8 HTML/CSS/JavaScript

Neste capítulo você vai entender como os programas (scripts) *em JavaScript encaixam-se e operam sobre uma estrutura muito maior e poderosa: o* documento *HTML. Vai estudar como utilizar comandos de marcação* (mark-up) *para identificar, caracterizar, acessar e modificar cada elemento de seu documento, como utilizar os elementos de estilo* (styles) *para produzir dos mais simples aos mais sofisticados efeitos de formatação e interação e como utilizar o JavaScript para manipular tudo isso dinamicamente, dando interatividade ao seu documento.*

Talvez o arquivo HTML mais simples que alguém pode criar seja algo como:

Listagem:

```
<html>
<body>
<p>Equações de Maxwell</p>
</body>
</html>
```

Se você salvar este arquivo como `teste.html` e em seguida clicar sobre ele, verá que o navegador vai mostrar o seguinte:

Resultado:

Equações de Maxwell

De tudo o que foi escrito no arquivo, somente o texto "Equações de Maxwell" foi apresentado. Todos os outros elementos são *marcadores*, (*tags*), que orientam o navegador

sobre o que fazer com os pedaços de texto que você coloca aqui e ali.

Sua vez... (8-1)

O HTML tem predefinidos outros elementos com comportamento semelhante, os cabeçalhos `<h1>...</h1>`, `<h2>...</h2>` etc. Troque o elemento `<p>...</p>` por estes outros elementos e veja o efeito. Quantos tipos diferentes de cabeçalhos seu navegador consegue identificar?

Você deve ter notado os marcadores `<html>...</html>` no início e no fim do arquivo. Este "envelope" diz ao navegador para interpretar o arquivo como um documento HTML. Se você procurar por aí, vai descobrir que este marcador pode ter uma estrutura bastante complexa, passando informações adicionais ao navegador. Não é objetivo deste texto aprofundar este assunto, mas se você quiser utilizar todo o poder dessa tecnologia, não vai conseguir escapar disso.

Há um envelope de segundo nível definido pelos marcadores `<body>...</body>`, que delimitam o *corpo* do documento. Este segundo envelope pode parecer redundante, mas não é. Como veremos, podemos associar a ele instruções de formatação global do documento (tamanho da fonte, cor ou imagem de fundo etc.) e ações específicas que serão executadas quando o navegador carregar o documento. Além disto, existem outros elementos que podem aparecer entre os marcadores `html` mas fora dos marcadores `body`.

O exemplo a seguir dá à página um título, que aparecerá na barra principal da janela quando o documento é carregado. Além disso, usa um elemento de estilo para colocar um fundo cinza claro (*lightgray*) em todo o documento e abre uma caixa de alerta dando as boas vindas ao usuário:

Listagem:

```
<html>
<title>Eletromagnetismo</title>
<body style="background-color:lightgray"
      onLoad="window.alert('Seja bem vindo!')">
<p>Equações de Maxwell.</p>
</body>
</html>
```

O próximo fato a ser notado é que os marcadores, salvo raríssimas exceções, aparecem sempre aos pares: `<html>...</html>`, `<body>...</body>`, `<title>...</title>`, `<p>...</p>`, sendo que o segundo elemento consiste na repetição do nome do marcador precedida de uma barra (`"/"`). O elemento `<p>...</p>` representa um parágrafo no documento. Experimente ver as diferenças entre as representações pelo navegador das sequências a seguir:

Listagem:

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

```
<p>Equações de Maxwell.</p>
<p>A base do eletromagnetismo clássico!</p>
</body>
</html>
```

Listagem:

```
<html>
<body>
<p>
Equações de Maxwell.
A base do eletromagnetismo clássico!
</p>
</body>
</html>
```

Listagem:

```
<html>
<body>
<p>
Equações    de
Maxwell.

A   base  do
Eletromagnetismo    cássico
!
</p>
</body>
</html>
```

Se você experimentou deve ter notado que as diferenças entre as duas primeiras sequências é grande: na primeira, cada frase saiu em uma linha diferente e há um espaçamento razoável entre elas, pois constituem dois parágrafos distintos, como indicam os marcadores `<p>...</p>`. Já entre a segunda e a terceira sequências não existem diferenças nas representações. O navegador encara tudo como pertencente a um único parágrafo e ignora espaços em branco ou linhas adicionais.

Inúmeras outras instruções podem ser passadas ao navegador através dos marcadores. Já vimos, por exemplo, que temos que envolver com `<i>...</i>` um trecho que queremos que apareça em *itálico* e com `<b>...</b>` um trecho que queremos que apareça em **negrito** (*boldface*). Podemos fazer $_{\text{subscritos}}$ com `<sub>...</sub>` e $^{\text{sobrescritos}}$ com `<sup>...</sup>`.

Sua vez... (8-2)

Escreva uma carta à sua namorada ou namorado explicando toda a beleza das equações de Maxwell. Utilize os elementos `<i>...</i>` e `<b>...</b>` para atribuir diferentes formatações a palavras no texto e os elementos de estilo `<p style="text-align:left">`, `<p style="text-align:right">` e `<p style="text-align:justify">` para atribuir diferentes modos de alinhamento

dos parágrafos:

Florianópolis, 20 de outubro de 2015

Querida:

Você não imagina o que aprendi essa semana: as **equações de Maxwell**! Estou pasmo! São apenas quatro, mas com elas podemos ...

Com saudades,

Fulano de Tal

Imagens

A inclusão de imagens no HTML é feita com o uso do marcador `<img ... />`, que é um dos poucos que não requer o par de fechamento. Se o arquivo com a imagem chama-se `densimetro.gif`, por exemplo, e está na mesma pasta que o documento HTML, a inclusão é feita como no exemplo a seguir.

Listagem:

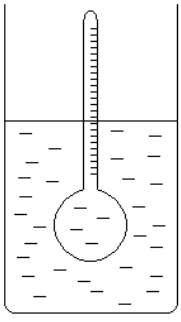
```
<p>
Um densímetro é um dispositivo desenhado para medir a
densidade de líquidos...
</p>

<p style="text-align:center">
<img src="densimetro.gif" style="width:100px" />
</p>

<p>
Como se pode notar, contém um bulbo meticulosamente
calibrado e uma escala...
</p>
```

Resultado:

Um densímetro é um dispositivo desenhado para medir a densidade de líquidos...



Como se pode notar, contém um bulbo meticulosamente calibrado e uma escala...

O elemento `img` requer um valor para o parâmetro `src` (*source*), a especificação da URL (*Universal Resource Locator*) da imagem, ou seu "endereço", que pode estar na mesma pasta do arquivo que a carregou, em outra pasta no mesmo computador.

Note também a declaração de estilo, que impõe que sua largura (*width*), no documento, seja exatamente de 100 pixels, independentemente do seu tamanho original.

Listas ordenadas e não-ordenadas

Particularmene úteis para ajudar na organização de um texto são as listas ordenadas e não-ordenadas. Uma lista ordenada é indicada pelos marcadores `<ol> ... </ol>` (*ordered list*) e seus elementos pelos marcadores `<li> ... </li>` (*list item*):

Listagem:

```
<p>Unidades de energia:</p>
<ol>
<li>Erg</li>
<li>Joule</li>
<li>Elétron-volt</li>
</ol>
```

Resultado:

Unidades de energia:

1. Erg
2. Joule
3. Elétron-volt

Uma lista não-ordenada é indicada pelos marcadores `<ul> ... </ul>` (*unordered list*) e seus elementos também são indicados pelos marcadores `<li> ... </li>`:

Listagem:

```
<p>Unidades de energia:</p>
<ul>
<li>Erg</li>
<li>Joule</li>
<li>Elétron-volt</li>
</ul>
```

Resultado:

Unidades de energia:

- Erg
- Joule
- Elétron-volt

Estas listas podem ser aninhadas (colocadas umas dentro de outras) e seus marcadores escolhidos dentre algumas possibilidades utilizando a declaração de estilo `style="list-style-type:`*`seletor`*`"`. Para listas ordenadas, alguns seletores disponíveis são: `decimal` (padrão), `lower-alpha`, `upper-roman`, entre outros. Para listas não ordenadas, alguns seletores disponíveis são: `disc` (padrão), `circle` e `square`.

Listagem:

```
<ol style="list-style-type:upper-alpha">
<li>Unidades de energia:</li>
   <ul style="list-style-type:square">
   <li>Erg</li>
   <li>Joule</li>
   <li>Elétron-volt</li>
   </ul>
<li>Unidades de potência:</li>
   <ul style="list-style-type:circle">
   <li>Watt</li>
   <li>Cavalo-vapor</li>
   <li>BTU</li>
   </ul>
</ol>
```

Resultado:

A. Unidades de energia:
   - Erg
   - Joule
   - Elétron-volt

B. Unidades de potência:
   - Watt
   - Cavalo-vapor
   - BTU

É também possível colocar qualquer imagem no lugar do conteúdo padrão da lista não-ordenada. O exemplo abaixo assume a existência de um arquivo `checkbox.gif` na mesma pasta em que se encontra o arquivo HTML e utiliza a imagem nele contida em cada item da lista:

Listagem:

```
<p>Para compreender bem as rotações você deve estudar:</p>
```

```
<ul style="list-style-image:url(checkbox.gif)">
<li>Momento de inércia</li>
<li>Torque</li>
<li>Momento angular</li>
</ul>
```

Resultado:

Para compreender bem as rotações você deve estudar:

- ❑ Momento de inércia
- ❑ Torque
- ❑ Momento angular

Sua vez.. (8-3)

Escreva um trecho de código HTML que explore os vários estilos de listas ordenadas e não ordenadas. Consulte a internet para conhecer todos os possíveis valores para o `list-style-type`:

1. Primeiro nível
    A. Segundo nível
        a. Terceiro nível
            I. Quarto nível
                i. Quinto nível

- Primeiro nível
    - Segundo nível
        - Terceiro nível
            - ❑ Quarto nível (produza sua própria imagem!)

JavaScript

Trechos de código JavaScript podem ser introduzidos em qualquer ponto de um arquivo HTML utilizando os marcadores `<script>...</script>`, inclusive no meio de uma simples frase. O exemplo a seguir mostra a função `fatorial(x)` definida no cabeçalho e evocada no corpo do documento.

Listagem:

```
<html>
<head>
<script>
function fatorial(num) {
   if (num==0) return 1;
   var res = 1;
   do {
      res = res * num;
```

```
      num--;
   } while (num>0);
   return res;
}
</script>
</head>
<body>
<script>
var num = window.prompt("Entre com um número inteiro positivo: ");
</script>
<p>O fatorial de <script>document.write(num)</script>
é <script>document.write(fatorial(num))</script>.</p>
</body>
</html>
```

É muito comum encontrar grandes quantidades de JavaScript na forma de funções nos cabeçalhos (*head*) dos arquivos HTML. Todas as funções e variáveis globais definidas neste contexto permanecem disponíveis a menos que sejam explicitamente apagadas.

O exemplo a seguir apresenta uma questão de prova em que duas velocidades são sorteados na hora em que a página é carregada. Os valores sorteados são armazenados e posteriormente utilizados para calcular automaticamente a resposta para o problema.

Listagem:

```
<script>
var vn = (1 + Math.random()).toFixed(1);
var vb = (15 + Math.random()).toFixed(1);
str = "<p>A distância entre a Ilha do Campeche e a \
       Praia do Campeche é de cerca de 1.8 km. Um nadador \
       sai da praia em direção à ilha nadando a uma \
       velocidade de " + vn + " km/h. Quanto tempo depois \
       um barco deve deixar a praia, navegando a " + vb + "km/h, \
       para chegar à ilha junto com o nadador?</p>";
// Resolução
// xn = vn * tn;
// xb = vb * tb;
var tn = 1.8/vn;
var tb = 1.8/vb;
var dt = (tn - tb);
str += "<p><i>Resposta:</i> " + dt.toFixed(1) + " horas.</p>"
document.write(str);
</script>
```

Resultado:

A distância entre a Ilha do Campeche e a Praia do Campeche é de cerca de 1.8 km. Um nadador sai da praia em direção à ilha nadando a uma velocidade de 1.3 km/h. Quanto tempo depois um barco deve deixar a praia, navegando a 15.2 km/h, para chegar à ilha junto com o nadador?

Resposta: 1.3 horas.

Note o uso de uma variável `str` inicializada com o enunciado da questão e que, depois dos cálculos, acumula (`+=`) a resposta e é impressa com a chamada final a `document.write(...)`. Note também que o enunciado da questão está dividido em várias linhas por conveniência editorial, terminadas por uma barra invertida (`\`) para que o navegador não ache que você simplesmente esqueceu de fechar as aspas.

Sua vez... (8-4)

Inspire-se no exemplo anterior e construa uma questão que sorteie quatro números a_1, a_2, b_1 e b_2 entre −1 e +1 que representem as componentes de dois vetores bidimensionais e calcule e imprima o resultado da sua soma (vetorial).

Tabelas

Tabelas são um importante recurso de organização da informação, particularmente quando esta vem em grandes quantidades. O exemplo a seguir ilustra como fazer uma tabela muito simples em HTML, com duas linhas e duas colunas:

Listagem:

```
<table border='1' cellspacing='5' cellpadding='5' width='300'>
   <tr>
      <td>linha 1, coluna 1</td>
      <td>linha 1, coluna 2</td>
   </tr>
   <tr>
      <td>linha 2, coluna 1</td>
      <td>linha 2, coluna 2</td>
   </tr>
</table>
```

Resultado:

linha 1, coluna 1	linha 1, coluna 2
linha 2, coluna 1	linha 2, coluna 2

Tabelas são construídas utilizando-se os marcadores `<table> ... </table>` (*tabela*), que sinalizam o início e o fim da tabela, `<tr> ... </tr>` (*table row*), que sinalizam o início e o fim de uma linha, e `<td> ... </td>` (*table data*), que sinalizam o início e o fim de uma célula de dados da tabela. Como em qualquer outra estrutura HTML, quebras de linha, espaços e tabulações no código fonte são ignorados e podem ser utilizados ao gosto do autor. A sequência abaixo levaria exatamente ao mesmo resultado:

```
<table border='1' cellspacing='5' cellpadding='5' width='300'>
```

```
<tr><td>linha 1, coluna 1</td><td>linha 1, coluna 2</td></tr>
<tr><td>linha 2, coluna 1</td><td>linha 2, coluna</td></tr>
</table>
```

O elemento `table` pode ter diversos atributos e receber definições de estilo (`style=...`), o que não é o caso desta tabela em particular. São utilizados os atributos `border`, que define a espessura da borda da tabela e das células, `cellspacing`, que define o espaçamento entre duas células, `cellpadding`, que especifica a distância entre a parede da célula e o texto dentro dela e `width`, que especifica a largura da tabela. A menos que explicitamente especificados de outra maneira, todos os valores são dados em pixels. Experimente mudar os valores e veja o que acontece.

Os marcadores para linhas (`tr`) e células (`td`), como praticamente todos os outros elementos HTML, também podem ter atributos e declarações de estilo.

A associação de tabelas, declarações de estilo e JavaScript pode ser bastante produtiva. O exemplo a seguir imprime em uma tabela a tabuada de 1 a 10, gerada por um script:

Listagem:

```
<p>
<table style='width:500; margin:auto'
       cellpadding='3' cellspacing='0' border='1'>
<script>
// sinal de vezes no canto superior esquerdo da tabela
var str = "<tr><th style='background-color:gray'>&times;</th>";
// cabeçalhos das colunas de 1 a 10
for (var col=1; col<=10; col++)
   str += "<th style='background-color:gray'>" + col + "</th>";
// fecha a linha de cabeçalhos das colunas
str += "</tr>";
// escreve as 10 linhas seguintes
for (var lin=1; lin<=10; lin++) {
   // começa a linha
   str += "<tr>";
   // célula escurecida no início da linha
   str += "<th style='background-color:gray'>" + lin + "</th>";
   // produtos propriamente ditos
   for (var col=1; col<=10; col++)
      str += "<td style='text-align:center'>" + lin*col + "</td>";
   // termina a linha
   str += "</tr>";
   }
document.write(str);
</script>
</table>
</p>
```

Resultado:

×	1	2	3	4	5	6	7	8	9	10

1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

Algumas novidades do exemplo acima:

1. O declaração de estilo `style='...'` foi utilizada no marcador `table` para definir sua largura e estabelecer margens automáticas, o que centraliza a tabela.

2. O marcador `<th> ... </th>` (*table header*) deixa o conteúdo da célula em negrito e centralizado automaticamente.

3. A declaração de estilo `style='background-color:gray'` atribui a cor cinza ao fundo da célula de dados.

4. A tabela a ser impressa foi "acumulada" em uma variável literal `str`, escrita no documento somente no final do processo.

Sua vez.. (8-5)

Para navegar melhor entre tantos `tr`'s e `td`'s, modifique o exemplo acima de modo que (a) passe a imprimir a tabela de multiplicação apenas de 1 a 5, (b) a linha e a coluna com fundo cinza passe a ter um fundo amarelo (`yellow`) e as demais células passem a ter um fundo verde-claro (`lightgreen`).

×	1	2	3	4	5
1	1	2	3	4	5
2	2	4	6	8	10
3	3	6	9	12	15
4	4	8	12	16	20
5	5	10	15	20	25

É possível juntar (mesclar) células em linhas e colunas. Isso é feito utilizando-se os atributos `rowspan` (abrangência de linhas) e `colspan` (abrangência de colunas). No exemplo abaixo, que mostra os isótopos do magnésio, suas meias-vidas e modos de decaimento, a

primeira linha define 4 células; a segunda define três, mas a célula central ocupa duas colunas e duas linhas; como 2 das 4 células da terceira linha já foram ocupadas pela extensão da célula da segunda linha, nela são definidas somente 2 células; finalmente, a quarta linha volta ao "normal", apresentando quatro células.

Listagem:

```
<table style='text-align:center; font-size: 10pt;'
       align='center' border='1' cellspacing='0' cellpadding='3'
       width='300'>
<tr>
   <td width='25%'>20<br>0.1 s<br>β<sup>+</sup>, p</td>
   <td width='25%'>21<br>122 ms<br>β<sup>+</sup>, p</td>
   <td width='25%'>22<br>3.86 s<br>β<sup>+</sup></td>
   <td width='25%'>23<br>11.32 s<br>β<sup>+</sup></td>
</tr>
<tr>
   <td>31<br>0.25 s<br>β<sup>-</sup>, n </td>
   <td colspan='2' rowspan='2'><font size="+4">Mg</font></td>
   <td>24<br>estável<br>(78.99%)</td>
</tr>
<tr>
   <td>30<br>0.33 s<br>β<sup>-</sup></td>
   <td>25<br>estável<br>(10.00%)</td>
</tr>
<tr>
   <td>29<br>1.3 s<br>β<sup>-</sup></td>
   <td>28<br>21 h<br>β<sup>-</sup></td>
   <td>27<br>9.45 min<br>β<sup>-</sup></td>
   <td>26<br>estável<br>(11.01%)</td>
</tr>
</table>
```

Resultado:

<table>
<tr><td>20
0.1 s
β^+, p</td><td>21
122 ms
β^+, p</td><td>22
3.86 s
β^+</td><td>23
11.32 s
β^+</td></tr>
<tr><td>31
0.25 s
β^-, n</td><td colspan="2" rowspan="2">Mg</td><td>24
estável
(78.99%)</td></tr>
<tr><td>30
0.33 s
β^-</td><td>25
estável
(10.00%)</td></tr>
<tr><td>29
1.3 s
β^-</td><td>28
21 h
β^-</td><td>27
9.45 min
β^-</td><td>26
estável
(11.01%)</td></tr>
</table>

O uso do `rowspan` e do `colspan` não é fácil, uma vez que faz com que a contagem de linhas e colunas em uma tabela não seja mais tão direta. Mesmo desenvolvedores experientes têm dificuldade em produzir tabelas com mesclagens complexas. Só a experiência (e persistência!) vai deixá-lo confortável com isso.

Sua vez... (8-6)

Reproduza as tabelas a seguir, que mostram os possíveis resultados de combinações lógicas utilizando os operadores E (*and*) e OU (*or*) entre valores que podem ser verdadeiros (V) ou falsos (F). *Dica:* comece pensando nelas como tabelas originalmente com quatro linhas e quatro colunas; que a célula no canto superior esquerdo (que contém **E** ou **OU** abrange 2 × 2 células; que a célula que contém a condição **A** abrange 2 colunas; que a célula que contém a condição **B** abrange duas linhas.

E		**A**	
		V	**F**
B	**V**	V	F
	F	F	F

OU		**A**	
		V	**F**
B	**V**	V	V
	F	V	F

Às vezes é interessante aninhar tabelas dentro de tabelas. No exemplo a seguir é construída uma tabela com 1 linha e 3 colunas, sem bordas, para conter, em cada uma de suas células, novas tabelas, desta vez com duas colunas, um número variável de linhas e bordas aparentes.

Listagem:

```
<p>
<table style='text-align:center' align='center'
       width='500' cellpadding='10'>
<tr>
   <td valign='center'>
      <table style='text-align:center' align='center'
            width='150' cellpadding='5' cellspacing='0'
            border='1'>
      <tr><td><sup>1</sup>H</td><td>estável</td></tr>
      <tr><td><sup>2</sup>H</td><td>estável</td></tr>
      <tr><td><sup>3</sup>H</td><td>12.26 a</td></tr>
      </table>
   </td>
   <td valign='center'>
      <table style='text-align:center' align='center'
            width='150' cellpadding='5' cellspacing='0'
            border='1'>
      <tr><td><sup>3</sup>He</td><td>estável</td></tr>
      <tr><td><sup>4</sup>He</td><td>estável</td></tr>
      <tr><td><sup>6</sup>He</td><td>0.808 s</td></tr>
      <tr><td><sup>8</sup>He</td><td>0.122 s</td></tr>
      </table>
   </td>
   <td valign='center'>
      <table style='text-align:center' align='center'
            width='150' cellpadding='5' cellspacing='0'
            border='1'>
      <tr><td><sup>5</sup>Li</td><td><10<sup>-20</sup> s</td></tr>
      <tr><td><sup>6</sup>Li</td><td>estável</td></tr>
      <tr><td><sup>7</sup>Li</td><td>estável</td></tr>
      <tr><td><sup>8</sup>Li</td><td>0.844 s</td></tr>
      <tr><td><sup>9</sup>Li</td><td>0.178 s</td></tr>
```

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

Resultado:

^{1}H	estável
^{2}H	estável
^{3}H	12.26 a

^{3}He	estável
^{4}He	estável
^{6}He	0.808 s
^{8}He	0.122 s

^{5}Li	$<10^{-20}$ s
^{6}Li	estável
^{7}Li	estável
^{8}Li	0.844 s
^{9}Li	0.178 s

Cores

Os elementos de um documento HTML podem ter cor, tanto no seu conteúdo quanto no seu fundo (*background*). As cores podem ser atribuídas utilizando declarações de estilo (`style`) com os atributos `color` e `background-color`. As cores propriamente ditas podem ser especificadas de três maneiras: através de seus nomes (em inglês), de uma função específica do atributo de estilo ou de sequências de caracteres hexadecimais.

As cores podem ser utilizadas em todos os elementos HTML, tais como parágrafos, linhas e células de tabelas, listas ordenadas e não-ordenadas e em seus itens individualmente etc. O exemplo abaixo ilustra alguns usos. Note em particular o uso do marcador `<span>...</span>` (abrangência) quando se deseja marcar um pedaço de texto dentro de algum outro elemento, como em um parágrafo, por exemplo.

Listagem:

```
<p style='color: white; background-color: black'>
Os atributos de estilo <code>color</code> e
<code>background-color</code> definem a cor do texto
e a cor do fundo, respectivamente. Já quando queremos
que um <span style='color: black; background-color: white'>
trecho qualquer </span> fique diferente utilizamos o
marcador <code>span</code>.
</p>

<ol style='background-color: silver'>
<li> Este item está na cor de fundo padrão da lista toda.
<li style='background-color: gray'> Este item teve a cor
alterada localmente.
<li> Este item está na cor de fundo padrão da lista.
<li> <span style='background-color: gray'> Este item teve
a cor alterada com o uso de <code>span</code>.</span>
<li> Este item está na cor de fundo padrão da lista.
</ol>
```

Resultado:

Os atributos de estilo `color` e `background-color` definem a cor do texto e a cor do

fundo, respectivamente. Já quando queremos que um trecho qualquer fique diferente utilizamos o marcador `span`.

1. Este item está na cor de fundo padrão da lista toda.
2. Este item teve a cor alterada localmente.
3. Este item está na cor de fundo padrão da lista.
4. Este item teve a cor alterada com o uso de `span`.
5. Este item está na cor de fundo padrão da lista.

Cores podem ser especificadas em declarações de estilo por nomes predefinidos — "black", "blue", "red", "gray", "silver", "navy", "teal", "purple" e dezenas de outros nomes exóticos. Cores também pode ser definidas através da declaração `rgb(R,G,B)`, onde *R*, *G* e *B* são números entre 0 e 255 (em base 10), ou por sequências hexadecimais do tipo `"#rrggbb"` em que 255 valores (00 a FF) podem ser atribuídos às componentes rr (*red*), gg (*green*) e bb (*blue*) que compõem a cor (dois caracteres hexadecimais para cada uma). Dessa maneira, estão disponíveis cerca de 16 milhões de cores.

O exemplo a seguir especifica as cores com declarações de estilo que utilizam a função `rgb(...)` mas identifica as cores, na tabela, com seus códigos hexadecimais.

Listagem:

```
<table align='center' cellspacing='10'>
<tr>
<td>
<table align="center" border="1" cellspacing="0" cellpadding="0"
       style="text-align:center" width="250">
<tr>
<th> </th><th> R </th><th> G </th><th> B </th>
</tr>
<tr>
 <td style="background-color:rgb(0,0,0)">#000000</td>
 <td>  0</td><td>  0</td><td>  0</td>
</tr>
<tr>
 <td style="background-color:rgb(127,127,127)">#7F7F7F</td>
 <td>127</td><td>127</td><td>127</td>
</tr>
<tr>
 <td style="background-color:rgb(255,255,255)">#FFFFFF</td>
 <td>255</td><td>255</td><td>255</td>
</tr>
<tr>
 <td style="background-color:rgb(255,0,0)">#FF0000</td>
 <td>255</td><td>0</td><td>0</td>
</tr>
<tr>
 <td style="background-color:rgb(0,255,0)">#00FF00</td>
 <td>0</td><td>255</td><td>0</td>
</tr>
</table>
</td>
<td>
<table align="center" border="1" cellspacing="0" cellpadding="0"
       style="text-align:center" width="250">
<tr>
 <th> </th><th> R </th><th> G </th><th> B </th>
</tr>
<tr>
```

```
 <td style="background-color:rgb(0,0,255)">#0000FF</td>
 <td>0</td><td>0</td><td>255</td>
</tr>
<tr>
 <td style="background-color:rgb(255,255,0)">#FFFF00</td>
 <td>255</td><td>255</td><td>0</td>
</tr>
<tr>
 <td style="background-color:rgb(255,0,255)">#FF00FF</td>
 <td>255</td><td>0</td><td>255</td>
</tr>
<tr>
 <td style="background-color:rgb(0,255,255)">#00FFFF</td>
 <td>0</td><td>255</td><td>255</td>
</tr>
<tr>
 <td style="background-color:rgb(255,127,63)">#FF7F3F</td>
 <td>255</td><td>127</td><td>63</td>
</tr>
</table>
</td>
</tr>
</table>
```

Resultado:

	R	G	B
	0	0	0
#7F7F7F	127	127	127
#FFFFFF	255	255	255
#FF0000	255	0	0
#00FF00	0	255	0

	R	G	B
#0000FF	0	0	255
#FFFF00	255	255	0
#FF00FF	255	0	255
#00FFFF	0	255	255
#FF7F3F	255	127	63

Note em particular os tons de cinza, indo do preto (`rgb(0,0,0)` ou hexadecimal `#000000`) ao branco (`rgb(255,255,255)` ou `#FFFFFF`), que são gerados fazendo com que as contribuições das três cores sejam iguais.

Sua vez.. (8-7)

O script a seguir define uma função que sorteia uma cor aleatória. A função é então utilizada para atribuir uma cor de fundo para um parágrafo que contém apenas um ` ` (*non breakable space*) para que não seja tratado como vazio, de modo que o navegador mostre uma faixa com a cor sorteada, como a que segue.

```
<script>
function geraCor() {
    var R = parseInt(255*Math.random());
    var G = parseInt(255*Math.random());
    var B = parseInt(255*Math.random());
    var cor = "rgb(" + R + "," + G + "," + B + ")";
    return cor;
}
var estilo = "style='background-color:" + geraCor() + "'";
```

```
var str = "<p " + estilo + "> </p>";
document.write(str);
</script>
```

Substitua as reticências (. . .) no script a seguir de modo que ele produza uma tabela com uma linha e quatro colunas, cada uma com uma cor aleatoriamente sorteada.

```
<script>
var str = "";
str += "<table style='width:100%'>";
str += "<tr>";
for (var i=0;i<4;i++) {
    var estilo = ...
    str += ...
}
str += "</tr>";
str += "</table>";
document.write(str);
</script>
```

Exercícios

1. Reproduza o seguinte texto HTML utilizando listas ordenadas e não-ordenadas:

   I. Mecânica Quântica: fundamentos
      A. Partículas e campos
         - Difração de partículas
         - Pacotes de ondas
      B. Princípio da incerteza
         - Relação de incerteza posição-momento
         - Relação de incerteza energia-tempo
   II. Mecânica Quântica: aplicações
      A. Equação de Schröedinger
      B. Potenciais
         - Potencial degrau
         - Poço quadrado infinito
         - Oscilador harmônico

2. Reproduza o seguinte texto HTML, no qual estão embutidos trechos de JavaScript que sorteiam o ângulo (entre 30 e 60 graus) e o módulo da velocidade de lançamento de um projétil (entre 5 m/s e 15 m/s) e calcula a sua altura máxima e o alcance em função dos valores sorteados:

   Em um lançamento oblíquo, um projétil lançado a um ângulo de 40 graus com uma velocidade inicial de 12 m/s vai atingir uma altura máxima de 3.0 m e terá

um alcance de 14.5 m. O tempo de vôo é de 1.6 s.

3. Reproduza as equações a seguir utilizando elementos do HTML:

   a. $\text{sen}(A \pm B) = \text{sen}\ A \cos B \pm \cos A\ \text{sen}\ B$

   b. $\text{sen}\ \theta = (e^{i\theta} - e^{-i\theta}) / 2i$

   c. $\int a f(x)\, dx = a \int f(x)\, dx$

   d. $\square \times \mathbf{B} = \mu_0 \varepsilon_0\, d\mathbf{E}/dt$

4. Reproduza as equações a seguir utilizando a sintaxe do $\LaTeX$ e o MathJax.js.

   a. $\frac{d}{dt}(\frac{\partial L}{\partial \dot{x}_i}) - \frac{\partial L}{\partial x_i} = 0$

   b. $T = \frac{1}{2}mv^2 + \frac{1}{2}I\dot{\theta}^2$

   c. $\Psi_{21\pm1} = \frac{1}{8\sqrt{2\pi}}\left(\frac{Z}{a_0}\right)^{3/2}\left(\frac{Zr}{a_0}\right)e^{-Zr/2a_0}\ \sin\theta\, e^{\pm i\phi}$

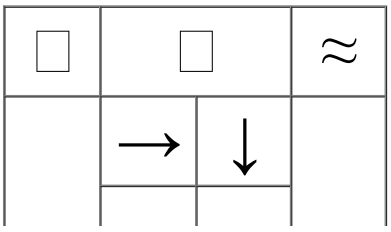
5. O trecho de script a seguir define uma matriz com 5 linhas e 5 colunas. Cada linha representa os dados de um aluno, a saber, matrícula, notas em três provas e a média final (inicialmente zerada):

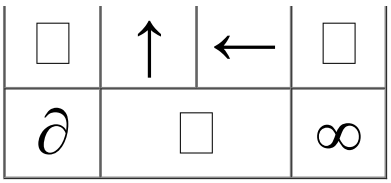
```
var aluno = new Array(5);
aluno[0] = ["08234045", 10,  9, 10,  0];
aluno[1] = ["08134033",  8,  7,  3,  0];
aluno[2] = ["09147001",  9,  7,  5,  0];
aluno[3] = ["09147076",  9,  6,  9,  0];
aluno[4] = ["08247804",  7,  7,  7,  0];
```

   Complete o script de tal modo que imprima uma tabela semelhante à que segue:

Matrícula	P1	P2	P3	Média
08234045	10.0	9.0	10.0	9.7
08134033	8.0	7.0	3.0	6.0
09147001	9.0	7.0	5.0	7.0
09147076	9.0	6.0	9.0	8.0
08247804	7.0	7.0	7.0	7.0

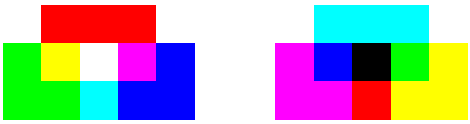
6. Reproduza a seguinte tabela:





Dica: é uma tabela 4 × 4 com os atributos `colspan` e `rowspan` para as células estrategicamente utilizados. Procure *html entities* na internet para obter os códigos dos símbolos apresentados na tabela.

7. As figuras abaixo mostram, à esquerda, o esquema de composição de cores aditivo, que é o que ocorre quando somamos luz, como em um monitor de vídeo, e, à direita, o esquema subtrativo, que é o que ocorre quando somamos tintas, como em uma impressora. No esquema aditivo as cores primárias são o vermelho, o verde e o azul. O vermelho com o verde leva ao amarelo, o vermelho com o azul leva ao magenta, o verde com o azul leva ao ciano e a soma dos três leva ao branco. No esquema subtrativo as cores primárias são o ciano, o magenta e o amarelo. O ciano com o magenta leva ao azul, o ciano com o amarelo leva ao verde, o magenta e o amarelo leva ao vermelho e a soma dos três leva ao preto.



As figuras foram montadas como duas tabelas de 3 linhas com 5 colunas cada. Reproduza-as!

8. O script abaixo escreve uma mensagem com letras vermelhas em meio a um conjunto de letras escolhidas e coloridas aleatoriamente. Se você pegar um pedaço de papel celofane vermelho (ou qualquer outro material translúcido vermelho) verá que a mensagem fica realçada enquanto as letras de outras cores ficam esmaecidas.

```
<script>
var cor = new Array("blue", "yellow", "green");
var msg = "AS CORES QUE VOCÊ VÊ DEPENDEM DA LUZ QUE VOCÊ USA";
var txt = "<p style='font-family:cursive; font-weight:bold;'>";
for (var i=0; i<(msg.length*4); i++) {
   var icor = parseInt(Math.random()*3);
   var ltr = String.fromCharCode(65+parseInt(Math.random()*26));
   txt += "<font color='" + cor[icor];
   txt += "'>" + ltr + "</font>" + " ";
   if ((i%4)==0) {
      txt += "<font color='red'>";
      txt += msg.charAt(i/4);
      txt += "</font>" + " ";
   }
}
txt += "</p>";
document.write(txt);
</script>
```

O script cria e inicializa uma matriz cujos três elementos são as cores alternativas à da mensagem; cria e inicializa a mensagem a ser apresentada em vermelho; cria a variável que vai conter o texto final a ser impresso e inicializa-a com a formatação do parágrafo

(fonte e peso das letras). Um laço `for` executado um número de vezes 4 vezes maior do que o tamanho da mensagem sorteia uma letra (um código entre 65 e 91) e em seguida um número entre 0 e 2 que vai indexar a cor da letra; adiciona a letra sorteada especificando a sua cor e finalmente, a cada quatro interações, coloca uma letra da mensagem. Findo o laço, encerra o parágrafo e imprime a mensagem.

W A M V J J S K A P F R I Z V C J F W U O R E E C R T O L Z E
H P L S S S M L M Q C U N Q R D L J U L X J T E S L F N T M X
K V O P K M O C H I X C L U N K Ê L U L B X N K I V B L H Q Ê E
J T G W V J T D Q W C O E N J C Y P Q E T N E I H B J N X H O
P D J V I L E S G J K M P P H Y X T T O D Q H O E A R P T M O
Q J D L Y E M Q U K Z M Q Z N P S L G P A Q Q J Z J H U Y C E
C E L U T Q I S F C V Y W I R O H M Y C C Z Q C U Ê X G H U N
N G Q U E K F N S S K X H A T G G

Modifique o script de modo que imprima 3 mensagens diferentes, uma vermelha, uma amarela e outra verde e que inclua mais duas cores a serem sorteadas aleatoriamente.