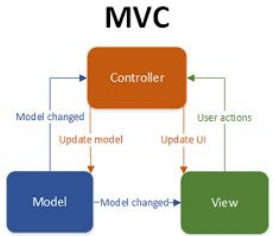
Des. gerência de aplicações Web

Marcelo B. Soares

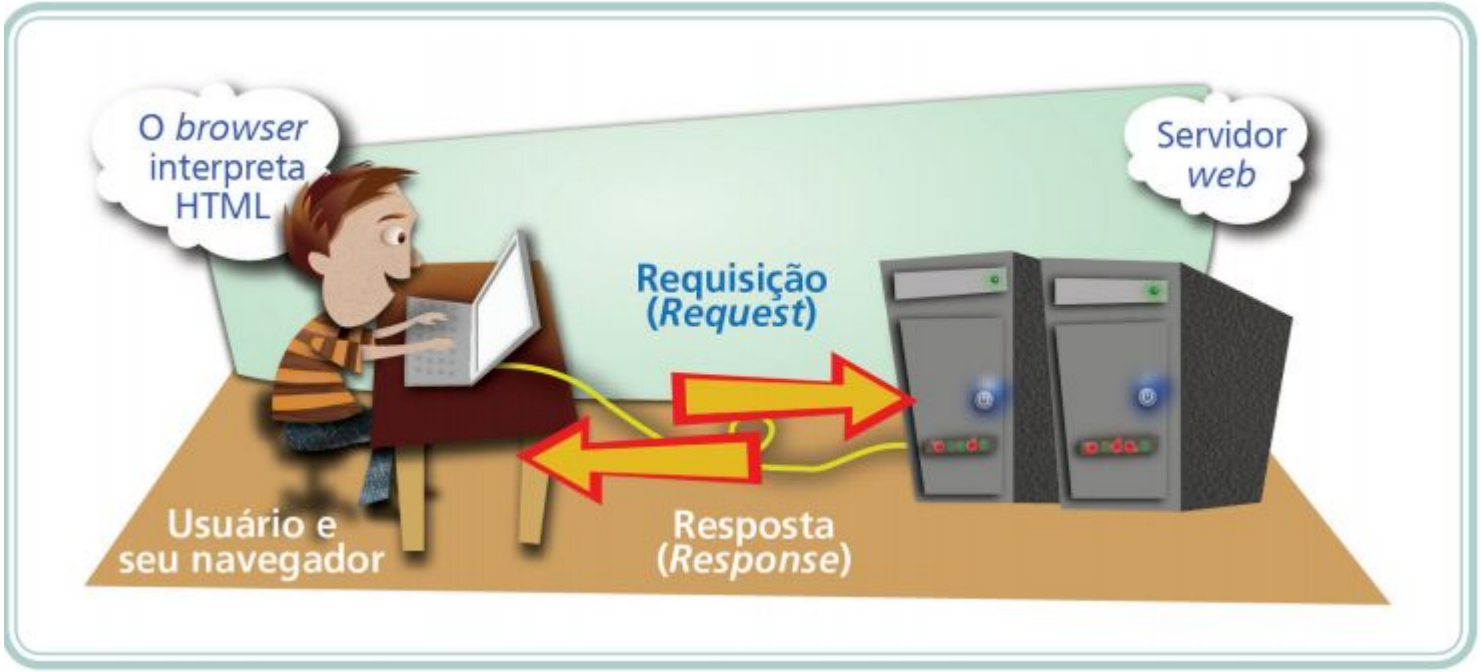
Email: mbsoares.infor@gmail.com

Aula 1

Des. gerência de aplicações Web



Fluxo de acesso ao recurso



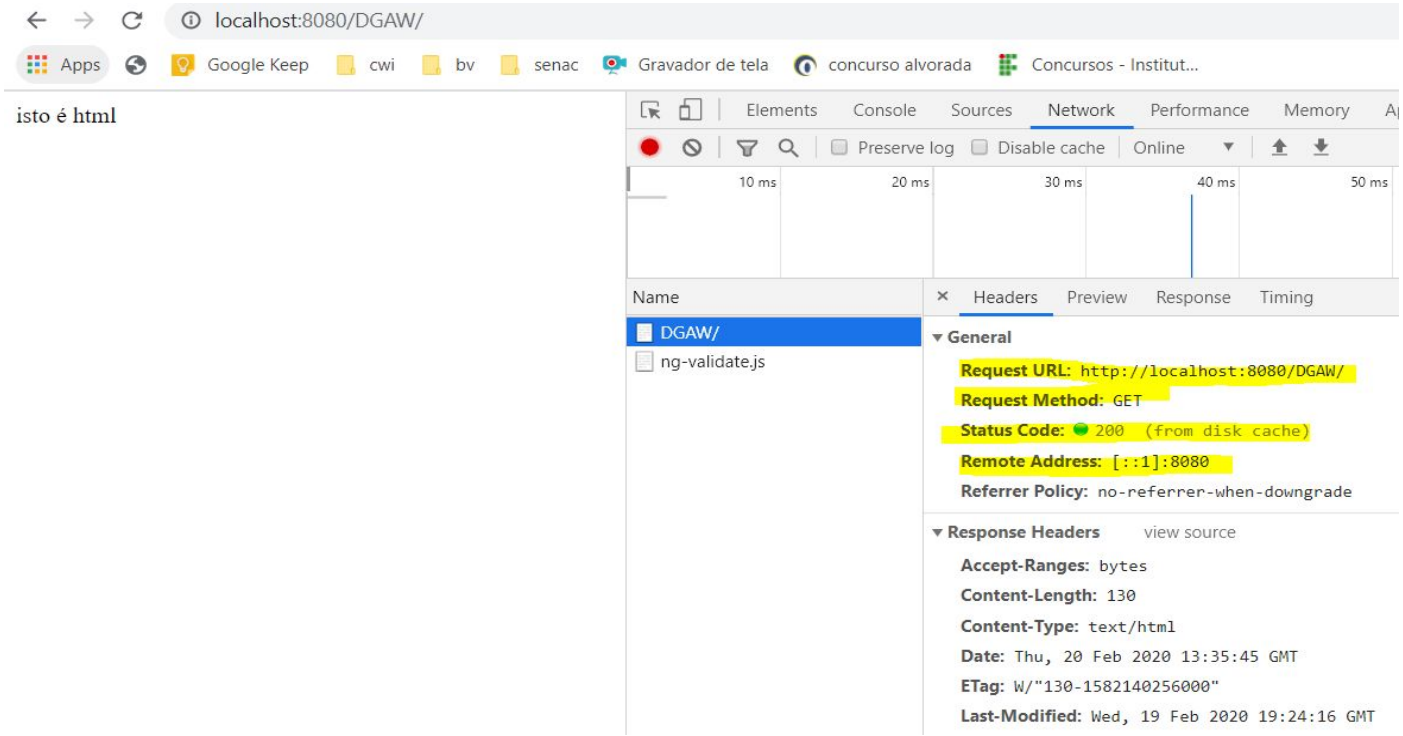
Desenvolvimento Web

- **Importante**
  - A resposta (response), retornada do servidor pode ser codificada em código da linguagem HTML (página), um arquivo, imagem.
  - O cliente entende somente HTML, JavaScript e CSS. E pode abrir os arquivos caso a response seja este tipo de informação.

HTTP

- Um HTTP request leva consigo todos os dados necessários para que o lado do servidor tome a decisão correta sobre o que fazer.
- Podemos usar algumas ferramentas para visualizar essa comunicação entre cliente/servidor, ou seja, a request e response.
- Essas ferramentas são FUNDAMENTAIS PARA O DESENVOLVIMENTO WEB.
- Dentre elas, temos SoapUI, PostMan, ou as que os navegadores disponibilizam.

HTTP



Ferramenta de desenvolvimento

HTTP

- Podemos visualizar diversas informações que o protocolo HTTP envia para o servidor.
- Também podemos visualizar a resposta do servidor.
- Umas das informações que podemos observar, é o método (forma) de acesso que o HTTP está fazendo a request e, o status da response.

HTTP

- O protocolo HTTP, define diferentes meios (métodos) para buscar/enviar informações ao servidor, como por exemplo, POST, GET, DELETE, PUT, HEADER. (Veremos com mais detalhes em futuras aulas)
- O status da response, indica se ocorreu erro, ou não. Por exemplo, 200 indica OK, e 500 error no servidor.
- Segue uma lista de status HTTP: link

MVC

- É um **padrão de arquitetura de aplicações** que divide a aplicação em três camadas: a visão (view), o modelo (model), e o controlador (controller).

- O padrão MVC foi desenvolvido em 1979 por Trygve Reenskaug com a finalidade de ser utilizado como arquitetura para aplicativos desktop. Entretanto, o padrão se popularizou em sistemas web, a partir da adesão de milhares de Frameworks de mercado

MVC

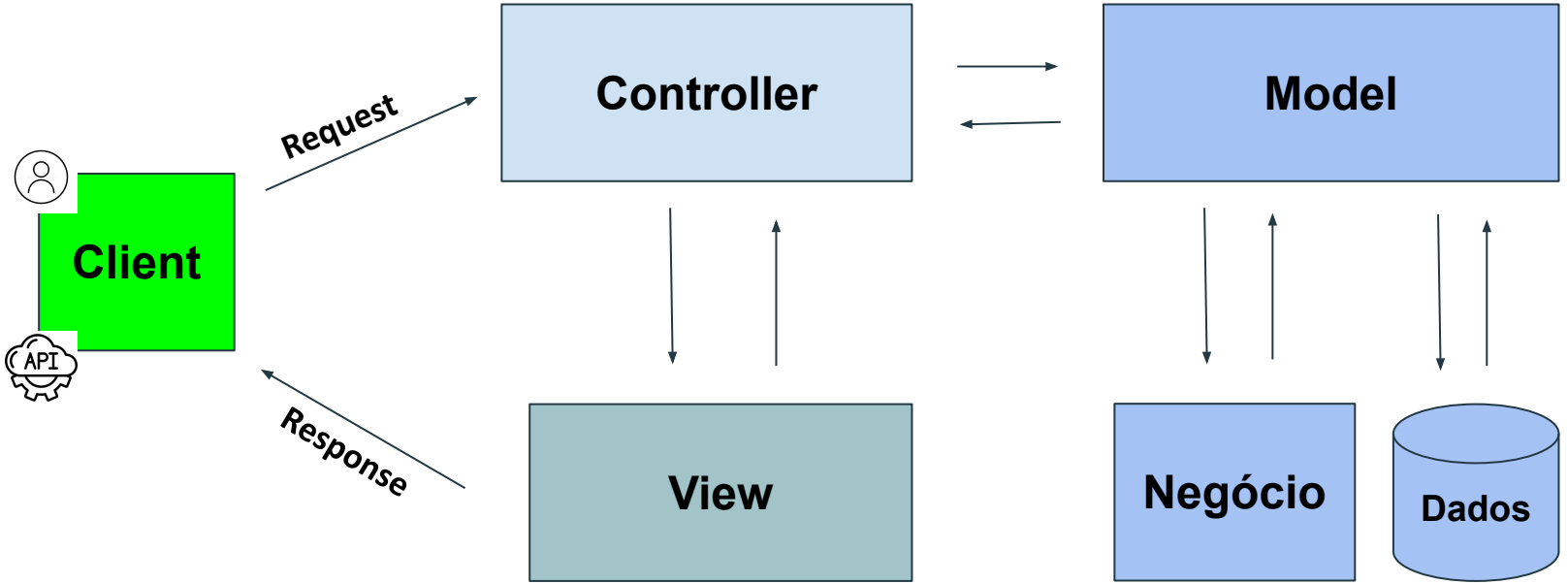
- Cada um destes componentes tem uma função específica e estão conectados entre si.
- O MVC diz como os **componentes da aplicação interagem**.
- Possibilita a divisão do projeto em camadas muito bem definidas. Cada uma delas, o Model, o Controller e a View, executa o que lhe é definido e nada mais do que isso.

MVC

- O uso do padrão MVC traz como benefício o isolamento das regras de negócios da lógica de apresentação, que é a interface com o usuário.

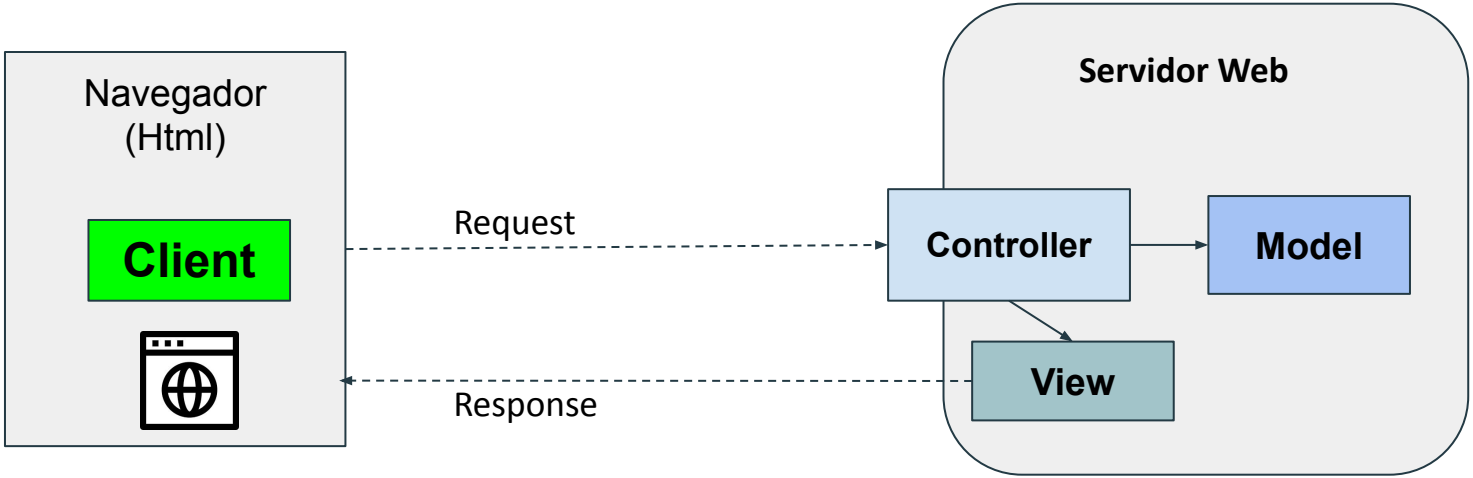
- A interação entre interfaces (View) e regras de negócios (model) é definida através de um controlador, que separa as camadas.

MVC


Client
API
Request
Response
Controller
View
Model
Negócio
Dados

MVC - Páginas Dinâmicas

- Podemos afirmar que MVC enquadra-se dentro das aplicações Web de Páginas Dinâmicas.



MVC - Controller

- É a camada de controle.
- Recebe as requisições do cliente e decide qual model e view a ser chamado. Gerência então qual model e view a ser executado.
- Tem total conhecimento de quais são os model e view existente no sistema

MVC - Controller

- O controller manipula e roteia as requisições dos usuários.
- Valida as requisições dos usuários de acordo com as regras de autenticação e autorização.
- Em resumo, é a controller que executa a model e repassa a informação para a view.

MVC - Model

- É a camada responsável pela manipulação e validação dos dados
- Responsável em acessar o banco de dados
- Responsável por tratar as regras de negócio.
- Responsável em informar ao controller, se a execução foi realizada com sucesso ou não.

MVC - View

- É a camada que transforma os dados em uma representação visual para o usuário final
- Faz a exibição dos dados, utilizando-se HTML por exemplo para aplicações web.
- É responsável por usar as informações modeladas para produzir interfaces de apresentação conforme a necessidade.
- Sua saída é realizada na resposta da requisição.

MVC - E por que usar ?

- Facilita a manutenção da aplicação
- Fácil alteração da interface (View), sem que haja necessidade de modificar a camada de negócio (Model) e vice-versa.
- Melhor desempenho e produtividade, graças à estrutura de pacotes modulares.

MVC - E por que usar ?

- Facilidade de testar a aplicação.
- A arquitetura modular permite aos desenvolvedores e designers trabalharem em paralelo (back/front).
- Partes da aplicação podem ser modificadas sem a necessidade de alterar outras.

MVC

- Podemos implementar MVC, usando alguns frameworks tais como:
  - Struts
  - Spring
  - JSF

Spring

- O spring é um framework que disponibiliza diversas ferramentas que facilitam o dia-a-dia do desenvolvedor. Dentre elas, temos o spring boot, que auxilia muito na criação de aplicações MVC e API´s.

Praticando

- Vamos analisar no projeto prático (enderecoMVC) na classe HelloControllerMVC.