FACULDADE IMPACTA

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

DESENVOLVIMENTO WEB

ALEX SOUSA

SÃO PAULO - 04/2024

SUMÁRIO

Desenvolvimento Web

Sites de referência

W3Schools

W3Schools é um site educacional voltado ao aprendizado de tecnologias Web (HTML, CSS, JavaScript) e outras linguagens de programação. Pode ser usado tanto para consultas de tags e atributos HTML, seletores e propriedades CSS, etc. Além disso possui exercícios e vários exemplos interativos;

https://www.w3schools.com/

https://www.w3schools.com/exercises/

MDN Web Docs

MDN Web Docs é uma fonte de documentação para desenvolvedores, mantida com o apoio de uma comunidade de desenvolvedores e escritores técnicos, além de hospedar muitos documentos sobre uma grande variedade de assuntos como: HTML, CSS, JavaScript, etc.

https://developer.mozilla.org/pt-BR/

Livros introdutórios

HTML e CSS

FREEMAN, E.; FREEMAN, E. Use a cabeça! HTML com CSS e XHTML. 2. ed. Rio de Janeiro: Alta Books, 2015.

JavaScript

DUCKETT, J. Javascript e JQuery: desenvolvimento de interfaces web interativas. 1.ed. Rio de Janeiro: Altabooks, 2016.

Framework Flask

GRINBERG, M. Flask web development: developing web applications with python. 2. ed. O'Reilly Media, 2018.

o que é internet

Grande rede de computadores interligadas, redes de outras redes. vários dispositivos de redes interligados.

Para que esta rede se comunique precisa de um protocolo de comunicação conjunto de regras para troca de informações.

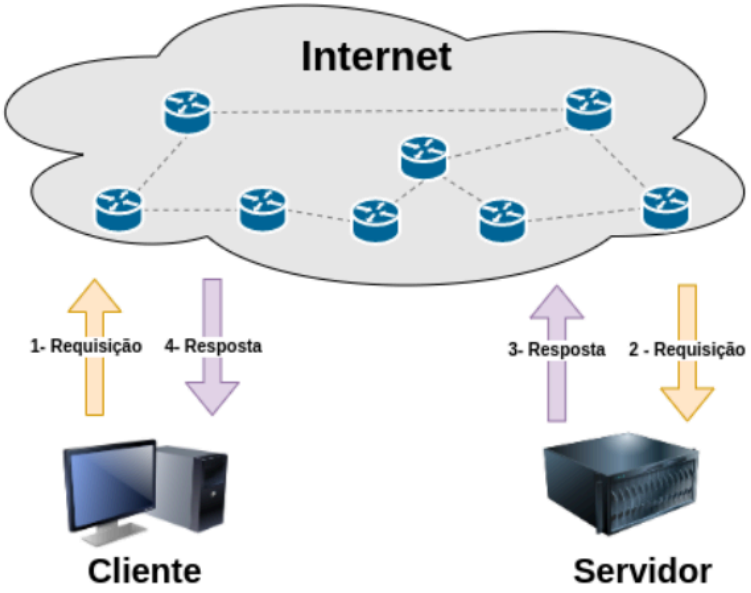
A Internet é formada por diversas redes de computadores conectadas entre si, trocando informações de todo tipo a todo instante. De uma simples mensagem de e-mail até o streaming de um vídeo, temos computadores (além de outros dispositivos conectados) trocando mensagens de forma padronizada e organizada.

Mas como essa troca se dá? Quais tecnologias estão envolvidas? Sabemos que tudo começa no navegador de Internet, o software que todos os computadores e dispositivos móveis possuem para entrar nos sites da Internet. Entender as partes envolvidas nesse processo, mesmo que de forma mais superficial, nos permite ter uma compreensão muito importante para o desenvolvimento de aplicações na web, seja qual for o foco desta aplicação.

Modelo Cliente-Servidor

Em termos gerais, essa troca de mensagens é feita sempre através de um aparelho que requisita (solicita) alguma informação e outro que responde a essa solicitação. Essa forma de comunicação, onde um ponto requisita uma informação e outro responde com ela é representada pelo modelo cliente-servidor [Maia, 2013], conforme pode ser visto na Figura 1.1.

Figura 1.1. Modelo Cliente-Servidor



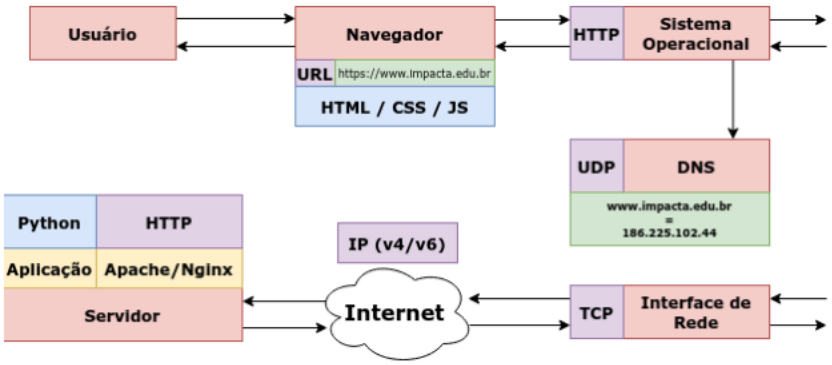
O modelo cliente-servidor abrange uma variedade de sistemas além da web, como por exemplo os caixas eletrônicos, pontos de venda em lojas, sistemas de e-mails, etc. Na Internet, costumeiramente, o cliente será o navegador de Internet e o servidor algum computador que hospeda páginas e outros recursos (imagens, vídeos, arquivos Word ou PDF, arquivos com código CSS, etc) que podem ser solicitados pelo navegador (cliente).

Caminho de uma requisição

Ao digitarmos o endereço do site que queremos acessar no navegador, a requisição feita por ele passa por diversos componentes, tecnologias e protocolos antes de podermos ver o resultado. Entender o papel de alguns destes pedaços ajuda a perceber a complexidade envolvida na construção da web e a construir aplicações com foco no funcionamento do sistema completo.

Vamos simplificar o caminho com os conceitos mais importantes para o desenvolvimento web, mostrados no diagrama da Figura 1.2.

Figura 1.2. Caminho da requisição



Fonte: do autor, 2021.

Tudo começa com o usuário manipulando o navegador para pedir algum recurso (em geral uma página web). Para isso, ele digita o endereço do recurso que ele está procurando na barra de endereço do navegador, usando o formato da URL. Com isso, o navegador empacota o pedido para o sistema operacional, usando o protocolo HTTP como o pacote da mensagem.

Caso o navegador não saiba o IP (Internet Protocol) do domínio digitado (se for a primeira vez que visita o site, por exemplo), o sistema operacional pergunta ao servidor de DNS (Domain Name System), que é o serviço responsável por traduzir domínios para IPs públicos na Internet. Essa conexão tradicionalmente é feita usando o protocolo UDP (User Datagram Protocol) que é um protocolo de transporte de mensagens característico por ser mais rápido por não checar se a informação chegou ou não ao destinatário. O IP é um protocolo de endereçamento na Internet, formado por números de 32 bits (IPv4) ou 128 bits (IPv6).

Após descobrir o IP público do servidor que possui o site pedido, o sistema operacional passa a mensagem para a interface de rede que irá estabelecer uma conexão com o servidor que possui o IP solicitado. A partir daqui, esses dados trafegam pela Internet usando protocolo TCP (Transfer Control Protocol) que, ao contrário do UDP, sempre verifica se a mensagem de fato chegou ao destinatário, fazendo com que seja um protocolo mais robusto (mas com uma perda no desempenho quando comparado com o UDP).

Com o IP e a conexão TCP estabelecida, a requisição HTTP chega ao servidor. Lá, ela precisa ser desempacotada por algum programa que entenda o protocolo HTTP, nesse caso chamado de servidor de aplicação HTTP, onde os exemplos mais famosos são o Apache, Nginx (pronuncia-se “engine-X”) e o Microsoft IIS. Na maioria dos casos esses softwares apenas traduzem a mensagem e repassam para alguma outra aplicação, que irá executar as ações necessárias para retornar a resposta para o cliente.

Com a resposta já formada, o servidor de aplicação HTTP a empacota para o retorno ao cliente (passando por todos os pontos novamente). Chegando no navegador (cliente), ele usa seus recursos para interpretar a resposta e mostrar o resultado ao usuário. Em geral, esses recursos são escritos usando as linguagens que o navegador consegue interpretar, como por exemplo o HTML, CSS e JavaScript.

Você sabia?

A Internet é o local com a maior quantidade de conhecimento aberto do mundo, mas nem sempre foi assim. O conceito foi criado pelo Departamento de Defesa dos EUA, no contexto da Guerra Fria ainda (década de 1960). A ideia era transferir documentos secretos entre pontos estratégicos, descentralizando as informações entre

vários locais. A subdivisão deste departamento que criou o conceito foi a ARPA (Advanced Research Projects Agency), por isso a primeira versão se chamava ARPANET (ESCOLA, s.d.).

Estrutura de uma requisição

A requisição no HTTP é formada basicamente por três grandes partes: a URL, o método HTTP e os cabeçalhos de requisição. Algumas requisições podem possuir um corpo (payload), que abriga dados adicionais enviados ao servidor.

Estrutura da URL

A URL (Uniform Resource Locator) é um sistema de endereçamento de recursos para web. Ela traz informações de localização do servidor na web, do recurso dentro do servidor, e qual protocolo deve ser utilizado para trazer o recurso. A sua estrutura básica e simplificada é:

esquema://domínio:porta/caminho/recurso?query_string#fragmento

O esquema mostra qual protocolo será utilizado na transmissão da mensagem. Dentre os mais comuns, temos como exemplo: http, ftp, smtp, etc.

O domínio é o conjunto de nomes e identificadores mais amigáveis aos humanos para computadores pela Internet. Um domínio em geral corresponde apenas a um endereço IP, embora existam técnicas para permitir múltiplos endereços. Tradicionalmente chamamos de domínio a parte do endereço que compõe o nome do negócio representado (ex: impacta) e os identificadores de negócio e local (.edu para educação e .br para sites no Brasil). Qualquer prefixo na frente do domínio principal (impacta.edu.br) é chamado de subdomínio. Por exemplo: www.impacta.edu.br e account.impacta.edu.br são subdomínios de impacta.edu.br.

A porta é um número inteiro que identifica o caminho lógico por onde uma comunicação de rede está passando. Toda comunicação de rede (e processos nos computadores) passam por uma porta lógica. Por padrão, o HTTP sempre utiliza a porta 80 (ou 443 para o HTTPS, uma versão do HTTP que utiliza criptografia). No navegador, sempre que se utiliza o HTTP e o HTTPS não é necessário escrever suas portas padrão, pois nesses casos ele já utilizará as portas 80 ou 443, implicitamente.

O caminho (também conhecido como path) identifica onde o recurso está dentro do servidor. Este caminho pode ser tanto físico (indicando pastas no servidor) quanto lógico (caminhos configurados dentro da aplicação para encontrar um recurso).

O recurso é o arquivo que se deseja acessar. Nem sempre essa parte da URL vai existir, pois alguns recursos são dinâmicos, ou seja, são gerados em tempo de execução da requisição (ex: lista de chamada de uma turma no dia). Quando o recurso for estático (ex: imagens) essa parte existirá.

A query string é uma forma de passar dados a mais para o servidor, como forma de ajudar na busca de recursos mais específicos. É bastante usada nos buscadores e possui o formato atributo=valor. Essa parte da URL é análoga ao envio de valores através de parâmetros para uma função em uma linguagem de programação, isto é, podemos enviar valores quaisquer para o servidor através da URL.

O fragmento identifica uma parte específica da página que está sendo procurada, em geral um id (identificador único) de algum elemento HTML para que o navegador já entre visualizando o elemento específico.

Métodos HTTP

Os métodos HTTP (ou verbos HTTP) são informações que servem para indicar uma intenção do que pode ocorrer no servidor ao enviar a requisição. Alguns métodos são usados apenas para obtenção de recursos (ex: GET), outros são específicos para alteração do estado da aplicação (ex: PUT, DELETE, POST) (HTTP request methods, s.d.).

Dentre os métodos HTTP mais comuns, destacamos:

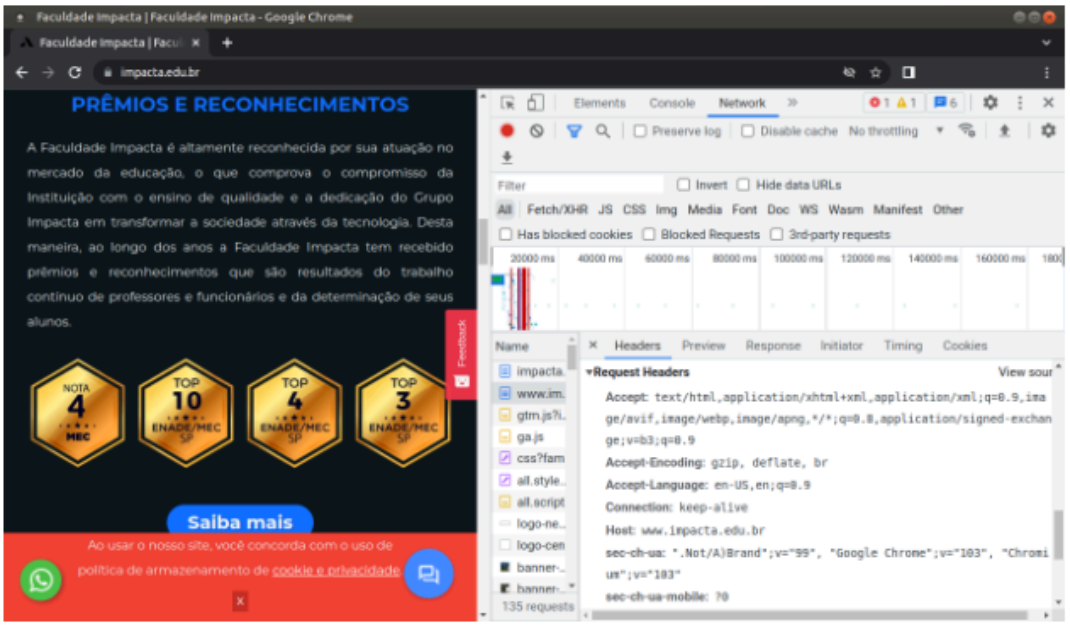
- GET: usado para obter um recurso qualquer do servidor;
- POST: envia dados para serem processados pelo servidor, em geral para criar ou alterar um recurso;
- DELETE: remove um determinado recurso do servidor;
- PUT: atualiza todas as informações de um recurso no servidor;
- PATCH: atualiza parte das informações de um recurso no servidor;

Cabeçalhos

Os cabeçalhos (headers) de requisição são uma série de informações necessárias para a comunicação entre o cliente e o servidor. Essas informações trafegam sempre no formato chave: valor, onde os valores são sempre tratados como dados textuais.

Algumas dessas informações são restritas ao navegador, como por exemplo o user-agent e os cookies. Também é possível criar nossos próprios cabeçalhos com bibliotecas JavaScript para passar informações específicas de nossa aplicação. A Figura 1.3 mostra alguns exemplos de informações contidas no cabeçalho de uma requisição. Essa mesma tela pode ser acessada no menu “Ferramentas do Desenvolvedor”, ou usando o atalho F12 (ou também Control+Shift+I) no navegador Google Chrome.

Figura 1.3. Visualização do cabeçalho de uma requisição usando o navegador Google Chrome



Os cookies são headers mais específicos, pois eles são a única informação que pode sobreviver entre uma requisição e outra. Eles são valores textuais com um nome e o domínio onde eles são aplicados. Toda requisição feita no mesmo domínio carrega todos os cookies registrados nele. O uso dos cookies é uma das maneiras mais clássicas de criar uma sessão de usuário na web. Estudaremos a relação entre sessões de usuário e cookies mais adiante em nosso curso.

Estrutura de uma resposta

As respostas HTTP possuem sempre três partes: código de status, cabeçalhos de resposta e corpo da resposta.

Códigos de status

Os códigos de status indicam se a requisição foi sucedida ou não, e o que aconteceu com ela em ambos os casos. Eles são divididos em números que podem ir de 100 a 599, mas divididos em faixas de 100 de acordo com o seu objetivo. Alguns dos códigos de status mais famosos são o 200 (OK), 404 (Not Found), 400 (Bad Request) e 500 (Internal Server Error).

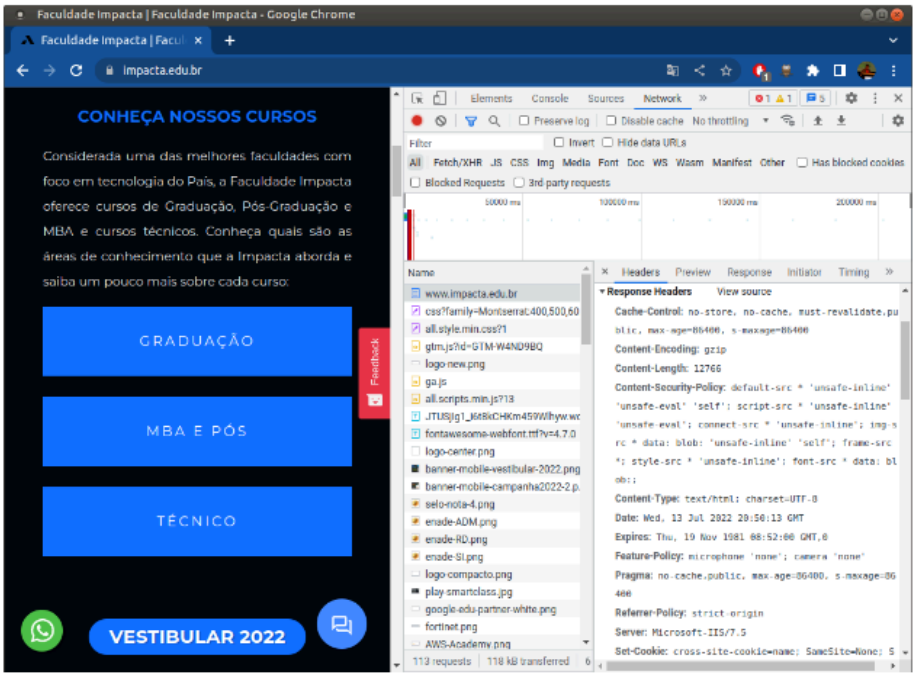
As faixas dos códigos são dividas em (HTTP response status codes, s.d.):

- 100 até 199: códigos informativos. São usados na comunicação do cliente com o servidor com informações intermediárias sobre a comunicação de rede. Requisições com números nessa faixa não são muito comuns no dia a dia;
- 200 até 299: códigos de sucesso. Indicam que a requisição foi bem sucedida no servidor. O sucesso de cada requisição depende muito do método sendo usado;
- 300 até 399: códigos de redirecionamento. Indicam que o navegador precisou tomar uma ação a mais para completar a requisição, como por exemplo, o redirecionamento de páginas, uso do cache, entre outros.
- 400 até 499: códigos de erro do cliente. Mostram que alguma coisa deu errado no cliente (navegador). Situações comuns são: algum problema na comunicação estabelecida pelo navegador, ou falta alguma informação ou condição para a requisição ser concluída (ex: não está logado).
- 500 até 599: códigos de erro do servidor. Indicam que o erro aconteceu no lado do servidor, podendo ser problemas na rede interna ou uma inconsistência na aplicação rodando no servidor.

Cabeçalhos de resposta

Da mesma forma que na requisição, a resposta também possui seus próprios cabeçalhos, no mesmo formato.

Muitos destes cabeçalhos são criados pelo servidor, baseado na resposta. Por exemplo, o cabeçalho Date mostra a data em que a resposta foi gerada e o Server indica o software rodando no servidor. Outros cabeçalhos podem ser criados pela aplicação de acordo com a necessidade dela. Já os cookies são definidos no navegador através do cabeçalho de resposta Set-Cookie. A Figura 1.4 mostra algumas informações disponíveis no cabeçalho de resposta.



Corpo da resposta

O corpo da resposta contempla o recurso que foi requisitado. Esse recurso pode ser qualquer coisa, por exemplo: um documento HTML, uma imagem, um vídeo, um arquivo xml, um arquivo PDF, etc.

HTML

HTML é uma sigla para HyperText Markup Language, que significa linguagem de marcação de hipertexto. Hipertextos são documentos que permitem que o leitor possa ler em uma ordem que ele bem entenda, usando ligações (links) entre os diferentes textos que compõem o hipertexto (HIPERTEXTO, s.d.). Na web, ao adentrar em um site, é possível navegar em diferentes páginas deste site da maneira que quisermos, clicando nos links internos dele. A web é a maior coleção de hipertextos que existe.

O HTML é uma linguagem de marcação, ou seja, é uma linguagem artificial que define um conjunto de sinais e códigos aplicados a um texto ou a dados para definir a sua configuração. A marcação não aparece no trabalho final. Assim, o HTML não é uma linguagem de programação, pela ausência de várias características: variáveis, de estruturas de decisão/repetição, funções, etc. A função do HTML é dar estrutura, significado e semântica para o conteúdo que desejamos mostrar nos sites da web. Podemos usar os marcadores (tags) para indicar um texto como sendo um parágrafo, títulos, citações, tabelas, etc, onde os navegadores usam esses marcadores para mostrar uma página estruturada na tela dos computadores. O navegador é, portanto, o interpretador da linguagem HTML, pois ele interpreta as tags e renderiza (exibe) o conteúdo formatado na tela.

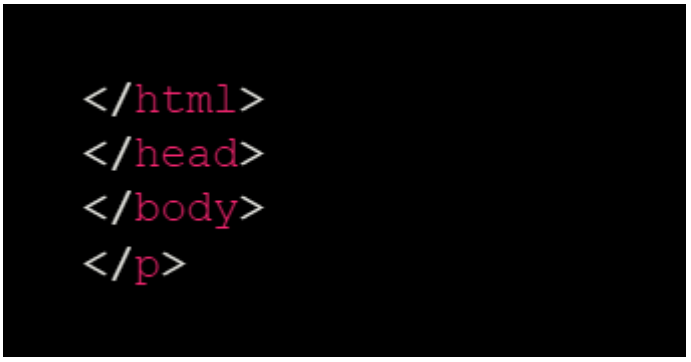
A primeira versão da linguagem HTML foi escrita em 1993 por Tim Berners-Lee, como forma de resolver o problema de compartilhar suas pesquisas com o seu grupo. Desde então o HTML evoluiu bastante, sendo a versão mais atual, o HTML5, uma revolução em termos de semântica e estrutura na web.

Sintaxe básica

O HTML é composto de marcadores, ou mais comumente chamados de tags, para dar significado semântico a um texto. Esse conteúdo tradicionalmente fica em arquivos com extensão .html, que podem ser manipulados em qualquer editor simples de texto.

As marcações seguem uma estrutura específica e sua própria nomenclatura. Todo marcador (tag) começa com o símbolo < (“menor que”), seguido do identificador da tag, e termina com > (“maior que”).

Após o marcador de abertura, podemos incluir algum conteúdo textual. Por fim, devemos fechar o marcador de forma semelhante à marcação de abertura, mas adicionando uma barra (/) entre o símbolo < (menor que) e o identificador da tag.



Outro detalhe importante é que o HTML não é sensível a maiúsculas ou minúsculas, ou seja, escrever uma determinada tag como <marcador>, <Marcador> ou <MARCADOR> não fará diferença no funcionamento, mas é uma boa prática escrever o identificador da tag com letras minúsculas. Ao contrário do que ocorre na linguagem Python, o navegador não diferencia espaços a mais que existam nos documentos HTML, sejam quebras de linha, tabulações (tabs) ou apenas espaços que não sejam os separadores de palavras.

Algumas tags não possuem o marcador de fechamento, e em geral são utilizadas para inserir algum conteúdo especial ou efeito na página: são tags ditas autocontidas. Um exemplo é a tag de quebra de linha, que é representada por
 ou
. Note que a barra de fechamento no fim da tag (segunda opção) é opcional no HTML5, e ambas as formas causarão o mesmo efeito.

A junção da tag e seu conteúdo (se houver) é chamado de elemento HTML.

```
<p>Salve este exemplo em um arquivo com extensão .html e abra-o no navegador. Note que o conteúdo do parágrafo fica entre as tags de abertura e fechamento.</p>
```

Atributos HTML

Dentro das tags HTML podemos colocar informações adicionais que auxiliam o navegador a entender e estruturar melhor o documento. Essas informações são chamadas de atributos HTML. Os atributos são adicionados sempre nas tags de abertura, no formato nome=”valor”, sendo possível haver mais de um atributo na mesma tag, bastando separá-los por um espaço em branco.

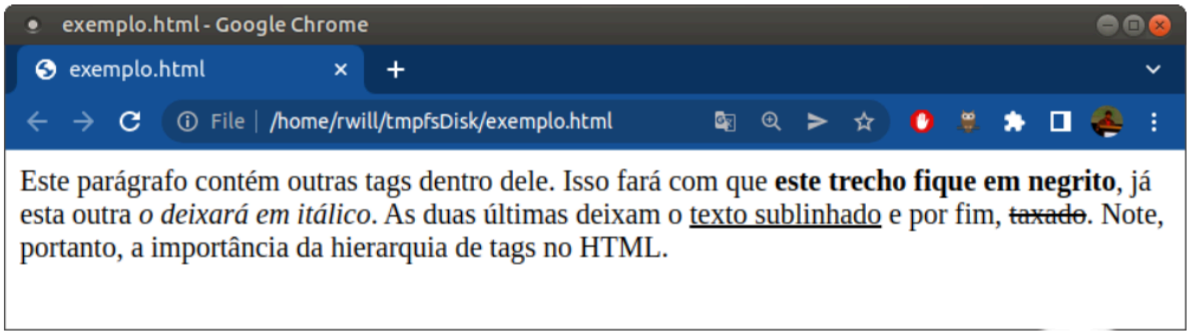
```
<p id="paragrafo" class="principal">Salve este exemplo em um arquivo com extensão .html e abra-o no navegador. Note que o conteúdo do parágrafo fica entre as tags de abertura e fechamento.</p>
```

Existem diversos atributos para as tags com os mais diferentes objetivos, alguns são de escopo global, ou seja, todas as tags podem usar (como os atributos id e class, por exemplo) e outros são específicos para tags específicas (exemplos: o atributo href para as tags <a> e <link>, ou o atributo src para as tags <img> e <script>). Uma relação mais completa pode ser vista em HTML Attributes (s.d.).

Tags aninhadas

O HTML permite escrever tags dentro de outras tags, o que permite criar uma estrutura hierárquica de marcadores, que definirá a organização do documento (uma página na web). É possível, por exemplo, ter várias tags que marcam textos simples dentro de um parágrafo

```
<p>Este parágrafo contém outras tags dentro dele. Isso fará com que <strong>este trecho fique em negrito</strong>, já esta outra <em>o deixará em itálico</em>. As duas últimas deixam o <ins>texto sublinhado</ins> e por fim, <del>taxado</del>. Note, portanto, a importância da hierarquia de tags no HTML.</p>
```



Estrutura básica do documento HTML

Para definir um documento em HTML, usamos a estrutura básica definida na Figura 2.6. Nela, podemos notar o uso das seguintes tags:

Tag	Descrição
<!DOCTYPE html>	Define que o tipo do documento é HTML
<html> ... </html>	Define o início (e o fim) do documento HTML
<head> ... </head>	Contém metadados, informações e configurações do documento
<body> ... </body>	Contém o corpo do documento. É aqui onde vamos adicionar elementos visuais (títulos de seção, parágrafos, tabelas, imagens, campos de formulários, etc)
<!-- ... -->	Define um comentário em HTML. Pode usar várias linhas e seu conteúdo não é renderizado pelo navegador

```html
<!DOCTYPE html>
<html>
<head>
<!-- Esta parte contém metadados e informações/configurações sobre o documento -->
</head>
<body>
<!-- Esta parte define o corpo do documento: títulos de seção, parágrafos, tabelas, imagens
etc. -->
</body>
</html>
```

Note que cada uma das tags possui uma função muito específica, e devem aparecer na ordem definida no exemplo. A seguir, explicaremos os principais marcadores (tags) da linguagem, dividindo-as em categorias para facilitar o entendimento.

Tipos de tags do HTML

O HTML5 define mais de 100 tags diferentes para diversos objetivos. Obviamente, estudar todas elas seria um trabalho muito grande, e por esse motivo vamos enfatizar as tags importantes e mais usadas.

Por mais que existam muitas tags, é possível dividi-las em algumas categorias para ajudar a encontrar uma que mais se adeque ao conteúdo que desejamos mostrar. As categorias de tags que dividimos neste curso são: configuração, marcação de texto, estruturas de texto, estruturas de documento e multimídia.

Tags de configuração

São tags escritas diretamente na head do documento, usadas para configurar como o navegador vai interpretar o documento como um todo. As mais comuns desta categoria são: title, meta, link, style e script. As tags style e script são específicas para o uso do CSS e do JavaScript respectivamente, e as detalharemos mais adiante no nosso curso.

A tag title define o título do documento. Esse título sempre será um texto, e irá aparecer na barra de título do navegador (na aba dele).

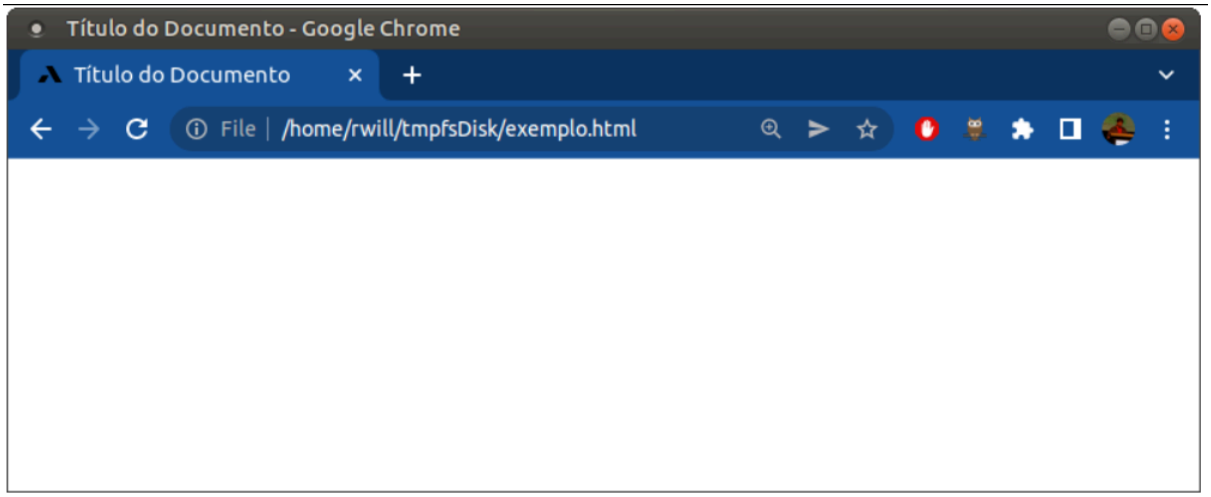
A tag meta é usada para configurar metadados (informações sobre o documento) que são invisíveis ao usuário, mas importantes ao navegador e buscadores: é uma tag autocontida que aceita vários atributos diferentes. O

atributo charset é comumente utilizado para indicar qual codificação de caracteres foi utilizada no arquivo HTML. O recomendado é que seja utilizada a codificação UTF-8.

A tag link é usada para carregar recursos externos usados na visualização pelo navegador, como arquivos CSS ou um ícone para a aba do navegador (favicon).

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Título do Documento</title>
   <link rel="icon" href="favicon.ico">
</head>
<body>

</body>
</html>
```



Note que para simular esse exemplo em sua plenitude, você deve ter um arquivo de ícone (uma imagem com extensão ico, png, jpg, etc) com o mesmo nome indicado no atributo href da tag link. O ícone sempre deve ser uma imagem quadrada, e de preferência pequena (16x16, 32x32, 64x64 pixels, etc).

Tags de marcação de texto

São tags utilizadas para marcar um texto específico, sem que haja uma quebra no fluxo do texto em si, como se fosse usado um marca texto físico. Elas são usadas para dar algum tipo de ênfase no texto que desejamos "marcar". As mais importantes são as tags: a, strong, em, ins e del.

A tag a (âncora) é uma das mais utilizadas na Internet, pois é ela que possibilita a navegação entre os diferentes hipertextos, criando os hiperlinks entre eles. O texto interno à tag é o que fica marcado como conteúdo clicável pelo navegador (ganhando a clássica cor azul e sublinhado como estilo padrão).

Para a âncora funcionar, ela precisa do atributo href definido. Este atributo terá a URL do recurso que a âncora está referenciando, onde será navegado quando ela for clicada. Essa URL pode ser tanto absoluta (com protocolo, domínio, etc.) quanto relativa (com apenas o caminho que falta dentro do mesmo domínio). Além

disso, ela pode referenciar um conteúdo interno da página, usando o fragmento da URL com o ID do conteúdo que ela referencia. Veja mais exemplos e detalhes em HTML a tag, (s.d.).

Já as tags strong, em, ins e del são usadas para realçar o texto com objetivos diferentes. Cada uma traz um significado semântico e um estilo diferente. A tag strong para textos importantes (negrito), em para textos enfáticos (itálico), ins para textos inseridos (sublinhado) e del para textos removidos (tachado).

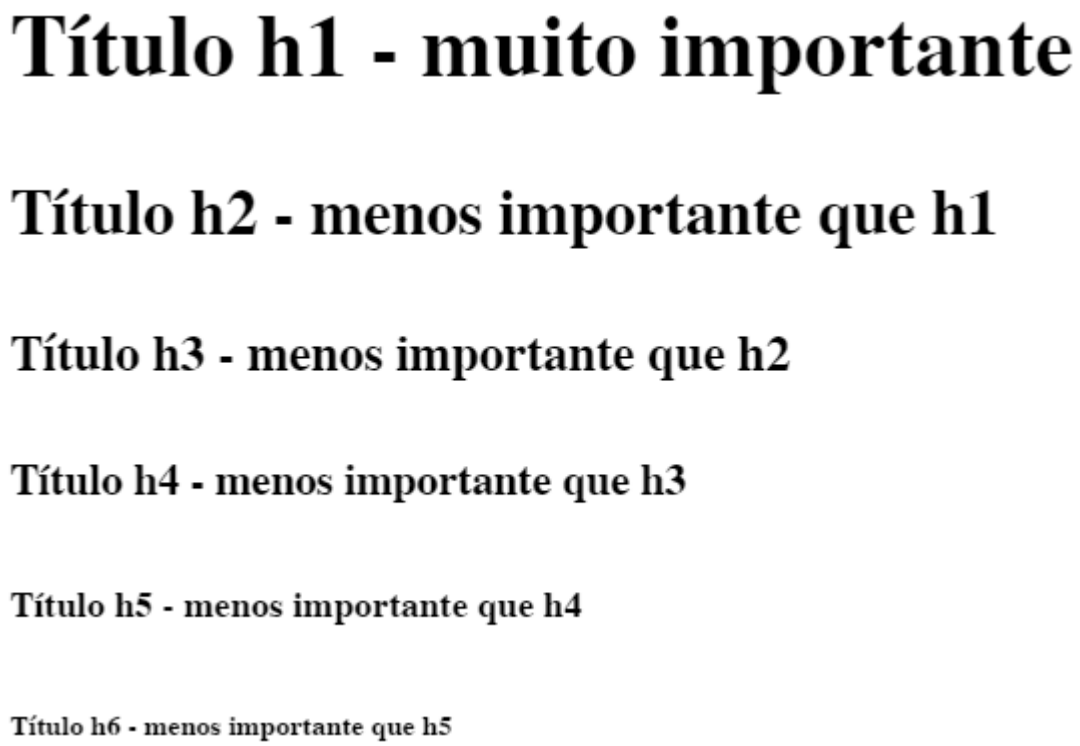
Tags de estrutura de texto

Essas tags vão definir os elementos textuais presentes nas páginas da web, construindo todo o conteúdo com elementos comuns em textos que vemos em outros lugares, como livros e jornais. As tags mais importantes desta categoria são: p, br, h1 até h6, ul, ol e table.

A tag p (já mostrada) representa uma unidade de um parágrafo textual, com os espaçamentos (margens) previstas.

Como qualquer quebra de linha feita por teclas enter são ignoradas pelo navegador, caso seja necessário quebrar a linha no HTML é necessário usar a tag br, que é uma tag autocontida (ou seja, basta escrever
), que causa o efeito de quebra de linha imediata.

Os títulos de sessão dentro de um documento são chamados de headings no HTML, e são marcados por tags que vão do nível 1 (mais prioritário) ao 6 (menos prioritário), ou seja: h1, h2, h3, h4, h5 e h6. Os números nos headings são usados como indicação do nível de sessão, ou seja, o h2 deve vir após um h1, o h3 após o h2, e assim por diante (HTML Headings, s.d.). A Figura 2.8 mostra um exemplo de como ficam os títulos de sessão.



As listas de itens e tabelas são elementos mais complexos, que envolvem mais de um tipo de tag. As listas básicas são divididas em dois tipos: ordenada (definida pela tag ol, de “ordered list”) e não ordenada (definida

pela tag ul, de “unordered list”). A lista ordenada possui como identificador uma sequência, que pode ser numérica ou alfabética. Já a lista não ordenada tem como identificadores símbolos iguais, em geral pontos (bullets). Ambas as listas abrem com suas respectivas tags, mas são seguidas por vários itens marcados com a tag li (“list item”), com exemplos em HTML Lists (s.d.).

Já a tabela possui diversos elementos, alguns opcionais, para definir sua estrutura. Comumente começamos por sua tag table. Dentro dela colocamos a sequência de linhas que construirão a tabela, usando a tag tr (“table row”). Dentro de cada linha (tr) colocamos as células que farão parte das linhas, de maneira similar aos programas de planilha. As células podem ser representadas pelas tags th (“table header”, para as linhas de cabeçalho) e td (“table data”, demais linhas), veja exemplos em HTML Tables (s.d.).

Tags de estrutura de documento

Conforme o documento HTML vai crescendo, surge a necessidade de organizar o código e o conteúdo de acordo com os seus significados e sua semântica. Essa categoria inclui uma grande lista de tags, cada uma com sua função semântica na separação de conteúdos. Algumas das tags desta categoria são: div, section, article, main, header, footer e nav.

Todos os marcadores desta categoria são usados para agrupar outras tags. Por exemplo, se estamos montando uma página de notícias, e cada notícia for composta por um heading (título da manchete) e alguns parágrafos, faz sentido que cada uma das notícias esteja dentro de article, e um conjunto de notícias (ex: todas de esporte) estejam dentro de uma section. Cabeçalhos e rodapés, seja de páginas ou outros elementos, devem estar dentro das tags header e footer e quaisquer lista de âncoras (links) devem estar dentro de um nav, para indicar elementos de navegação. Veremos mais adiante sobre o conteúdo semântico das tags.

Tags de multimédia

As páginas da web não são feitas apenas de textos simples, temos muitos conteúdos de multimédia, como imagens, vídeos e sons e o HTML tem tags específicas para esses casos. As mais importantes nessa categoria são: img, audio e video.

A tag img é amplamente utilizada em praticamente todos os documentos HTML. Trata-se de uma tag autocontida que precisa de um atributo chamado src para indicar onde o navegador deve procurar a imagem (que pode ser indicada utilizando uma URL absoluta ou relativa). É recomendado que o atributo alt seja definido também, como um texto alternativo para exibir uma mensagem quando a imagem não carregar, e também para facilitar a navegação de usuários com dificuldades de visão, além de auxiliar buscadores da Internet.

As tags audio e video possuem uma estrutura similar. Ambas precisam definir a fonte dos conteúdos a serem tocados, através das tags secundárias source. Para ambas é possível definir mais de uma tag source, além de ser possível definir outros atributos de controles ou autoplay nas tags, como pode ser visto em Video and audio content (s.d.).

Você conhece?

O professor e divulgador pelo Youtube Gustavo Guanabara publica seu material, de forma totalmente gratuita, desde 2013, no seu site pessoal e no seu canal do Youtube. Possui dois cursos bem completos de desenvolvimento Web com HTML e CSS, além de outros temas relacionados a programação e computação geral. Confira mais detalhes no site dele https://www.cursoemvideo.com/.

Estrutura básica do documento HTML

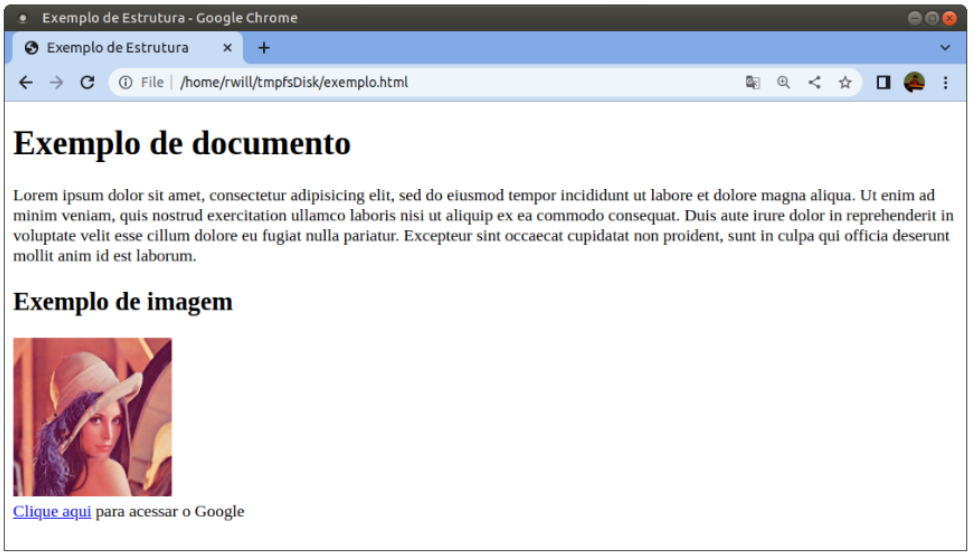
Assim como outros tipos de documentos digitais, o HTML possui uma estrutura própria que devemos seguir. Apesar dessa estrutura não ser obrigatória, uma vez que o navegador consegue entender tags “soltas”, é extremamente recomendável escrevê-la.

Como vimos, algumas tags são usadas para marcar conteúdo que o usuário irá ver: textos, imagens, tabelas, listas, títulos de seção (headings) etc. Essas tags compõem o que chamamos de “corpo” (body) do documento. Outras tags são usadas pelo navegador como configuração, e compõem a “cabeça” (head) do documento, como por exemplo: as tags title (titulo na aba), link (para importar algum recurso visual para o navegador) ou meta (para configurar algo no navegador, como a codificação a ser usada).

Além do head e do body, todo documento HTML possui mais dois elementos: a tag raiz html e a diretiva DOCTYPE. O HTML é uma linguagem de marcação do tipo SGML (Standard Generalized Markup Language), que é um padrão internacional para linguagens de marcação como o HTML e o XML (eXtensible Markup Language). Na SGML é obrigatório os documentos começarem com a diretiva de tipo de documento (DOCTYPE) e um primeiro elemento raiz, onde o resto do documento será escrito. No caso do HTML, o tipo sempre vem com a marcação <!DOCTYPE html> e o elemento raiz sempre é a tag <html>.

Portanto, todo documento HTML considerado correto fica com a estrutura representada na Figura 2.9.

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8">
        <title>Exemplo de Estrutura</title>
</head>
<body>
        <h1>Exemplo de documento</h1>
        <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut
labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut
aliquip ex ea commodo  consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum
dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum.</p>
        <h2>Exemplo de imagem</h2>
        <img src="lenna.jpg" alt="Imagem de exemplo">
        <div>
                <a href="http://www.google.com">Clique aqui</a> para acessar o Google
        </div>
</body>
</html>
```



Semântica HTML

O HTML surgiu como uma maneira de estruturar um documento na web e transferi-lo pela rede. Conforme o conteúdo presente evoluiu e a web pública ficou cada vez mais importante e maior, tornou-se necessário trabalhar mais na marcação deste conteúdo, de forma a facilitar o entendimento dele por mecanismos de busca e por usuários que usem tecnologias assistivas. O HTML na versão 5 trouxe uma revolução em termos de semântica para os documentos, ou seja, no sentido que os conteúdos querem trazer para os usuários que os estão consumindo.

Antes do HTML5 era comum dividir o conteúdo dos sites usando a tag div, que era usada apenas para dividir para organizar o código e facilitar a aplicação de estilos. Para facilitar o entendimento do conteúdo do documento HTML, outras tags de divisão com valor semântico foram adicionadas.

Agora pode-se utilizar a tag section para dividir o conteúdo de uma mesma página em seções (partes de um todo), ou usar a tag article para conteúdos mais independentes. As tags header e footer identificam com precisão onde está o cabeçalho e o rodapé das páginas, enquanto que a tag main identifica onde está o conteúdo principal. A tag nav mostra onde estão as opções de navegação, internas ou externas às páginas.

Além da divisão do documento, existem tags para dar conteúdo semântico aos textos específicos. As tags strong e em, por exemplo, além de deixarem o texto em negrito e itálico, respectivamente, também marcam o texto como algo importante para o documento (no caso do strong) e algo enfático (no caso do em), onde sua ênfase pode mudar o significado da frase que está inserido (ex: ironia).

Um site com o conteúdo semântico bem definido e dividido ajuda aos buscadores entenderem e referenciarem mais a página em específico, sendo que essa construção é uma das bases do Search Engine Optimization (SEO), além de auxiliar na acessibilidade do site por usuários guiados por tecnologias assistivas.

Você quer ler?

Vimos que a semântica é importante na construção de sites mais entendíveis para tecnologias assistivas e os motores de busca. Existem outras coisas que devemos considerar na hora de escrever código HTML para melhorar a acessibilidade, como textos e atributos de tags HTML. Veja mais detalhes em HTML: Boas práticas em acessibilidade (s.d.), disponível no link: https://developer.mozilla.org/pt-BR/docs/Learn/Accessibility/HTML.

aula 3
Formulários
parei no hora 1:04:00 da aula