POO usando Java: Encapsulamento

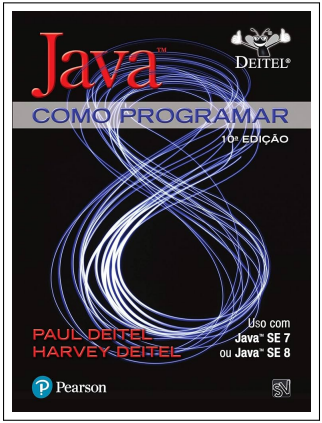
Programação Orientada a Objetos

Profs. Aleffer Rocha
aleffer.rocha@ifpr.edu.br
Prof. William Deus
william.deus@ifpr.edu.br

INSTITUTO FEDERAL
Paraná
Campus
Pinhais

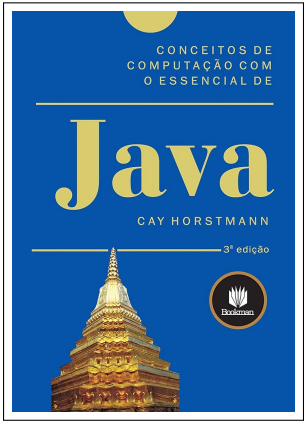
Material atualizado em 24 de agosto de 2024

Materiais utilizados



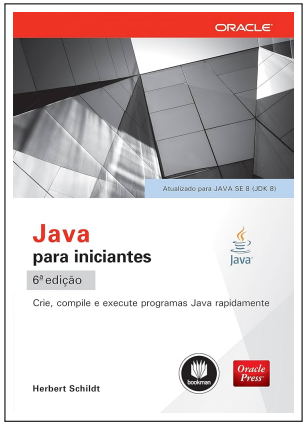
DEITEL, Paul; DEITEL, Harvey. **Java: como programar.** 10. ed. São Paulo, SP: Pearson Education do Brasil, 2017.

Materiais utilizados



HORSTMANN, Cay S. **Conceitos de computação com o essencial de Java.** 3. ed. Porto Alegre, RS: Bookman, 2005.

Materiais utilizados



SCHILDT, Herbert. **Java para iniciantes:** crie, compile e execute programas Java rapidamente. 6. ed. Porto Alegre, RS: Bookman, 2015.x

Introdução

INSTITUTO FEDERAL
Paraná
Campus Pinhais

O que é encapsulamento?

- Princípio da programação orientada a objetos.
- Oculta detalhes internos de uma classe.
- Restringe o acesso direto aos membros da classe.

Métodos

- `get`: retorna valor de um atributo.
- `set`: define valor de um atributo.

Objetos do encapsulamento

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Proteção de dados

- Atributos privados.

Controle de acesso

- Modificadores como `private`, `public` e `protected`.

Manutenção

- Facilita alterações internas sem impactar o código externo.

Exemplo

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Métodos `get` e `set` no exemplo

- `getModelo()`: retorna o valor do atributo `modelo`.
- `setModelo(String novoModelo)`: define um novo valor para o atributo `modelo`.
- `getAno()`: retorna o valor do atributo `ano`.
- `setAno(int novoAno)`: define um novo valor para o atributo `ano`.

```java
public class Carro {
    private String modelo;
    private int ano;

    public String getModelo() {
        return modelo;
    }

    public void setModelo(String novoModelo) {
        this.modelo = novoModelo;
    }

    public int getAno() {
        return ano;
    }

    public void setAno(int novoAno) {
        this.ano = novoAno;
    }
}
```

Benefícios do Encapsulamento

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Controle de acesso

- Restringe o acesso direto a membros internos da classe.
- Fornece uma interface controlada para interações externas.
- Evita manipulação direta de dados sensíveis.

Validação

- Verifica e valida dados antes de permitir modificações.
- Previne a inserção de dados inválidos ou inconsistentes.
- Garante a integridade dos dados dentro da classe.

Benefícios do Encapsulamento

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Segurança

- Protege contra acesso não autorizado aos dados internos.
- Oculta a implementação interna da classe, reduzindo riscos de manipulação inadequada.
- Contribui para a construção de sistemas mais seguros e robustos.

Manutenção

- Facilita modificações internas sem afetar o código externo.
- Permite a evolução da implementação sem quebrar dependências externas.
- Contribui para o desenvolvimento de software mais sustentável e de fácil manutenção.

Exemplo: Classe `Cliente`

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Classe Cliente sem encapsulamento.

```java
public class Cliente {
    public String nome;
    private String senha;

    public Cliente(String nome, String senha) {
        this.nome = nome;
        this.senha = senha;
    }
}

// Uso sem encapsulamento
Cliente cliente = new Cliente("Maria", "senha123");
System.out.println(cliente.senha);  // Acesso direto a senha
```

Exemplo: Classe `Cliente`

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Classe Cliente com encapsulamento.

```java
public class Cliente {
    private String nome;
    private String senha;

    public Cliente(String nome, String senha) {
        this.nome = nome;
        setSenha(senha);
    }

    public String getNome() {
        return nome;
    }

    public String getSenha() {
        return senha;
    }
```

```java
    public void setSenha(String senha) {
        if (senha.length() >= 6) {
            this.senha = senha;
        } else {
            System.out.println("Senha invalida");
        }
    }
}

// Uso com encapsulamento
Cliente cliente = new Cliente("Maria", "senha123");

// Acesso controlado atraves do metodo getSenha()
System.out.println(cliente.getSenha());
```

Atividades para fixação

Vamos colocar em prática o que aprendemos na aula de hoje.

Exercício 1: Classe `Aluno`

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Objetivo: criar uma classe Java chamada `Aluno` que encapsule informações sobre um estudante, incluindo nome, idade e média.

Requisitos

- Declare atributos privados para `nome` (String), `idade` (int) e `media` (double).
- Implemente métodos `get` e `set` para cada atributo.
- Adicione validações nos métodos `set` para garantir que a idade seja um valor positivo e que a média esteja no intervalo [0, 10].

Exercício 2: Sistema Bancário

INSTITUTO FEDERAL
Paraná
Campus Pinhais

Objetivo: desenvolver um sistema bancário simples com foco em encapsulamento.

Requisitos

- Crie uma classe `ContaBancaria` com os atributos privados para `saldo` (double) e `titular` (String).
- Implemente métodos `get` e `set` para ambos os atributos.
- Adicione um método `depositar(double valor)` que adiciona um valor ao saldo.
- Adicione um método `sacar(double valor)` que retira um valor do saldo, respeitando a disponibilidade.