

Lista, links e formatação de texto

Neste capítulo, você irá aprender os primeiros passos para formatar uma página HTML.

Você vai aprender as formas corretas de criar listas, tanto aquelas que são numeradas quando as que contêm apenas marcadores. Cada uma possui uma forma correta de ser aplicada.

Além disso, também irá aprender como destacar texto na página, sem esquecer de que você deve sempre pensar no usuário que irá ler sua página e também nos robôs que irão interpretar cada *tag* colocada no código.

E, por fim, vai começar a entender como utilizar CSS para alterar a aparência da página. Vamos começar?

1 Lista ordenada e não ordenada

Muito bem, até aqui você aprendeu a criar parágrafos e títulos, e já sabe também como inserir imagens. Agora, vamos aprender a criar listas numeradas (conhecidas em HTML como lista ordenada) e marcadores (conhecidos em HTML como listas não ordenadas).

Esses elementos são muito úteis para listar itens no texto, como descrever uma lista de ingredientes e uma lista com as etapas de uma receita. No primeiro exemplo, a lista não precisa ser ordenada, mas no segundo exemplo sim.

As *tags* utilizadas para a construção de lista são as *tags* `` (Unordered List) e `` (Ordered List). E cada linha da lista é identificada pela *tag* ``. Vamos analisar os exemplos a seguir.

1.1 Lista não ordenada

Vejamos como criar uma lista não ordenada, tomando como exemplo ingredientes de uma receita de panqueca:

```
<ul>
  <li>1 xíc. farinha</li>
  <li>2 ovos</li>
  <li>2 xíc. de leite</li>
  <li>sal a gosto</li>
  <li>uma colher de manteiga</li>
</ul>
```

Percebeu como nós abrimos a *tag* ``, listamos todas as linhas, e só depois a fechamos? Com isso, você acaba de ver mais um exemplo de *tag* pai e *tag* filho, onde uma está um nível abaixo da outra.



IMPORTANTE

A tag `` representa uma linha dentro da lista. Portanto, é fundamental que você sempre feche essa tag para que o navegador entenda que a linha acabou ali e que o que virá na sequência deve ser representado logo abaixo como uma nova linha.

1.2 Lista ordenada

Vejamos agora, como criar uma lista ordenada, onde usaremos o exemplo de etapas da preparação de uma panqueca:

```
<ol>
  <li>Misture os ovos e o leite e a manteiga;</li>
  <li>Acrescente a farinha aos poucos e vá batendo até
formar uma massa;</li>
  <li>Acrescente sal a gosto;</li>
  <li>Em uma frigideira sobre fogo baixo, derrame uma
concha de massa de panqueca(obs: se a frigideira não for
boa, coloque uma colher pequena de manteiga);</li>
  <li>Vire quando estiver dourada;</li>
  <li>Coloque o recheio que quiser e está pronta!</li>
</ol>
```

1.3 Lista com subitens

Também é possível criar uma lista com subitens. Para ficar mais claro, basta imaginar os episódios de sua série preferida, por exemplo:

```
<ol>
  <li>Primeira temporada</li>
  <li>Segunda temporada</li>
  <ol>
```

```
temporada</li>                <li>Primeiro episódio da segunda
                                <li>Segundo episódio da segunda
temporada</li>                <li>Terceiro episódio da segunda
temporada</li>
                                </ol>
                                <li>Terceira temporada</li>
</ol>
```

2 Links (âncora)

Você se lembra que no capítulo anterior vimos que hipertextos são elementos conectados por palavras, imagens, etc., formando uma grande rede de informação?

Pois bem, o que permite que isso ocorra é o elemento âncora (anchor), representado em HTML pela tag <a>.

É esse elemento que faz a Web ser, verdadeiramente, uma rede. É ele que permite pegar o trecho de um texto, por exemplo, e vincular a qualquer página da Web, ou outro tipo de arquivo.



PARA SABER MAIS

Quando o arquivo vinculado à uma âncora não pode ser aberto em um navegador, imediatamente é feito o download dele.

Veja um exemplo de trecho de texto transformado em link para abrir um site:

```
<a href="http://w3.org">Clique nesse trecho para acessar a
W3C</a>
```

Percebeu que utilizamos o atributo "href"? A tag <a> é uma das que não tem efeito sem atributo.

O atributo href (hiperlink reference) é o que vai indicar para onde o navegador deve ir quando o usuário clicar no link.

3 Negrito e itálico

Assim como nas regras da ABNT para elaboração de TCC, o negrito e itálico possuem uma função específica, no HTML também é assim.

Use a tag para destacar algum texto, tal como quando vai fazer uma observação importante, um alerta, etc.

Exemplo:

```
<p>
<strong>IMPORTANTE!</strong>
Não esqueça de respeitar aquilo que em HTML5 é
imprescindível: a <strong>semântica</strong>.
</p>
```

Use a tag para dar ênfase para um texto, como um título de uma obra, por exemplo.

```
<p>Hoje, vamos estudar a geografia de <em>Game Of
Thrones</em> e lembrar como aquele final foi horrível</p>
```

Além dessas duas tags, também temos as tags (bold) e <i> (itálico). Diferente das tags explicadas acima, essas duas possuem uma função mais estética do que semântica. Portanto, utilize-as somente quando não fizer sentido utilizar as tags acima.

4 CSS para tratar texto

E agora, vamos começar a dar “forma” para tudo isso que estudamos. Para isso, iremos utilizar a linguagem de estilo: CSS.

O CSS (*Cascade Style Sheet*, tradução: Folha de Estilo em Cascata) é a linguagem que descreve como o HTML deve ser exibido no navegador.

Para começar a criar o estilo, iremos utilizar a *tag* `<style>`. Dentro dela, iremos inserir todos os estilos que usaremos na página.

Para criar, basta localizar a *tag* `<head>` e, dentro dela, escrever a *tag* `<style>`.

Veja no exemplo abaixo:

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

    </style>
  </head>
  <body>
    <h1>Título da página</h1>
    <p>Este é um parágrafo usado para demonstrar
como o estilo afeta texto em uma página HTML.</p>
    <p>Este é um segundo parágrafo usado para
demonstrar como <strong>todos</strong> os Elementos P
serão afetados com o estilo.</p>
  </body>
</html>
```

O atributo “type” permite especificar o tipo de folha de estilo utilizada. Atualmente, o único valor existente é “text/css”.

Dentro da *tag* `<style>` nós colocaremos as *regras*, que é como chamamos o código CSS. As regras vão determinar a aparência de cada elemento HTML e devem sempre ser escritas em letras minúsculas.

Vamos começar aprendendo como tratar o texto, utilizando a estrutura de código acima.

Você pode copiar em um bloco de notas e salvar como “html” para testar.

4.1 Cor

Esse estilo é utilizado para alterar a cor do texto. Porém, saber só isso não é suficiente. Você precisa informar qual elemento HTML será afetado por essa cor.

Vejamos alguns exemplos:

```
body{color:red;} //aplicando vermelho em TODO texto da página
```

Agora, vamos entender o que cada parte representa.

- `body` – esse é o elemento que será afetado;
 - `color` – essa é a propriedade que altera a cor do texto. Como ela está dentro das chaves do elemento `body`, significa que o texto da página inteira vai ser afetado.
 - `red` – esse é o valor da propriedade. No caso, a cor que será aplicada no elemento.
 - `//` – quando temos duas barras em um código CSS significa que estamos dando uma instrução para que o navegador ignore e não tente interpretar o texto à direita da barra como parte do código. Com isso, esse texto fica reservado para comentários.
1. Essa regra deve ser colocada dentro da tag `<style>`.
 2. Todas as propriedades e seus respectivos valores devem ficar dentro das chaves.

3. Para separar uma propriedade de outra, usa-se o sinal de “,” (pon-
to e vírgula).

Vejamos agora, outro exemplo de regra CSS:

```
p{color:red;} //aplicando vermelho APENAS nos elementos P  
da página
```

Note que com essa regra, a cor é aplicada apenas nos parágrafos. O título não é afetado.

```
strong{color:red;} //somente o texto dentro do elemento  
strong será afetado
```

O exemplo anterior mostra que qualquer elemento relacionado ao texto pode utilizar a propriedade color. Já um elemento img, por exemplo, não consegue utilizar essa propriedade.

4.2 Fonte

Agora, vamos estudar algumas regras para alterar o estilo da fonte do texto.

```
h1{font-family:Arial, Helvetica, Sans-serif;}
```

No exemplo acima, temos a propriedade “font-family”, e como o nome já diz, é uma família de fontes.

A regra acima diz o seguinte para o navegador: exibir o texto do parágrafo (elemento p) usando a fonte Arial. Na falta dessa, usar a fonte Helvetica. E na falta de qualquer uma, aplicar uma fonte *sem serifa* (sans-serif). Lembrando que o navegador exibe de acordo com as fontes disponíveis no computador do usuário.


```
p{font-family: "Times New Roman", Georgia, serif;}
```

Note como tivemos que utilizar aspas para "Times New Roman". Isso é preciso para que o navegador entenda que se trata de uma fonte, e não três. E todas as fontes desta regra são *serifadas*.

E agora que você já viu duas propriedades (color e font-family), vejamos como alterar duas propriedades na mesma regra:

```
p{color:blue; font-family:Impact, Sans-serif;}
```

4.3 Tamanho

Para alterar o tamanho do texto utilizamos a propriedade "font-size". E o valor desta propriedade pode ser absoluto ou relativo.

Absoluto é quando representamos em pixel.

```
h1{font-size:14px;}
```

De acordo com o exemplo acima, todos os títulos do nível 1 (h1) terão seu tamanho alterado para 14px.

Você utiliza pixel quando sabe, por exemplo, o tamanho do local onde o texto será exibido, como a tela de um smartphone ou um televisor ou uma folha A4.



IMPORTANTE

Ainda que, neste caso, o título tenha assumido um tamanho muito pequeno, isso não afeta a semântica do código. Ou seja, a tag `<h1>` continua sendo o título principal e uma das informações mais importantes do

texto da página. Por isso que não se deve usar o HTML pela aparência, pois essa pode ser alterada pelo CSS.

Tamanho relativo é quando o tamanho é definido pelos elementos ao redor do texto, e quando você quer permitir que o usuário altere o tamanho da fonte da sua página, como pessoas com baixa visão que precisam aumentar o texto para leitura. Para isso, você deve utilizar a medida “em”. Veja o exemplo:

```
p{font-size:2.5em;}
```

Quando não informamos o tamanho que o texto deve ter, o tamanho padrão que o navegador utilizará é 16px. Logo, 1em corresponde a 16px.

Para sabermos como converter uma medida em *pixel* para uma medida em *em*, basta pegarmos a medida e dividirmos por 16, que é o valor padrão.

Exemplo de medida fixa:

```
p{font-size:24px;}
```

Exemplo de medida flexível:

```
p{font-size:1.5em;} // 24/16 = 1,5
```

Nos exemplos acima, a fonte será exibida com o mesmo tamanho, mas o valor mostrado no segundo exemplo permite que o usuário altere o tamanho pelo navegador.

4.4 Borda

Imagine que você precise dar um destaque a um trecho do texto, como uma “nota”, um “lembrete” ou um título. Podemos criar uma borda ao redor do elemento.

Exemplo de aplicação:

```
p {border-style: solid;}
```

Estilo da borda (border-style)

Além do valor “solid”, também podemos utilizar outros valores: double, groove, dotted, dashed, ridge, inset, outset, none (nenhum) e hidden (oculta).



IMPORTANTE

Se as outras propriedades de borda não funcionarem, é porque você ainda não definiu o estilo da borda.

Largura da borda (border-width)

Pode ser representada por diversas medidas (como px, pt, em, cm, etc.).

Exemplo:

```
p {border-width:12px;}
```

Pode ser representada com larguras diferentes em cada uma das quatro posições (topo, direita, inferior e esquerda).

Exemplo:

```
p {border-top:6px;}  
p {border-right:20px;}  
p {border-bottom:6px;}  
p {border-left:20px;}
```

Exemplo de como abreviar:

```
p {border:6px 20px 6px 20px;}
```

E você também pode abreviar com as várias propriedades de borda. Por exemplo, criando uma borda sólida, com 6px de espessura e cor vermelha:

```
p {border:solid 6px #ff0000;}
```

Cor da borda (border-color)

Pode ser definida por nome, valor hexadecimal, valor em RGB ou transparente.

```
p {border-color:red;}
```

4.5 Cor de fundo

A cor de fundo pode ser aplicada em diversos elementos HTML, tanto para alterar o fundo da página quanto alterar o fundo da linha de um parágrafo.

A aplicação é bem simples:

```
body {background-color:pink;} //o fundo da página fica rosa
```

Além dos nomes das cores, também podemos utilizar valores hexadecimais, que são formados por 3 pares de dois dígitos, cada um representando, respectivamente, as cores R(red) G(green) B(blue).

Os valores vão de 0 a 9, e depois de A à F, onde 0 (zero) representa o branco, F representa o branco, e os valores intermediários são combinados para produzir demais cores.

Exemplo da cor vermelha representada em hexadecimal:

#	ff	00	00
	R	G	B

Aplicada na propriedade color:

```
p{background-color:#ff0000;}
```

Exemplo da cor branca representada em hexadecimal:

#	ff	ff	Ff
	R	G	B

Aplicada na propriedade background-color:

```
p{background-color:#ffffff;}
```

Outra maneira de definir cor é utilizando valor em RGB, onde o mínimo é 0 e o máximo é 255.

```
p{background-color:rgb(255,0,0);} //resulta em vermelho  
p{background-color:rgb(255,255,0);} //resulta em amarelo.  
p{background-color:rgb(255,255,255);} //resulta em branco.
```

Atividade

Abaixo temos o mesmo layout do capítulo 1, porém, com algumas modificações.

Para exercitar o que foi aprendido nesta aula, proponho que você pegue a página criada no capítulo anterior e modifique para ficar conforme o layout abaixo.

1. Body – cor de fundo igual a “ghostwhite”;
2. Título h1 – tamanho 72px, fonte “Cambria”;
3. Título h2 – tamanho 14px, cor #777777;
4. Título h3 – tamanho 16px, cor #777777;
5. Título h4 – tamanho 24px, fonte “Cambria”, cor darkcyan;
6. Parágrafo – Fonte “Arial”, cor #777777;
7. - cor #ffffff, cor de fundo darkcyan, estilo da borda sólido, largura da borda 1px, cor da borda #cccccc.



Considerações finais

Nesse capítulo, vimos, na prática, a separação de estrutura (HTML) e estilo (CSS). Recapitulamos a importância de respeitar a semântica dos elementos HTML e o uso da *tag* que faz a Web ter sentido: a tag <a>. Sem ela, não conseguiríamos criar vínculo entre uma página e outra, formando uma rede.

A partir de agora, você já consegue criar uma estrutura HTML e vincular uma página à outra, assim como melhorar a aparência do seu texto.

Referências

GOMES, Ana Laura. **XHTML/ CSS**: criação de páginas web. São Paulo: Ed. Senac São Paulo, 2010.