INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA BA - IFBA

APOSTILA: HTML5

PROFESSOR: ADDSON ARAÚJO DA COSTA

VALENCA - BA

Sumário

Lista de Figuras

1. Introdução

Muitos pensam que é difícil criar um website e correm para soluções que entregam o site pronto através de gerenciadores de conteúdo como o Wix, mas os sites ficam todos iguais, sem qualidade gráfica servindo apenas para projetos iniciantes. Outros imaginam ser necessário softwares caros como Dreamweaver, Photoshop ou Ilustrator, mas tudo o que você precisa é um browser (Firefox, Internet Explorer, Chrome, …) e um programa editor de textos compatível com programação (Geany, Atom, VS Code, ...).

O objetivo dessa apostila é fornecer os conhecimentos básicos partindo do zero. Não é requerido qualquer conhecimento prévio e em breve você verá como é divertido criar websites. Não se aborreça se ficarem feios, pois estamos apenas iniciando nosso curso.

Para criar um website, é necessário o uso da linguagem HTML, não existindo outro meio pois essa é a única linguagem aceita pelos browsers. Para ver o que estamos falando, acesse uma página WEB qualquer de sua escolha e aperte ctrl + U. Esse código exibido pelo browser é o código HTML do site e o navegador o exibe de forma gráfica de acordo com alguns padrões, por exemplo, o código abaixo serve para exibir a palavra “Exemplo” em negrito.

```
<b> Exemplo </b>
```

Exemplo 1: Negrito

Para testar o exemplo acima, crie no seu editor de textos favorito, um arquivo com o conteúdo “ Exemplo ” onde o nome do arquivo é “*Exemplo.html”* e abra esse arquivo em um browser. O resultado deverá ser:

Exemplo

Nessa apostila, todos os códigos com fundo cinza devem ser testados usando essa forma descrita acima e nos pontos onde não há código, está sendo descrito o que deve ser feito. É necessário testar cada *tag*, cada atributo, cada combinação possível pois só é possível aprender a nossa disciplina com prática. A abordagem teórica dessa apostila é restrita às primeiras páginas.

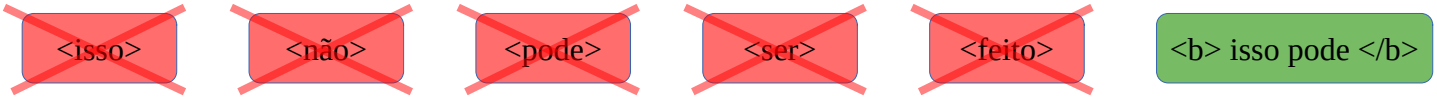
2. Tags

No HTML, como vimos no exemplo anterior, são usados marcações especiais para definir como é visto o que está escrito. Essas marcações seguem um padrão e são chamadas de “*tags*” (do inglês, significa etiqueta/marcação).

Para escrever uma *tag* de forma correta é preciso escrever onde inicia e onde acaba. No exemplo anterior, a *tag* inicia onde está escrito “” e acaba onde está escrito “”. O conteúdo entre a *tag* de abertura e *tag* de fechamento é o que será afetado por ela.

Para que o código seja válido, note que a *tag* tem um nome bem específico onde “B” significa negrito. Este padrão é regulamentado pelo *World Wide Web Consortium* – W3C. Assim, não é possível definirmos nossas próprias *tags* de acordo com nosso gosto ou desejo.

Existem inúmeras além da *tag* B (negrito) que aprenderemos ao longo da apostila. Cada uma tem seu papel e significado. Após terem sido aprovadas pelo W3C, elas foram implementadas nos navegadores, assim, é impossível inventarmos as nossas.



2.1 Tag de abertura e tag de fechamento

O padrão para abrir uma *tag* é **<***NomeDaTag***>** e para fechar é **</***NomeDaTag***>**. A diferença

entre elas está na presença da barra “/” na *tag* de fechamento logo após o sinal de menor “<”. Tudo que estiver entre a *tag* de abertura e fechamento será processado de acordo com a definição da *tag*.

Note que não está sendo ensinado usando espaços em branco dentro da *tag* de abertura nem de fechamento, o único caso que é permitido uso de espaço em branco é após o nome da *tag*, nunca antes. Abaixo, veja em vermelho casos errados e em verde os casos corretos.

2.2 Tags vazias

Existe também *tags* que não podem ter uma *tag* de fechamento, tendo apenas uma *tag* de abertura. Estas são poucas e ao longo desta apostila será demonstrado apenas três casos. A quebra de linha “BR” (do inglês, *Break Row*, significa quebra de linha), “HR” (do inglês, *Horizontal Row*, linha horizontal) e tag “IMG” (do inglês image, imagem).

Nesse caso, é usado essa outra forma de escrita porque elas não necessitam de conteúdo a processar, são comandos isolados e independentes como, por exemplo, a *tag* para criar uma quebra de linha cujo código está abaixo. Nesse caso, a quebra de linha funciona independentemente de qualquer texto a processar, ela simplesmente cria uma quebra de linha na posição onde estiver inserida. O mesmo ocorre com a *tag* de criar linha horizontal. Ela simplesmente cria uma linha, não sendo possível especificar um texto a ser processado ou inserido junto a linha.

```
<br>
<hr>
```

Exemplo 2: Quebra de linha

Note que existem *tags* que são escritas no padrão de abrir e fechar em *tags* diferentes como, por exemplo, o negrito. Ao mesmo tempo existem *tags* vazias que não necessitam de tag de fechamento. A *tag* de negrito sempre será escrita em duas *tags* e a *tag* de quebra de linha ou linha horizontal sempre será escrita em *tag* única, não sendo possível escolher.

3. HTML

O HTML é um conjunto de *tags*, cada uma com sua função. Usando-as juntas podemos obter qualquer resultado visual desejado. O padrão tem uma forma correta de escrita e poderá ter resultados diferentes em cada browser, pois é o browser o responsável por interpretar o HTML, podendo cada um fazer sua interpretação de forma diferente, contudo existe um padrão aceito pela grande maioria dos browsers que aprenderemos aos poucos.

O padrão HTML não define se os nomes das *tags* devem ser escritos em minúsculo ou maiusculo, porém no padrão XHTML (última versão antes do HTML5, versão atual) é requerido utilizar letras minusculas para maior legibilidade e é uma boa prática manter este padrão em qualquer código.

O HTML usa algumas *tags* padrão que devem estar presentes só uma vez no seu código. Elas são mostradas abaixo no exemplo. Usamos a *tag* “html” para identificar onde começa e acaba o documento.

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

Exemplo 3: Tags Estruturais

Em todos os códigos é necessário o uso das *tags* acima uma vez que cada uma tem sua função como será mostrado abaixo. Essas *tags* devem estar na hierarquia mostrada no exemplo acima, pois não faria sentido nenhuma outra combinação. Isto não apenas por significarem cabeça, corpo e sabermos a posição desses itens no corpo humano, mas no HTML isto torna mais rápido e eficiente o processamento para que a página carregue de forma correta seu conteúdo, uma vez que em cada uma dessas *tags* contem um pedaço específico do documento.

3.1 Cabeça e Corpo

A *tag* “head” (cabeça) é usada como se fosse o cérebro do site, isto é, contém informações que não são diretamente visíveis como, por exemplo, elementos de definição como CSS (estética), que estudaremos em nossa próxima apostila, importação de elementos de frameworks/bibliotecas, título do site, entre outros. Note que nunca devemos, na *tag* head, inserir *tags* que geram efeito visual diretamente, mas sim itens que alteram visual de algo ou importam informações externas.

No “body” (corpo) deve conter a parte visível, ou seja, todas as *tags* que geram efeito visual na tela. Isto abrange quase todas como, por exemplo, a *tag* “b” que representa o negrito e a *tag* “br” que efetua a quebra de linha. Veremos mais *tags* no próximo tópico, todas elas deverão ser colocadas dentro do “body” (corpo), podendo ser combinadas de acordo com seu desejo.

Existe ainda no HTML4 a *tag* “foot” (pé) onde devemos inserir os códigos que geram movimento/animações (criados em Javascript/Jquery). Contudo esta *tag* foi removida do padrão XHTML e HTML 5, por isso não será usada além de que não usaremos Javascript nesse início de curso.

Aqui é comum confundir as *tags* header e footer com as apresentadas aqui que foram head e foot. Header e footer não são a cabeça e pé, e sim o cabeçalho e o rodapé e são usadas para propósitos bem diferentes, isto é, topo e rodapé do site e por ter elementos visíveis elas são colocadas dentro do body, mas isto é conteúdo de HTML5 que veremos a frente. A confusão ocorre por alunos que buscam conteúdo além dessa apostila, por isso, a dúvida já removida.

3.2 Comentários no código

Para que sejam escritos comentários, é usado uma notação que parece uma *tag*, mas não segue o padrão explicado no item 2 dessa apostila pois poderia confundir com *tags* de elementos. O padrão para um trecho de comentário no HTML é usando “<!-- Comentário -->”. Aqui o comentário é iniciado com “<!--” e encerrado com “-->”.

```
<html>
        <head>
                <title> Minha primeira página!! </title>          <!-- Aqui é um elemento não visual -->
        </head>
        <body>
                <b> Funciona!!! \o/ </b>                          <!-- Aqui é um elemento visual -->
        </body>
</html>
```

Exemplo 4: Primeira página

4. Identação

Quando colocamos conteúdo dentro de uma *tag* no HTML usamos a idéia que a *tag* passa a ser pai de seu conteudo e o conteúdo passa a ser filho desta *tag*.

Esse conceito será muito importante mais a frente. Por enquanto ele será útil para identarmos nosso arquivo corretamente. Identação é o uso do recuo da linha em relação a sua margem.

Deixar esse espaçamento é muito importante pois, nossos códigos, aos poucos, ficarão maiores e precisaremos identificar rapidamente qual elemento é filho de qual. Dessa forma usamos a regra de que todo elemento filho estará um recuo mais a direita que seu pai.

Para realizar esses recuos de forma rápida, ao invés de usarmos a barra de espaço do teclado, devemos usar o botão Tab, localizado a esquerda do teclado ao lado da letra “Q”. Note também que a *tag* de abertura e fechamento dos pais devem estar com mesma distância de recuo.

Em muitos editores de texto, ao selecionar várias linhas e apertar “tab”, todas as linhas selecionadas irão um espaçamento para a direita e se ao invés de apertarmos “tab” (com as linhas selecionadas), apertarmos “shift” + “tab”, essas linhas perderão um espaçamento.

Verifique o Exemplo 4, acima, para notar que todas as *tags* com um espaçamento a direita, são filhos da *tag* HTML e todas as *tags* com dois espaçamentos a direita são netos da *tag* HTML, isto é, são filhos de seus filhos. Note ainda que elementos sem filhos podem ser colocados em uma só linha como, por exemplo, as *tags* B e *Title* que não contém *tags* filhas.

5. Tags básicas para manipular texto

Agora aprenderemos algumas tags usadas para manipular texto. Existem várias tags possíveis no HTML para tal, mas boa parte delas foram eliminadas do padrão HTML5 que estudaremos mais a frente, por isso, aqui será abordado apenas as tags que continuam presentes no HTML5, a versão mais atual do HTML.

5.1 Parágrafo

O parágrafo é definido usando a tag “p”. A vantagem de usar a tag de parágrafo ao invés de digitar o texto de outra forma é que após o texto de cada parágrafo, será automaticamente quebrado a linha além de ser possível definir o recuo da primeira linha no parágrafo, mas isso só será possível com CSS, que veremos em seguida. Exemplo:

```
<p> Meu primeiro parágrafo! </p>
```

Exemplo 5: Parágrafo

5.2 Cabeçalhos

No HTML existe seis tags para alterar o tamanho do texto para tamanhos pré-definidos. Estas tags são a “h1”, “h2”, “h3”, … “h6” e significam, respectivamente, cabeçalho de nível 1, 2, 3 etc, em outras palavras, texto de primeiro nível, segundo nível (menos importante) e terceiro nível (menor ainda) etc.

O objetivo destas tags é criar uma hierarquia nos tópicos existentes de conteúdo de nossa página web. Por exemplo, nesta apostila o item “5. Tags básicas para manipular texto” tem hierarquia maior que o item “5.2 Cabeçalhos”, assim, caso o item 5 seja um cabeçalho h1, o item 5.2 seria um cabeçalho h2, pois está dentro do conteúdo anteriormente citado.

Isso é importante mesmo que nossa página não esteja marcado desta forma, pois existirão seções de conteúdo em mesma hierarquia e dentro delas existirão subtópicos, portanto, existirá uma hierarquia.

Essa tag altera o tamanho do texto para configurações fixas com aspecto de negrito. Em

geral, estamos interessados em ajustar o tamanho do texto para um valor específico de tamanho. Mais para frente veremos a possibilidade de usar CSS para personalizar essas tags, mas por enquanto é interessante testar o código abaixo.

```
<h1>Cabeçalho</h1>
<h2>Subtítulo</h2>
<h3>Sub-subtítulo</h3>
```

Exemplo 6: Cabeçalhos

5.3 Itálico, Negrito e Sublinhado

Assim como podemos usar a tag “b” para transformar um texto em negrito, podemos usar a tag “i” para deixá-lo como itálico ou “u” para sublinhado. Também podemos combiná-las, experimente.

```
<i>Este texto deve ser itálico</i>
<b>Estará em negrito</b>
<u>E sublinhado</u>
```

Exemplo 7: Itálico, Negrito e Sublinhado

5.4 Listas

Para criar uma lista de marcadores, usamos a tag “ul”, porém não basta, pois após criarmos a lista de marcadores, precisamos informar quais são os itens da lista. Para definir os itens da lista, usamos a tag “li”.

A forma de usar estas tags são conforme o exemplo abaixo. Note que a lista pode ter vários itens de lista (abrindo a tag ul antes de começar a descrever os itens e fechando após todos eles) e cada item de lista tem, entre suas tags de abertura e fechamento, o conteúdo pertencente ao item.

```
<ul>
        <li>Um item de lista</li>
        <li>Outro item de lista</li>
</ul>
```

Exemplo 8: Listas

Existem também as listas numeradas. Experimente no exemplo anterior modificar o uso da tag “ul” (unordered list, do inglês lista não ordenada) por “ol” (ordered list, do inglês lista ordenada).

Lembro que dentro do “li” (item de lista), podemos ter quaisquer elementos o que inclui a possibilidade de colocar uma lista dentro de outra.

6. Atributos

Na tag de abertura podemos também especificar informações que a tag necessita para seu funcionamento. O exemplo mais simples é quando queremos criar um link. Nesse caso a tag que executa esse funcionamento é a tag “a” (a letra “a” é originada da palavra âncora, em inglês, anchor).

Para definirmos um link precisamos informar que queremos criar uma âncora (para isso usamos a tag “a”), precisamos informar qual texto ficará visível para o usuário clicar (colocamos esse texto entre a tag de abertura e fechamento) e precisamos também informar o endereço do site que será aberto quando o usuário clicar (para isso necessitamos usar um atributo para a âncora). Veja o exemplo abaixo e depois a descrição.

```
<a href='http://www.google.com'> Clique aqui! </a>
```

Exemplo 9: Âncoras

Atributos são características associadas a algum elemento. Nesse caso o elemento é a tag “a” do html, o atributo é o “href” que significa o endereço para o qual o link aponta e entre aspas temos o valor deste atributo, neste caso o endereço.

Para definirmos atributos em html usamos a seguinte notação usando a **tag de abertura**:

<NomeDaTag atributo1='valor' atributo2='valor' atributo3...>

Um outro atributo da tag “a” (âncora), é o target que define onde será aberto o link. Caso o valor de target seja “_blank”, o link abrirá em nova janela. Não confundir “_blank” com “blank”. No primeiro caso o link sempre abrirá em nova janela, no segundo caso abrirá a primeira vez em nova janela e nas demais vezes abrirá na mesma janela que já havia sido aberta previamente, como se estivesse abrindo sempre na janela que você chamou de blank, outras palavras quaisquer podem ser usadas, tendo resultado semelhante.

7. Tabelas

Tabelas são estruturas que precisam, necessariamente que exista um alinhamento semelhante a um quadriculado, ou seja, a divisão de linhas e colunas percorre toda a estrutura criando um alinhamento quadriculado no seu conteúdo. Tal forma de organização é chamada de grade (do inglês, grid).

7.1 Estrutura

É muito comum e útil uso de tabelas. Em HTML para usarmos tabelas precisaremos usar três tags. A primeira, “table”, define uma tabela. A segunda, “tr”, define uma linha da tabela. A terceira, “td”, define uma coluna da tabela. Para melhor entendimento veja o exemplo abaixo.

```
<table width='600px' height='400px' border='1'>
	<tr>
		<td width='50%'> Coluna 1, Linha 1 </td>
		<td align='right'> Coluna 2, Linha 1 </td>
	</tr>
	<tr valign='bottom'>
		<td> Coluna 1, Linha 2 </td>
		<td> Coluna 2, Linha 2 </td>
	</tr>
</table>
```

Exemplo 10: Tabelas

Note que usando essa notação podemos ter qualquer quantidade de linhas e qualquer quantidade de colunas em cada linha, teste o que ocorreria caso o número de colunas seja diferente entre as linhas.

Uma observação importante aqui é que as *tags* devem estar uma dentro da outra nesta ordem apresentada no exemplo, nunca em ordem diferente. Ou seja, a Tabela (*Table*) é sempre pai de linhas (TR). As linhas sempre são pais de colunas (TD). Esta ordem não pode ser invertida de forma nenhuma.

7.2 Atributos

- width (Pode ser usado nas *tags* “table” ou “td”)

O width define a largura da tabela ou da coluna, de acordo com a tag onde está inserido. O valor da largura pode ser um número para que possamos definir a largura em pixels ou um valor seguido do “%” para especificar uma porcentagem de tamanho em relação a largura que o elemento pode ocupar.

Veja no Exemplo 11 o uso dessas medidas. Quando usamos pixels, usamos a unidade “px”, enquanto usando porcentagem usamos o simbolo de porcentagem “%”.

- height (Pode ser usado nas *tags* “table” ou “tr”)

Semelhante ao width, porém define a altura.

- border (Pode ser usado na *tag* “table”)

O atributo border é usado diretamente na tag de abertura da tabela. Ele define se a tabela terá borda ou não. Seu valor deve ser um número maior ou igual a zero e indica a expessura da borda. Aqui não é possível ter controle em pixels nem porcentagem.

- align (Pode ser usado na *tag* “tudo”, “tr” ou “table”)

Mais um atributo útil em tabelas é o “align”, ele define o alinhamento do texto (ou imagens) dentro da coluna, isto é, caso seja usado na tag “td”. Seus valores podem ser “left” (alinhar à esquerda), “center” (centralizar) ou “right” para alinhar a direita.

- valign (Pode ser usado na *tag* “tr” e “td”)

Semelhante ao align, o valign também efetua alinhamento, porém esse alinhamento é vertical. Seus valores podem ser “top” (topo), “middle” (meio, não confundir com center), bottom (baixo) e baseline (linha base, é a linha onde o texto está alinhado).

- colspan e rowspan (Pode ser usado na *tag* “td”)

Quando se está usando tabelas em algum editor de texto ou de planilhas, temos a opção de mesclar células, ou seja, fazer com que uma célula ocupe o lugar de duas ou mais. Para fazermos isto em html definimos o número de células que serão unidas horizontalmente através do colspan e o número de células que serão unidas verticalmente através do rowspan. O valor desses atributos é o número de células a ser unidas. Veja o exemplo abaixo.

```
<table>
        <tr>
                <td> coluna 1 </td> <td> coluna 2 </td> <td> coluna 3 </td>
        </tr>
        <tr>
                <td colspan='2' rowspan='2'> Duas colunas e duas linhas </td> <td> coluna 3</td>
        </tr>
        <tr>
                <td> coluna 3</td>
        </tr>
</table>
```

Exemplo 11: Colspan e Rowspan

8. Imagens

Uma página HTML sem imagens é uma página sem vida. Para colocar imagens usaremos a tag “img”. Essa tag tem a notação abaixo, onde o atributo “src” informa onde está a imagem a ser exibida. Exemplo:

```
<img src='nome_da_imagem.jpg'>
<img src='http://html.net/site/graphics/logo.png'>
```

Exemplo 12: Imagens

Caso a imagem esteja na mesma pasta que seu arquivo HTML, basta colocar no “src” o nome da sua imagem (incluindo a extensão, por exemplo, jpg, png, gif, bmp etc). Caso a imagem esteja dentro de uma pasta, coloque primeiro o nome da pasta seguido do caracter barra “/” e depois o nome da imagem. Caso exista mais pastas entre o arquivo HTML e a imagem, coloque o caminho a partir do arquivo HTML.

Pode-se ainda usar o link de uma imagem na internet no src da imagem. Nesse caso o src deve conter o endereço completo incluindo o “http://”, por exemplo, veja o Exemplo 12.

9. Formulários

Ótimo, já sei colocar um bocado de informações no html e ver na minha tela, mas como faço para permitir o usuário do meu site digitar algo para mim? Formulários são a forma do usuário inserir informações no site para preencher campos, ter botões para clicar, etc.

9.1 Estrutura

Para definir um formulário usaremos a tag “form” e dentro do formulário podemos colocar campos para o usuário digitar, esses campos são criados usando a tag “input”.

Cada “input” corresponde a um campo onde o usuário pode digitar. Existem diversos tipos de input, inicialmente estudaremos os tipos “text” e “submit” que devem ser escolhidos usando o atributo “type”.

O tipo “text” informa ao input que deve-se ali permitir que o usuário digite informações, enquanto o tipo “submit” informa que trata-se de um botão para enviar esses dados para o seu site. Experimente o exemplo de formulário abaixo.

```
<form>
        <input type='text'/>
        <input type='submit'/>
</form>
```

Exemplo 13: Formulários

10. Tipos suportados pela *tag* input

Vimos anteriormente dois tipos de campos “input”, o campo de texto e o botão de submissão. Agora vamos aprender mais alguns tipos de campos input. Para testá-los use a sintaxe normal do input e no atributo “type”, coloque os valores em negrito abaixo. Teste usando diferentes navegadores.

10.1 text

Usado para que o usuário possa inserir qualquer informação textual com tamanho pequeno.

10.2 password

Esse tipo funciona exatamente como o campo de texto, porém o que for digitado é exibido como asteriscos, por segurança.

10.3 checkbox

Esse campo mostra uma caixinha que pode ser marcada ou desmarcada pelo usuário. A idéia

é que o usuário pode escolher quais caixinhas marcar, podendo inclusive escolher todas.

10.4 radio

O tipo radio ao invés de mostrar uma caixinha para o usuário marcar, mostrará um círculo a ser marcado. Nesse caso a idéia é exibir várias opções ao usuário que poderá escolher só uma, isto é, só conseguirá marcar um dos círculos.

```
<form>
        Sexo:
        <input type='radio' name='sexo'/> Masculino
        <input type='radio' name='sexo'/> Feminino
</form>
```

Exemplo 14: Radio Buttons

Para que isso seja possível, além de criarmos o campo input com atributo “type” com o valor “radio”, precisamos também usar o atributo “name” para nomear os inputs do tipo radio. É preciso nomear os inputs pois o usuário pode escolher apenas uma opção dentre um conjunto de opções. Definimos qual o grupo de opções nesse caso colocando elementos com mesmo nome. Veja o Exemplo 14.

10.5 date

Esse input é utilizado para que o usuário possa escolher datas. Seu uso mostra um calendário para ajudar na escolha.

Esse tipo de input ainda não é suportada por todos os navegadores. Isso ocorre porquê esta tag faz parte do padrão HTML 5 que é novo. Uma observação importante é que caso o tipo de campo não seja reconhecido pelo browser, ele irá interpretar como sendo um campo de texto. Teste em diferentes navegadores para ver como funciona nos principais.

O suporte atual deste tipo de campo de entrada é de 97.6% de todos os navegadores (fonte: caniuse.com). Não é suportado pelo opera mini e internet explorer, por exemplo.

10.6 file

Usado para que o usuário possa escolher um arquivo de seu computador para enviar.

10.7 reset

O input do tipo reset mostra na tela um botão que quando apertado, apaga os valores do formulário, fazendo os a voltar para como estavam antes do usuário começar a preencher.

10.8 submit

Todo formulário precisa ser enviado para que seus dados possam ser usados. Entraremos em detalhes mais a frente sobre esse processo, mas, para enviar, o usuário precisará clicar em um input do tipo submit (ficará com a aparência de um botão) ou um input do tipo image, mostrado abaixo. Existe ainda a opção de usar a tag button que será abordada em outra apostila.

10.9 image

Aqui não estamos falando de como colocar imagens em sites e sim em criar botões com a mesma funcionalidade do tipo “submit”, porém utilizando imagens para definir a estética, isto é, o usuário pode clicar na imagem e terá o mesmo efeito que faria com um botão.

Nesse caso além de criar o atributo “type” com valor “image”, devemos também criar o

atributo “src” informando o caminho onde está a imagem assim como fariamos usando a tag “img”.

10.10 email

Esse tipo de input funciona como um campo de texto porém ao clicar em um botão de submit, é verificado se a informação digitada é, de fato, um email.

10.11 hidden

Quando enviamos um formulário, muitas vezes precisamos enviar junto algum campo com informações que o usuário não deve ou não precisa ter acesso, por exemplo, enviar junto ao formulário a quantidade de tempo que o usuário demorou digitando, sendo isto calculado sem o usuário visualizar ou ter controle sobre esse dado.

11. Tags de outros campos de entrada

Além da tag input, existe outras tags para entrada de dados, veremos elas abaixo. Entre elas as principais são as tags select e textarea.

11.1 select

A tag select cria uma combobox, veja na Figura 1. Para criar uma combobox precisamos definir, também, as opções a ser exibidas quando clicar na seta ao lado do campo. Para criar uma combobox, usamos a tag select e para criar as opções a exibir, usamos a tag option.

```
<select>
        <option>Volvo</option>
        <option>Saab</option>
        <option>Mercedes</option>
        <option>Audi</option>
</select>
```

Exemplo 15: Select

Figura 1: Select

11.2 textarea

O textarea cria um campo de texto assim como o input com tipo texto, porém o campo textarea tem tamanho maior para que possa ser digitado textos maiores e o usuário pode redimensionar o textarea facilmente clicando no canto inferior direito. Um exemplo de textarea pode ser visto na Figura 2.

```
<textarea>
        Digite aqui...
</textarea>
```

Exemplo 16: Textarea

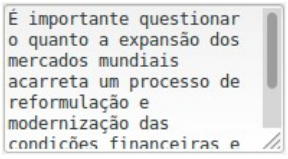


Figura 2: Textarea

12. Fieldset

Um site pode ter vários formulários em uma mesma página, é preciso organizá-los para que o usuário saiba quais campos e botões pertencem a cada formulário. Para isso, existe a tag “fieldset” (conjunto de campos) que cria uma borda ao redor dos campos que estiverem dentro dessa tag e usando a tag “legend” podemos definir o nome do formulário que ficará posicionado dentro da borda.

É mais fácil entender com um exemplo, execute o código abaixo no seu browser e o resultado deverá ser o mesmo apresentado na Figura 3.

```
<fieldset>
        <legend>Formulário</legend>
        <textarea>
                Digite aqui...
        </textarea>
</fieldset>
```

Exemplo 17: Fieldset

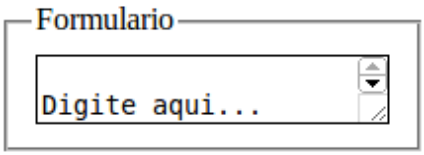


Figura 3: Exemplo de fieldset

Esta tag pode ser usada também para subdividir um formulário grande em partes menores ou agrupar qualquer conjunto de inputs com seus labels. Por exemplo, separar os campos de dados pessoais de campos de endereço, dados bancários etc.

13. Iframe

Em alguns casos pode ser útil incluir conteúdo de outro site dentro do seu, para isso usamos a tag iframe. Para usá-la basta usar o atributo src com valor igual ao endereço do site a ser aberto. O iframe funcionará como uma janela para outro site.

O exemplo abaixo não mais funciona devido o vídeo ter saído do ar na plataforma youtube, porém experimente acessar o youtube e colocar um vídeo dentro do seu site através da dica abaixo.

Dica: Ao abrir o vídeo no youtube, clique em “share” (compartilhar) e depois em “embed” (incorporar). Copie o código que o youtube lhe dará e cole em seu site. Ele usará a tag iframe com alguns atributos, tente modificá-los.

```
<iframe width='420' height='315' src='//www.youtube.com/embed/4xQb9Kl-O3E' frameborder='0' allowfullscreen></iframe>
```

Exemplo 18: Iframe

14. Label

Quando se usa botões do tipo radio ou checkboxes torna-se incoveniente para o usuário clicar em um espaço tão pequeno. Como precisamos colocar ao lado desse tipo de botão um texto indicando o que deve ser preenchido ou o que significa escolher cada opção, não seria ótimo se clicar nesse texto fosse equivalente a clicar no botão? Para isso podemos usar a tag label.

Para que isso funcione é preciso então que a tag label saiba a qual campo ela está relacionada e precisamos também que o campo tenha uma informação única que não pode se repetir no nosso arquivo HTML.

A informação única de cada tag HTML é definida no atributo “id” (um identificador) e na tag label é usado o atributo “for” (do inglês, significa “para”) para indicar, então, o id do campo a que se relaciona.

```
<b> Sexo: </b>
<input type='radio' id='mas' name='sexo'/> <label for='mas'>Masculino</label>
<input type='radio' id='fem' name='sexo'/> <label for='fem'>Feminino</label>
```

Exemplo 19: Label

Apesar da tag label ser especialmente útil no caso de inputs pequenos para facilitar o clique, essa tag pode ser usada com quase qualquer tipo de campo e executará processamento idêntico (equivale a clicar no campo a que ela está relacionada).

15. HTML4, XHTML e HTML5

Trata-se de versões diferentes do HTML, mas usam as mesmas tags com mesmos significados. A diferença está em algumas restrições que são obrigatórias, remoção de elementos que foram descontinuados e acréscimo de elementos novos.

Em outubro de 2014 foi recomendado o uso de HTML 5 para substituir o XHTML que era recomendado em maio de 2001 e antes disso a recomendação era que fosse usado o HTML 4.

No futuro estudaremos a fundo HTML 5, com isso será acrescentado novas tags para propósitos de acessibilidade e incorporar funções como áudio e vídeo, contudo o que estudamos nesta apostila já nos permite criar nossos primeiros códigos em HTML 5.

15.1 Doctype

Já que estamos usando uma versão específica do HTML, precisamos avisar isto ao browser para que este saiba o que está por vir. A forma de fazer isto é acrescentar na primeira linha, antes mesmo de iniciar a tag “html”, a informação de doctype (em português significa tipo de documento). No doctype abaixo estamos informando ao browser que o tipo de documento é HTML5.

Uma observação importante é que o doctype não é uma tag html, embora se assemelhe a uma. Por isso ela não fecha, uma vez que não é uma tag html. A informação de doctype é usada também em outras linguagens como o XML, SVG, entre outras, por isso essa informação é adicionada antes de qualquer tag da linguagem usada. Ela deve ser inserida exatamente como está abaixo e é obrigatório seu uso no HTML5.

```
<!DOCTYPE html>
```

Exemplo 20: Doctype

15.2 Atributo lang

O atributo lang deve ser adicionado na tag html e serve para informar o idioma que está escrito o conteúdo de nosso site. Abaixo, no Exemplo 21, está sendo definido o idioma como português.

```
<html lang=’pt’>
```

Exemplo 21: Lang

15.3 Elementos obrigatórios

As tags html, head, title e body são obrigatórias e devem sempre estar presentes. Note que a tag foot não existe no XHTML ou HTML5, por isso, não deve ser usada.

15.4 Regras de escrita

Todas as tags devem estar corretamente aninhadas, isto é, elementos filhos sempre devem ser fechados na sequencia que foram abertos. Além disso, todas as tags e seus atributos devem estar escritos com letras minúsculas devido a boas práticas.

~~ Estáerrado ~~ ~~<P>Estáerrado </B>~~ Estácorreto

As tags sempre devem estar fechadas. Note que temos dois tipos de tags explicados no início dessa apostila, o primeiro tipo abre e fecha em tags diferentes, tendo uma tag de abertura e uma de fechamento, contudo as tags vazias são tags únicas e são elas BR, HR e IMG, as únicas estudadas até o momento e estas não possuem tag de fechamento por serem únicas.

Os valores dos atributos devem sempre estar entre aspas e nenhum atributo pode ser usado se não estiver com seu valor definido. Aqui pode ser aspas simples ou duplas, mas note que nessa apostila está sendo usado apenas aspas simples nos atributos pois isso ajudará no futuro quando for misturado HTML com outra linguagem. Por enquanto, é aconselhado criar o costume de usar aspas simples ao invés de duplas, mas não é errado usar aspas duplas nos valores de atributos.

15.5 Charset

É obrigatório em HTML5 definirmos o charset, isto é, conjunto de caracteres que será usado em nosso idioma. O charset funciona como uma tabela convertendo caracteres em números binários. Nesse aspecto é importante usarmos uma tabela que seja aceita mundialmente para que nosso site funcione em qualquer lugar do mundo. Um padrão tem se mostrado a tabela utf-8, que contém caracteres de todos os idiomas do mundo. Para usá-lo, é preciso adicionar uma metatag demonstrada abaixo.

```
<meta charset='utf-8'>
```

Exemplo 22: Charset

15.6 CSS

Em HTML5 não podemos usar os atributos width, height, align, valign e border aprendidos nesta apostila para serem usados em tabelas. Ao invés disso, devemos usar CSS para definir essas características.

O motivo é que o código fica muito difícil de manter se ele ficar misturado estrutura e características visuais, o que fez com que o XHTML e HTML5 removessem boa parte das tags que eram usadas para esse propósito no HTML4 como, por exemplo, as tags font, center, entre várias outras.

Para não tentarmos estilizar nossa página usando tags ultrapassadas, partimos para uma forma de usar CSS *inline*, isto é, substituiremos os atributos estudados acima em tabelas por versões em CSS destes. Esta ainda não é a melhor forma, mas usaremos por enquanto.

Tag	HTML4	HTML5
Table	align='left'	style='margin-right: auto;'
Table	align='right'	style='margin-left: auto;'
Table	align='center'	style='margin: auto;'
TR e TD	align='left'	style='text-align: left;'
TR e TD	align='right'	style='text-align: right;'
TR e TD	align='center'	style='text-align: center;'
TR e TD	valign='top'	style='vertical-align: top;'
TR e TD	valign='bottom'	style='vertical-align: bottom;'
TR e TD	valign='middle'	style='vertical-align: middle;'
Table e TD	width='500px'	style='width: 500px;'
Table e TR	height='500px'	style='height: 500px;'
Table	border='1'	style='border: 1px solid black;'

Tabela 1: Conversão para CSS

Um detalhe importante é que uma *tag* não pode ter mais de um atributo com mesmo nome, assim, não é possível ter mais de um atributo *style* na mesma *tag*. Contudo, notemos que dentro do atributo *style*, temos uma regra acabada em ponto e vírgula, assim, podemos juntar várias dentro de um mesmo atributo *style, por exemplo.*

```
<table style='width:600px; height: 400px; border: 1px solid black;'>
	<tr>
		<td style='width:50%;'> Coluna 1, Linha 1 </td>
		<td style='text-align:right;'> Coluna 2, Linha 1 </td>
	</tr> <tr style='vertical-align:bottom;'>
		<td> Coluna 1, Linha 2 </td>
		<td> Coluna 2, Linha 2 </td>
	</tr>
</table>
```

Exemplo 23: Tabelas

O Exemplo 23, acima, é um trecho compatível com HTML5 do código demonstrado no Exemplo 10, quando demonstrado tabelas, contudo apesar da conversão aplicada usando a Tabela 1, o efeito visual não é idêntico devido as bordas que estão agora apenas em volta da tabela sem subdividir suas colunas e linhas.

Verifique que no Exemplo 23 foi adicionado bordas apenas na *tag Table*, então o navegador não irá inserir bordas em todas as colunas, mas podemos adicionar em todas as *tags* a regra *border*. Fica como exercício proposto tentar deixar todos os TDs com borda e personalizar as cores das bordas.

Um material de auxílio interessante é o Guia Rápido CSS criado pela W3C. Neste guia é explicado todas as regras CSS possíveis do CSS 2.1. Todas as regras CSS 2.1 são válidas no CSS 3, que é a última versão, então vale muito a pena aprender CSS 2 primeiro. O guia está disponível em <https://ceweb.br/media/docs/publicacoes/13/guia-css-w3cbr.pdf>.

Uma boa prática para verificarmos se nosso site está seguindo as diretrizes do HTML 5 é validar através de um validador online. O validador mais conhecido é o da W3, disponível em <https://validator.w3.org>.

16. Código inicial HTML5

Abaixo temos um código inicial para que sirva de referência rápida para iniciar códigos, contudo note que durante o desenvolvimento é necessário seguir as diretrizes apontadas acima.

```
<!doctype html>
<html lang='pt'>
	<head>
		<meta charset='utf-8'>
		<title> titulo do site </title>
	</head>
	<body>
	</body>
</html>
```

Exemplo 24: Código inicial

Experimente validar o código acima no validador online da W3C disponível em <https://validator.w3.org/>.