

Curso de HTML – Aula 1: Estrutura básica de páginas web

Esta é a primeira aula do Curso de Desenvolvimento Web com HTML e nela veremos inicialmente alguns conceitos e fundamentos na criação de websites para, em seguida, conhecermos a estrutura básica de uma página web e realizarmos o primeiro exercício: o famoso “Olá mundo!”.

Histórico

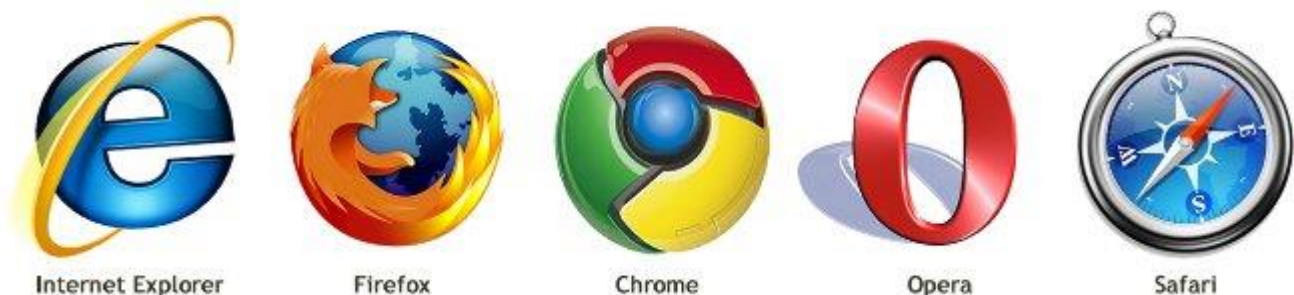
O conceito de web como conhecemos hoje foi criado por **Tim Berners-Lee** em 1989 quando ele trabalhava na seção de Computação da Organização Europeia de Pesquisa Nuclear na Suíça. Na época ele pesquisava um método que possibilitasse aos cientistas do mundo inteiro compartilhar eletronicamente os documentos. Então, em 1990, ele criou o protótipo de um navegador chamado Nexus, um protocolo para transmitir arquivos denominado HTTP e uma linguagem de marcação de texto chamada HTML.

A linguagem HTML

Em essência, todos os sites que você já conhece são arquivos codificados em uma linguagem específica e estão armazenados em um servidor web. Este servidor, nada mais é do que um computador conectado à internet que aguarda ser consultado para disponibilizar os arquivos (páginas web) que ele armazena.

Para consultar as páginas web em um servidor, precisamos de um programa que atenda a uma série de regras (protocolos) e, principalmente, entenda a linguagem utilizada para escrever essas páginas. Alguns exemplos desse tipo programa, genericamente chamado de *browser* ou navegador, são: Internet Explorer, Mozilla Firefox, Google Chrome, etc.

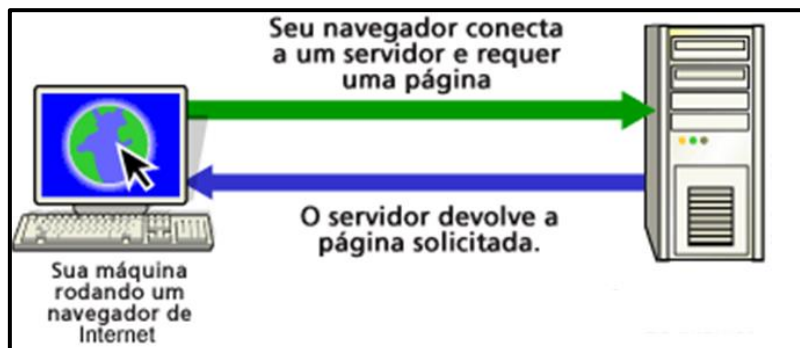
PRINCIPAIS NAVEGADORES DE INTERNET



Todos os navegadores de internet citados acima possuem a capacidade de interpretar o HTML (abreviação para a expressão inglesa *HyperText Markup Language*) que é uma linguagem de marcação de texto utilizada para produzir páginas na web. Basicamente, o HTML estabelece

como um determinado elemento deve ser visualizado, não sendo, portanto, uma linguagem de programação, e sim uma linguagem de formatação de conteúdo.

Resumindo, um arquivo HTML que está armazenado em um servidor web é formado essencialmente por texto (conteúdo do site) e **tags**. Sendo assim, quando o navegador recebe o arquivo e “lê” o HTML, ele interpreta todas as tags e atribui a formatação no conteúdo antes de exibir na tela para o usuário.



Conhecendo as tags

A linguagem de marcação de texto HTML baseia-se em etiquetas (**tags**) com valor semântico, englobando trechos de conteúdo, dotando-os de sentido e valor. Essas tags são comandos de texto escrito entre os sinais de menor "<" e maior ">" que informam ao navegador como deve ser apresentado o conteúdo. Salvo algumas exceções que veremos mais adiante, as tags devem ser escritas aos pares e, por isso, costumamos dizer que abrimos e fechamos uma tag.

Para escrever uma linha de código HTML precisamos de uma tag de abertura e uma de fechamento. Como você pode notar no exemplo abaixo, a diferença entre elas é que na tag de fechamento existe uma barra "/".

Exemplo de tag:

1 | **** Conteúdo a ser Formatado ****

A tag **** informa ao navegador que todo o texto colocado entre **** e **** deve ser mostrado em negrito (O comando "b" é uma abreviação para "bold" – negrito).

Como já disse anteriormente neste [curso de HTML](#), toda regra tem sua exceção e a exceção aqui no HTML é que algumas tags não "trabalham" em pares e, por isso, a abertura e o fechamento ocorrem na mesma tag. Segue abaixo um exemplo:

Exemplo de tag isolada:

1 | **
**

No exemplo acima a tag **
** informa ao navegador para realizar um pulo de linha, ou seja, dar um espaço entre dois parágrafos. Repare que esta tag não requer um conteúdo para ser processado e justamente por essa razão não há uma tag de fechamento, mas sim, outra forma de referenciar o fechamento, com um espaço e "/" antes do sinal de maior.

Estrutura de uma página HTML

Um site, em geral, é formado por uma ou mais páginas HTML que, por sua vez, é formado por um conjunto de tags atreladas ao conteúdo do site, seja ele em forma de texto, imagens, vídeos, etc. Para escrevermos uma página html, por mais simples que seja, precisamos utilizar no mínimo as seguintes tags abaixo:

<html> é a tag inicial de qualquer documento no formato html;

<head> é uma seção da página que apresenta informações gerais;

<title> é a tag que informa o título da página na barra do navegador;

<body> é a seção principal, na qual será escrito o conteúdo da página.

Com o conteúdo apresentado até aqui e com a tag <p> utilizada para criar parágrafos, já podemos escrever a nossa primeira página em HTML. Para isso, observe o modelo abaixo e logo em seguida os comentários de cada linha do código.

Modelo básico de uma página HTML

```
1<html>
2<head>
3  <title>Título da minha página</title>
4</head>
5<body>
6  <p>Olá Mundo! Esta é minha primeira página web</p>
7</body>
8</html>
```

Observação: os número na frente do código servem exclusivamente para a explicação de cada linha

Na linha 1 abrimos a tag <html> que informa ao navegador que esta é uma página html;

Na linha 2 abrimos a tag <head> na qual ficará o cabeçalho da página;

Na linha 3 temos a tag <title> que deverá sempre estar dentro da tag <head>. Observe que abrimos com a tag <title>, colocamos o título da página e, em seguida, fechamos com a tag </title>;

Na linha 4 fechamos a tag <head> que abrimos na linha 2;

Na linha 5 abrimos a tag <body> que conterá todo o conteúdo da nossa página;

Na linha 6 abrimos a tag <p> para colocarmos o texto “Olá Mundo! Esta é minha primeira página web” e logo em seguida fechamos o parágrafo com a tag </p>

Na linha 7 fechamos a tag <body> que abrimos na linha 5

Na linha 8 finalizamos o documento fechando a tag <html> que abrimos na linha 1.

Primeiro Exercício

Falamos no início desse curso sobre servidores que armazenam as páginas na internet, mas, para reproduzirmos o exemplo anterior e os exercícios que faremos aqui, não precisaremos de nada sofisticado, precisaremos apenas de um navegador e um simples [editor de texto](#) como, por exemplo, o Bloco de Notas de Windows.

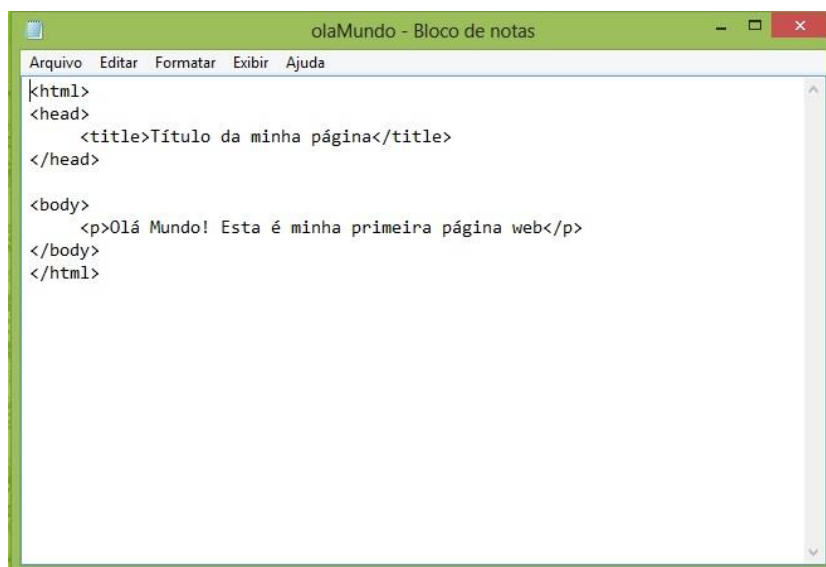
Mais adiante, em outro curso, veremos com detalhes a questão de servidor web, mas, por enquanto, já temos o suficiente para criar nossa primeira página web. Então, vamo lá!

1º passo

Abra o Bloco de Notas do Windows ou qualquer outro editor de texto de sua preferência; apenas certifique-se de que ele consiga criar um arquivo de texto simples com uma extensão “.html”.

2º passo

Assumindo que você esteja usando o Bloco de Notas, digite no programa o modelo básico de página HTML estudado anteriormente exatamente como mostra a imagem abaixo:

A screenshot of the Windows Notepad application window titled "olaMundo - Bloco de notas". The menu bar includes "Arquivo", "Editar", "Formatar", "Exibir", and "Ajuda". The text area contains the following HTML code:

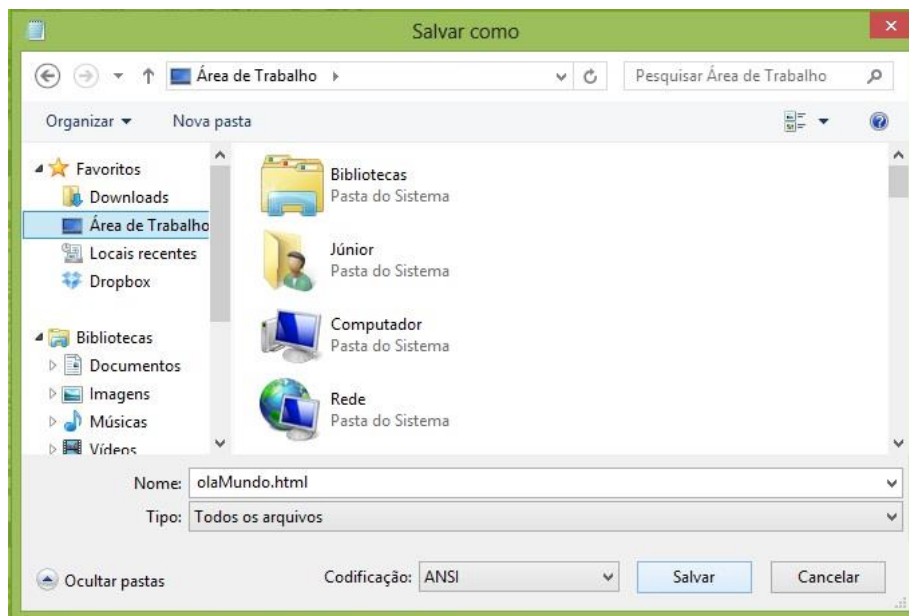
```
<html>
<head>
  <title>Título da minha página</title>
</head>

<body>
  <p>Olá Mundo! Esta é minha primeira página web</p>
</body>
</html>
```

3º passo

Depois de digitado o código HTML no Bloco de Notas, precisamos salvar o arquivo no formato “.html” para visualizarmos no navegador. Para isso:

- No Bloco de Notas clique no menu Arquivo e escolha a opção “Salvar como...”;
- Conforme mostra a imagem abaixo, escolha “Todos os arquivos” na opção Tipo; digite “olaMundo.html” na opção Nome e selecione a área de trabalho para salvar o arquivo.

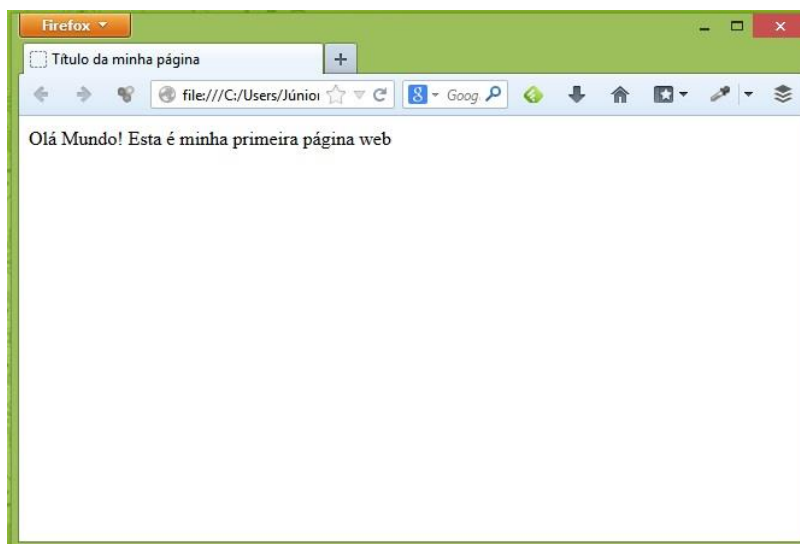


4º passo

Encontre o arquivo "*olaMundo.html*" na área de trabalho; clique com o botão direito do mouse; selecione a opção "Abrir com..." e escolha algum navegador instalado no seu computador (Internet Explorer, Firefox ou Chrome).

5º passo

Caso você tenha seguido corretamente todos os passos descritos aqui, você deverá ver abrir o seu navegador com um conteúdo semelhante ao da imagem abaixo:



Parabéns! Você criou a sua primeira página web no nosso curso de HTML. Minha sugestão é que agora você altere o conteúdo desse exercício para que compreenda melhor o funcionamento da linguagem HTML. Em breve, com o conteúdo das próximas aulas, você incrementará esse exercício tornando uma página web muito mais interessante.

Curso de HTML – Aula 2: Títulos, listas e outras tags

Esta é a segunda aula do Curso de Desenvolvimento Web com HTML. Depois de conhecermos na aula anterior alguns conceitos sobre a criação de páginas web e a sintaxe da linguagem de marcação de texto HTML, estudaremos agora outras tags para fazermos um exercício mais elaborado: o currículo do nosso amigo José da Silva.

Títulos

Assim como em um livro, uma página web pode conter uma hierarquia de títulos e subtítulos para estabelecer uma divisão de seu conteúdo e, para conseguirmos realizar essa tarefa, utilizamos as tags de título h1, h2, h3, h4, h5 e h6.

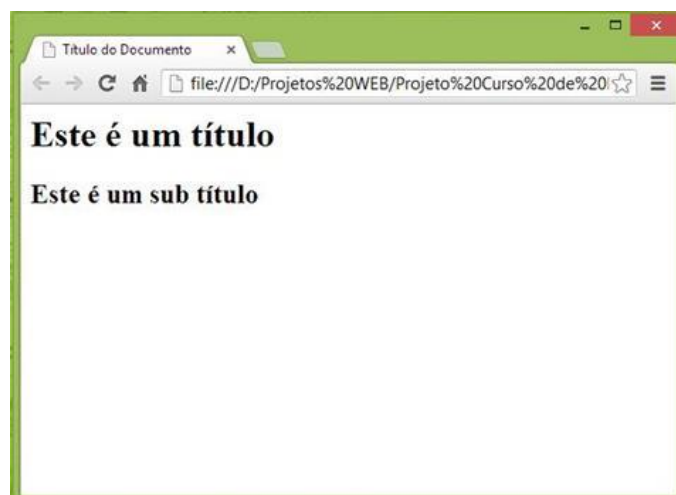
As tags <h1>, <h2>, <h3>, <h4>, <h5> e <h6> informam ao navegador que trata-se de um título (h vem de “*heading*” – cabeçalho ou título), sendo <h1> o título de primeiro nível apresentado com o maior tamanho de texto; <h2> o subtítulo de segundo nível apresentado com tamanho de texto um pouco menor; e <h6> o subtítulo de sexto nível apresentado com o menor tamanho de texto. Vamos a um exemplo:

Exemplo de cabeçalhos:

1<h1>Este é um título</h1>

2<h2>Este é um sub-título</h2>

Veja na imagem abaixo como o navegador exibe o código do exemplo acima.



Para cada tag de título o navegador atribui um tamanho diferente de fonte (letra). Esse tamanho pode ser alterado através de [regras CSS](#) que veremos posteriormente em outro curso, mas, por enquanto, utilizaremos o tamanho padrão da fonte definido pelo navegador.

Dica extra: Sites como o Google, Yahoo! e Bing utilizam as tags de título como parâmetro de ranqueamento nos resultados das suas buscas orgânicas. Dessa forma, devemos utilizar essas

tags de forma a atender as chamadas **técnicas de SEO** (*Search Engine Optimization*) que ajudam a melhorar o ranqueamento de páginas dentro dos buscadores.

Lista ordenada e não ordenada

Em diversas situações do nosso dia a dia necessitamos listar todo tipo de coisas. Na criação de páginas web não é diferente, e por essa razão aprenderemos agora a criar os dois tipos de listas mais comuns: as listas ordenadas e não ordenadas.

Listas ordenadas, como o próprio nome já diz, são utilizadas para exibir qualquer conteúdo de forma ordenada. Por exemplo:

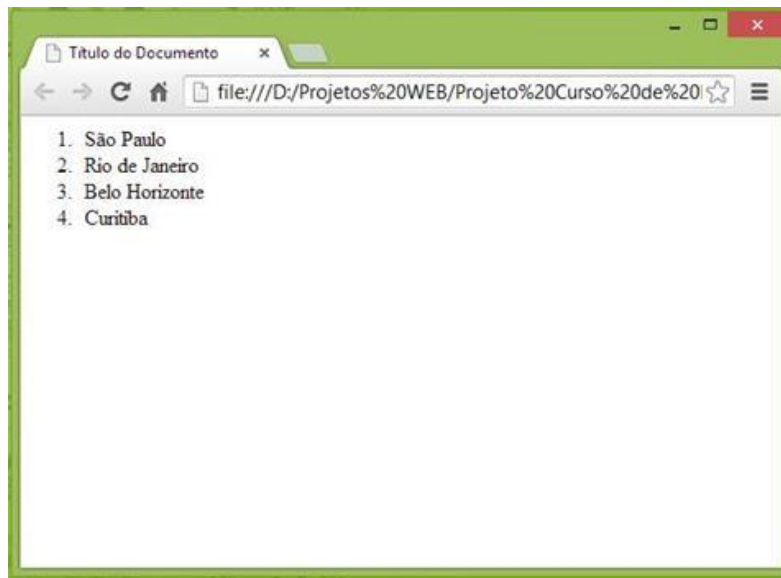
1. São Paulo
2. Rio de Janeiro
3. Belo Horizonte
4. Curitiba

Para criarmos uma lista assim precisaremos dessa vez de duas tags para trabalhar em conjunto, a tag `` e a ``. Segue abaixo um exemplo de como fazemos para reproduzir em html a lista ordenada acima.

```
1<ol>
2  <li>São Paulo</li>
3  <li>Rio de Janeiro</li>
4  <li>Belo Horizonte</li>
5  <li>Curitiba</li>
6</ol>
```

Observação: A tag `` deve possuir pelo menos uma tag `` para ser considerada uma lista ordenada.

Veja agora na imagem abaixo como o navegador exibe o código da lista ordenada no exemplo acima.



Listas não ordenadas, por sua vez, são utilizadas para exibir qualquer conteúdo de forma aleatória, não ordenada. Por exemplo:

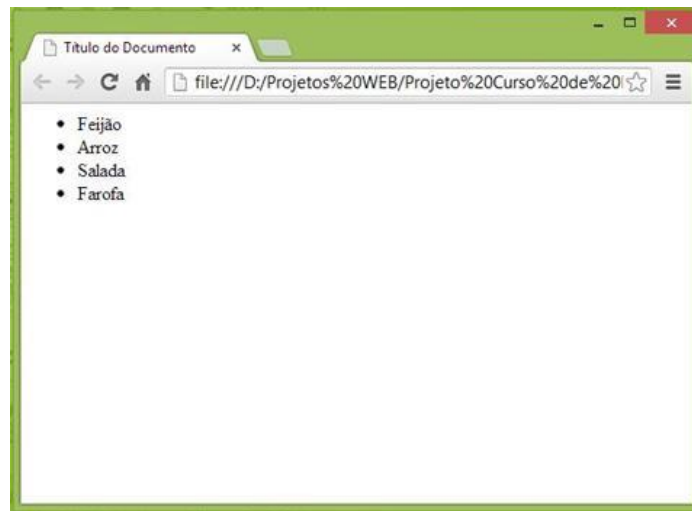
- Feijão
- Arroz
- Salada
- Farofa

Para criarmos uma lista assim precisaremos também trabalhar em conjunto com duas tags, com a `` e a ``. Segue abaixo um exemplo de como fazemos para reproduzir em html a lista não ordenada acima.

```
1<ul>
2  <li>Feijão</li>
3  <li>Arroz</li>
4  <li>Salada</li>
5  <li>Farofa</li>
6</ul>
```

Observação: A tag `` deve possuir pelo menos uma tag `` para ser considerada uma lista não ordenada.

Veja na imagem abaixo como o navegador exibe o código da lista não ordenada do exemplo acima.



Tags isoladas

Tags isoladas são aquelas que são abertas e fechadas em uma única tag como, por exemplo, a tag `
` já citada no capítulo anterior.

Exemplo de tag isolada:

1 | `<p>`Início do meu parágrafo e `
`continua na linha abaixo`</p>`

No exemplo acima a tag `
` informa ao navegador para realizar uma quebra de linha e, embora todo o texto esteja dentro da tag `<p>`, vemos o parágrafo da seguinte forma:



Observação: Note que a tag é escrita como se fosse uma mistura de tag de abertura e de fechamento com uma barra `'/'` no final: `
`.

Outra tag isolada é a `<hr />` que serve para definir uma linha horizontal. Veja no código abaixo um exemplo de como podemos utilizar essa tag.

```
1<h1>Titulo do meu site</h1>
2<hr />
3<p>Linha horizontal acima</p>
```

Veja agora na imagem abaixo como o navegador exibe o código do exemplo acima.



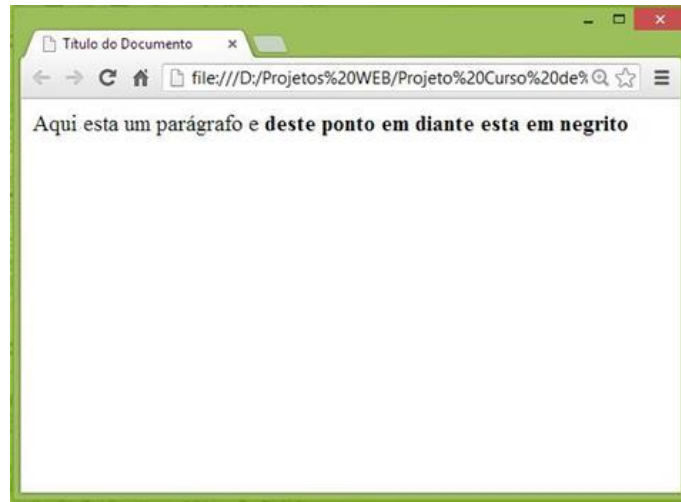
Formatação em negrito e itálico

De acordo com os novos padrões de desenvolvimento definidos pela W3C, não é mais correto utilizar as tags e <i> para formatar textos com negrito e itálico. Porém, enquanto não chegamos às tão esperadas [CSS no próximo curso](#), vamos utilizar essas tags em desuso nos nossos exercícios para efeito didático e também de conhecimento.

Para colocarmos um texto em negrito devemos adicioná-lo entre a tags e da seguinte forma:

```
1<p>Aqui esta um parágrafo e <b>deste ponto em diante está em
2  negrito</b></p>
```

Veja na imagem abaixo como o navegador exibe o código do exemplo acima.



Para colocarmos um texto em itálico devemos adicioná-lo entre a tags <i> e </i> da seguinte forma:

```
1<p>Aqui esta outro parágrafo e <i>deste ponto em diante está em  
2  itálico</i></p>
```

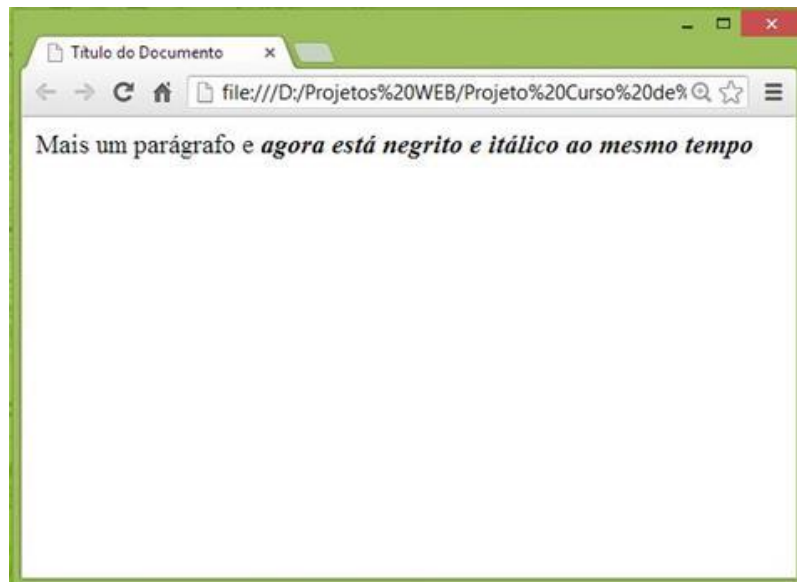
Veja na imagem abaixo como o navegador exibe o código do exemplo acima.



Se for necessário podemos também combinar as tags de negrito e itálico trabalhando da seguinte forma:

```
1<p>Mais um parágrafo e <b><i>agora está negrito e itálico ao  
2  mesmo tempo</i></b></p>
```

Veja na imagem abaixo como o navegador exibe o código do exemplo acima.



Dica extra: As tags e causam visualmente o mesmo resultado que as e <i>, respectivamente. A diferença está no fato de que além de causarem o efeito visual de negrito e itálico, as tags e dão ênfase ao texto informando para as ferramentas de buscas que aquele trecho em específico é importante e merece maior destaque nos resultados de buscas.

Exercício:

Utilizando as tags estudadas no capítulo 1 e 2 desse curso, crie uma página HTML que exiba o currículo do nosso amigo José da Silva, conforme a imagem abaixo.



Curso de HTML – Aula 3: Inserindo imagens no html

Esta é a terceira aula do Curso de Desenvolvimento Web com HTML na qual aprenderemos inserir imagens na nossa página web. Caso ainda não tenha lido os artigos anteriores, recomendo que leia antes: [Curso de Desenvolvimento Web com HTML](#), [Aula 1: Estrutura de uma página web](#) e [Aula 2: Títulos, listas e outras tags](#)

Geralmente, quando uma pessoa acessa um site ela está à procura de conteúdo que, na maioria das vezes, está em formato de texto. No entanto, se as páginas web fossem formadas apenas por texto, elas seriam muito monótonas e entediadas para a leitura do internauta. Felizmente, temos à nossa disposição uma tag muito fácil de utilizar para inserir imagens ao conteúdo das nossas páginas e a aprenderemos agora mesmo neste capítulo.

Inserindo imagens no HTML

Para inserirmos uma imagem em nossa página web temos que utilizar a tag `` e dentro dela informar onde a imagem está localizada através do atributo `"src"` (abreviatura para *"source"* – local de armazenagem). Para entendermos melhor, vamos a um exemplo:

```
1 | 
```

No exemplo acima estamos inserindo em nossa página uma imagem chamada `"bandeira.jpg"`. Note que a tag ``, semelhante às tags `
` e `<hr />`, é do tipo *"tag isolada"* e por essa razão, temos uma só tag de abertura e fechamento, não necessitando de um texto inserido nela.

Em algumas situações precisamos acrescentar mais informações às nossas tags através de atributos, como no caso da tag ``, em que o `"src"` informa qual é o endereço da imagem que queremos exibir na nossa página. Existem diversos tipos de atributos para as mais diversas situações e neste capítulo aprenderemos alguns deles relativos à inserção de imagens.

Regra geral: Atributos são escritos dentro das tags seguidos por um sinal de igual e entre aspas é declarado o valor do atributo.

O valor do atributo, no caso do `"src"`, deve ser o nome da imagem ou, quando necessário, o endereço da imagem seguido do nome dela. Vamos exemplificar para entender melhor:

```
1 | 
```

Exemplo 1: Apenas o nome da imagem

Para que esse comando html funcione precisamos que a imagem que você quer inserir na página esteja nomeada exatamente assim: `bandeira.jpg` . Onde, “bandeira” é o nome do arquivo e “jpg” é a extensão do tipo da imagem. Além dessas condições, é imprescindível também que a imagem esteja na mesma pasta que a sua página html.

```
1 | 
```

Exemplo 2: Endereço (pasta) e nome da imagem

Já nesse exemplo, temos um hábito muito comum entre os desenvolvedores: separar as imagens colocando-as dentro de uma única pasta. Nesse caso, estamos informando através do atributo “src” que a mesma imagem (`bandeira.jpg`) está localizada agora dentro de uma pasta chamada “imagens”.

Tipos de imagens: JPG, GIF e PNG

Tal como a extensão “.html” para documentos html, o “.jpg” informa ao navegador que o arquivo é uma imagem. Veja abaixo os três tipos de imagens mais comuns:

- JPG (*Joint Photographic Experts Group*)
- GIF (*Graphics Interchange Format*)
- PNG (*Portable Network Graphics*)

Em geral, imagens do tipo GIF são mais utilizadas para desenhos, ícones e ilustrações. Por trabalhar com apenas 256 cores, as imagens ficam com o arquivo muito leve e carregam rapidamente no navegador. Além disso, o formato GIF permite inserir uma sequência de imagens em um único arquivo para criar pequenas animações.

As imagens do tipo JPG são muito utilizadas em fotografias por aliar duas características importantes: níveis razoáveis de qualidade de imagem (trabalha com 16,8 milhões de cores) e geração de arquivos de tamanhos pequenos quando comparado a outros formatos.

Tradicionalmente os formatos GIF e JPG são os mais utilizados na [internet](#), mas ultimamente o formato PNG tem se tornado cada vez mais popular por reunir a qualidade das imagens JPG (milhões de cores) e as características tão bem aceitas do formato GIF: animação, fundo transparente e compressão sem perda de qualidade.

Atributos: *alt*, *title*, *width* e *height*

Toda vez que utilizamos a tag `` obrigatoriamente temos que informar o atributo “src”. Além dele existem alguns outros atributos que podem ser bem úteis quando estamos trabalhando com imagens. Vamos conhecer então mais alguns atributos.

Atributo *alt*

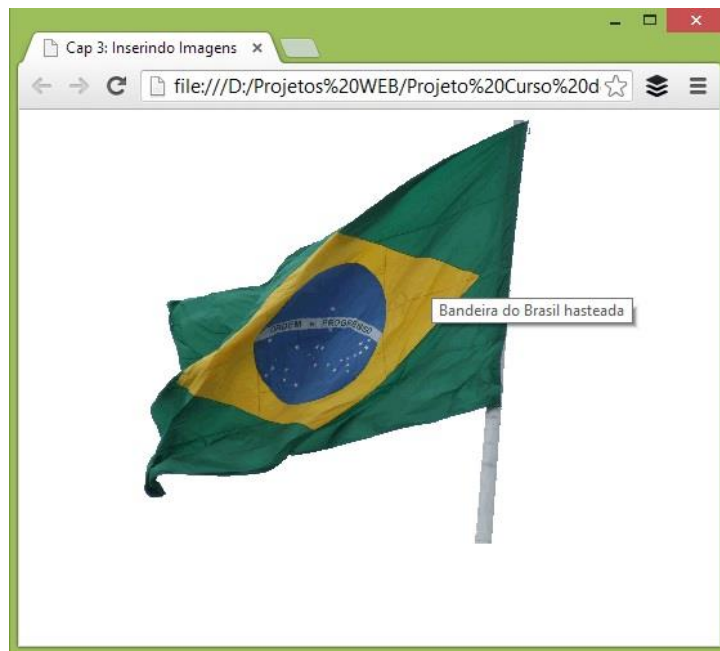
Este atributo é usado para fornecer uma descrição textual da imagem, principalmente por suas razões: Primeiro, caso a imagem não apareça para o usuário é exibido o texto contido no atributo e, segundo, informa para os [mecanismos de buscas](#) sobre o conteúdo da imagem em questão. Exemplo:

```
1 | 
```

Atributo *title*

Alguns navegadores mostram uma caixa *pop-up* quando o usuário passa o mouse sobre a imagem. Este conteúdo é informado no atributo "title" e deve ser usado para fornecer uma curta descrição da imagem. Exemplo:

```
1 | 
```



Colocando o ponteiro do mouse sobre a imagem, sem clicar, aparecerá uma caixa *pop-up* semelhante à da imagem acima com o conteúdo do atributo "title".

Atributos *width* e *height*

Outros dois atributos importantes na tag são o "width" e "height". Eles são usados para definir respectivamente, a largura e a altura da imagem.

Os navegadores, tendo informações sobre a altura e a largura das imagens, não precisam esperar que as imagens descarreguem completamente para continuar a exibir o resto da página. Além disso, com os atributos "width" e "height", você pode alterar a dimensão da imagem, se eles não forem definidos, a imagem será inserida com seu tamanho real. Exemplo:

```
1 | 
```

Exercício: Meu currículo em HTML

Para exercitarmos os conhecimentos adquiridos nos capítulos anteriores, crie um documento HTML em um arquivo chamado *meucurriculo.html* com os seus dados pessoais e profissionais. Não se esqueça de utilizar as tags de cabeçalhos para títulos mais importantes e inserir a sua foto (*minhafoto.jpg*) com os devidos atributos necessários.

Curso de HTML – Aula 4: Navegando pelos links

Esta é a quarta aula do Curso de Desenvolvimento Web com HTML na qual aprenderemos a inserir links em uma página web. Por mais simples que um site possa ser, ele será formado normalmente por no mínimo duas páginas web. E, geralmente, estas páginas estão interligadas através dos *hiperlinks* (também conhecidos apenas por links) que permitem aos usuários “saltarem” de uma página para outra nos sites; uma atividade que ficou mais popularmente conhecida como “navegar” na internet.

Um link é uma ligação entre páginas web, podendo esta ligação ser para uma página no mesmo site (link interno) ou para uma página em outro site.

Como criar um link

Para criarmos um link, devemos utilizar a tag <a>. Porém, essa tag sem atributos não criará nenhum link, seja interno ou externo. Para que um link seja criado, devemos, no mínimo, declarar na tag <a> o atributo href com o caminho absoluto ou relativo de uma página. A seguir um exemplo de link para o site Hiperbytes.

```
1 | <a href="https://hiperbytes.com.br">Um link para Hiperbytes</a>
```

Neste exemplo criamos um link com a tag <a> e em seguida declaramos o atributo **href** com o valor “http://www.hiperbytes.com.br”, que é o endereço absoluto do Hiperbytes. Após declarar o **href**, devemos inserir o texto ([Um link para Hiperbytes](#)) que será mostrado no navegador como link e, em seguida, fechar a tag com um .

Se você quer criar links entre páginas de um mesmo site você não precisa escrever o endereço absoluto, ou seja, o endereço completo de cada página como no exemplo anterior. Por exemplo, se você tem duas páginas (pagina1.html e pagina2.html) e salvou as duas em uma mesma pasta, você pode criar um link de uma página para a outra usando somente o nome do arquivo no link. Dessa maneira, um link da pagina1.html para a pagina2.html ficará da seguinte forma:

```
1 | <a href="pagina2.html">Link para a página 2</a>
```

Se a pagina2.html for colocada em uma pasta diferente que a pagina1.html, o link na pagina1.html para a pagina2.html será então da seguinte forma:

```
1 | <a href="pasta/pagina2.html">Link para a página 2</a>
```

Por outro lado, um link da pagina2.html para a pagina1.html será da seguinte forma:

```
1 | <a href=" ../pagina1.html">Link para a página 1</a>
```

A notação “../” aponta para a pasta a um nível acima do arquivo onde foi feito o link. Seguindo o mesmo princípio, você pode apontar para dois (ou mais) níveis acima, escrevendo “../../”.

Atributos *title* e *target*

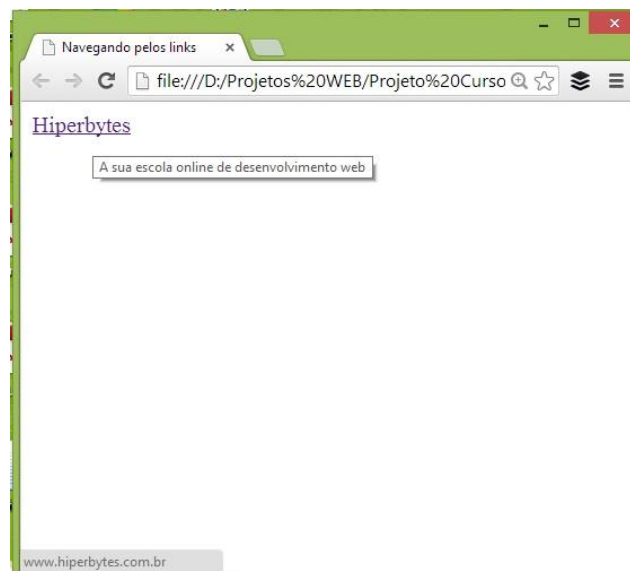
Além do href, podemos informar também outros dois atributos quando estamos criando um link: o *title* e o *target*.

Atributo *title*

Semelhante ao atributo title utilizado na tag do capítulo anterior, aqui na tag <a> este atributo é utilizado para fornecer uma breve descrição do link. Exemplo:

```
1 <a href="http://www.hiperbytes.com.br"
2   title="A sua escola online de desenvolvimento web">Hiperbytes</a>
```

Veja na imagem abaixo como o navegador exibe o código do exemplo acima.



Colocando o ponteiro do mouse sobre o link, sem clicar, aparecerá uma caixa *pop-up* com o texto “A sua escola online de desenvolvimento web”.

Atributo *target*

Utilizamos o atributo target para informar onde o link do novo documento deve ser aberto, sendo possíveis os seguintes valores:

- `_blank` (em uma nova janela ou aba)
- `_self` (na mesma janela do documento que contém o link)
- `_parent` (em um frame que seja o “pai” do frame* no qual o link se encontra)
- `_top` (na mesma janela do documento que contém o link)

Veja abaixo alguns exemplos mais comuns:

```
1<a href="pag1.html" target="_blank">Abre em outra janela </a>
```

```
2<a href="pag2.html" target="_self">Abre na mesma janela </a>
```

```
3<a href="pag3.html">Abre na mesma janela </a>
```

Conforme você pode observar na terceira linha do exemplo acima, o comportamento padrão de um link é abrir o documento na mesma página caso o atributo target não seja utilizado.

Nota: O desenvolvimento de sites com frames é uma técnica que foi largamente empregada no passado, mas, sob o ponto de vista das Web Standards, devemos evitar o uso desta técnica, pois traz sérias restrições à acessibilidade.

Âncoras

Além de [criar links](#) para outras páginas, podemos criar links para uma determinada seção dentro da própria página na qual o link se encontra. Esse recurso chama-se ancoragem, pois as seções para as quais queremos criar um link devem possuir uma âncora, e para criarmos este recurso precisaremos do atributo id e o símbolo "#".

Precisamos do atributo id para marcar o elemento que é o destino do link. Por exemplo:

```
1 | <h1 id="secao1">Seção 1</h1>
```

Você agora pode criar um link que leve àquele elemento usando o símbolo "#" no atributo href do link. O símbolo "#" informa ao navegador para ficar na mesma página e ele deve ser seguido pelo valor do atributo id para onde o link vai. Por exemplo:

```
1 | <a href="#secao1">Link para a seção 1</a>
```

Uma utilização muito comum para este recurso é quando temos vários capítulos ou seções em uma única página com muito conteúdo e para encontrarmos rapidamente uma parte específica, criando no início da página um índice com link para cada capítulo ou seção.

Observação: Para visualizar o efeito de ancoragem em nossa página precisamos ter conteúdo suficiente para aparecer a barra de rolagem vertical do navegador, só assim conseguimos perceber o movimento de deslocamento da âncora.

Exercícios

1. Crie duas páginas HTML index.html e contato.html. As duas páginas devem possuir três links cada, sendo um para a página index, um para a página de contato e outro para o site do [Google](#) (link externo).
2. Crie um documento HTML em um arquivo chamado ancora-pagina1.html que contenha um link que aponta para uma âncora dentro da própria página. Dica: insira um conteúdo suficientemente grande para que a barra de rolagem vertical do navegador apareça e coloque a âncora no final da página.

Curso de HTML – Aula 5: Criação de tabelas

Esta é a quinta aula do Curso de Desenvolvimento Web com HTML na qual aprenderemos a estruturar tabelas em nossa página web. Caso ainda não tenha lido os artigos anteriores a este, segue os links:

Imagine que você esteja desenvolvendo um site e precise criar uma página HTML com os dados de um relatório no formato de planilha eletrônica semelhante ao Microsoft Excel. Os dados oriundos desse relatório precisam ser tabulados, ou seja, precisam ser organizados de forma lógica em uma tabela composta por linhas e colunas como ilustra a imagem abaixo.

Marca:	Modelo:	Ano:	Valor:
Volkswagen	Golf	2009	R\$38.500
Fiat	Palio	2010	R\$19.700
Honda	Civic	2012	R\$61.500
O melhor preço da região			

Criar tabelas não é um dos assuntos mais fáceis de entender na linguagem HTML. No entanto, se você acompanhar atentamente desde o início o passo a passo da explicação, você verá que não é tão difícil assim. Vamos lá então.

Estrutura de uma tabela em HTML

Uma tabela em linguagem HTML é formada por, no mínimo, três tags básicas que são: <table>, <tr> e <td>. Sendo que:

- A tag **<table>** significa “*tabela*” indica onde começa e termina uma tabela;
- A tag **<tr>** significa “*table row*” (linha de tabela) e indica onde começa e termina e uma linha horizontal da tabela; e
- A tag **<td>** significa “*table data*” (dados da tabela) e indica onde começa e termina cada célula contida nas linhas da tabela.

Vejamos no exemplo abaixo a codificação de uma tabela bem simples e, logo em seguida, vamos analisar cada linha do código para entender a diferença entre as tags.

```

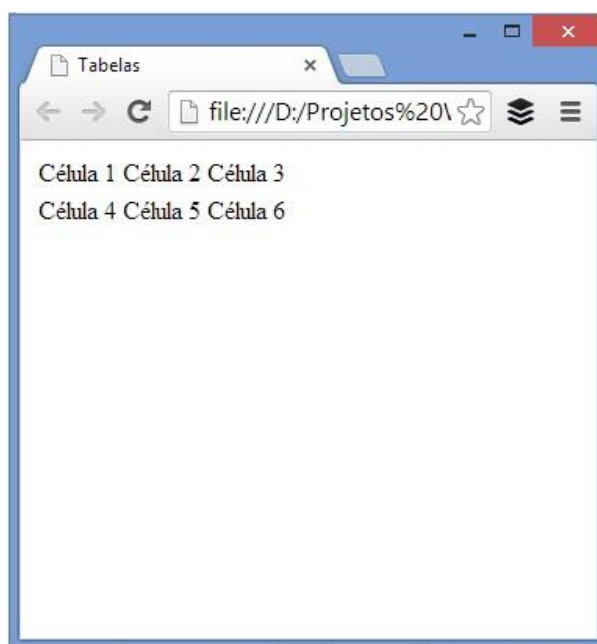
1 <table>
2   <tr>
3     <td>Célula 1</td>
4     <td>Célula 2</td>
5     <td>Célula 3</td>
6   </tr>
7   <tr>
8     <td>Célula 4</td>
9     <td>Célula 5</td>
10    <td>Célula 6</td>
11  </tr>
12</table>

```

Explicando cada tag da tabela

- No exemplo acima começamos uma tabela com a tag <table> e em seguida utilizamos a tag <tr> indicando o início de uma linha;
- Dentro dessa primeira linha da tabela inserimos três células (colunas) que são representados pelo conjunto de tags <td> e </td>, que são responsáveis por exibir o conteúdo de cada célula no navegador;
- A primeira linha termina com a tag </tr> e uma nova linha começa imediatamente com a tag <tr>, seguindo a mesma estrutura da linha anterior e, ao final dela, temos a tag <table> indicando o fim da tabela.

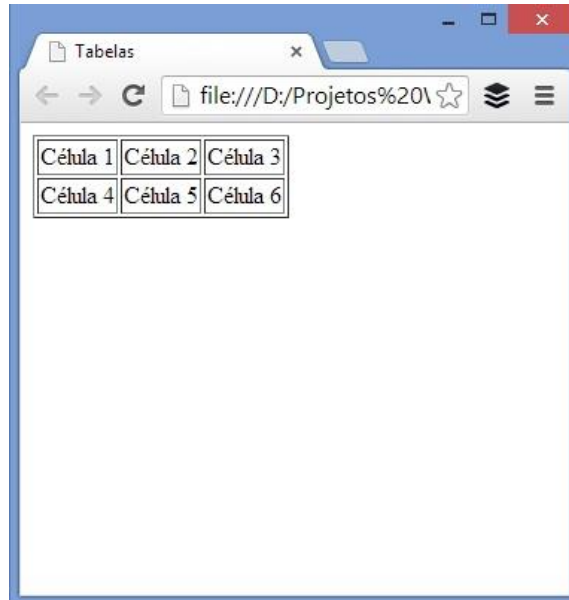
Veja na imagem abaixo como o navegador exibe o exemplo acima.



Perceba que a tag <table> não é utilizada sozinha. Ela necessita de pelo menos um ou mais elementos <tr> que, por sua vez, necessitam de pelo menos um ou mais elementos <td>. No caso do nosso exemplo, a tabela possui duas linhas (horizontais) e três colunas (verticais), sendo que as Células 1, 2 e 3 formam uma linha e as Células 1 e 3 formam uma coluna.

Observação: Uma tabela pode conter um número ilimitado de linhas e colunas.

Para uma maior legibilidade da nossa tabela podemos utilizar o atributo **border** para definir bordas entre as células da tabela. Alterando apenas a tag <table> para **<table border="1">**, temos o seguinte resultado.



Célula 1	Célula 2	Célula 3
Célula 4	Célula 5	Célula 6

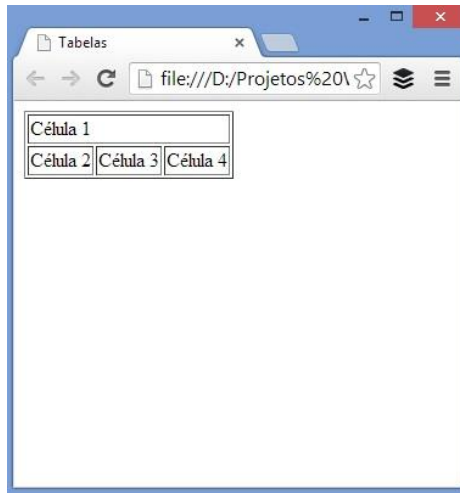
Atributos **colspan** e **rowspan**

Outros dois atributos muito utilizados em tabelas são o **colspan** e **rowspan**.

O **colspan** (abreviação para "column span") é utilizado na tag <td> para indicar quantas colunas estarão contidas em uma célula. Veja no código abaixo um exemplo da utilização desse atributo.

```
1 <table border="1">
2   <tr>
3     <td colspan="3">Célula 1</td>
4   </tr>
5   <tr>
6     <td>Célula 2</td>
7     <td>Célula 3</td>
8     <td>Célula 4</td>
9   </tr>
10</table>
```

Veja na imagem abaixo como o navegador exibe o código do exemplo acima.



Célula 1		
Célula 2	Célula 3	Célula 4

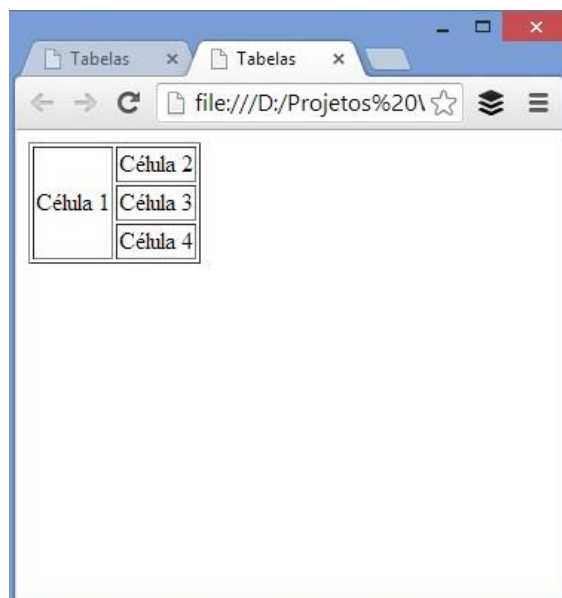
Como você já deve ter imaginado, o *rowspan* especifica quantas linhas estarão contidas em uma célula. Veja no código abaixo um exemplo da utilização desse atributo.

```

1 <table border="1">
2   <tr>
3     <td rowspan="3">Célula 1</td>
4     <td>Célula 2</td>
5   </tr>
6   <tr>
7     <td>Célula 3</td>
8   </tr>
9   <tr>
10    <td>Célula 4</td>
11  </tr>
12</table>

```

No exemplo acima o *rowspan* é definido em "3" na Célula 1. Isto significa que uma célula deve conter 3 linhas. A Célula 1 e Célula 2 estão na mesma linha, enquanto Célula 3 e Célula 4 formam duas linhas independentes. Confuso não é? Veja então na imagem abaixo como o navegador exibirá esse código e ficará mais fácil de entender a ideia.



Célula 1	Célula 2
	Célula 3
	Célula 4

Mais tags para as tabelas

Além das tags apresentadas até aqui, existem mais quatro que utilizamos para criar tabelas eficientes e que atendam as recomendações da W3C.

Visualmente elas mudarão pouca coisa na nossa tabela, porém, é importante que se conheça para que no próximo de [curso de CSS](#) seja possível aplicar estilos a essas marcações HTML. Segue abaixo essas quatro novas tags e seu significado.

- **th** – define uma célula de cabeçalho da tabela
- **thead** – define o cabeçalho da tabela
- **tbody** – define o corpo da tabela
- **tfoot** – define o rodapé da tabela

Para entender melhor a utilização dessas tags, observe atentamente o código abaixo e veja a aplicação de cada uma delas dentro do contexto criado na tabela apresentada no início dessa aula.

```
<table border="1">
1   <thead>
2       <tr>
3           <th>Marca:</th>
4           <th>Modelo:</th>
5           <th>Ano:</th>
6           <th>Valor:</th>
7       </tr>
8   </thead>
9   <tbody>
10      <tr>
11          <td>Volkswagen</td>
12          <td>Golf</td>
13          <td>2009</td>
14          <td>R$38.500</td>
15      </tr>
16      <tr>
17          <td>Fiat</td>
18          <td>Palio</td>
19          <td>2010</td>
20          <td>R$19.700</td>
21      </tr>
22      <tr>
23          <td>Honda</td>
24          <td>Civic</td>
25          <td>2012</td>
26          <td>R$61.500</td>
27      </tr>
28  </tbody>
29  <tfoot>
30      <tr>
31          <td colspan="4">O melhor preço da região</td>
32      </tr>
33  </tfoot>
34</table>
```


Marca:	Modelo:	Ano:	Valor:
Volkswagen	Golf	2009	R\$38.500
Fiat	Palio	2010	R\$19.700
Honda	Civic	2012	R\$61.500
O melhor preço da região			

Qual o motivo da existência dessas tags?

Essas quatro tags (*th*, *thead*, *tbody* e *tfoot*) possui muito mais valor semântico do que aplicação. No entanto, quando precisamos de criar estilos diferentes em partes diferentes da nossa tabela, a utilização dessas tags facilita muito a nossa vida.

Nos primórdios da internet as tabelas eram amplamente utilizadas para construir o *layout* dos sites, porém, temos atualmente uma maneira muito mais racional de fazer isso com as **regras de estilo CCS**. Dessa forma, a recomendação é que as tabelas sejam utilizadas unicamente para o seu real propósito, que é apresentar dados tabulares.

Exercício

Crie uma página HTML em um arquivo chamado *horarios.html* que contenha uma tabela de acordo com a imagem abaixo:

Tabela	Hora da saída	Hora de chegada	Classe do ônibus	Tarifa normal	Frequência
Cidade A	06:00	14:00	Convencional	20,00	Diária
Cidade B	12:00	15:00	Convencional	10,00	Domingos
Cidade C	14:00	17:00	Leito	15,00	Diária
Cidade D	16:00	17:00	Convencional	10,00	Diária
Cidade E	18:30	20:00	Executivo	30,00	Domingos
Cidade F	20:00	23:00	Convencional	80,00	Sábados
Cidade G	21:00	23:30	Convencional	26,00	Diária
Cidade H	22:00	23:00	Executivo	16,00	Diária
Horários de ônibus					

Curso de HTML – Aula 6: Introdução a formulários

Esta é a sexta aula do Curso de Desenvolvimento Web com HTML na qual aprenderemos sobre criação de formulários. Caso ainda não tenha lido os artigos anteriores a este, segue os links:

A criação de formulários na linguagem HTML é, em minha opinião, uma das áreas mais importantes no desenvolvimento web e devemos quadruplicar a nossa atenção quando estamos aprendendo sobre esse assunto. Pode parecer um pouco de exagero da minha parte, mas sempre friso bem isso para meus alunos porque, diferentemente de outras marcações HTML que “permitem” certos erros, sem prejudicar o resultado final, na criação de formulários os erros são inadmissíveis por que afetam o funcionamento correto do formulário.

Além de ser um assunto que merece o máximo da nossa atenção, ele é bem extenso e complexo. Por essa razão, achei melhor dividir em duas partes para que não te sobrecarregue com muita informação e possamos aprender de forma efetiva a criar formulários na linguagem HTML.

Finalidade dos formulários

Em desenvolvimento web é através dos formulários que os usuários (internautas) interagem de forma muito mais dinâmica com os sites. Simplificando ao máximo o conceito, podemos dizer que os formulários recebem os dados digitados pelo usuário e depois, com o auxílio de uma [linguagem de programação](#), esse dados são utilizados e/ou armazenados em um banco de dados.

Os formulários fazem então o papel de interface do nosso sistema, no qual sua única finalidade é receber os dados do usuário e repassá-los de forma organizada para outra página contendo uma linguagem de programação, podendo essa linguagem ser em php, java, asp, etc.

Criando um formulário

Todos os formulários em HTML devem, sem exceção, possuir a tag <form> de abertura e a sua correspondente de fechamento, a tag </form>. Além disso, são imprescindíveis também dois atributos que por enquanto não nos serão úteis, mas é bom conhecermos agora para tornar mais fácil o aprendizado sobre formulários.

O primeiro atributo é o **action** que serve para definirmos o nome do arquivo que receberá os dados preenchidos no formulário. O único objetivo desse atributo é informar para aonde serão enviados os dados do formulário e esse arquivo que receberá os dados, criado em linguagem php, por exemplo, ficará responsável por analisar, processar e/ou armazenar as informações.

O segundo atributo também obrigatório em um formulário é o **method**, que serve para especificarmos a forma de envio dos dados, podendo ser do tipo *GET* ou *POST*. Segue abaixo a definição de ambos:

- **GET** – método que envia as variáveis digitadas pelo usuário pela URL, ou seja, podemos ver as variáveis sendo passadas pela URL da página de destino. Não é muito aconselhável o uso do método GET, pois ele expõe os nomes e os valores das variáveis.
- **POST** – método que envia as variáveis digitadas pelo corpo da página, sendo completamente transparente para o usuário. É o método mais aconselhável.

Depois de conhecermos um pouco sobre os atributos *action* e *method*, vamos observar um exemplo de como codificar a tag `<form>` com seus atributos obrigatórios.

```
1<form action="gravar_dados.php" method="post" >
2// campos do formulário
3</form>
```

Recapitulando

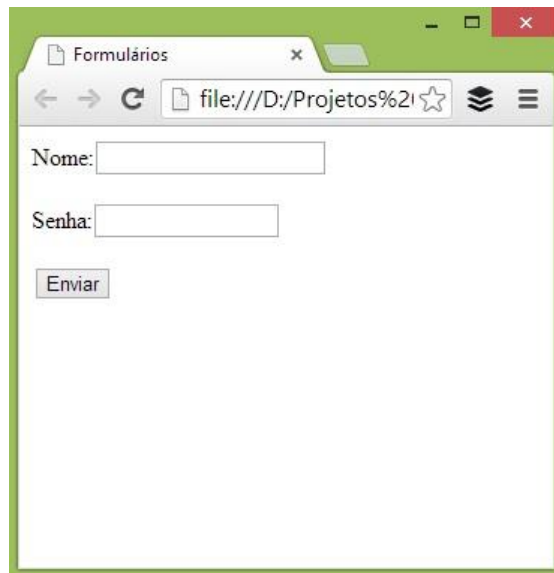
Utilizamos o atributo **action** para informar que os dados do formulário devem ser passados para o arquivo “gravar_dados.php”, que está localizado na mesma pasta do arquivo do formulário.

Já no atributo **method**, informamos que os dados do formulário devem ser enviados pelo método *POST*, ou seja, pelo corpo da página.

Poderíamos ter definido o valor *GET* ou invés de *POST* no atributo **method**. Assim, os valores preenchidos nos campos do formulário seriam enviados pela URL do navegador. No entanto, devemos ter cuidado ao utilizar o método *GET*, pois, além de uma série de outras implicações, nem sempre queremos que algumas informações sejam exibidas na URL como, por exemplo, uma senha.

Campos de entrada de dados

Tal como a [tag <table> que aprendemos na aula anterior](#), a tag `<form>` sozinha também não faz nada. Ela serve apenas para informar o início e o término de um formulário e, todas as tags que aprenderemos desse ponto em diante, devem estar posicionados dentro da tag `<form>`, para que os dados sejam enviados corretamente para o arquivo definido no atributo *action*. São vários elementos possíveis para trabalharmos com formulários e, entre os mais comuns que aprenderemos nessa aula estão: **campo de texto**, **campo de senha** e **botão de envio**. Para aprendermos a criar um formulário simples que contenha os elementos citados acima, utilizaremos como exemplo uma tela de *login* semelhante à da imagem abaixo:

A screenshot of a web browser window titled 'Formulários'. The address bar shows a file path: 'file:///D:/Projetos%20...'. The page content includes a form with two text input fields. The first field is labeled 'Nome:' and the second is labeled 'Senha:'. Below these fields is a button labeled 'Enviar'.

Para criarmos esse formulário com tela de *login* precisamos, é claro, começar com a [estrutura básica de uma página web](#) e dentro dela inserir a nossa tag `<form>` conforme exemplo já citado. Em seguida, precisamos acrescentar outros elementos (tags) que veremos a seguir.

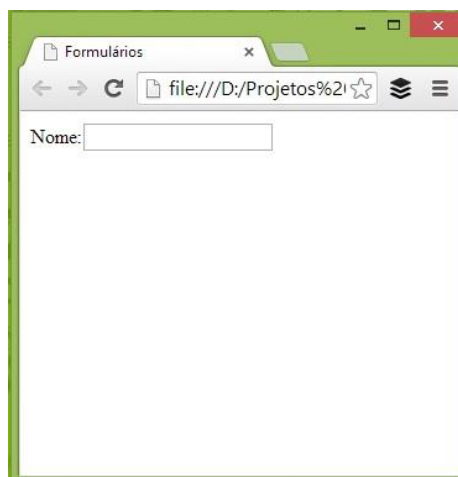
Campo de texto

A tag `<input>` é uma tag com a qual podemos criar vários tipos de campos de entrada alterando apenas o atributo ***type*** que, como o próprio nome já diz, serve para informar o tipo de campo que desejamos criar. Para nosso exemplo, vamos começar criando um **campo de texto** simples com a tag `<input>`, definindo no atributo *type* o valor "text". Além disso, precisamos também definir o atributo ***name*** que será usado pelo "programa" que receberá os dados do formulário. Veja abaixo no exemplo:

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

Observação: colocamos antes da tag `<input>` um parágrafo para que tenhamos o efeito de rótulo na caixa de texto no qual o usuário irá digitar seu nome.

Veja na imagem abaixo como o navegador exibe o código do exemplo acima.

A screenshot of a web browser window titled 'Formulários'. The address bar shows the same file path. The page content shows the rendered output of the code: a text input field with the label 'Nome:'.

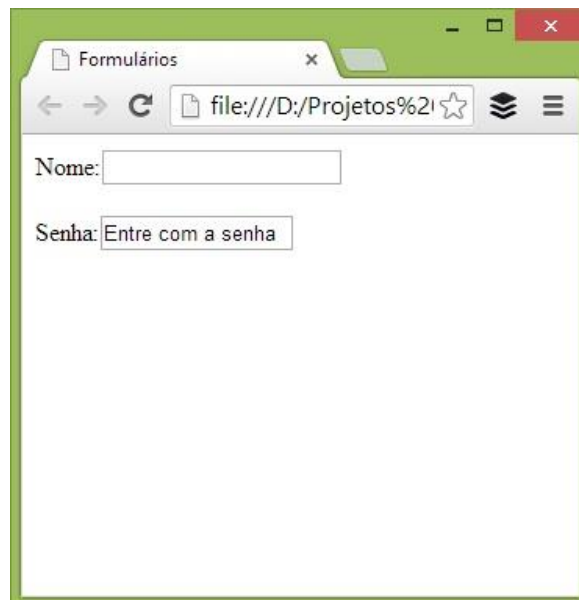
Além dos atributos *type* e *name*, existem outros que deixam o nosso formulário muito mais sofisticado e profissional, alguns deles são:

- **size** – define o tamanho do campo da caixa de texto que será exibido pelo navegador;
- **maxlength** – define a quantidade máxima de caracteres permitida no campo de texto;
- **value** – é utilizado quando há necessidade de se pré-definir um texto para o campo, podendo este valor normalmente ser alterado pelo usuário.

Veja abaixo um exemplo da tag <input> com esses novos atributos:

```
1<p>Senha:<input type="text" name="senha"  
2      size="15" maxlength="5" value="Entre com a senha"/></p>
```

Veja agora na imagem abaixo como o navegador exibe este código para entendermos melhor alguns detalhes.



A primeira alteração que podemos notar é o tamanho menor do campo senha que está assim por que definirmos o valor "15" ao atributo *size*. Quando não definirmos um valor, o navegador se encarrega de definir um valor padrão como podemos observar no campo nome.

Outra coisa que notamos é que o campo senha vem com a mensagem "Entre com a senha" no seu interior. Nesse caso, precisamos apagar este texto para digitar a senha. O atributo responsável por essa característica é o *value*.

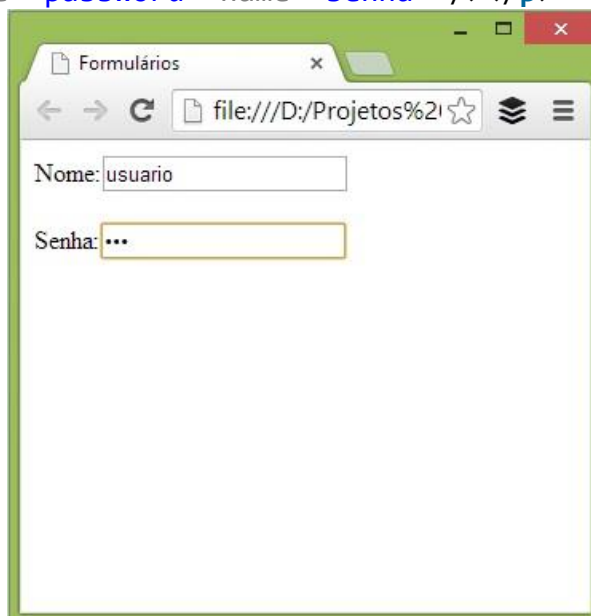
Embora o campo da senha possua o atributo *size* definido em 15, se tentarmos digitar qualquer coisa dentro desse campo, conseguiremos inserir apenas 5 caracteres, sejam eles letras, números ou ambos. Isso acontece por que o atributo *maxlength* permite que o campo receba no máximo os cinco caracteres.

Campo de senha

Podemos utilizar o campo de texto para o usuário entrar com uma senha conforme vimos no exemplo anterior. Porém, isso não é muito comum, pois temos uma forma específica de criar um campo para o nosso usuário entrar com senhas. A tag utilizada é a `<input>`, mas o valor do atributo `type` nesse caso será `"password"`. Dessa forma, quando o usuário digitar a senha, os caracteres serão substituídos por asterisco (*).

Veja no exemplo abaixo o código e em seguida uma imagem da tela.

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



Observações:

1. O campo senha não possui nenhum tipo de criptografia, apenas coloca uma máscara (o asterisco) no texto inserido;
2. os atributos `size` e `maxlength` funcionam da mesma forma na tag `<input>` quando utilizamos o valor `"password"` no atributo `type`.

Botão de envio

Qualquer formulário necessita de um elemento que dispare uma ação e, normalmente, isso é feito através de um botão de envio criado também com a tag `<input>` mas, nesse caso, o valor do atributo `type` para a ser `"submit"`.

Este botão de envio (submit) tem como objetivo passar os valores digitados no formulário para a página definida no atributo `action` da tag `<form>`. Veja no exemplo abaixo uma maneira simples de criar um botão enviar.

```
1 | <input type="submit" name="enviar" value="Enviar" />
```

Nesse caso não precisamos colocar a tag `<input>` entre parágrafos já que no botão enviar não teremos normalmente um rótulo. Entretanto, vimos com frequência que a descrição do botão está no próprio botão e, para isso, precisamos definir no atributo *value* o texto que desejamos que apareça no botão.

Tela de *login*

Apenas com as tags `<form>`, `<input>` e os atributos que aprendemos nessa aula, já é possível criar alguns formulários bem simples. Segue agora o código completo da tela de *login* apresentada no início dessa aula para que possamos ter mais um exemplo para realizarmos o exercício a seguir.

```
1 <html>
2 <head>
3     <title>Formulários</title>
4 </head>
5 <body>
6 <form action="gravar_dados.php" method="post">
7     <p>Nome:<input type="text" name="nome" /></p>
8     <p>Senha:<input type="password" name="senha" size="15" /></p>
9     <input type="submit" name="enviar" value="Enviar" />
10</form>
11</body>
12</html>
```

Curso de HTML – Aula 7: Criando formulários elaborados

Esta é a sétima aula do Curso de Desenvolvimento Web com HTML na qual continuaremos a aprender sobre criação de formulários HTML. Caso ainda não tenha lido os artigos anteriores a este, segue os links:

Nesta aula daremos continuidade no assunto abordado no último artigo e aprenderemos a fazer um formulário mais elaborado, adicionando novos elementos para oferecer mais interatividade na nossa página web.

Observação: Para todos os exemplos desta aula, utilizaremos o código abaixo que contem a estrutura básica para a criação de um formulário em uma página web.

```
1 <html>
2 <head>
3     <title>Formulário elaborado</title>
4 </head>
5 <body>
```

```
6      <form action="gravar.php" method="GET">
7
8      </form>
9 </body>
10</html>
```

O elemento ComboBox

O primeiro elemento que tornará o nosso formulário mais interativo é o **ComboBox**, que permite ao usuário escolher um item de uma lista pré-definida, ou seja, você oferece para seu usuário algumas opções e ele escolhe uma.

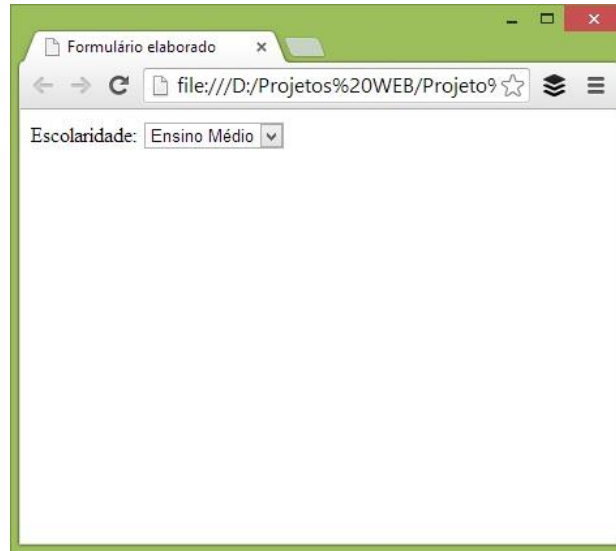
Para criar uma lista em um ComboBox é necessário a utilização da tag **<select>** e, para cada item dessa lista, é necessário uma tag **<option>**. O texto de cada item da lista que será exibido para o usuário deve ser especificado entre as tags **<option>** e **</option>**. Porém, o valor que será armazenado no banco de dados deve ser informado no atributo *value*. Vamos a um exemplo para entendermos melhor como funciona o código.

```
1<p>Escolaridade:
2<select name="escolaridade">
3    <option value="em"> Ensino Médio </option>
4    <option value="nt"> Nível Técnico </option>
5    <option value="ns"> Nível Superior </option>
6</select>
7</p>
```

Explicando cada linha do código

1. Na primeira linha temos a tag **<p>** apenas para criarmos o rótulo “Escolaridade:” na frente do ComboBox;
2. Na segunda linha estamos informando ao navegador que vamos criar uma lista de itens que será identificada por “escolaridade” através do atributo *name*;
3. Na terceira linha estamos criando o primeiro item que compõe a nossa lista. Para isso, abrimos a tag **<option>**, definimos um valor para esse item através do atributo *value*, colocamos o texto que irá aparecer para o usuário e, por fim, fechamos com a tag **</option>**;
4. As duas linhas seguintes seguem o mesmo princípio da anterior. E, podemos acrescentar quantas linhas forem necessárias para preencher a nossa lista de opções no ComboBox;

5. Na sexta linha estamos finalizando a nossa lista de itens e na última, o parágrafo criado no rótulo “Escolaridade:”.



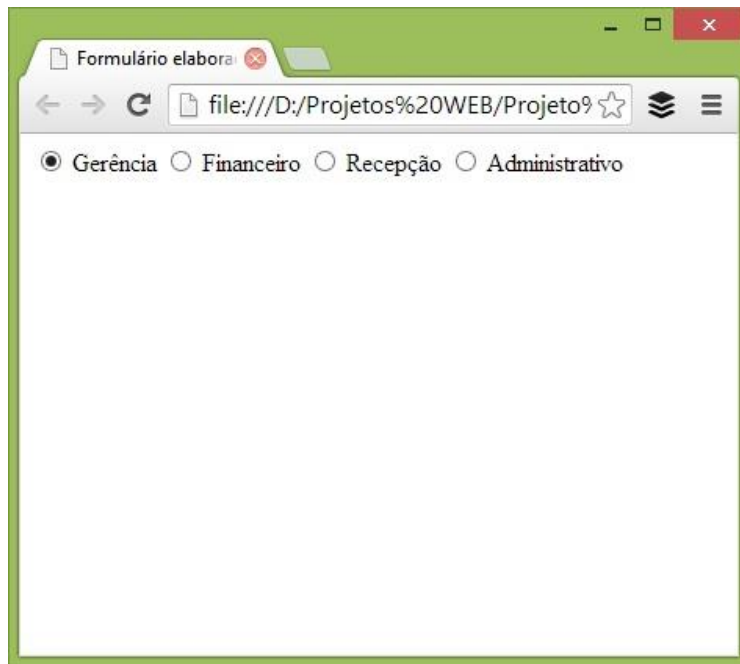
O elemento “Botão Radio”

O elemento “**botão rádio**” é utilizado quando precisamos oferecer múltiplas escolhas para o usuário e ele só pode escolher uma única opção. Para que o navegador saiba as opções que fazem parte do mesmo grupo, e permita que só uma seja selecionada, basta definir o mesmo nome no atributo *name* de cada tag **<input type=“radio”>**.

Além do atributo *name*, é preciso também o *value* para definirmos o valor associado à opção escolhida pelo usuário e o *checked*, em apenas uma tag **<input type=“radio”>** do grupo, pois essa opção será a *default*, ou seja, caso o usuário não escolha nenhuma opção, essa marcada com *checked* será o valor padrão.

Veja abaixo um exemplo para a criação de um grupo de **Botões Radio**.

```
1<input type="radio" name="cargo" value="1" checked /> Gerência
2<input type="radio" name="cargo" value="2" /> Financeiro
3<input type="radio" name="cargo" value="3" /> Recepção
4<input type="radio" name="cargo" value="4" /> Administrativo
```



Observação: Todos os atributos *name* do grupo devem conter o mesmo nome. Em contrapartida, no atributo *value*, os valores precisam ser diferentes, pois são eles que serão gravados posteriormente no banco de dados da aplicação.

O elemento “Botão CheckBox”

Este elemento também é utilizado para oferecer múltiplas opções para o usuário, mas com a diferença que no “**Botão CheckBox**” é possível escolher várias opções dentro de um grupo. Para tanto, cada opção deve possuir um atributo *name* independente.

O atributo *value* funciona da mesma forma que no elemento “Botão Radio”. Já o atributo *checked* é opcional.

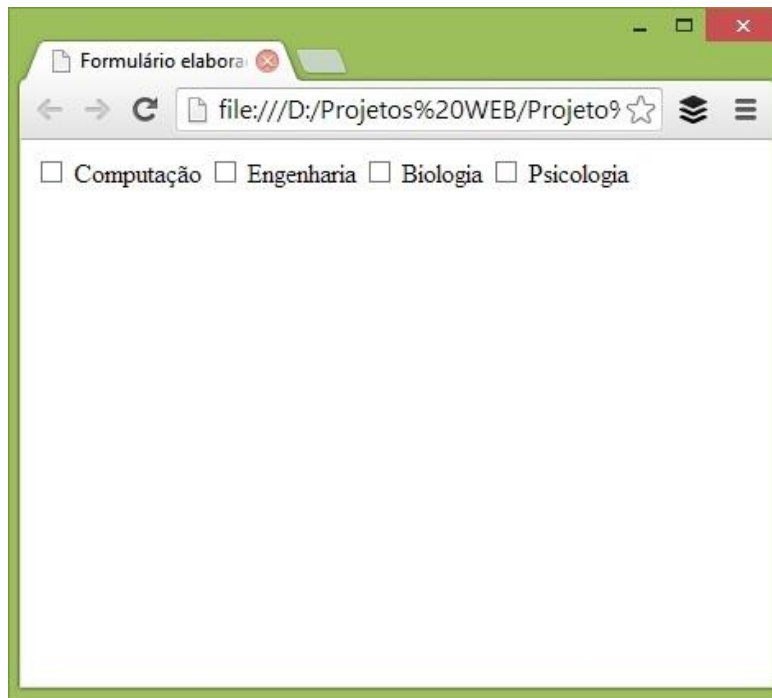
Veja abaixo um exemplo para a criação de um grupo de **Botões CheckBox**.

1<input type="checkbox" name="area1" value="com" /> Computação

2<input type="checkbox" name="area2" value="eng" /> Engenharia

3<input type="checkbox" name="area3" value="bio" /> Biologia

4<input type="checkbox" name="area4" value="psi" /> Psicologia



O Botão Reset

O **"Botão Reset"** é utilizado para "limpar" todas as alterações realizadas no formulário e voltar os campos para os valores *default*, ou seja, os valores iniciais especificados nos atributos *value* de cada um dos elementos.

Veja abaixo um exemplo para a criação de um **Botão Reset**.

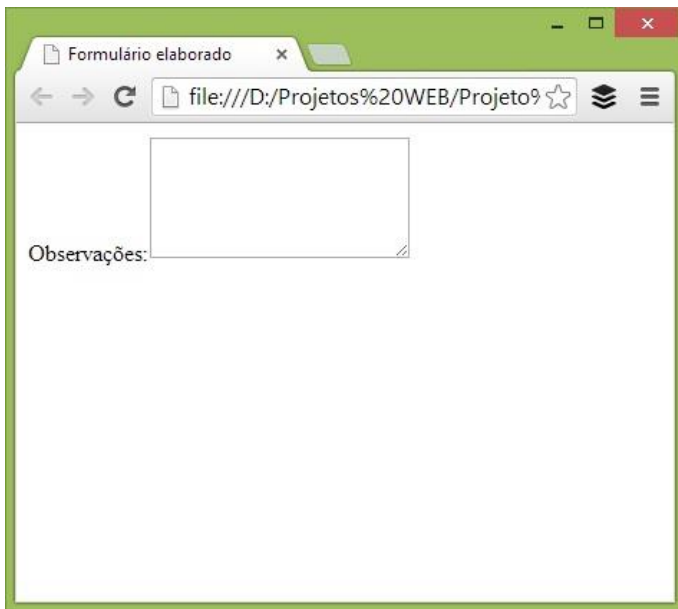
```
1 |<input type="reset" value="Limpar" />
```

O elemento "Área de Texto"

A tag `<textarea>` exibe uma **área de texto** na qual o usuário poderá inserir um texto qualquer. A diferença desta tag em relação à tag `<input type="text">` é que a `<textarea>` permite a inserção de textos longos e com quebras de linha.

Neste elemento é necessário especificar os atributos *cols* e *rows*, que são utilizados para definir o número de caracteres nas colunas e o número de linhas. Outro detalhe importante é que não existe neste elemento o atributo *value*. Dessa forma, caso seja necessário definir um texto inicial, ele deve estar entre as tags `<textarea>` e `</textarea>`. Vejamos agora um exemplo para a criação de uma **área de texto**.

```
1|<p>Observações:<textarea name="obs"  
2|                                cols="20" rows="5"></textarea></p>
```



As tags <label>, <fieldset> e <legend>

As tags <label>, <fieldset> e <legend> são utilizadas exclusivamente nos formulários e diferem um pouco das tags que estudamos até o momento. Basicamente, elas funcionam como marcações e rótulos para os elementos que criamos em nossos formulários.

No primeiro exemplo dessa aula nós utilizamos o texto “Escaridade” em uma tag <p> para criar o rótulo do ComboBox. Na prática isso funciona, mas seria mais correto substituir a tag <p> pela <label> que foi criada justamente para essa finalidade. Veja abaixo como ficaria o primeiro exemplo modificado com a tag <label>.

```
1<label>Escaridade:
2<select name="escalaridade">
3    <option value="em"> Ensino Médio </option>
4    <option value="nt"> Nível Técnico </option>
5    <option value="ns"> Nível Superior </option>
6</select>
7</label>
```

As tags <fieldset> e <legend> trabalham em conjunto e servem para criarmos uma divisão dos elementos dentro do nosso formulário. Imagine, por exemplo, que você precise criar um formulário de cadastro com diversos campos para o usuário preencher. Em situações como essa é muito provável que você consiga separar os campos por categorias, podendo ser, por exemplo, dados pessoais e dados profissionais. Vamos a um exemplo bem simples para compreendermos essa ideia.

```

1 <p>Dados pessoais</p>
2   <label>Nome:<input type="text" name="nome" /></label><br />
3   <label>Idade:<input type="text" name="idade" /></label><br />
4 <p>Dados profissionais</p>
5   <label>Escolaridade:
6   <select name="escolaridade">
7       <option value="em"> Ensino Médio </option>
8       <option value="nt"> Nível Técnico </option>
9       <option value="ns"> Nível Superior </option>
10  </select>
11  </label>
12  <br />
13  <label>Área de atuação
14      <input type="radio" name="cargo" value="1" checked> Gerência
15      <input type="radio" name="cargo" value="2" > Financeiro
16      <input type="radio" name="cargo" value="3" > Recepção
17  </label>

```

Se observarmos com muita atenção, percebemos que no exemplo acima temos uma separação entre os dados pessoais e profissionais. Entretanto, existe uma forma muito mais correta e elegante de fazer essa separação com as tags `<fieldset>` e `<legend>`.

A tag `<fieldset>` é utilizada para criar uma seção, separando um conjunto de elementos do formulário com uma linha ao redor. Já a tag `<legend>` é utilizada para criar a legenda de cada seção, neste caso, o texto que aparece na parte superior de cada conjunto.

Vamos então modificar o exemplo anterior utilizando agora as tags `<fieldset>` e `<legend>`.

```

1 <fieldset>
2     <legend>Dados pessoais</legend>
3     <label>Nome:<input type="text" name="nome" /></label><br />
4     <label>Idade:<input type="text" name="idade" /></label><br />
5 </fieldset>
6 <fieldset>
7     <legend>Dados profissionais</legend>
8     <label>Escolaridade:
9     <select name="escolaridade">
10         <option value="em"> Ensino Médio </option>
11         <option value="nt"> Nível Técnico </option>
12         <option value="ns"> Nível Superior </option>
13     </select>
14 </label>
15 <br />
16 <label>Área de atuação
17     <input type="radio" name="cargo" value="1" checked> Gerência
18     <input type="radio" name="cargo" value="2" > Financeiro
19     <input type="radio" name="cargo" value="3" > Recepção
20 </label>
21</fieldset>

```

Formulário elaborado

file:///D:/Projetos%20WEB/Projeto9

Dados pessoais

Nome:

Idade:

Dados profissionais

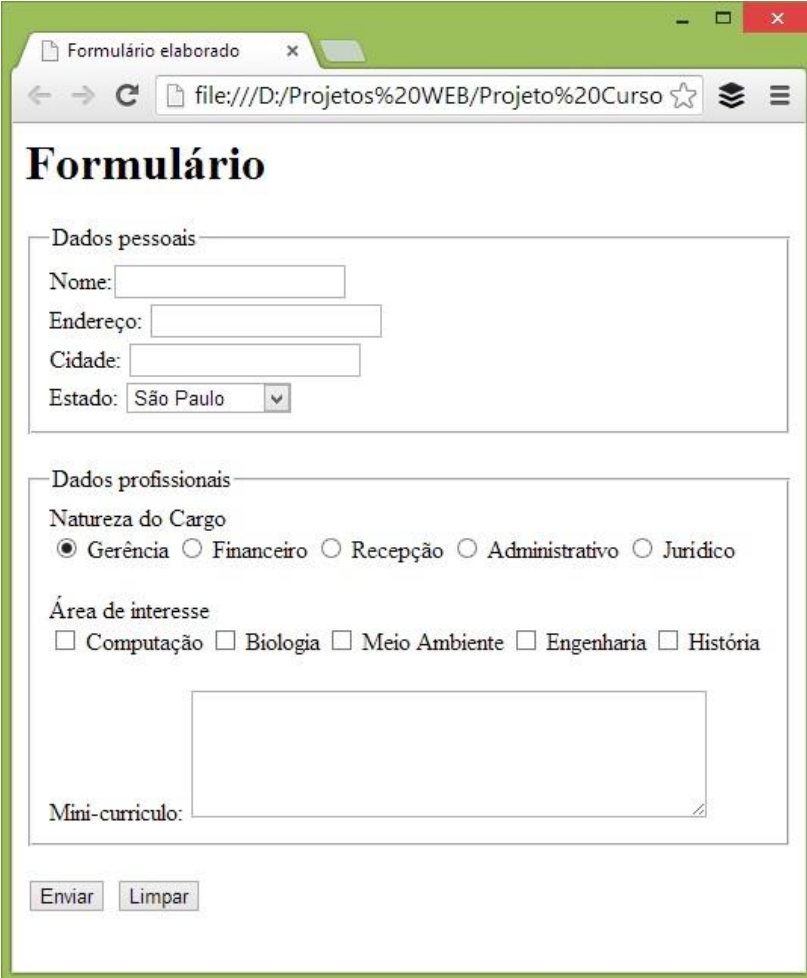
Escolaridade:

Área de atuação ☒ Gerência ☐ Financeiro ☐ Recepção

Podemos observar que o nosso formulário ficou com uma melhor divisão e uma aparência muito mais interessante, mesmo sem termos ainda utilizado a estilização das CSS ([folha de estilo em cascata](#)).

Exercício:

Seguindo o modelo da imagem abaixo, crie um formulário utilizando os elementos que aprendemos nessa sétima aula do curso de desenvolvimento web com HTML.



The image shows a web browser window with a single tab titled 'Formulário elaborado'. The address bar shows a local file path: 'file:///D:/Projetos%20WEB/Projeto%20Curso'. The form itself is titled 'Formulário' in a large, bold font. It is divided into two main sections: 'Dados pessoais' and 'Dados profissionais'. The 'Dados pessoais' section contains four input fields: 'Nome:', 'Endereço:', 'Cidade:', and 'Estado:'. The 'Estado:' field is a dropdown menu currently showing 'São Paulo'. The 'Dados profissionais' section contains a group of radio buttons for 'Natureza do Cargo' with options: 'Gerência' (selected), 'Financeiro', 'Recepção', 'Administrativo', and 'Jurídico'. Below this is a group of checkboxes for 'Área de interesse' with options: 'Computação', 'Biologia', 'Meio Ambiente', 'Engenharia', and 'História'. At the bottom of this section is a large text area labeled 'Mini-curriculo:'. At the very bottom of the form are two buttons: 'Enviar' and 'Limpar'.

Formulário elaborado

file:///D:/Projetos%20WEB/Projeto%20Curso

Formulário

Dados pessoais

Nome:

Endereço:

Cidade:

Estado:

Dados profissionais

Natureza do Cargo

☒ Gerência ☐ Financeiro ☐ Recepção ☐ Administrativo ☐ Jurídico

Área de interesse

☐ Computação ☐ Biologia ☐ Meio Ambiente ☐ Engenharia ☐ História

Mini-curriculo:

Curso de HTML- Aula 8: Meta tags e perguntas frequentes

No artigo de hoje do nosso [curso de HTML](#) aprenderemos sobre as meta tags utilizadas na seção *head* da nossa página web. Dessa forma, todas as tags que veremos a seguir devem estar entre as tags `<head>` e `</head>`.

O que são meta tags?

São parâmetros dentro do **html** que foram criados para apontar algumas informações importantes a respeito do conteúdo da página web.

Exemplos de meta tags

```
1 | <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
```

Meta tag utilizada para informar qual o conjunto de caracteres que será utilizado dentro da página web. No *charset*, você define a codificação do conteúdo, sendo os mais comuns o UTF-8 e iso-8859-1.

```
1 | <meta http-equiv="content-language" content="pt-br" />
```

Meta tag utilizada para apontar a linguagem do conteúdo da página. Ajuda bastante as ferramentas de busca que precisam direcionar a página no idioma a que ela se refere. (pt-br: Português do Brasil; en: Inglês; es: Espanhol)

```
1 | <meta name="author" content="Júnior Gonçalves" />
```

Meta tag utilizada geralmente para informar o nome do autor do conteúdo inserido na página.

```
1 | <meta name="keywords" content="css, html, php, webstandards" />
```

Esta meta tag era utilizada antigamente para informar às [ferramentas de busca](#) quais eram as palavras-chaves da página web. Entretanto, já é de conhecimento comum que os buscadores deixaram a muito tempo de levar essa informação como fator de ranqueamento. Mesmo assim, costumamos utilizar essa tag por saudosismo, colocando no máximo nove palavras referente ao assunto abordado na página em questão.

```
1 | <meta name="description" content="Descrição do conteúdo da página" />
```

Esta meta tag é utilizada para definirmos uma breve descrição do conteúdo da nossa página, sendo essencial limitar-se a no máximo 180 caracteres. Essa meta tag é importante por que

está no conjunto de fatores analisados pelas ferramentas de busca para verificar o conteúdo e a qualidade de uma página. Além disso, é interessante também dar atenção a essa meta tag, pois, quando compartilhamos um link no Facebook, por exemplo, o conteúdo dessa meta tag é exibido logo abaixo do link.



Exemplo da utilização da meta tag description

Para finalizarmos a 8ª aula do nosso curso, listarei abaixo algumas perguntas frequentes realizadas pelos alunos.

Perguntas frequentes

1) As tags devem ser escritas com letras maiúsculas ou minúsculas?

Para a maioria dos navegadores é indiferente se você usa maiúscula, minúscula ou mesmo uma mistura delas. Entretanto, a maneira profissional de criar páginas web seguindo os [padrões de desenvolvimento da W3C](#), é utilizando sempre letras minúsculas.

2) Posso usar várias tags simultaneamente?

Sim, você pode usar quantas tags queira desde que as aninhe convenientemente. Veja como fazer isto no exemplo abaixo:

1 | `<i>Texto em negrito e itálico.</i>`

Observe que no exemplo a primeira tag de abertura `` corresponde a última tag de fechamento ``. Ou seja, as últimas tags a serem abertas têm que ser as primeiras a serem fechadas, e as primeiras a serem abertas terão que ser as últimas a serem fechadas.

3) Qual a diferença entre HTM e HTML?

Ambas as extensões são a mesma coisa. O que acontece é que antigamente as extensões em computadores MS-DOS possuíam apenas três letras e com o advento do Windows, essas três letras deixaram de ser uma regra e passou-se a utilizar como padrão a extensão `“.html”`.

4) Sempre terei que usar `` e `` juntos?

Sim. Você sempre deve usar `` e `` juntos ou `` e ``. Nenhum desses elementos faz sentido sem o outro. Lembre-se, uma lista é na verdade um grupo de itens: o elemento `` é usado para identificar cada item, e o elemento `` é usado para agrupá-los.

5) Posso colocar texto ou outros elementos dentro de um elemento ou ?

Não. Os elementos e foram criados para funcionarem apenas com o elemento

6) Qual a diferença entre as tags <tr> e <td>?

A tag <tr> significa “table row” (linha de tabela) e indica onde começa e termina e uma linha horizontal na tabela e a tag <td> significa “table data” (dados da tabela), indicando onde começa e termina cada célula contida nas linhas da tabela.

7) O que é atributo?

Atributos são escritos dentro da tag, seguidos por um sinal de igual e depois entre aspas são declaradas as informações do atributo.