**Módulos do curso:**

M01 -> Primeiros passos HTML e CSS -> Conceitos básicos, preparação do ambiente, semântica do HTML 5, textos, títulos, ligações, multidimia, estilos.

M02 -> Deixando as coisas mais bonitas -> Fundamentos de disign, psicologia das cores, tipografia, elementos CSS, modelo de caixas, wireframe, responsividade.

M03 -> colocando um protótipo no ar -> Versionamento de software, hospedagem de sites estáticos, tabelas.

M04 -> aprofundando conhecimentos -> quadros em linha, formulários, media queries mobile first.

M05 -> Novas Tecnologias -> Flexbox, Grid Layout, Projeto final.

**Como Funciona a internet:**

Representação de dados:

O computador como equipamento eletrônico funciona através de sinal, que baseados em ondas elétricas baseados em 0 e 1, são chamados de códigos binários, para que a gente tenha algum dado, precisamos de 8 bits ou declarado como 1 byte -> 1 caractere = letra A. o computador só entende essas ondas binarias ou ondas quadradas, que são linhas retinhas representadas por 0 e 1.

Código multibyte UTF-8:

Mesmo conceito ensinado na aula de medida de armazenamento: múltiplos de 210, essa representação de dados já está clara porque além de estudas já foram ensinadas por mim dentro de sala de aula, parando na medida de petabyte, já que não chegamos nem perto ainda de usar uma medida maior nos dias de hoje.

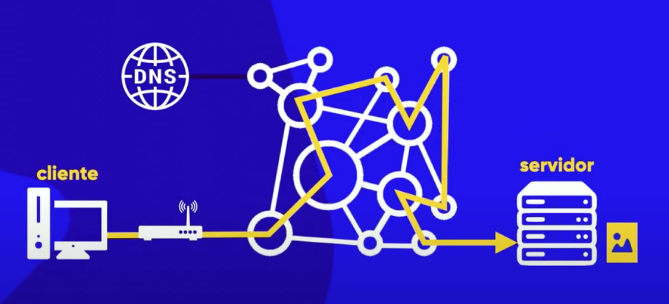
MB: Megabytes -> eles servem para medida de armazenamento

Mb: Megabits -> esse já mais utilizados para transmissão, como uma rede de internet.

Como nos conectamos a internet:

Começamos com o nosso dispositivo que pode ser declarado como cliente, acessamos o site que está alocado na internet, que podemos ter acesso por um modem ou roteador, todo esse conjunto me da o acesso a internet. Para que o acesso funcione bem, temos que entender que o aparelho que dá o acesso à internet, ele usa a modulação (que seria mudar a codificação do aparelho para a internet) e a demodulação (que muda a codificação da internet para o aparelho).

Servidor, essa e base de acesso para todos os arquivos que estão na internet, esse servidor guarda uma numeração do site que está ou que acessar, e também guarda a sua numeração, pois é, quem navega também tem uma numeração.

Servidor DNS: sistema de nome de domínio, são servidores especiais, quando você tenta acessar algo, primeiro ele passa pelo DNS que é quem administra essa transição, depois o DNS organiza e te mando para o caminho que você pediu. O roteamento pode seguir diversos caminhos diferentes, pode pegar um mais rápido se tiver tudo bem, se tiver congestionado ele pode ir por uma rota mais longa.

Está aí uma imagem que ilustra esse caminho 🡪

**Domínio de hospedagem**

Situação 1: Guardado no servidor

Quando um URL é digitado, ele manda a informação para o DNS que vai retornar para o cliente o numero do servidor e depois o seu dispositivo procura no servidor.

Situação 2: Guardado no meu PC

Aqui os arquivos criados ficaram no meu computador, se quero que amigos acessem site oque faço? Eles vão ficar sem ver.

Situação 3: guardado em outro PC

Meu amigo fez um site e quer me mostrar como faço? Não tem como ainda.

O HTML e CSS só funciona se estiver no servidor, a partir do momento que mando para um servidor, crio meu endereço pro site. Sempre que criar um site e quiser ele em funcionamento tem que ter em mente dois nomes:

**Domínio** -> Nome único – Pago anualmente – vários TLDs

**Hospedagem** -> Espaço para armazenar os arquivos – pago mensalmente – espaço, memória, recurso -> á hospedagem é um item delicado que tem de ser escolhido com bastante atenção.

Vamos entender uma **URL** 🡪

Ali ficaram dois exemplos com formatos diferentes. É importante lembra que o **http://** é um protocolo que não faz parte dessa leitura de **URL**.

**Diferença entre HTML, CSS e Java Script**

Como Funcionam a HTML e as CSS?

Não se programa nessas duas formas de criação de site, até porque nenhuma das duas são linguagens de programação, HTML e CSS são usados para desenvolver ou criar sites e nunca programar.

**HTML** 🡪 **H**YPER**T**EXT – **M**ARKUP – **L**ANGUAGE = linguagem de marcação para hipertexto.

**CSS** 🡪 **C**ASCADING – **S**TYLE – **S**HEETS = Folhas de estilo em cascada.

A linguagem **HTML** é a parte do conteúdo de um site, todo o conteúdo de um site é feito por essa linguagem, seja ele um texto, imagem, vídeos e tabelas entre outra, existem muitas possibilidades de conteúdo. A **CSS** tem foco total em design, cores, sombras, tamanhos e posicionamento. O **JavaScritp** que é uma linguagem de programação, pode trabalhar junto com essa outras ferramenta para interatividade do site, para dar mais vida, menus interativos, animações, popups e validações.

O conjunto das 3 linguagens que dão vida para um site bem feito e essenciais para um programador que quer ter domínio nessa área.

**Conteúdo em HTML**

O **<h1>** no inicio é abertura de tag

**<h1>exemplo de título<h1>**

O “exemplo de título é o conteúdo, e o **<h1>** no final fecha a tag.

**<p>Exemplo de parágrafo<p>**

**<img src= “foto.png” alt=”Exemplo de foto”>**

A grande maioria das tags tem abertura e fechamento, algumas não tem, como mostra o exemplo da imagem ali em cima. A grande mais dessa tags tem parâmetro e valor. No exemplo **alt=** é um parâmetro e **Exemplo de foto** é um valor.

**Estilo CSS**

Anatomia do css 🡪

Estrutura básica de documento HTML

<!DOCTYPE html>

<html lang=”pt-br”>

<head>

<meta charset= “UTF-8”>

<meta name=”viewport”

Content=”width=device-width,

Initial-scale=1.0”>

<title>Document</title>

<head>

<body>

<h1>Olá, Mundo!</h1>

<body>

</html>

Como funciona a HTML e as CSS?

Depois de colocar a HTML e CSS, quem for fazer acesso, vai receber o dado visual de todas as linhas que foram digitadas no navegador.

Entendendo essa estrutura de HTML que foi digitada:

<!DOCTUPE html>

Serve para o navegador entender que é a linguagem de html mais recente, versão mais moderna.

<html lang=”pt-br>...</html>

Esse é o esqueleto principa da página.

<head>...</head>

Ali não se guarda conteúdo, e sim partes que que são feitas fora do html, links de arquivos de estilo do CSS, scripts (javaScritp) e outros.

<meta charset=”UTF-8”>

Define como as letras e símbolos serão entendidos, o UTF-8 permite acentos, cedilha e caracteres de várias línguas.

<meta name="viewport" content="width=device-width, initial-scale=1.0">

Para design responsivo, mostra o site proporcionalmente.

**Front-end, Back-end e Full Stack**

As tecnologias de HTML, CSS e javaScritp são cliente-side, o servidor ele não usa essa tecnologia, ele apenas de entrega o arquivo, e o seu navegador utiliza essa tecnologia.

Esse é o termo para o front-end, tecnologias que funcionam a partir do cliente, com as tecnologias citadas a cima.

A tecnologias do lado do servidor que são chamadas de server-side, que são chamados de beck-end quem é especializado nessa área. As linguagens Server-side são php, javaScript(a mesma linguagem do front-end, mas com tecnologias diferentes), C#, python, Ruby, java.

Quando essas duas áreas são bem estruturadas no estudo, ou seja quando você domina essas duas áreas temos o desenvolvedor full Stack. É uma área respeitada, mas exige muito estudo, muita pratica entre outros. Guanabara manda o recado de coração que é impossível chegar nesse nível de Full Stack.

**Começando HTML**

Acredito que acostumar com um editor de código, que no caso será usado o visual code, é essencial e fundamenta mental, é como se acostumar com um novo ambiente de trabalho, é preciso olhar e olhar, e manter aquele local/software em mente, até ter clareza do que ele faz.

Na primeira aula de introdução ao Visual code, vi que é uma área de fácil compreensão, criei algumas linhas, parágrafos e separações a mais, só pra não ficar naquela de copia e cola. A compressão total a área de trabalho do editor veio depois de ver vídeo, ler o PDF que foi criado como forma de estudo, e ler livro de introdução ao HTML.

Indo para parte prática 🡪 **<h1>** tag que inicia título como principal**. </h1>** tag que finaliza esse título, essas tags vão de **h1** até **h6**.

**<p>** e **</p>** tags que abrem e fecham parágrafos, dentro dessas tags com utilizar **<br>** para saltar uma linha dentro do site, **&lt;** e essa tag(na verdade não sei se é tag, mas serve para mostra uma kkkk.

**Exemplo** 🡪 tenho que criar um site onde mostrar exatamente a tag **<p>,** e eu não posso simplesmente digitar ela, então o **&lt;** vai ser usado no lugar dos **<>**.

**Vou fazer uma separa para ficar mais fácil:**

* **<h1> -> serve como tag de título**
* **<p> -> serve como tag de parágrafo**
* **<br> -> Serve dentro do parágrafo para pular linha, e não precisa ser fechada**
* **&lt; e &gt; 🡪 serve para mostrar tags dentro do parágrafo no site**

Símbolos e emojis no site

Esses emojis e símbolos são adicionados a partir de código que consigo adquirir em sites da internet, vi que pelo visual code da pra colocar alguns padrões por ele mesmo, mas usando o atalho “WIN +.” é possível adicionar os emojis de forma bem mais pratica.

**Imagens no site**

**Direitos autorais de imagem** 🡪 não é correto pegar imagens da internet e simplesmente colocar no meu site, é algo totalmente ilegal, vamos para fontes onde posso usar, onde tenho o direito de usar no meu site.

Aqui dei uma busca por fora e vi que é mais fácil utilizar sites que tem imagens que te dão a liberdade de usar em seu site, esses sites são:

PEXELS -> https://www.pexels.com/pt-br/

PIXABAY -> https://pixabay.com/pt/

UNSPLASH -> <https://unsplash.com/>

Exemplo de como usar se precisar de uma por site:

📸 Foto por [Nome do autor] via Unsplash  
ou  
Imagem: [Autor] – Pexels

💡 Exemplo real:

Foto por Andrea Piacquadio via Pexels

De forma mais fácil, posso simplesmente pedir para que o ChatGPT crie uma imagem, ou usar minha própria imagem, tudo isso se encaixa em uma realidade de direitos autorais para o site.

**Falando sobre formatos de imagem**:

O Gimp é o programa que vai ser usado para trabalhar com essas imagens que vão ser lançadas nos sites.

Ele é um programa mais limitado, mais é gratuito oque ajuda muito pra esse tipo de aprendizado, já tinha ele instalado então não é meu primeiro contato com o software.

O melhor formado para uma imagem de fundo grande é o JPG.

Uma imagem flutuando por cima com fundo transparente é o PNG.

Geralmente as imagens maiores vem no formato JPG que são imagens JPEG, esse formato de imagem não tem suporte a transparência.

Imagens com fundo transparente, como logos ou personagens sem fundos, esses vêm no formato PNG.

Já entendia sobre os formatos de imagens, mas não tinha essa clareza de tamanho e especificação de cada formato de imagem.

De forma individual posso lembrar dos sites que compactam as imagens, pode ajudar na otimização do site ter um site que carrega mais rápido, mais fluido.

**Tamanho de imagens**

Aqui pegamos uma imagem no formato de JPG, baixamos a imagem, e modificamos para deixar com o tamanho ideal de uma imagem de site, assim o site não fica pesado.

Passos feitos:

Abri a imagem com GIMP 3, programa de edição gratuito, fui na opção -> imagem -> Redimensionar imagem -> em tamanho: L1500/A1000 e resolução x e y em 50,000.

Depois das alterações feitas, o processo não foi aplicar a alteração imagem, e sim salvar uma nova, assim mantenho a original para se precisar de outro formato ou alteração, em qualidade deixamos em 70, e o resultado foi apenas uma imagem menor, em questão de tamanho e de espaço no armazenamento, não notei nenhuma queda de qualidade da imagem.

O guanabara desceu até 650px, e cara, vai depender de como vai ficar no site, é tudo questão de teste.

Mesmo processo foi feito em uma imagem PNG.

**Adicionando imagens no site**

Comecei baixando algumas imagens que posso utilizar no formato png, foram 2 imagens da logo do html e do css.

Adicionar imagens por aqui, é mais fácil do que eu estava imaginando, dentro o GIMP tem a possibilidade de retirar os pixels sobrando coma guilhotina, é como recortar uma parte que você não quer da imagem. É ideal usa a guilhotina para ter imagens alinhadas dentro do site, fiz no meu antes de usar a ferramenta, e é nítido a diferença.

Falando sobre como adicionar imagens, é um processo simples: digita img -> enter -> entre a primeira aspa use “Ctrl+espaço” – selecione a subpasta e depois o arquivo, em alt -> especifique sobre o que se trata aquela imagem.

É possível trazer imagem através de um link, deixar isso só como curiosidade, não a intenção de fazer isso, mas bastar colar o url dentro do lugar da imagem.

**Ícones de favoritos**

O ícone de favorito é oque aparece no canto superior do site, basicamente mostra a logo do site.

Temos a possibilidade de baixar uma icone pronto -> <https://www.iconarchive.com/>

Temos a possibilidade de criar nosso icone -> <https://www.favicon.cc/>

Temos a possibilidade de converter uma img em icon -> <https://favicon.io/>

Para adicionar o icone no html: parte do head -> vá na linha antes do titulo -> link: favicon.

**Hierarquia dos títulos**

Cada título tem um nível:

Nível 1 -> Título principal -> h1

Nível 2 -> subtítulo -> h2

Dentro de cada nível eu aplico uma tag sobreposta, se for dentro do h1 a tag vai ser h2, se for dentro do h2 vai ser h3, e assim por diante. Não existe a limitação de usar o h2 só uma vez ou o h3 só uma vez, posso criar sim com a mesma tag para ter o mesmo alinhamento.

O html limita os níveis até o nível 6, ou seja, só pode ser usado até o h6.

**Semântica na HTML 5**

O significado dos vocábulos, por oposição a sua forma.

É sempre bom ficar de olhos nas tags obsoletas, tags que pararam de funcionar ou estão no sinal amarelo, uma hora pode parar de funcionar.

HTML 5 é focado em semântica, em significado e não em forma, ter em mente e sempre pesquisar, procurar informação correta antes de usar uma tag ou comando dentro do HTML.

Aula totalmente focada em explicar para sempre ficar por dentro do que está em funcionamento, ficar sempre atualizado é importante, ainda falando em uso de uma tecnologia como linguagem de marcação e outras.

**Negrito e itálico**

Existem duas formas de deixar o texto em negrito:

**<b>** que dá a forma de **negrito** para o texto

**<Strong>** esse além de dar a forma, da um sentindo para o texto “aqui eu quero destaque”.

CTRL+SHIFT+P -> abre uma aba onde posso pegar a parte de Wrap with abbreviation, e com ela posso escolher uma tag para envelopar o que estiver selecionado.

Para o itálico:

**<i>** que dá a forma para o itálico

**<em>** que já dá sentindo, semântica.

As tags (**i / b**) não são obsoletas, mas o guanabara recomenda que use o que dê sentido, já que o HTML é isso.

**Tags que podem ser usadas diariamente**

Aqui de Início aprendemos a tag **<mark>**, serve para adicionar um marcador ao texto, dentro da tag, posso usar style que já é um comando do css e não html para configurar a cor. Em casos de usar a tag diversas vezes e querer o mesmo formato, posso criar um seletor dentro da cabeça, ou seja, dentro de head, usando style e definindo a tag **mark** e a cor que vou usar nela em todo o projeto html.

Vimos as tags **<big>** e **<small>**, para deixar o texto grande e pequeno, a tag big está obsoleta dentro do html.

Outra tag interessante é a de **<sub>** e **<sup>**, para deixar o texto sobrescrito ou subscrito.

**Citações e códigos**

A tag **<code>** é muito utilizável para o texto ter uma separação de caracteres com a mesma distância, ótima para ler códigos, para códigos fica ainda melhor envelopar a tag code com a tag **<pre>**, fica divino a formatação.

O envelopar ali em cima caso eu esqueça é basicamente colocar uma tag dentro da outra.

Algo legal que foi ensinado também são macetes com a tabulação, selecionar um texto (sem as tags) e dar tab, ele acrescenta uma tabulação em tudo que foi digitado, segurando o shift, ele volta uma tabulação, algo extremamente útil.

Quando quiser adicionar aspas quem algo, se for algo para dar sentido, temos a tag **<q>**, claro que da simplesmente para digitar com as aspas, mas aí vem a questão da semântica.

**<Blockquote>**, outra tag aprendida que serve para citações, de forma visual, apenas acrescenta uma tabulação no texto.

**<ABBR>**, tag top que serve para apresentar a explicação de uma sigla que foi digitada.

**<bdo>**, uma merda. “adrem amu”

**Listas Ordenadas**

A melhor forma de criar listas dentro do html é com tags.

**<ol>** essa é tag que cria o formato de lista ordenada

Dentro dessa tag usamos **<li>**, que são itens de lista, assim ele já faz a numeração de lista. Algo interessante de citar é que essa tag de itens, ela não precisa ser fechada, bom pelo menos não é algo obrigatório.

Dentro da tag **<ol>** é possível usar o **type**, ele define o tipo de formato da lista, por exemplo **<ol type= “1”>** ele vai cria a lista numérica, “A” ele cria lista alfabética, e assim entre outros.

**Listas não ordenadas**

Cria uma lista, mas sem a Ordem alfabética, e sim com símbolos, aqui é só substituir a tag **<ol>** por **<ul>**, elas seguem o mesmo princípio do **<li>**.

Aqui como variações no **type** temos -> square, circle e o modo padrão.

**Para unir listas basta pensar e usar logica, abre um novo formato de lista e fecha, claro que de forma alinhada e organizada.**

Lista de definições

A lista de definição é criada dentro da tag <dl>, depois vem a tag <dt> que vai ficar a palavra que vai receber uma definição, e <dd> que vai ser a definição da palavra/sigla.

**Link e Âncoras**

Procurar sobre SEO mais para frente.

**Links e Âncoras**

Mais pra frente procurar sobre SEO. E estudar o algoritmo do google.

Para usar uma palavra ou parte do texto como forma de redirecionar a pessoa para outro ponto, ou seja, como link basta envelopar a parte do texto/site com a tag <a>, ele vai adicionar de forma automática a formação da tag, entre aspa fica o link que a pessoa será redirecionada. É importante que dentro dessa tag tenha a target=\_blank”, para que o usuário abra outra aba com seu link e não sobreponha o que ela estava usando.

Exemplo 🡪 **<li><a href="https://github.com/Luide-da-Paz" target="\_blank" rel="external">Meu repositorio GitHub</a></li>**

**Links Internos**

É possível criar outras páginas dentro da mesma pasta, e usar o mesmo comando anterior, sem a target claro, para passar de uma parte da página para outra parte. A seleção de pagina fica da mesma forma que seleção de imagem dentro do visual code, e dá medo a forma que esse programa de ajuda na criação, ele já está quase fazendo por conta própria.

O sistema de adicionar o rel depois do link, mostra que você quer mostrar o mapeamento de página, assim o google não faz isso por conta própria, e sua página pode ficar mais visível com simples alterações. Ele não tem nenhuma alteração visual, mas tem a questão da semântica, dar sentido a algo.

"../index.html"

Esse foi usado para usar o recurso de pasta anterior, se precisar voltar uma pasta para pegar um arquivo, preste atenção nos pontos, dois pontos vão voltar uma pasta.

**Links para Downloads**

Para identificação de cada tipo de arquivo existe o site media type, é necessário saber para identificar de forma correta o formato do arquivo.

Download de arquivo pdf

        <li><a href="../livro/SESSÃO DE TREINO PERSONALIZADO.pdf" download="meu livro.pdf" type="application/pdf">Livro de TREINO</a></li>

Download de arquivo ZIP

        <li><a href="../livro/SESSÃO DE TREINO PERSONALIZADO.zip" download="meu livro.zip" type="application/zip">Livro de TREINO (ZIP)</a></li>

O que escrever dentro de type?

<https://www.iana.org/assignments/media-types/media-types.xhtml>

Aqui temos o site que lista essa informação.

(“”)

**Imagens dinâmicas**

Trabalhando com múltiplas imagens com tamanhos diferentes:

Aqui só criamos imagens de tamanhos diferentes para dispositivos diferentes, não foi falado nada dentro do contexto de HTML.

Imagens que se adaptam sozinhas

De forma explicativa, tenho que ter diversos formatos de imagens dentro do site para que ele consiga se adaptar a todos os dispositivos que vão acessar.

Tamanhos comuns 🡪 1200 – 768 – 480.

Caso precise de resolução maior posso acrescentar a resolução fullHD ou 4K.

O dpi, pelos testes, o ideal é 72, na hora de salvar optar por algo entre 70 e 85 de resolução, depois é só exportar e deixar com nomenclatura de resolução para não se confundir ao aplicar as imagens.

**📁 Estrutura de Pastas Recomendada**

/projeto-site/

│

├── index.html

│

└── /imagens/

├── /banners/

│ ├── banner-480.webp

│ ├── banner-768.webp

│ └── banner-1200.webp

│

├── /produtos/

│ ├── produto1-480.webp

│ ├── produto1-768.webp

│ ├── produto1-1200.webp

│ ├── produto2-480.webp

│ ├── produto2-768.webp

│ └── produto2-1200.webp

│

├── /galeria/

│ ├── foto1-480.webp

│ ├── foto1-768.webp

│ ├── foto1-1200.webp

│ └── ...

│

└── /icones/

├── logo.webp

├── menu.webp

└── fechar.webp

📌 **Organização:**

* banners/ → imagens grandes da página inicial ou seções principais.
* produtos/ → imagens de cards, produtos, serviços.
* galeria/ → fotos complementares.
* icones/ → logos e ícones simples (não precisam de múltiplos tamanhos).

Essa é recomendação de Larguras e organização do ChatGPT, não dá pra ficar só nessas do guanabara.

**Até deu pra entender como funciona essa parte, mas me deixou bem insatisfeito, se não tiver mais assunto sobre imagem, vou ter que estudar por fora**.

**Colocando áudio no site**

Aqui foi recomendado sempre usar formatos MP3 ou ogg, o formato wav é compatível para ser utilizado, mas é um formato muito grande e pode pesar o site.

Imagens com qualidade top, e áudio com qualidade top são para download, para site não fica legal.

A tag áudio já faz quase tudo com clareza, usamos esse prompt final:

* <audio preload="auto" autoplay controls loop>
* <source src="Arquivo.mp3" type="audio/mp3">
* <p>infelizmente não conseguimos reproduzir o áudio. <a href="Arquivo.MP3">Clique aqui para baixar</a></p>
* </audio>

Não é áudio difícil de aplicar no HTML.

Outra informação que é listada é sobre

**Formatos de vídeos para site**

Pela pixels é possível baixar vídeos gratuitos também, caso precise.

Para converter vídeos foi recomendado usar o HandBreak, parece ser um conversor de vídeos e também um otimizador para os vídeos mais leve para o site, o que é algo útil e importante de se fazer.