

Programação WEB

CONTEÚDO:

- HTML E CSS
- CONCEITOS INICIAIS E SINTAXE BÁSICA
- TAGS PRINCIPAIS
- FORMULÁRIOS
- COMO INSERIR O CSS NAS PÁGINAS
- SELETORES CSS
- TAG DIV E TAG SPAN
- EXEMPLOS
- EXERCÍCIOS

Alguns conceitos básicos (HTML)

HTML (Hypertext Markup Language) = Linguagem de Marcação de Hipertexto

A linguagem HTML desde a sua primeira especificação teve como objetivo principal estruturar documentos; sua especificação não visa a formatação ou função de apresentação visual como cor de fonte, aspectos de layout, etc.

Vale ressaltar que algumas tags possuem uma formatação prévia, porém são formatações que ainda estão dentro dos objetivos de estruturar o documento HTML.

Alguns conceitos básicos (CSS)

CSS (Cascading Style Sheet) = Folha de Estilo em Cascata

O CSS é uma linguagem que nos permite adicionar estilos as nossas páginas, em outras palavras, seria a formatação visual (aparência, layout) das páginas, como tipo de fonte, cores, espaçamentos, entre outros.

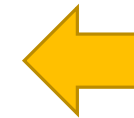
A W3C (World Wide Web Consortium) também especifica o CSS, existem diversos módulos que os navegadores aceitam em relação a última versão.

Com o uso do CSS externo, diminuímos o trabalho devido a possibilidade de reutilização dos estilos em diferentes tags e principalmente em diferentes páginas.

HTML e CSS?

Resumindo:

- O HTML é utilizado para estruturar as páginas, ou seja, definir parágrafos, listas numeradas, tabelas, cabeçalhos etc.
- O CSS é utilizado para formatar o visual da página (layout) estruturada com HTML, ou seja, adicionar a formatação de parágrafos, fontes, tabelas, entre outros.



HTML e CSS?

A linguagem HTML precisa do CSS (folhas de estilo) para formatar o conteúdo de um documento e do Javascript para dar interatividade.

Um exemplo simples, as páginas abaixo possuem o mesmo código html, porém em uma delas aplicamos o CSS

Título da Página

Subtítulo

- [Home](#)
- [Artigos](#)
- [Sobre](#)
- [Contato](#)

- [Link 1](#)
- [Link 2](#)
- [Link 3](#)
- [Link 4](#)
- [Link 5](#)

Título do artigo

Agora o Brasil é o segundo e tem oito pontos, dois a menos que os Estados Unidos e o mesmo que a Rússia, com vantagem no set average. Para confirmar a vice-liderança, a equipe de Bernardinho ainda precisa bater a Alemanha, quarta colocada, na rodada decisiva, para decidir quem é o segundo lugar nos critérios de desempate.

Título do artigo

Agora o Brasil é o segundo e tem oito pontos, dois a menos que os Estados Unidos e o mesmo que a Rússia, com vantagem no set average. Para confirmar a vice-liderança, a equipe de Bernardinho ainda precisa bater a Alemanha, quarta colocada, na rodada decisiva, para decidir quem é o segundo lugar nos critérios de desempate.

SOMENTE HTML5

Título da Página

Subtítulo

Home	Artigos	Sobre	Contato
<ul style="list-style-type: none">• Link 1• Link 2• Link 3• Link 4• Link 5	<h3><u>Título do artigo</u></h3> <p>Agora o Brasil é o segundo e tem oito pontos, dois a menos que os Estados Unidos e o mesmo que a Rússia, com vantagem no set average. Para confirmar a vice-liderança, a equipe de Bernardinho ainda precisa bater a Alemanha, quarta colocada, na rodada decisiva, para decidir quem é o segundo lugar nos critérios de desempate.</p> <h3><u>Título do artigo</u></h3> <p>Agora o Brasil é o segundo e tem oito pontos, dois a menos que os Estados Unidos e o mesmo que a Rússia, com vantagem no set average. Para confirmar a vice-liderança, a equipe de Bernardinho ainda precisa bater a Alemanha, quarta colocada, na rodada decisiva, para decidir quem é o segundo lugar nos critérios de desempate.</p>		

HTML5 + CSS

HTML



Tags, alguns detalhes

Geralmente uma tag é utilizada delimitando um texto, com sua abertura e fechamento, no HTML existem algumas tags que não possuem fechamento.

Exemplo de tag com fechamento

`<p>` Está tag define um parágrafo `</p>`

Exemplo de tag sem fechamento

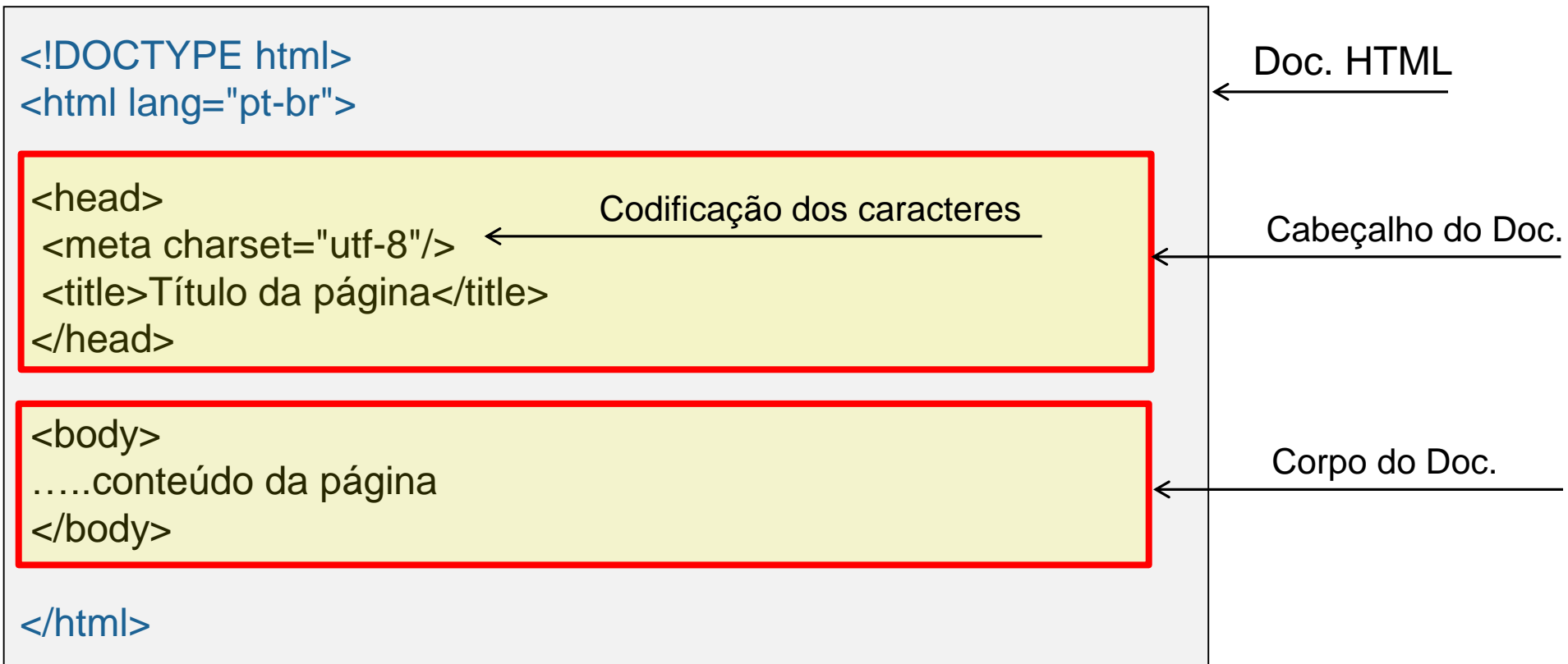
Está tag quebra a linha. `
`

As tags são formadas pelos sinais "<" e ">", como no exemplo da estrutura básica. `<html> </html>`

Existem algumas tags que não são mais utilizadas devido ao uso do CSS.

Estrutura básica

Um documento HTML5 é formado basicamente pela estrutura:



Elementos de uma página HTML

- Os Elementos sempre ocupam um espaço quadrado na tela, que tem uma largura e uma altura. Quando estudarmos CSS veremos que eles podem ter seu tamanho alterado assim como ter sua posição natural alterada (podendo, inclusive, se sobreporem)
- Elementos podem conter outros Elementos (um bloco pode conter dois outros blocos menores dentro dele, por exemplo)
- O que define um elemento no HTML são as TAGs. Um elemento que pode ter conteúdo marcado deve ter uma TAG inicial e uma TAG de fechamento. Elementos que não devem ter conteúdo delimitado (como imagem ou linha horizontal) não possuem TAG de fechamento.

Padrão na criação do código

Para criarmos nossos documentos HTML, iremos seguir as observações abaixo:

- Devemos criar os documentos bem-formados.
- Todas as tags devem ser escritas com **letras minúsculas**.
- Uso de tags de fechamento é obrigatória.
- Elementos vazios (br, hr, ...) podem ser fechados com "/".
- Atributos devem ser escritos também com **letras minúsculas**.
- Os valores dos atributos devem ser escritos dentro de aspas ("....").
- Todos os atributos devem ter nome e valor associados.

Exemplo da tag `div` com um atributo `id`:

```
<div id="principal">Esta é o texto de uma div</div>
```

Caracteres Especiais

A HTML possui algumas codificações para caracteres, por exemplo, não se pode usar os caracteres "<" e ">" em um texto, pois o navegador pode confundi-los com as marcações do documento, devemos substituí-los por códigos.

Alguns exemplos de codificação e seus caracteres:

Caracteres	Nome no HTML	Descrição
"	"	Aspas (quotation mark)
'	'	Apóstrofe
&	&	"E" comercial (ampersand)
<	<	Menor que (less-than)
>	>	Maior que (greater-than)
	 	Espaço (branco)

Codificação de caracteres

"Conteúdo é composto de uma sequência de caracteres. Caracteres representam letras do alfabeto, pontuação, etc. Conteúdos são armazenados em um computador como uma sequência de bytes, que são valores numéricos. Em alguns casos um simples caractere é representado por mais de um byte. Tal como os códigos usados em espionagem a maneira como uma sequência de bytes é convertida em caracteres depende do formato como o conteúdo foi codificado. Nesse contexto tal *formato* é denominado **codificação de caracteres**. Uma página HTML pode ter apenas uma codificação de caracteres. A codificação baseada em Unicode, tal como UTF-8, oferece suporte para vários idiomas e assim sendo admite páginas e formulários em qualquer combinação de idiomas." (W3C, 2014)

No HTML5, a codificação padrão de caracteres é UTF-8.

Para mais informações acesse:

<http://www.w3schools.com/charsets/>

<http://www.w3.org/International/O-charset.pt-br.php>

Tag <meta />

Tag usada para definir informações do documento como autor, palavras-chave etc. Vamos utilizar somente a tag meta para definir a codificação de caracteres das páginas e a visualização em aparelhos móveis.

Exemplos:

- especifica a codificação de caracteres da página

```
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Espaços em branco

Em linguagens de programação, espaços em branco (gerados pela tecla de espaço ou pelo tecla TAB do teclado) servem para separar os termos. Porém, mais de um espaço em sequência poderia ter o mesmo significado que um único espaço.

Assim é no HTML.

Por exemplo, os dois textos a seguir tem o mesmo resultado numa página HTML, porque espaços repetidos serão ignorados:

Este texto é separado por espaços únicos

Este texto é separado por espaços únicos

Novas linha (tecla Enter do teclado)

Em algumas linguagens de programação, a separação em linhas (inseridas pela tecla Enter do teclado) servem apenas para organizar o documento. Ou seja, se elas não forem colocadas, dá na mesma!

Assim é no HTML

Por exemplo, os dois textos a seguir tem o mesmo resultado numa página HTML (a utilização das tags *br* ou *p*, por exemplo, forçariam as quebras de linha) :

Tudo é apresentado em uma linha

Este texto está em uma linha

Este texto é
está em
uma linha

Criamos parágrafos com *p* e quebra de linha com *br*

`<p>Este texto é um parágrafo</p>`

Este texto

está em

várias linhas.

Tags principais

<p> - utilizada para delimitar um parágrafo

<p>Este conteúdo foi definido como um parágrafo</p>

 - texto importante

em negrito

 - ênfase

em itálico

 - quebra de linha

Linha 1

Linha 2

Tags principais

<h1> até <h6> - cabeçalho, ou títulos

<h1>Título da página grande</h1>

<h2>Título da página menor</h2>

<h3>Título da página ainda menor</h3>

<hr> - quebra temática de linha

Linha 1

<hr />

Linha 2

Listas

Em HTML podem ser criadas listas ordenadas (numéricas, alfabéticas etc.) ou não ordenadas (utiliza marcadores como imagens).

`` - lista ordenada

```
<ol>  
  <li> item 1</li>  
  <li> item 2</li>  
</ol>
```

1. Item 1
2. Item 2
3. Item 3

`` - lista não ordenada

```
<ul>  
  <li> item 1</li>  
  <li> item 2</li>  
</ul>
```

- Item 1
- Item 2
- Item 3

Tabelas

Para definir uma tabela básica, utilize as tags:

`<table>` define uma tabela

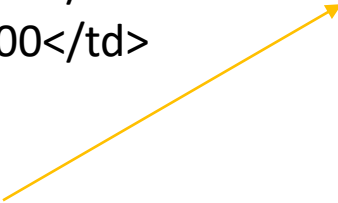
- `<tr>` define uma linha na tabela
- `<td>` define uma coluna na tabela

```
<table>
  <tr>
    <td>Linha 1 - Coluna 1</td>
    <td>Linha 1 - Coluna 2</td>
  </tr>
  <tr>
    <td>Linha 2 - Coluna 1</td>
    <td>Linha 2 - Coluna 2</td>
  </tr>
</table>
```

Tabelas

É possível construir uma tabela simples apenas com as tags **tr** e **td**:

```
<table border="1">
  <tr>
    <td>Mês</td>
    <td>Vendas</td>
  </tr>
  <tr>
    <td>JAN</td>
    <td>600</td>
  </tr>
  <tr>
    <td>FEV</td>
    <td>600</td>
  </tr>
  <tr>
    <td>Soma</td>
    <td>R$ 1200,00</td>
  </tr>
</table>
```



Mês	Vendas
JAN	600
FEV	600
Soma	R\$ 1200,00

Tabelas

Para definir uma tabela completa:

Mês	Vendas
JAN	600
FEV	600
Soma	R\$ 1200,00

```
<table border="1">  
  <thead>  
    <tr>  
      <th>Mês</th>  
      <th>Vendas</th>  
    </tr>  
  </thead>  
  <tbody>  
    <tr>  
      <td>JAN</td>  
      <td>600</td>  
    </tr>  
    <tr>  
      <td>FEV</td>  
      <td>600</td>  
    </tr>  
  </tbody>  
  <tfoot>  
    <tr>  
      <td>Soma</td>  
      <td>R$ 1200,00</td>  
    </tr>  
  </tfoot>  
</table>
```

- 1 - <thead> = cabeçalho da tabela
 - 1.1 - <th> colunas do cabeçalho
- 2 - <tfoot> = última linha da tabela
- 3 - <tbody> = conteúdo da tabela

A tag div

A função básica da tag DIV é de criar blocos de textos, porém também é utilizada na criação de layouts dos sites, para definir regiões genéricas da página, ou seja, regiões que não precisam de uma semântica (significado) no conteúdo. Também pode ser utilizada para marcar conteúdos que serão manipulados pelo JavaScript.

```
<div>
```

```
  <div> Itens do cabeçalho do site</div>
```

```
  <div> Itens do conteúdo do site</div>
```

```
</div>
```

Exemplo

Olimpíadas 2012 - Londres

- Conteúdo
- Esportes
 - Quantidade de medalhas
 - Países

Esportes

Atletismo

Descrição

O atletismo é o esporte mais tradicional do programa olímpico. Disputado desde a primeira edição, atualmente distribui 47 medalhas de ouro, 24 entre os homens e 23 entre as mulheres.

As provas são divididas em:

- *Campo* (saltos e arremessos)
- *Pista* (corridas com e sem obstáculos e revezamento)
- *Rua* (maratona e marcha)
- *Combinadas* (decatlo e heptatlo)

Basquete

Descrição

O objetivo é marcar o maior número de pontos, acertando uma cesta de 45 cm que fica a 3,05 m do chão. Cada cesta comum no basquete vale dois pontos. Lances livres (cobranças de infrações) valem um ponto. Arremessos a uma distância igual ou superior a 6,75 m da cesta valem três pontos. O jogo é disputado em quatro quartos de 10 minutos, e, a cada parada da bola, o cronômetro também para.

Quantidade de medalhas

País	Ouro	Prata	Bronze
China	29	16	14
EUA	27	14	15
Grã-Bretanha	16	10	10

Países

1. China
2. EUA
3. Grã-Bretanha

Referências Absolutas

Uma referência absoluta é aquela que inclui todo o caminho de um arquivo, incluindo o protocolo de comunicação (HTTP, por exemplo). Essa referência é válida sempre que o arquivo permaneça em um mesmo diretório.

Utilizam-se referências absolutas sempre que se deseja referenciar um arquivo que não faz parte da página construída.

```

```


Referências Relativas

Uma referência relativa é aquela que se expressa a partir de um diretório conhecido, dentro da mesma página.

no mesmo diretório:

```

```

ou na pasta filha `img`:

```

```

Caso a pasta "img" esteja um nível acima da pasta raiz, utiliza-se algo como:

```

```

Tag para Links

Utilizado para ligação das páginas.

Exemplos:

- para acessar um arquivo que está dentro de uma pasta filha **paginas**, a partir do diretório principal do projeto

```
<a href="paginas/gato.html">Texto ou conteúdo do enlace</a>
```

- para acessar um arquivo que se encontra em um nível superior, a partir da pasta do arquivo atual

```
<a href=" ../cavalo.html"> Texto ou conteúdo do enlace</a>
```

- para acessar outros arquivos na web

```
<a href="http://www.microsoft.com.br/index.html ">Site da Microsoft Brasil</a>
```

Nunca crie seus links com o caminho físico do arquivo, ou seja, sua localização real na máquina, como **C:\site\index.html**, pois caso você disponibilize esse arquivo na Web e em outros sistemas como Linux, irá resultar em Página não encontrada.

Tag para Imagens

Imagens suportadas nas páginas Web

- GIF / GIF animado
 - ideal para desenhos, ícones, figuras simples e animações simples
 - Máximo de 256 cores
 - permite transparência
- JPG (a mesma coisa de JPEG)
 - ideal para banners de sites, imagens para compor o layout da página e fotos
 - mais de 16 milhões de cores
- PNG
 - formato livre para imagens
 - aceita mais de 16 milhões de cores
 - não perde resolução no processo de salvar a imagem
 - permite transparência


Testar exemploTAG_IMG.html disponível no arquivo zipado exemplosHTML.

Formulários

Quem nunca preencheu um formulário de cadastro na WEB? A ideia de formulário todos conhecem. Em resumo, sua principal utilidade é coletar os dados do usuário e fazer algum tipo de processamento no servidor, porém também podemos utilizar para processar algo em JavaScript, no computador cliente.

Podemos criar nossos formulários através do HTML utilizando tags para cada tipo de elemento de entrada de dados. No HTML5 também temos alguns atributos que irão facilitar a validação do formulário sem a necessidade de desenvolver scripts em outras linguagens, ou seja, para validar alguns dados não precisaremos utilizar o JavaScript diretamente.

Nome	<input type="text" value="Nome"/>
Email	<input type="text" value="Email"/>
Telefone	<input type="text" value="Telefone"/>
Assunto	<input type="text" value="Assunto"/>
Mensagem	<input type="text" value="Mensagem"/>
<input type="button" value="Enviar"/>	

Email:	<input type="text" value="name@example.com"/>	
Password:	<input type="password" value="....."/>	
Esqueceu sua senha?		

Formulários

Como a linguagem HTML5 ainda está em desenvolvimento, alguns atributos e comandos de formulários não são compatíveis com todos os navegadores, por isso iremos utilizar somente o que for compatível com pelo menos dois navegadores (Google Chrome e Firefox).

Para criarmos um formulário podemos utilizar a tag `<form>` ou em alguns casos somente o elemento como um botão para disparar uma ação na nossa página.

Tag form

```
<form name="form1" method="post" action="">
```

Elementos do formulário

```
</form>
```

get ou post

Endereço, página
do servidor

Formulários (tag input)

Tag para entrada de dados, que poderá ter diversas variações, conforme exemplos abaixo:

Exemplos:

Nome: `<input type="text" name="nome" />`

Senha: `<input type="password" name="senha" />`

E-mail: `<input type="email" name="usermail" />`

URL: `<input type="url" name="homepage" />`

Submeter o form: `<input type="submit" value="Enviar" />`

Omitimos a tag label (ver depois) para facilitar o exemplo.

Formulários (tag input) cont.

Campo oculto

```
<input name="oculto" type="hidden" value="algumacoisa"/>
```

Checkbox (o usuário pode marcar mais de uma opção)

```
<input type="checkbox" name="carros" value="Camaro" />Camaro <br/>  
<input type="checkbox" name="carros" value="Ferrari" />Ferrari
```

Button para chamar funções em Javascript

```
<input name="botao" type="button" value="Clique aqui" />
```

Omitimos a tag label (ver depois) para facilitar o exemplo.
Observe que o valor do argumento *name* é o mesmo para todos os checkboxes.

Formulários (tag input) cont.

File para carregar arquivos no formulário

Escolha um arquivo `<input name="arquivo" type="file" />`

Image para definir um botão com imagem para enviar o formulário

`<input name="imagem" type="image" src="images.jpg" />`

Radio (neste tipo o usuário poderá marcar somente uma opção desde que o name seja o mesmo para os elementos do mesmo grupo)

`<input type="radio" name="genero" value="F" />Feminino
`

`<input type="radio" name="genero" value="M" />Masculino`

Omitimos a tag label (ver depois) para facilitar o exemplo.
Observe que o valor do argumento *name* é o mesmo para todos os radiobuttons.

Comentários sobre os argumentos *id* e *name*

Nos exemplos anteriores mostramos a utilização dos argumentos ou atributos *id* e *name* nos diferentes tipos de tags *input*.

```
<input type="radio" name="sexo" id="masculino" value="m" checked>
```

```
<input type="text" name="nome" id="nome" />
```

O atributo **id** é frequentemente utilizado em JavaScript para acessar os elementos do documento (da página) e efetuar algum processamento com os mesmos no computador do cliente.

O atributo **name** é frequentemente utilizado em elementos de formulários (Form) que serão enviados para um servidor Web, permitindo desta forma o processamento dos dados em alguma "página destino".

O atributo **name** possui características de vetor, sendo interessante sua utilização em componentes como listas de itens e grupos de checkboxes, permitindo que a "página destino" consiga recuperar os valores selecionados pelo usuário na "página inicial".

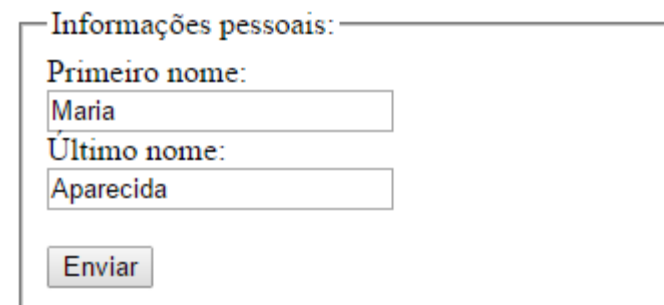
Este assunto será aprofundado na disciplina Programação Web.

Formulário (tag fieldset e legend)

FIELDSET é utilizado para agrupar elementos ou itens de formulários com características em comum, o Chrome por exemplo, delimita essa tag com uma borda, lembrando que isso pode ser formatado em CSS

LEGEND é usado para definir uma legenda para o fieldset

```
<form action="pagina.php">
  <fieldset>
    <legend>Informações pessoais: </legend>
    Primeiro nome:
    <br>
    <input type="text" name="firstname" value="Maria">
    <br/> Último nome:
    <br>
    <input type="text" name="lastname" value="Aparecida">
    <br/>
    <br/>
    <input type="submit" value="Enviar">
  </fieldset>
</form>
```



Informações pessoais:

Primeiro nome:
Maria

Último nome:
Aparecida

Enviar

Formulários (tag button)

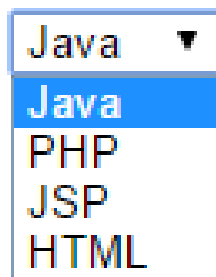
Nesta tag você poderá colocar texto ou imagens como um botão *clicável*. Este comando pode ser utilizado fora de um formulário.

```
<button type="button">  
    
</button>
```

Formulários (tag select)

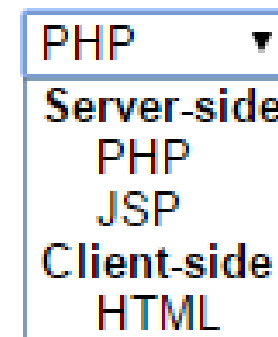
Cria uma lista de opções (conhecido como combobox)

```
<form>
  <select>
    <option value="Java">Java</option>
    <option value="PHP">PHP</option>
    <option value="JSP">JSP</option>
    <option value="HTML">HTML</option>
  </select>
</form>
```



Java ▼
Java
PHP
JSP
HTML

```
<form>
  <select>
    <optgroup label="Server-side">
      <option value="PHP">PHP</option>
      <option value="JSP">JSP</option>
    </optgroup>
    <optgroup label="Client-side">
      <option value="HTML">HTML</option>
    </optgroup>
  </select>
</form>
```



PHP ▼
Server-side
 PHP
 JSP
Client-side
 HTML

Formulários (tag textarea)



Define um campo de texto de várias linhas

```
<form>
```

```
  <textarea rows="8" cols="70" name="msg"></textarea>
```

```
</form>
```

Formulário (alguns atributos comuns)

Atributo	Descrição
disabled	Especifica que o campo está desabilitado, o usuário não pode alterar e o mesmo não é enviado para o servidor
maxlength	Especifica o número máximo de caracteres que podemos digitar no elemento
pattern	Especifica uma expressão regular, um formato pré-determinado
readonly 	Especifica que o campo está somente em modo leitura, semelhante ao disabled, porém neste caso, o valor é enviado para o servidor
required	Especifica que o campo é obrigatório
size	Especifica o tamanho (largura) do campo em caracteres
value 	Especifica um valor padrão inicial para o campo

O **exemplo1.html** demonstra a utilização de diferentes tags em um formulário e utiliza alguns destes atributos.

Formulários (atributo required)

Este atributo pode ser utilizado em campos para obrigar a digitação dos dados. A mensagem será ativada quando o usuário tentar enviar o formulário.

Pode ser utilizado em diversas tags de entrada de dados no formulário.

```
<input name="nome" required="required" />
```

```
<input type="submit"/>
```

Formulários (atributo autofocus)

Este atributo identifica o elemento ao qual será dado o foco quando a página for carregada. Pode ser utilizado em diversos elementos de um formulário.

```
<form action="x.php">
```

```
  Nome:<input type="text" name="nome" autofocus="autofocus" /><br />
```

```
  Idade: <input type="text" name="idade" /><br />
```

```
  <input type="submit" />
```

```
</form>
```


Formulários (atributo placeholder)

Atributo exclusivo de elementos input e textarea. Define uma dica para digitação no campo. Assim que o usuário começar a digitar, a dica é automaticamente retirada.

```
<form action="demo_form.asp">  
  <input type="text" name="user" placeholder="Digite seu Usuário" /><br />  
  <input type="text" name="senha" placeholder="Digite sua Senha" /><br />  
  <input type="submit" value="Submit" />  
</form>
```

Formulário + server side

<form action="processa.php" method="post">

Nome:

Sexo: ☒ M ☐ F

E-mail:

Assunto:

Curso:

Mensagem:

</form>

Processa.php

```
<?php
header('Content-Type: text/html; charset=utf-8');
echo "Dados recebidos do formulário!!!<br/><br/>";
echo "Você digitou:<br/>";
echo "Nome: " .$_POST["nome"]."<br />";
echo "Sexo: " .$_POST["sexo"]."<br />";
echo "E-mail: " .$_POST["email"]."<br />";
echo "Assunto: " .$_POST["assunto"]."<br />";
echo "Curso: " .$_POST["curso"]."<br />";
echo "Mensagem: " .$_POST["mensagem"]."<br />";
?>
```

Exibe no navegador

Dados recebidos do formulário!!!

Você digitou:

Nome: ALCIDES TEIXEIRA BARBOZA JUNIOR

Sexo: M

E-mail: alcidesprof@html.css

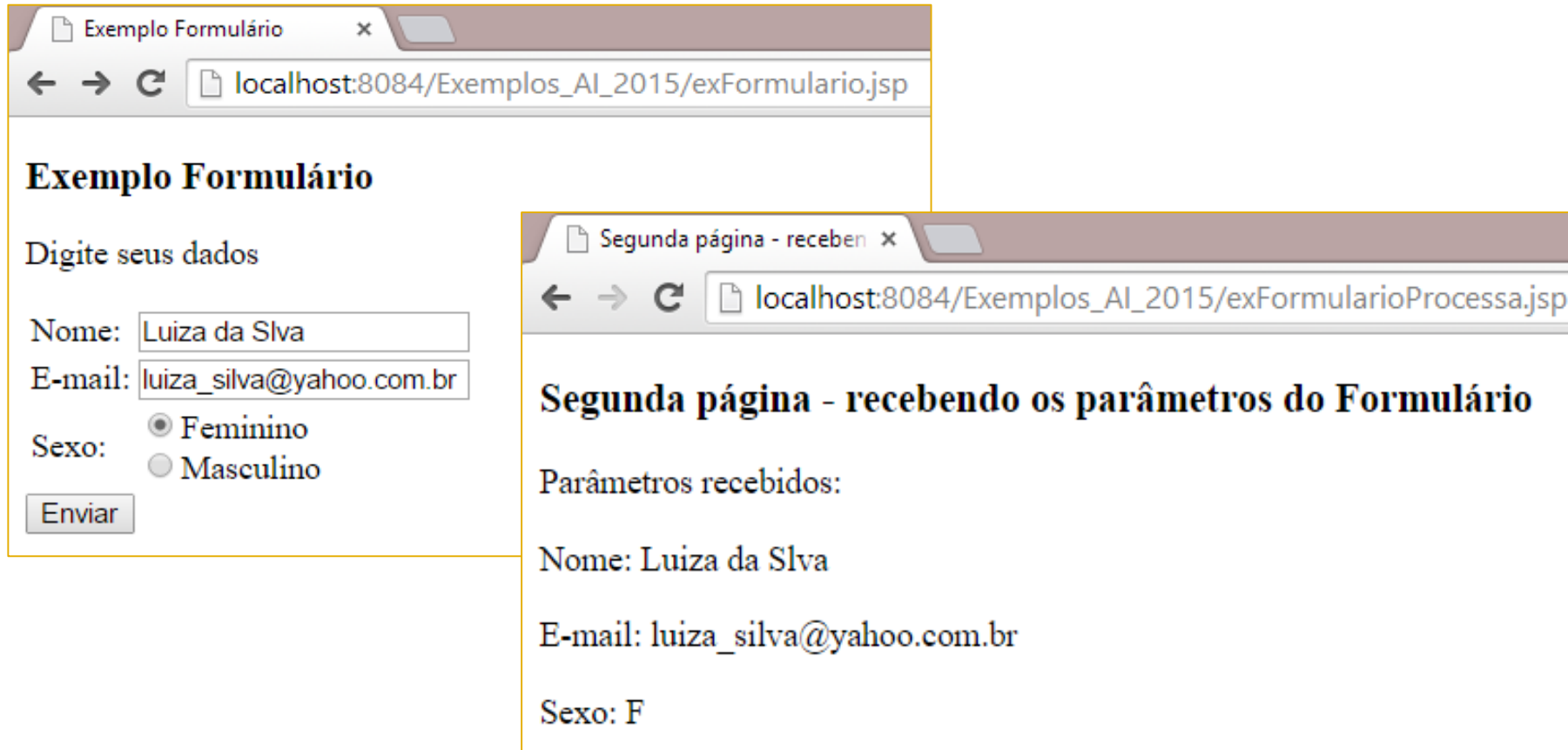
Assunto: Teste de Aula

Curso: TJD

Mensagem: Dúvida sobre o bloco de notas...

Para rodar esse exemplo completo precisamos ter o Servidor Web + PHP

Formulário + server side



The image displays two browser windows side-by-side. The left window, titled 'Exemplo Formulário', shows a web form at 'localhost:8084/Exemplos_AI_2015/exFormulario.jsp'. The form contains fields for 'Nome' (Luiza da Silva), 'E-mail' (luiza_silva@yahoo.com.br), and 'Sexo' (Feminino selected). An 'Enviar' button is at the bottom. The right window, titled 'Segunda página - recebendo', shows the processing page at 'localhost:8084/Exemplos_AI_2015/exFormularioProcessa.jsp'. It displays the received parameters: 'Nome: Luiza da Silva', 'E-mail: luiza_silva@yahoo.com.br', and 'Sexo: F'.

Exemplo Formulário

Digite seus dados

Nome:

E-mail:

Sexo: ☒ Feminino ☐ Masculino

Segunda página - recebendo os parâmetros do Formulário

Parâmetros recebidos:

Nome: Luiza da Silva

E-mail: luiza_silva@yahoo.com.br

Sexo: F

Para rodar esse exemplo completo precisamos ter o Servidor Web Tomcat ou GlassFish

Formulário + server side

```
<body>
  <h3>Exemplo Formulário</h3>
  <form action="exFormularioProcessa.jsp" method="post">
    <p>Digite seus dados</p>
    <table>
      <tr>
        <td> Nome: </td>
        <td>
          <input name="nome" maxlength="20" autofocus="autofocus" required="required" />
        </td>
      </tr>
      <tr>
        <td> E-mail: </td>
        <td>
          <input name="email" maxlength="15" required="required" />
        </td>
      </tr>
      <tr>
        <td> Sexo: </td>
        <td>
          <input type="radio" name="sexo" value="F" checked />Feminino <br/>
          <input type="radio" name="sexo" value="M" />Masculino
        </td>
      </tr>
    </table>
    <input type="submit" value="Enviar" />
  </form>
```

Formulário + server side

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Segunda página - recebendo os parâmetros do Formulário</title>
  </head>
  <body>
    <h3>Segunda página - recebendo os parâmetros do Formulário</h3>
    <%
      String nome = request.getParameter("nome");
      String email = request.getParameter("email");
      String sexo = request.getParameter("sexo");
    %>
    <p>Parâmetros recebidos:</p>
    <p>Nome:  <%=nome%> </p>
    <p>E-mail: <%=email%> </p>
    <p>Sexo:  <%=sexo%> </p>
  </body>
</html>
```

Alguns atalhos no Visual Studio Code

```
! Ou html:5
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Document</title>
  </head>
  <body>

  </body>
</html>
```

```
link:css
<link rel="stylesheet" href="style.css" />
```

```
ul>li*5
<ul>
  <li></li>
  <li></li>
  <li></li>
  <li></li>
  <li></li>
</ul>
```

```
ul>li.item$*5
<ul>
  <li class="item1"></li>
  <li class="item2"></li>
  <li class="item3"></li>
  <li class="item4"></li>
  <li class="item5"></li>
</ul>
```

```
style
<style></style>

script
<script></script>

script:src
<script src=""></script>
```

```
table>tr>td
<table>
  <tr>
    <td></td>
  </tr>
</table>
```

No arquivo cheatsheet-a5.pdf tem vários atalhos.

CSS



Como inserir o CSS?

Para inserir o CSS em uma página HTML, podemos escolher as seguintes opções:

- CSS externo
- CSS incorporado
- CSS inline (deve ser evitado, porque mistura a estrutura com o visual)

CSS externo

Toda a configuração de formatação fica dentro de um arquivo .css que é chamado no cabeçalho do documento html com a tag link


Ex:

```
<link rel="stylesheet" href="estilo.css" type="text/css" />
```

As configurações definidas no arquivo estilo.css valem para todas as páginas que fazem referência ao arquivo.

Exemplo simples:

```
<!--exemplo1.html-->
<html>
<head>
  <title> Teste CSS externo </title>
  <link rel="stylesheet" href="estilo.css" type="text/css" />
</head>
<body>
<p> Somente um parágrafo </p>
</body>
</html>
```



```
/*estilo.css*/
p {
  font-family: "verdana";
  color: red;
}
```

CSS incorporado

Todas as configurações ficam dentro do cabeçalho (HEAD) da página e são delimitadas pelas tags <style>...</style>

As configurações valem para a página inteira.

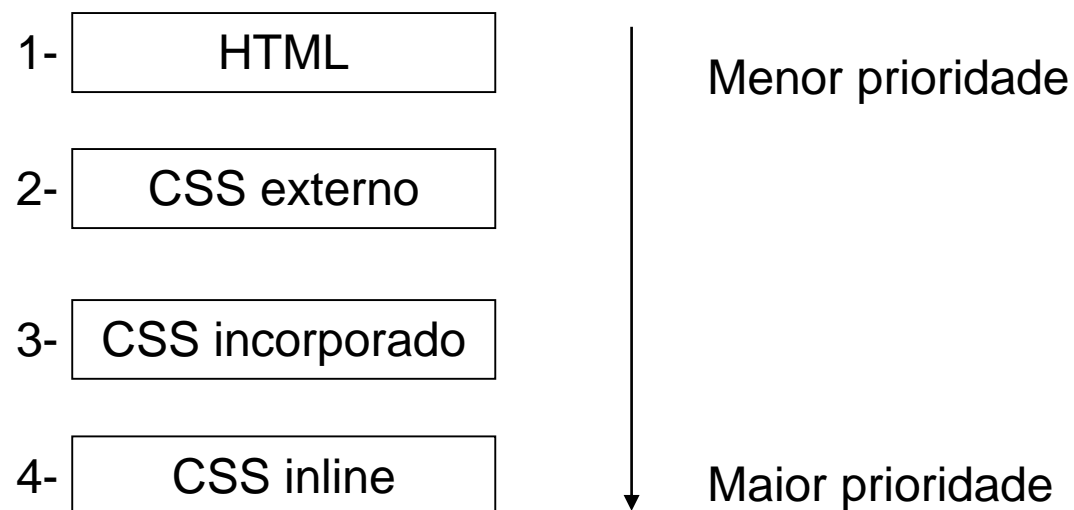
Este tipo de CSS não permite a reutilização, como seria no CSS externo.

Exemplo simples:

```
<!--exemplo2.html-->
<html>
<head>
  <title> Teste CSS embutido </title>
  <style type="text/css">
    p {
      font-family: "verdana";
      color: red;
    }
  </style>
</head>
<body>
  <p> Somente um parágrafo </p>
</body></html>
```

Prioridades

Caso você utilize as três formas de inserir CSS em um documento HTML, deve ficar atento a prioridade entre eles, conforme demonstrado abaixo:



Na figura acima, o CSS inline sobrepõe os demais estilos, caso este não exista, o CSS incorporado sobrepõe o externo e assim ocorre com as demais variações.

Seletores CSS

Um seletor CSS é um padrão criado para ser aplicado ao elemento(s) desejado(s) no HTML.

Existem diversos seletores, aqui iremos abordar as principais, porém um estudo aprofundado pode ser obtido através dos links:

- http://www.maujor.com/tutorial/seletores_css21_parte1.php
- <http://www.w3.org/TR/css3-selectors/>
- http://www.w3schools.com/cssref/css_selectors.asp

Iremos utilizar os seguintes seletores (básicos):

- Para um elemento específico (seletor tipo)
- Para classes genéricas (seletor classe)
- Para estilos individuais (seletor id)

Para um elemento específico (seletor tipo)

Podemos definir nossos estilos para qualquer elemento do HTML.

Sintaxe:

elemento { **propriedade**: **valor** }

↓
Tag HTML

↓
Atributo que
queremos
modificar

↓
Valor da
propriedade

Exemplo

```
p { color:blue}

b {

  color:red;
  font-family: verdana;

}

i {

  color:red;
  font-family: "sans serif";

}
```

Um padrão que é utilizado para a criação dos estilos é inserir uma propriedade em cada linha, isso facilita a leitura posterior.

```
body {
  color:green;
  font-family: "sans serif";
}
```

```
<!--exemplo3.html-->
<html><head>
<style type="text/css">
  p { color:blue}
  b { color:red; font-family: verdana; }
  i { color:red; font-family: "sans serif"; }
  body {
    color:green;
    font-family: "sans serif";
  }
</style>
</head>
<body>
  <p> parágrafo </p>
  <b> negrito</b><br />
  <i> itálico </i><br />
  Texto do body.
</body></html>
```

Classes e estilos individuais

Até o momento, definimos configurações de apresentação para elementos específicos do HTML. Sempre que usamos o elemento formatado em nosso documento, ele assume a formação definida no CSS. Surge então um problema: como formatar um mesmo elemento de diferentes formas em nosso documento?

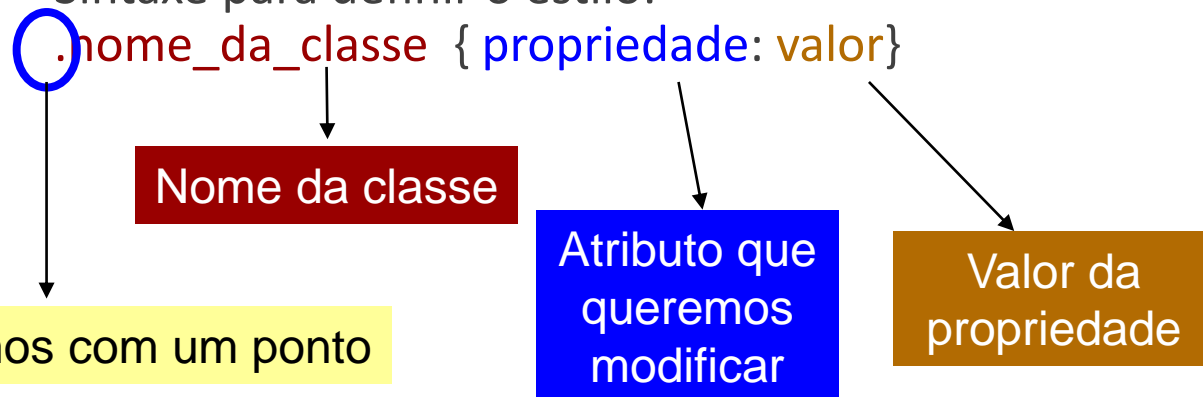
Para resolver esse problema, existem as classes e os estilos individuais.

Para classes genéricas (seletor classe)

Podemos criar classes genéricas (não estão associadas a nenhum elemento) que podem ser aplicadas a qualquer elemento do HTML, sempre que for necessário, assim, uma classe pode ser usada várias vezes em um mesmo documento.

Para utilizar uma classe em um marcador, acrescentamos o atributo **class**.

Sintaxe para definir o estilo:



Exemplo

```
<!--exemplo4.html-->
```

```
<html>
```

```
<head>
```

```
<title>Exemplo</title>
```

```
<style type="text/css">
```

```
.cor_azul { color:blue}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<p> parágrafo 1 sem formatação </p>
```

```
<p class="cor_azul"> parágrafo 2 com formatação</p>
```

```
<i class="cor_azul"> itálico com formatação </i><br />
```

```
Texto do body.
```

```
</body></html>
```

Cria a classe "cor_azul" genérica, logo, pode ser usada em qualquer tag do HTML.

Exemplo

parágrafo 1 sem formatação

parágrafo 2 com formatação

itálico com formatação

Texto do body.

Aplica a classe "cor_azul" na tag p e na tag i. Veja que para aplicar o estilo, devemos chamar a classe com o atributo "class".

Para estilos individuais (seletor id)

Assim como as classes genéricas, os estilos individuais podem ser aplicados a qualquer elemento do HTML. A diferença é que esses estilos são únicos, logo, só podem ser utilizados uma única vez dentro do mesmo documento.

São utilizados também para identificar um elemento HTML para posteriormente ser manipulado através da linguagem JavaScript. Este estilo é a identificação da tag.

Para utilizar um estilo individual em um marcador, acrescentamos o atributo **id**.

Sintaxe para definir o estilo:

#nome_do_id { propriedade: valor }

Nome do identificador

Atributo que
queremos
modificar

Valor da
propriedade

Iniciamos com #

Exemplo

```
<!--exemplo5.html-->
```

```
<html>
```

```
<head>
```

```
<title>Exemplo</title>
```

```
<style type="text/css">
```

```
#cor_azul { color:blue}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<p> parágrafo 1 sem formatação </p>
```

```
<p id="cor_azul"> parágrafo 2 com formatação</p>
```

```
<i> itálico </i><br />
```

```
Texto do body.
```

```
</body></html>
```

Cria o id "cor_azul", pode ser usado em qualquer tag do HTML, porém somente um vez.

Aplica o id "cor_azul" na tag p. Veja que para aplicar o estilo, devemos chamar o identificador com o atributo "id".



parágrafo 1 sem formatação

parágrafo 2 com formatação

itálico

Texto do body.

Vamos lembrar

Podemos criar classes genéricas que podem ser aplicadas a qualquer tag do HTML. Essas classes podem ser usadas mais de uma vez dentro do documento. Para acioná-la, usamos o atributo class na tag desejada.

Podemos criar um estilo individual que aplica um estilo e identifica uma tag do HTML. Esta forma de estilo só pode ser usada uma vez dentro do documento. Para acioná-lo, usamos o atributo id na tag desejada.

Podemos ter em uma mesma tag, os atributos class e id.

Ao chamar a classe em uma tag com o atributo class, colocamos somente o nome da classe, ou seja, na chamada da classe, não utilizamos o ponto.

O mesmo ocorre quando chamamos o estilo individual com o atributo id, também não inserimos o #, somente o nome do estilo.

CSS Combinators

Existem, também, os chamados CSS Combinators, para estabelecer estilos a tags que se encontram relacionadas com outras pela colocação dentro do documento.

Por exemplo, podemos construir estilos hierárquicos, de forma a aplicar um estilo apenas a tags que se encontrem dentro de outra, com as sintaxes:

tag1 > tag2

para uma tag2 filha de tag1

tag1 tag2

para uma tag2 descendente de tag1

```
<style type="text/css">
  body{
    background-color:#ebebeb;
  }
  p {
    color:black;
  }
  div > p {
    color:red;
  }
  div span {
    color:blue;
  }
</style>
```

CSS Combinators - exemplo

```
<style type="text/css">
  body{
    background-color:#ebebeb;
  }
  p {
    color:black;
  }
  div > p {
    color:red;
  }
  div span {
    color:blue;
  }
</style>
```

Estilos hierárquicos

Parágrafo que é filho direto da div alterado com o estilo hierárquico `div > p`

Parágrafo que não é filho direto da div não é alterado com o estilo `div > p`

Span que é filho direto da div alterado com o estilo hierárquico `div span`

Span que não é filho direto da div alterado com o estilo hierárquico `div span`

```
<body>
  <h3>Estilos hierárquicos</h3>

  <div>
    <p>Parágrafo que é filho direto da div alterado com o estilo hierárquico div > p </p>
    <span><p>Parágrafo que não é filho direto da div não é alterado com o estilo div > p</p></span>
  </div>

  <div>
    <span>Span que é filho direto da div alterado com o estilo hierárquico div span</span>
    <p><span>Span que não é filho direto da div alterado com o estilo hierárquico div span</span></p>
  </div>
</body>
```

Comentários no código CSS

Como geralmente os arquivos com CSS possuem muitas configurações, é interessante você inserir comentários descrevendo o que cada bloco de estilo faz. Para isso usamos `/*.....*/`

Exemplo:

```
<style type="text/css">  
    /*Cria um id chamado cor_azul*/  
    #cor_azul { color:blue}  
  
</style>
```

Tag div e tag span

Os estilos CSS estudados até o momento foram aplicados a tags do HTML que já possuem um formato padrão inicial, como por exemplo a tag **b** que deixa o texto em negrito. Porém, existem situações em que precisamos de tags "limpas", sem formatação padrão. Para estes casos, temos a disposição a tag **div** e a tag **span** e as tags de estrutura semântica (article, aside, nav etc).

A tag **div** é muito utilizada na criação de layouts das páginas e áreas para conteúdos. Já a tag **span** é utilizada para aplicar configurações dentro do conteúdo, isso devido ao fato de não possuir nenhum formato pré-definido. Já as tags de estrutura semântica devem ser utilizadas no conteúdo para definir um significado para cada área do conteúdo do site.

Todas as tags aceitam o uso de classes (class) e estilos individuais (id) e também aceitam o uso de ambos os atributos.

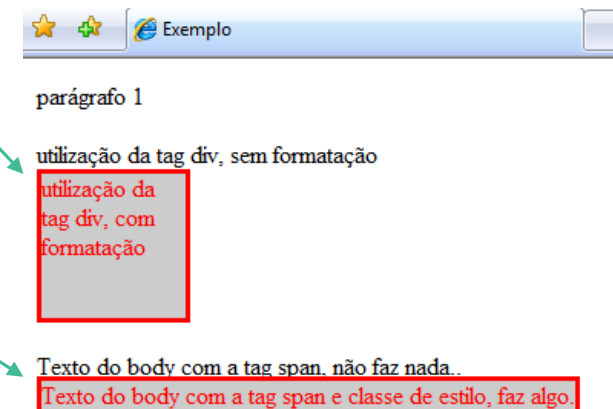
Exemplo

```
<!--exemplo6.html-->
<html>
<head>
<title>Exemplo</title>
<style type="text/css">
  .caixa {
    border: 3px solid #ff0000;
    background-color: #cccccc;
    color: red;
  }
  #aviso {
    width: 100px;
    height: 100px;
  }
</style>
</head>
```

```
<body>
<p> parágrafo 1 </p>
<div> utilização da tag div, sem formatação</div>
<div class="caixa" id="aviso"> utilização da tag div, com
formatação</div>
<br />
<span>Texto do body com a tag span, não faz nada.</span>.
<br />
<span class="caixa">Texto do body com a tag span e classe de estilo, faz
algo.</span>
</body>
</html>
```

Uso dos atributos class e id

Faz uso da classe caixa.



Exemplo

```
<!DOCTYPE HTML><html><head>
<title>Untitled Document</title>
<style type="text/css">
body {
    background-color:#003366;
}

#principal {
    width: 757px;
    margin:0 auto;
}
h2 {
    color:#990000;
}
.data {
    color:#999999;
    font-size:8pt;
}
#topo_geral {
    background-image:url(imagens/topo.JPG);
    height: 62px;
    background-repeat:no-repeat;
}
```

<!--exemplo7.html-->

```
article>p { text-align:justify; }
article {
    background-color:#FFF;
    padding:50px;
}
ul {
    list-style:none;
    padding:0;
    margin:0;
}
li { display:inline; }
nav {
    background-color:#003300;
    color:#FFFFFF;
    padding:5px;
}
</style>
</head>
```

Exemplo

```
<body>
<div id="principal">
  <header id="topo_geral"></header>
  <nav>
    <ul>
      <li>HOME</li> | <li>Notícias</li> | <li>Galeria</li> | <li>Contato</li>
    </ul>
  </nav>
  <article>
    <header id="topo_article">
      <h2>Greve faz Dilma trocar PF por militares na Copa</h2>
      <span class="data">21 de agosto de 2012 | 22h 30</span>
    </header>
    <p>    texto...    </p>
    <p>    texto...    </p>
  </article>
</div>
</body>
```

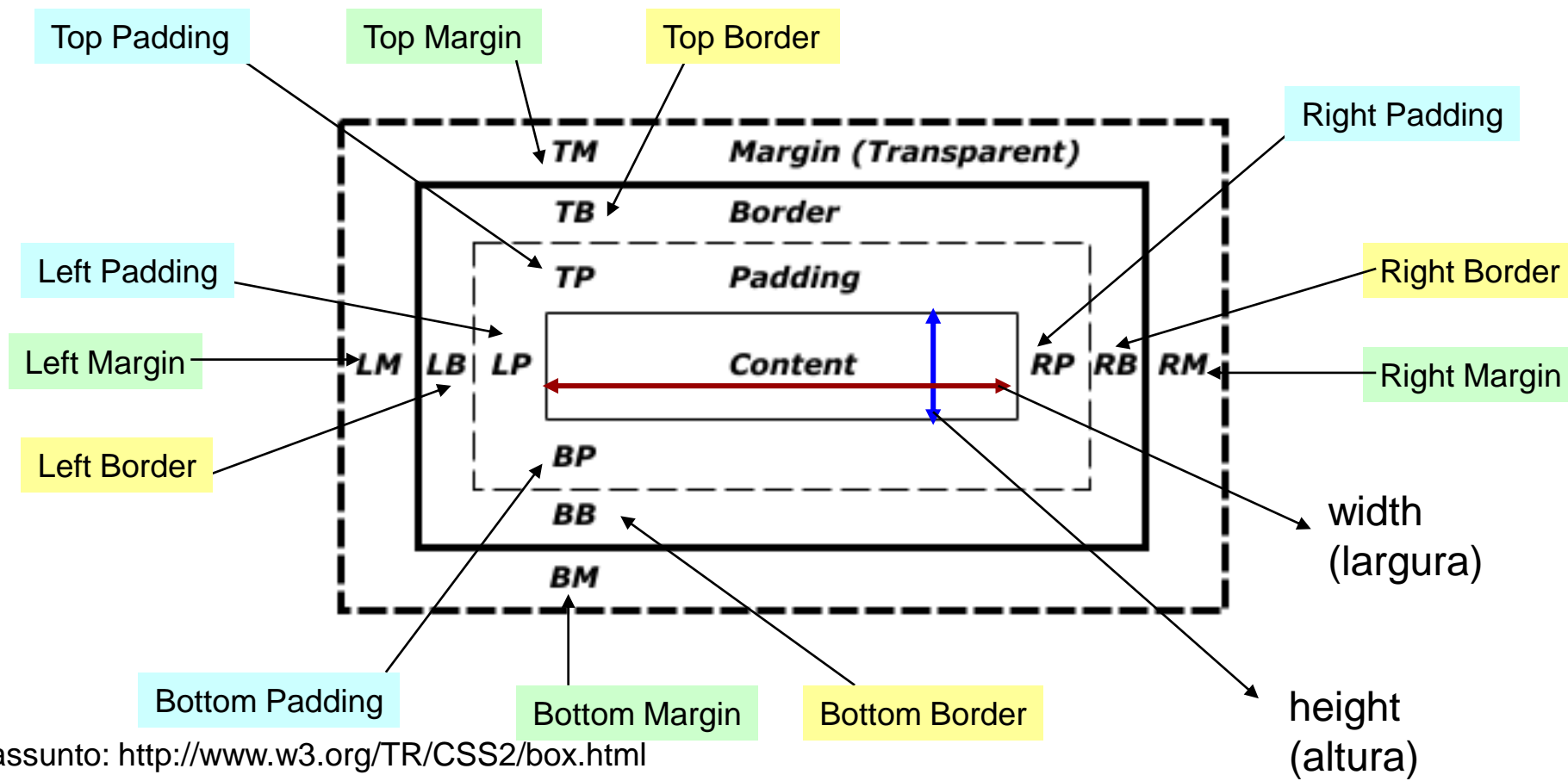
Observações (Box Model CSS)

Os boxes são criados por elementos que formam blocos, como por exemplo temos as tags: p, h1 – h6, div, aside, article, entre outras. Entenda as diferentes regiões de um box.



Observações (Box Model CSS)

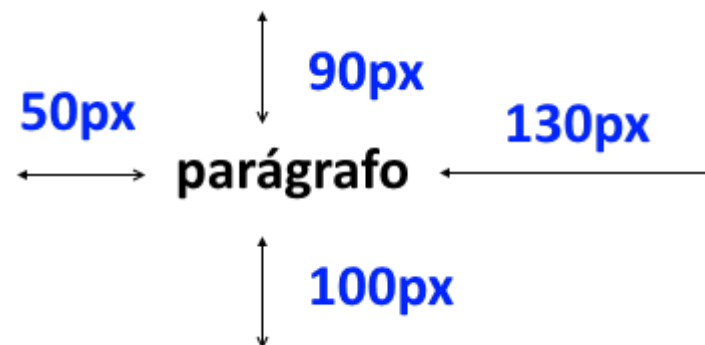
As regiões detalhadas de um box.



Margens com tamanhos individuais para os lados

Exemplo:

```
p {  
  margin-top: 90px;  
  margin-bottom: 100px;  
  margin-right: 130px;  
  margin-left: 50px;  
}
```



Margin – utilizando uma notação simplificada

A propriedade *margin* poderá ter de um a quatro valores.

- **margin: 25px 50px 75px 100px;**
 - margem superior 25px
 - margem direita 50px
 - margem inferior 75px
 - margem esquerda 100px
- **margin: 25px 50px 75px;**
 - margem superior 25px
 - margem direita e esquerda 50px
 - margem inferior 75px
- **margin: 25px 50px;**
 - margem superior e inferior 25px
 - margem direita e esquerda 50px
- **margin: 25px;**
 - todas as margens 25px

Exemplo:

```
p {  
    margin: 70px 50px;  
}
```

Bordas

O atributo **border** poderia especificar, adicionalmente, cor, estilo e largura da borda.

O atributo **border-radius** permite criar bordas arredondadas em elementos do tipo box.

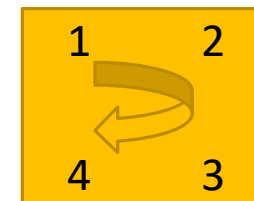
`border-radius` = define tamanho de arredondamento dos cantos

Exemplos: `border-radius: 10px 50px 50px 10px;`

`border-radius: 10px;`

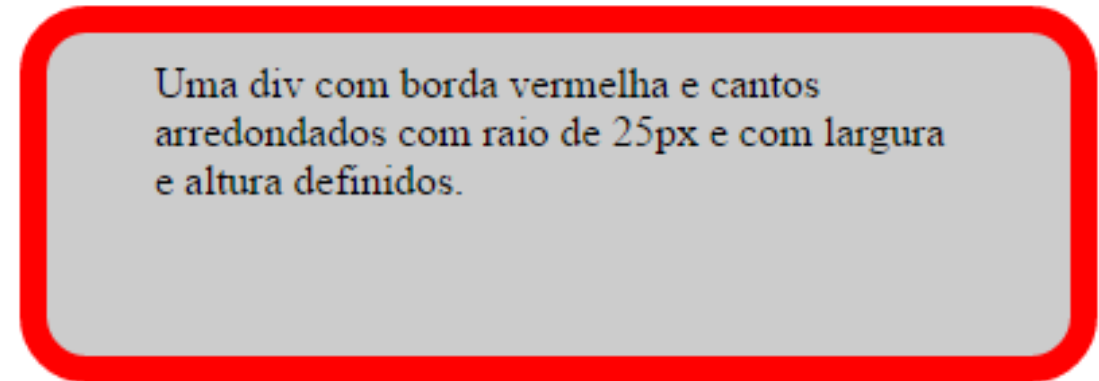
ou podemos definir separadamente o arredondamento de cada esquina:

- `border-top-left-radius: tamanho;`
- `border-top-right-radius: tamanho;`
- `border-bottom-right-radius: tamanho;`
- `border-bottom-left-radius: tamanho;`



Exemplo de div com borda

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Exemplos</title>
  <style>
    div {
      width: 300px;
      height: 100px;
      padding: 10px 40px;
      background: #ccc;
      border: 10px solid #f00;
      border-radius: 25px;
    }
  </style>
</head>
<body>
  <div>Uma div com borda vermelha e cantos
    arredondados com raio de 25px e com
    largura e altura definidos.
  </div>
</body>
</html>
```



Uma div com borda vermelha e cantos arredondados com raio de 25px e com largura e altura definidos.

Uma borda pode ser: solid, dashed, dotted, double, inset, outset,...

Elementos de caixa com sombra

O atributo **box-shadow** adiciona uma sombra em algum box.

box-shadow: h-sombra v-sombra borrão propagação cor ...

h-sombra, v-sombra são as posições da sombra horizontal e vertical (podem ser valores negativos)

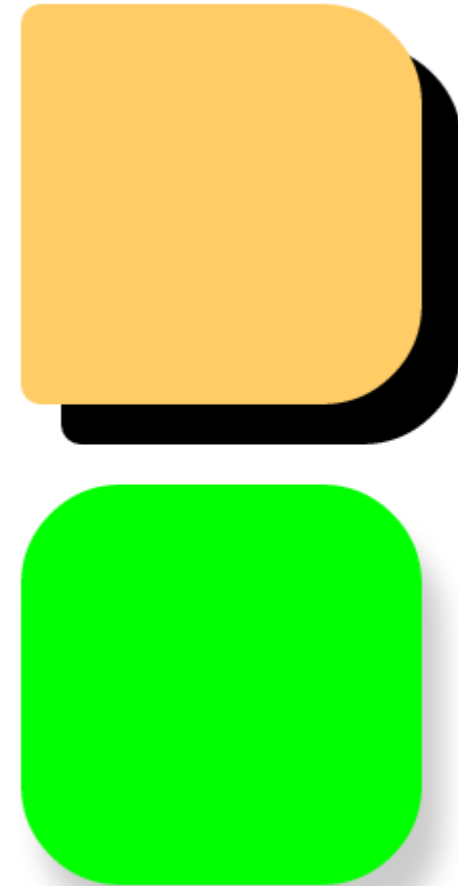
borrão ou *blur* é a distância deste efeito (opcional)

propagação (disseminação) ou *spread* é o tamanho da sombra (opcional)

cor é a cor da sombra (opcional)

Bordas e Sombra (exemplo)

```
<style>
  .borda1 {
    width: 200px;
    height: 200px;
    background-color: #FFCC66;
    border-radius: 10px 50px 50px 10px;
    box-shadow: 20px 20px #000000;
  }
  .borda2 {
    width: 200px;
    height: 200px;
    background-color: #00FF00;
    border-radius: 50px;
    box-shadow: 10px 20px 20px #CCCCCC;
  }
</style>
```



Cores

As cores são exibidas através da combinação **RGB** (vermelho, verde, azul: **Red**, **Green**, **Blue**)

As cores em CSS podem ser especificadas por:

- Valores hexadecimais
- Valores RGB
- Valores RGBA
- Valores HSL
- Valores HSLA
- Predefinidos (nomes)

Cores

Valores hexadecimais

- Combinação dos valores para RGB no formato hexadecimal, o menor valor seria 00 e o maior seria FF (255).
 - Exemplo: #FF0000; (cor vermelha)
#F00; (também cor vermelha)
- Caso a sequência de valores se repita como o exemplo anterior, poderemos utilizar somente três dígitos; para as demais cores devemos utilizar os 6 dígitos
 - Exemplo: #6495ED (tom de azul)

Exercício 1

Recrie a tabela com dados de vendas de aparelhos com os sistemas operacionais de celulares abaixo. Insira os dados e faça a formatação seguindo as cores do modelo.

Period	Android	iOS	Windows Phone	Others
2016Q1	83.4%	15.4%	0.8%	0.4%
2016Q2	87.6%	11.7%	0.4%	0.3%
2016Q3	86.8%	12.5%	0.3%	0.4%
2016Q4	81.4%	18.2%	0.2%	0.2%
2017Q1	85.0%	14.7%	0.1%	0.1%

Source: IDC, May 2017

Exercício 2

Com base nos conceitos passados em aula e os arquivos de referências (HTML e CSS), crie o formulário conforme o modelo ao lado, utilize o CSS externo para formatar o formulário. Procure chegar o mais próximo possível do modelo.

Formulário de Contato

Preencha todos os campos.

Nome :

Email :

Assunto :

Mensagem :

Sua mensagem

Enviar

Exercício 2 (html)

Sugestão de código.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Untitled Document</title>
  <link rel="stylesheet" href="estilo.css" type="text/css" />
</head>
<body>
  <form action="" method="post" class="modelo1">
    <h1>Formulário de Contato<br/>
      <span class="titulo2">Preencha todos os campos.</span>
    </h1>
    <div>
      <div class="rotulos">Nome :</div>
      <input id="nome" type="text" name="nome" placeholder="Nome completo" class="entrada" />
    </div>
    <div>
      <div class="rotulos">Email :</div>
      <input id="email" type="email" name="email" placeholder="E-mail Válido." class="entrada" />
    </div>
    <div>
      <div class="rotulos">Assunto :</div>
      <select name="opcoes" class="entrada">
        <option value=""></option>
        <option value="Dúvidas do produto">Dúvidas do produto</option>
        <option value="Atraso na entrega">Atraso na entrega</option>
      </select>
    </div>
    <div>
      <div class="rotulos">Mensagem :</div>
      <textarea id="mensagem" name="mensagem" placeholder="Sua mensagem" class="entrada"></textarea>
    </div>
    <div>
      <div class="rotulos">&nbsp;</div>
      <input type="button" value="Enviar" class="botao" />
    </div>
  </form>
</body>
</html>
```


Livros sugeridos

Mauricio SAMY Silva. **HTML 5 - A Linguagem de Marcação que revolucionou a WEB.** São Paulo: Novatec, 2011

MAURICIO SAMY SILVA. **Construindo Sites Com Css e (X)Html.** São Paulo: Novatec, 2007.

ERIC FREEMAN; ELISABETH FREEMAN. **Use a Cabeça! Html Com Css e Xhtml.** São Paulo: Alta Books, 2008.

ADOBE CREATIVE TEAM. **Dreamweaver Cs3 - Classroom In a Book.** São Paulo: Artmed, 2008.

Links

<http://www.w3.org/TR/html5-diff/>

<http://www.whatwg.org/specs/web-apps/current-work/#is-this-html5>

http://www.w3schools.com/html/html5_intro.asp

<http://dev.w3.org/html5/spec/>

<http://www.w3.org/International/getting-started/language.pt-br.php?changelang=pt-br>

[http://msdn.microsoft.com/en-us/library/ms533052\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms533052(v=vs.85).aspx)

<http://www.w3c.br/Cursos/CursoCSS3>

<http://www.css3.info/selectors-test/>

Links

<http://fmbip.com/>

http://www.456bereastreet.com/archive/200601/css_3_selectors_explained/

<http://www.findmebyip.com/litmus/>

<http://css3generator.com/>

http://www.quackit.com/css/css_color_codes.cfm

<http://www.w3schools.com/tags/default.asp>

<http://www.w3schools.com/cssref/default.asp>

<http://www.w3schools.com/html/default.asp>

<http://www.w3schools.com/css/default.asp>