# HISTÓRIA DA PROGRAMAÇÃO WEB

Feito por: Guilherme Henrique Almeida da Silva

## Introdução

A programação web é o processo de criação de sites e aplicações web que operam na internet. Isso envolve o uso de várias linguagens de programação, frameworks e ferramentas para desenvolver páginas web interativas e dinâmicas. A programação web se tornou uma parte integrante de nossas vidas cotidianas, e sua história abrange várias décadas.

### Surgimento da internet e do HTML

A **internet** é uma rede global composta por um conjunto de redes de computadores interconectados, distribuídos por todas as regiões do planeta. Esses computadores (ou nós) trocam dados e mensagens entre si utilizando um protocolo comum: o conjunto de protocolos **TCP/IP**. Essa arquitetura permite a **interconexão descentralizada** entre milhões de usuários — sejam eles pessoas físicas, instituições públicas, privadas ou organizações jurídicas.

O surgimento da internet remonta à década de **1960**, quando o **Departamento de Defesa dos Estados Unidos**, por meio da agência **ARPA (Advanced Research Projects Agency)**, iniciou um projeto chamado **ARPANET**. O objetivo era criar um sistema de comunicação **robusto, descentralizado e resistente a falhas**, especialmente em um cenário de guerra — como uma eventual guerra nuclear durante a **Guerra Fria**.

A ideia central era que, mesmo que um ou mais pontos da rede fossem destruídos, os dados ainda pudessem circular por **outros caminhos**, assegurando a continuidade da comunicação. Assim nasceu a **primeira rede de comutação de pacotes**, que se tornou o **embrião da internet moderna**.

A **ARPANET** funcionava com um sistema conhecido como **chaveamento de pacotes** (packet switching), no qual as informações são divididas em pequenos pacotes. Cada pacote contém um trecho dos dados, o endereço do destinatário e metadados que permitem a remontagem da mensagem original no destino. O temido ataque inimigo nunca ocorreu, mas o que o Departamento de Defesa não sabia era que estava dando origem ao **maior fenômeno midiático do século XX** — um meio de comunicação que, em apenas quatro anos, atingiria cerca de **50 milhões de pessoas**.

Um sistema técnico fundamental para o funcionamento da internet é o **Protocolo de Internet (IP)**. Ele é responsável pelo **endereçamento** e **roteamento** de pacotes de dados entre dispositivos conectados à rede. Isso significa que o IP

identifica cada equipamento e determina o caminho que os dados devem percorrer para chegar ao destino correto, mesmo em redes complexas e descentralizadas.

Abaixo seguem os principais parâmetros e características do protocolo:

O **endereço IP** é um número único atribuído a cada dispositivo conectado à internet (como computadores, smartphones e roteadores). Ele funciona como um "endereço residencial digital", permitindo identificar quem envia e quem recebe as informações.

Existem dois formatos principais de endereço IP:

- **IPv4 (Internet Protocol version 4)**: formato tradicional com quatro blocos de números (ex: 192.168.0.1);

- **IPv6 (Internet Protocol version 6)**: versão mais moderna, com maior capacidade de endereçamento (ex: 2001:0db8:85a3:0000:0000:8a2e:0370:7334).

O IP não apenas identifica os dispositivos, mas também define **como os pacotes de dados devem viajar pela rede**, saltando entre roteadores até alcançar o destino. Esse processo é chamado de **roteamento**. Cada pacote pode tomar rotas diferentes, e o IP garante que todos sejam **remontados corretamente** no destino.

Cada **pacote de dados** inclui:

- Endereço IP de origem (quem está enviando);

- Endereço IP de destino (quem deve receber);

- Informações para controle e reconstrução do pacote na outra ponta.

O funcionamento do IP está diretamente relacionado aos **modelos de arquitetura de rede**, especialmente o modelo **TCP/IP** e o modelo **OSI (Open Systems Interconnection)**. Esses modelos dividem a comunicação em **camadas**, para facilitar o entendimento, o desenvolvimento e a padronização dos protocolos.

## Modelo TCP/IP (modelo da Internet)

O IP faz parte do modelo TCP/IP, que possui quatro camadas:

1. **Camada de Aplicação** – onde estão os protocolos usados por aplicativos (HTTP, FTP, SMTP);

2. **Camada de Transporte** – garante a entrega dos dados (ex: TCP e UDP);

3. **Camada de Internet** – onde atua o Protocolo IP, responsável pelo roteamento e endereçamento;

4. **Camada de Acesso à Rede** – define como os dados são fisicamente enviados (por cabos, Wi-Fi etc.).

## Modelo OSI (referência acadêmica)

O modelo OSI, com sete camadas, serve como base teórica para compreender a comunicação em rede. O **Protocolo IP atua na Camada 3 – Camada de Rede**, que tem como função principal o **endereçamento lógico** e o **roteamento entre redes diferentes**.

Portanto, o **Protocolo IP** é o que garante que os dados enviados pela internet **cheguem corretamente ao destino**, mesmo que atravessem diversos intermediários. Ele atua de forma invisível para o usuário, mas é essencial para que possamos **acessar sites, enviar mensagens e consumir conteúdo online** com confiabilidade.

Já o **HTML (HyperText Markup Language)** surgiu nos anos **1990**, com o objetivo de **estruturar e formatar documentos** da recém-criada **World Wide Web (WWW)** — um serviço que utiliza a internet como meio de transmissão. A Web permitia aos usuários acessar e compartilhar documentos interconectados, chamados **hipertextos**.

**Observação: Veja que o HTML é uma linguagem de marcação de texto e não uma linguagem de programação propriamente dita.**

O **hipertexto** é uma forma de organização não linear da informação, onde conteúdos como textos, imagens e vídeos são ligados entre si através de **hiperlinks**, criando uma experiência de navegação fluida. Essa ligação entre documentos é realizada por meio do protocolo **HTTP (HyperText Transfer Protocol)**.

A Web revolucionou o acesso à informação, tornando possível a publicação de conteúdo acessível globalmente. As primeiras páginas eram documentos simples, contendo apenas texto e links. Em **1991**, foi lançada a **primeira página da web**,

marcando o início de uma nova era de comunicação e compartilhamento de informações.

Além da Web, diversos serviços e protocolos foram desenvolvidos para ampliar as funcionalidades da internet. Entre os principais, destacam-se:

- **Telnet** – Acesso remoto a computadores;
- **FTP (File Transfer Protocol)** – Transferência de arquivos;
- **POP e SMTP** – Envio e recebimento de e-mails;
- **P2P (Peer-to-Peer)** – Compartilhamento direto de arquivos entre usuários;
- **Chats** – Comunicação em tempo real (mensagens instantâneas).

Naquela época, as páginas web eram estáticas e careciam de elementos interativos, com base neste problema, pelos web sites serem bastantes estáticos, surgiu a necessidade do dinamismo e da estilização surgiu a introdução de **CGI, JavaScript, CSS e linguagens server-side** como PHP, ASP, JSP, entre outras.

## Evolução para Web Dinâmica: CGI, Scripting e Interatividade

Durante os primeiros anos da Web, os sites eram essencialmente **documentos HTML estáticos**. Isso limitava sua funcionalidade, pois não havia possibilidade de interações complexas com o usuário nem de alterações de conteúdo sem a intervenção manual do desenvolvedor.

Com o crescimento da demanda por **conteúdo dinâmico**, surgiram tecnologias que possibilitaram a geração de páginas **em tempo real**, baseadas em ações do usuário ou em dados vindos de bancos de dados. Foi nesse cenário que nasceu o **CGI (Common Gateway Interface)**

### CGI (Common Gateway Interface)

O CGI foi um dos primeiros mecanismos para criar páginas dinâmicas. Ele permite que um servidor web execute scripts externos, normalmente escritos em linguagens como Perl, Python ou C, para gerar conteúdo HTML dinâmico.

- Funcionamento: quando um usuário acessa uma URL específica, o servidor executa o script CGI e retorna o resultado como uma página HTML.

- Limitação: cada requisição CGI gera um novo processo no servidor, o que não é escalável para aplicações com grande volume de acesso.

### Scripting do Lado do Servidor

Para superar as limitações do CGI, surgiram linguagens e plataformas mais eficientes para gerar conteúdo dinâmico diretamente no servidor:

- PHP (1995) – linguagem simples e amplamente adotada, embutida diretamente no HTML;

- ASP (Active Server Pages) – da Microsoft, introduziu um modelo baseado em componentes COM;

- JSP (JavaServer Pages) – versão Java para páginas web dinâmicas com integração à plataforma Java EE.

Essas linguagens permitiram a criação de sistemas complexos, como e-commerces, sistemas de login, gestão de conteúdo e integração com bancos de dados relacionais como MySQL, PostgreSQL e SQL Server.

## Interatividade no Cliente: HTML Dinâmico, JavaScript e CSS

Enquanto os scripts do lado servidor cuidavam da lógica e da geração de conteúdo, surgiu a necessidade de tornar as páginas **mais interativas** no navegador do usuário — sem recarregar toda a página.

### JavaScript (1995)

Criado inicialmente pela Netscape, o **JavaScript** permitiu a execução de scripts no navegador, possibilitando validações de formulários, interações com o DOM (Document Object Model), atualizações dinâmicas de conteúdo e resposta a eventos de usuário (como cliques e teclas)

Com o tempo, o JavaScript evoluiu e passou a permitir experiências muito mais ricas, especialmente com a introdução de:

- AJAX (Asynchronous JavaScript and XML) – possibilita a comunicação com o servidor sem recarregar a página;

- Bibliotecas e frameworks como jQuery, Angular, React e Vue.js, que facilitaram o desenvolvimento front-end moderno.

### CSS (Cascading Style Sheets)

Introduzido em 1996, o CSS trouxe uma forma padronizada de definir estilos (cores, fontes, layouts, espaçamentos) separadamente do conteúdo HTML. Isso promoveu a separação de responsabilidades (markup vs estilo) e o reuso de regras de formatação.

## Convergência e Avanço: Web 2.0

A partir dos anos 2000, com a popularização da **Web 2.0**, os sites passaram a oferecer maior interação com o usuário, colaboração em tempo real (como redes sociais e wikis) e interfaces altamente responsivas.

As aplicações web tornaram-se verdadeiros softwares, dando origem aos chamados **SPA (Single Page Applications)** e aos **Progressive Web Apps**, que oferecem experiências próximas às de aplicativos nativos.

## Arquiteturas modernas: APIs, Microserviços e Backend evoluído

À medida que os sistemas web se tornaram mais complexos, houve uma separação clara entre **frontend (interface)** e **backend (lógica, dados e regras de negócio)**. Isso deu origem à chamada **arquitetura desacoplada** — ou arquitetura orientada a serviços —, com foco em **escalabilidade**, **manutenibilidade** e **distribuição**.

### APIs (Application Programming Interfaces)

As **APIs** se tornaram a principal forma de comunicação entre o frontend e o backend. Ao invés de retornar HTML, o servidor retorna **dados estruturados**, geralmente em **JSON**, que são consumidos e renderizados no cliente.

Duas abordagens populares para APIs:

- **REST (Representational State Transfer)** – arquitetura leve baseada em recursos acessados por URLs, amplamente adotada;

- **GraphQL** – linguagem de consulta desenvolvida pelo Facebook, que permite ao cliente escolher exatamente os dados que quer receber.

### Microserviços

Com a evolução das necessidades de negócios, o modelo monolítico (todo o sistema em um único bloco) começou a dar lugar ao **modelo de microserviços**,

onde cada funcionalidade é dividida em serviços menores, independentes, e que se comunicam entre si via APIs.

Vantagens dos microserviços:

- **Escalabilidade individual** por serviço;
- **Facilidade de deploy contínuo**;
- **Adoção de múltiplas linguagens e tecnologias por serviço** (ex: Node.js no auth, Python no analytics, Go em filas).

Ferramentas e práticas como **Docker, Kubernetes, CI/CD, mensageria com RabbitMQ/Kafka** também se tornaram comuns nesse contexto.

## Frontend moderno: SPAs, Frameworks e SSR

Com o crescimento da lógica do lado cliente, surgiram os **frameworks e bibliotecas JavaScript** que revolucionaram a forma como interfaces são construídas:

### SPAs (Single Page Applications)

Uma SPA carrega uma única página HTML e atualiza dinamicamente o conteúdo conforme o usuário navega, sem recarregar a página por completo.

Frameworks populares para SPAs:

- **React** (Facebook) – baseado em componentes e Virtual DOM;
- **Vue.js** – progressivo e fácil de adotar;
- **Angular** (Google) – framework completo, com TypeScript;
- **Svelte** – compilador que gera código altamente otimizado.

Essas ferramentas permitiram construir **interfaces reativas, modulares, performáticas e altamente escaláveis**.

**SSR (Server-Side Rendering) e SSG (Static Site Generation)**

Apesar da eficiência das SPAs, elas enfrentam problemas com **SEO** e **primeira renderização lenta**. Para resolver isso, surgiram abordagens híbridas:

- **SSR (Next.js, Nuxt.js)** – renderiza páginas no servidor antes de enviá-las ao cliente;
- **SSG (Static Site Generators)** – como Gatsby ou Astro, que geram HTML estático no build, otimizando performance e SEO.

## Cloud Computing, DevOps e a Web escalável

A **cloud computing** mudou a forma como aplicações web são hospedadas e escaladas. Com a nuvem, deixou-se de depender de servidores físicos para adotar **infraestrutura elástica sob demanda**.

Principais provedores:

- **AWS** (Amazon Web Services)
- **Google Cloud Platform**
- **Microsoft Azure**

Serviços populares:

- **Lambda Functions / Functions as a Service** (serverless)
- **Firebase** (backend as a service para apps)
- **S3 + CDN** para entrega estática

A cultura **DevOps** surgiu como uma resposta à necessidade de integrar desenvolvimento e operações de forma contínua e automatizada. Ferramentas como **GitHub Actions, GitLab CI/CD, Jenkins** e **Docker** fazem parte do arsenal moderno de deploy.

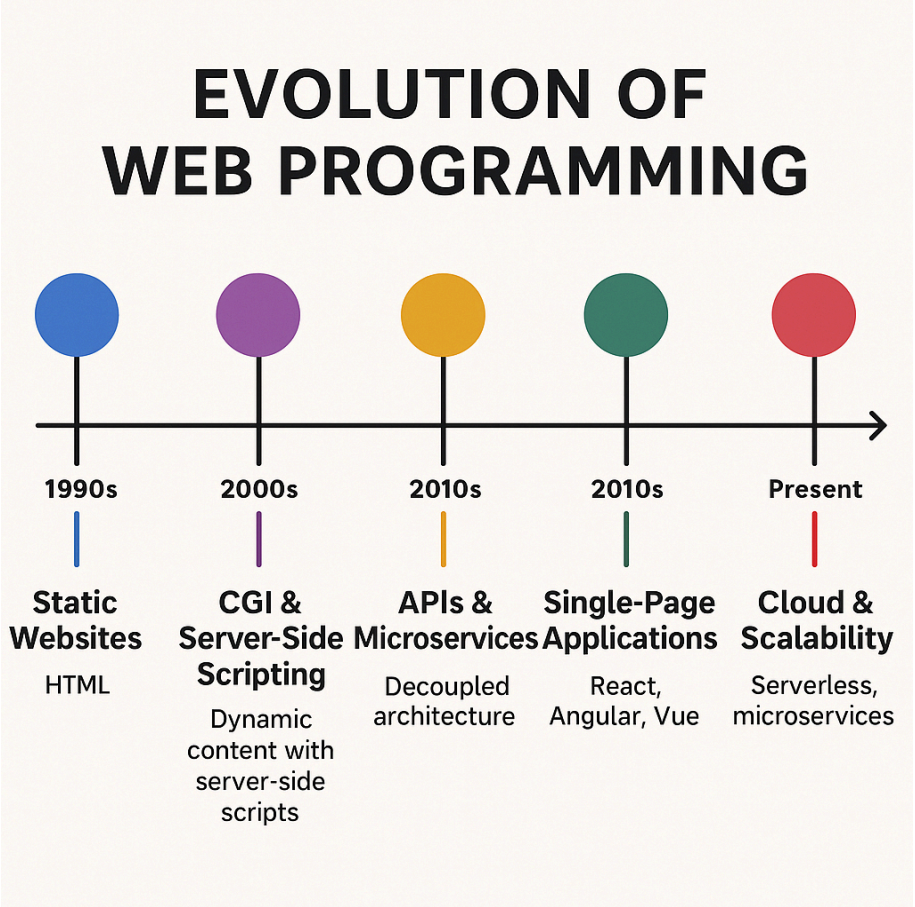
## Considerações Finais

A **programação web evoluiu de documentos estáticos em HTML simples para sistemas complexos distribuídos**, rodando em múltiplas linguagens, plataformas e em escala global. Hoje, um desenvolvedor web precisa dominar:

- **Frontend:** HTML, CSS, JavaScript, frameworks modernos;
- **Backend:** APIs, segurança, autenticação, bancos de dados;
- **DevOps:** deploy, monitoramento, escalabilidade;
- **Cloud e arquiteturas modernas:** serverless, microserviços, CI/CD.

A web se tornou um verdadeiro **ecossistema multifacetado**, em constante transformação, impulsionada por inovação, novas demandas e usuários cada vez mais exigentes.

## Linha do Tempo



## Fontes


A História da Programação Web | html | NetCoders
A História do Desenvolvimento Web: Do HTML aos Frameworks Modernos | html | NetCoders
História da Internet – Wikipédia, a enciclopédia livre
O que é a Internet (conceito, definição e história) - Significados
World Wide Web – Wikipédia, a enciclopédia livre
Pesquisas feitas em IA no chatgpt