
FIAP

Domain Driven Design
using Java

AGENDA

1. Padrões Arquiteturais em Java
2. Exercícios

Introdução

- Padrões abordados:
  - Arquitetura em Camadas (*Layered Architecture*)

Padrão Arquitetura em Camadas

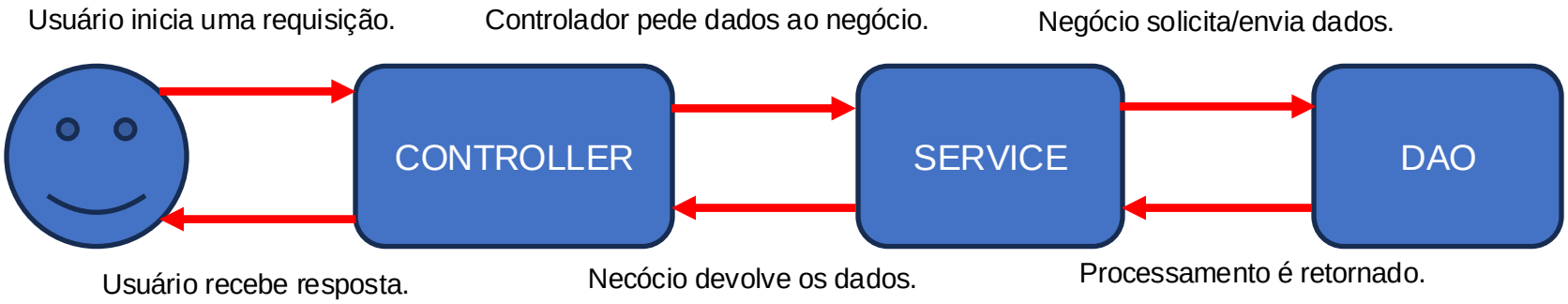
- O que é?
  - A arquitetura em camadas é uma estrutura de design que separa a aplicação em diferentes níveis ou camadas, cada uma com responsabilidades específicas.

- Porque aprender MVC:
  - Permite modularidade.
  - Facilita manutenção e testes.
  - Promove reutilização de código.
  - Amplamente utilizada em sistemas corporativos, como ERPs, CRMs e aplicações web.

Estrutura da Arquitetura em Camadas

- Camada de Apresentação (Presentation Layer):
  - Responsável pela interface com o usuário.
  - Exibe informações e captura interações.
  - Exemplos: Interfaces gráficas (Swing, JavaFX) ou web (HTML, JSP).
- Camada de Negócio (Business Layer):
  - Contém a lógica principal do sistema.
  - Aplica regras de negócio e validações.
  - Exemplos: Métodos que calculam preços ou verificam permissões.
- Camada de Persistência (Data Layer):
  - Gerencia o acesso aos dados no banco.
  - Realiza operações como CRUD (Create, Read, Update, Delete).
  - Exemplos: Classes DAO (Data Access Object).

Estrutura da Arquitetura em Camadas



HTML.
APP Mobile.
App Desktop

Benefícios da Arquitetura em Camadas

- Modularidade:
  - Cada camada é independente, permitindo alterações sem impactar o sistema como um todo.
- Manutenção simplificada:
  - A separação de responsabilidades facilita identificação e correção de bugs.
- Reutilização:
  - Componentes da camada de negócios ou persistência podem ser usados em diferentes contextos.
- Escalabilidade:
  - Camadas podem ser distribuídas em diferentes servidores para aumentar a capacidade do sistema.

Exemplo de Arquitetura em Camadas

- Camada de apresentação

```java
import javax.swing.*;
import java.awt.event.ActionListener;

public class ProdutoView extends JFrame {
    private JTextField nomeField = new JTextField(20); // Campo para o nome
    private JTextField precoField = new JTextField(10); // Campo para o pre
    private JButton salvarButton = new JButton("Salvar Produto"); // Botão
    private JButton listarButton = new JButton("Listar Produtos"); // Botão
    private JTextArea displayArea = new JTextArea(10, 30); // Área para exi

    public ProdutoView() {
        JPanel panel = new JPanel();
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setSize(400, 300);

        panel.add(new JLabel("Nome:"));
        panel.add(nomeField);
        panel.add(new JLabel("Preço:"));
        panel.add(precoField);
        panel.add(salvarButton);
        panel.add(listarButton);
        panel.add(new JScrollPane(displayArea)); // Área para exibir dados

        this.add(panel);
    }

    // Métodos para capturar dados inseridos pelo usuário
    public String getNome() {
        return nomeField.getText();
    }

    public double getPreco() {
        try {
            return Double.parseDouble(precoField.getText());
        } catch (NumberFormatException e) {
            return 0; // Retorna 0 se o preço for inválido
        }
    }

    // Métodos para adicionar eventos aos botões
    public void addSalvarListener(ActionListener salvarListener) {
        salvarButton.addActionListener(salvarListener);
    }

    public void addListarListener(ActionListener listarListener) {
        listarButton.addActionListener(listarListener);
    }

    // Exibe os produtos na área de texto
    public void displayProdutos(String produtos) {
        displayArea.setText(produtos);
    }

    public void displayErrorMessage(String mensagemErro) {
        JOptionPane.showMessageDialog(this, mensagemErro);
    }
}
```

Exemplo de Arquitetura em Camadas

- Classe de Negócio

```java
import java.util.ArrayList;
import java.util.List;

public class ProdutoService {
    private ProdutoDAO dao = new ProdutoDAO(); // Interação com a persistên

    // Valida e processa o cadastro de um produto
    public void salvarProduto(String nome, double preco) throws Exception {
        if (nome == null || nome.isEmpty()) {
            throw new Exception("O nome do produto não pode ser vazio!");
        }
        if (preco <= 0) {
            throw new Exception("O preço do produto deve ser maior que zero
        }

        // Salva no DAO
        dao.salvarProduto(new Produto(nome, preco));
    }

    // Retorna a lista de produtos
    public List<Produto> listarProdutos() {
        return dao.listarProdutos();
    }
}
```

Exemplo de Arquitetura em Camadas

- Camada DAO

```java
import java.util.ArrayList;
import java.util.List;

public class ProdutoDAO {
    private List<Produto> produtos = new ArrayList<>(); // Simulação de ban

    public void salvarProduto(Produto produto) {
        produtos.add(produto);
        System.out.println("Produto salvo: " + produto.getNome());
    }

    public List<Produto> listarProdutos() {
        return produtos;
    }
}
```

Exemplo de Arquitetura em Camadas

- Entidade

```java
public class Produto {
    private String nome;
    private double preco;

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

    public String getNome() {
        return nome;
    }

    public double getPreco() {
        return preco;
    }
}
```

Exemplo de Arquitetura em Camadas

- Controller

```java
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ProdutoController {
    private ProdutoView view;
    private ProdutoService service;

    public ProdutoController(ProdutoView view, ProdutoService service) {
        this.view = view;
        this.service = service;

        // Vincula os botões aos eventos
        this.view.addSalvarListener(new SalvarProdutoListener());
        this.view.addListarListener(new ListarProdutosListener());
    }

    class SalvarProdutoListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent e) {
            try {
                String nome = view.getNome();
                double preco = view.getPreco();

                service.salvarProduto(nome, preco);
                view.displayProdutos("Produto salvo com sucesso!");
            } catch (Exception ex) {
                view.displayErrorMessage("Erro ao salvar produto: " + ex.ge
            }
        }
    }
```

```java
    class ListarProdutosListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent e) {
            StringBuilder lista = new StringBuilder();
            for (Produto produto : service.listarProdutos()) {
                lista.append(produto.getNome())
                     .append(" - R$ ")
                     .append(produto.getPreco())
                     .append("\n");
            }
            view.displayProdutos(lista.toString());
        }
    }
}
```

Exemplo de Arquitetura em Camadas

- Main

```
public class Main {
    public static void main(String[] args) {
        ProdutoView view = new ProdutoView();
        ProdutoService service = new ProdutoService();
        ProdutoController controller = new ProdutoController(view, service)

        // Exibe a interface gráfica
        view.setVisible(true);
    }
}
```

Exercícios

1 - Camada de Apresentação:

- Crie uma interface gráfica para cadastrar clientes (Swing ou JavaFX)..

2 - Camada de Negócio:

- Adicione validação para garantir que o nome e o e-mail não estejam vazios.

3 - Camada de Persistência:

- Salve os dados em uma lista ou banco (DAO).

4 - Teste:

- Insira clientes e exiba a lista cadastrada.


FIAP