

UNIFAI – CENTRO UNIVERSITÁRIO DE ADAMANTINA

DESENVOLVIMENTO FRONT-END PARA WEB I

HTML e CSS

Prof. Welinton Ozelin
welinton.ozelin@fai.com.br

SUMÁRIO

1 INTRODUÇÃO	04
2 HTML	04
2.1 Sintaxe	05
2.2 Estrutura básica	06
2.3 Títulos e subtítulos	07
2.4 Estilos de textos	08
2.5 Listas	09
2.5.1 Listas ordenadas	09
2.5.2 Listas desordenadas	10
2.5.3 Listas por definição	10
2.6 Imagens	11
2.7 Links	11
2.7.1 Links para o mesmo diretório	12
2.7.2 Links para outro diretório	12
2.7.3 Links como índices	12
2.7.4 Link em imagens	13
2.7.8 Link para e-mails	13
2.8 Tabelas	14
2.8.1 Título da tabela (elemento caption)	14
2.8.2 Table row (elemento tr)	14
2.8.3 Table headings (elemento th)	14
2.8.4 Table data (elemento td)	15
2.8.5 Border	15
2.8.6 Align	16
2.8.7 Valign	16
2.8.8 Rowspan e colspan	17
2.8.9 Largura da célula	17
2.9 Formulários	17
2.9.1 Construção de formulários	18
2.9.2 A TAG <input>	18
2.9.2.1 Elemento text	19
2.9.2.2 Elemento password	19

2.9.2.3 Elemento radio	20
2.9.2.4 Elemento checkbox	20
2.9.2.5 Elemento submit	21
2.9.2.6 Elemento reset	21
2.9.3 A TAG <textarea>	22
2.9.4 A TAG <select>	22
2.9.5 Exemplo de formulário completo	24
3 CSS	26
3.1 Sintaxe	26
3.2 Tipos de folhas de estilo	27
3.2.1 Estilos externos	28
3.2.2 Estilos incorporados	28
3.2.3 Estilos inline	29
3.3 Comentários	29
3.4 Classes de estilos	30
3.5 O seletor ID	31
3.6 Utilizando a TAG <div>	31
3.7 Utilizando a TAG 	32
3.8 Propriedades básicas	32
3.8.1 Cores	32
3.8.2 Plano de fundo	33
3.8.3 Texto	34
3.8.4 Fonte	35
3.8.5 Borda	36
3.8.6 Margin e Padding	37
3.8.7 Lista	38
3.8.8 Tabela	39

1 INTRODUÇÃO

Uma das principais dificuldades no desenvolvimento para internet é a mistura dos diversos códigos. Além do HTML, é necessário utilizar o CSS, que é uma linguagem para configurar o visual das páginas e o JavaScript, que é responsável por cuidar do comportamento da página, por exemplo, o que acontece quando o usuário clica em um botão.

Há também as chamadas Linguagens Server-Side (linguagem de servidor), que são linguagens como PHP, Python, Ruby, ASP, etc. Como o próprio nome diz, elas rodam no servidor, realizando os processamentos necessários e retornando uma resposta, que geralmente é um código HTML, para o navegador do usuário.

Na Web, as informações estão dispostas na forma de páginas ligadas entre si. Uma página é transferida de um computador remoto para o usuário, onde o browser faz o trabalho de interpretar os códigos daquele documento e mostrar a página que o usuário vê. A Web está estruturada em dois princípios básicos: HTTP (Hiper Text Transfer Protocol) e HTML (Hiper Text Markup Language).

HTTP é o protocolo de transferência de hipertexto, ou seja, é o protocolo que permite a navegação na Web, com o simples clicar do mouse sobre o texto (ou imagem) que esteja associado a outro link.

2 HTML

O HTML é uma das linguagens utilizadas para o desenvolvimento de websites. O acrônimo HTML vem do inglês e significa Hypertext Markup Language ou em português Linguagem de Marcação de Hipertexto.

Trata-se da linguagem base da internet. Foi desenvolvida para ser de fácil entendimento por seres humanos e também por máquinas, como por exemplo o Google ou outros sistemas que percorrem a internet capturando informação.

Foi criado por Tim Berners-Lee, para a comunicação e disseminação de pesquisas entre ele e seu grupo de colegas. O HTML ficou bastante conhecido quando começou

a ser utilizado para formar a rede pública daquela época, o que se tornaria mais tarde a internet conhecida hoje.

2.1 Sintaxe

O HTML é uma linguagem baseada em marcação. Os elementos são marcados para mostrar quais informações a página exibe. Por exemplo, um título importante (título do artigo, da manchete do site, etc.) pode ser marcado com uma tag/elemento chamada H1, como no exemplo abaixo:

```
<h1>Aqui vai o texto do título</h1>
```

No exemplo acima é possível perceber que o texto está entre duas marcações, que são denominadas TAGs. No início foi aberta a tag com <h1> e fechada no final com </h1>. O que está dentro é o conteúdo mostrado para o usuário.

Geralmente as tags são utilizadas aos pares, no início e no fim, como citado acima, mas também podem ser utilizadas individualmente, como no exemplo abaixo, utilizando para a quebra de linha:

```
<br />
```

Já os parágrafos são marcados com a TAG “p”, como mostrado abaixo:

```
<p>Aqui é inserido o texto do parágrafo, que geralmente possui mais palavras, com fontes menores que os títulos</p>
```

Com a utilização das tags, o navegador é informado sobre o que é cada informação. O que é título, parágrafo, botão, formulário, etc.

Essas informações também são utilizadas por sistemas de busca, como o Google, que, nesse caso, para exibir os resultados de busca, precisa saber o que é um parágrafo e o que é um título. Ele sabe disso por meio das TAGs.

2.2 Estrutura básica

Uma página HTML possui três partes básicas: estrutura principal, cabeçalho e o corpo de página.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="utf-8">
    <title>Título da página</title>
  </head>
  <body>
    Corpo da página
  </body>
</html>
```

A primeira linha se chamada DOCTYPE, e tem a função de avisar aos browsers, robôs de busca, leitores de tela e outras coisas, qual o tipo de documento que eles estão prestes a carregar. Existem outros códigos que podem ser carregados, como XML, por exemplo. Por isso a necessidade do Doctype avisar o browser para que ele saiba como se comportar ao ler o código.

Depois inicia-se com a tag HTML. Isso quer dizer que tudo o que estiver entre as tags <html></html> é escrito em HTML. Ao lado da palavra HTML tem um atributo chamado “lang”, onde é indicado qual o idioma dos textos.

Logo após a tag <html> encontra-se a tag <head>, onde é indicado o título do documento por meio da tag <title>, a tabela de caracteres que o browser deve usar para renderizar o texto e outras informações necessárias para navegador que não são visíveis para o usuário.

O Google e outros sistemas de busca utilizam a tag <title> para indicar em suas buscas o título da página. Isso é muito importante para definir se e como o site aparecerá nas buscas.

Logo após a TAG de fechamento </head> inicia-se a tag <body>. Dentro deste elemento é escrito todo o código HTML do resto do site, que será visualizado pelo usuário.

```

<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="utf-8">
    <title>Título da página</title>
  </head>
  <body>
    <h1>Aqui vai o texto do título</h1>
    <p>Aqui é inserido o texto do parágrafo, que geralmente possui mais
    palavras, com fontes menores que os títulos</p>
  </body>
</html>

```

2.3 Títulos e subtítulos

Em HTML, é possível criar títulos para os documentos utilizando hierarquias, ou seja, criando níveis como título, subtítulo e assim por diante.

A tag que faz isso é o h (que é a letra inicial do inglês heading). Usa-se a tag h com um número logo após, no qual determina a sua importância para o documento. Esse número vai de 1 a 6, sendo 1 o relativamente o mais importante e 6 o menos.

Os browsers tratam visualmente os títulos (headings) modificando seu tamanho. Abaixo é possível observar o código e sua visualização:

```

<h1>Título 1</h1>
<h2>Título 2</h2>
<h3>Título 3</h3>
<h4>Título 4</h4>
<h5>Título 5</h5>
<h6>Título 6</h6>

```

Resultado:

Título 1

Título 2

Título 3

Título 4

Título 5

Título 6

2.4 Estilos de textos

Existem algumas TAGs bastante úteis que permitem aplicar uma formatação a um trecho do texto, apenas adicionando as tags de abertura e fechamento antes e depois do texto a ser formatado.

- ``: marca o texto como em negrito (bold).
- `<i></i>`: marca o texto como em itálico (italics).
- `*<u></u>`: marca um texto como sublinhado (underlined).
- `*<s></s>` ou `<strike></strike>`: marca um texto como riscado.
- ``: marca um texto como subscrito.
- ``: marca o texto como sobrescrito.
- `*<center></center>`: centraliza o texto.

Observação: as tags marcadas com asterisco (*) foram descontinuadas na versão 5 da HTML, ou seja, seu uso não é mais indicado.

```
<b>Texto em negrito</b>  
<i>Texto em itálico</i>  
<u>Texto sublinhado</u>  
<s>Texto riscado</s>  
<sub>Texto como subscrito</sub>  
<sup>texto como sobrescrito</sup>  
<center>Texto centralizado</center>
```

Resultado:

Exemplo: **Texto em negrito**

Exemplo: *Texto em itálico*

Exemplo: Texto sublinhado

Exemplo: ~~Texto riscado~~

Exemplo: _{Texto como subscrito}

Exemplo: ^{texto como sobrescrito}

Exemplo:

Texto centralizado

2.5 Listas

2.5.1 Listas ordenadas

Nas listas ordenadas, como o próprio nome já diz, os itens serão listados de forma ordenada, seja por número, letra, ou algarismos romanos.

Uma lista ordenada começa com a tag “” e seus respectivos itens da lista ficam dentro da tag “”. Por padrão, as listas ordenadas são ordenadas por números, mas também é possível ordená-las por outros métodos, como letras, adicionando o atributo type=”a” ou algarismos romanos, adicionando o atributo type=”I”.

```
<ol>
  <li>Nome</li>
  <li>Telefone</li>
  <li>Endereço</li>
  <li>País</li>
</ol>

<ol type="a">
  <li>Nome</li>
  <li>Telefone</li>
  <li>Endereço</li>
  <li>País</li>
</ol>

<ol type="I">
  <li>Nome</li>
  <li>Telefone</li>
  <li>Casa</li>
  <li>Estado</li>
</ol>
```

Resultado:

1. Nome
2. Telefone
3. Endereço
4. País

- a. Nome
- b. Telefone
- c. Endereço
- d. País

- I. Nome
- II. Telefone
- III. Casa
- IV. Estado

2.5.2 Listas desordenadas

Já as Listas Desordenadas são iniciadas com a tag e são representadas por pequenos “bullets”, conforme o código abaixo:

```
<ul>
  <li>Nome</li>
  <li>Telefone</li>
  <li>Casa</li>
  <li>Estado</li>
</ul>
```

Resultado:

- Nome
- Telefone
- Casa
- Estado

2.5.3 Listas por definição

São constituídas de duas partes: um termo e uma descrição. Para codificar uma lista de definição são necessários três elementos HTML: um container <dl>, um termo de definição <dt> e uma descrição <dd>.

```
<dl>
  <dt>Html</dt>
  <dd>Html Básico</dd>
  <dd>Html Avançado</dd>

  <dt>ASP.NET</dt>
  <dd>ASP.NET Básico</dd>
  <dd>ASP.NET Intermediário</dd>
  <dd>ASP.NET Avançado</dd>
</dl>
```

Resultado:

Html
 Html Basico
 Html Avançado
ASP.NET
 ASP.NET Básico
 ASP.NET Intermediário
 ASP.NET Avançado

2.6 Imagens

As imagens são definidas com a TAG . Essa tag é vazia, o que significa que ela precisa somente de atributos e não tem TAG de fechamento.

Para exibir uma imagem em uma página, é necessário usar o atributo “src”, que significa "source" (fonte). O valor do atributo “src” é a URL da imagem que se pretende exibir na página.

A sintaxe para definir uma imagem é:

```

```

Atributos da tag :

- **src**: URL da imagem.
- **width**: Define a largura da imagem, o que pode ser feito em pixels (número absoluto) ou em percentual da tela (com o símbolo de %).
- **height**: Define a altura da linha, o que pode ser feito em pixels (número absoluto) ou em percentual da tela (com o símbolo de %).
- **align**: Alinhamento da imagem pode ser, **Left**, **Right** e **Center**, ou seja, esquerda, direita e ao centro, respectivamente.
- **alt**: Exibe um texto alternativo caso a imagem não possa ser visualizada.
- **border**: Especificação da largura da borda da imagem. Valor em pixel.
- **vspace**: Para especificar o espaço que deve ser deixado acima e abaixo da imagem. Valor em pixel.
- **hspace**: Especifica o espaço que deve ser deixado nas laterais da imagem. Valor em pixel.

Observação: o único atributo obrigatório é o src.

2.7 Links

O principal poder do HTML vem da capacidade de interligar partes de um texto, imagens a outros documentos.

A interligação entre documentos não se restringe somente às ligações com outras páginas. Em páginas muito longas onde um assunto tem vários tópicos, é possível utilizar índices onde os links têm a função de interligar os tópicos de um texto e que com apenas um clique em um dos tópicos do índice, o item é exibido.

2.7.1 Links para o mesmo diretório

Basta especificar o nome do arquivo que será chamado e a sua extensão:

```
<a href="Default.aspx">Home</a>
```

Onde:

- **<a>**: abertura da TAG de link;
- **href="NomeDoArquivo.extensão"**: deve ser informado o nome completo do arquivo que será acessado;
- **Texto ou imagem**: que servirá como link;
- ****: encerramento da TAG de link.

2.7.2 Links para outro diretório

Para criar links para uma página localizada em outros diretórios é necessário indicar o caminho completo do arquivo. Para a Web, isto tem uma forma um pouco diferente do Windows, conforme explicado abaixo:

- A barra utilizada para separar os diretórios é a barra convencional (/);
- O ponto de partida para localizar um arquivo em outro diretório é o atual;
- Para baixar um nível deve utilizar os sinais "../".

```
<a href="../Cadastros/Alunos.aspx">Alunos</a>
```

2.7.3 Links como índices

Quando se tem uma página muito extensa, é interessante a utilização de links que funcionem como índices, e direcione o usuário para pontos distintos dentro da mesma página, facilitando a chegada a pontos específicos.

Para isso, basta criar uma tag <a>, com um atributo chamado “id”, definindo o ponto de destino:

```
<a id="Topico1">Tópico 1</a>
```

Em seguida, resta apenas criar o link que direcionará o usuário para este ponto, adicionando o caractere “#”, no atributo “href”, seguindo do texto inserido no atributo “id” do ponto de destino:

```
<a href="#Topico1">Tópico 1</a>
```

2.7.4 Link em imagens

Para colocar links em imagens ocorre basicamente da mesma forma que com texto. Bastando inserir uma TAG de imagem dentro da TAG de link, conforme o exemplo:

```
<a href="Default.aspx"></a>
```

2.7.8 Link para e-mails

Com HTML é possível criar um link que faça a abertura do cliente de e-mail do usuário já com algumas informações preenchidas, que facilitem o envio de e-mails do usuário para um endereço configurado. Para isso basta utilizar a TAG <a>, adicionando o endereço de e-mail no atributo “href”, precedido do comando “mailto:”, conforme o exemplo:

```
<a href="mailto:exemplo@exemplo.com">Enviar e-mail</a>
```

É possível adicionar um assunto ao e-mail que será enviado, adicionado o comando “subject=” na URL:

```
<a href="mailto:exemplo@exemplo.com?subject=Assunto do e-mail">Enviar e-mail</a>
```

Já para adicionar um texto no corpo da mensagem, utiliza-se “body=”:

```
<a href="mailto:exemplo@exemplo.com?body=Exemplo de texto">Enviar e-mail</a>
```

E para combinar todas as opções que permitem que o visitante abra o cliente de e-mail já com o endereço, assunto e texto introduzido, utiliza-se da seguinte forma:

```
<a href="mailto:exemplo@exemplo.com?subject=Assunto do e-mail&body=Exemplo de texto">Enviar e-mail</a>
```

2.8 Tabelas

O uso de tabelas era muito comum há alguns anos para a definição de áreas do site. Seu uso para essa finalidade acabou se tornando prejudicial pela complexidade da marcação, o que dificulta bastante a manutenção das páginas. Além disso havia uma implicação direta na definição de relevância do conteúdo das tabelas para os indexadores de conteúdo por mecanismos de busca. Entretanto, ainda hoje, quando é necessário exibir uma série de dados tabulares, é indicado o uso da tag de tabela.

A estrutura e o conteúdo da tabela devem ficar dentro das TAGs <table></table>.

2.8.1 Título da tabela (elemento caption)

A TAG <caption> especifica o título de uma tabela. Por exemplo:

```
<caption>Título da tabela</caption>
```

2.8.2 Table row (elemento tr)

A TAG <tr> indica o início de uma linha na tabela. Cada linha da tabela pode conter várias células, e, portanto, é necessário que se faça uso de uma marcação que indique exatamente o ponto de quebra de uma linha e início de outra. Toda linha deve terminar com um </tr>.

2.8.3 Table headings (elemento th)

A TAG <th> é usada para especificar as células de cabeçalho da tabela. Essas células são diferentes das outras, pois seu conteúdo aparece geralmente em negrito. O elemento “th” pode ser apresentado sem conteúdo algum: isso corresponde a uma

célula em branco. As tabelas podem ainda conter mais de um “th” para uma dada coluna, ou linha, ou simplesmente não conter nenhum elemento “th”, isto é, não conter em nenhuma célula em destaque.

2.8.4 Table data (elemento td)

A TAG <td> especifica a células de dados de uma tabela. Por se tratar de dados comuns (e não cabeçalhos), essas células possuem seu conteúdo escrito em fonte normal, sem nenhum destaque e alinhamento à esquerda. Assim como o “th”, pode-se construir células em branco, usando o elemento “td”.

Exemplo de tabela com os elementos citados acima:

```
<table>
  <tr>
    <th>Título da primeira coluna</th>
    <th>Título da segunda coluna</th>
  </tr>
  <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>
```

2.8.5 Border

O “**border**” é um atributo opcional para ser usado com “**table**”, que quanto está presente, a tabela é formatada com linhas de borda.

Esse atributo recebe um valor que vai estabelecer qual a espessura (além da existência) da linha de borda da tabela (**border="valor"**). Se o valor atribuído for 0 (zero), o “**border**” funciona exatamente como o caso padrão, sem o “**border**”. Dessa maneira, é possível colocar tabelas em maior destaque, atribuindo um valor maior que 1.

Exemplo:

```
<table border="1">
  <tr>
    <td>Linha 1, coluna 1.</td>
    <td>Linha 1, coluna 2.</td>
  </tr>
</table>
```

2.8.6 Align

Este atributo pode ser aplicado a “th”, “td” ou “tr”, e faz controle do alinhamento do texto dentro de uma célula, com relação as bordas laterais. Quando aplicado a “tr”, ele define o alinhamento de toda uma linha da tabela.

O exemplo abaixo mostra como o “**align**” aceita os valores **left**, **center** ou **right**, para alinhar à esquerda, centralizar ou alinhar à direita, respectivamente.

```
<table border="1">
  <tr>
    <td align="center">Centro</td>
    <td align="left">Esquerda</td>
    <td align="right">Direita</td>
  </tr>
</table>
```

2.8.7 Valign

Pode ser aplicado a “th” e “td” e define o alinhamento do texto em relação às bordas superior e inferior.

Aceita os valores top, middle, e bottom para alinhar na parte de cima, no meio e na parte de baixo, respectivamente.

Exemplo:

```
<table border="1">
  <tr>
    <td valign="top">top</td>
    <td valign="middle">middle</td>
    <td valign="bottom">bottom</td>
  </tr>
</table>
```


2.8.8 Rowspan e colspan

O “rowspan” define quantas linhas uma mesma célula pode abranger. Por padrão, na maioria dos navegadores, cada célula adicionada em uma tabela corresponde a uma linha. Pode ser aplicado em “th” ou “td”, proporcionando o mesmo efeito.

Já o “colspan” define quantas colunas uma célula pode abranger. Por padrão, na maioria dos navegadores, cada célula adicionada em uma tabela corresponde a uma coluna. Pode ser aplicado em “th” ou “td”, proporcionando a mesma abrangência.

Exemplo:

```
<table border="1">
  <tr>
    <td rowspan="2">Linha 1 e linha 2 mescladas.</td>
    <td colspan="2">Coluna 2 e coluna 3 mescladas.</td>
  </tr>
  <tr>
    <td>Linha 2, coluna 2.</td>
    <td>Linha 2, coluna 3.</td>
  </tr>
</table>
```

2.8.9 Largura da célula

Para alterar a largura de uma célula da tabela basta acrescentar o parâmetro **width** dentro da tag <td>.

Exemplo:

```
<table border="1">
  <tr>
    <td width="100">width = 100 (pixels)</td>
    <td align="middle" width="200">width = 200 (pixels)</td>
  </tr>
</table>
```

2.9 Formulários

A linguagem HTML permite que o cliente (navegador) interaja com o servidor, preenchendo campos, clicando em botões e passando informações.

O elemento “**form**”, da linguagem HTML, é justamente o responsável por tal interação. Ele provê uma maneira agradável e familiar para coletar dados do usuário através da criação de formulários com janelas de entrada de textos, botões, etc.

2.9.1 Construção de formulários

Para a construção de um formulário, todos os elementos que serão apresentados nos tópicos abaixo deverão ficar dentro das TAGs **<form>** **</form>**.

Dentro da TAG “**form**” deve ser inserido o atributo “**method**”, que é responsável por definir a forma como os dados serão enviados após salvar um formulário. Ele pode assumir um dos dois valores abaixo:

- **GET** – Envia os dados adicionados aos campos do formulário associados à URL da página, suportando até 128 caracteres.
- **POST** – É o mais utilizado, pois envia cada dado de forma separada da URL. Com este método, as informações adicionadas aos campos fazem parte do corpo da mensagem enviada para o servidor e transfere grande quantidade de dados.

Também podem ser adicionados os atributos “**id**” e “**name**”, que podem ter os mesmos valores e servem para identificar o formulário na página e no servidor, respectivamente.

2.9.2 A TAG **<input>**

A TAG **<input>** especifica uma variedade de campos editáveis dentro de um formulário, podendo receber diversos atributos que definem o tipo de mecanismo de entrada (botões, caixas de texto, etc.), o nome do campo, o alinhamento e o campo do valor mostrado.

Um dos atributos mais importantes do “**input**” é o “**name**”. Ele associa o valor da entrada do elemento a um nome, para que este possa ser usado pela linguagem de programação do lado do servidor.

Outro atributo importante é o “**type**”, que determina o tipo de campo de entradas de dados. Por exemplo:

```
<input type="text" name="txtNome" />
```

Para mudar o comprimento do campo, pode ser utilizado o atributo “**size**”, conforme abaixo:

```
<input type="text" name="txtNome" size="10" />  
<!-- 10 é um exemplo, podendo ser substituído pelo valor desejado -->
```

Já o atributo “**value**” adiciona ao campo um valor de preenchimento:

```
<input type="text" name="txtNome" size="10" value="Exemplo de conteúdo" />
```

2.9.2.1 Elemento text

Conforme mostrado nos exemplos do tópico anterior, o **type=”text”** consiste em um campo de texto comum, para a entrada de dados por parte do usuário.

Resultado:

2.9.2.2 Elemento password

Esse elemento tem funcionamento semelhante ao anterior, com o diferencial de ocultar da tela o que é digitado nele, evitando assim que campos sigilosos como senhas, sejam exibidos. Por exemplo:

```
<input type="password" name="txtSenha" size="50" value="" />
```

Resultado:

2.9.2.3 Elemento radio

Quando o usuário deve escolher uma única alternativa em um conjunto de várias opções, utiliza-se o radio button.

Exemplos típicos do uso desse botão é um campo que deve ter como resposta “Sim” ou “Não”; e um campo para a seleção do sexo, que aceitaria “Masculino” ou “Feminino”.

Uma característica importante desse campo é a necessidade de todos os radios buttons de um mesmo grupo, ou seja, referentes à mesma pergunta, terem o mesmo atributo “**name**”. Também é necessário informar um valor para o atributo “**value**”, e um identificador único para o atributo “**id**”, conforme o exemplo abaixo:

```
<input type="radio" id="rdbSexoMasculino" name="rdbSexo" value="M"> Masculino  
<input type="radio" id="rdbSexoFeminino" name="rdbSexo" value="F"> Feminino
```

Resultado:

☐ Masculino ☐ Feminino

Para que um dos valores já venha selecionado basta utilizar o atributo **checked="checked"**, como abaixo, onde o item “Feminino” virá preenchido:

```
<input type="radio" id="rdbSexoMasculino" name="rdbSexo" value="M"> Masculino  
<input type="radio" id="rdbSexoFeminino" name="rdbSexo" checked="checked"  
value="F"> Feminino
```

Resultado:

☐ Masculino ☒ Feminino

2.9.2.4 Elemento checkbox

Diferente do radio button, o checkbox permite que mais de uma opção seja escolhida como resposta para o campo.

Lembrando que assim como o radio button, o elemento checkbox necessita que os itens integrantes de um mesmo grupo tenham o mesmo atributo “**name**”.

Exemplo:

```
<input type="checkbox" id="chkLinguagemHTML" name="chkLinguagens" />  
HTML  
<input type="checkbox" id="chkLinguagemCSS" name="chkLinguagens" />  
CSS  
<input type="checkbox" id="chkLinguagemJavaScript" name="chkLinguagens" />  
JavaScript  
<input type="checkbox" id="chkLinguagemAspNet" name="chkLinguagens" />  
ASP.NET  
<input type="checkbox" id="chkLinguagemAspNetMVC" name="chkLinguagens" />  
ASP.NET MVC  
<input type="checkbox" id="chkLinguagemPHP" name="chkLinguagens" />  
PHP
```

Resultado:

☐ HTML ☐ CSS ☐ JavaScript ☐ ASP.NET ☐ ASP.NET MVC ☐ PHP

Para trazer itens selecionados basta apenas também utilizar o **checked="checked"**, como no tópico anterior.

2.9.2.5 Elemento submit

O “**submit**” é um botão, e sua função é submeter os dados do formulário para o script que os tratará. Exemplo:

```
<input type="submit" id="btnSalvar" name="btnSalvar" value="Salvar" />
```

Resultado:

Salvar

2.9.2.6 Elemento reset

Já o **reset** tem a função de limpar todos os campos do formulário. Exemplo:

```
<input type="reset" id="btnLimpar" name="btnLimpar" value="Limpar" />
```

Resultado:

Limpar

2.9.3 A TAG <textarea>


Trata-se de um campo para entrada de texto, que permite várias linhas de conteúdo.

Para definir o tamanho do campo mostrado na tela, usa-se os atributos “**cols**” e “**rows**”, que especificam, respectivamente, o número de colunas e linhas que se deseja mostrar para o usuário.

O atributo não é aceito neste elemento, mas é possível iniciar o elemento já com um texto preenchido, inserindo entre as TAGs de abertura de fechamento do “**textarea**”, conforme o exemplo abaixo:

```
<textarea id="txtObservacoes" name="txtObservacoes" cols="50" rows="5">Exemplo de conteúdo</textarea>
```

Resultado:



2.9.4 A TAG <select>

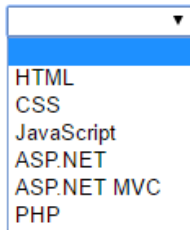
A TAG <select> permite definir uma lista de opções, com barra de rolagem ou fixa na tela do navegador, para que o usuário selecione um item.

Dentro da TAG <select> devem ser inseridas as TAGs <option>, que são os itens da lista, e dentro destas devem ser inseridos os atributos “value”, que servem para denotar o valor do item selecionado.

Exemplo:

```
<select id="cmbLinguagens" name="cmbLinguagens">
  <option value=""></option>
  <option value="1">HTML</option>
  <option value="2">CSS</option>
  <option value="3">JavaScript</option>
  <option value="4">ASP.NET</option>
  <option value="5">ASP.NET MVC</option>
  <option value="6">PHP</option>
</select>
```

Resultado:

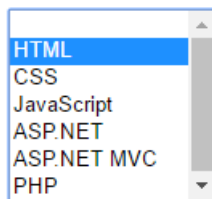


Dessa forma o campo vem fechado, e é aberto para a seleção quando o usuário clicar no mesmo. Para que ele venha aberto por padrão, utiliza-se o atributo **multiple="multiple"** na TAG <select>, conforme mostrado abaixo:

Exemplo:

```
<select id="cmbLinguagens" name="cmbLinguagens" multiple="multiple">
  <option value=""></option>
  <option value="1">HTML</option>
  <option value="2">CSS</option>
  <option value="3">JavaScript</option>
  <option value="4">ASP.NET</option>
  <option value="5">ASP.NET MVC</option>
  <option value="6">PHP</option>
</select>
```

Resultado:



Já para que um dos itens venha selecionado ao abrir a página, utiliza-se o atributo **selected="selected"** na TAG <option>. Exemplo:

```
<select id="cmbLinguagens" name="cmbLinguagens">
  <option value=""></option>
  <option value="1">HTML</option>
  <option value="2">CSS</option>
  <option value="3">JavaScript</option>
  <option value="4" selected="selected">ASP.NET</option>
  <option value="5">ASP.NET MVC</option>
  <option value="6">PHP</option>
</select>
```

Resultado:

ASP.NET ▼

2.9.5 Exemplo de formulário completo

Abaixo é apresentado um modelo de formulário completo, utilizando os campos apresentados nos tópicos acima e organizados dentro de uma tabela para que fiquem alinhados.

Também é interessante observar que os campos de radio e checkbox, foram colocados dentro de TAGs <label></label>, isso serve para que o texto que acompanha o campo radio ou checkbox possa ser usado para clicar e selecionar o item.

Código-fonte:

```
<form name="frmExemplo" method="post">
  <table>
    <tr>
      <td>Nome: </td>
      <td>
        <input type="text" id="txtNome" name="txtNome" size="50" value="" />
      </td>
    </tr>
    <tr>
      <td>Endereço: </td>
      <td>
        <input type="text" id="txtEndereco" name="txtEndereco" size="50"
value="" />
      </td>
    </tr>
    <tr>
      <td>Cidade:</td>
      <td>
        <select id="cmbCidade" name="cmbCidade">
          <option value=""></option>
          <option value="">Adamantina</option>
          <option value="">Bastos</option>
          <option value="">Dracena</option>
          <option value="">Flórida Paulista</option>
          <option value="">Inúbia Paulista</option>
          <option value="">Junquirópolis</option>
          <option value="">Lucélia</option>
          <option value="">Osvaldo Cruz</option>
          <option value="">Pacaembu</option>
          <option value="">Parapuã</option>
          <option value="">Tupã</option>
        </select>
      </td>
    </tr>
  </table>
</form>
```



```

<tr>
  <td>Sexo: </td>
  <td>
    <label>
      <input type="radio" id="rdbSexoMasculino" name="rdbSexo" value="M" />
      Masculino
    </label>

    <label>
      <input type="radio" id="rdbSexoFeminino" name="rdbSexo"
checked="checked" value="F" />
      Feminino
    </label>
  </td>
</tr>
<tr>
  <td>Linguagens dominadas:</td>
  <td>
    <label>
      <input type="checkbox" id="chkLinguagemHTML" name="chkLinguagens" />
      HTML
    </label>
    <label>
      <input type="checkbox" id="chkLinguagemCSS" name="chkLinguagens" />
      CSS
    </label>
    <label>
      <input type="checkbox" id="chkLinguagemJavaScript"
name="chkLinguagens" />
      JavaScript
    </label>
    <label>
      <input type="checkbox" id="chkLinguagemAspNet"
name="chkLinguagens" />
      ASP.NET
    </label>
    <label>
      <input type="checkbox" id="chkLinguagemAspNetMVC"
name="chkLinguagens" />
      ASP.NET MVC
    </label>
    <label>
      <input type="checkbox" id="chkLinguagemPHP" name="chkLinguagens" />
      PHP
    </label>
  </td>
</tr>
<tr>
  <td colspan="2" align="center">
    <input type="submit" id="btnSalvar" name="btnSalvar" value="Salvar" />
    <input type="reset" id="btnLimpar" name="btnLimpar" value="Limpar" />
  </td>
</tr>
</table>
</form>

```

Resultado:

Nome:

Endereço:

Cidade:

Sexo: ☐ Masculino ☒ Feminino

Linguagens dominadas: ☐ HTML ☐ CSS ☐ JavaScript ☐ ASP.NET ☐ ASP.NET MVC ☐ PHP

3 CSS

Quando não se define uma formatação específica para os elementos HTML de uma página WEB, eles são exibidos com o estilo determinado pelo navegador utilizado. Há dois problemas principais em deixar os navegadores decidirem qual formatação deve ser adotada. O primeiro é que cada navegador pode adotar uma formatação diferente. Consequentemente, uma mesma página WEB não será exibida exatamente da mesma forma em todos os navegadores. O segundo problema é que as formatações adotadas pelos principais navegadores não são bonitas e agradáveis.

É possível padronizar a forma que as páginas WEB são exibidas nos diferentes navegadores e obter um visual agradável definindo uma própria formatação. O estilo das páginas deve ser definido com a linguagem CSS (Cascading Style Sheets ou Folha de Estilo em Cascata).

Dentre as vantagens do CSS, pode-se destacar as seguintes:

- Economia de tempo;
- Diminuição do tamanho do código da página;
- Páginas carregadas com maior rapidez;
- Maior facilidade para manter e fazer alterações na página;
- Maior controle do layout da página.

3.1 Sintaxe

Cada estilo criado é definido como uma regra CSS, e cada regra deve utilizar a seguinte sintaxe:

elemento { atributo1: valor; atributo2: valor; }

Onde:

- **Elemento** - Descreve o elemento de design ao qual o estilo será aplicado. A mesma tag HTML, mas sem os sinais de maior e menor. Essa parte da regra é, às vezes, chamada de seletor.
- **Atributo** - O aspecto específico do elemento que você quer usar como estilo. Deve ser um nome de atributo CSS válido, como o atributo **font-size**, por exemplo.
- **Valor** - A configuração aplicada ao atributo. Deve ser uma configuração válida para o atributo em questão, como 12px (12 pixels) para **font-size**.
- **Atributo: valor** – É a declaração da regra. Pode-se atribuir múltiplas declarações e separá-las com ponto-e-vírgula (;).

Como exemplificação, abaixo é apresentada uma regra CSS que especifica que todos os títulos de nível 1 (tags <h1>) sejam exibidos em uma fonte de 24 pixels:

```
h1 {  
    font-size: 36px;  
}
```

Já no caso abaixo, todos os títulos de nível 2 (tags <h2>) devem ter tamanho de 24 pixels e cor azul:

```
h2 {  
    font-size: 24px;  
    color: blue;  
}
```

3.2 Tipos de folhas de estilo

É possível definir regras de CSS em três lugares. E, por definição, pode-se utilizar uma combinação dos três métodos. A maneira como as regras interagem entre si está relacionada à parte "em cascata". Os três lugares são:

- A. Em um documento separado, fora de todos os documentos HTML;
- B. No cabeçalho de um documento HTML;
- C. Dentro de uma tag de HTML.

Cada um destes métodos tem um nome e afeta as páginas HTML do site de um modo diferente, como será apresentado nos tópicos seguintes.

3.2.1 Estilos externos

Estilo externo significa que as regras de CSS são colocadas em um arquivo separado, e então a página HTML pode fazer um link para esse arquivo. Essa abordagem permite definir regras em um ou mais arquivos que podem ser aplicadas em uma ou mais páginas do site.

Para definir um conjunto de regras de estilo externo, basta criar um arquivo texto, dar um nome e salvar com a extensão **“.css”**.

E para utilizar um arquivo externo, basta colocar a TAG `<link>` no cabeçalho do HTML que referencie um arquivo **“.css”**, conforme o exemplo abaixo:

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

Observação: o texto acima deve ser inserido entre as TAGs `<head>` `</head>`, atentando-se para o endereço do arquivo no atributo **“href”**.

3.2.2 Estilos incorporados

Em um estilo incorporado significa que as regras de CSS são especificadas no cabeçalho do documento. Essas regras afetam somente a página atual.

Para criar um conjunto de estilos que se aplicam a uma única página, pode-se configurar os estilos exatamente da forma como seriam configuradas em um arquivo externo, com o diferencial que desta vez serão adicionados dentro de TAGs `<style>` `</style>`, que devem ficar dentro das TAGs `<head>` `</head>`.

Exemplo:

```

<head>
  <title>Exemplo</title>
  <style type="text/css">
    p {
      font-family: "Times New Roman", Times, serif;
      font-size: 12px;
    }
  </style>
</head>

<body>
</body>
</html>

```

No caso acima é apresentado um estilo incorporado a uma página HTML, onde é configurado a fonte e o tamanho da fonte das TAGs <p>.

3.2.3 Estilos inline

Já um estilo inline significa que as regras de CSS são especificadas dentro da TAG de HTML. Essas regras afetam somente a TAG atual.

Esses estilos são os que têm menos efeitos. Eles afetam somente a TAG atual - não outras TAGs na página e tampouco outros documentos. A sintaxe para definir um estilo inline é apresentada com o seguinte exemplo:

```

<a href="http://www.fai.com.br" style="color: green; text-decoration:
none;">Exemplo</a>

```

No caso acima é possível notar que ao invés das tags <style> </style>, apenas foi utilizado um atributo “**style**” dentro da TAG para definir o estilo. E, ao invés de colocar as regras de CSS entre chaves, elas foram colocadas entre aspas, separando-as com ponto-e-vírgula como de costume.

3.3 Comentários

Assim como em outras linguagens, os comentários são utilizados para explicar o código, sendo, portanto, essenciais para ajudar o desenvolvedor a achar erros ou modificar áreas específicas do site.

O navegador irá ignorar linhas comentadas, logo, elas não farão parte da sua formatação e não influenciarão no tempo de execução do site. Em CSS, as linhas comentadas começam com "/*" e terminam com "*/", conforme o exemplo abaixo:

```
/* Mudando a cor do plano de fundo */  
background-color: #ccc;
```

3.4 Classes de estilos

Com as classes de estilo, é possível definir diversas variações de uma única TAG. Por exemplo, é possível criar um estilo de parágrafo com o texto alinhado à direita, um estilo de parágrafo com o texto centralizado, e assim por diante, criando múltiplos temas em torno da mesma TAG de parágrafo (<p>).

Tais classes de estilo podem ser definidas tanto em folhas de estilo externa como nas incorporadas, ressaltando que não haveria sentido em definir uma classe em um estilo inline. A sintaxe é praticamente idêntica para os estilos externo e incorporado, com a adição de um ponto e o nome da classe.

Sintaxe:

```
.NomeDaClasse { atributo: valor; }
```

Exemplo:

```
/* CSS */  
.AlinhadoEsquerda {  
    text-align: left;  
}  
  
.AlinhadoCentro {  
    text-align: center;  
}  
  
.AlinhadoDireita {  
    text-align: right;  
}
```

```
<!-- HTML -->  
<p class="AlinhadoEsquerda">Exemplo 1</p>  
<p class="AlinhadoCentro">Exemplo 2</p>  
<p class="AlinhadoDireita">Exemplo 3</p>
```

Resultado:

Exemplo 1

Exemplo 2

Exemplo 3

3.5 O seletor ID

Diferente de uma classe de estilos, que pode ser utilizada múltiplas vezes na mesma página, o ID deve ser único. Um seletor ID de determinado nome só pode ser aplicado a UM e somente UM elemento HTML dentro do documento.

O ID deve receber um nome, que deverá ser único, e para ser utilizado no CSS deverá ter esse nome precedido pelo sinal de “#”, conforme o exemplo:

Sintaxe:

```
#NomeID { atributo: valor; }
```

Exemplo:

```
/* CSS */
#ExemploDeID {
  /* Negrito */
  font-weight: bold;
}
```

```
<!-- HTML -->
<p id="ExemploDeID">Exemplo de ID</p>
```

3.6 Utilizando a TAG <div>

As TAGs HTML <div> </div> podem ser usadas para formatar um grande bloco de texto - uma divisão - abrangendo diversos parágrafos e outros elementos. Isso as torna uma boa opção para definir estilos que afetam grandes seções de um texto em uma página.

Exemplo:

```
/* CSS */
.Disciplinas {
  /* Sublinhado */
  text-decoration: underline;
}
```

```
<!-- HTML -->
<div class="Diciplinas">Matemática</div>
<div class="Diciplinas">Português</div>
<div class="Diciplinas">Geografia</div>
```

Ao colocar na tag <div> o atributo “class”, todos os elementos que estejam englobados nesta TAG seguiram estes padrões.

3.7 Utilizando a TAG

As TAGs são como as TAGs <div>...</div> no sentido de que se pode utilizá-las para definir estilos que formatam um bloco de texto. Ao contrário de <div>, contudo, que é utilizada para divisões de texto grandes, a tag é especializada para blocos de textos menores, que podem ser tão pequenos como um único caractere.

Exemplo:

```
/* CSS */
.TextoVermelho {
    color: #F00;
}
```

```
<!-- HTML -->
<div><span class="TextoVermelho">Exemplo</span> de SPAN</div>
```

Resultado:

Exemplo de SPAN

3.8 Propriedades básicas

3.8.1 Cores

Esta é uma das propriedades mais comum, pois se aplica a muitos seletores. As cores no CSS obedecem ao padrão hexadecimal RGB e as mais básicas podem também ser designadas pelo nome.

Exemplos:

```
p {  
  color: blue; /* Nome da cor */  
}  
  
p {  
  color: #0026ff; /* Hexadecimal */  
}
```

3.8.2 Plano de fundo

O corpo de um documento geralmente pode vir preenchido por uma cor ou figura. A partir da propriedade "**background**", não só ele, mas como qualquer outro elemento pode ter o plano de fundo modificado.

Propriedades:

- **background-color**: Preencherá o documento com a cor desejada.
- **background-image**: O valor atribuído a esta propriedade, deverá ser o nome do arquivo da figura, que deverá estar localizada na mesma pasta da folha de estilos ou com o caminho até a imagem especificado. Quando o plano de fundo é preenchido por uma imagem, surgem mais propriedades a serem especificadas.
- **background-position**: A posição da imagem na página. Top, center ou bottom combinados com left, center e right.
- **background-repeat**: Opção usada para repetir a imagem pelo plano de fundo.

Exemplo:

```
body {  
  background-color: #FFFFFF;  
  background-image: url('ImagemFundo.png');  
  background-position: bottom right;  
  background-repeat: no-repeat;  
}
```

Para diminuir a quantidade de código também é possível especificar todas as propriedades acima em uma única propriedade **background**, conforme o exemplo a seguir:

```
body {  
  background: #ffffff url("ImagemFundo.png") no-repeat right top;  
}
```

3.8.3 Texto

O texto pode ter características alteradas em CSS que não poderiam ser alteradas em HTML. Podendo citar como exemplo, o espaçamento entre as linhas.

Com o uso do CSS e de suas propriedades, pode-se caracterizar textos em qualquer elemento do HTML.

Propriedades:

- **color:** Define a cor do texto.
- **text-indent:** Define a distância de recuo do texto no início do parágrafo.
- **line-height:** Define o espaçamento entre as linhas.
- **text-align:** Define o alinhamento do texto, que pode ser ao centro, à direita, à esquerda ou no estilo justificado.
- **text-decoration:** Define a decoração de um texto e é feita com os seguintes valores:
 - **underline:** sublinhado
 - **overline:** sobrelinhado
 - **line-through:** uma linha em cima do texto
 - **blink:** faz piscar o texto
- **text-transform:** Define uma transformação ao texto, que podem ser as seguintes:
 - **uppercase:** torná-las todas maiúsculas
 - **lowercase:** todas minúsculas;
 - **capitalize:** todas as primeiras letras maiúsculas

Exemplo:

```
h1 {  
  color: #DDA0DD;  
  text-decoration: underline;  
  text-transform: uppercase;  
}  
  
h2 {  
  color: #3366FF;  
  text-decoration: line-through;  
  text-transform: none;  
}  
  
p {  
  text-indent: 1cm;  
  line-height: 2px;  
  text-align: center;  
}
```

3.8.4 Fonte

Muitas das funções apresentadas nestes tópicos são bem específicas e impossíveis de serem aplicadas pelo HTML.

Neste tópico, assim como no anterior, todas as propriedades são aplicadas a seletores relacionados a textos, como <p> e <h1>.

Propriedades:

- **font-family:** Refere-se à família da fonte. O valor pode ter o nome específico da fonte (Arial, Verdana, Times New Roman) ou de fontes genéricas (monospace, serif). Vale ressaltar que a fonte escolhida deverá estar instalada na máquina do usuário.
- **font-size:** Pode-se escolher o tamanho da fonte usando valores numéricos ou nomenclaturas: x-small, xx-small, x-large, xx-large, small, medium, large, smaller e larger.
- **font-style:** Há 3 opções: normal, italic e oblique, que se referem a letras em sua fonte normal na vertical, letras inclinadas e letras oblíquas, respectivamente.
- **font-weight:** Define a intensidade de negrito que a fonte vai receber. Pode assumir 3 opções: bold, bolder e lighter ou valores numéricos.

- **font-variant:** Varia o tamanho das letras maiúsculas quando recebe o valor small-caps.

Exemplo:

```
.Exemplo1 {  
  color: #DDA0DD;  
  font-family: arial;  
  font-weight: bold;  
}  
  
.Exemplo2 {  
  color: #3366FF;  
  font-family: arial;  
  font-variant: small-caps;  
}  
  
.Exemplo3 {  
  font-family: serif;  
  font-size: x-small;  
  font-style: italic;  
}  
  
.Exemplo4 {  
  font-family: sans-serif;  
  font-size: 16px;  
  font-style: oblique;  
}
```

3.8.5 Borda

As bordas são muito úteis, pois, dependendo da criatividade do programador, assumem várias funções no desenvolvimento da página. Podem ser empregadas como elemento decorativo, separação entre textos e muitos outros recursos.

- **border-width:** Define a espessura da borda. Assume thin, medium e thick (fina, média e grossa respectivamente) como valores, ou um valor numérico.
- **border-color:** Define a cor da borda. Assume "cor" ou "#AAAAAA" como valores.
- **border-style:** Define o estilo da borda. Valores assumidos: dotted, dashed, solid, double, groove, ridge, inset, outset.

Exemplo:

```
.ExemploBorda1 {  
    border-width: thick;  
    border-style: dotted;  
    border-color: gold;  
}  
  
.ExemploBorda2 {  
    border-width: 20px;  
    border-style: outset;  
    border-color: red;  
}  
  
.ExemploBorda3 {  
    border-top-width: 1px;  
    border-style: dashed;  
    border-color: blue;  
}
```

A borda também pode ser definida utilizando apenas um atributo “**border**”, com a adição dos parâmetros na frente, conforme o exemplo:

```
.ExemploBorda {  
    border: 5px solid #0094ff;  
}
```

3.8.6 Margin e Padding

As margens definem o espaçamento entre os elementos HTML, que possuem quatro lados: right, left, top e bottom; e também de elementos de parágrafo ou cabeçalhos. Os valores assumidos pelas margens são em pixels.

- **margin-top:** Define a margem superior.
- **margin-right:** Define a margem direita.
- **margin-bottom:** Define a margem inferior.
- **margin-left:** Define a margem esquerda.
- **Margin:** Define os valores das quatro margens na seguinte ordem: **top**, **right**, **bottom**, **left**.

Exemplo:

```
.ExemploMargin {  
  margin-top: 10px;  
  margin-right: 5px;  
  margin-bottom: 10px;  
  margin-left: 5px;  
}
```

Abaixo são apresentadas as mesmas configurações para a margem, utilizando apenas uma linha:

```
.ExemploMargin {  
  margin: 10px 5px 10px 5px;  
}
```

Já o padding tem um funcionamento muito similar ao do margin, entretanto, ao invés de dar um espaçamento externo, ele dá um interno.

- **padding-top:** Define o padding superior.
- **padding-right:** Define o padding direita.
- **padding-bottom:** Define o padding inferior.
- **padding-left:** Define o padding esquerda.
- **padding:** Define os valores dos quatro paddings na seguinte ordem: **top, right, bottom, left.**

Exemplo:

```
.ExemploPadding1 {  
  padding-top: 10px;  
  padding-right: 5px;  
  padding-bottom: 10px;  
  padding-left: 5px;  
}  
  
.ExemploPadding2 {  
  padding: 10px 5px 10px 5px;  
}
```

3.8.7 Lista

Esta propriedade cria uma lista de elementos definidos pelo programador, usando como marcadores imagens ou números.

- **list-style-image:** Define uma imagem como marcador da lista. Valor: url('ExemploImagem.png')
- **list-style-position:** Posiciona o marcador da lista. Valores: outside e inside.
- **list-style-type:** Define o tipo de marcador da lista. Valores: disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha, upper-alpha.
- **list-style:** Define todas as propriedades acima em uma única linha na seguinte ordem: image, position e type.

Exemplo:

```
/* CSS */
ul.inside {
    list-style-position: inside;
}

ul.outside {
    list-style-position: outside;
}

ul.square {
    list-style-type: square;
}

ul.uproman {
    list-style-type: upper-roman;
}
```

```
<!-- HTML -->
<ul class="inside">
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
<ul class="outside">
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
<ul class="square">
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
<ul class="uproman">
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
```

3.8.8 Tabela

As propriedades de tabelas são:

- **table-layout:** Permite indicar se a tabela terá o seu fluxo com o tamanho fixo ou o seu tamanho acompanhará o fluxo do conteúdo. Valores: auto ou fixed.
- **border-collapse:** Essa propriedade define se as bordas em uma tabela vão se fundir ou vão ser separadas. Valores: collapse e separate.
- **border-spacing:** Define um espaçamento (horizontal e vertical) entre os elementos da tabela. Valores em px.

Ex.: border-spacing: 50px;

Pode-se usar `border-spacing: 50px 50px`, especificando em pixels os espaçamentos horizontais e verticais, respectivamente.

- **caption-side:** Pode ser usado para oferecer uma breve descrição de uma tabela, tal como uma legenda de uma imagem. Valores: top, bottom.

Exemplo 1:

```
/* CSS */  
table.TamanhoAuto {  
    table-layout: auto;  
}  
  

```

Exemplo 2:

```
/* CSS */  
table.TamanhoFixo {  
    table-layout: fixed;  
}  
  

```


Exemplo 3:

```
/* CSS */
table.BordasJuntas {
    border-collapse: collapse;
}
<!-- HTML -->
<table class="BordasJuntas" border="1">
    <tr>
        <td>Peter</td>
        <td>Griffin</td>
    </tr>
    <tr>
        <td>Lois</td>
        <td>Griffin</td>
    </tr>
</table>
```

Exemplo 4:

```
/* CSS */
table.BordasSeparadas {
    border-collapse: separate;
}
<!-- HTML -->
<table class="BordasSeparadas" border="1">
    <tr>
        <td>Peter</td>
        <td>Griffin</td>
    </tr>
    <tr>
        <td>Lois</td>
        <td>Griffin</td>
    </tr>
</table>
```

Exemplo 5:

```
/* CSS */
table.BordasSeparadas2 {
    border-collapse: separate;
    border-spacing: 20px;
}
<!-- HTML -->
<table class="BordasSeparadas2" border="1">
    <tr>
        <td>Peter</td>
        <td>Griffin</td>
    </tr>
    <tr>
        <td>Lois</td>
        <td>Griffin</td>
    </tr>
</table>
```

Exemplo 6:

```
/* CSS */
table.BordasSeparadas3 {
    border-collapse: separate;
    border-spacing: 40px 60px;
}
<!-- HTML -->
<table class="BordasSeparadas3" border="1">
    <tr>
        <td>Cleveland</td>
        <td>Brown</td>
    </tr>
    <tr>
        <td>Glenn</td>
        <td>Quagmire</td>
    </tr>
</table>
```

Exemplo 7:

```
/* CSS */
caption {
    caption-side: top;
}
<!-- HTML -->
<table border="1">
    <caption>Exemplo de caption</caption>
    <tr>
        <td>Cleveland</td>
        <td>Brown</td>
    </tr>
    <tr>
        <td>Glenn</td>
        <td>Quagmire</td>
    </tr>
</table>
```