Trilha Fundamentar

* HTML
* Módulo 1 – Ambiente DEV de outro mundo.

Esse módulo foi para ensinar a instalar o Microsft Edge, o VS Code e o Node JS para Windowns, Linux e MAC.

* Módulo 2 – Guia Estelar de HTML
* Instalando plugin de Preview HTML no VS Code – O Vs code nos permite instalar alguns plugins de dentro do próprio programa. No caso, para poder ver o resultado do que estamos codando, vamos instalar o HTML Preview.  
  Depois de instalado, começamos um novo arquivo no VS CODE. Indo na opção “explorer” , nova pasta, definindo onde eu vou guardar as coisas que eu vou escrever no VS CODE selecionar “New Untitled Text file” e sempre lembrar que a extensão desse arquivo tem que ser “.HTML”. Porém é preciso ter atenção a algumas regras na hora de nomear o arquivo. Ele não pode conter espaços entre as letras e nem usar acentos.  
  Ao começar a escrever, é necessário sempre salvar, utilizando o atalho CRTL +S. O Arquivo me avisa se está salvo através de um bolinha branca que fica no alto ao lado do nome da aba. Se a bolinha sumir, ele está salvo.
* O que é HTML – HTML é um acronico para Hyper Text Markup Language (Linguagem de marcação de Hipertexto) Ela por ser uma linguagem, indica que ela tem regras, ou seja, ela tem semântica, sintaxe e etc. A ideia do HTML é definir o conteúdo e a estrutura da sua página.   
  O Hipertexto é um texto não linear, com diversos conteúdos interconectados (Links)
* Comentários – É possível fazer comentários dentro do seu código, explicando alguma coisa, ou guardando algum dado como guia. No VS Code usamos a tag “<!-- -->” para abrir e fechar um comentário. E tudo que estiver dentro do comentário não ira para o código que estamos montando, eles não aparecem no resultado final.
* Anatomia da Tag – Uma tag tem sua estrutura definida por:
* Abertura da Tag (EX: <h1>)
* Conteúdo (O que vai ser escrito dentro da tag)
* Fechamento da Tag (EX: </h1>
* Elemento (Que é o conjunto das três estruturas anteriores)

OBS: Existem elementos que podem ser vazios. São elementos que possuem atributos dentro deles e não tem mais nada depois. Eles são abertos com a tag mas só são fechados com o sinal”>”  
Esses elementos não possuem conteúdo, nem tag de fechamento.

* Atributos HTML – Atributos HTML servem como informações extras para colocar no elemento ou configuração pura.   
  Um atributo também segue uma estrutura. Eles é sempre formado pelo nome (SCR) o símbolo igual (=) e um abre e fecha aspas (“”). E semrpe que for usar um outro na mesma linha, é necessário dar um espaço entre eles.   
  SCR= “ Aqui fica o conteúdo ”   
    
  Os atributos também podem ser Booleanos, que são atributos que não precisam de conteúdo para funcionarem. Booleano quer dizer verdadeiro ou falso.   
  Ex: Atributo DISABLE ( que desabilita uma linha)
* Atributos Globais –

Os atributos mais utiliazados:

-Class

-contenteditable

-data

-Hidden

- ID

-Style

-tabindex

-Title

* Aninhamento Hierarquia - É possível colocar uma tag dentro da outra, isso se chama aninhamento. Porém é preciso seguir algumas regras para que o seu HTML não tenha problemas.

-Hierarquia -> é preciso respeitar a hierarquia das tags. Respeitando a tag que vem primeiro. Então se eu abro uma tag primeiro, ela vai ser a última a ser fechada.

Ex: <p>

Vou <em> escrever um parágrafo aqui. </em>

</p>

Repare que por eu ter começado com a tag <p>, a última tag a ser usada para fechar o meu código vai ser a mesma tag </p>.

-Fluxo -> Isso corresponde a ordem em que os código foram escritos. O HTML vai ler os códigos na ordem em que eles foram escritos. Um abaixo do outro.

-Posicionamento dos elementos -> Algumas tags tem como padrão alguns posicionamentos. Então usando o primeiro exemplo:

<p> Vou <em> escrever um parágrafo aqui</em>   
<p> Aqui é outro texto </p>

</p>

A tag <em> não vai quebrar a linha do meu texto, porque o seu posicionamento a coloca na mesma linha do texto.   
Já a tag <p> por significar parágrafo, ela inicia uma nova linha. Quebrando a minha linha. Ela trabalha em bloco.

* Caracteres reservados - O HTML ignora os espaços que nós deixamos nos textos.   
  Caso eu queira que ele identique e use os espaços que eu dou no meu texto, eu preciso usar caracteres reservados, caracteres especiais, como “&nbsp” (É um caracter de espaçamento).

Alguns caracteres são reservados para o uso no HTML, como o <, “”, >.

Caso eu queira usárlos no meu texto, eu preciso usar um comando especial para isso, “&lt” (Do inglês, significa Lower Than) , “&gt (Do inglês, greater than) e “” (&quot).  
&apos( apóstofre)

* Anatomia Documento – O Documento precisa começar com a Tag “! Doctype html” – Para indicar para o navegador que você está escrevendo um.
* Títulos e Parágrafos – Cabeçalhos e parágrafos são importantes para a organização da página e para deixar a página mais atraente para o usuário. Mostrando o que é mais relevante no seu texto.   
  Normalmente, começamos uma página usando uma tag de Título (<h1>).  
  é importante ter atenção de que o 1 não representa o primeiro título, mas sim a ordem de importância na página.   
  Então, H1 é mais importante que o H2, que é mais importante que o H3 e assim sucessivamente.
* Listas – As vezes é necessário criar algum tipo de conteúdo em formato de listas. Essas listas são compostas por uma tag, a <Li> </Li> que já coloca todo contéudo separado.  
  É possível multiplicar a tag para a quantidade de itens que são necessários.   
  Basta usar a tag, com o sinal de vezes e número que você precisa. EX: <Li \*5   
  E ele vai criar 5 tags iguais.   
  É possível transformar essa lista em ordenada e não ordenada. A lista ordenada vai ser enumerada, e usa a tag *<Ol></Ol>*  
  Já a lista não ordenada, segue somente com os bullet points usando a tag *<UL> </UL>*
* Citações – Quando existe a necessidade de referênciar um texto, usar uma citação, nós temos algumas tags para isso.   
  Você primeiro tem o *<Blockquote>* que é usado para identificar que um texto externo foi citado.   
  O <Cite> que serve para indicar um link de onde esse texto externo foi retirado.   
  E o <q> que é usado para citações curtas que não precisam de parágrafos ou quebras de linha.
* Abreviações – Caso eu precise abreviar uma palavra e quiser explicar para o meu usuário o que aquela abreviação significa, para facilitar o entendimento, eu uso a tag <abbr>. Ela vai marcar a abreviação que eu coloquei com um pontilhado embaixo e quando eu descansar o mouse sobre ela, ela vai me dar o significado daquela palavra.
* Detalhes de contato – Quando eu quiser identificar o autor de uma página, colocando a informação do autor do código, eu posso usar a tag <address>.
* Lista de descrição – É possível montar uma lista de descrição, onde você vai pegar um item e descrever algo sobre ele. Para isso, vamos usar uma sequência de três tags.   
  Abrimos a lista de descrição com <DL> </DL>, Colocamos o item que vai ser descrito na <DT></DT> e a descrição fica dentro da tag <DD></DD>  
  Ex:   
  <dl>  
    
  <dt> Maça</dt>  
    
  <dd> Maça é uma fruta vermelha</dd>   
    
  </dl>
* Representação de códigos – Exitem duas tags mais usadas para apresentar o nosso código. A tag <code> vai ser usada quando eu precisar criar uma parte genérica do meu código. Ela não vai respeitar a formatação que eu colocar no código e sim a tradicional do HTML.   
  Já a tag <Pre>, cria um bloco de código, pois essa tag mantém os espaços em branco e os recuos que eu colocar no meu código.
* Elementos Genéricos – Usamos dois elementos genéricos no HTML. Um vai servir para agrupar conteúdo <div>, porque ele trabalha com uma ideia de bloco e o outro para agrupar texto <span> , porque ele trabalha com uma ideia de linha.  
     
  OBS: Contêiner – Muitas vezes precisamos agrupar contéudo relacionado em um documento para fins de estruturação ou estilização, ou mesmo demarcar uma região da página com alguma semântica específica (Um menu de navegação, o conteúdo principal, ETC). Para isso, temos dois agrupadores genéricos, Span e Div bem como contêineres semânticos.  
    
  -*Span – Agrupa um conjunto de elementos relacionados dentro de uma linha.   
  -Div – Agrupa elementos relacionados como um bloco (Força uma quebra de linha).  
  -Header – Define uma região como cabeçalho da página.   
  -Footer – Define uma região como rodapé da página.*

*-Nav- Define um menu de navegação, contendo vários links.   
-Main – Demarca o contéudo principal da página.   
-Aside – Demarca um contéudo separado do contéudo principal.   
-Article- Demarca um artigo, ou seja, um contéudo autocontido que possa ser lido fora do contexto da página.   
-Section – Demarca uma seção.*

* Links – Conhecendo a tag Ancora – A tag ancora é a tag <a> </a>. Que funciona carregando um link para aquele conteúdo.   
  A sua anatomia é composta pela abre e fecha a tag e um caminho para colocar o endereço para onde aquele link vai nos levar:  
  EX: <a href=””> Nome do botão</a>  
    
  Essa tag recebe atributos globais, como Tittle, Id, Class. E o Href, que indica referência, indica o caminho para onde o link vai me levar. E nele podemos usar:   
    
  - URL Completa   
  - Um fragmento da url   
  -Email   
  -Telefone  
  -Download   
    
  Outro atributo que o <a> pode ter junto do Href é o Target. Que me permite abrir o link na página atual (Como forma padrão) e usando o \_Blank, que vai abrir aquele link numa página nova. Mantendo a anterior aberta.   
  Ex: <a href=”http://google.com” target=”\_blank”> Google</a>
* Utilizando a tag Ancora (fragmento) – É uma tag usada para levar o usuário para fragmentos da página. Alguma parte da página específica, como um “Sobre mim”, ou alguma parte de um texto naquela página. Essa tag consegue se associar ao atributo ID para poder se localizar. Ele utiliza o # para identificar as configurações de cada ID.  
  EX: <a href=”#about”> Sobre mim</a>  
  <h1 id=”About”>Sobre mim</h1>
* Caminhos e URLS – URLs (Uniform Resource Locator) é uma sequencia de texto que define onde algo está localizado na web. A URL usam caminhos para encontrar arquivos.   
  Já os caminhos dos arquivos é onde, no explorador de arquivos, um recurso está localizado.
* Como navegar pelos diretórios – Podemos navegar pelos caminhos de algumas formas diferentes, todas elas através da tag <a></a>:   
    
  -Pelo mesmo diretório -> O mesmo diretório corresponde a navegar por aquivos que já estão dentro do meu projeto.   
    
  -Entrando em diretórios diferentes -> Corresponde a usar arquivos que estão em projetos diferentes, em diretórios diferentes.   
  Para isso, eu separo os endereços por “/”.   
  Ex: <a href=”outros/conteudo-a-element.html”> Arquivo em outra pasta </a>  
    
  -Saindo de diretórios -> Quando eu preciso pegar algo que está fora da minha pasta, fora do meu diretório, eu uso o comando “../” antes de digitar o arquivo que eu quero citar.   
  Ex: <a href=”../hyperlink.html”>arquivo na mesma pasta</a>  
     
  -Diretório raiz (Root ou pai) -> Diretório Raiz é principal é quando o arquivo está na mesma pasta do projeto. Para usá-lo, começamos a usar a nossa referência com “./”  
  Ex: <a href= “./urls-caminho.html”> Arquivo no diretório raiz </a>
* Caminhos absolutos e caminhos relativos- Caminhos Absolutos são caminhos que incluem o protocolo e nome de domínio. Ele sempre apontará para o mesmo local, pois é absoluto. E ele sempre vai ser absoluto, e precisa ter todo o caminho especificado.   
  EX: <Http://www.rocketseat.com.br/projeto/Index.html>  
    
  Já o relativo, é relativo a pasta que eu estou citando. Ele só vai poder me trazer aquele link se ele estiver dentro do caminho que eu estou usando. Ele vai apontar para lugares diferentes.   
    
  EX da diferença dos dois:  
    
  <a href=”google.com”> relativo: ir para o google </a>  
    
  <a href=”https:// google.com”> Absoluto: Ir para o google</a>
* Tabelas - Tabelas – O elemento <Table> nos permite criar tabelas dentro do HTML. Ele tem como vantagens criar visualização de dados via linhas e colunas, dar uma boa acessibilidade para a leitura dos dados. Porém ela tem como contra ser pouco flexível e precisa de estilizaçãopara melhor visualização.   
  E ela não deve ser usada para criar o seu layout.
* Organizando tabelas – Começamos organizando nossa tabela com o elemento <table>, Após ele usamos o elemento <caption> que vai descrever sobre o que é a nossa tabela. O nosso cabeçalho fica logo após dentro da tag <Thead>. Os dados da tabela ficam dentro da nossa tag <tbody>. E o nosso rodapé pode ser criado pela tag <Tfoot>, colocando os dados totais da tabela, por exemplo.  
  A tag <Tr> significa Table Row, ou seja, uma linha da tabela. Ela é usada para que eu passe os dados inseridos para a próxima linha.   
    
  Ex: <table>  
  <Caption> Pessoas por Idade</Caption>  
  <thead>  
  <tr>  
  <Th> Nome </Th>  
  <Th>Idade </Th>  
  </tr>  
  </Thead>  
  <Tbody>  
  <tr>  
  <td> Mayke</td>  
  <td> 35 anos</td>  
  </Tr>  
  </Tbody>  
  <tfoot>  
  <tr>  
  <td> Total:</td>  
  <td> 1 pessoa</td>  
  </tr>  
  </Tfoot>  
  </table>
* Tabelas Complexas – Para criar uma tabela complexa eu preciso saber como ela vai ser estruturada. Então segue o exemplo: Uma tabela que mostre o número de produção e venda de dois produtos em duas lojas diferentes.   
    
  Começamos criando o table, colocando a descrição da tabela com o Caption. Em seguida criaremos o <Thead> para começarmos o nosso cabeçalho, abrimos uma <TR> dentro para a linha e colocamos dentro desse <tr> 3 <th>, o primeiro sendo vazio.   
  Na próxima linha já temos os produzidos e vendidos, criamos 4 <th> diretos e colocamos os produzidos e vendidos.   
  Notamos que assim ainda fica desorganizado, então usaremos o atributo RowSpan=”2” (Que indica para aquela situação, quantas linhas ele deve ocupar na tabela), na primeira linha, para ocupar 2 linhas, empurrando os elementos para o lado.   
  Agora, vemos que sobraram 2 colunas vazias, então usaremos o atributo de coluna ColSpan= “2” nas duas lojas.   
  Em seguida, iremos criar o <Tbody> , abrir uma linha com o <tr>, colocar o nome do produto, porém ele será um <Th>, ou seja um cabeçalho, criaremos 4 deles para colocarmos o tanto de produtos produzidos e vendidos. Repetiremos esse passo para o próximo produto.   
  Para ainda organizar melhor a nossa tabela, colocamos uma tag <colgroup> para mexer nas nossas colunas. Dentro dela colocamos três tags <Col> e na segunda e na terceita utilizamos o atributo span =”2” para o background color nas colunas.   
  Para finalizar, usamos nos <th> o atritubo Scope=””, que serve para permitir que essa acessibilidade saiba que o escopo do cabeçalho é relacionado com, o agrupamento, ou a coluna, ou a linha.
* Cabeçalho – Head- O head é toda aquela parte do código que não é visível no navegador. Onde se faz configurações, contém informações como o título, links para o CSS, para o favicon.   
  A tag <meta> serve para definir metadados, como codificação de caracteres especiais e portabilidades para dispositivos mobiles. Ele normalmente virá com o atributo “name” para especificar a meta, “content” para conteúdo, mas há também como o atributo charset, para caracteres especiais. Além disso, o Meta ajuda os sistemas de buscas e os navegadores a interpretarem melhor a nossa página.  
    
  O Favicon é uma abreviação para “favorite Icon”, refere-se aos ícones dos favoritos a alguns anos atrás, mas nos dias de hoje acabou ficando este termo. No passado, ele era por 16pixels, agora com o avanço da tecnologia isso mudou.   
  Para colocarmos o ícone usaremos a tag <link> como atributo rel=”icon”, que seria para representar relação, no caso a tag link vai conter um ícone, depois o href para mostrar onde está esse ícone.   
  Ex: <link rel=”icon” href=”/icons/icon-48x48.png?v=cfca599”/>
* Posso ver e ouvir o HTML
* Vídeo – Para adicionar vídeos na minha página, eu posso usar a tag <Video>. Nela eu vou colocar o caminho do vídeo dentro do SCR:””, igual a tag image.   
  O problema da tag vídeo é que nem todo navegador suporta todos os tipos de vídeos nem todos os tipos de tags.   
  Eu posso usar a tag controls, para colocar a opção de play e pause no vídeo.   
  Posso colocar a tag autoplay para o vídeo começar automaticamente ao carregar a página, posso usar a tag muted para o vídeo começar mutado.
* Áudio – A caixa de áudio segue de maneira muito parecida com a tag vídeo. Variando as funções das tags baseado no navegador, com possibilidade de começar mutado, rodar o áudio em loop entre outras.
* Iframe – Tag usada para carregar vídeos e áudios (Conteúdos) de terceiros. Para não gastar a nossa banda com o download desses arquivos.   
  Os atributos mais usadas pelo Iframe são:   
    
  -SRC – Vai indicar o caminho desse arquivo.  
  -Height – Determina a altura do vídeo.   
  -Width – Determina a largura do vídeo.   
  -Title – Determina a acessibilidade do meu vídeo. Normalmente usa-se o título do vídeo.   
  -Allow – Essa tag vai trazer algumas possíbilidades que o site onde vídeo está hospedado disponibiliza.   
  -Allowfullscreen – Tag que possibilita ver o vídeo em tela cheia.   
  -Frameborder – Determina a borda do vídeo. O uso do 0 faz o vídeo não ter bordas.
* Images – Para colocar uma imagem na minha página, basta usar a tag <Image> .   
  Quando eu uso a tag, automaticamente ela puxa dois atributos. O SRC (Source), que vai dizer qual o caminho dessa imagem, de onde ele deve buscar essa imagem. E o ALT (Alternativa a imagem) é uma alternativa caso algo dê errado ao carregar a minha foto. Então é uma mensagem de texto referênciando aquela foto.  
  Outros atributos importantes para essa tag são:  
    
  -Title – Que coloca uma mensagem de texto para quando eu descansar o mouse em cima da foto.   
  -Width – Define uma largura para a minha imagem.   
  -heigth – Define uma altura para a minha imagem.   
  -Adicionar um link – Eu posso usar a tag <a href=”” para adiconar um link a imagem. Para quando eu clicar nela, ela me leve para algum outro site.   
    
  Obs: O Background Image é adicionado com a propriedade style. Mas é usada somente para estilo. Pois no css ela teria que ser configurada de uma maneira diferente.
* Figures – Caso eu queira criar um título ou dar créditos a uma foto, eu posso usar a tag <Figures>, para colocar dentro dela a minha tag <img>, e dentro da figure usar a <figcaption> que é uma tag que vai linkar o meu texto com a imagem específica que eu coloquei.   
  E caso eu queira transformar tanto a imagem como o texto num caminho clicavél, eu uso a tag <a> antes da tag figure.
* Introdução de SVG – SVG é uma marcação, estilo HTML, mas não é para textos e sim para fazer iamgens. Ela possuí elementos para gerar formas.   
  Normalmente usamos imagens rasteirizadas ( que são imagens trabalhadas por pixels) mas o SVG trabalha com imagens vetoriais (Que são imagens montadas por vetores)   
  As imagens vetoriais tem alguns benefícios, como serem mais leves, mais detalhadas, Não mostrarem pixels quando dão zoom, tem maior acessibilidade e SEO, pode ser editada via Css ou atributos.   
  Porém as suas desvantagens são: Pode ser mais complicado de trabalhar, Quando mais complexa a imagem, mais trabalho para o navegador e navegadores mais antigos não possuem suporte para essa tag.   
    
  Para o uso de fotografias, ainda é indicado o uso de imagens raxterizadas(Pixeladas).  
    
  Obs: Para fazer o SVG, normalmente, utilizamos programas para gerar esse código. Como o figma, por exemplo. Mas é possível fazer no HTML.  
  Por exemplo:   
    
  <SVG width=”200px” “height=200px”>  
   <Circle cx=”100” cy=”100” r=”80” stroke= “red”  
   stroke-width= “6” fill= “Blue” />  
    
  Width = Largura   
  Height = Altura   
  Circle = Circulo   
  Cy= Posicionamento do eixo Y  
  Cx=Posicionamento do eixo X  
  r=Raio do Círculo  
  Stroke = Cor Borda  
  Stroke-Width = Largura da Borda   
  Fill= Preenchimento