

Aula 2

Professores:

Carlos Bazílio
Isabel Rosseti

Introdução à POO em Java

Conteúdo:

- Classes e Objetos em Java
- Atributos
- Membros de Classes
- Construtores de Classes
- Métodos

Classes em Java

➡ Como representamos uma classe em Java?



Classe ClienteBanco em Java

```
public class ClienteBanco {  
  
    String nome;  
    int conta;  
    float saldo;  
  
    void RealizaSaque (float s) {  
        saldo = saldo - s;  
    }  
    float RequisitaSaldo() {  
        return saldo;  
    }  
}
```



Classes em Java

➡ Assim, temos um exemplo de fôrma (classe) para clientes de banco em Java.



Classes em Java - Criação de Instâncias

```
public class ClienteBanco {  
    String nome;  
    int conta;  
    float saldo;  
    // RealizaSaque()  
    // RequisitaSaldo()  
  
    public static void main (String arg[]) {  
  
        ClienteBanco cliente1, cliente2;  
  
        cliente1 = new ClienteBanco ("eu", 0001, 500);  
        cliente2 = new ClienteBanco ("voce", 0002, 2000);  
  
        System.out.println ("Nome do Cliente : " +  
                             cliente1.nome);  
        System.out.println ("Saldo : " + cliente1.saldo);  
    }  
}
```



Classes em Java - new



Classes em Java - Construtor



Classes em Java - Construtor

```
public class ClienteBanco {  
    String nome; int conta; float saldo;  
  
    // RealizaSaque() e RequisitaSaldo()  
  
    ClienteBanco (String pNome, int pConta, float pSaldo) {  
        nome = pNome; conta = pConta; saldo = pSaldo;  
    }  
  
    public static void main (String arg[]) {  
        ClienteBanco cliente1, cliente2;  
        cliente1 = new ClienteBanco ("eu", 0001, 500);  
        cliente2 = new ClienteBanco ("voce", 0002, 2000);  
        System.out.println ("Nome do Cliente : " +  
                             cliente1.nome);  
        System.out.println ("Saldo : " + cliente1.saldo);  
    }  
}
```



Classes em Java - Exercícios

➡ Implementar e testar a classe ClienteBanco dada.

Classes em Java (Construtor)

➡ Não é necessário que todos os valores dos atributos sejam passados para o construtor.



Classes em Java (Construtor)

➡ Uma alternativa para a implementação anterior é a seguinte:

```
ClienteBanco (String pNome, int pConta)
{
    ClienteBanco(pNome, pConta, 200);    (1)
}
```



Classes em Java (Construtores)

➡ Uma classe pode ter tantos construtores quanto necessário:

```
public class ClienteBanco {  
    String nome; int conta; float saldo;  
    // RealizaSaque() e RequisitaSaldo()  
  
    ClienteBanco (String pNome, int pConta, float pSaldo) {  
        nome = pNome; conta = pConta; saldo = pSaldo;  
    }  
  
    ClienteBanco (String pNome, int pConta) {  
        nome = pNome; conta = pConta; saldo = 200;  
    }  
    ...  
}
```



Distinção entre Construtores



Construtor Padrão



Construtores e Métodos



Distinção entre Métodos

```
public class ClienteBanco {  
    String nome; int conta; float saldo;  
    // Outros métodos  
    void RealizaDeposito (float pValor) {  
        saldo = saldo + pValor;  
    }  
    void RealizaDeposito (float pValorNota, int pQuantNotas) {  
        RealizaDeposito(pValor*pQuantNotas);  
    }  
    public static void main (String arg[]) {  
        ClienteBanco cliente1, cliente2;  
        cliente1 = new ClienteBanco ("eu", 0001, 500);  
        cliente2 = new ClienteBanco ("voce", 0002, 2000);  
        cliente1.RealizaDeposito(230);  
        cliente2.RealizaDeposito(50, 8);  
    }  
}
```

cliente1	cliente2
"eu"	"voce"
0001	0002
500	2000



Importância dos Métodos

➡ Qual a diferença entre estas 2 retiradas?

```
public class ClienteBanco {  
    String nome; int conta; float saldo;  
    void RealizaSaque (float s) {  
        saldo = saldo - s;  
    }  
    public static void main (String arg[]) {  
        ClienteBanco cliente1, cliente2;  
        cliente1 = new ClienteBanco ("eu", 0001, 500);  
        cliente1.saldo = cliente1.saldo - 100;  
        ..  
        cliente1.RealizaSaque(100);  
    }  
}
```



Importância dos Métodos

➡ Com isso, devemos nos habituar com a definição e uso de métodos;



Atributos estáticos



Atributos estáticos

```
public class ClienteBanco {  
    String nome;  
    int conta;  
    float saldo;  
    static float taxa_CPMF = 0.01F; // Exemplo: 1%  
  
    void RealizaDeposito (float pValor) {  
        saldo = saldo + pValor*(1 - taxa_CPMF);  
    }  
}
```



Exercício

- ➡ Implementar um método para realizar a operação de transferência entre contas.
- ➡ Codifique este e os outros métodos, compile, execute e compare as saídas das versões sem e com o desconto do CPMF.

Solução do Exercício de Transferência entre Contas s/ Desconto

```
public class ClienteBanco {  
    ...  
    void TransferirOutraConta (float pValor,  
                               ClienteBanco pBeneficiado) {  
        RealizaSaque(pValor);  
        pBeneficiado.RealizaDeposito(pValor);  
    }  
}
```



Solução do Exercício de Transferência entre Contas c/ Desconto

```
public class ClienteBanco {  
    ...  
    void TransferirOutraConta (float pValor,  
                               ClienteBanco pBeneficiado) {  
        RealizaSaque(pValor*(1 - taxa_CPMF));  
        pBeneficiado.RealizaDeposito(pValor*(1 - taxa_CPMF));  
    }  
}
```



Identificando Atributos

➡ Considere o método de transferência entre contas sem chamadas a métodos:



Identificando Atributos

➡ Esta pergunta é respondida observando a chamada ao método;

```
public class ClienteBanco {  
    String nome; int conta; float saldo;  
    void TransferirOutraConta (float pValor,  
                               ClienteBanco pBeneficiado) {  
        saldo = saldo - pValor;  
        pBeneficiado.saldo = pBeneficiado.saldo + pValor;  
    }  
    public static void main (String arg[]) {  
        ClienteBanco cliente1, cliente2;  
        cliente1 = new ClienteBanco ("eu", 0001, 500);  
        cliente2 = new ClienteBanco ("voce", 0002, 2000);  
        cliente1.TransferirOutraConta(100,cliente2);  
    }  
}
```



Operador this

```
public class ClienteBanco {
    String nome; int conta; float saldo;
    void TransferirOutraConta (float pValor,
                               ClienteBanco pBeneficiado) {
        this.saldo = this.saldo - pValor;
        pBeneficiado.saldo = pBeneficiado.saldo + pValor;
    }
    public static void main (String arg[]) {
        ClienteBanco cliente1, cliente2;
        cliente1 = new ClienteBanco ("eu", 0001, 500);
        cliente2 = new ClienteBanco ("voce", 0002, 2000);
        cliente1.TransferirOutraConta(100,cliente2);
    }
}
```

