

Desenvolvimento Web II

Aula 02 - Padrões de Arquitetura

Prof. Fabricio Bizotto

Instituto Federal Catarinense
fabricio.bizotto@ifc.edu.br

Ciência da Computação
3 de março de 2024

1 Padrões de Projeto para Web

- Conceitos
- MVC
- MVP
- MVVM

2 Conclusão

3 Material Complementar

- Quiz

4 Tarefa

5 Experimentos

Os **Padrões de Projeto para Web** são soluções reutilizáveis para problemas comuns de design de software que surgem no desenvolvimento de aplicativos web. Eles fornecem **diretrizes e estruturas** para organizar o código, melhorar a escalabilidade, a manutenibilidade e a eficiência do desenvolvimento. Os mais comuns são:

- **Modelo-Visão-Controlador (MVC)**
- **Modelo-Visão-Presenter (MVP)**
- **Modelo-Visão-ViewModel (MVVM)**

Padrões de Projeto

MVC

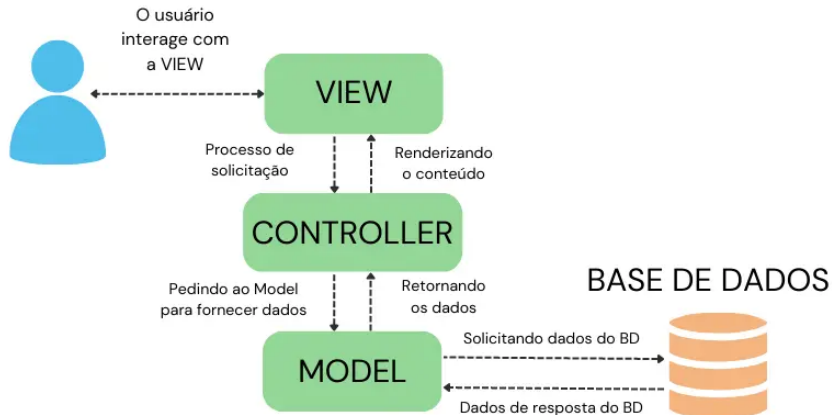
Model-View-Controller

Definição

É um dos padrões de projeto mais conhecidos e adotados pela indústria de software. Foi introduzido pela primeira vez no final da década de 1970 por Trygve Reenskaug, um cientista da computação norueguês, e desde então se tornou um elemento básico na Projeto de aplicativos. O padrão facilita a separação de interesses dividindo o aplicativo em três componentes principais.



Padrões de Projeto para Web



- **Model:** representa os dados e a lógica de negócios da aplicação. Ele encapsula o estado e o comportamento dos dados, e responde a consultas sobre esses dados, bem como a atualizações desses dados. O modelo não está ciente da interface do usuário nem da forma como os dados são apresentados ou manipulados.
- **View:** é a camada de apresentação da aplicação. Ela é responsável por exibir os dados ao usuário de maneira visualmente compreensível e interativa. A visão normalmente consome dados do modelo e os apresenta de forma apropriada para o usuário. É importante destacar que a visão não manipula diretamente os dados, mas apenas os exibe.
- **Controller:** atua como intermediário entre o Modelo e a Visão. Ele recebe as entradas do usuário, como cliques do mouse ou pressionamentos de teclas, e traduz essas entradas em ações que devem ser realizadas no Modelo ou na Visão. Ele manipula as requisições do usuário, atualiza o estado do Modelo conforme necessário e seleciona a Visão apropriada para exibir os resultados ao usuário.

Padrões de Projeto para Web

MVC - Model-View-Controller

Onde é usado?

Frameworks populares como Rails, Django (MVT), Laravel e Spring MVC são exemplos de tecnologias que adotam o padrão MVC para criar aplicativos da web. Nesses frameworks, o modelo representa os dados e a lógica de negócios, a view cuida da interface do usuário e o controller gerencia as solicitações do usuário e coordena a interação entre o modelo e a view.

Padrões de Projeto

MVP

Model-View-Presenter

Aborda algumas das desvantagens da abordagem MVC tradicional. Originou-se no início da década de 1990 na Taligent, uma joint venture entre Apple, IBM e Hewlett-Packard. Foi ainda mais popularizado pelo Dolphin Smalltalk em 1998 e, em 2006, a Microsoft adotou o MVP para programação de interface de usuário no framework .NET.



- **Model:** assim como no padrão MVC, o Modelo (Model) no padrão MVP representa os dados e a lógica de negócios da aplicação. Ele é responsável por armazenar e manipular os dados, além de conter a lógica de negócios da aplicação. No entanto, no MVP, o Modelo é geralmente mais simples e passivo do que no MVC.
- **View:** é responsável pela apresentação dos dados ao usuário. Ela exibe a interface do usuário e recebe interações do usuário. No entanto, ao contrário do MVC, a Visão no MVP é mais passiva e tem um papel mais limitado na aplicação. Ela não realiza diretamente ações sobre o Modelo.
- **Presenter:** é o componente central do padrão MVP e desempenha um papel semelhante ao do Controlador no MVC. Ele atua como um intermediário entre o Modelo e a Visão. O Apresentador responde às interações do usuário na Visão, solicita dados ou ações ao Modelo e atualiza a Visão conforme necessário. Ele contém a lógica de apresentação da aplicação e coordena as interações entre o Modelo e a Visão.

Onde é usado?

O MVP é particularmente adequado para aplicativos de **UI complexos**, onde é necessária uma extensa ligação de dados, e para projetos que usam estruturas como WPF, UWP, Angular e Xamarin. Com seu forte foco no desenvolvimento de UI, o MVP se tornou popular no mundo do **desenvolvimento móvel**.

Model View Presenter



Padrões de Projeto

MVVM

Model-View-ViewModel

Padrões de Projeto para Web

MVVM - Model-View-ViewModel

- Criado em 2005 por John Gossman, arquiteto do WPF e Silverlight na Microsoft.
- Conceitualmente idêntico ao MVP, mas com uma diferença fundamental: o ViewModel.
- Excelente para aplicativos com uma UI complexa.
- Como benefício adicional, o MVVM facilita a separação de interesses e a manutenção do código.
- Amplamente usado em frameworks como Angular, Vue.js, Knockout.js, etc.

- **Model:** representa os dados e as regras de negócios. Ele é responsável por recuperar, armazenar e processar dados.
- **View:** é a interface do usuário. Ela é responsável por exibir os dados ao usuário e capturar a entrada do usuário.
- **ViewModel:** é um intermediário entre a View e o Model. Ele é responsável por expor métodos, comandos e outras propriedades que ajudam a manter o estado da View, manipular a entrada do usuário e se comunicar com o Modelo.
- **Data Binding:** é o mecanismo que sincroniza automaticamente a View e o ViewModel. Quando os dados no ViewModel mudam, a View é atualizada automaticamente e vice-versa.
- **Send Notifications:** o ViewModel envia notificações para a View quando os dados mudam. Isso permite que a View saiba quando atualizar a interface do usuário.

Model-View-ViewModel

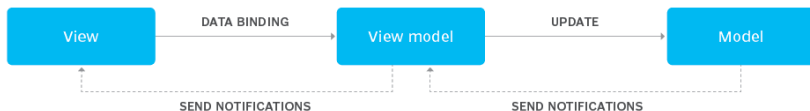
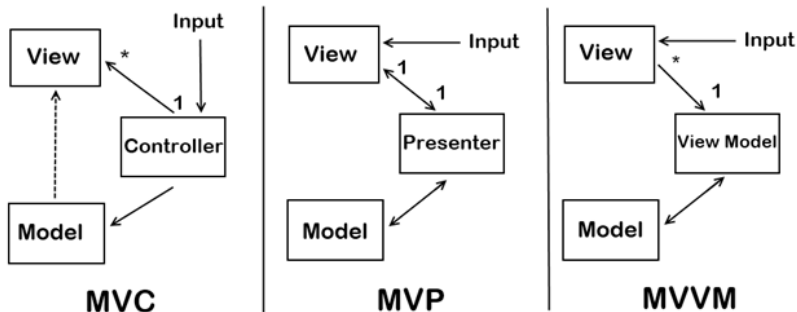


Figura: Estrutura do MVP.

Padrões de Projeto para Web

Conclusão

- Os padrões de projeto MVC, MVP e MVVM são soluções reutilizáveis para problemas comuns de design de software que surgem no desenvolvimento de aplicativos web.
- Eles fornecem diretrizes e estruturas para organizar o código, melhorar a escalabilidade, a manutenibilidade e a eficiência do desenvolvimento.



- 🔗 **Padrão MVC (Model - View - Controller).**
Canal **Cod3r Cursos**.
- 🔗 **MVVM (A Arquitetura de Apps Mobile).**
Canal **Código Fonte TV**.
- 🔗 **MVVM in 100 Seconds.**
Canal **Philipp Lackner**.
- 🔗 **MVC, MVP ou MVVM?**
Canal **ArjanCodes**.
- 🔗 **MVC MVP and MVVM architecture pattern.**
Artigo **itCraft**.
- 🔗 **Understanding MVC, MVVM, and MVP: A Comprehensive Comparison.**
Medium **Gulshan**.

Vamos praticar um pouco o que vimos até agora?

QUIZ - Padrões de Projeto MVC, MVP e MVVM

Padrões de Projeto

Tarefa

Lista de Exercícios

Observações

- As respostas devem ser entregues manuscritas (escritas à mão).
- **Entrega:** antes do início da próxima aula.

- **Questão 01:** Explique resumidamente o que é um padrão de projeto no contexto web.
- **Questão 02:** Explique o padrão de projeto MVC (Model-View-Controller). Destaque as responsabilidades de cada componente (Modelo, Visão e Controlador) e como eles interagem para criar uma aplicação estruturada.
- **Questão 03:** Explique o padrão MVP (Model-View-Presenter) e compare-o com o padrão MVC em termos de responsabilidades e interações entre os componentes. Destaque as vantagens do MVP em cenários específicos.
- **Questão 04:** Como o padrão MVC aborda a testabilidade em uma aplicação? Compare isso com as estratégias de teste no padrão MVP. Destaque os desafios e benefícios associados à testabilidade em ambos os padrões.
- **Questão 05:** Explique como a separação de preocupações é alcançada nos padrões MVC, MVP e MVVM. Destaque como essa separação facilita a manutenção do código e a colaboração entre equipes de desenvolvimento.
- **Questão 06:** Considere um cenário em que a interface do usuário precisa ser atualizada dinamicamente com base em mudanças frequentes nos dados do Modelo. Qual padrão de projeto (MVC, MVP ou MVVM) você escolheria para otimizar a atualização da interface do usuário e por quê?

Padrões de Projeto


Exemplos Práticos

Padrões de Projeto para Web

Exemplo Prático - Sem padrão de projeto

```
1  from flask import Flask, render_template, request, redirect, url_for
2
3  app = Flask(__name__)
4
5  tasks = []
6  task_id_counter = 1
7
8  @app.route('/')
9  def index():
10     return render_template('tasks_no_pattern.html', tasks=tasks)
11
12 @app.route('/add_task', methods=['POST'])
13 def add_task():
14     global task_id_counter
15     description = request.form['task']
16     task = {'id': task_id_counter, 'description': description, 'completed': False}
17     tasks.append(task)
18     task_id_counter += 1
19     return redirect(url_for('index'))
20
21 @app.route('/mark_completed/<task_id>', methods=['POST'])
22 def mark_completed(task_id):
23     for task in tasks:
24         if task['id'] == int(task_id):
25             task['completed'] = not task['completed']
26     return redirect(url_for('index'))
27
28 if __name__ == '__main__':
29     app.run(debug=True)
```

Exemplos Práticos

 Exemplo do padrão MVC com Python/Flask.

■ Experimento 01

- Modifique o exemplo do padrão MVC para adicionar a funcionalidade de **excluir** uma tarefa

■ Experimento 02

- Crie outro exemplo simples de aplicação web utilizando o padrão MVC, MVP ou MVVM para demonstrar o funcionamento do padrão.

Observações

- O experimento 02 deve ser publicados no **GitHub**. No arquivo **README.md**, explique resumidamente o que foi feito e como foi feito.
- **Entrega:** à combinada com o professor.
- **Avaliação:** Faz parte da 1º avaliação.