

Classes, Objetos, Comportamento, Estado, Operações e Atributos

Conceitos fundamentais da Orientação a Objetos com exemplos em Java

1. Introdução

Para compreender a Programação Orientada a Objetos é essencial dominar um conjunto de termos que descrevem como um objeto é estruturado e como ele se comporta dentro de um sistema. Os principais conceitos são: classe, objeto, estado, comportamento, atributos e operações. Este texto explica cada um deles e apresenta exemplos práticos em Java.

2. Classe

Uma **classe** é a definição, o modelo ou o "molde" que descreve como serão os objetos criados a partir dela. Ela especifica quais atributos (dados) e quais operações (métodos) os objetos terão, mas não representa nenhum objeto específico — é apenas a estrutura genérica.

```
public class Pessoa {  
    // Atributos (definidos pela classe)  
    private String nome;  
    private int idade;  
  
    // Operações (definidas pela classe)  
    public void apresentar() {  
        System.out.println("Olá, meu nome é " + nome);  
    }  
}
```

A classe Pessoa apenas descreve a estrutura; nenhum objeto foi criado ainda.

3. Objeto

Um **objeto** é uma instância concreta de uma classe, criada em tempo de execução com a palavra-chave `new` em Java. Cada objeto possui seus próprios valores para os atributos definidos na classe, mesmo que compartilhe a mesma estrutura e os mesmos métodos de outros objetos da mesma classe.

```
public class Main {  
    public static void main(String[] args) {  
        Pessoa pessoa1 = new Pessoa(); // objeto 1  
        Pessoa pessoa2 = new Pessoa(); // objeto 2  
  
        pessoa1.setNome("Ana");  
        pessoa2.setNome("Bruno");  
  
        pessoa1.apresentar(); // Olá, meu nome é Ana  
        pessoa2.apresentar(); // Olá, meu nome é Bruno  
    }  
}
```

Observe que pessoa1 e pessoa2 são dois objetos distintos, criados a partir da mesma classe Pessoa, mas cada um mantém seu próprio valor para o atributo nome.

4. Atributos

Os **atributos** (também chamados de campos ou variáveis de instância) representam as características ou propriedades de um objeto. Eles armazenam os dados que descrevem o objeto em um determinado momento.

```
public class Produto {  
    private String nome;        // atributo  
    private double preco;      // atributo  
    private int quantidade;    // atributo  
}
```

Nesse exemplo, nome, preco e quantidade são atributos que descrevem cada objeto do tipo Produto.

5. Estado

O **estado** de um objeto é o conjunto de valores atuais de todos os seus atributos em um determinado instante. O estado pode mudar ao longo do tempo, conforme o objeto executa operações que alteram seus atributos.

```
public class Produto {
    private String nome;
    private double preco;
    private int quantidade;

    public Produto(String nome, double preco, int quantidade) {
        this.nome = nome;
        this.preco = preco;
        this.quantidade = quantidade;
    }

    public void venderUnidade() {
        if (quantidade > 0) {
            quantidade--; // o estado do objeto muda
        }
    }
}

// Uso:
Produto p = new Produto("Caderno", 12.50, 10);
// Estado inicial: quantidade = 10
p.venderUnidade();
// Novo estado: quantidade = 9
```

Antes e depois da chamada de `venderUnidade()`, o objeto `p` possui estados diferentes, pois o valor do atributo `quantidade` foi alterado.

6. Comportamento e Operações

O **comportamento** de um objeto é definido pelo conjunto de ações que ele pode realizar — é o que o objeto "sabe fazer". Esse comportamento é implementado por meio de **operações**, também chamadas de **métodos**, que são blocos de código associados à classe. As operações podem consultar o estado do objeto, modificá-lo, ou realizar cálculos e interações com outros objetos.

```

public class ContaBancaria {
    private double saldo;

    public ContaBancaria(double saldoInicial) {
        this.saldo = saldoInicial;
    }

    // Operação que consulta o estado (não altera nada)
    public double consultarSaldo() {
        return saldo;
    }

    // Operação que modifica o estado (comportamento)
    public void depositar(double valor) {
        saldo += valor;
        System.out.println("Depósito realizado. Novo saldo: " + saldo);
    }

    // Operação que modifica o estado, com regra de negócio
    public void sacar(double valor) {
        if (valor <= saldo) {
            saldo -= valor;
            System.out.println("Saque realizado. Novo saldo: " + saldo);
        } else {
            System.out.println("Operação negada: saldo insuficiente.");
        }
    }
}

```

Aqui, `consultarSaldo()`, `depositar()` e `sacar()` são operações que definem o comportamento da classe `ContaBancaria`. As duas últimas alteram o estado do objeto (o valor de `saldo`), enquanto a primeira apenas o consulta.

7. Relação entre os Conceitos

Esses seis conceitos estão intimamente relacionados: a **classe** define a estrutura (quais atributos e operações existirão); o **objeto** é a instância real dessa estrutura; os **atributos** armazenam os dados; o conjunto de valores desses atributos em um dado momento forma o **estado** do objeto; e as **operações** implementam o **comportamento**, ou seja, as ações que o objeto pode executar — muitas vezes alterando seu próprio estado.

```

public class ExemploCompleto {
    public static void main(String[] args) {
        // Classe: ContaBancaria (definida acima)
        // Objeto: conta
        ContaBancaria conta = new ContaBancaria(100.0);

        // Estado inicial: saldo = 100.0
        conta.depositar(50.0); // Comportamento -> novo estado: saldo = 150.0
        conta.sacar(30.0);    // Comportamento -> novo estado: saldo = 120.0

        System.out.println("Saldo final: " + conta.consultarSaldo());
    }
}

```

8. Conclusão

Compreender a diferença entre classe e objeto, e a relação entre atributos, estado, operações e comportamento, é a base para modelar sistemas orientados a objetos de forma correta. A classe fornece o projeto; o objeto dá vida a esse projeto; os atributos guardam os dados; o estado reflete a situação atual desses dados; e as operações — que expressam o comportamento — permitem que o objeto interaja com o restante do sistema e evolua ao longo do tempo.