DevWeb

Capítulo 21

Conteúdos em Tabelas

Ao construir os conteúdos que vão compor o seu site, provavelmente você vai se deparar com a necessidade de criar uma ou mais tabelas. A HTML tem uma série de *tags* para a organização tabular de dados e precisamos dominá-las. Este capítulo vai te mostrar muitas peculiaridades das tabelas e suas várias possibilidades de personalização.

Tableless: tabelas são para apresentar conteúdos, não para construir estruturas de sites

Antes de começar a estudar as tabelas, vamos retirar um assunto da frente. Houve uma época em que os sites eram desenvolvidos como se fossem tabelas. Foi um tempo sombrio e que ficou no passado!

Com a evolução das CSS e com o surgimento da HTML na sua versão 5, os sites podem ser criados de forma mais organizada e dinâmica e não dependem mais das tabelas para existir, um movimento do design que ficou conhecido como *tableless layout* (traçado sem tabelas).

Mas não vamos demonizar as tabelas! Elas são boas e importantes, só não precisam ser usadas para criar a estrutura do seu site. Deixe as tabelas para o seu objetivo principal: apresentar dados de forma tabular. Só isso!

Nossa primeira tabela

Vamos começar com uma tabela simples para podermos entender a estrutura básica dos dados tabulados:

População das Unidades Federativas

Cidade	População
São Paulo	45.919.049
Minas Gerais	21.168.791
Rio de Janeiro	17.264.943
Bahia	14.873.064
Paraná	11.433.957
Total	**110.659.804**

FONTE DOS DADOS: A tabela anterior foi construída com base nos dados disponíveis em Junho de 2020 no site da Wikipedia, no link a seguir.

https://pt.wikipedia.org/wiki/Lista_de_unidades_federativas_do_Brasil_por_população

Vamos começar analisando a anatomia básica da tabela, demarcando algumas áreas importantes:

População das Unidades Federativas ❶	
Cidade ❷	**População** ❷
São Paulo	45.919.049
Minas Gerais	21.168.791
Rio de Janeiro	17.264.943 ❸
Bahia	14.873.064
Paraná	11.433.957
Total ❹	**110.659.804** ❹

Em primeiro lugar, temos o **título** (*caption*) da tabela, marcado com o **número 1**. Em seguida, temos o **cabeçalho** (*thead*) da tabela com o **número 2**, que serve para identificar cada coluna de dados. Já o **corpo** (*tbody*) da tabela apresenta todos os dados de forma tabular na área demarcada com o **número 3**. Por fim, temos uma área final que totaliza todos os dados, que é o **rodapé** (*tfoot*) da tabela, delimitado pela área de **número 4**.

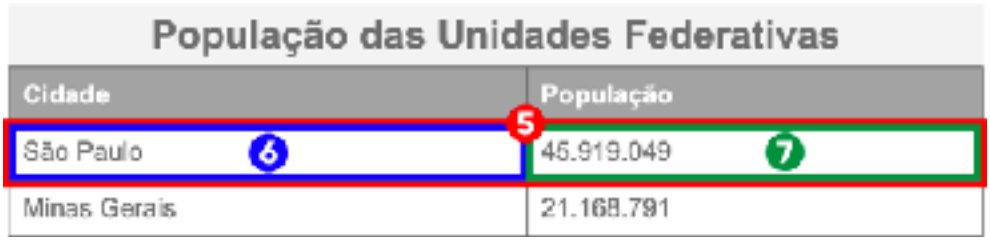
O código base para criar a tabela acima será:

```
<table>
    <caption>
        <!-- título da tabela -->
    </caption>
    <thead>
        <!-- cabeçalho da tabela -->
    </thead>
    <tbody>
        <!-- corpo da tabela -->
    </tbody>
    <tfoot>
        <!-- rodapé da tabela -->
    </tfoot>
</table>
```

Compare o código HTML com a estrutura da imagem com as quatro áreas delimitadas e tudo ficará bem claro. Esse será o "esqueleto" da nossa tabela que vai receber os dados.

SUA TABELA É GRANDE OU PEQUENA? Ao criar tabelas pequenas, não é tão necessário delimitar as áreas de <thead>, <tbody> e <tfoot>. Essas tags semânticas são mais usadas para tabelas com muito conteúdo, segundo a W3C. Mas eu sempre aconselho a usá-las para deixar claro a estrutura da sua tabela.

Agora vamos nos focar na estrutura das linhas de cada área para entender como demarcar a hierarquia de células:



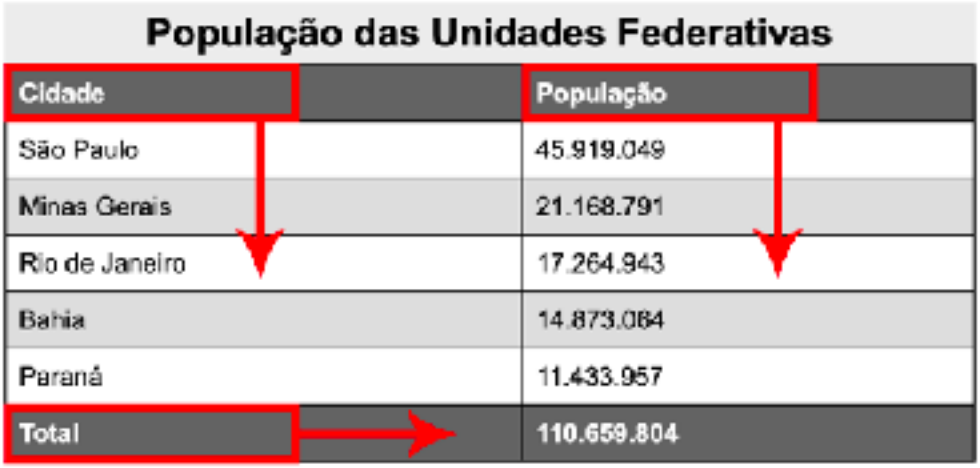
Em primeiro lugar, cada área é dividida em **linhas** (*table rows* - `tr`), que é a área marcada com o **número 5** e ocupa a largura total da tabela. Dentro das linhas, temos **células de dados** (*table data* - `td`) ou **células de cabeçalho** (*table header* - `th`). Sendo assim, para fazer o código relativo ao trecho que marcamos na imagem acima, vamos escrever:

```
<tbody> <!-- corpo da tabela -->
    <tr> <!-- primeira linha -->
        <td>São Paulo</td> <!-- primeira célula da primeira linha -->
        <td>45.919.049</td> <!-- segunda célula da primeira linha -->
    </tr>
    <tr> <!-- segunda linha-->
        <td>Minas Gerais</td> <!-- primeira célula da segunda linha -->
        <td>21.168.791</td> <!-- segunda célula da segunda linha -->
    </tr>
    ...
</tbody>
```

QUANDO USAR <td> OU <th>? Uma dúvida muito comum é quando usar cada uma dessas *tags*. De forma resumida, vamos usar o `<td>` quando a célula contiver um dado (**São Paulo**, por exemplo) e usaremos o `<th>` quando a célula contiver um identificador de dados (**Cidade**, por exemplo). Porém, ambos estarão dentro de linhas `<tr>`.

Ao usar o <th>, defina o escopo

Agora que você já sabe que a tag `<th>` é usada para definir títulos de colunas ou linhas, agora é importante saber o que é escopo desse título. Ainda considerando a tabela que estamos construindo, vamos nos focar nas células marcadas na imagem a seguir. Elas são as nossas células de título, marcadas com `<th>`.



População das Unidades Federativas

Cidade	População
São Paulo	45.919.049
Minas Gerais	21.168.791
Rio de Janeiro	17.264.943
Bahia	14.873.064
Paraná	11.433.957
Total	110.659.804

Perceba também que as células de "Cidade" e "População", possuem seus dados abaixo delas (o que chamamos de **escopo de coluna**). Já a célula "Total" tem seu dado ao lado dela (o que chamamos de **escopo de linha**). Sendo assim, vamos fazer as declarações pontuais em cada caso:

```
<tr>
    <th scope="col">Cidade</th>
    <th scope="col">População</th>
</tr>
...
<tr>
    <th scope="row">Total</th>
    <td>110.659.804</td>
</tr>
```

Existem ainda os escopos de `colgroup` e de `rowgroup`, mas só vamos falar sobre essas configurações nos vídeos do curso, pois precisamos antes falar de mesclagem de células com `rowspan` e `colspan`.

Minha tabela ficou feia, e agora?

Antes de falar em aplicar folhas de estilo a tabelas, precisamos entender que a hierarquia dos componentes de uma tabela é algo **primordial**! Qualquer erro na hierarquia, qualquer *tag* no lugar errado, vai causar a desorganização dos dados. Dessa maneira, se por acaso os dados não estiverem aparecendo na organização correta, é sinal de que existe um erro na estrutura da tabela.

Se a sua tabela estiver sendo apresentada na tela de forma organizada, mas sem bordas, sem cores e sem graça, chegou a hora de criar as configurações de CSS.
A primeira configuração será configurar tamanhos e bordas:

```css
table {
    width: 600px;
    border: 1px solid black;
    border-collapse: separate;
}

td, th {
    border: 1px solid black;
}
```

O resultado será uma tabela com as células delimitadas por bordas finas. Note que existe um pequeno espaço entre as células, que talvez incomode um pouco. Essa distância pode ser personalizada através da propriedade `border-spacing`.

População das Unidades Federativas

Cidade	População
São Paulo	45.919.049
Minas Gerais	21.168.791
Rio de Janeiro	17.264.943
Bahia	14.873.064
Paraná	11.433.957
Total	110.659.804

Para isso, mude o parâmetro `border-collapse` do valor `separate` para o valor `collapse`. A tabela vai mudar um pouco:

População das Unidades Federativas

Cidade	População
São Paulo	45.919.049
Minas Gerais	21.168.791
Rio de Janeiro	17.264.943
Bahia	14.873.064
Paraná	11.433.957
Total	110.659.804

Agora vamos adicionar um pequeno `padding` (8px) às configurações de `td` e `th` que fizemos anteriormente e adicionar mais alguns seletores bem simples e intuitivos:

```css
caption {
    font-weight: bolder;
    font-size: 1.5em;
    text-align: center;
    background-color: rgb(236, 236, 236);
    padding: 5px;
}
```

```css
thead {
    background-color: #636363;
    color: white;
}

tfoot {
    background-color: #636363;
    color: white;
    font-weight: bolder;
}
```

Todas as configurações acima já foram explicadas nos capítulos anteriores. Se surgir alguma dúvida, consulte o material impresso ou assista os vídeos relativos a cada capítulo. Como eu sempre digo, não tente fazer tudo na correria ou o resultado do seu aprendizado pode ser bem frustrante.

O resultado visual do código CSS acima aplicado à nossa tabela será:

População das Unidades Federativas

Cidade	População
São Paulo	45.919.049
Minas Gerais	21.168.791
Rio de Janeiro	17.264.943
Bahia	14.873.064
Paraná	11.433.957
Total	**110.659.804**

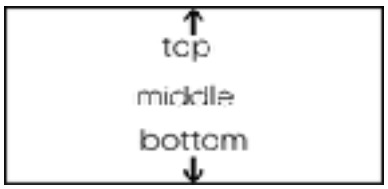
Alinhamento do conteúdo da célula

Alinhar o conteúdo de uma célula é muito simples e pode ser realizado em duas dimensões: a **horizontal** e a **vertical**.

O **alinhamento horizontal** é realizado exatamente da mesma maneira de uma `div` comum ou qualquer outro elemento exibido como um bloco: usando a propriedade `text-align`. Os valores aceitos para o alinhamento horizontal são:

Existe também o valor `justify`, que vai alinhar o conteúdo primordialmente pelo lado esquerdo, mas se o texto for grande, vai alinhar também à direita.

Já o **alinhamento vertical** é realizado pela propriedade `vertical-align`, que aceita os seguintes valores:



Como criar o efeito zebrado?

Nunca ouviu falar no *"efeito zebrado"* em tabelas? Pois saiba que o conceito é muito simples e consiste em intercalar as cores das células pares e ímpares, criando um efeito "listrado" (tá aí a referência a uma zebra) que facilita na leitura dos dados em tabelas muito grandes e com muito conteúdo.

Para obter esse resultado na tabela que estamos criando, vamos acrescentar os seguintes seletores ao estilo:

```
tbody > tr:nth-child(even) {
    background-color: #dddddd;
}

tbody > tr:nth-child(odd) {
    background-color: #ffffff;
}
```

A pseudo-classe `nth-child()` vai permitir selecionar as linhas **pares** (*even*) ou as linhas **ímpares** (*odd*), configurando uma cor para cada uma delas. No exemplo acima, inclusive, se o fundo da página já está branco, o segundo seletor (para linhas ímpares) torna-se desnecessário.

Mudando o alcance das células

É possível modificar o alcance das células para obter resultados ainda mais organizados. Isso é feito através das propriedades HTML nas células: `colspan` e `rowspan`. Dê uma boa olhada na tabela a seguir:

Grupo	Nomes	Filmes Favoritos		
Mulheres	Ana Maria Santos	Alien	Rambo	Vingadores
	Beatriz Souza	Hulk	Inception	Batman
	Cláudia Melo	Oblivion	Matrix	Big Hero
Homens	Bruno Mendonça	Intocáveis	Amnésia	Gladiador
	Daniel Lourenço	Wall-E	Oldboy	Dangal
	Fabiano Mota	Star Wars 5	Taxi Driver	Toy Story

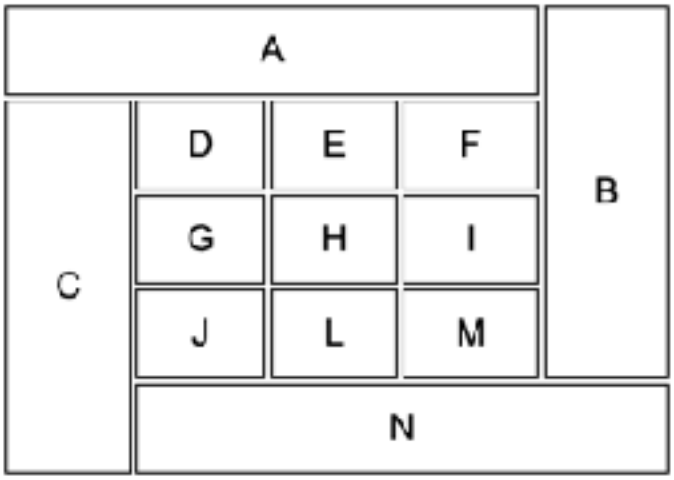
Note que, no cabeçalho da tabela, temos uma célula para *"Filmes Favoritos"* que está ocupando o espaço de três colunas. Enquanto isso, na lateral esquerda da tabela, as células com as palavras *"Mulheres"* e *"Homens"*, que estão ocupando tês linhas cada. Para efetuar essas mesclagens, basta usar as propriedades `colspan` e `rowspan`, respectivamente.

```
<table>
    <thead>
        <tr>
            <th>Grupo</th>
            <th>Nomes</th>
            <th colspan="3">Filmes Favoritos</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td rowspan="3">Mulheres</td>
            <td>Ana Maria Santos</td>
            <td>Alien</td>
            <td>Rambo</td>
            <td>Vingadores</td>
        </tr>
        <tr>
            <td>Beatriz Souza</td>
            <td>Hulk</td>
            <td>Inception</td>
            <td>Batman</td>
        </tr>
```

Para treinar um pouco mais, crie aí no seu ambiente de testes as seguintes tabelas: a primeira contendo números e com algumas mesclagens pontuais.

<table>
<tr><td>1</td><td>2</td><td rowspan="2">3</td><td>4</td></tr>
<tr><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="3">8</td><td>9</td></tr>
<tr><td rowspan="3">10</td><td colspan="2">11</td><td>12</td></tr>
<tr><td>13</td><td>14</td><td>15</td></tr>
</table>

A segunda tabela vai ser um pouco diferente, vai conter letras e terá as seguintes mesclagens:



Se você conseguiu fazer as tabelas acima, vamos ao desafio final e reproduza a seguinte tabela:

Área	Disciplinas	Notas		Média
		Nota 1	Nota 2	
Exatas	Matemática	0.0	0.0	0.0
	Física	0.0	0.0	0.0
	Química	0.0	0.0	0.0
	Biologia	0.0	0.0	0.0
Média de Exatas				0.0
Humanas	História	0.0	0.0	0.0
	Geografia	0.0	0.0	0.0
Média de Humanas				0.0

PRATIQUE MUITO! Os desafios acima usam algumas configurações muito importantes. Saber montar tabelas assim é essencial para provar a si mesmo que aprendeu direito! Não pule os desafios acima, pratique cada um deles em seu computador.

Personalização de colunas

Podemos personalizar uma linha inteira ou um conjunto de linhas de uma tabela criando seletores voltados para os `<tr>` da nossa tabela. Mas como podemos criar configurações específicas para as colunas?

Antes de mais nada, vamos criar uma tabela simples, com 4 linhas e 5 colunas:

```
<table>
    <tr>
        <td>1A</td>
        <td>1B</td>
        <td>1C</td>
        <td>1D</td>
        <td>1E</td>
    </tr>
    <tr>
        <td>2A</td>
        <td>2B</td>
        <td>2C</td>
        <td>2D</td>
        <td>2E</td>
    </tr>
    <tr>
        <td>3A</td>
        <td>3B</td>
        <td>3C</td>
        <td>3D</td>
        <td>3E</td>
    </tr>
    <tr>
        <td>4A</td>
        <td>4B</td>
        <td>4C</td>
        <td>4D</td>
        <td>4E</td>
    </tr>
</table>
```

O resultado visual dessa tabela, com algumas bordas e preenchimentos aplicados no estilo, será:

1A	1B	1C	1D	1E
2A	2B	2C	2D	2E
3A	3B	3C	3D	3E
4A	4B	4C	4D	4E

Note que temos 5 colunas, representadas pelos itens com as letras A, B, C, D e E respectivamente. Podemos criar um agrupamento de colunas, definindo um `<colgroup>` dentro da tabela, logo abaixo da tag `<table>`:

```
<colgroup>
    <col class="c1">
    <col class="c2" span="2">
    <col class="c3">
    <col class="c1">
</colgroup>
```

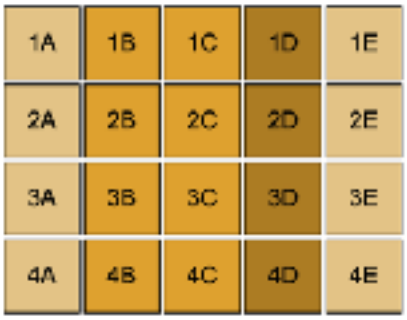
Isso vai atribuir uma classe a cada coluna (ou conjunto de colunas, quando usamos `span`). Ao criar uma configuração de cor para cada uma dessas três classes (`c1`, `c2` e `c3`), podemos criar um efeito de configuração em grupos de colunas:

```css
col.c1 {
    background-color: #E3C386;
}

col.c2 {
    background-color: #DEA12F;
}

col.c3 {
    background-color: #AC7C24;
}
```

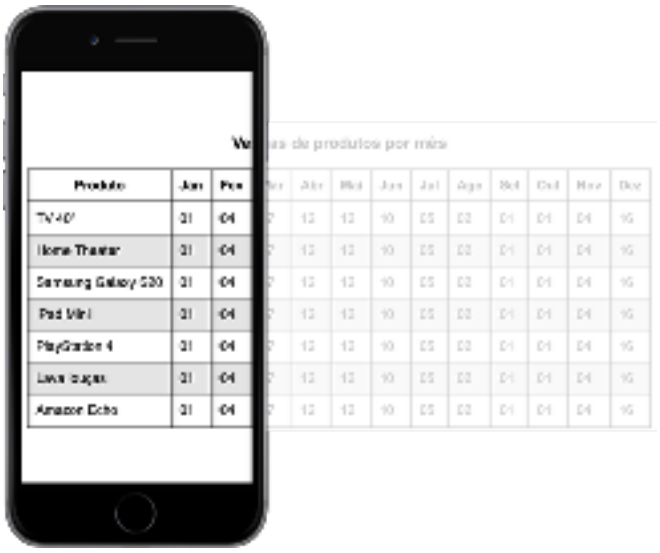
O resultado visual disso será:



Note que usar o span no HTML fez com que a segunda e a terceira colunas fizessem parte de um mesmo conjunto de colunas que receberam a mesma formatação de cor. Já a primeira e a última coluna tiveram a mesma cor de fundo aplicada, já que as duas `<col>` possuem a mesma classe configurada.

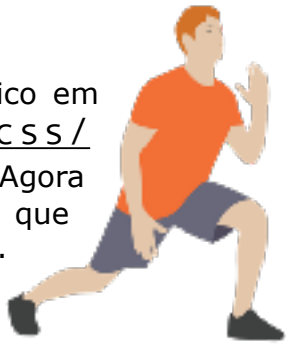
Tabelas largas e responsivas

O que será que acontece quando carregamos uma página que contenha uma tabela que tem muitas colunas e que esse conteúdo não caiba na largura da tela? A imagem a seguir mostra exatamente essa situação:



Para resolver essa limitação, principalmente para telas de celular, e garantir a visualização de todos os dados, uma das soluções mais simples é colocar a tabela dentro de uma caixa qualquer (pode ser uma `div`) e configurar a sua propriedade CSS `overflow-x` para os valores `auto` ou `scroll`. Isso vai fazer com que todo e qualquer conteúdo que *"transborde"* a largura (eixo x) do tamanho do *box*, vai ganhar uma barra de rolagem horizontal (por se tratar do eixo x apenas) caso seja necessário.

Hora de exercitar



Chegou a hora de acessar o endereço do nosso repositório público em https://gustavoguanabara.github.io/html-css/desfios/ e executar os **exercícios 023** no seu computador. Agora tente atingir esse mesmo resultado em casa, sem copiar o código que eu criei. Nesse momento, a prática é algo que você mais precisa. Se por acaso ficar difícil, pode acessar o repositório público de HTML e CSS e dar uma olhada nos comandos, mas **EVITE COPIAR**.

Eu já falei sobre isso no YouTube?

Eu sei que às vezes as pessoas gostam mais de assistir vídeos do que ler livros, e é por isso que eu lanço há anos materiais no canal Curso em Vídeo no YouTube. O link que vou compartilhar contigo tem o conteúdo explicado como você leu aqui, só que de forma mais ilustrada. Reserve um tempo dos seus estudos para assistir esse vídeo todo.

Curso em Vídeo: https://www.youtube.com/playlist?list=PLHz_AreHm4dlAnJ_jJtV29RFxnPHDuk9o