Trilha Fundamentar

* CSS
* Conhecendo o CSS – Css é um acrônimo para Cascading style Sheet, uma forma de escrever uma folha de estilos em cascata. O Css é um código para criar estilos no HTML, que como vimos, é a estrutura de todo o documento, Já o CSS seria a beleza do documento.   
  O CSS não é uma linguagem de programação, é uma linguagem style sheet, pois mesmo não sendo de programação, ainda possui uma ideia de sintaxe, e assim, precisamos aprender a forma correta de se escrever.
* Comentários – Os comentários no CSS não irão afetar seu código, mas pode ajudar a lembrar de blocos de códigos ou agrupar e organizar nosso código, deixa dicas para a leitura e ajuda outros a entenderem nosso código. É importantissímo que não esqueçamos de fechar um comentário aberto. Os comentários abrem com /\* e fecham com \*/  
  Você também pode usar um comentário para desabilitar partes do seu código, o que possibilitaria um debug mais fácil, ou caso deseje fazer alguma alteração sem quebrar todo o seu coódigo.
* Anatomia – Para criar um CSS que altere um elemento do HTML, temos que usar o nome do elemento que vamos alterar, abrir uma Chave { , no meio vamos ter propriedades e seus valores e fechar a chave }.   
  Toda propriedade é seguida de dois pontos : e um valor, que é encerrado com ponto e vírgula ;  
  Ex:   
  h1{   
   Color: Blue;  
   Font-Size: 60px;  
   background: gray;  
   }  
  Os elementos do CSS são:
* Selectors: Nesse caso o H1, que vai buscar no HTML a tag H1 e aplicar mudanças.
* Declaration: As chaves e tudo dentro delas.
* Properties: As coisas a serem alteradas.
* Properties Values: Os novos valores que estamos atribuindo a tais propriedades.
* Seletores – Os seletores são o que conectam um elemento HTML com o CSS. Existem vários tipos. Existe um seletor global, que vai se aplicar para todo o HTML, o asterisco \*. O seletor de ID que é usado através de um jogo da velha # e um Id do elemento HTML, class selector, que é representado por um ponto .
* Box Model – O CSS trabalha com o conceito de Caixas. Todos os espaços dele são caixas retangulares. Quase todos os elementos de uma página é considerado uma caixa. Posicionamento, tamanhos, espaçamentos, Bordas, cores. Essas caixas possuem as mesmas propriedades de uma caixa 2D, e tem como propriedades:  
     
  🡪Tamanho (Largura x Altura): Width e Height, respectivamente.   
  🡪Conteúdo: O Content   
  🡪Bordas: O Border   
  🡪Preenchimento Interno: O Padding   
  🡪 Espaços fora da Caixa: A Margin
* A Cascata (Cascating) – O C de CSS. A cascata se baseia na escolha do Browser de qual regra ele vai aplicar, caso haja muitas regras para o mesmo elemento. Seu estilo é lido de cima para baixo, ou seja, caso haja algum selector com informações conflitantes, o mais embaixo é o que será atribuido. Para isso, são levado em consideração três fatores:   
  -A origem do Estilo.   
  -A especificidade.   
  - A Importância .  
    
  🡪 A Origem do estilo – A origem segue a regra de prevalência do CSS que estiver Inline, na linha do código HTML, Depois a tag Style escrita no HTML e por ultimo a tag link.
* Especificidade – A especificidade é um cálculo matemático, onde cada tipo de seletor e origem do estilo possuem valores a serem considerados. Os mais fracos são o universal selector, o asterisco \*, com o valor 0. Em seguida, os tipos de elementos que tem valor de 1. Também temos a classe de seletor de atributos, com o valor de 10 , temos as ID, com um valor de 100 e por último os inlines que tem valor 100.
* A Regra Important – A regra important não é considerada uma boa prática. É melhor evitar o seu uso. Ela quebra o fluxo natural da cascata.
* At Rules – São regras relacionadas ao comportamento do CSS, começa com um sinal de @ seguido do identificador e do valor.  
  Ex:   
  @import – Serve para incluir um CSS externo.  
  @media – São regras condicionais para vários dispositivos.   
  @font-face – Serve para colocar fontes externas   
  @Keyframes – Serve para as animations do CSS.
* Shorthand – O shorthand é basicamente a ideia de junção de propriedades, para que elas sejam escritas de forma resumida e legível. Ao usar uma shorthand, ela não irá considerar propriedades anteriores, ou seja, caso eu a use, apenas ela será considerada, quaisquer propriedades anteriores serão substituídas pelas do shorthand.   
  Os valores que não forem especificados irão assumir o valor padrão.  
  Por fim, geralmente a ordem descrita não importa, mas, caso haja muitas propriedades com valores semelhantes, podemos encontrar problemas.   
  Ex:   
    
    
  {  
  /\* Background properties\*/  
  Background-color: #000;  
  Background-image: url( images/bg.gif);  
  Background-repeat: no-repeat;  
  background-position: Left top;

/\* Background shorthand\*/  
Background: #000 url(images/bg.gif) no-repeat left top;  
}

* Funções – As funções são como caixas onde eu vou inserir argumentos. Ela tem um nome, seguido de abre e fecha parenteses e recebe argumentos que são seus valores.  
  EX:   
  Color: RGB( 255, 0, 100);  
  Rgb – É o nome   
  (255, 0, 100) – São os argumentos, os valores que esse nome está recebendo.
* Valores e unidades de medida no CSS -   
  O CSS sempre vai ser escrito da forma em que temos uma “property: Value”  
  Ou seja, ele sempre vai ter uma propriedade e um valor.   
  Os valores podem ser reconhecido por esse termo ou como data types.
* Tipos numéricos e unidades comuns -   
  No css nós temos os tipos numéricos mais comuns, que são:   
    
  🡪 <integer> - Número inteiro como -10 ou 223   
    
  🡪<Number> - Número Decimal como 2.4, 64 ou 0.234  
    
  🡪<dimension> - É um <Number> com unidade de medida junto como 90deg, 3s, 8px  
    
  🡪<percentagem> - Representa a fração de outro número: 50%  
    
  Temos também algumas unidades comuns no CSS:   
    
  🡪 <length> Representa um valor de distância: px, em, Vw  
    
  🡪<angle> Representa um ângulo: Deg, rad, turn  
    
  🡪<time> Representa um tempo: S, M  
    
  🡪<resolution> Representa resoluções para dispositivos: Dpi.
* Distâncias Absolutas e Relativas - Distâncias absolutas são valores fixos e que não alteram o seu valor. O mais utilizado é o pixel. E não é recomendado usar os CM.  
  Ex:   
    
  CM – Centímetro – 1cm= 96px/2.54  
  IN – Inches (polegadas) – 1 in = 2.54cm = 96px  
  PX – Pixels – 1px = 1/96th of 1 in  
    
  Já as distâncias relativas, elas são relativas a um outro valor, podendo ser ao elemento pai, ou root, ou o tamanho da tela.   
    
  EM – Tamanho da fonte do elemento pai   
  REM – tamanho da fonte do elemento raiz (Root/HTML)  
  VW – 1% da viewport Widgh   
  VH – 1% da viewport height  
    
  A Viewport é a parte da tela que está sendo exibida. No caso dos navegadores web, é o que é exibido na janela/tela do documento. Conteúdos que estão fora do viewport só serão exibidos quando feito um scroll da área da visualização.  
    
  Normalmente, o tamanho da fonte padrão do navegador é de 16px. Para mudar esse valor, temos que fazer a alteração no root ou no elemento HTML.
* Porcentagens - As porcentagens são valores bem flexíveis. Em muitos casos é tratado da mesma maneira que as distâncias <Lenght>.  
  E sempre será relativo a algum valor.
* Position – O valor position representa um conjunto de coordenadas 2D:   
  Top, right, bottom, left e Center.   
  Usado para alguns tipos de propriedades.   
  Não confundir o valor <position> com a propriedade ‘position’
* Funções – Em programação, funções são reconhecidas por causar um reaproveitamento de código.  
  Ex:   
    
  Url () – Que é uma função que reaproveita um link já criado e tráz para o seu código.
* Strings e identificadores – As Strings são textos envolto em aspas, conteúdos que vem depois do elemento.  
    
  .box:: after{  
   content: Isso é uma string}  
    
  Já os identificadores são nomes. Podemos ter como nomes de cor, por exemplo:   
  Red, Black, Gold.
* Uma caixa dentro da outra
* Box Model - O Box model é fundamental para fazer layouts para Web porque ele vai te dar maior facilidade na hora de aplicar o CSS.  
  O que é o box model?   
  É o modelo em que cada elemento é representado por uma caixa retangular.   
  Essa caixa possui propriedades de uma caixa em 2 dimensões (Largura x Altura)  
    
  🡪Propriedades da caixa:   
    
  Tamanho (Largura x Altura) - Width/ Height  
  Conteúdo – Content  
  Bordas – Border   
  Preenchimento Interno – Padding  
  Espaços fora da caixa – Margin

Margin

Content

Padding

Border

* Box – Sizing – É o responsável pelo calculo do tamanho total da caixa (box). Por padrão, o navegador vai calcular o tamanho da caixa pelo *Content -box*  e vai somar com outros boxes. Por exemplo, numa caixa com width e height de 100px e um padding de 20% a caixa vai ficar com uma largura de 140px. Para que isso não aconteça, é possível mudar qual vai ser a referência para o calculo do tamanho da caixa adicionando a propriedade  *Box-Sizing: Border -Box;*  
  Dessa forma, o elemento vai ficar com a largura (Width) determinado, que no caso do exemplo anterior é de 100px.   
  Normalmente usa-se esse conteúdo como forma de “resetar” o box-sizing que vem por padrão nos navegadores.  
    
  O uso do \* no CSS serve para selecionar todos os elementos da página.
* Display Block vs Display Inline – O display é como as caixas se comportam em relação as outras coisas e o comportamento externo das caixas.   
  O **Display Block** ocupa toda a linha, colocando o próximo elemento abaixo desse. Ele respeita Width e Height e padding, margin e border irão funcionar normalmente.   
  Já o **Display Inline** coloca os elementos ao lado um do outro e não empurram outros elementos para baixo. Nesse tipo de display, o Width e Height não funcionam. E somento os valores horizontais de Margin, padding e Border
* Margin – Margin é o espaço (Margem) entre os elementos da página. Ele pode ser dividido em 4 valores:   
    
  Margin-top/ Margin-right/ Margin-bottom/ Margin- Left   
    
  Os seus valores normalmente são <lenght>, <porcentagem> e Auto.   
    
  Geralmente, eles são escritos de forma abreviada (Shorthand) para escrever o margin. Esse formato segue no sentido horário iniciado pelo top, seguindo para right, bottom e left.   
  margin: 12px 16px 10px 4px   
  Quando estão só três elementos ele faz top, lados(Right e left) e bottom   
  Quando estão só dois elementos ele faz top/bottom e Right e Left.  
    
  Quando eles estão usando o display block, é importante tomar cuidado com o margin collapsing que é quando o top de um elemento se junta com o bottom de outro elemento.
* Padding – O padding é o preenchimento interno da caixa. A propriedade padding pode ser escrita como no Margin. Considerando o top, right, bottom, left.   
  Igual ao margin, ele também é escrito de forma abreviada (Shorthand).   
  O padding pode ser escrito com valores (Values) de comprimento (px, em, rem) ou de porcentagem(%)   
  O padding poderá causar diferença na largura de um elemento.
* Border -outline - O Border são as bordas da caixa. Elas são escritas com o uso de três valores. Style, Width e Color.  
  O Style tem como propriedades o Solid, Dotted, Dashed, Double, Groove, ridge, Inset, Outset  
  Width tem a propriedade de length. Ou Seja, px, em ou rem.  
  A Color tem como propriedade o uso de cores.   
    
  O outline é muito semelhante ao border, mas difere em 4 sentidos:   
    
  - Ele não muda o tamanho da caixa, pois não é parte do box model.  
  -Poderá ser diferente de retangular.   
  -Não permite ajustes individuais.   
  - Mais usado pelo user agent para acessibilidade.