Programação I

Introdução à Orientação a Objetos

Samuel da Silva Feitosa

Aula 6

Introdução

- A ideia da Orientação a Objetos é **aproximar** a programação ao **mundo real**.
- Paradigma idealizado na década de 70 por Allan Key, dando origem ao SmallTalk.
- Exemplos de linguagens: Java, C++, C#, PHP, Python, Perl, etc.

O que já sabemos?

- Aprendemos em outras disciplinas a trabalhar com **tipos de dados compostos** / **abstratos**.
  - Um tipo que pode armazenar **várias** informações.
- Como funciona em outras linguagens?
  - Em C, um tipo composto é criado a partir de uma **struct**.
  - ○ Em Haskell, utilizamos um **algebraic data type** (ADT).
  - ○ Em Java, utilizamos uma **classe**.

Exemplo em Java

- Um tipo composto para armazenar informações de um veículo.

```
public class Veiculo {
    String marca;
    String modelo;
    int nrRodas;
    int nrPortas;
    double consumoCidade;
    double consumoEstrada;
    boolean ligado;
}
```

Domínio e Aplicação

- Um **domínio** é composto pelas entidades, informações e processos relacionados a um determinado contexto.
- Uma **aplicação** pode ser desenvolvida para automatizar ou tornar factível as tarefas de um domínio.
- Suponha que estamos interessados em desenvolver uma **aplicação** para facilitar as tarefas do cotidiano de um banco.

Domínio e Aplicação

- Podemos identificar como entidades do domínio:
  - Clientes
  - Funcionários
  - Agências
  - Contas

Conceito de Abstração

- A **abstração** é considerada por diversos autores como o primeiro pilar da OO.
  - **Abstração** é a habilidade de se concentrar nos **aspectos essenciais** de um contexto qualquer, **ignorando** características menos importantes ou acidentais.
  - Em modelagem orientada a objetos, uma **classe** é uma **abstração** de entidades existentes no domínio do sistema de software.

Mais sobre Abstração

- A identificação de elementos de um domínio é uma **tarefa difícil**.
  - Depende fortemente do **conhecimento** das entidades, informações e processo.
  - Em geral, as pessoas que possuem esse conhecimento ou parte dele estão em contato constante com o domínio e **não possuem** conhecimentos técnicos.
  - Desenvolvedores buscam mecanismos para tornar mais eficiente o entendimento do domínio.

Objetos

- As entidades identificadas no domínio devem ser representadas de alguma forma dentro da aplicação.
  - Nas aplicações orientadas a objetos, as entidades são representadas por **objetos**.
  - Uma aplicação OO é composta por diversos objetos.
- Suponha que no **domínio** bancário exista um **cliente** chamado **João**.
  - Dentro de uma aplicação OO, deve existir um **objeto** para representar o cliente João.

Atributos

- Algumas informações do cliente João são importantes para o banco:
  - Nome, sexo, data de nascimento, etc.
  - Já que esses dados são relevantes para o domínio, o objeto que representa esse cliente deve possuir essa informação.
- Esses dados são armazenados nos **atributos** do objeto que representa o João.
  - Um atributo é uma variável que pertence ao objeto.
  - Os dados de um objeto são armazenados nos seus atributos.

Métodos

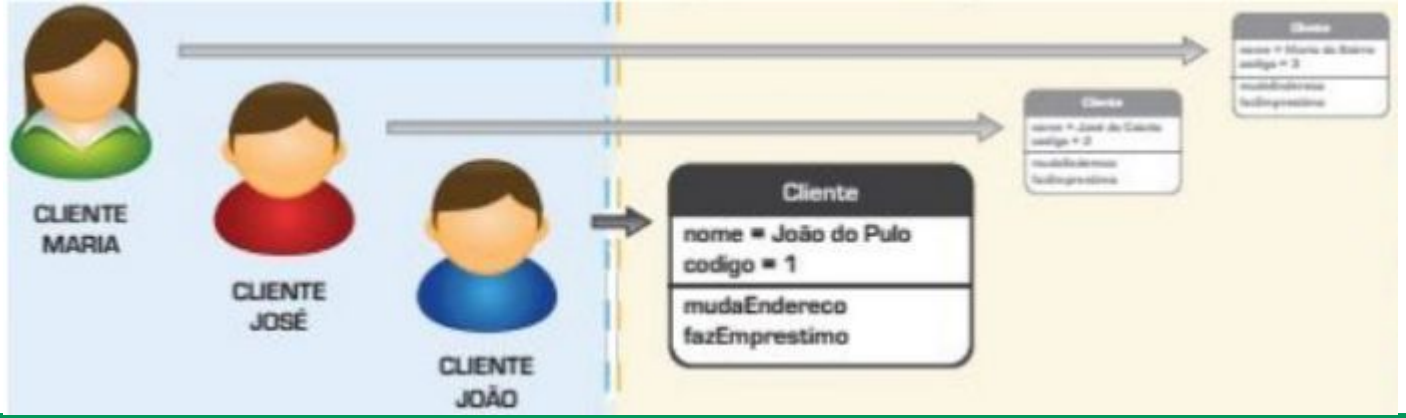
- O próprio objeto deve realizar **operações** de **consulta** ou **alteração** dos valores de seus atributos.
- Essas operações são definidas nos **métodos** do objeto.
  - As tarefas que um objeto pode realizar são definidas pelos seus métodos.
  - Um objeto é composto por **atributos** e **métodos**.

Métodos

- Os **métodos** também são utilizados para possibilitar **interações** entre os **objetos** de uma aplicação.
  - Um método pertencente a um objeto pode **invocar** métodos a partir de outros objetos.
  - Geralmente chamamos este processo de **trocas de mensagens** entre objetos.
- Quando um cliente requisita um saque:
  - O objeto que representa o **caixa eletrônico** deve interagir com o objeto da **conta** do cliente.

Exemplo no domínio bancário

- Não é adequado utilizar o objeto que representa um determinado cliente para representar outro cliente do banco.
  - Os **dados** de cada cliente são **diferentes**.
  - Para **cada cliente** do banco, deve existir um **objeto** dentro do sistema para representá-lo.



Mais sobre Objetos…

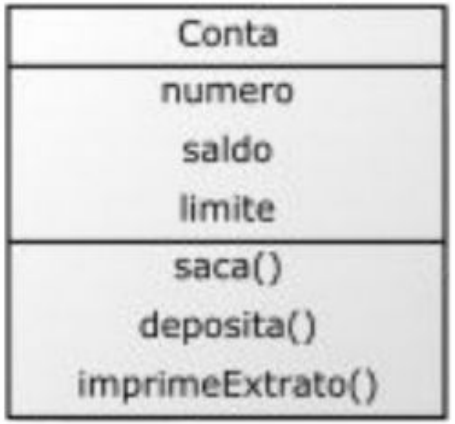
- Objetos não representam apenas coisas **concretas** como os clientes do banco.
- Eles também devem ser usados para representar coisas **abstratas** como uma conta de um cliente ou um serviço que o banco oferta.

Classes

- **Antes** de um objeto ser **criado**, devemos definir quais serão seus **atributos** e **métodos**.
  - Atributos representam o **estado** de um objeto.
  - Métodos são as **operações** que permitem modificar o **estado** de um objeto.
- Essa definição é realizada através de uma **classe** elaborada por um programador.
- A partir de uma **classe**, podemos construir **objetos** na **memória** do computador que executa a nossa aplicação.

Modelando Classes

- Podemos representar uma classe através de **Diagramas de Classe UML**.
  - Este é composto pelo nome da classe e pelos atributos e métodos que ela define.
  - Todos os objetos criados a partir da classe **Conta** terão os **atributos** e **métodos** mostrados no diagrama UML.
  - Os valores dos atributos de dois objetos criad classe conta **podem ser diferentes**.



Analogia: Objetos

- Um **objeto** é como se fosse uma **casa** ou um **prédio**.
  - Para ser construído, precisa de um **espaço físico**.
- No caso dos objetos, esse espaço físico é algum trecho vago na **memória** do computador que executa a aplicação.
- No caso das casas e dos prédios, o espaço físico é algum terreno vazio.

Analogia: Classe (1)

- Um prédio é construído a partir de uma **planta** criada por um engenheiro ou arquiteto.
  - Para criar um objeto, é necessário algo semelhante a uma planta, para que sejam “desenhados” os atributos e métodos que o objeto deve ter.
- Em orientação a objetos, a **planta** de um objeto é o que chamamos de **classe**.

Analogia: Classe (2)

- Uma classe funciona como uma **receita** para criar objetos.
- Inclusive, **vários objetos** podem ser criados a partir de uma **única classe**.
  - Assim como várias casas ou prédios podem ser construídos a partir de uma única planta.
  - Vários bolos podem ser preparados a partir de uma única receita.
  - Vários carros podem ser construídos a partir de um único projeto.

Exemplos

Exemplos

- As diferenças entre dois objetos criados a partir da **classe Casa** são os **valores** de seus atributos.
  - Duas casas construídas a partir da **mesma planta** podem possuir características **diferentes**.

Considerações Finais

- Nesta aula estudamos os **conceitos introdutórios** da orientação a objetos:
  - Classes, objetos, atributos e métodos.
- Através de analogias do mundo real, foi possível relacionar os conceitos de classes e objetos.
- Vimos também o conceito de **abstração** que é o primeiro pilar da orientação a objetos.