Linguagem de Programação – CSS

Abertura
Descrição
Nestas próximas aulas, vamos ver os fundamentos do CSS, como funciona, o que é o CSS afinal de contas, conceitos básicos como cascatas, especificidade, box model e muito mais.

Para consumir esse curso, é interessante que você já tenha um conhecimento de HTML.
Vamos embarcar nessa jornada?

Conhecendo o CSS

Descrição
Começaremos nossa introdução pelo que exatamente quer dizer CSS, que é um acrônimo para Cascading Style Sheet, uma forma de escrever uma folha de estilos em cascata, mas ainda precisamos saber o que exatamente é uma folha de estilos e uma cascata, mas falaremos disso mais pra frente.

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, a parte bonita.

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.

Podemos na nossa página HTML colocar um H1 como titulo e no CSS colocar a hora que quisermos a cor deste titulo que está em H1.

1º Exemplo.

No HTML - <h1> Nosso título da página</h1>

No CSS – h1 { color: blue; }

Não esquecer de linkar o arquivo .css na página html.

<link href=”nome-do-arquivo.css” rel=”stylesheet”>

2º exemplo.

No próprio .html colocar o código css inline

<h2 style=”color: pink;”> Nosso título da página</h2>

3º Exemplo.

<h2 style=”background-color:yellow;”>nosso título de página com background-color</h2>.

4º Exemplo.

Podemos adicionar quantas propriedades quisermos, desde que terminemos cada uma com um ponto e virgula ;

```
<h2 style="
color:brown;
background-color: blue;">nosso novo título</h2>
```

Uma propriedade CSS e um par de valores como colo:green; é chamado de DECLARAÇÂO.

Origem do CSS

Descrição
Vamos primeiro aprender a adicionar um estilo no nosso documento HTML, certo?

Temos 4 maneiras de adicionar CSS ao nosso elemento HTML.

- INLINE – Atributo style
- <STYLE> - Tag html que irá conter o CSS
- <LINK> - Arquivo CSS externo
- @IMPORT – Arquivo CSS externo

Começaremos pelo inline, que é dentro do próprio HTML, através da tag style, utilizada das seguintes formas:

```
<h1 style="color: blue;">Título
<strong style="color: red;">alo</strong>
</h1>
```

Ou na head do HTML, assim:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Document</title>
<style>
h1 {
color: blue;
}

strong {
color: red;
}
</style>
</head>
```

Porém, a forma mais comum, é através da tag link, onde vamos linkar um documento CSS externo, um outro arquivo para nosso documento HTML, feito da seguinte forma:

<link rel="stylesheet" href="style.css">
Neste caso, o nosso documento CSS se chama style.css e sua relação com o HTML é de stylesheet.

A última forma é através do @import, que é na verdade uma regra do CSS, portanto, deve ser usada dentro do css, ao invés de dentro do HTML, como as duas primeiras formas, e seu uso é mostrado a seguir:

@import 'https://fonts.googleapis.com/css2?family=Roboto:wght@300&display=swap'
Não é recomendado seu uso, pois leva um pouco mais de tempo do que através da tag link, fazendo a página ficar menos responsiva, demorando mais para o carregamento da mesma.

Comentários no CSS

Descrição

- Não irá afetar seu código.
- Ajuda a lembrar blocos de códigos.
- Deixa dicas para leitura.
- Ajuda outros a entenderem.
- Nunca esqueça de fechar um comentário aberto.
- Desabilita uma parte qualquer do nosso código.

Os comentários no css não irão afetar seu código, mas pode nos ajudar a lembrar de blocos de códigos ou agrupar e organizar nosso código, deixa dicas para a leitura e ajuda os outros a entenderem nosso código.

Apenas não esqueça de fechar um comentário aberto.

Os comentários abrem com /* e terminam com */ .

```
h1 {
color: blue;
} /* essa linha de comando faz todo meu título usando h1 ficar em azul */
```

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 código.

```
/*h1 {
color: blue;
}*/
```

Anatomia do CSS

Descrição
Nesta aula, falaremos sobre a anatomia do CSS, como criar esse CSS para poder alterar algum elemento do HTML.

Na verdade, é bem simples, temos o nome de algum elemento, uma chave aberta e uma fechada embaixo, e no meio vamos ter propriedades e seus valores.

Toda propriedade é seguida de : (dois pontos) e um valor e um ; (ponto e vírgula) para encerrar essa ideia de valor.

Abaixo um exemplo de CSS:

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

Os elementos do CSS são então:

Selectors: Nesse caso o h1, que vai buscar no HTML a tag h1 e aplicar as mudanças.

Declaration: As chaves abrindo, fechando e tudo dentro delas.

Properties: As coisas a serem alteradas.

Property values: Os novos valores que estamos atribuindo a tais propriedades.

Definindo

H1 – Selector

h1 { color: blue; font-size: 60px; background: gray; } - Declaração

Color, font-size e background – properties

Blue, 60px e gray – properties Values

Alguns atributos e valores para textos.

Style=”font-size: 60px” – altera o tamanho do texto

Style=”font-family: courier new;” – altera a fonte do texto

Style=”font-style: italic;” – altera o texto para itálico

Style=”font-weight: bold” – altera o texto para negrito

Etiqueta de estilo

As páginas da web modernas têm muito estilo, o que significa que elas têm muitas propriedades para torna-las ótimas. Uma maneira de adicionar estilo a um grupo de elementos é definir um estilo para cada elemento individualmente.

Exemplo:

```
<h3> Lista dos Faraós </h3>

<p style=”font-family: cursive;”> Tutankhamum</p>

<p style=”font-family: cursive;”> Cleópatra</p>

<p style=”font-family: cursive;”> Rmases II</p>
```

Existe uma maneira mais fácil de adicionar o mesmo estilo a um tipo de elemento.

1. Começamos adicionando tags de style dentro das tags head. Dentro das tags de style, especificamos qual o elemento estilizar com um seletor de tags, como p ou h3.
2. Em seguida, adicionamos uma chave de abertura, { , seguida por uma chave de fechamento, }.
3. Qualquer declaração que adicionarmos a partir da chave de abertura será aplicada a todos os elementos p na página da web.

```
<html>

<head>

<style>

p {

font-family: cursive;

}
```

Quando adicionamos p { seguido font-family:cursive; , e então } , criamos uma regra CSS.

Podemos adicionar outras declarações como font-family: courier new desde que estejam em outra linha. Usamos a tag style para poder estilizar vários elementos ao mesmo tempo.

```
p {

font-family: cursive;

color: purple;

font-weight: bold;

font-style: italic;

}

</style>

</head>

</html>
```

Estilizando o corpo (body)

Se quisermos que um estilo se aplique a toda a página, podemos usar o seletor de BODY.

Usamos o backgroud-color para definir o plano de fundo para todo o corpo. Estilo como color:black; , quando definimos dentro da regra do BODY, será aplicado a todos os elementos dentro da tag do BODY.

```
<html>

<head>

<style>

body {

background-color: lightblue;

}

</style>
```

Podemos estilizar quantos elementos quisermos dentro das tags de style, desde que adicionemos um espaço entre as diferentes regras.

```
<style>

body {

background-color: lightblue;

}

h1 {

font-familly: helvética;

}

</style>
```

Usar a tag BODY é uma das maneiras mais fáceis de adicionar regras de sobreposição em elementos.

Vinculando Folhas de estilo

Para tornar páginas da Web maiores mais gerenciáveis, podemos mover nosso CSS para uma folha de estilo ou um arquivo especial apenas para estilizar a página da web. Criamos um arquivo .CSS para podemos estilizar o nosso .HTML.

Dentro do arquivo .CSS

```
body {
Font-family: helvética neue;
Text-align: “center”;
}
h1 {
color: darkblue;
}
Img {
Border-radius: 10px;
Margin-top: 8px;
Margin-left: 2px;
Margin-right: 2px;
}
```

Para incluir uma folha de estilo em um arquivo HTML, usamos o elemento link. <link> é um elemento vazio e vai dentro do elemento head.

Para sabermos que tipo de arquivo incluir, a tag do LINK de abertura precisa do atributo REL definido usando REL=”STYLESHEET” . Para especificar a localização da folha de estilo, defina o atributo HREF como “STYLE.CSS” .

Uma vez a folha de estilo incluída, quaisquer alterações na folha de estilo tornam-se visíveis na página da Web.

Dentro do código HTML no arquivo .HTML

```
<head>
<link href=”arquivo.css” rel=”stylesheet”>
```

Seletores

Os seletores são o que conectam um elemento HTML com o CSS, existem vários tipos, inclusive, nós vimos um na última aula, o Element/Type selector, mas também temos o seletor global, que é um * (asterisco), ID selector, que é # (cerquilha, cardinal) e o ID do elemento HTML, class selector, que é um . (ponto) e o nome da classe, e mais alguns que podemos entender mais tarde no curso.

- Selectors – conecta um elemento HTML com o CSS
- Global Selector - * (asteristico)
- Element / type selector – h1 , h2 , p , div
- Id Selector - #box , #container
- Class Selector - .red , .m-4

- Atribute Selector , Pseudo-class , Pseudo-element , e outros

  Os seletores que mais usaremos serão realmente apenas os anteriormente citados, e vamos entrar em exemplos de como usá-los:

  HTML

```
<div id="container" class="m-40">
<h1>Título</h1>
<h2>Subtitulo</h2>
</div>
```

-

  CSS

```
/* ID selector */
#container {
width: 200px;
}

/* Class selector */
.m-40 {
margin: 40px;
}

/* Element/Type selector + Agrupamento de seletores */
h1, h2 {
color: blue;
font-size: 60px;
background: gray;
}
```

Se houver vários elementos do mesmo tipo em uma página da Web, seletores como h1, h2 ou p alteram todos esses elementos.

Se quisermos selecionar um ou mais elementos, podemos definir um atributo de CLASS para os elementos exatos que queremos alterar.

As classes não são únicas, então podemos definir a mesma classe para vário elementos.

Para definir uma classe como seletor em CSS, adicione um ponto (.) seguido do nome da classe. Nesse caso nosso abaixo é .gray-element .

```
.gray-element {

Blackground-color: lightgrey;

}
```

Um seletor e suas propriedades formam uma regra CSS

Configurando tamanho e bordas

Tamanho

Quando bem feito, os sites fornecem as informações corretas de maneira rápida e fácil. Vamos aprender sobre uma das maneiras mais fácies de priorizar informações: alterar o tamanho dos elementos ou adicionar bordas.

Para alterar a altura de um elemento, usamos a propriedade HEIGHT.

Para alterar a largura de um elemento, usamos a propriedade WIDTH

Dentro do arquivo . CSS

```
Img {

Height: 100px;

Width: 150px;

}
```

Medimos a altura (height) e a largura (width) de um elemento em pixels, como 50px. Pixels são pequenos pontos que compõe o que você vê na tela.

Bordas

Elementos em uma página da web podem ter bordas ao redor deles. Para ver como fica, adicione a propriedade border.

Podemos definir a largura (width) de uma borda adicionando um número em pixels logo após o valor da propriedade solid.

Para definir a cor, vamos adicionar red no final do valor da cor para red.

```
.coupon {

border: solid 10px red;

}
```

Border-radius é uma propriedade que arredonda os cantos de um elemento. Se definirmos o raio para 10px, a borda curva 10px antes do canto.

```
p {

border: radius 10px;

backgroud-color: lightblue;

border: solid;

}
```

Essa propriedade funciona em todos os elementos, até mesmo em imagens. É uma maneira fácil de fazer com que as imagens fiquem ótimas em uma página da Web.

```
img {

border: radius 100px;

border: solid 5px black;

}
```

Para tornar uma imagem um círculo, definimos border-radius para metade da largura de uma imagem. Só funciona se a imagem for um quadrado.

```
img {

heigth: 100px;

width: 100px;

border: radius 50px;

border: solid 5px;

}
```

Construindo com o modelo de caixa

Nesta aula falaremos sobre o conceito de caixas, já que o CSS trabalha com essa ideia de caixas, ou seja, o box model. Mas o que exatamente é esse 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. Ao entender os conceitos do Box Model muitas questões do CSS começam a fazer sentido.

- Você irá perceber que (quase) tudo são caixas no CSS
- Posicionamentos, tamanhos, espaçamentos, bordas, cores
- Caixas podem ficar uma ao lado da outra, ou acima
- Elementos HTML são caixas.

Cada elemento é representado como uma caixa retangular
Essa caixa possui propriedades de uma caixa em 2 dimensões (largura x altura).

- Tamanho (largura x altura): width e height, respectivamente
- Conteúdo: content
- Bordas: border
- Preenchimento interno: padding
- Espaços fora da caixa: margin

Quase todo elemento de uma página é considerado uma caixa: Posicionamentos, tamanhos, espaçamentos, bordas, cores, então, em suma, elementos HTML são caixas, assim como quase tudo no CSS.

BOX SIZING

Descrição
O box-sizing é o responsável pelo cálculo do tamanho total da caixa (box).

HTML:
```
<div>
CSS é incrível!
</div>
```

CSS:
```
div {
width: 100px;
height: 100px;
border: 1px solid red;
margin: 10%;
}
```

Quando o padding é adicionado (padding: 0 20px;) faz com que aumente a largura da caixa, deixando de respeitar os 100px de largura:

```
CSS:
div {
width: 100px;
height: 100px;
border: 1px solid red;
margin: 10%;

Padding: 0 20px;
}
```

E é por isso que é tão importante conhecer a propriedade do box-sizing.

Por padrão o navegador vai calcular o tamanho da caixa pelo content-box e vai somar com os outros boxes, no exemplo acima no lugar de 100px a caixa vai ficar com uma largura de 140px. Para que isso não aconteça, é possível mudar qual vai ser a referência para o cálculo 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 citado é de 100px.

```
CSS:
div {
width: 100px;
height: 100px;
border: 1px solid red;
margin: 10%;

Padding: 0 20px;

Border-sizing: border-box;
}
```

Normalmente usa-se o código abaixo como forma de "resetar" o box-sizing que vem por padrão nos navegadores.

```
* {
box-sizing: border-box;
}
```

O seletor * seleciona todos os elementos da página.

DISPLAY-BLOCK-INLINE

Descrição
display: block; vs display: inline;
Como as caixas se comportam em relação as outras caixas
Comportamento externo das caixas
Display Block
Ocupa toda a linha, colocando o próximo elemento abaixo desse

No html

um conteúdo outro conteúdo

width e height são respeitados

No css

Div {

Height: 100px;

}

padding, margin, border irão funcionar normalmente

div {

width: 80px;

height: 40px;

margin:10px;

padding: 20px;

border: 1px solid red;

}

  <section>, todos os headings <h1> <h2>...

Display Inline

Os elementos ficam ao lado do outro e não empurram outros elementos para baixo

html

Um <stgrong> texto qualquer

Width e height não funcionam

Span{

Height: 100px;

}

Somente valores horizontais de margin

Span {

Margin: 10px; /* só aplica na horizontal*/

Border: 1px solid green;

<a>

Para transformar um bloco em linha usamos o a propriedade display com seu atributo inline

Para transformar uma linha em bloco usamos a propriedade display com seu atributo block

MARGIN

As Margens geram espaço redor de um elemento, fora de quaisquer preenchimento e bordas.

Para alterar as margens em todos os quatro lados de um elemento, usamos a propriedade MARGIN.

Margin, é o espaço (margem) entre os elementos

Podemos dividir o margin em 4 valores:

/* margin-top | margin-right | margin-bottom | margin-left */
values: <length> | <percentage> | auto

Geralmente usamos uma forma abreviada (shorthand) para escrever o margin. Esse formato segue o sentido horário iniciando pelo top, seguindo para right, bottom e left.

O margin é aplicado em elementos com display block
Cuidado com o margin collapsing que é quando o top se junta ao bottom

O navegador faz a leitura destes 4 valores da seguinte maneira, começando pelo top (em cima) depois right (direita) depois botton (em baixo) e por fim left (esquerda).

- Margin: 10px 20px 30px 40px; - 10 top – 20 right – 30 botton – 40 left
- Margin: 10px 20px 30px; – 10 top - 20 right e left – 30 botton
- Margin: 10px 20px; - 10 top e botton – 20 right e left
- Margin: 10px; - 10 para todos os elementos

O body como padrão já tem 8px de margin

img {

margin: 30px 0 0 0;

}

PADDING

Para alterar os preenchimentos em todos os quatros lados de um elemento, usamos a propriedade PADDING.

```
P {
Padding: 50px;
Border:1px solid black;
Backgroud-color: lavander;
}
```

PADDING-LEFT

Para alterar o espaçamento no lado esquerdo de um elemento, usamos a propriedade PADDING-LEFT.

```
.leftwing {
Padding-left: 200px;
Background-color: crimson:
}
```

PADDING-RIGHT

Para alterar o espaçamento no lado direito de um elemento, usamos a propriedade PADDING-RIGHT.

```
.rightwing {
padding-right: 200px;
background-color: lightblue;
}
```

PADDING-TOP

Para alterar o espaçamento no topo de um elemento, usamos a propriedade PADDING-TOP.

```
.lightObjects {
padding-top: 20px;
}
```

PADDING-BOTTOM

Para alterar o espaçamento na parte inferior de um elemento, usamos a propriedade PADDING-BOTTOM.

```
.lightObjects {
padding-top: 20px;
padding-bottom: 60px;
}
```

TEXT-ALIGN

Para alterar o alinhamento do conteúdo, usamos a propriedade TEXT-ALIGN.

Para centralizar o conteúdo, definimos a propriedade TEXT_ALIGN como CENTER.

```
.center {
Text-align: center;
Border-radius: 100px;
Background-color: lightblue;
}
```

Estofado é o espaço entre o conteúdo e a borda de um elemento.

MARGIN-TOP

Para alterar as margem no topo de um elemento, usamos a propriedade MARGIN-TOP.

```
img {
margin-top: 30px;
border: 5px solid;
}
```

MARGIN-BOTTOM

Para alterar as margem na parte inferior de um elemento, usamos a propriedade MARGIN-TOP.

```
img {
margin-bottom: 30px;
border: 5px solid;
}
```

MARGIN-LEFT

Para alterar a margem no lado esquerdo de um elemento, usamos a propriedade MARGIN-LEFT.

```
img {
margin-left: 30px;
border: solid;
}
```

MARGIN-RIGHT

Para alterar a margem do lado direito de um elemento, usamos a propriedade MARGIN-RIGHT.

```
img {
margin-right: 30px;
border: solid;
}
```

Embora ambos adicionem espaçamento ao redor deles, margens e preenchimentos são diferentes.

O espaçamento adiciona espaço entre o conteúdo e a borda, enquanto a margem aciona espaço entre a borda e outros elementos.

Adicionando Espaçamento com uma linha

Qualquer elemento tem quatro preenchimento contados no sentido horário de cima para a direita, para baixo para a esquerda.

```
img {
padding-top: 20px;
padding-right: 20px;
padding-botton: 20px;
padding-left: 20px;
border: solid 2px gray;
width: 100px
heght: 100px;
background-color: White;
}
```

Podemos definir cada valor de espaçamento em uma linha usando a propriedade de PADDING e um valor de pixel para cada lado.

```
img {
padding: 20px 20px 20px 20px;
border: solid 2px gray;
width: 100px
heght: 100px;
background-color: White;
}
```

Para posicionar o conteúdo mais abaixo dentro de um elemento, adicione um espaçamento superior definindo o primeiro valor como 100px.

```
img {
padding: 100px 0px 0px 0px;
border: solid 2px gray;
width: 100px
heght: 100px;
background-color: White;
}
```

Para definir o espaçamento correto para algo como 40px, alteramos o segundo valor. Para definir o espaçamento esquerdo para 100px, alteramos o quarto valor.

```
img {
padding: 0px 40px 0px 100px;
border: solid 2px gray;
width: 100px
heght: 100px;
background-color: White;
}
```

Para adicionar espaçamento na parte inferior, defina o terceiro valor como 100px.

```
img {
padding: 0px 0px 100px 0px;
border: solid 2px gray;
width: 100px
heght: 100px;
background-color: White;
}
```

Quando não queremos adicionar espaçamento a um lado, podemos adicionar 0 e deixar de fora px, pois é opcional.

```
Img {
padding: 0 0 100px 0;
border: solid 2px gray;
width: 100px
heght: 100px;
background-color: White; }
```

Estilizando cantos com uma linha

As taquigrafias trabalham com as propriedades MARGIN e BODER-RADIUS, tornando possível criar estilos interessantes com menos linhas de códigos.

Para definir as marges direita e esquerda para 15px em uma linha, podemos usar a abreviação de MARGIN.

```
button {
margin: 0 15px 0 15px;
width: 100px;
height: 100px;
border-radius: 15px;
background-color: red;
border: solid 5px dimgray;
}
```

Para adicionar margens superior e inferior, definimos o primeiro e o terceiro valor como 40px.

```
button {
margin: 40px 15px 40px 15px;
width: 100px;
height: 100px;
border-radius: 15px;
background-color: red;
border: solid 5px dimgray;
}
```

A propriedade BORDER-RADIUS com quatro valores arredonda os cantos no sentido horário a partir do canto superior esquerdo.

```
img{
border-radius: 30px 30px 30px 30px;
width: 250px;
height: 300px;
}
```

Podemos arredondar apenas o canto superior esquerdo alterando o primeiro valor para 50px e deixando o restante como 0.

```
img{
border-radius: 50px 0 0 0 ;
width: 250px; }
```

Para levar BODER-RADIUS extremo, podemos definir um canto com o mesmo valor da altura ou largura, como 100px neste caso.

```
p{
border-radius: 0 0 0 100px;
width: 100px;
height: 100px;
}
```

Podemos usar BORDER-RADIUS para dar um toque único às imagens fazendo coisas como arredondar os cantos superior direito e inferior para 20px.

```
img{
border-radius: 0 20px 0 20px ;
width: 75px;
height: 75px;
}
```

A Cascata - Cascading

Descrição
A escolha do browser de qual regra 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 atribuído.
São levados em consideração 3 fatores:

- A origem do estilo;
- A especificidade;
- A importância;

Origem do Estilo

- tag Inline é mais forte que a tag style e a tag link do css. As tags style e link vai prevalecer quem estiver com a mesma configuração e na ultima linha.

Especificidade

Descrição
É um cálculo matemático, onde cada tipo de seletor e origem do estilo possuem valores a serem considerados.

Os mais fracos são universal selector, combinators e negation pseudo-class, com o valor de 0.
Em seguida, com valor de 1, são os element type selector e pseudo-elements.

Também temos os classes e attribute selectors, com valor de 10, ou seja, são mais fortes que os anteriores.

O segundo mais forte, ID selector, com um valor de 100, vence todos os selectors anteriores.

Por fim, temos o inline, com o valor 1000, quaisquer desses selectors anteriormente citado

Por fim, é o cálculo matemático, onde cada tipo de seletor e origem do estilo, possuem valores a serem considerados. Como mostra abaixo.

- 0. Universal selector, combinators e nagation pseudo-class (:not())
- 1. Element type selector e pseudo-elements (::before, ::after)
- 10. classes e atribute selector ([type=”radio”])
- 100. ID selector
- 1000. Inline

Funções

Descrição
Um tipo de valor existente no CSS, é estruturado com um nome seguido de abre e fecha parênteses.

Recebe um argumento, que são seus valores.
Em programação, funções são reconhecidas por causar um reaproveitamento de código.
Exemplos de funções do CSS:

rgb() / hsl() / url() / calc() /

Dentro dos parêntesis são passados argumentos

Função TRANSFORM

A propriedade transform aceita funções como valores. Para adicionar uma função que gire uma imagem, codificamos rotate() e especificamos por quantos graus a imagem vai girar.

Para girar a imagem no sentido horário, adicionamos um valor numérico seguido de DEG.

Para girar a imagem no sentido anti-horário, adicionamos um sinal de menos (-) na frente do valor numérico seguido de DEG.

Podemos virar as imagens em qualquer ângulo que quisermos.

```
Img {

Transform: rotate(90deg); - girar 90 graus no sentido horário

}
```

```
Img {

Transform: rotate(-90deg); - girar 90 graus no sentido anti-horário

}
```

Podemos usar essa propriedade e funcionar com qualquer elemento, não apenas imagens. Por exemplo, podemos definir o elemento button.

```
button{

transform: rotate(90deg);

}
```

Função FILTER

Vamos mergulhar em mudanças CSS mais visuais, como desfoque. Para fazer grandes alterações visuais nos elementos, usamos a propriedade FILTER.

FILTER pode usar algumas funções diferentes conforme seu valor. Vamos ver algumas delas.

- Blur() – A função blur torna a imagem embaçada. Para usá-lo, adicionamos um número de valor em pixels dentro dos parênteses, como 2px. Um número alto como 10px, torna o elemento muito embaçado, pois o número diz quantos pixels devem se misturar. Como outras funções, o Blur() funciona em muitos elementos diferentes. Por exemplo em títulos como H2, P, STRONG, etc.

```
Img {

Filter: blur(2px);

}
```

- Grayscale() – Usamos o valor em porcentagem %. Tons de grayscale transforma imagens em diferentes tons de cinza. Defini-lo como 0% significa que não afeta a imagem, com um valor de 60% podemos ver seu efeito de cinza nas imagens. Definir o valor para 100% torna a imagem completamente cinza.

```
Img {

Filter: grayscale(60%);

}
```

- Opacity() – Essa função altera a visibilidade dos elementos. Quanto menor a opacidade, menos visível a imagem se torna. Também usamos porcentagem % como valor. Opacidade significa não transparente, configurá-lo 100% o torna visível.

```
Img {

Filter: opacity(60%);

}
```

- Brightness() – Altera o brilho de uma imagem, onde 100% é o nível padrão da imagem. Quanto menor o brilho, mais escura a imagem fica. Podemos vêlo em ação com o brilho definido para 20%.

```
Img {

Filter: brightness(60%);

}
```

- Saturate() – Ajusta a vívida das cores. Torna a imagem menos colorida quando codificando para 50%.

```
Img {
Filter: saturate(50%);
}
```

- Contrast() – Ajusta a diferença entre as partes claras e escuras de uma imagem. Quanto menor a porcentagem, mais cinza a imagem fica.

```
Img {
Filter: contrast(50%);
}
```

* Podemos adicionar várias funções para FILTER codificando-as uma após a outra.

*Para efeitos exagerados, podemos definir brilho, saturação e contraste para valores acima de 100%.

Personalizando Gradientes

Para criamos um gradiente colorido como plano de fundo começamos com a propriedade background e configuramos com a função linear-gradient().

Para alterar gradualmente o plano de fundo de cima para baixo, coloque as cores dentro dos parênteses separando-as por virgula (,).

Para girar um gradiente em 90 graus, ou da esquerda para a direita, coloque 90deg antes das cores.

A direção de um gradiente vai no sentido horário de 0 a 360.

Caso queira que o gradiente mude o sentido horário basta colocar o sinal de -.

```
body{
background: linear-gradient(-45deg, turquoise, teal);
}
```

Classes CSS

Elementos de extensão

Às vezes, queremos estilizar apenas uma pequena parte de uma página da web, como apenas algumas palavras. Vamos aprender como fazer isso usando a tag .

O elemento é usado para aplicar estilos a partes do texto.

Estilizamos partes do texto em uma página da web usando uma tag de abertura , seguida de texto e depois uma tag de fechamento .

```
<H2>Melhores times Paulistas</H2>
<ul>
<li>Santos é o melhor</li>
<li>São Paulo é furada</li>
<li>Curinthians só tem 1 Mundial</li>
<li>Palmeiras não tem <span style="color: #a8561e;">Mundial</span></li>
</ul>
```

Como com outros elementos, podemos usar o seletor de tags para estilizar todas as tags ao mesmo tempo.

No HTML

```
<H2>Melhores times Paulistas</H2>
<ul>
<li>Santos é o <span>melhor</span></li>
<li>São Paulo é <span>furada</span></li>
<li>Curinthians só tem 1 <span>Mundial</span></li>
<li>Palmeiras não tem <span>Mundial</span></li>
</ul>
```

NO CSS

```
span {
color: #0078e6;
}
```

Se houver muitos elementos diferentes em uma página de web, podemos usar seletores de ID ou CLASSES para aplicar estilos diferentes.

Podemos então adicionar estilos diferentes aos seletores de classe para criar páginas da web mais intuitivas.

No HTML

```
<H2>Melhores times Paulistas</H2>
<ul>
<li>Santos é o <span class="santos">melhor</span></li>
<li>São Paulo é <span class="sao-paulo">furada</span></li>
<li>Curinthians só tem 1 Mundial</li>
<li>Palmeiras não tem <span class="palmeiras">Mundial</span></li>
</ul>
```

No CSS

```
.santos{
color: #0078e6;
}
.sao-paulo{
color: #6D3ACF;
}
.palmeiras{
color: #A8561E;
}
```

Estilizando Grupos de Elementos

Vamos aprender como estilizar grupos de elementos em uma página da web com a ajuda de elementos .

No HTML

```
<div>
<h1>Qual o melhor time do mundo?</h1>
<h3>O melhor time do mundo com certeza é o Santos</h3>
</div>
```

Podemos estilizar o grupo de elementos que criamos, usando DIV como seletor de tags.

No CSS

```
div{
border: solid 3px lightgray;
text-align: center;
}
```

Podemos adicionar outras propriedades para estilizar o grupo de elementos. Por exemplo, definindo background-color como bisque.

```
div{
border: solid 3px lightgray;
text-align: center;

background-color: bisque;
}
```

Para melhorar o espaçamento dentro e fora do grupo de elementos, podemos adicionar margins e espaçamento.

```
div{
    border: solid 3px lightgray;
    text-align: center;
    background-color: bisque;
    margin: 20px;
    }
```

Elementos Pais e Filhos

Aprendemos que o HTML usa algo chamado de caixa. O fato é que tudo em HTML é considerado uma caixa dentro da outra.

Quando adicionamos um elemento <DIV> ao redor de outros elementos, colocamos uma caixa maior ao redor de outras caixas.

Como o elemento tem outro elemento aninhado dentro dele, nós chamamos de elemento pai. O elemento <p> dentro dele é chamado de elemento filho.

Quando adicionamos mais paragrafos dentro do elemento , teremos 3 elementos filhos dentro do elemento pai .

No HTML

```
<div>
        <p>Santos o Melhor time do Mundo</p>
        <p>São Paulo é fraquinho</p>
        <p>Palmeiras não tem mundial</p>
    </div>
```

Quando estilizamos um elemento pai, existem algumas propriedades que afetam apenas o pai, como border, padding, margin.

No CSS

```
div{
    border: solid 3px lightgray;
}
```

Mas outras propriedades, como color, são transferidas para os elementos filhos de seu pai.

No CSS:

```
div{
border: solid 3px lightgray;
font-wight: bold;
}
```

Quando um elemento filho recebe uma propriedade de seu pai, chamamos isso de herança.

Não há uma regra rígida para quais propriedades são herdadas.

Regra !important

Descrição
sintaxe: !important

São raras as ocasiões nas quais se devem usar um iportant, pois é em geral uma má prática, visto que quebra o fluxo natural da cascata e pode acabar te atrapalhando caso você a deixe em algum lugar no seu código.

Portanto evite ao máximo seu uso.

- Cuidado, evite o uso
- Não é considerado uma boa prática
- Quebra o fluxo natural da cascata

At rules

Descrição
São regras relacionadas ao comportamento do CSS, começa com um sinal de @ seguido do identificador e do valor.

São as seguintes:

@import serve para incluir um CSS externo.

@import “http://local.com/style.css”; ou @import url(“http://local.com/style.css”);

@media são regras condicionais para vários dispositivos.

```
@media (min-width: 500px) {
Regras aqui
}
```

@font-face é para colocar fontes externas.

```
@font-face {
Regras aqui
}
```

@keyframes serve para as animations do CSS.

```
@keyframes nameofranimation{
Regras aqui
}
```

Shorthand

Descrição
É basicamente a ideia de junção de propriedades, para que fiquem de forma resumida e legível.

- Junção de propriedade
- Mais resumido
- Mais legível

Abaixo um exemplo de propriedades com e sem o shorthand:

```
{
/* 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;

/* font properties */
font-style: italic;
font-weight: bold;
font-size: .8em;
line-height: 1.2;
font-family: Arial, sans-serif;

/* font shorthand */
font: bold italic .8em/1.2 Arial, sans-serif;
}
```

Algumas das características do shorthand:

Não irá considerar propriedades anteriores, ou seja, caso faça um shorthand, apenas ele será considerado, 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, poderemos encontrar problemas.

Propriedades que aceitam Shorthand

Animation, background, border, border-botton, border-color, border-left, border-radius, border-rigth, border-style, border-top, border-width, column-rule, columns, flex, flex-flow, font, grid, grid-area, grid-column, grid-row, grid-template, list-style, margin, offset, outline, overflow, padding, place-content, place-items, place-self, text-decoration, transition

DevTools

Descrição
Uma das ferramentas mais importantes para o desenvolvedor CSS é o DevTools (do inglês, Ferramentas de Desenvolvedor), é recomendado que você estude e faça uso da mesma, pois será muito utilizada.

Abra o navegador e aperto o F12

Cuidados com a escrita

Descrição
É importante prestar atenção à sua escrita do CSS, identar seu código para facilitar a leitura, e mais importante, manter tudo organizado e funcionando!

Vendor prefixes

Descrição
São coisas que permitem que browsers adicionem features a fim de colocar em uso alguma novidade que vemos no CSS.

Exemplos:

```
p {
-webkit-background-clip: text; /*Chrome, Safari, iOS e Android*/
-moz-background-clip: text; /* Mozilla (Firefox) */
-ms-background-clip: text; /* Internet Explorer ou Edge*/
-o-background-clip: text; /* Opera */

}
```

Você também pode consultar se a feature pode ser utilizada através dos sites:

https://ireade.github.io/which-vendor-prefix

https://caniuse.com