5954024 - Introdução ao Desenvolvimento Web

Perspectiva Histórica da WEB e Fundamentos de seu Desenvolvimento

Prof. Dr. Denis M. L. Martins
DCM | FFCLRP | USP

Objetivos de Aprendizagem

- Compreender o **histórico** do desenvolvimento Web.
- Revisar o **protocolo HTTP** e seus métodos.
- Entender a **infraestrutura** e **arquitetura** geral da Web.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

Por que estudar Web?

- Possivelmente a **maior plataforma distribuída** da história, com bilhões de usuários e requisições por segundo.
- Entender sua evolução ajuda a **projetar sistemas** escaláveis e seguros.
- HTTP/HTML/CSS/Javascript são a **base para diversas aplicações** modernas.

História da Web

Histórico Resumido

- **1989:** Tim Berners-Lee propõe o **World Wide Web (WWW)** como um sistema de hipertexto.
- **1991:** Gropher é criado como protocolo para navegação hierárquica em servidores.
- **1993:** Lançamento do **HTML 1.0** e do primeiro navegador (NCSA Mosaic).
- **2004:** Surgimento da **Web 2.0**, com foco em interatividade e colaboração.
- **2020s:** Emergência da **Web 3.0**, baseada em blockchain e descentralização.

Web 1.0 – A Era Estática (Anos 90–2004)

Características Principais

- **Páginas estáticas:** Conteúdo gerado pelo servidor, sem interatividade.
- **Tecnologias:**
  - HTML 4.0 (1997).
  - CSS 1.0 (1996) para estilização básica.
  - JavaScript (1995), mas limitado a manipulação do DOM.
- **Modelo de negócios:** Sites como cartões de visita digitais.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

A Web 1.0 era como uma biblioteca: você podia ler os livros, mas *não* pode fazer nenhuma modificação neles.

Web 2.0 – A Era da Interatividade (2004–2019)

- **Interatividade em tempo real:** Usuários podem criar, editar e compartilhar conteúdo (ex.: blogs, redes sociais).
  - Introdução de **AJAX** (Asynchronous JavaScript and XML) para atualizações dinâmicas.
- **Tecnologias Chave:**
  - **JavaScript moderno:** jQuery (2006), Node.js (2009).
  - **APIs RESTful** e JSON para trocas de dados.
  - Frameworks como **React (2013)**, Angular, Vue.
- **Exemplos de Aplicações:** Facebook, YouTube.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

Pergunta

Como a Web 2.0 mudou a forma como *interagimos* com sistemas distribuídos? Quais desafios surgiram?"

https://siteefy.com/how-many-websites-are-there

Tendências Atuais (Web 3.0, PWAs)

- Progressive Web Apps (PWAs): Aplicações Web que funcionam como nativas (e.g., Spotify).
- Web 3.0 (Decentralizada): Blockchain, smart contracts e identidade digital.
- **Performance e Acessibilidade:** Lazy loading, CSS Grid/Flexbox, ARIA para inclusão.

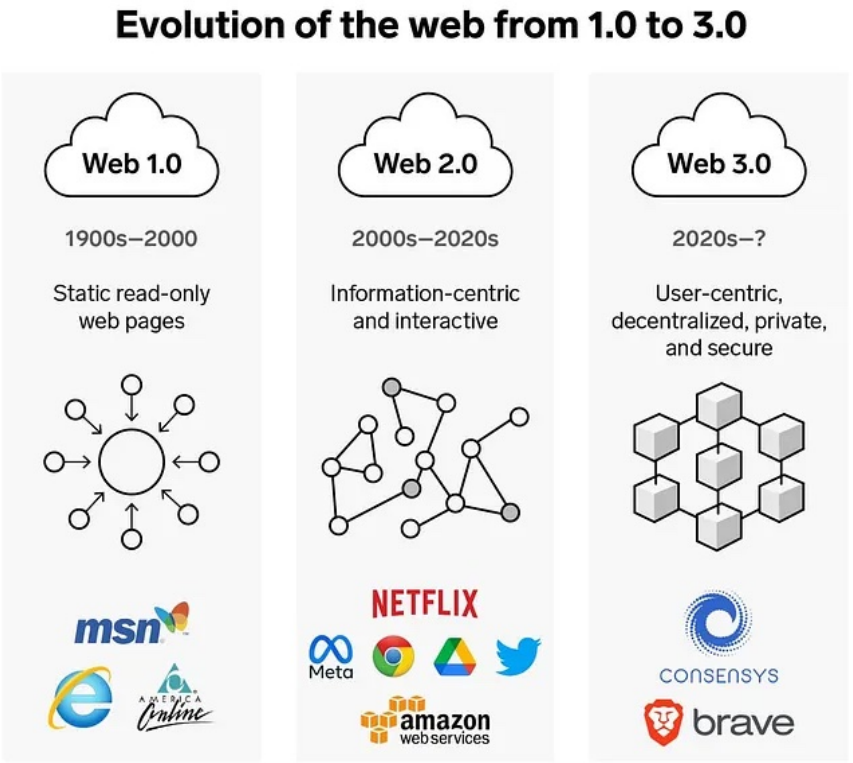

Fig. .1: Evolução da Web. Fonte: POPaPOTEMUS @Medium.

Infraestrutura Básica da Web

Arquitetura Cliente-Servidor

- **Cliente:** Inicia requisições (ex.: navegador, app mobile).
- **Servidor:** Processa requisições e retorna respostas (ex.: backend, API).
- **Comunicação:** Ocorre via protocolos como HTTP/HTTPS.

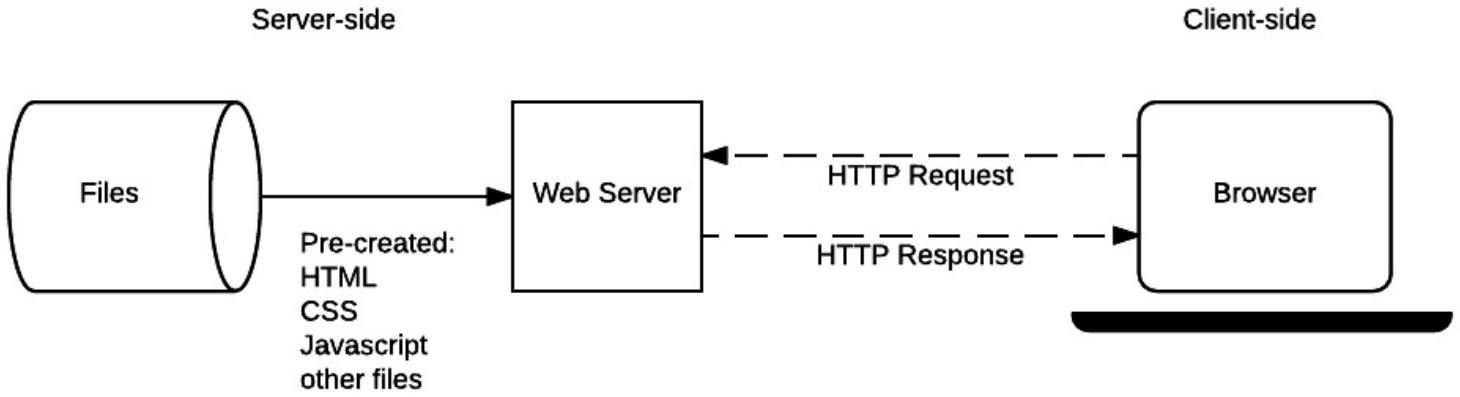

Fig. .1: Arquitetura Cliente-Servidor. Fonte: MDN Client-Server Overview.

Variantes: N-tier, rich/heavy client, thin/lean client.

HyperText Transfer Protocol (HTTP)

- Protocolo cliente-servidor para buscar recursos como documentos HTML.
- É a base de qualquer troca de dados na Web.
- Um documento completo é tipicamente construído a partir de recursos como conteúdo de texto, instruções de layout, imagens, vídeos, scripts e mais.
  - Navegadores modernos criam processos específicos para diferentes partes da interface do usuário (UI).

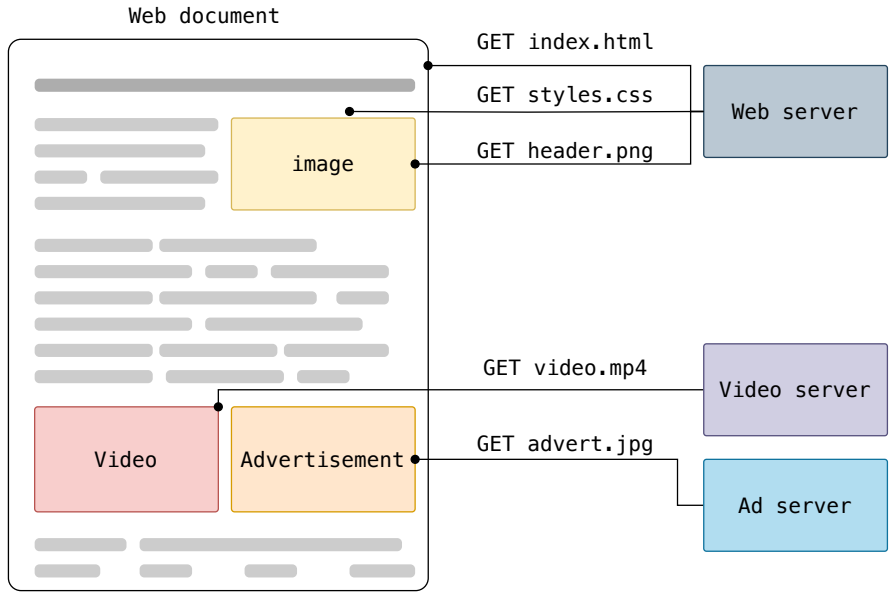

Fig. .1: Documento HTML e requisições HTTP. Fonte: MDN Client-Server Overview.

HTTP e Camadas

HTTP opera na **camada de aplicação** sobre o TCP (Transmission Control Protocol), ou sobre Transport Layer Security (TLS), anteriormente chamada de Secure Sockets Layer (SSL).

> Veja também: Anatomia de um pacote por Julia Evans.

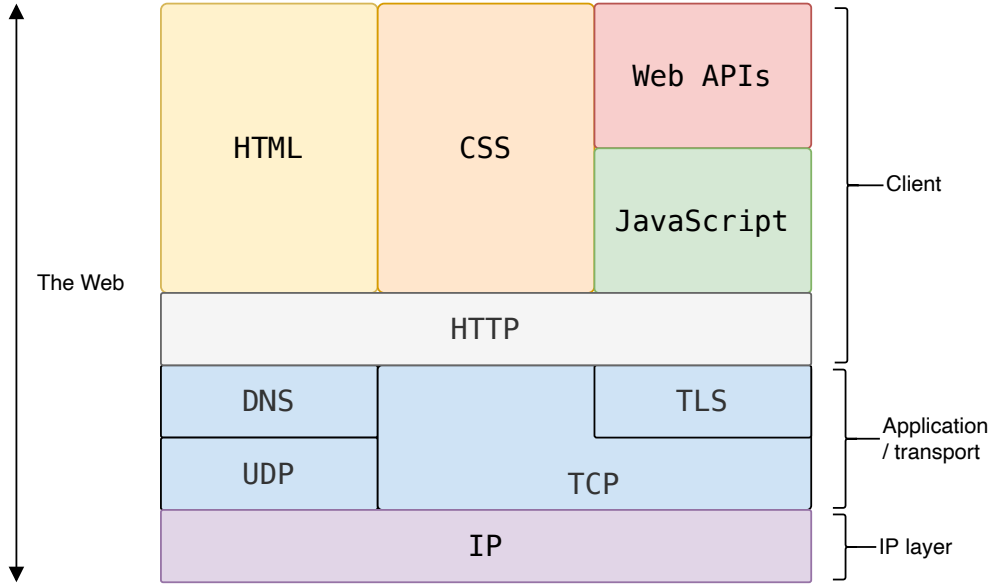

Fig. .1: HTTP e camadas de rede. Fonte: MDN Client-Server Overview.

HyperText Transfer Protocol

Métodos HTTP:

Método	Uso
GET	Buscar dados (sem alteração).
POST	Enviar dados (ex.: formulário).
PUT	Atualizar recurso.
DELETE	Remover recurso.

Status Codes Importantes:

- **200** (OK): Requisição bem-sucedida.
- **404** (Not Found): Recurso não encontrado.
- **500** (Internal Server Error): Erro no servidor.

HTTP é ***stateless***: o servidor esquece o cliente após a requisição.
Cookies são usados para manter dados no cliente e enviá-los de volta ao servidor.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

HTTP Request: Exemplo

```
GET /en-US/search?q=client+server+overview&topic=apps&topic=html&
topic=css&topic=js&topic=api&topic=webdev HTTP/1.1
Host: developer.mozilla.org
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: https://developer.mozilla.org/en-US/
Accept-Encoding: gzip, deflate, sdch, br
Accept-Language: en-US,en;q=0.8,es;q=0.6
Cookie: sessionid=6ynxs23n521lu21b1t136rhbv7ezngie;
csrftoken=zIPUJsAZv6pcgCBJSCj1zU6pQZbfMUAT;
dwf_section_edit=False; dwf_sg_task_completion=False;
_gat=1; _ga=GA1.2.1688886003.1471911953; ffo=true
```

Uniform Resource Identifier (URI)

- String que identifica um recurso (ou entidade)
  - RFC 3986
  - Formato: **<scheme>:<scheme-specific-part>**
- Usado como *target* nas requisições HTTP
- Exemplos:
  - **http : // www.ietf.org /rfc/rfc2396.txt**
  - **ldap : //[2001:db8::7]/c=GB?objectClass?one**
  - **mailto : John.Doe@ example.com**

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

Uniform Resource Locator (URL)

- Endereço de rede que localiza um recurso na web (páginas, imagens, APIs).
- **esquema ou protocolo://domínio:porta/caminho/recurso?query_string#fragmento**
- **Componentes:**
  - Esquema (Protocol): **https://** (HTTP, HTTPS, FTP).
  - Domínio: **www.example.com**.
  - Porta (opcional): **:8080** (padrão: HTTP=80, HTTPS=443).
  - Caminho: **/path/to/resource**.
  - Parâmetros de consulta: **?query=value** (ex.: filtros em buscas).
  - Fragmento: **#fragment** (âncoras para seções na página).

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

HTTP e Parâmetros da URL

As requisições **GET** codificam dados na URL enviada ao servidor adicionando pares chave-valor no final dela.

Exemplo: **http://example.com?name=Fred&age=11**.

- Ponto de interrogação (**?**) separando o resto da URL dos parâmetros
- Sinal de igual (**=**) separando cada nome do seu valor associado
- Um *ampersand* (**&**) separando cada par chave-valor.

Parâmetros de URL podem ser alterados pelos usuários e então reenviados. Não devem ser usados para requisições que atualizam dados no servidor.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

HTTP: Fluxo para sites dinâmicos

Exemplo: Site gerenciador de equipes esportivas. Um treinador escolhe o nome da sua equipe e o número de jogadores em um formulário HTML e receber uma recomendação de "melhor escalação" para o próximo jogo.

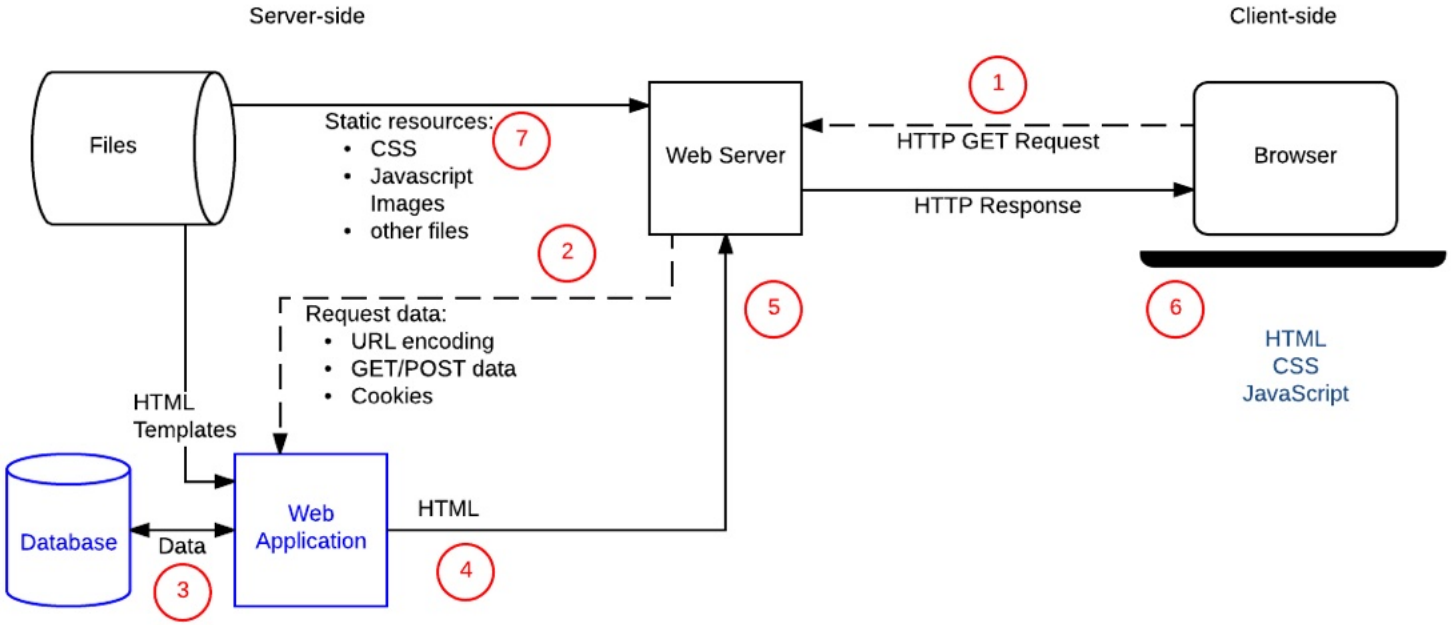

Fig. .1: Anatomia de uma requisição HTTP em sites dinâmicos. Fonte: MDN Client-Server Overview.

HTTP: Fluxo para sites dinâmicos (cont.)

1. Cliente envia requisição HTTP **GET** para o servidor usando a URL base do recurso (**/best**) e passando o nome da equipe e o número de jogadores via URL (ex.: **/best?team=spo&show=11**)
2. Servidor detecta que a requisição é "dinâmica" e a encaminha para a Aplicação Web para processamento. **Como?**
3. Aplicação Web identifica que a intenção da solicitação é obter a recomendação de melhores jogadores com base na URL (**/best/**) e extrai da URL o nome da equipe e o número de jogadores necessários.
4. Aplicação Web cria dinamicamente uma página HTML inserindo os dados (do Banco de Dados) em placeholders dentro de um modelo HTML pré-definido.
5. Aplicação Web retorna a página HTML gerada para o navegador web (via Servidor Web), junto com um código de status HTTP **200** ("sucesso").
6. Navegador Web então começará a processar a HTML retornada, enviando requisições separadas para obter quaisquer outros arquivos CSS ou JavaScript referenciados.
7. Servidor carrega arquivos estáticos do sistema de arquivos e os retorna diretamente para o navegador.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

Alicerces do Desenvolvimento Web

Tecnologia	Papel	Exemplo de Uso
HTML (HyperText Markup Language)	Estrutura semântica: cabeçalhos, parágrafos, listas, links, imagens.	`<h1>Bem-vindo</h1>`
CSS (Cascading Style Sheets)	Apresentação: cores, tamanhos, posicionamento, animações.	`body { background:#f0f8ff; }`
JavaScript	Comportamento dinâmico: eventos, manipulação DOM, requisições AJAX.	`button.onclick = () => alert('Olá!');`

Fluxo típico de uma página web:

Navegador solicita **index.html** → HTML é interpretado → CSS é aplicado → JavaScript roda e pode alterar o Document Object Model (DOM).

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

Quick Tour: Browser Tools

- Navegadores modernos oferecem ferramentas especializadas para verificar HTML, CSS, JavaScript e requisições HTTP.
  - Atalho de teclado no Firefox: **Ctrl + Shift + I** ou **F12**
- Demo rápida: https://www.informationelle-selbstbestimmung-im-internet.de/

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

Conclusão

Conceitos-Chave:

1. **Histórico da Web:** estática (1.0) → interativa (2.0) → descentralizada (3.0).
2. **HTTP:** a base da comunicação cliente-servidor.
3. **HTML, CSS e JavaScript:** alicerce para desenvolvimento front-end.

Material adicional

- MDN Web Docs – Guias: https://developer.mozilla.org/pt-BR/docs/MDN/Guides
- HTTP Request Methods Part 1 and Part 2, HTTP 2 by Julia Evans.
- Documentário do Discovery Channel sobre a Guerra dos Navegadores (um pouco antigo, mas vale a pena assistir):
  - Versão original em inglês: https://www.youtube.com/watch?v=M02uZp_tkAc YouTube
  - Versão dublada: https://www.youtube.com/watch?v=yggAiSFD4Sw YouTube

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.

Dúvidas e Discussão

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br - Created with Marp & MarpX.