
FIAP

Domain Driven Design
using Java

AGENDA

1. Explorando Classes Abstratas, Métodos Finais, Atributos Estáticos e Interfaces em Java
2. Exercícios

Introdução

- Este capítulo aborda quatro pilares importantes da programação em Java:

    1. **Classes e Métodos Abstratos**: Estruturam funcionalidades comuns para subclasses.
    2. **Classes e Métodos Finais**: Garantem imutabilidade de classe e método.
    3. **Atributos e Métodos Estáticos**: São compartilhados pela classe em vez de instâncias.
    4. **Interfaces**: Definem contratos de comportamento entre classes.

Classes e Métodos Abstratos

- Classe Abstrata:
  - Não pode ser instanciada.
  - Serve como base para outras classes.
  - Pode conter:
    - Métodos abstratos (sem implementação).
    - Métodos concretos (com implementação).
- Métodos Abstratos:
  - Declarado, mas sem corpo.
  - Deve ser implementado por subclasses concretas.

Exemplo de Classe Abstrata

```java
// Classe abstrata
public abstract class Animal {
    public abstract void emitirSom(); // Método abstrato
    public void dormir() {
        System.out.println("O animal está dormindo.");
    }
}

// Subclasse concreta
public class Cachorro extends Animal {
    @Override
    public void emitirSom() {
        System.out.println("O cachorro late.");
    }
}
```

Carregar

```java
// Uso
public class Main {
    public static void main(String[] args) {
        Animal cachorro = new Cachorro();
        cachorro.emitirSom(); // Saída: O cachorro late.
        cachorro.dormir(); // Saída: O animal está dormindo.
    }
}
```

Prof. Evando Borges

Exercício – Classes e Métodos Abstratos

- Crie uma classe abstrata chamada FormaGeometrica com:
  - Um método abstrato calcularArea().
  - Um método abstrato calcularPerimetro().
- Crie subclasses Circulo e Retangulo que implementem os métodos abstratos.
- No programa principal, instancie as subclasses e calcule a área e o perímetro de ambas.

Classes e Métodos Finais

- Classe Final:
  - Não pode ser herdada.
  - Garante que sua implementação permaneça inalterada.

- Métodos Final:
  - Não pode ser sobrescrito por subclasses.
  - Preserva o comportamento original.

Exemplo de Classe Final

```java
// Classe final
public final class Utilidade {
    public void mostrarMensagem() {
        System.out.println("Mensagem da classe Utilidade.");
    }
}

// Método final
public class Pessoa {
    public final void mostrarIdentidade() {
        System.out.println("Documento de identidade.");
    }
}

public class Funcionario extends Pessoa {
    // Tentativa de sobrescrita gera erro
    // @Override
    // public void m Carregar dentidade() { ... }
}
```

Prof. Evando Borges

Exercício – Classes e Métodos Final

- Crie uma classe final Banco com um método exibirDadosBancarios() que imprime o nome do banco e sua agência.
- Crie uma classe Conta com:
    - Um método final exibirSaldo() que exibe o saldo da conta.
- No programa principal, mostre os dados do banco e o saldo de uma conta.

Atributos e Métodos Estáticos

- Atributo Estático:
  - Pertence à classe, não à instância.
  - Compartilhado por todas as instâncias da classe.

- Métodos Estático:
  - Pertence à classe e pode ser chamado diretamente, sem instanciar.

Exemplo de Atributos e Métodos Estáticos

```java
public class Calculadora {
    public static int contador = 0; // Atributo estático

    public static int somar(int a, int b) { // Método estático
        contador++;
        return a + b;
    }
}

public class Main {
    public static void main(String[] args) {
        int resultado = Calculadora.somar(5, 3); // Chamando método estátic
        System.out.println("Resultado: " + resultado); // Saída: Resultado:
        System.out.println("Contador: " + Calculadora.contador); // Saída: 
    }
}
```

Exercício – Métodos e Atributos Estáticos

- Crie uma classe Estoque com:
  - Um atributo estático totalProdutos.
  - Um método estático adicionarProduto() que incrementa o total de produtos.
- No programa principal:
  - Adicione três produtos ao estoque.
  - Exiba o total de produtos.

Interfaces

- Definição:
  - Define métodos que devem ser implementados por uma classe.
  - Uma classe pode implementar várias interfaces.
  - Pode conter:
    - Métodos abstratos.
    - Métodos padrão (default).
    - Métodos estáticos

Diferenças entre Interface e Classe Abstrata

- Interface:
  - Define um contrato que outras classes devem seguir.
  - Contém apenas declarações de métodos (sem implementação), mas pode ter métodos com implementação padrão (default) e métodos estáticos (a partir do Java 8)
  - Uma classe pode implementar várias interfaces..
- Classe Abstrata:
  - Uma classe que serve como base para outras, combinando métodos abstratos (sem implementação) e concretos (com implementação).
  - Pode conter atributos e construtores.
  - Uma classe só pode estender uma única classe abstrata.

Exemplo de Atributos e Métodos Estáticos

```java
// Definição da interface
public interface Veiculo {
    void acelerar();
    void frear();

    // Método com implementação padrão
    default void ligarMotor() {
        System.out.println("O motor está ligado.");
    }
}

// Implementação da interface
public class Carro implements Veiculo {
    @Override
    public void acelerar() {
        System.out.println("O carro está acelerando.");
    }

    @Override
    public void frear() {
        System.out.println("O carro está freando.");
    }
}

// Uso
public class Main {
    public static void main(String[] args) {
        Veiculo carro = new Carro();
        carro.ligarMotor(); // Saída: O motor está ligado.
        carro.acelerar(); // Saída: O carro está acelerando.
    }
}
```

Carregar

Prof. Evando Borges

Exercício – Interfaces

- Crie uma interface Pagamento com:
  - Métodos abstratos realizarPagamento() e estornarPagamento().
- Crie classes CartaoCredito e Boleto que implementem a interface.
- No programa principal:
  - Instancie as classes e chame os métodos de pagamento e estorno.

Conclusão

- **Classes Abstratas:** Servem como base para subclasses com métodos abstratos e concretos.
- **Métodos Finais:** Preservam comportamento; Classes Finais previnem herança.
- **Atributos/Métodos Estáticos:** Pertencem à classe, não à instância.
- **Interfaces:** Permitem múltiplos comportamentos e contratos para implementação.


FIAP