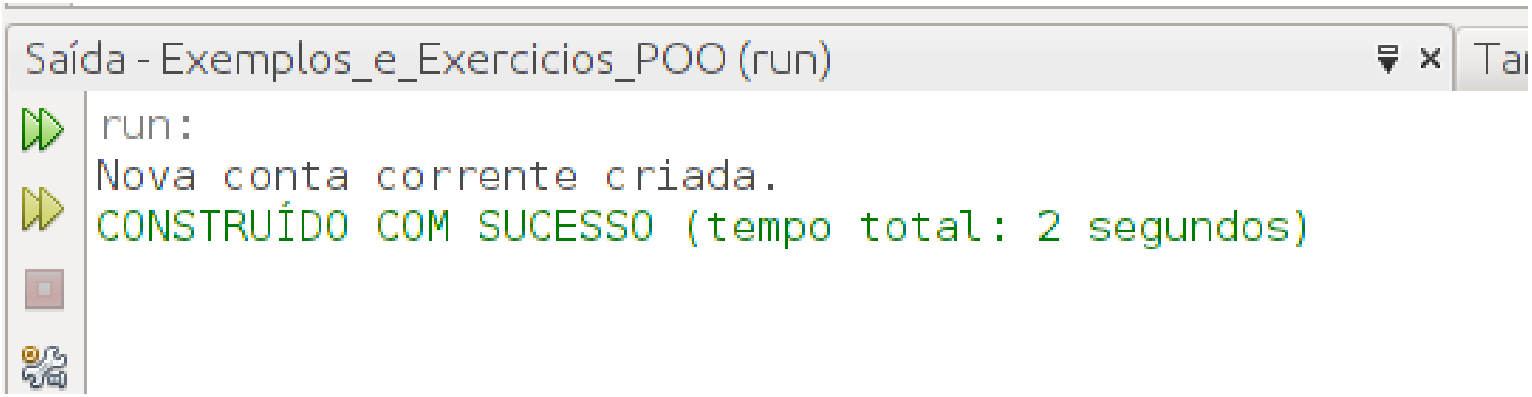
CONSTRUTORES

CÓDIGO:

```
public class TestarContaCorrente {

    public static void main(String [] args){
        ContaCorrente conta = new ContaCorrente();
    }
}
```

EXECUÇÃO:



Construtores

- É possível definir parâmetros nos métodos construtores, podendo assim inicializar algum tipo de informação:

```
public class ContaCorrente {
    private float saldo;
    private Cliente clienteConta;


    public ContaCorrente(float saldoInicial){
        System.out.println("Nova conta corrente criada.");
        saldo = saldoInicial;

    }

    ...
```

Construtores

- No exemplo anterior, a conta recebe por parâmetro o valor do saldo inicial
- Assim, ao criarmos a conta, ela já terá um valor inicial de saldo.
- Sendo assim, esse valor deve ser passado por parâmetro quando utilizarmos o **new** para criar um novo objeto.

CONSTRUTORES

```
public class TestarContaCorrente {

    public static void main(String [] args){
        ContaCorrente conta = new ContaCorrente(500);
    }
}
```

- Como o método construtor possui um parâmetro, esse parâmetro deve ser determinado na chamada do método.
- A partir de agora, se tentar criar um objeto sem passar o valor do saldo inicial por parâmetro, um erro será detectado.

Construtores

- Por que os construtores são úteis ou necessários?
  - Eles dão possibilidades ou obrigam o usuário de uma classe de passar argumentos para o objeto durante o processo de criação
  - No exemplo anterior, ao criar uma conta corrente,o valor do saldo inicial deve, necessariamente ser informado.
    - Não é possível criar a conta se esse valor não for informado

Construtores

- É possível criar mais de um construtor em uma mesma classe, entretanto, eles devem possuir assinaturas diferentes(quantidade e tipos de parâmetros diferentes)

```
public class ContaCorrente {
    private float saldo;
    private Cliente clienteConta;

    public ContaCorrente(float saldoInicial){
        System.out.println("Nova conta corrente criada.");
        saldo = saldoInicial;

    }

    public ContaCorrente(Cliente novoCliente, float saldoInicial){
        clienteConta = novoCliente;
        saldo = saldoInicial;
    }

    ...
```

Construtores

- Quando for criar um objeto, é possível escolher qual construtor utilizar:

```
public class TestarContaCorrente {

    public static void main(String [] args){
        ContaCorrente conta1 = new ContaCorrente(500);

        Cliente meuCliente = new Cliente();
        ContaCorrente conta2 = new ContaCorrente(meuCliente, 1000);
    }
}
```

Construtores

- Quando for criar um objeto, é possível escolher qual construtor utilizar:

```
public class TestarContaCorrente {

    public static void main(String [] args){
        ContaCorrente conta1 = new ContaCorrente(500);

        Cliente meuCliente = new Cliente();
        ContaCorrente conta2 = new ContaCorrente(meuCliente, 1000);
    }
}
```

Usando o construtor 1

Usando o construtor 2

Exemplo 1

- Criar o método construtor para a classe **Cliente** que receba uma String como parâmetro para definir o nome do cliente:

```
public class Cliente {

    private String nome;
    private String endereco;
    private float renda;
    private String profissao;

    public Cliente(String n){
        nome = n;
    }

    public String getNome() {
        return nome;
    }

    ...
```

Exemplo 1

- Criar o método construtor para a classe **Cliente** que receba uma String como parâmetro para definir o nome do cliente:

```
public class Cliente {

    private String nome;
    private String endereco;
    private float renda;
    private String profissao;

    public Cliente(String n){
        nome = n;
    }

    public String getNome() {
        return nome;
    }

    ...
```

Exemplo 1

- Agora, para instanciar um objeto do tipo Cliente, devemos, necessariamente passar o nome do cliente por parâmetro. Alterando o exemplo anterior, teremos agora:

```
public class TestarContaCorrente {

    public static void main(String [] args){
        ContaCorrente conta1 = new ContaCorrente(500);

        Cliente meuCliente = new Cliente("José da Silva");
        ContaCorrente conta2 = new ContaCorrente(meuCliente, 1000);
    }
}
```

Exemplo 1

- Agora, para instanciar um objeto do tipo Cliente, devemos, necessariamente passar o nome do cliente por parâmetro. Alterando o exemplo anterior, teremos agora:

```
public class TestarContaCorrente {

    public static void main(String [] args){
        ContaCorrente conta1 = new ContaCorrente(500);

        Cliente meuCliente = new Cliente("José da Silva");
        ContaCorrente conta2 = new ContaCorrente(meuCliente, 1000);
    }
}
```

Exemplo 2

- Criar o método construtor para a classe **Automovel** que possua três parâmetros para definir a velocidade, a marca e a cor :

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

    public Automovel(int v, String c, String m ){
        velocidade = v;
        cor = c;
        marca = m;
    }

    void buzinar(){
        System.out.println("BEEEEEP...");
    }
            ...
```

EXEMPLO 2

- Criar o método construtor para a classe **Automovel** que possua três parâmetros para definir a velocidade, a marca e a cor :

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

    public Automovel(int v, String c, String m ){
        velocidade = v;
        cor = c;
        marca = m;
    }

    void buzinar(){
        System.out.println("BEEEEEP...");
    }
            ...
```

Exemplo 2

- Para criar objetos da classe Automovel, devemos fazer:

```
public class TestarAutomovel {

    public static void main(String [] args){
        Automovel meuCarro = new Automovel(10, "Vermelho", "Ford");
        meuCarro.buzinar();
        System.out.println("A velocidade é: " + meuCarro.getVelocidade());
        meuCarro.setVelocidade(0);
        meuCarro.acelerar(10);
        meuCarro.acelerar(20);
        System.out.println("Velocidade atual: " + meuCarro.getVelocidade());
        meuCarro.reduzir(5);
        System.out.println("Velocidade atual: " + meuCarro.getVelocidade());

    }
}
```

Exemplo 2

- Para criar objetos da classe Automovel, devemos fazer:

```
public class TestarAutomovel {

    public static void main(String [] args){
        Automovel meuCarro = new Automovel(10, "Vermelho", "Ford");
        meuCarro.buzinar();
        System.out.println("A velocidade é: " + meuCarro.getVelocidade());
        meuCarro.setVelocidade(0);
        meuCarro.acelerar(10);
        meuCarro.acelerar(20);
        System.out.println("Velocidade atual: " + meuCarro.getVelocidade());
        meuCarro.reduzir(5);
        System.out.println("Velocidade atual: " + meuCarro.getVelocidade());

    }
}
```

Exercícios

- Faça os procedimentos abaixo antes de iniciar a resolução dos exercícios
  - Crie um novo pacote no projeto Exemplos_e_Exercicios_POO chamado aula15_exercicios.
  - Copie as classes que iremos utilizar dentro desse pacote. Se for questionado em algum momento sobre a cópia, selecione a opção Refatorar. As classes são:
    - classe Lampada (origem: pacote aula11_exercicios)
    - classe Data (origem: pacote aula13_exemplos)

EXERCÍCIOS

1. **Na classe Lâmpada seu pacote aula15_exercicios faça**:

a) Crie um construtor para a classe Lampada que receba por parâmetro um valor do tipo boolean referente ao estado da lâmpada. Atribua o valor passado por parâmetro ao atributo **acesa.**

b) Crie um outro construtor para a classe Lampada que receba por parâmetro um valor do tipo boolean referente ao estado da lâmpada e um valor int referente à potência da lâmpada. Atribua o valores passado por parâmetro ao atributo **acesa** e **potencia** respectivamente.

c) Crie uma nova classe **TestarLampada** no pacote aula15_exercicios para testar a classe criada. Nessa classe, crie um método **main** que realize as seguintes operações:

- Crie um objeto do tipo Lampada com o nome **lampadaLab1**, utilizando o construtor criado na questão **a**. Passe por parâmetro o valor **true**.
- Crie um outro objeto do tipo Lampada com o nome lampadaLab2, utilizando o construtor criado na questao **b**. Passe por parâmetro o valor **false** e o valor **20**.
- Chame o método informarSituacao do objeto **lampadaLab1**
- Chame o método informarPotencia do objeto **lampadaLab1**
- Chame o método informarSituacao do objeto **lampadaLab2**
- Chame o método informarPotencia do objeto **lampadaLab2**

d) Execute a classe TestarLampada

EXERCÍCIOS

2. **Na classe Data seu pacote aula15_exercicios faça**:

a) Crie um construtor para a classe Data que receba por parâmetro três valores inteiros referentes ao dia, mês e ano e atribua os valores passados por parâmetro aos atributos dia, mês e ano, respectivamente.

b) Defina também os atributos dia, mes e ano da classe Diata como **private** e crie os métodos **get** e **set** para cada um dos atributos.

c) Crie uma nova classe **TestarData** no pacote aula15_exercicios para testar a classe criada. Nessa classe, crie um método **main** que realize as seguintes operações:

- Crie um objeto do tipo Data com o nome **hoje**, utilizando o construtor criado na questão **a**. Passe por parâmetro o dia, o mês e o ano correspondente à data de hoje.
- Chame o método escreverAData do objeto **hoje** para mostrar a data na tela
- Crie um objeto do tipo Data com o nome **natal** e passe por parâmetro os valores correspondentes ao dia do Natal (exemplo: dia 25, mês 12, ano 2012)
- Chame o método escreverAData do objeto **natal** para mostrar a data na tela

d) Execute a classe **TestarData**.

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
- PORTAL EDUCAÇÃO - Cursos Online : Mais de 900 cursos online com certificado http://www.portaleducacao.com.br/informatica/artigos/7852/moderadores-de-acesso#ixzz2AAmxO3JD
- http://www.slideshare.net/regispires/java-08-modificadores-acesso-e-membros-de-classe-presentation