

MÓDULO

PROGRAMAÇÃO WEB – HTML/CSS

UNIDADE

EFEITOS VISUAIS E ESTILOS DE LISTAS

ÍNDICE

OBJETIVOS.....	3
INTRODUÇÃO.....	4
1. CRIAÇÃO DE ELEMENTOS <DIV> E 	5
1.1. A PROPRIEDADE MAX-WIDTH	9
2. EFEITOS VISUAIS.....	10
2.1. A PROPRIEDADE OVERFLOW.....	10
2.2. A PROPRIEDADE CLIP	13
2.3. A PROPRIEDADE DISPLAY	16
2.3.1. DISPLAY – BLOCK.....	17
2.3.2. DISPLAY – INLINE.....	17
2.3.3. DISPLAY – NONE.....	18
2.3.4. DISPLAY – INLINE-BLOCK	19
3. ESTILOS DE LISTAS	25
3.1. A PROPRIEDADE LIST-STYLE-TYPE	25
3.2. A PROPRIEDADE LIST-STYLE-IMAGE	30
3.3. A PROPRIEDADE LIST-STYLE-POSITION.....	32
3.4. SHORTHAND LIST.....	34
CONCLUSÃO.....	37
AUTOAVALIAÇÃO	39
SOLUÇÕES.....	43
PROPOSTAS DE DESENVOLVIMENTO DO ESTUDO.....	44
BIBLIOGRAFIA	45

OBJETIVOS

Com esta unidade didática, pretende-se que desenvolva os seguintes objetivos de aprendizagem:

- Criar elementos HTML através da definição da sua largura (width) e altura (height).
- Alterar os elementos HTML e o seu conteúdo, e controlar o comportamento dos mesmos através de propriedades de efeitos visuais.
- Estilizar listas ordenadas e não ordenadas.

INTRODUÇÃO

Uma página web bem estruturada e com os elementos corretos no lugar certo é uma mais valia em web design, porque não só mostra profissionalismo como torna a navegação na página mais fácil e intuitiva.

Esta unidade servirá para compreender quais são os elementos a usar e quando deve usá-los, e como se podem controlar, visualizar e alterar para que a página web fique bem estruturada e com uma aparência melhorada.

1. CRIAÇÃO DE ELEMENTOS <DIV> E

Já foi abordado neste módulo, em exemplos anteriores, a criação de elementos div e span em HTML/CSS.

Uma div é um elemento que gera uma caixa que tem, implicitamente, uma linha de quebra no final. Por outro lado, um span é um elemento que gera uma caixa simples.

Se não forem definidas dimensões para estas caixas, as mesmas ocuparão, por defeito, toda a largura (width) e altura (height) que necessitarem. Por exemplo, se não forem definidas medidas para uma caixa que contenha um texto composto por duas linhas, a caixa ocupará o espaço referente às duas linhas.

As medidas das caixas são definidas pelas propriedades width e height e podem ser expressas em qualquer unidade aceite por HTML/CSS, sendo que as mais comuns são px e %. Por exemplo, uma caixa com 200px de largura, terá, exatamente, 200 px de largura, enquanto uma caixa com 80% de largura, ocupará 80% da largura da janela de visualização, independentemente do zoom que se possa aplicar.

A sintaxe será:

```
elemento{  
    width: valor px | valor %;  
    height: valor px | valor %;  
}
```

Veja-se um exemplo de dois elementos div, com dimensões definidas em px e em %.

Exemplo

As dimensões da caixa 1 são definidas em px. Caso se aplique zoom, o elemento não sofrerá qualquer alteração na sua dimensão. Já a caixa 2, como está definida em %, sempre que se aplicar zoom na página, sofrerá alteração na sua dimensão e adaptar-se-á à janela de visualização. Recomenda-se então que copie o código para o seu editor e abra no browser para que possa ver a diferença.

```
<!DOCTYPE html>  
  
<html lang="pt">  
  
<head>  
    <title> div e span css</title>  
  
    <meta charset="UTF-8">  
    <meta name="description" content="div e span css">  
    <meta name="author" content="Sara Granja">  
  
<style>
```



```
#caixa1{  
  
    width:200px;  
    color:#ffffff;  
    background-color:#ff0000;  
}
```

```
#caixa2 {  
    width:50%;  
    color:#ffffff;  
    background-color:#ff0000;  
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1>Criação de caixas div e span - propriedades <i>width</i> e  
<i>height </i></h1>
```

```
<div id="caixa1">
```

Esta caixa tem 200 px de largura.

Não foi definida altura e, portanto, esta vai ser automática e ocupará o espaço que necessitar para mostrar todo o conteúdo.

```
</div>
```

```
<br>
```

```
<div id="caixa2">
```

Esta caixa ocupa 50% da largura da janela de visualização.

Não foi definida altura e, portanto, esta vai ser automática e ocupará o espaço que necessitar para mostrar todo o conteúdo.

Experimente fazer *zoom*, para que observe que a caixa se mantém sempre com a mesma largura.

```
</div>
```

```
</body>
```

```
</html>
```

Criação de caixas `div` e `span` - propriedades *width* e *height*

Esta caixa tem 200 px de largura. Não foi definida altura e, portanto, esta vai ser automática e ocupará o espaço que necessitar para mostrar todo o conteúdo.

Esta caixa ocupa 50% da largura da janela de visualização. Não foi definida altura e, portanto, esta vai ser automática e ocupará o espaço que necessitar para mostrar todo o conteúdo. Experimente fazer *zoom*, para que observe que a caixa se mantém sempre com a mesma largura

1.1. A PROPRIEDADE MAX-WIDTH

Pode acontecer um problema aquando da criação de uma div: se a janela de um browser tiver uma largura menor do que um elemento div, o browser vai adicionar um scrollbar horizontal à página web.

É possível resolver esse problema através da propriedade max-width, definindo, assim, a largura máxima de um elemento. Esta propriedade pode ser especificada em qualquer unidade reconhecida por HTML/CSS (px, cm, %, etc.).



2. EFEITOS VISUAIS

Os efeitos visuais são usados para mudar a aparência visual dos elementos e o do seu conteúdo, mas também para controlar o comportamento dos mesmos.

2.1. A PROPRIEDADE OVERFLOW

A propriedade overflow permite controlar o comportamento de um elemento div (uma caixa, por exemplo), no caso em que a área ocupada pelo seu conteúdo é maior do que a área ocupada pelo próprio elemento.

Exemplo

Imagine um texto que é maior do que o elemento em que está inserido. Como se pode observar, o conteúdo irá aparecer fora da caixa definida.

A propriedade *overflow*

O texto abaixo, é muito longo para as dimensões da caixa que o contém.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exercicio ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis in vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue du dolore facilisi te feugiat nula Nam liber tempor cum soluta nobis eleifend opção congue nihil imperdiet doming id quod mazim placerat facer possim presumir. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem.

Para resolver o problema do exemplo acima, pode recorrer à propriedade *overflow*. Esta propriedade pode assumir um dos seguintes valores:

- **visible**: valor padrão. O conteúdo pode ser visto fora do elemento que o contém (exemplo mostrado acima).
- **hidden**: o conteúdo que pode ser visto é apenas aquele que “cabe” no elemento que o contém.

A propriedade *overflow*

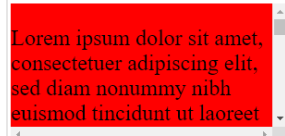
O texto abaixo, é muito longo para as dimensões da caixa que o contém.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat

- scroll: é gerado um scrollbar para que o utilizador possa rolar a página e ver todo o conteúdo.

A propriedade *overflow*

O texto abaixo, é muito longo para as dimensões da caixa que o contém.

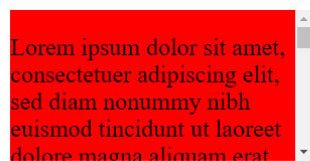
A screenshot of a web browser showing a text box with a red background. The text inside is "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet". The text is truncated on the right side, and a vertical scrollbar is visible on the right edge of the text box, indicating that the content is longer than the container's width.

Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed diam nonummy nibh
euismod tincidunt ut laoreet

- auto: semelhante ao scroll, mas apenas gera scrollbars quando é necessário.

A propriedade *overflow*

O texto abaixo, é muito longo para as dimensões da caixa que o contém.

A screenshot of a web browser showing a text box with a red background. The text inside is "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat". The text is truncated on the right side, and a vertical scrollbar is visible on the right edge of the text box, indicating that the content is longer than the container's width.

Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed diam nonummy nibh
euismod tincidunt ut laoreet
dolore magna aliquam erat

A sintaxe da propriedade overflow é:

```
elemento{  
    overflow: valor;  
}
```

2.2. A PROPRIEDADE CLIP

A propriedade clip é usada quando se pretende recortar uma imagem. Esta propriedade permite especificar um retângulo com determinadas medidas. Este retângulo é definido através de quatro coordenadas, que são especificadas em relação ao canto superior esquerdo do elemento a ser recortado.

A sua sintaxe é:

```
elemento{  
    clip: valor;  
}
```

Esta propriedade pode assumir os seguintes valores:

- auto: valor padrão. Não será aplicado nenhum recorte ao elemento.
- shape: recorta um elemento na forma retangular tendo em conta as coordenadas (top, right, bottom e left). A forma de a definir é clip: rect (**top**, **right**, **bottom** e **left**).
- Initial: especifica a propriedade com o seu valor inicial (por defeito, auto).
- inherit: herda o valor do seu elemento pai.

Exemplo

Neste exemplo iremos aplicar a propriedade clip com o valor shape. Recomendase que o formando copie o código para o seu editor e mude os valores top, right, bottom e left para compreender melhor como a propriedade funciona.

```
<!DOCTYPE html>

<html lang="pt">

<head>

  <title> div e span css</title>

  <meta charset="UTF-8">

  <meta name="description" content="div e span css">

  <meta name="author" content="Sara Granja">

  <style>

    #imagem-clip {

      position: absolute;

      clip: rect(0px,250px,150px,0px);

    }

  </style>

</head>

<body>

  <h1>A propriedade <i>clip</i></h1>
```



```
<h2> Imagem original </h2>



<h2> Imagem aplicando propriedade clip </h2>



</body>

</html>
```

A propriedade *clip*

Imagem original



Imagem aplicando propriedade clip





Curiosidade

Esta propriedade é muitas vezes utilizada em páginas web para tentar acelerar o carregamento de imagens.



Nota

A propriedade clip não funciona se estiver definido o overflow como visible.

2.3. A PROPRIEDADE DISPLAY



**Atenção**

A propriedade display é a mais importante de todas as propriedades para controlar o layout de uma página web.

A propriedade display determina se e como um elemento é exibido. Todos os elementos HTML, dependendo do que são, têm uma propriedade display padrão definida. Para a grande maioria dos elementos esta propriedade assume o valor block ou inline. O display pode ainda assumir os valores none e inline-block.

2.3.1. DISPLAY – BLOCK

Um elemento que tenha atribuída uma propriedade display:block, começa sempre numa nova linha e ocupa toda a largura da janela de visualização disponível.

A sua sintaxe é:

```
elemento{  
    display:block;  
}
```

Alguns exemplos de elementos block são: divs, títulos (h1 a h6), parágrafos (p), formulários (form), cabeçalhos (header), rodapés (footer) e secções (section).

2.3.2. DISPLAY – INLINE

Elementos inline são elementos que não começam numa nova linha e apenas ocupam o espaço necessário, tendo em conta o seu conteúdo.

A sua sintaxe é:

```
elemento{  
    display:inline;  
}
```

Alguns exemplos de elementos inline são: span, links (a) e imagens (img).

2.3.3. **DISPLAY – NONE**

A propriedade display:none é usada para esconder elementos, sem os apagar do código.

A sua sintaxe é:

```
elemento{  
    display:none;  
}
```



+ Informações

A propriedade display:none pode ser substituída pela propriedade visibility:hidden mas, esta última, irá continuar a ocupar o espaço na página web e estar ainda presente no layout.

2.3.4. DISPLAY – INLINE-BLOCK

O valor inline-block da propriedade display é semelhante ao valor inline, mas permite definir uma largura e uma altura de um elemento, os paddings superior e inferior são respeitados, ao contrário do que acontece com o valor inline. Comparando com o valor block, a grande diferença é a de que não cria uma quebra de linha após o elemento, logo é possível posicionar os elementos lado a lado.

A sua sintaxe é:

```
elemento{  
  display:inline-block;  
}
```

Exemplo

Veja-se a diferença entre elementos com valores de display inline, block e inline-block.

```
<!DOCTYPE html>  
<html lang="pt">  
<head>  
  <title> display css</title>  
  
  <meta charset="UTF-8">  
  <meta name="description" content="display css">  
  <meta name="author" content="Sara Granja">  
  
<style>  
span.span-inline {  
  width: 100px;  
  height: 100px;
```

```
padding: 5px;
background-color: coral;
}
span.span-block {
display: block;
width: 100px;
height: 100px;
padding: 5px;
background-color: coral;
}
span.span-inlineblock {
display: inline-block;
width: 100px;
height: 100px;
padding: 5px;
background-color: coral;
}
</style>
</head>

<body>

<h1>A propriedade <i>display</i> - valores inline, block e inline-
block</h1>

<h2> Inline </h2>

<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Vestibulum consequat scelerisque elit sit amet consequat.

Aliquam erat volutpat. <span class="span-inline">Aliquam</span>
<span class="span-inline">venenatis</span> gravida nisl sit amet
facilisis.

Nullam cursus fermentum velit sed laoreet. </div>
```

```
<h2> Block </h2>
```

```
<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Vestibulum consequat scelerisque elit sit amet consequat.
```

```
Aliquam erat volutpat. <span class="span-block">Aliquam</span>  
<span class="span-block">venenatis</span> gravida nisl sit amet  
facilisis.
```

```
Nullam cursus fermentum velit sed laoreet. </div>
```

```
<h2> Inline-block </h2>
```

```
<div>Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Vestibulum consequat scelerisque elit sit amet consequat.
```

```
Aliquam erat volutpat. <span class="span-  
inlineblock">Aliquam</span> <span class="span-  
inlineblock">venenatis</span> gravida nisl sit amet facilisis.
```

```
Nullam cursus fermentum velit sed laoreet. </div>
```

```
</body>
```

```
</html>
```

A propriedade *display* - valores inline, block e inline-block

Inline

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

Block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat.

Aliquam

venenatis

gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

Inline-block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit

sed laoreet.

Exemplo

Neste exemplo, alterou-se o valor do `display` de uma lista de `block` (padrão) para `inline` e para `inline-block`, para que seja possível ver a diferença entre os três valores.

```
<!DOCTYPE html>
<html lang="pt">
<head>
  <title> display css</title>

  <meta charset="UTF-8">
  <meta name="description" content="display css">
  <meta name="author" content="Sara Granja">

  <style>

    li.list-inline {
      display: inline;
    }

  </style>
</head>

<body>

  <h1>A propriedade <i>display</i></h1>

  <h2> Esta é uma lista com valor display padrão (block) </h2>
```



```
<ul>

  <li>Item 1</li>

  <li>Item 2</li>

  <li>Item 3</li>

</ul>


<h2> Esta é uma lista com o display inline </h2>


<ul>

  <li class="list-inline">Item 1</li>

  <li class="list-inline">Item 2</li>

  <li class="list-inline">Item 3</li>

</ul>


</body>

</html>
```

A propriedade *display*

Esta é uma lista com valor display padrão (block)

- Item 1
- Item 2
- Item 3

Esta é uma lista com o display inline

Item 1 Item 2 Item 3

Definir a propriedade de exibição de um elemento altera apenas a forma como o elemento é exibido, não altera que tipo de elemento ele é. Portanto, a um elemento embutido com `display:block`; não é permitido ter outros elementos de bloco dentro dele.



Nota

O `display` altera apenas a forma como o elemento é definido e não o tipo de elemento que ele é.

Não é permitido, por isso, que um elemento `inline` com `display:block` tenha outro elemento `block` dentro dele.

3. ESTILOS DE LISTAS

Recorde-se que, em HTML, as listas são uma série de textos mostrados em lista descendente, ordenados () ou não (). O CSS permite alterar o estilo de listas, como, por exemplo, a sua numeração, o estilo de “bullet”, inserir uma imagem como marcador de uma lista, adicionar cores de fundo, etc. Este ponto dedicar-se-á ao estudo das propriedades que permitem realizar tais ações.

3.1. A PROPRIEDADE LIST-STYLE-TYPE

Com a propriedade list-style-type é possível especificar o tipo de marcador (bullet ou numeração) de uma lista. Segue-se uma lista com alguns valores que esta propriedade pode assumir:

- disc: gera um bullet estilo disco.
- circle: gera um bullet estilo círculo.
- square: gera um bullet estilo quadrado.
- decimal: gera numeração decimal.
- decimal-leading-zero: gera numeração decimal iniciando no zero.
- lower-roman: gera numeração romana minúscula.
- upper-roman: gera numeração romana maiúscula.
- lower-greek: gera letra grega minúscula.

- lower-latin: gera letra latina minúscula.
- upper-latin: gera letra latina maiúscula.
- armenian: gera letra arménia.
- georgian: gera letra georgiana.
- lower-alpha: gera letra alfabeto minúscula.
- upper-alpha: gera letra alfabeto maiúscula.
- none: não mostra bullet nem numeração.

A sintaxe é:

```
elemento{  
    list-style-type: valor;  
}
```

Exemplo

```
<!DOCTYPE html>  
<html lang="pt">  
<head>  
    <title> display css</title>  
    <meta charset="UTF-8">  
    <meta name="description" content="display css">  
    <meta name="author" content="Sara Granja">  
<style>  
ul.a {  
    list-style-type: circle;  
}  
ul.b {  
    list-style-type: square;
```

```
}  
ol.c {  
    list-style-type: upper-roman;  
}  
ol.d {  
    list-style-type: lower-latin;  
}  
ol.e {  
    list-style-type: disc;  
}  
ol.f{  
    list-style-type: lower-greek;  
}  
</style>  
</head>  
  
<body>  
<h2>Listas - A propriedade <i>list-style-type</i></h2>  
<p>Exemplos de listas não ordenadas:</p>  
<ul class="a">  
    <li>bullet circle 1</li>  
    <li>bullet circle 2</li>  
    <li>bullet circle 3</li>  
</ul>  
<ul class="b">  
    <li>bullet square 1</li>  
    <li>bullet square 2</li>
```

```
<li>bullet square 3</li>
</ul>
<ul class="e">
  <li>bullet disc 1</li>
  <li>bullet disc 2</li>
  <li>bullet disc 3</li>
</ul>

<p>Exemplos de listas não ordenadas:</p>
<ol class="c">
  <li>numeração upper-roman 1</li>
  <li>numeração upper-roman 2</li>
  <li>numeração upper-roman 3</li>
</ol>
<ol class="d">
  <li>numeração lower-alpha 1</li>
  <li>numeração lower-alpha 2</li>
  <li>numeração lower-alpha 3</li>
</ol>
<ol class="f">
  <li>numeração lower-greek 1</li>
  <li>numeração lower-greek 2</li>
  <li>numeração lower-greek 3</li>
</ol>

</body>
```

```
</html>
```

Listas - A propriedade *list-style-type*

Exemplos de listas não ordenadas:

- bullet circle 1
- bullet circle 2
- bullet circle 3

- bullet square 1
- bullet square 2
- bullet square 3

- bullet disc 1
- bullet disc 2
- bullet disc 3

Exemplos de listas não ordenadas:

- I. numeração upper-roman 1
- II. numeração upper-roman 2
- III. numeração upper-roman 3

- a. numeração lower-alpha 1
- b. numeração lower-alpha 2
- c. numeração lower-alpha 3

- α. numeração lower-greek 1
- β. numeração lower-greek 2
- γ. numeração lower-greek 3

Para remover os valores padrão dos bullets/numeração das listas basta aplicar a propriedade `list-style-type:none`.

3.2. A PROPRIEDADE LIST-STYLE-IMAGE

A propriedade list-style-image permite substituir o marcador de cada elemento da lista por uma imagem.

A sua sintaxe é:

```
elemento{  
    list-style-image: url("image");  
}
```

Exemplo

```
<!DOCTYPE html>  
  
<html lang="pt">  
  
<head>  
    <title> list css</title>  
  
    <meta charset="UTF-8">  
    <meta name="description" content="list css">  
    <meta name="author" content="Sara Granja">  
  
<style>  
  
    ul.img-b{  
        list-style-image: url("bullet-img.jpg");  
    }  
  
</style>
```



```
</head>

<body>

<h2>Listas - A propriedade <i>list-style-image</i></h2>

<ul class="img-b">
  <li>word 1</li>
  <li>word 2</li>
  <li>word 3</li>
</ul>

</body>

</html>
```

Listas - A propriedade *list-style-image*

- ✕ word 1
- ✕ word 2
- ✕ word 3

3.3. A PROPRIEDADE LIST-STYLE-POSITION

A posição dos bullets/numeração de uma lista pode ser aplicada com a propriedade `list-style-position`. Os marcadores podem estar dentro (`inside`) do item da lista ou fora (`outside`) do item da lista. Por defeito, os bullets/numeração têm sempre a propriedade `outside` atribuída, e são alinhados verticalmente.

Exemplo

Observe a diferença entre a aplicação da propriedade `list-style-position`: `outside` e `list-style-position: inside`.

```
<!DOCTYPE html>
<html lang="pt">
<head>
  <title> list css</title>
  <meta charset="UTF-8">
  <meta name="description" content="list css">
  <meta name="author" content="Sara Granja">
  <style>

  ul.lista-out{
    list-style-position: outside;
  }
  ul.lista-in{
    list-style-position: inside;
  }
</style>
</head>

<body>
```

```
<h2>Listas - A propriedade <i>list-style-position</i></h2>

<p> outside </p>
<ul class="lista-out">
  <li>item 1</li>
  <li>item 2</li>
  <li>item 3</li>
</ul>
<p> inside </p>
<ul class="lista-in">
  <li>item 1</li>
  <li>item 2</li>
  <li>item 3</li>
</ul>

</body>
</html>
```

Listas - A propriedade *list-style-position*

outside

- item 1
- item 2
- item 3

inside

- item 1
- item 2
- item 3

3.4. SHORTHAND LIST

Também para o estilo de listas, existe um shorthand code. Pode simplificar a estilização de listas através da seguinte propriedade:

```
list-style: type position image;
```

Exemplo

Imagine uma lista que tenha as seguintes propriedades:

```
lista {  
    list-style-type: square;  
    list-type-position: outside;  
    list-style-image: url("image.jpg");  
}
```

Este código pode ser simplificado através da propriedade shorthand. Assim, ficaria:

```
lista {  
    list-style: square outside url("image.jpg")  
}
```


CONCLUSÃO

Esta unidade dedicou-se ao estudo e à compreensão de alguns efeitos visuais a serem adicionados a elementos HTML, bem como estilos de listas.

Uma página web bem estruturada e com os elementos certos é essencial em web design, uma vez que, além de a componente visual ser uma mais-valia, torna a navegação na página mais fácil e intuitiva.

Deste modo, recorrer aos elementos div e span, definindo as suas dimensões através das propriedades width, height e max-width, alterar elementos HTML e controlar o comportamento dos mesmos através de propriedades de efeitos visuais, assim como estilizar listas ordenadas e não ordenadas são ações cruciais para apresentar uma página web bem estruturada e com uma aparência melhorada.

AUTOAVALIAÇÃO

1. **Qual a diferença entre um elemento div e um elemento span?**
 - a) O elemento div gera uma “caixa” que contém implícita uma quebra de linha no final, e o elemento span não inclui qualquer quebra de linha.
 - b) O elemento span gera uma “caixa” que contém implícita uma quebra de linha no final, e o elemento div não inclui qualquer quebra de linha.
 - c) Não existe qualquer diferença entre estes dois elementos.
 - d) O elemento div consiste numa divisória horizontal e o elemento span consiste numa divisória vertical.

2. **Qual é a diferença entre criar um elemento div com dimensões expressas em px e em %?**
 - a) Em px o elemento será sempre maior do que em %.
 - b) Em % o elemento terá um valor exato e em px o elemento adapta-se à dimensão da janela de visualização.
 - c) As dimensões de um elemento div nunca podem ser especificadas em %.
 - d) Em px o elemento terá um valor exato e em % o elemento adapta-se à dimensão da janela de visualização.

3. **Se um elemento div tiver uma largura (width) definida com o valor 80%, que espaço ocupará na janela de visualização?**
 - a) Exatamente 20% da largura da janela de visualização.
 - b) Exatamente 80 px de largura.
 - c) Exatamente 80% da largura da janela de visualização.
 - d) As larguras não podem ser definidas em %.

4. **Qual a forma correta de definir um elemento HTML com 500 px de largura, em CSS?**
 - a) elemento { largura:500px;}.
 - b) elemento { width:500px;}.
 - c) width-elemento { 500px;}.
 - d) elemento { width=500px;}.

5. **Para garantir que a largura de um elemento, por exemplo, div, não ultrapassa a largura da janela de visualização, qual é a propriedade recomendada a atribuir?**
 - a) width.
 - b) height.
 - c) max-width.
 - d) max-height.

6. **Caso pretenda adicionar scrollbars a um elemento overflow, qual é a opção a usar?**
 - a) Deve usar-se a propriedade visible.
 - b) Deve usar-se a propriedade hidden.
 - c) Deve usar-se a propriedade auto ou scroll.
 - d) Deve usar-se a propriedade scrollbar.

- 7. Para que serve o atributo clip?**
- a) Para inserir um clip de música.
 - b) Para marcar um elemento como importante.
 - c) Para inserir um vídeo.
 - d) Para recortar uma imagem em CSS.
- 8. Qual é a diferença entre o uso da propriedade `display:none` e `visibility:hidden`, quando se pretende ocultar um elemento HTML?**
- a) A propriedade `display:none`, apesar de ocultar o elemento, continua a ocupar espaço na página web.
 - b) A propriedade `visibility:hidden`, apesar de ocultar o elemento, continua a ocupar espaço na página web.
 - c) Não existe qualquer diferença entre ambos.
 - d) Os elementos HTML não aceitam a propriedade `display:none`.
- 9. Se pretender que uma lista não ordenada contenha bullets tipo quadrado, qual é a propriedade que deve aplicar aos itens da lista?**
- a) `square`.
 - b) `circle`.
 - c) `disc`.
 - d) `rectangle`.
- 10. Qual é a diferença entre os valores `outside` e `inside` quando aplicados a um elemento com propriedade `list-style-position`?**
- a) Não existe qualquer diferença entre os dois valores.
 - b) O valor `outside` faz com que os bullets fiquem alinhados com o texto.
 - c) O valor `inside` faz com que os bullets fiquem fora do texto dos itens, enquanto que com o valor `outside`, os bullets ficam dentro do texto dos itens.
 - d) O valor `outside` faz com que os bullets fiquem fora do texto dos itens, enquanto que com o valor `inside`, os bullets ficam dentro do texto dos itens.

SOLUÇÕES

1.	a	2.	d	3.	c	4.	b	5.	c
6.	c	7.	d	8.	b	9.	a	10.	d

PROPOSTAS DE DESENVOLVIMENTO DO ESTUDO

Para aprofundar os conceitos abordados nesta unidade didática, assista ao vídeo de apoio e realize os exercícios correspondentes.

BIBLIOGRAFIA

- Willard, Wendy (2009), *HTML: A Beginner's Guide*. California: McGraw Hill.
- W3schools (2020), *CSS max-widtht property*. Disponível em: <https://www.w3schools.com/>. Consultado a 25 de novembro de 2020.
- Imagens provenientes de Shutterstock.

