DESENVOLVIMENTO WEB

Aula 4:

“CSS: Outros Conceitos”

Prof. Nathan Cirillo
nathan.silva@docente.unip.br

Prof. Sergio Soares
sergio.soares@docente.unip.br

Universidade Paulista UNIP
Ciência da Computação

Trabalhando com Ícones

- Para inserir ícones na página é necessário usar uma biblioteca de ícones;
- Acrescente o nome da classe do ícone a qualquer elemento HTML;
- Os ícones são imagens vetoriais que podem ser customizados;
- A customização dos ícones é feita através do uso de CSS.

Importando as Bibliotecas

Font Awesome (*https://fontawesome.com/*):

```
<head>
        <script src="https://kit.fontawesome.com/<seucodigo>.js"></script>
</head>
```

Se **cadastre no site** para acessar o seu código!

Bootstrap Icons (*https://getbootstrap.com/docs/3.3/components/*):

```
<head>
<link rel="stylesheet" href=" https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css">
</head>
```

Google Icons (*https://material.io/resources/icons/?style=baseline*):

```
<head>
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet" >
</head>
```

Adicionando os Ícones

Veja que o CSS pode **estilizar os ícones!**

Para mais ícones, consulte:
https://fontawesome.com/icons

Font Awesome

```
<i class="fas fa-thumbs-up" style="font-size: 24px; color:red;"></i>
<i class="fas fa-thumbs-up" style="font-size: 36px; color:blue;"></i>
<i class="fas fa-thumbs-up" style="font-size: 48px; color:green;"></i>
<i class="fas fa-thumbs-up" style="font-size: 60px; color:navy;"></i>
```

Para mais ícones, consulte:
https://getbootstrap.com/docs/3.3/components/

Bootstrap

```
<i class="glyphicon glyphicon-heart" style="font-size: 24px; color:pink;"></i>
<i class="glyphicon glyphicon-heart" style="font-size: 36px; color:yellow;"></i>
<i class="glyphicon glyphicon-heart" style="font-size: 48px; color:purple;"></i>
<i class="glyphicon glyphicon-heart" style="font-size: 60px; color:red;"></i>
```

Para mais ícones, consulte:
https://material.io/resources/icons/

Google

```
<i class="material-icons" style="font-size:24px; color:navy;">email</i>
<i class="material-icons" style="font-size:36px; color:brown;">computer</i>
<i class="material-icons" style="font-size:48px; color:PaleGreen">flight</i>
<i class="material-icons" style="font-size:60px; color:OrangeRed">time_to_leave</i>
```

Estilizando Links

- Os links são estilizados usando qualquer propriedade CSS: *color, font-family, background,etc*;
- Possuem estados que variam com a ação do usuário: ***a:link***, ***a:visited***, ***a:hover***, ***a:active***;
- Existem regras ao estilizar links: *a:hover* ***deve vir depois*** *de a:link e a:visited*;
- E o item *a:active* ***deve vir depois*** *de a:hover*.

a:link – não visitado (normal)

a:visited – usuário já visitou

a:hover – mouse em cima

a:active – quando é clicado

Veja um Exemplo

```
a:link {
        font-size:30px;
        text-decoration:none;
        color:red;
        background-color:yellow;
    }
```

Google

```
a:hover{
        font-size:30px;
        text-decoration:underline;
        color:navy;
        background-color:lightblue;
    }
```

Google

Google

```
a:visited{
        font-size:30px;
        text-decoration:none;
        color:darkgrey;
        background-color:grey;
    }
```

```
a:active{
        font-size:30px;
        text-decoration:underline;
        color:darkgreen;
        background-color:lightgreen;
    }
```

Listas Ordenadas e Não Ordenadas

Modificando os Tipos de Marcadores:

```
<p>Exemplo de Lista Não Ordenada:</p>
<ul class="a">
  <li>Arroz</li>
  <li>Feijão</li>
  <li>Batata</li>
</ul>
```

```
ul.a
{
  list-style-type: circle;
}
```

Opções Disponíveis:
Circle, Disc, Square, None

Exemplo de Lista Não Ordenada:

- Arroz
- Feijão
- Batata

```
<p>Exemplo de Lista Ordenada:</p>
<ol class="b">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>
```

```
ol.b
{
  list-style-type: upper-roman;
}
```

Opções Disponíveis:
Decimal, Lower-roman, Upper-roman, Lower-latin,Upper-latin

Exemplo de Lista Ordenada:

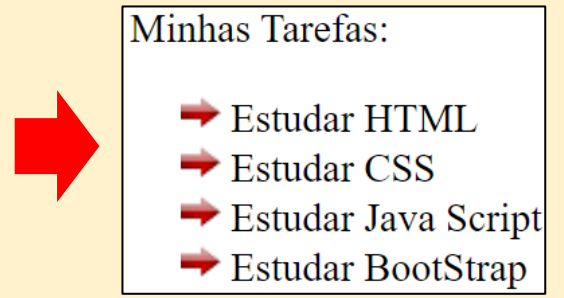
I. Coffee
II. Tea
III. Coca Cola

Listas Ordenadas e Não Ordenadas

Adicionando Imagem como Marcador:

```
<p>Minhas Tarefas:</p>
<ul>
   <li>Estudar HTML</li>
   <li>Estudar CSS</li>
   <li>Estudar Java Script</li>
   <li>Estudar BootStrap</li>
</ul>
```

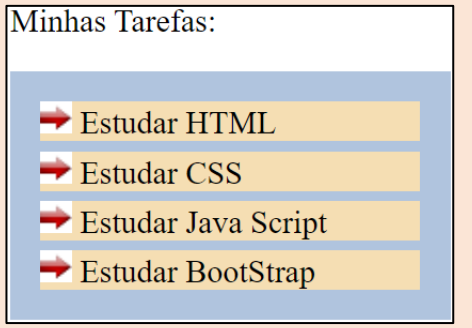
```
ul {
      list-style-image: url('seta.png');
   }
```



Colorindo a Lista Apresentada:

```
ul {
      list-style-image: url('seta.png');
      background-color: #B0C4DE;
      padding: 10px;
      list-style-position: inside;
      width:200px;
   }
```

```
li {
      background-color: #F5DEB3;
      margin:5px;
   }
```

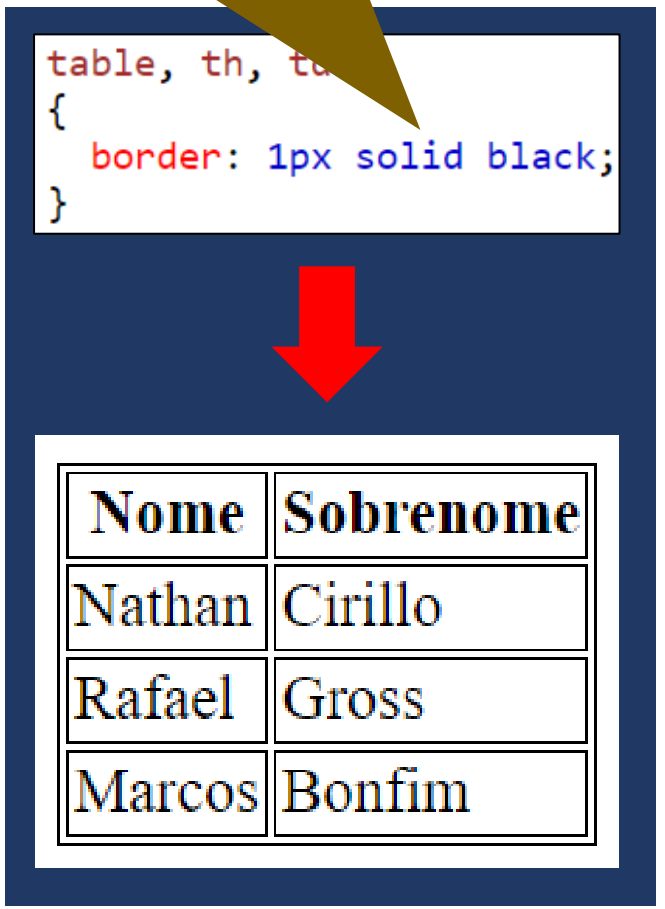


Trabalhando com Tabelas

Especificando as suas Bordas:

Para bordas simples, adicione:
border-collapse: collapse;

- As bordas são especificadas através da propriedade border do CSS;
- As bordas devem ser adicionadas aos elementos: *table*, *th e td*;
- A tabela terá bordas duplas, pois cada elemento possui bordas separadas.



Trabalhando com Tabelas

Definindo a sua Largura e a Altura

- A **largura e a altura de uma tabela** são definidos pelas propriedades ***width*** e ***height*** do CSS;

- Essas propriedades *também podem ser aplicadas* às suas respectivas células: ***<th>*** *e* ***<td>***.

```
table
{
    width:300px;
}
```

```
th
{
    height: 40px;
}
```

```
td
{
    width:50%;
    height:20px;
}
```

Nome	Sobrenome
Nathan	Cirillo
Rafael	Gross
Marcos	Bonfim

Trabalhando com Tabelas

Alinhamento Horizontal do Conteúdo:

- A propriedade ***text-align*** alinha o conteúdo (***left, right, center***);
- Ela pode ser aplicada nos elementos **<table>**, **<th>** ou **<td>**;
- O **<th>** por padrão é centralizado e o **<td>** alinhado à esquerda.

```
th {
        height: 40px;
        text-align: left;
    }
```

```
td {
        width: 50%;
        height:20px;
        text-align: center;
    }
```

Nome	Sobrenome
Nathan	Cirillo
Rafael	Gross

Trabalhando com Tabelas

Alinhamento Vertical do Conteúdo:

- A propriedade ***vertical-align*** alinha o conteúdo (***top, bottom, middle***);
- Ela pode ser aplicada nos elementos **<table>**, **<th>** ou **<td>**;
- Por padrão o **alinhamento da vertical** de uma tabela é no centro.

```
th {
        height: 40px;
        text-align: left;
        vertical-align:top;
    }
```

```
td {
        width: 50%;
        height:30px;
        text-align: center;
        vertical-align:bottom;
    }
```

Nome	Sobrenome
Nathan	Cirillo
Rafael	Gross

Trabalhando com Tabelas

Aumentando o Padding dos Elementos:

- O padding é o ***espaço entre a borda do elemento e o seu conteúdo***;
- Para controlá-lo utilize essa propriedade nos elementos: **<th>** ou **<td>**.

```
th{
    height: 40px;
    text-align:left;
    vertical-align:top;
    padding: 10px;
  }
```

```
td
{
    width:50%;
    height:30px;
    text-align:center;
    vertical-align:bottom;
    padding: 10px;
}
```

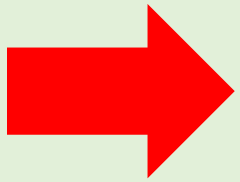
Nome	Sobrenome
Nathan	Cirillo
Rafael	Gross

Trabalhando com Tabelas

Utilizando o Seletor :hover em Linhas:

- Como nos links, *o hover serve para identificar a passagem do mouse*;
- Podemos usá-lo para destacar as linhas da tabela ao passar o mouse.

```
tr:hover {
        background-color:gray;
        }
```



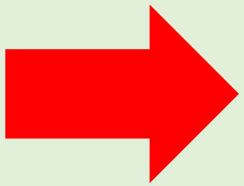
Nome	Sobrenome
Nathan	Cirillo
Rafael	Gross

Trabalhando com Tabelas

Aplicando o Seletor :nth-child em Linhas:

- O **:nth-child** permite criar ***tabelas zebradas com linhas coloridas***;
- Ele irá colorir somente as **linhas pares (*even*)** ou **ímpares (*odd*)**.

```
tr:nth-child(even) {background-color: #00FFFF;}
```



Nome	Sobrenome
Nathan	Cirillo
Rafael	Gross
Marcos	Bonfim

Trabalhando com Tabelas

Criando Tabelas Responsivas:

- Uma **barra de rolagem horizontal** é criada para mostrar o conteúdo;
- Um **container ( )** deve ser **usado ao redor** dessa tabela;
- O seu **atributo style** deve ter o seguinte **valor:** ***"overflow-x:auto"***.

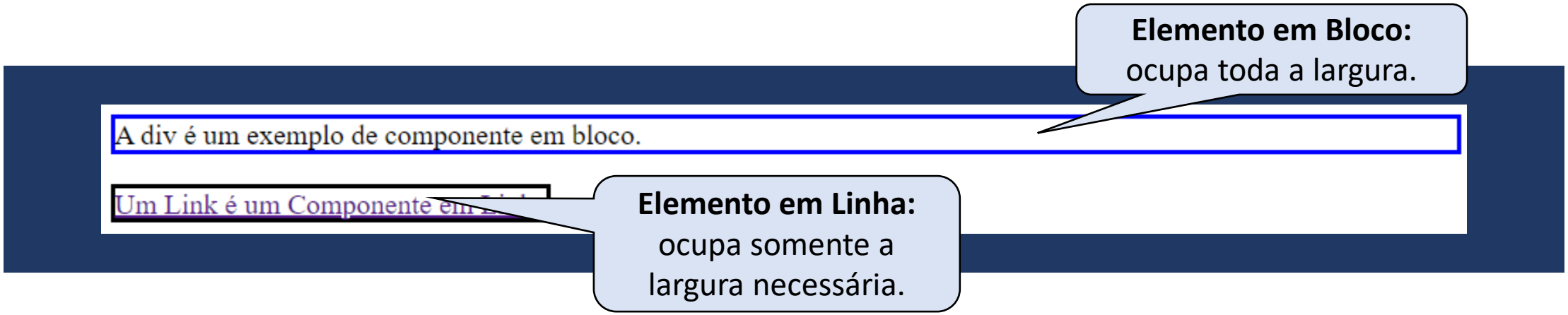
Nome	Sobrenome	Pontos	Pontos	Pontos	Pontos	Pontos
Jill	Smith	70	30	90	20	50
Joyce	Magna	90	20	60	50	50

```
<div style="overflow-x:auto;">
  <table>
    <tr>
      <th>Nome</th>
      <th>Sobrenome</th>
      <th>Pontos</th>
      <th>Pontos</th>
      <th>Pontos</th>
      <th>Pontos</th>
      <th>Pontos</th>
    </tr>
    <tr>
      <td>Jill</td>
      <td>Smith</td>
      <td>70</td>
      <td>30</td>
      <td>90</td>
      <td>20</td>
      <td>50</td>
    </tr>
    <tr>
      <td>Joyce</td>
      <td>Magna</td>
      <td>90</td>
      <td>20</td>
      <td>60</td>
      <td>50</td>
      <td>50</td>
    </tr>
  </table>
</div>
```

Sobre a Propriedade Display

- A propriedade display define se um elemento será apresentado e de que forma será a apresentação;
- Todos os elementos HTML possuem valores padrão para a propriedade display: ***block*** *e* ***inline***;
- Um ***elemento em bloco*** sempre **começa em uma nova linha** e **ocupa toda a largura da tela**;
- Um ***elemento em linha*** **não começa em uma nova linha** e **ocupa só a largura necessária**.

Veja a Diferença: *Block vs Inline*



Exemplos de elementos em bloco:

-
- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>

X

Exemplos de elementos em linha:

-
- <a>
- <img>

Ocultando Elementos HTML

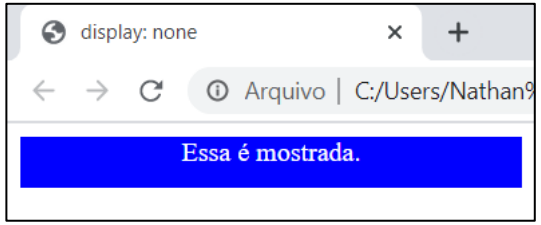
- A propriedade **display** pode ser usada para **ocultar elementos HTML**;
- Os elementos **não ocuparão mais espaço** na página web (*como não existisse*);
- Esse técnica é **muito usada em JavaScript** para esconder elementos.

```
<div class="divH"> Essa outra está oculta. </div>
<div class="divS"> Essa é mostrada. </div>
```

Outra Opção:
visibility:hidden
(ocupa espaço)

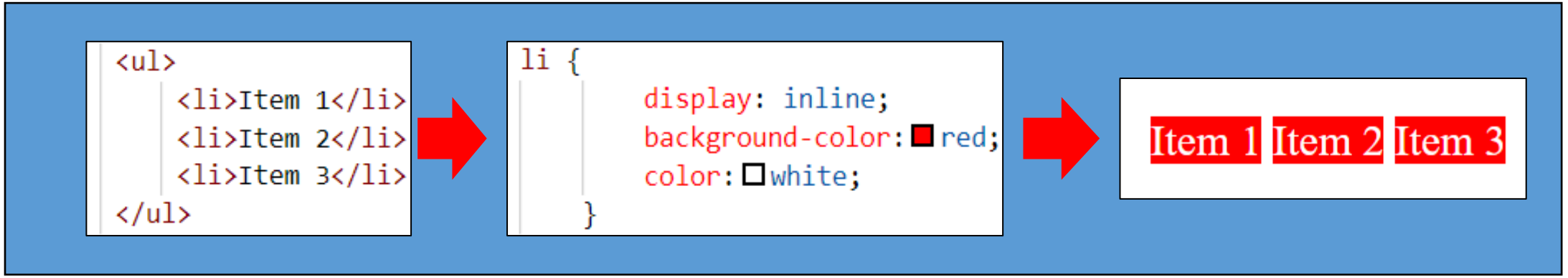
```
div.divH {
    display:none;
    background-color:yellow;
    text-align:center;
    color:brown
}
```

```
div.divS {
    background-color:blue;
    text-align:center;
    color:white;
    width: 300px;
    height: 30px;
}
```



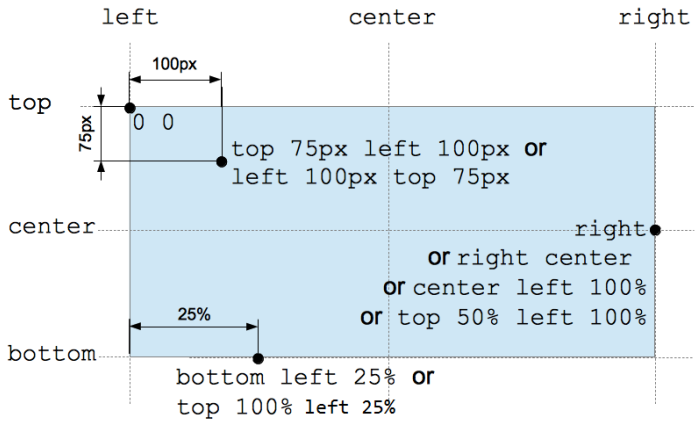
Mudando a Apresentação dos Elementos

- *Todos os elementos HTML* possuem **valores para a propriedade display: *inline ou block*;**
- Tais valores podem ser **sobrescritos** assim **um elemento em bloco é mostrado em linha ou vice-versa**;
- Esse recurso é muito útil quando é necessário **modificar a aparência** de um dado elemento.



Propriedade Position

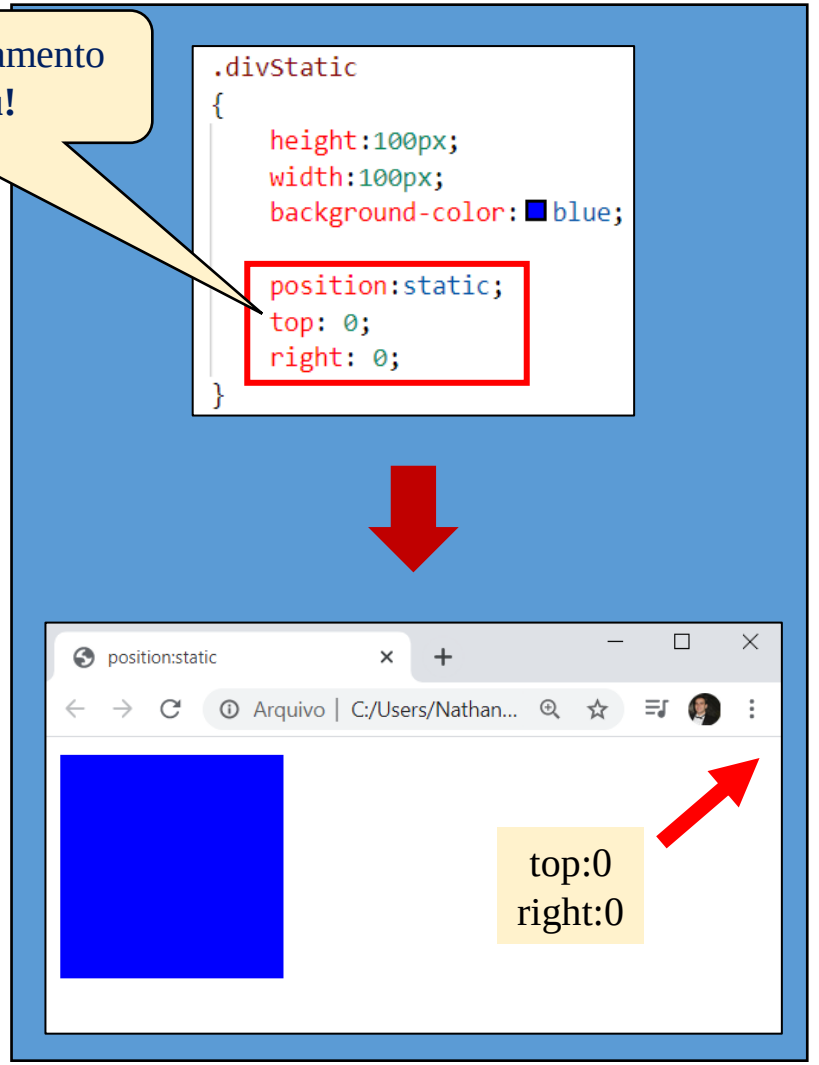
- A **propriedade position** define o **método de posicionamento** do elemento;
- Os seus valores de posicionamento são: *static*, *relative*, *fixed* e *absolute*;



- Ao especificar o position é possível **mover o elemento para onde quiser**: *top*, *bottom*, *left*, *right*;
- Se a *propriedade position e o seu valor não forem definidos*, o **posicionamento não irá funcionar**.

Position: static

- Ao adicionarmos um elemento HTML ele já é **estático por padrão** (*default*);
- Ele não é posicionado de modo especial, apenas **segue o fluxo da página**;
- Elementos estáticos **não são afetados** por: *top*, *bottom*, *left e right*.

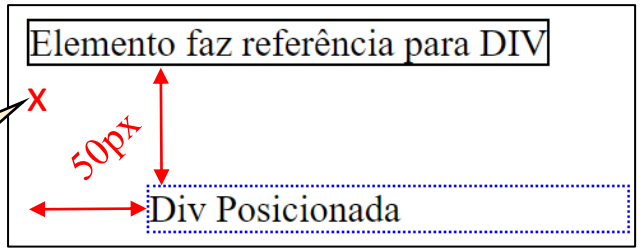


Position: relative

- Um **elemento relativo** é posicionado com base em sua **posição atual** (*elemento pai*);
- ***Left, right, top e bottom*** posicionará o elemento a partir de sua posição atual;
- Os espaços deixados pelo elemento **não são preenchidos** por outro conteúdo.

```
<span>Elemento faz referência para DIV</span>

<div id="divRelative">Div Posicionada</div>
```

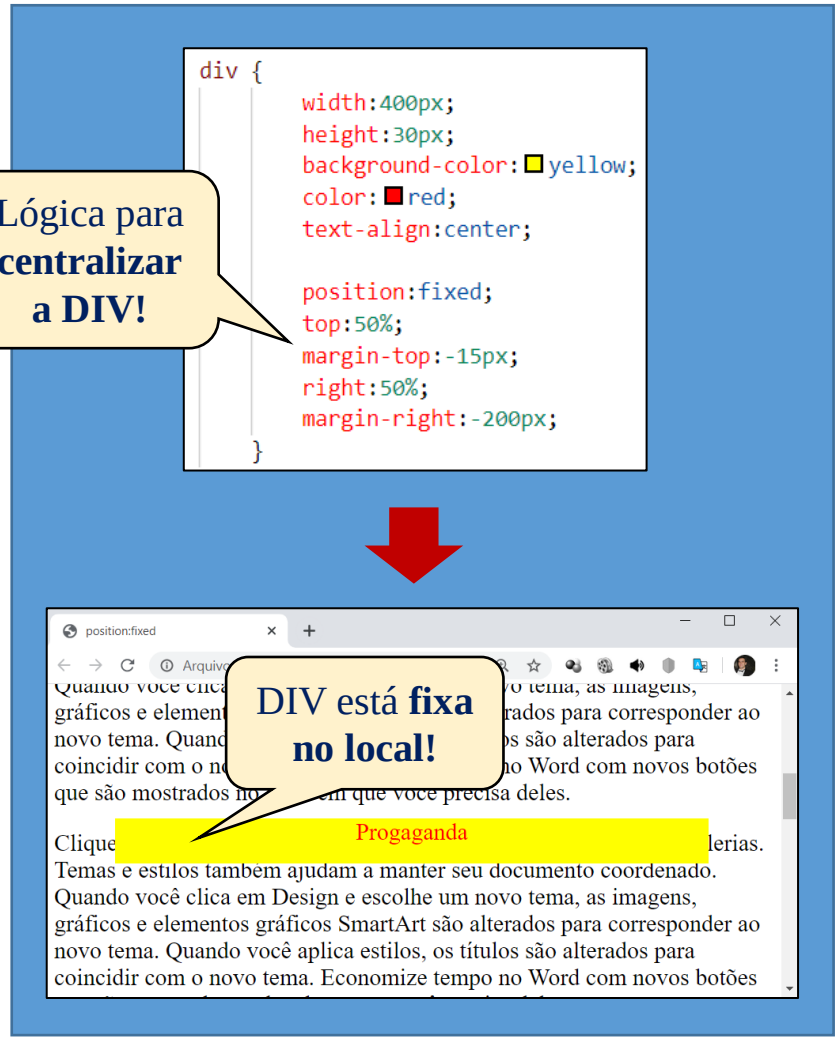
```
span {border: 1px solid black;}

#divRelative
{
    border: 1px dotted blue;
    width:200px;

    position:relative;
    left: 50px;
    top: 50px;

}
```



Posicionou o elemento a partir da **posição atual!**

Position: fixed

- Um elemento fixo **permanece sempre no mesmo local**;
- Mesmo **rolando a página (*scroll*)** ele fica fixo no local;
- Ele é posicionado em relação a **janela principal (*viewport*)**;
- O elemento fixo **não ocupa espaço onde está localizado** (*sobreposição*).



Position: absolute

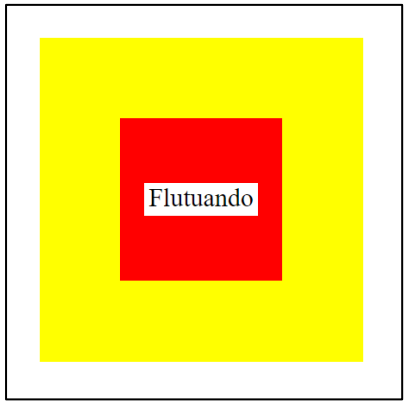
- O elemento é posicionado em **relação ao seu elemento pai**;
- Se **não houver um elemento pai,** *se movimentará pela página*;
- A *posição absoluta* **não ocupa espaço, "flutua"** pela página.

```html
<div id="divP"></div>

<div id="divS"><span id="spn">Flutuando</span></div>
```

```css
#divP {
        height:200px;
        width:200px;
        background-color: yellow;
        position:absolute;
        top: 20px;
        left:20px;
      }
```

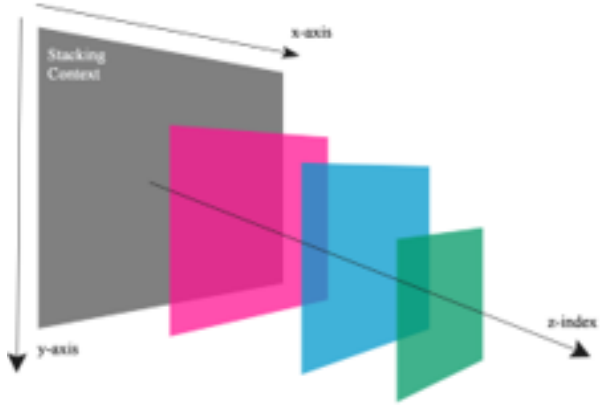
```css
#spn {
        position:absolute;
        background-color: white;
        width:70px;
        height:20px;
        text-align:center;
        top:40px;
        left:15px;
      }
```

```css
#divS {
        height:100px;
        width:100px;
        background-color: red;
        position:absolute;
        top:70px;
        left:70px;
}
```



Sobreposição de Elementos

- Ao posicionar elementos, **um pode acabar sobrepondo outro**;
- **O último elemento adicionado no HTML** será *mostrado em cima*;



- A **propriedade z-index** serve para especificar a **ordem da pilha de elementos** (*prioridade*);
- O elemento que possuir o **maior valor da propriedade z-index** sempre **ficará na frente**.

Sem o uso do Z-INDEX

```css
#camada1 {
        height:400px;
        width:400px;
        background-color: blue;
        position:absolute;
        left:30px;
        top:30px
    }
```

Maior camada:
400 x 400

```css
#camada2 {
        height:200px;
        width:200px;
        background-color: yellow;
        position:absolute;
        left:30px;
        top:30px;
    }
```

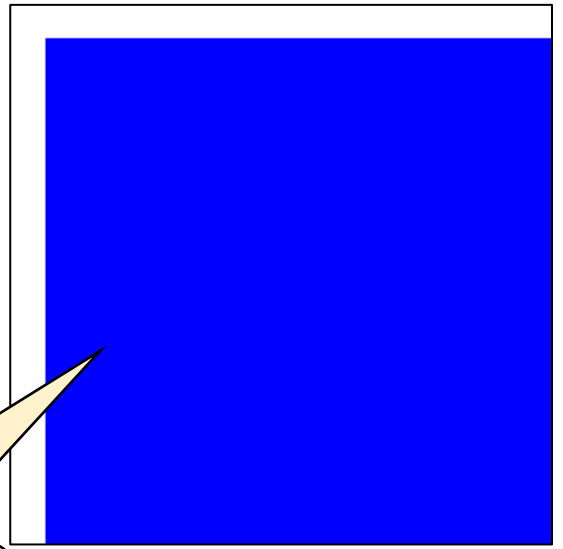
Camada Média:
200 x 200

```css
#camada3 {
        height:100px;
        width:100px;
        background-color: green;
        position:absolute;
        left:30px;
        top:30px;
    }
```

Camada Pequena:
100 x 100

Ao **adicionar no HTML** as **DIVs foram Invertidas**!

```html
<div id="camada3"></div>
<div id="camada2"></div>
<div id="camada1"></div>
```



As demais DIVs foram sobrepostas!

Com o uso do Z-INDEX

```css
#camada1 {
        height:400px;
        width:400px;
        background-color: blue;
        position:absolute;
        left:30px;
        top:30px;
        z-index:1;
    }
```

Terceiro
Z-INDEX: 1

```css
#camada2 {
        height:200px;
        width:200px;
        background-color: yell;
        position:absolute;
        left:30px;
        top:30px;
        z-index:2;
    }
```

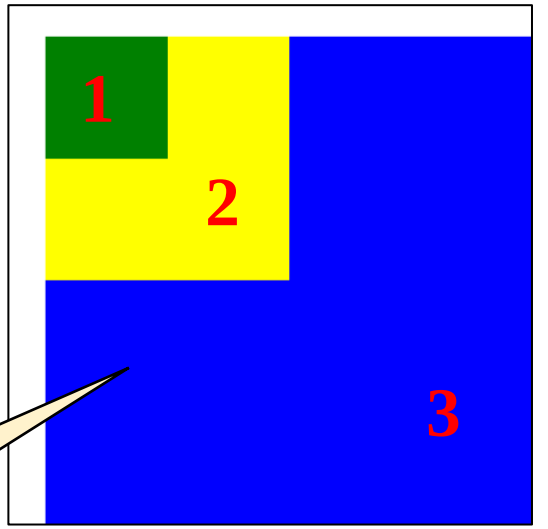
Segundo
Z-INDEX: 2

```css
#camada3 {
        height:100px;
        width:100px;
        background-color: green;
        position:absolute;
        left:30px;
        top:30px;
        z-index:3;
    }
```

Primeiro
Z-INDEX: 3

Veja que as DIVs continuam invertidas!

```html
<div id="camada3"></div>
<div id="camada2"></div>
<div id="camada1"></div>
```



Mas agora está respeitando o Z-INDEX!

Dúvidas?

nathan.silva@docente.unip.br

sergio.soares@docente.unip.br

Vamos Trabalhar!

*Resolva a **Lista da Aula 4**.*

Atenção:

*Enviar pelo **TEAMS** conforme padrão estabelecido.*