O QUE É CSS?

- Cascading style sheet
- código para criar estilos no HTML
- HTML é a estrutura, e o css é a beleza
- Não é uma linguagem de programação
- é uma linguagem style sheet

Comentários

- Não irá afetar o seu código
- ajuda a lembrar blocos de códigos
- deixa dicas para leitura
- ajuda outros a entenderem
- nunca esqueça de fechar um comentário aberto
- comentários começam com /* e terminam com */

Anatomia do css

```
h1{
   color: blue;
   font-size: 60px;
   background: aliceblue;
}
```

Seletores

- Conecta um elemento HTML com o CSS

Tipos

- global selector *
- element / type selector h1, h2, p, div
- ID selector # #container
- class selector . .red
- attribute selector, pseudo- class, pseudo-element e outros

Box-model

- você irá perceber que quase tudo são caixas do CSS
- posicionamentos, tamanhos, espaçamentos, bordas, cores
- caixa pode ficar ao lado uma da outra, ou acima
- elementos HTML são caixas

A cascata (cascading)

A sua escolha do browser de qual regra aplicar, caso haja muitas regras para o mesmo elemento.
Seu estilo é lido de cima para baixo
É levado em consideração 3 fatores
1. origem do estilo
2. especificidade
3. importância

Origem do estilo

inline > tag style > tag link
Especificidade
É um cálculo matemático, onde cada tipo de seletor e origem do estilo, possuem velores a serem considerados.

0. Universal selector, combinators e negation pseudo-class(:not())
1. element type selector e pseudo-elements (::before, ::after)
100.classes e attribute selectors{(type="radio")}
1000.inline

A regra !important
- cuidado, evite o uso
- não é considerado uma boa prática
- quebra o fluxo natural da cascata

At-rules
- Está relacionado ao comportamento do CSS
- começa com o sinal de @ seguido do identificador e valor.

Exemplos comuns :
@import /* incluir um CSS externo */ ex: "http://local.com/styte.css"
@media /* regras condicionais para dispositivos */ ex: @media (min-width: 500px)
@font-face /* fontes externas */
@keyframes /*animation */
ex: nameofanimations{
}

Shorthand
- junção de propriedades
- resumido
- legível

ex: font : italic bold .8em/1.2 arial, sans-serif;
background: #000 url (   )no-repeat left top;
Detalhes
- não irá considerar propriedades anteriores
- valores não especificados irão assumir o valor padrão
- Geralmente, a ordem descrita não importa, mas , se houver muitas propriedades com valores semelhantes, poderemos encontrar problemas.

Funções
- nome seguido de abre e fecha parênteses
- recebe argumentos

Exemplos
@important url("http://urlaqui.com/stylr.css")
{
color: rgb(255, 0, 100);
width: calc(100% - 10px);
}

Valores e unidades de medida
- Cada propriedades possui valores `property: value`
- estudo constante a fim de entender as propriedades e seus valores

Prática
- Como conhecer e estudar os valores na documentação?

* <color> <length>
- os termos podem variar. `value` ou `data types`

Tipos numéricos
- <integer> número inteiro como -10 ou 223
- <number> número decimal como -2.4, 64 ou 0.234
- <dimension> é um <number> com uma unidade junto : 90deg, 2s, 8px
- <percentagem> representa a fração de outro número: 50%

Unidades comuns
- <length> representa um valor de distância: px, em, vw
- <angle> representa um ângulo: deg, rad, turn
- <time> representa um tempo: s, ms
- <resolution> representa resoluções para dispositivos: dpi

Distâncias absolutas <length>
São fixas e não alteram seu valor.
- unidade Nome Equivalência
  cm centímetros 1cm=90px/2.54
  in Inches(polegadas) lin=2.54cm =96px
  px Pixels 1px = 1/96th of lin
- O mais comum e mais utilizado e o px
- não recomendado usar cm

Distâncias relativas
São relativas a algum outro valor, pode ser o elemento pai, ou root, ou o tamanho da tela.
Benefício: maior adaptação aos diferentes tipos de tela

Unidade	Relativo
em	Tamanho da font do pai.
rem	Tamanho da font do elemento raiz(root/html)
vw	1% da viewport width
vh	1% da viewport height

Porcentagem %
- Em muitos casos é tratado da mesma maneira que as distâncias <length>
- sempre será relativo a algum valor

Posições
Representa um conjunto de coordenadas 2D:
- top
- right
- bottom
- left e center

Usado para alguns tipos de propriedades
Não confundir com a propriedade `position`

Funções
Em programação, funções são reconhecidas por causarem um reaproveitamento de código.
- rgb( )
- hsl( )
- url( )
- calc( )

Strings e identificadores
- strings: texto envolto em aspas
- identificadores: red, black, gold;

BOX MODEL
- Fundamental para fazer layouts para a web
- maior facilidade para aplicar o css

O que é ?
uma caixa retangular. Essa caixa possui propriedades de uma caixa (2D)
- Tamanho (largura x altura) width | height
- conteúdo content
- bordas border
- preenchimento interno padding
- espaços fora da caixa margin

* Cada elemento na sua página, será considerado uma caixa.

box-sizing
Como será calculado o tamanho total da caixa?
content-box |border-box
```
*{box-sizing: border-box;}
div{
  box-sizing : border-box;
}
```

display: block vs display: inline
- como as caixas se comportam em relação às outras caixas
- comportamento externo das caixas

block
- ocupa a linha, colocando o próximo elemento abaixo desse
- width e height são respeitados
- padding, margin, border irão funcionar normalmente

inline
- elemento ao lado do outro
- width e height não funcionam
- somente valores horizontais de margin, padding e border

exemplo:
block: <p> <section>, todos os headings <h1><h2>...
inlline: <a> <strog>

margin
Espaços entre os elementos
- margin-top
- margin-right
- margin-bottom
- margin-left
- values : <length> <percentagem> auto

Ex: div{
   margin: 12px 16px 10px 4px;
   margin: 12px 16px 0;
   margin: 8px 16px;
   margin: 8px;
}

padding
preenchimento interno da caixa
- padding-top
- padding-right
- padding-bottom
- padding-left
- values: <legth> <percentagem> auto

exemplo:
div{
   padding: 12px 16px 10px 4px;
   padding: 12px 16px 0;
   padding: 8px 16px;
   padding: 8px;
}

border (e outline)
as bordas da caixa
- value: <border-style> <border-width> <border-color>
- style: solid dotted dashed double groove ridge inset outset
- width: <length>
- color: <color>

e outline?
difere em 4 sentidos:
- não modificada o tamanho da caixa, pois não é parte do box model
- podera ser diferente de retangular
- não permite ajuste individuais
- mais usado pelo user agent para acessibilidade

CORES
Usamos CSS para alterar cores do nosso documento
Tipos
- background-color (para caixas)
- color (para textos)
- border-color (para-caixas)
- outros...

Valores
Podemos definir os valores por:
- palavra-chave (blue, transparent)
- hexadecimal (#990011)
- funções: rgb, rgba, hsl, hsla

Background
- define um fundo para o nosso elemento
- sua área de atuação é a caixa toda
- por padrão, é transparente

Exemplos:
- usar cores sólidas
- usar imagens
- controlar: posição das imagens, se elas se repetem ou não, o tamanho delas na caixa.
- usar cor e imagem juntas
- usar com gradiente

Exemplos:
background
background-color: rgb(55, 100, 50);
background-image: url( )
background-repeat: no-repeat (não deixa que a imagem repita) repeat-y;(vai repetir só no eixo y) repeat-x(vai repetir só no eixo x)
background-position: podemos mudar o eixo horizontal e vertical (right, center, top, bottom, left.
background-size: mudamos o tamanho da imagem (contain, cover, %, em, px...)
background-origin: comportamento da imagem(border-box, content-box,padding-box)
background-clip:
background-attachment:
shorthand
background: (linear-gradient)(radial-gradient)
múltiplos valores:(separar por vírgulas)

Posicionamentos
Controlar onde, na página, o elemento irá ficar, alterando o fluxo normal dos elementos.
Name: position
- position: static; seguem o fluxo normal
- position: relative;
- position: absolute;
- position: fixed;

Flexbox
- Nos permite posicionar os elementos dentro da caixa
- controle em uma dimensão (horizontal ou vertical)
- alinhamento, direcionamento, ordenar e tamanhos

Exemplo:
```
div{
 display: flex;
}
```
flex-direction
- qual a direção do flex: horizontal ou vertical
- row | column

Alinhamento
- justify-content
- align-items

Grid
- posicionamento dos elementos dentro da caixa
- posicionamento horizontal e vertical ao mesmo tempo
- pode ser flexível ou fixo
- cria espaços para elementos filhos habitarem

Fontes e textos
Tipografia
- negrito
- tamanho
- estilo

Basic Font Properties
- font-family
- font-weight
- font-style
- fonte-size

Font family
- Tipo de fonte de um elemento
- lista de fontes e ordem de prioridade
- inclui *fallback* font

```
p{
font-family: "Times New Roman", Times, serif;
}
```

Font weight
- Peso da fonte

Font-style
- O estilo da fonte (italic, oblique ...)

Fonte-size
- Tamanho da fonte (px, rem., em)

Web fonts
- fontes do sistema x fontes da web
- como usar fontes para web?

*@font-face
*@import
*link

Letter-spacing
- Espaços entre caracteres

word-spacing
- Espaçamento entre palavras

Line-height
- espaços entre linhas
- pode ser com unidades ou sem unidades de medida
- comuns: 1.5 ou 2

text-transform
- transformação do texto

text-transform: uppercase; (capitalize | lowercase | none)

Text- decoration

- aparência decorativa de um texto
- line: underline | overline | line-through
- podemos aplicar mais de um valor
- style: wavy | dotted | double | dashed | solid
- color: <color> values

Text-align

- alinhamento de um texto

text-align : center; (left, right, center, justify)

Text-shadow

- Sombra aplicada a um texto
- permite múltiplos valores

Seletores

Conecta um elemento HTML com o css a fim de alterar o elemento.
Tipos:

- element selector : Todos os elementos HTML

Ex:

```
p{
margin: 0;
}
```

- ID selector : um elemento que tenha um 'id'. Cada id deverá ser único

Ex:

```
#title{
margin: 0;
}
```

- Class selector : os elementos que contenham um atributo 'class'.Podemos ter uma ou mais classes.

Ex:

```
.title{
 text-align: justify;
}
```

- Attribute selector: um elemento que tenha um atributo específico.

Ex:

```
[title]{
color: red;
}
```

- Pseudo-class selector: elementos em um estado específico

Ex:

```
p:hover{
color: blue;
}
```

Múltiplos

Você pode selecionar múltiplos elementos e aplicar alguma regra css para todos eles.
usamos uma separação por vírgulas para isso.
Ex:

```
h1, p, a{
color: blue;
}
```

Combinadores
Combinadores, pois eles trabalham para buscar e combinar seletores a fim de aplicar uma estilização
Descendant combinator

- identificado por um espaço entre os seletores
- busca um elemento dentro de outro

body article h2
Ex:

```
<body>
      <article>
           <h2>Um titulo</h2>
       </article>
</body>
```

CSS:

```
body article h2{
   color: red;
}
```

Child combinator

- identificado pelo sinal > entre dois seletores
- seleciona somente o elemento que é filho do pai
- elementos depois do filho direto serão desconsiderados

body > ul > li
Ex:

```
<body>
    <ul>
    <li>Item 1</li>
    <ul>
    <li>Item 2</li>
    </ul>
</ul>
</body>
```

CSS:

```
body > ul > li{
    color: blue;
}
```

Adjacent síbling combinator

- Identificado pelo sinal + entre dois seletores
- seleciona somente o elemento do lado direto que é o irmão direto na hierarquia.

h1 + p
Ex:

```
<h1>Título.</h1>
   <p>Este é um parágrafo.</p>
<p>Mais um parágrafo.</p>
```

Css:

```
h1 + p{
color: red;
}
```

General síbling combinator

- Identificado pelo sinal ~ entre dois seletores

- seleciona todos os elementos irmãos

h1 ~ p
Ex:
```
<h1>Título.</h1>
  <p>Este é um parágrafo.</p>
<p>Mais um parágrafo.</p>
```
CSS:
```
h1 ~ p{
   color: red;
}
```
Pseudo-classes
É um tipo de seletor que irá selecionar um elemento que estiver em um estado específico
Ex: é o primeiro elemento dentro de uma caixa, ou o elemento está com o ponteiro do mouse em cima dele.
Pseudo-classes começam com 2 pontos seguido do nome da pseudo class
:pseudo-class-name
Selecionando elementos com pseudo-classes
- :first-child (primeiro filho)

Ex:
```
<ul>
   <li>Gratidão</li>
   <li>Esperança</li>
   <li>Fé</li>
</ul>
ul li:first-child{
   font-weight: bold;
}
```
- :nth-of-type()

```
<article>
   <h3>Gratidão</h3>
   <p>ser gratooooooooo</p>
   <p>ter feeeeeeeeeeee</p>
   <p>ser felizzzzzzzzz</p>
</article>
article p:nth-of-type(1){
   font-weight: bold;
   font-size: 18px;
}
```
- :nth-child()

```
<article>
   <h3>Gratidão</h3>
   <p>ser gratooooooooo</p>
   <p>ter feeeeeeeeeeee</p>
   <p>ser felizzzzzzzzz</p>
</article>
article p:nth-child(3){
   font-weight: bold;
   font-size: 18px;
}
```

Ações de usuário
- :hover
- :focus

Atributos
- :disabled
- :required

Pseudo-elements
Com pseudo-elements nós podemos adicionar elementos HTML pelo próprio CSS
::pseudo-element-name
Exe:
- ::before
- ::after
- ::first-line

LAYOUTS CSS
Layout tem a ver com a maneira que os elementos estão distribuídos na tela
Normal flow
Ou flow layout é a maneira que os elementos `block` e `inline` ficam na página.
```
<p>Texto block com<strong>inline</strong>dentro</p>
```
Tables
É a maneira de trabalho onde a tag `table` recebe um display `table` fazendo com que os elementos internos com `td` e `tr` possam ser usados para montar uma tabela.
```
<table>
<tr>
   <td>HORA</td>
   <td>20:00</td>
</tr>
   <tr>
         <td>HORA</td>
         <td>20:37</td>
   </tr>
</table>
```
Tabless
Uso das propriedades `float`, `clear` para que os elementos possam mudar de posição na tela.
```
<div style="float: left">
   esquerda
</div>
<div style="float: right">
   direita
</div>
<div style="clear:both">
   normal
</div>
```

Flexbox
A caixa se torna flex, fazendo com que os elementos internos possam receber melhor:
- alinhamento
- ordenação

- tamanhos flexíveis

Terminologia
Flex Container
    flex item
Nesting
Axis
  main
    start, end
  cross
    start, end
Flex sizing
- flexível
- altera width/height dos itens para preenchimento dos espaços do flex container

Propriedades do Flex Container
- direção dos itens
- multi linhas
- alinhamento (principal, cruzado)
- espaços entre os itens

Direção dos itens
- Flex é uma dimensão (horizontal ou vertical)
- podemos mudar a direção com `flex-direction`
- valores: row, row-reverse, column, column-reverse

flex-wrap
- Podemos usar multi linhas
- cada nova linha, um novo flex container

flex-flow
- shorthand
- flex-direction || flex-wrap

Ex:
```
.box{
display: flex;
flex-flow: column wrap;
}
```

Justify-content
- Alinhamento dos elementos dentro do container
- distribuição dos elementos

Valores
- flex-start
- flex-end
- center
- space-around
- space-between
- space-evenly

Align-items
- alinhamento dos elementos no eixo cruzado

Valores
- stretch
- flex-start
- flex-end
- center

Gap
- Espaços entre os elementos

Valores

Unidades de medidas
- fixas: px, pt
- flexíveis:%, em, rem

Propriedades para os itens
- flex-basis (Largura e altura dos itens com flex-basis)
- flex-grow (Crescimento e adaptação dos itens)
- flex-shrink (Encolhimento e encaixe dos elementos)
- flex
- order

Shorthland para flex-grow flex-shrink flex-basis