Programação Orientada a Objetos I

CÁSSIO CAPUCHO PEÇANHA – 05

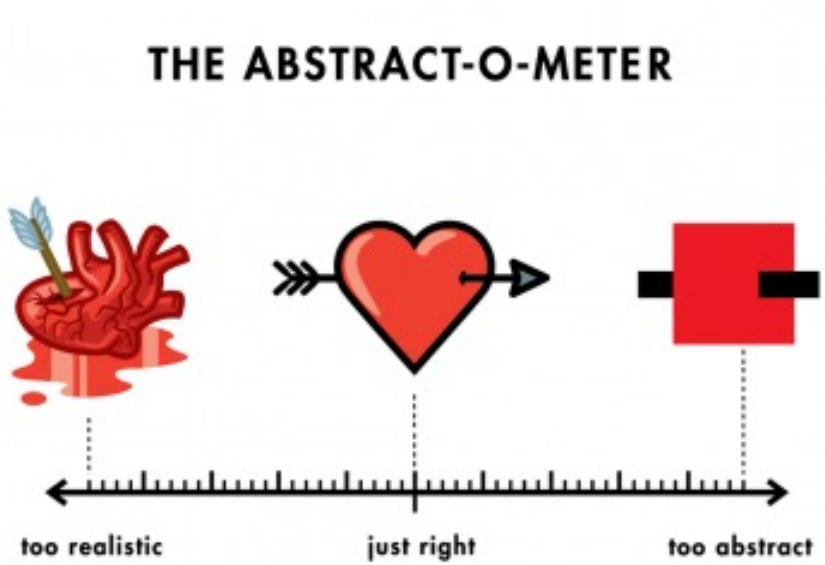
Introdução OO

Conceitos OO fundamentais

- **Abstração:**
  - Desprezar conceitos irrelevantes.
- **Encapsulamento:**
  - Separar a interface da implementação.
- **Modularidade:**
  - Agrupar objetos em módulos coesos independentes.
- **Hierarquia:**
  - Organizar abstrações em hierarquias quando necessário.

POOI - Abstração

- Uma forma de representar algo do mundo real na forma de ideias
- Selecionar aspectos específicos de uma problema a ser analisar e relevar outros
- “Pelo princípio da abstração, isolamos os objetos que queremos representar do ambiente complexo em que se situam, e nesse objeto representamos apenas somente as características que são relevantes para o problema em questão.”



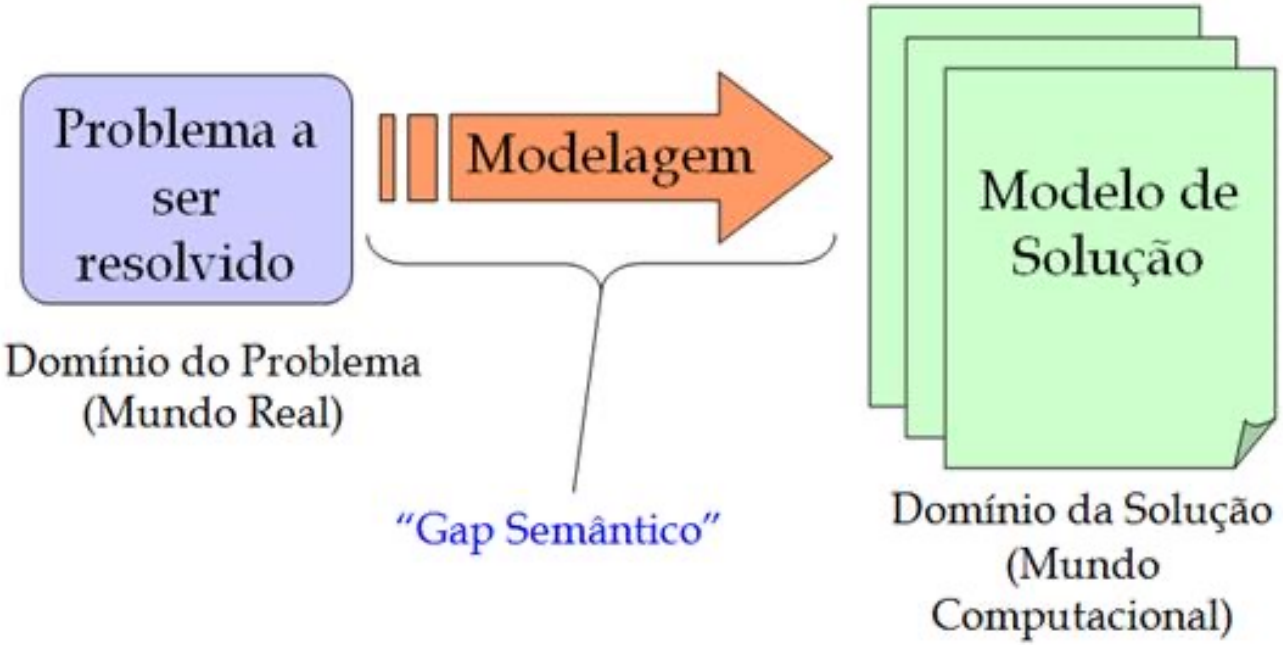
Abstração e OO

- Dentre os paradigmas existentes, a Orientação a Objetos destaca-se pelo nível de abstração:
  - Elementos do mundo real são modelados como objetos no mundo computacional;
  - Objetos possuem propriedades e comportamento, assim como no mundo real;
  - O código expressa a solução em termos mais próximos do problema.

Gap semântico

- O objetivo é escrever as soluções em termos cada vez mais próximos do mundo real.
- Nossa mente é orientada a objetos;
  - Se nossa LP também for OO, podemos diminuir o gap semântico:
- Os recursos oferecidos são os mesmos, porém os mecanismos de abstração são mais poderosos;
- Passamos a construir a solução em termos do domínio do problema (mundo real – objetos);
- Usamos também objetos que não são parte do domínio do problema.

Gap semântico



O que é um objeto?

- Todo objeto no mundo real possui 3 características:
  - Estado;
  - Comportamento;
  - Identidade única.

O que é um objeto?

Objeto	Estado	Comportamento
Cão	Nome Raça Cor	Caçando Comendo Balançando o ramo
Bicicleta	Velocidade do pedal Marcha Velocidade da Bike	Freiar Acelerar Mudar de marcha

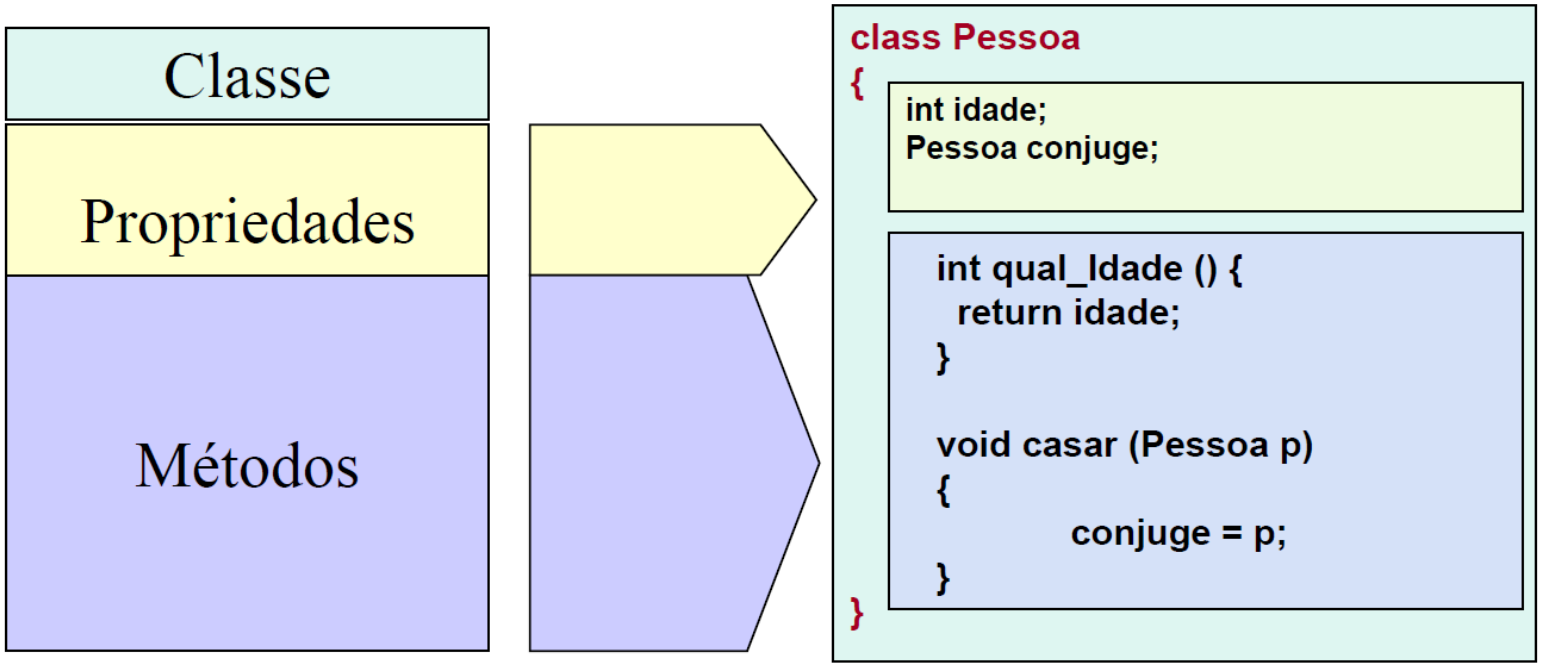
Classificação de objetos (Classe)

- Desde Aristóteles que o ser humano classifica os objetos do mundo;
- Juntamos objetos com mesmas características em categorias que chamamos de “classes”:
- Todas as contas de banco tem um saldo, mas cada conta pode ter um saldo diferente;
- Todas as contas de banco podem sofrer depósitos ou serem encerradas.
- Classes são usadas por linguagens OO para modelar tipos compostos. São modelos abstratos que definem os objetos da classe.

Instâncias de classes

- Objetos são instâncias de classe;
- Para criar um novo objeto, devemos especificar de qual classe;
- Podemos criar quantos objetos quisermos de uma determinada classe;

Estrutura de uma Classe



Definição de uma classe

- Uso da palavra reservada class;
  - Significado: “segue abaixo a especificação de como objetos deste tipo devem se comportar”.

```
class NomeDaClasse {
   /* Especificação da classe vai aqui. */
}
```

- Depois de definida a classe, podemos definir variáveis (referências) e criar objetos:

```
NomeDaClasse obj = new NomeDaClasse();
```

Membros da classe

- Uma classe pode ter dois tipos de membro:
  - Variáveis (em jargão OO: “atributos”);
  - Funções (em jargão OO: “métodos”).
- Atributos são como partes de um tipo composto;
- Métodos são funções que são executadas no contexto de uma classe/objeto.

Atributos

- Definidos como variáveis no escopo da classe:

```
class SoDados {
   int i;
   float f;
   boolean b;
}
```

- As variáveis não são declaradas dentro de um bloco de função, mas diretamente no bloco da classe.

Atributos

- Acesso via operador de seleção (“.”):

```
SoDados d = new SoDados();
d.i = 47;
d.f = 1.1;
d.b = false;
```

Exemplo

CLASSE

```
class Conta {
        int numero;
        String dono;
        double saldo;
        double limite;
// ...
}
```

UTILIZAÇÃO

```
class Programa {
public static void main(String[] args) {
        Conta minhaConta;
        minhaConta = new Conta();
        minhaConta.dono = "Duke";
        minhaConta.saldo = 1000.0;
        System.out.println("Saldo: " +
        minhaConta.saldo);
}
}
```

O valor null

- O valor default para atributos referência (objetos) é null;
- Um “objeto nulo” é uma referência que não aponta para nenhum objeto;
- Usar uma referência nula como se ela apontasse para um objeto causa NullPointerException.

```
UmDado d = null;
System.out.println(d);      // OK!
System.out.println(d.i);    // NPE!
```

Métodos

- Um método é uma função que opera no contexto de uma classe (mensagem que o objeto recebe):

- É a maneira (método) de se fazer algo num objeto.

```
class UmDado {
   int i = 150;
   void imprimir() {
      System.out.println(i);
   }
}

/* Em outro ponto do código... */
UmDado ud = new UmDado();
ud.imprimir(); // 150
ud.i = 300;
ud.imprimir(); // 300
```

Exemplo

CLASSE

```
class Conta {
    int numero;
    String dono;
    double saldo;
    double limite;

    void sacar(double qtd) {
        double novoSaldo = this.saldo - qtd;
        this.saldo = novoSaldo;
    }
    void depositar(double qtd) {
        this.saldo += qtd;
    }
}
```

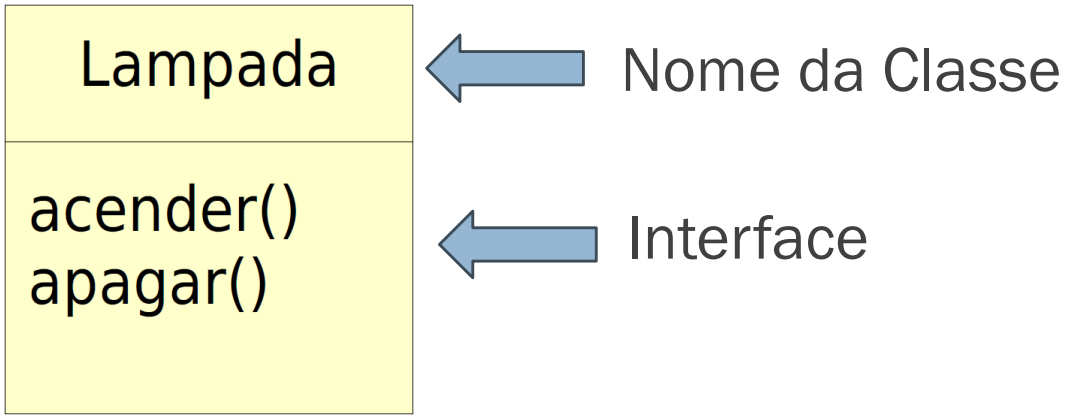
UTILIZAÇÃO

```
class Programa {
public static void main(String[] args) {
    Conta minhaConta = new Conta();
    minhaConta.dono = "Duke";
    minhaConta.saldo = 1000;
    minhaConta.sacar(200);
    minhaConta.depositar(500);
    // Saldo: 1300.0
    System.out.println("Saldo: " + minhaConta.saldo);
}
}
```

Um objeto tem uma interface

- A interface define o que um objeto pode fazer:

```
Lampada l = new Lampada();
l.acender();
```



Um objeto tem uma implementação

- Cada mensagem que o objeto pode aceitar deve ter um código de implementação associado;

```
public class Lampada {
   public void acender() {
      /* Implementação... */
   }
}
```

Envio de mensagens

- Objetos enviam mensagens a outros objetos.