



escola
britânica de
artes criativas
& tecnologia

Desenvolvedor Full Stack Python

CSS

Conteúdo do módulo

- Sintaxe
- Seletores
- Box Model
- Box Sizing
- A propriedade display
- A propriedade position
- Trabalhando com cores
- Estilizando textos

Sintaxe

Conteúdo da aula

Sintaxe

- Sintaxe do CSS
- Formas de adicionar estilo ao HTML

Sintaxe do CSS

No CSS trabalhando com **seletores**, **propriedades** e **seus valores**.

Uma sentença CSS é iniciada com { e concluída com }

```
Seletor {  
propriedade: valor;  
}
```

Exemplo de CSS funcional:

```
h1 {  
font-size: 24px;  
}
```

Formas de adicionar estilo ao HTML

Podemos adicionar o CSS a nossa página HTML de três formas:

1. Diretamente nas tags, através do atributo style—CSS inline;
2. Na head, escrevendo o CSS dentro da tagstyle;
3. Importando o arquivo CSS.

Exemplo 1:

```
<h1 style="font-size: 24px">Olá CSS</h1>
```

Formas de adicionar estilo ao HTML

Exemplos

Exemplo 1 – CSS Inline:

```
<h1 style="font-size: 24px">Olá CSS</h1>
```

Exemplo 2 – Na head:

```
<head>  
<style>  
h1 {  
font-size: 22px;  
}  
</style>  
</head>
```

Exemplo 3 – Importando o arquivo:

```
<head>  
<link rel="stylesheet" href="meu_arquivo.css" />  
</head>
```

Seletores

Conteúdo da aula

Seletores

- Tags, classes e IDs
- Selecionando elementos
- Pseudo-seletores

Tags, classes e ID

No CSS podemos estilizar os elementos através do nome da tag, do nome da classe ou pelo seu ID.

Para adicionarmos uma classe no elemento HTML usamos o atributo `class`.

```
<h1 class="title">Olá CSS</h1>
```

Selecionando elementos

Para selecionar os elementos que queremos estilizar, podemos usar os seletores, são eles:

Nome da tag, exemplo: `h1 { ... }`

ID da tag, representado por #: `#meuld`

Classe da tag, usamos o ponto: `.minhaClasse { }`

Podemos selecionar mais de um elemento por vez:

`.title, .subtitle { ... }`

Podemos navegar entre os elementos:

`.header .title { ... }`

Nesse exemplo estamos estilizando o elemento `.title` que está dentro do elemento `.header`

Pseudo-classes

Podemos estilizar além dos elementos, seu estado. Fazemos isso com as pseudo-classes, por exemplo: quando passamos o mouse sobre um link e ele muda de cor, essa mudança é feita através da **estilização de pseudo-classes**.

`seletor:estado { ... }`

Exemplo, quando passar o mouse sobre as tags H1 o texto ficará vermelho:

`h1:hover { color: red; }`

Lista das principais pseudo-classes

:first-child— para selecionar a primeira ocorrência do elemento

:last-child— para selecionar a última ocorrência do elemento

:hover— para estilizar o elemento quando passarmos o mouse por cima

:focus— para estilizar um elemento quando o foco estiver nele

:checked—para estilizar inputs do tipo checkbox e radio, quando estes estiverem marcados

:required—para estilizar inputs que sejam obrigatórios

:optional—para estilizar inputs que sejam opcionais

Alguns exemplos de uso de pseudo-classes

Selecionando o primeiro item de uma lista:

```
ulli:first-child{...}
```

Selecionado um campo quando ele está com o foco:

```
input:focus{...}
```

Selecionando um link quando passamos o mouse:

```
a:hover {...}
```

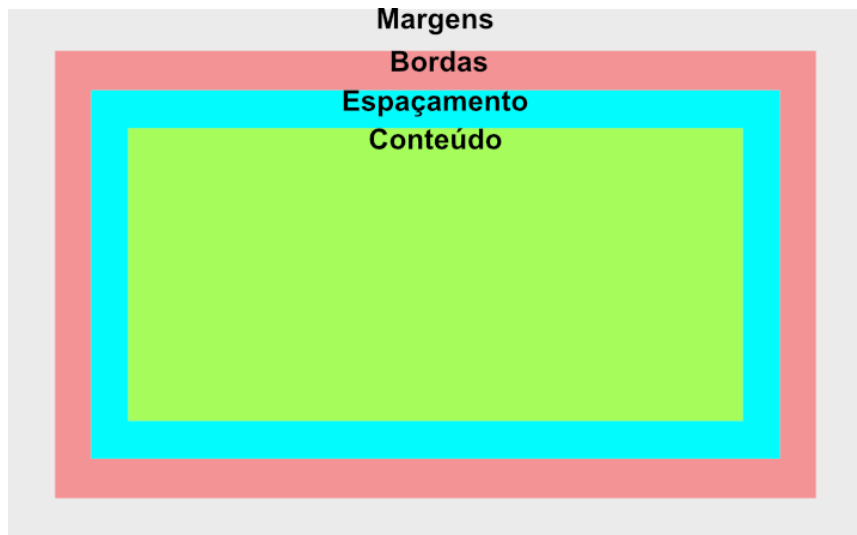
Box Model

Conteúdo da aula

- Sobre o box model
- A propriedade margin
- A propriedade border
- A propriedade padding
- Identificando as propriedades através do navegador
- Sobre a dimensão dos elementos

Sobre o Box Model

Box model é o termo que usamos para nos referir ao conjunto: **conteúdo**, **espaçamento**, **bordas** e **margens**.



A propriedade **Margin**

Com a propriedade **margin**, aplicamos **margem aos elementos**, podemos adicionar margem a cada lado do elemento, usando uma propriedade específica:

margin-left (ou right, top, bottom)

Ou aplicar a margem em apenas uma linha de CSS:

margin: top rightbottomleft;

Exemplos:

```
section{  
    margin-top: 16px;  
    margin-bottom: 16px;  
}
```

```
section{  
    margin: 16px 0 16px;  
}
```

A propriedade **Border**

Com a propriedade **border**, aplicamos uma borda aos elementos, a forma mais simples de adicionar uma borda é através da propriedade border:

border: tamanho estilo cor;

Podemos configurar o tamanho, estilo e cor da borda de forma individual, respectivamente:

border-width: 1px;

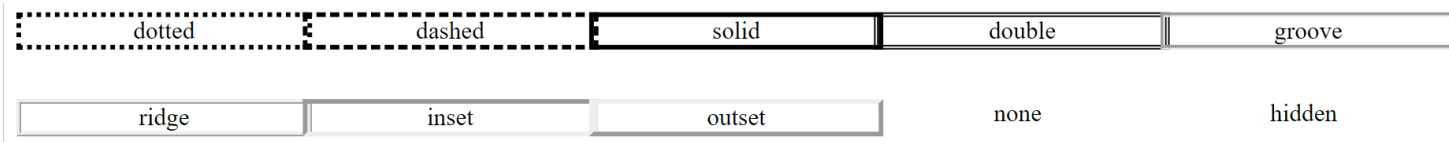
border-style: solid;

border-color: black;

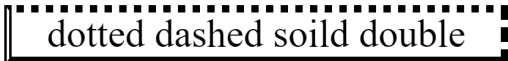
A propriedade **Border-style**

Podemos trabalhar com o estilos de borda:

dotted, dashed, solid, double, groove, ridge, inset, outset, nonee hidden.



É possível combinar estilos:



A propriedade **Padding**

Com a propriedade **padding**, aplicamos espaçamento interno aos elementos, podemos adicionar espaçamento a cada lado do elemento, usando uma propriedade específica:

Padding-left (ou right, top, bottom)

Ou aplicar o espaçamento em apenas uma linha de CSS:

padding: top right bottom left;

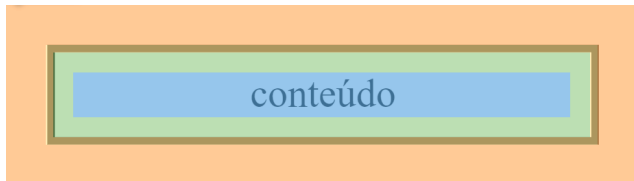
Exemplos:

```
section{  
  padding-top: 16px;  
  padding-bottom: 16px;  
}
```

```
section{  
  padding: 16px 0 16px;  
}
```

Identificando as propriedades através do navegador

Usando o inspetor de elementos do **DevTools**, conseguimos visualizar de forma simples o box model aplicado a um elemento:



Laranja: margens

Preto: bordas

Verde: espaçamento

Azul: o conteúdo que está dentro da tag

Sobre as dimensões dos elementos

O box model é importante para nos ajudar a calcular a dimensão de um elemento, considere o CSS abaixo:

```
div{  
  width: 200px;  
  padding: 8px;  
  margin: 16px;  
  border: 4px solidblack;  
}
```

A largura do elemento div não será 200px, é necessário considerar as propriedades padding, margin e border neste valor. O que resultará em uma div com 224px de largura, sendo:

200 (largura inicial) + 8 (padding right) + 8 (padding left)
+ 4 (border right) + 4 (border left)

A margem por ter um efeito externo ao elemento não afeta sua dimensão.

Box Sizing

Conteúdo da aula

Box Sizing

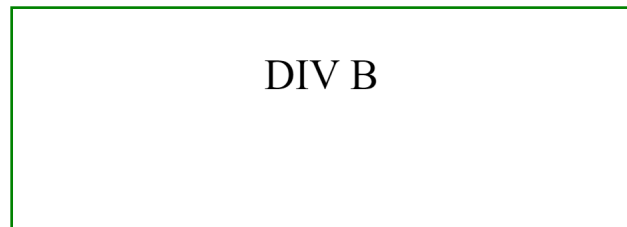
- A propriedade box-sizing
- O seletor *

A propriedade box-sizing

A propriedade `box-sizing` nos permite incluir ou não o espaçamento e as bordas à dimensão total do elemento.

Para exemplificar temos a div A e div B, onde a única diferença está na presença de um espaçamento na div B:

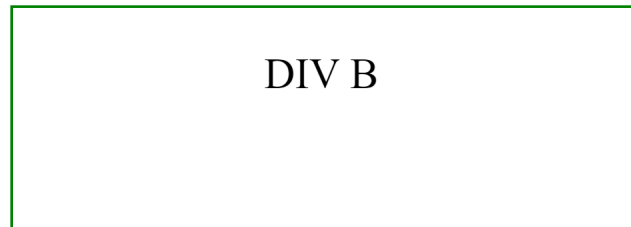
```
.divA{  
width: 200px;  
height: 50px;  
border: 1px solidblue;  
}  
  
.divB{  
width: 200px;  
height: 50px;  
padding: 16px;  
border: 1px solidgreen;  
}
```



A propriedade box-sizing

Apesar de atribuirmos a largura e a altura dos elementos, com width: 200px, height: 50px, a diferença entre a div A e div B é muito grande, isso por causa do efeito do espaçamento na composição do box-model.

Adicionando a propriedade **box-sizing** com o valor **border-box** podemos informar que o espaçamento deve ser considerado nas medidas informadas pela largura e altura.



A propriedade box-sizing

Os valores possíveis para a propriedade **box-sizing** são:

content-box: é o valor padrão da propriedade, a largura e altura, incluindo o conteúdo, sem considerar espaçamento e bordas

border-box: a largura e altura incluindo o conteúdo, espaçamento e bordas

initial: valor padrão da propriedade

inherit: o valor propriedade do elemento-pai

O seletor *

A propriedade `box-sizing` é uma das propriedades que iremos querer atribuir para todas as tags, podemos fazer isso com o seletor `*`:

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

Display e Flexbox

Conteúdo da aula

Box Sizing

- A propriedade display
- Flexbox

A propriedade display

Com a propriedade **display** informamos ao navegador como e se um elemento deve ser exibido.

Os principais valores que iremos explorar são: **none**, **block** e **inline**.

Display: none

Com o valor none o elemento não será exibido.

Display: block

O elemento será exibido como um bloco, não tendo nenhum outro elemento a seu lado, como se houvesse uma quebra de linha antes e depois do elemento.

Display: inline

O elemento será exibido na mesma linha, como se fosse uma tag, span, por exemplo as listas possuem seus itens separados por linhas, com o display inline sendo aplicado na tag li veremos os itens lado a lado.

Flexbox

O **Flexbox** não é uma propriedade CSS e sim um **módulo**.

É composto por algumas propriedades que exploraremos, como: **flex-direction**, **flex-wrap**, **align-items** e **align-content**.

Para começarmos a trabalhar com o Flexbox temos um requisito que é a construção de um container que possua o **display configurado com o valor flex**.

Flexbox Container

Para construir o container precisamos apenas de uma **div** com a propriedade **display** possuindo o valor **flex**.

É através do container que iremos controlar o layout.

Vamos considerar o layout:



Flexbox – Justify Content

Com a propriedade `justify-content` informaremos ao navegador qual é a forma que os elementos dentro do container devem ser distribuídos.

Os dois principais valores são **Flex-Start** e **Flex-End**, mas também temos outros como o `center`, `space-around` e `space-between`.

Flexbox – Justify Content

Com `justify-content: flex-start`, informamos que os **elementos** devem estar concentrados ao lado esquerdo do container, onde ele se inicia:



Flexbox – Justify Content

Com `justify-content: flex-end`, informamos que os elementos devem estar concentrados ao lado direito do container, onde é seu termino:



Flexbox – Justify Content

Com `justify-content: center`, informamos que os elementos devem ser centralizados:



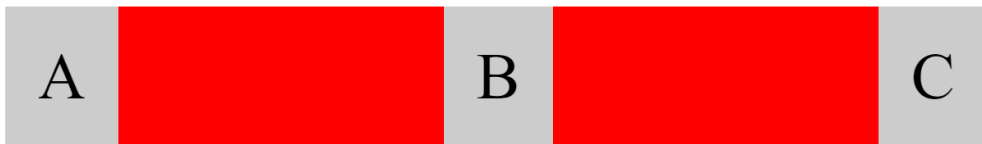
Flexbox – Justify Content

Com `justify-content: space-around`, informamos que os elementos devem ser separados e o espaço vago do container deverá ser distribuído de forma igual ao contorno dos elementos:



Flexbox – Justify Content

Com `justify-content: space-between`, informamos que os elementos devem ser separados e o espaço vago do container deverá ser distribuído de forma igual porém sem um contorno entre os elementos:



Flexbox – Align Items

O Flexbox também nos ajuda no **alinhamento de itens na vertical**.

Trouxe uma abordagem muito mais simples do que as existentes até a introdução do módulo.

Consideraremos um container com os elementos H1, H3 e H6, por padrão seriam exibidos assim:

Texto em H1 Texto em H3 Texto em H6

Flexbox – Align Items

Para alinharmos esses elementos considerando o canto inferior usamos a propriedade **align-items** com o valor **baseline**:

Texto em H1 Texto em H3 Texto em H6

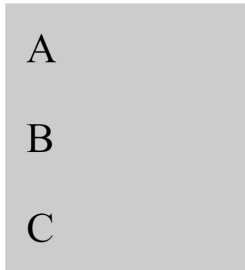
Para centralizarmos bastaria usar o valor **center**:

Texto em H1 Texto em H3 Texto em H6

Flexbox – Flex Direction

Com a propriedade **flex-direction** podemos alterar a forma que os elementos dentro do container irão aparecer.

O valor padrão é **row** mas podemos mudar para **column** para que os elementos fiquem um embaixo do outro, se organizando em formato de colunas.



Note que com o **display: flex** os elementos buscam ocupar toda a área disponível.

Position

Conteúdo da aula

Position

- A propriedade position
- Posição fixa
- Posição relativa
- Posição absoluta

A propriedade Position

Para posicionar os elementos no CSS iremos usar a propriedade **position**, através dela diremos de qual forma queremos posicionar o elemento, se é de forma fixa, relativa ou absoluta.

Além dessa propriedade devemos configurar as propriedades: **top, right, left e bottom**.

Posição fixa

A posição fixa irá **posicionar o elemento na mesma posição** independente do tamanho da tela e da rolagem da página.

Exemplo:

```
.alert{  
position: fixed;  
bottom: 0;  
right: 0;  
}
```

O elemento está sempre no canto inferior direito da tela, independente do tamanho da tela.

Posição relativa

Com a posição relativa iremos **posicionar um elemento de forma relativa a sua posição inicial**.

Esse tipo de abordagem é bastante usada quando precisamos mover um item sem que isso desloca os elementos que estão a sua volta.

Posição absoluta

Com a posição absoluta, o elemento **será posicionado de forma absoluta ao elemento que ele pertence**, desde que esse elemento-pai possua a propriedade position: **relative**.

Apesar de sua posição estática o elemento com a posição absoluta não será visível quando houver rolagem de página.

Cores

Conteúdo da aula

- Formatos de cores aceitos
- Aplicação de cores em textos
- Aplicação de cor de fundo em elementos

Formatos de cores aceitos

No CSS podemos trabalhar com os formatos de cores:

hexadecimal, hexadecimal com transparência, RGB, RGBA, HSL e HSLA.

Aplicação de cores em texto

Para aplicar uma cor a um texto fazemos o uso da propriedade `color`:

```
p {  
  color: #333333;  
}
```

Aplicação de cor de fundo em elementos

Para aplicar uma cor de fundo ao elemento usamos a propriedade `background-color` ou apenas `background`.

```
.header {  
  background-color: red;  
}
```

Textos

Conteúdo da aula

Textos

- Alterando a fonte de texto
- Alinhando textos
- Aplicando estilos à textos

Alterando a fonte de texto

Podemos alterar a fonte da nossa página utilizando a propriedade **font-family**:

```
p {  
  font-family: "Segoe UI", arial, sans-serif;  
}
```

Colocamos o nome da fonte entre aspas quando contém espaço.

Para alterar o tamanho da fonte, usamos a propriedade **font-size**:

```
p {  
  font-size: 16px;  
}
```


Alinhando textos

Para fazer o alinhamento de texto usamos a propriedade `text-align`, por padrão seu valor é left.

```
p {  
  text-align: left OU center OU justify OU right;  
}
```

Aplicando estilos à textos

Podemos aplicar alguns estilos aos textos usando CSS, como itálico, negrito (mudança de peso da fonte), linhas ao texto, entre outros.

Texto em itálico: para aplicar o itálico a um texto usamos a propriedade **font-style** com o valor **italic**.

Para mudar o peso de uma fonte: podemos mudar o peso de uma fonte usando a propriedade **font-weight**, os principais valores que temos para essa propriedade são: **lighter**, **normal**, **bold**, **bolder**, e de 100 ... 900 (mais leve até o mais pesado, acrescentando-se 100).

Aplicando estilos à textos

Adicionando linhas ao texto: podemos adicionar algumas linhas para efeito aos textos, como o underline, usamos a propriedade **text-decoration**, onde os principais valores são **underline**, **line-through** e **overline**.

Manipular o tipo de caixa dos textos: podemos mudar a forma que o texto se apresenta, se é em CAIXA ALTA ou caixa baixa, ou Capitalizado, fazemos isso através da propriedade: **text-transform** com os respectivos valores **uppercase**, **lowercase** e **capitalize**.