Criando e posicionando um layout

9

- ✓ Dimensão real dos elementos;
- ✓ Configuração das margens dos elementos;
- ✓ Posicionamento dos elementos;
- ✓ Definição das camadas;
- ✓ Definição do modo de apresentação dos elementos;
- ✓ Tabindex;
- ✓ Tipos de mídia.

9.1.Introdução

A transformação de um layout em formato de imagem criado por um designer para um formato HTML que servirá um site ou Web App é um processo que requer alguns passos. Nesta leitura, abordaremos como criar um layout e posicioná-lo corretamente em um navegador, de maneira que envolva as folhas de estilo em cascata (CSS) e os elementos que aprendemos até aqui.

Um dos métodos utilizados para a criação de layout em HTML é o chamado **Tableless**, que é o nome atribuído à metodologia de construção de sites sem o uso de **tables** (tabelas). Para realizar essa construção, utiliza-se o HTML para determinar a estrutura dos dados e as folhas de estilo CSS para formatar sua exibição.

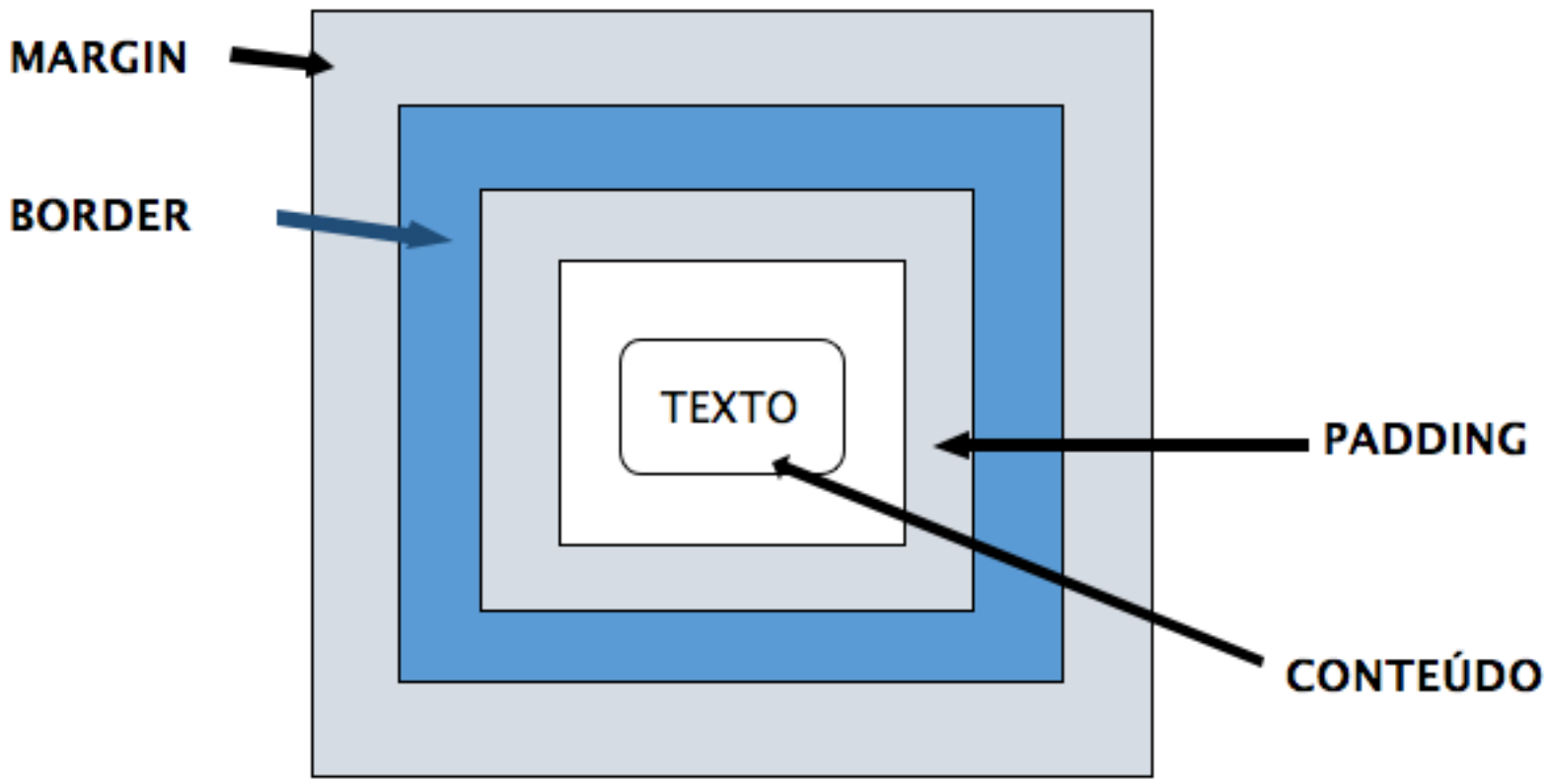
Antigamente, era comum o uso de tabelas para a estruturação e criação do layout da página, porém, desde o XHTML não há necessidade de utilizá-las.

Devemos ter em mente que as tags, inclusive a **<table>**, continuam a ser utilizadas de acordo com suas funções. A construção de sites por meio da metodologia **Tableless** requer a formatação de elementos por meio de arquivos CSS, que representam as folhas de estilo.

9.2.Dimensão real dos elementos

A dimensão real ocupada por um elemento dentro de uma página deve ser conhecida para que se possa compreender os conceitos envolvidos no processo de construção de um layout. A área ocupada por cada um dos elementos de uma página é denominada contêiner e tem a forma de um retângulo.

O contêiner de um elemento possui todos os itens que dizem respeito a ele, como seu conteúdo, seus espaços em branco, suas margens e suas linhas de contorno. O desenho a seguir demonstra quais são as áreas de um contêiner de um elemento. Observe:



Em que:

- **MARGIN**: Espaço que separa o contêiner de outros elementos que compõem a página. Vale destacar que a margem (margin) não está dentro dos limites de um elemento. As margens são utilizadas com a finalidade de mover a caixa do elemento;

- **CONTEÚDO**: Todos os itens contidos em um elemento, os quais permanecem dentro do contêiner;

- **BORDER**: Limites de um elemento, isto é, as linhas de contorno do contêiner;

- **PADDING**: Espaços em branco existentes entre o conteúdo e os limites de um elemento.

A largura do contêiner, no qual está contido o elemento, é determinada pela largura do conteúdo somada às larguras referentes ao **border** e ao **padding**. Já a largura do elemento em si é determinada apenas pela largura ocupada por seu conteúdo, assim como sua altura.

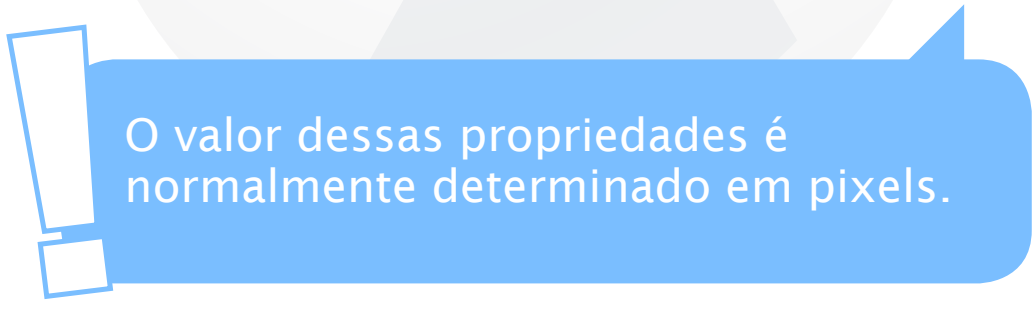
Em uma página, todos os elementos visíveis são de grupo ou **inline**. Os elementos de bloco têm sua largura determinada pela largura do bloco em que estão. Eles iniciam-se sempre em uma linha nova e, depois de finalizados, há uma nova mudança de linha. Já os elementos **inline** têm sua largura determinada apenas por seu conteúdo e, ao contrário dos elementos de bloco, não se iniciam sempre em uma nova linha. Dessa forma, concluímos que os elementos **inline** comportam-se da mesma maneira que um texto simples.

Como exemplo de elemento de bloco, podemos mencionar o **<table>**. Já como exemplo de um elemento **inline**, podemos mencionar o ****.

9.3.Configurando as margens dos elementos

Em CSS, a espessura das margens dos elementos é definida pelas propriedades dessas margens, as quais podem ser:

- **margin-bottom**: Permite determinar a espessura referente à margem inferior do elemento;
- **margin-top**: Permite determinar a espessura referente à margem superior do elemento;
- **margin-right**: Permite determinar a espessura referente à margem direita do elemento;
- **margin-left**: Permite determinar a espessura referente à margem esquerda do elemento.



Observe o exemplo a seguir, em que temos a definição das margens de um elemento:

- **style.css**

```
p {
    margin-top: 70px;
    margin-left: 50px;
}
```

- **aula9_1.html**

```
<!doctype HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <title>HTML5 Fundamentos - CSS</title>
        <link type="text/css" href="css/style.css"rel="StyleSheet" />
    </head>
    <body>
        <main role="main">
            <p>
                Testando o atributo
                margin-top.
            </p>
        </main>
    </body>
</html>
```

Ao executarmos esse código, o resultado obtido é o seguinte:

9.4.Posicionando os elementos

Quando trabalhamos com arquivos CSS, podemos posicionar elementos e construir o layout com algumas propriedades. Duas propriedades do CSS3 que serão utilizadas com o objetivo de criar e posicionar um layout são **Flexible Box Layout**, ou simplesmente **Flexbox**, que é uma forma simples e sem cálculos para posicionar um layout, e o **Módulo Grid Template Layout**, ainda sendo padronizado em 2013.

Além disso, temos outras duas propriedades que já eram utilizadas no XHTML: **float** e **position**. Neste tópico, veremos como trabalhar com elas.

9.4.1.Position

A propriedade **position** permite determinar o posicionamento de um conteúdo na tela do usuário. Esse posicionamento pode ser: **estático**, **fixo**, **absoluto** e **relativo**.

9.4.1.1. Posicionamento estático

Este tipo de posicionamento é determinado por meio do valor **static**, o qual é desnecessário, pois o posicionamento estático é padrão para os elementos. Portanto, a propriedade **position**: **static** não precisa ser declarada de forma explícita na folha de estilo.

Um elemento cuja posição é estática coloca-se imediatamente abaixo de seu elemento anterior e acima de seu posterior. No entanto, vale destacar que isso apenas ocorre caso os elementos anterior e posterior não tenham seu posicionamento determinado de forma diferente da estática.

9.4.1.2. Posicionamento fixo

Este tipo de posicionamento determina que o elemento seja posicionado de acordo com a parte que pode ser visualizada do **User Agent**, no qual é exibida a página. Para determinar que um elemento terá seu posicionamento fixo, é preciso utilizar o valor **fixed** juntamente à propriedade **position**.

9.4.1.3. Posicionamento absoluto

Este tipo de posicionamento determina que o elemento seja posicionado levando-se em consideração o local em que está o elemento mais próximo, cuja posição pode ser definida como **fixa**, **absoluta** ou **relativa**.

Nas situações em que não há um elemento mais próximo, considera-se o posicionamento do elemento **<body>.** Para determinar o posicionamento de um elemento como sendo **absoluto**, basta utilizar o valor **absolute** em conjunto à propriedade **position**.

Vale destacar que há quatro propriedades que se aplicam aos elementos cujo posicionamento é determinado como **absoluto**. São elas: **bottom**, **top**, **left** e **right**.

Observe o exemplo a seguir:

- **style.css**

```css
#imagem {
    position: absolute;
    left: 30px;
    top: 60px
}
```

- **aula9_2.html**

```html
<!doctype HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <title>HTML5 Fundamentos - CSS</title>
        <link type="text/css" href="css/style.css"rel="StyleSheet" />
    </head>
    <body>
        <main role="main">
            <img src="images/logotipo.png" alt="Logo HTML5" id="imagem">
        </main>
    </body>
</html>
```

Ao executarmos o código anterior, o resultado obtido é o seguinte:

Podemos observar, no exemplo apresentado, a presença da propriedade **position**, a qual recebe o valor **absolute** e é complementada pelas propriedades **left** e **top**. Com isso, o posicionamento do elemento é determinado tomando-se como base o canto superior esquerdo da viewport. O elemento é representado neste exemplo por **img**, que possui a **id=”imagem”**.

Podemos observar, ainda, que a propriedade **left** determina o distanciamento da imagem a partir da margem esquerda e a propriedade **top** determina o distanciamento a partir da margem superior.

9.4.1.4. Posicionamento relativo

Este tipo de posicionamento determina a posição do elemento de acordo com sua própria posição. Para determinar o posicionamento de um elemento como **relativo**, basta utilizar o valor **relative** juntamente à propriedade **position**.

Quando trabalhamos com o posicionamento relativo de elementos, podemos utilizar as propriedades **bottom**, **top**, **right** e **left**.

Observe o exemplo a seguir:

- **style.css**

```css
.imagem {
    position: relative;
    left: 30px;
    top: 60px
}
```

- **aula9_3.html**

```html
<!doctype HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <title>HTML5 Fundamentos - CSS</title>
        <link type="text/css" href="css/style.css"rel="StyleSheet" />
    </head>
    <body>
        <main role="main">
            <img src="images/logotipo.png" alt="Logo HTML5" class="imagem">
            <img src="images/css3.png" alt="Logo css3" class="imagem">
        </main>
    </body>
</html>
```

Veja o resultado:

9.4.2. Float

Os elementos que são declarados como **float** são convertidos para elementos de bloco de forma automática, podendo ter sua altura e largura declaradas. Mais especificamente, a largura dos elementos **float** deve ser declarada, pois os browsers consideram a não declaração dessa largura um erro.

Quanto ao posicionamento de um elemento **float**, este ocorre imediatamente após o elemento em bloco anterior a ele, embora tais elementos fiquem fora do fluxo. Os valores que podem ser utilizados juntamente ao atributo **float** são: **none**, **left**, **right** e **inherit**, cujo valor é igual ao valor da propriedade referente ao elemento pai.

Veja o exemplo a seguir:

- **style.css**

```css
img {
    float: left;
    width: 15%;
}
p {
    font-size: 20px;
    margin-top: 0px
}
```

- **aula9_4.html**

```html
<!doctype HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <title>HTML5 Fundamentos - CSS</title>
        <link type="text/css" href="css/style.css"rel="StyleSheet" />
    </head>
    <body>
        <main role="main">
            <img src="images/logotipo.png" alt="Logo HTML5" class="imagem">
            <p>Texto principal</p>
        </main>
    </body>
</html>
```

Veja o resultado obtido:

Se não tivéssemos utilizado o atributo **float** no exemplo anterior, o resultado obtido seria o seguinte:

9.5.Definindo as camadas

Alguns elementos de bloco, como **section**, **div**, **article** e outros mencionados anteriormente, oferecem atributos que permitem definir as camadas que serão utilizadas na construção do layout da página.

Veja, a seguir, quais são esses atributos:

- **id**: Permite determinar como a camada será identificada;
- **class**: Permite determinar qual classe já declarada no arquivo CSS deve ser aplicada;
- **z-index**: Permite determinar o nível de empilhamento;
- **visibility**: Permite determinar a visibilidade da camada. Os valores que podem ser utilizados por este atributo são os seguintes: **visible**, **inherit** e **hidden**.

O atributo **z-index**, que tem a finalidade de ordenar a prioridade de visualização dos elementos, funciona apenas com os elementos que foram configurados com o posicionamento **absoluto**, o que significa que são os elementos cuja propriedade **position** é configurada como **absolute**. Vale destacar que quanto maior o valor do atributo **z-index**, maior a prioridade de visualização do elemento. Por exemplo, um elemento que possui **z-index**: 5 tem prioridade sobre um elemento que possui **z-index**: 3.

Observe o exemplo a seguir:

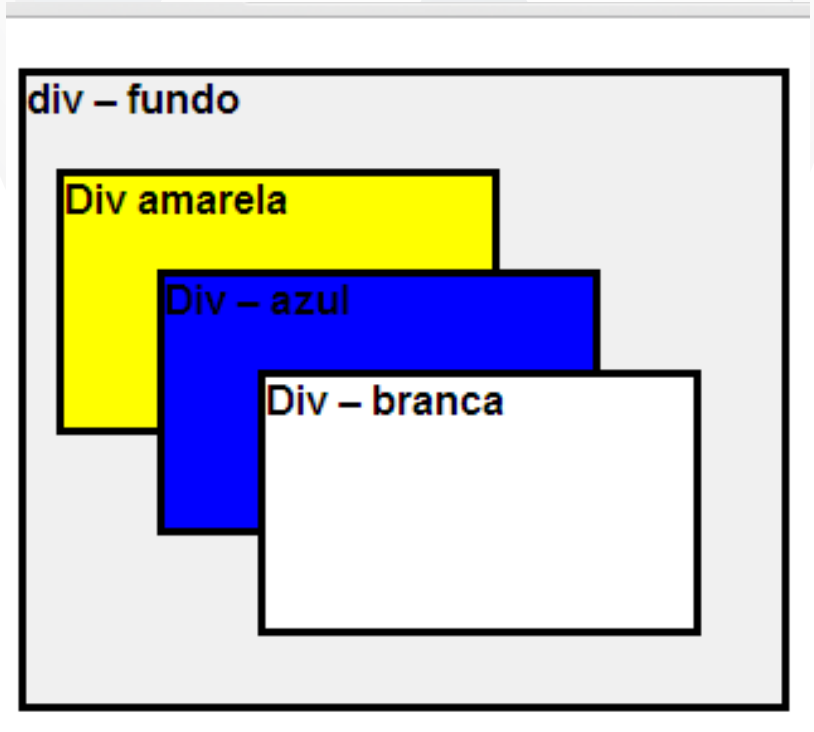
- **style.css**

```css
.posicao{
    position: absolute;
    width:170px; height: 100px;
    font-family: arial;
    font-size: 16px;
    font-weight: bold;
}
#branca{
    top:140px;left:100px;
    background-color: #fff;
    border: solid 3px #000;
    z-index: 4;
}
#azul{
    top:100px;left:60px;
    background-color: #00f;
    border: solid 3px #000;
    z-index: 3;
}
#amarela{
    top:60px;left:20px;
    background-color: #ff0;
    border: solid 3px #000;
    z-index: 2;
}
#fundo{
    top:20px;left:5px;
    width:300px;height: 250px;
    background-color: #eee;
    border: solid 3px #000;
    z-index: 1;
}
```

- **aula9_5.html**

```
<!doctype HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <title>HTML5 Fundamentos - CSS</title>
        <link type="text/css" href="css/style.css"rel="StyleSheet" />
    </head>
    <body>
        <main role="main">
            <div id="amarela">Div amarela</div>
            <div id="azul">Div - azul </div>
            <div id="branca">Div - branca</div>
            <div id="fundo">div - fundo</div>
        </main>
    </body>
</html>
```

O resultado produzido será o seguinte:



Neste exemplo, temos o posicionamento de várias camadas. Com isso, algumas sobreposições foram criadas.

9.6.Definindo o modo de apresentação dos elementos

O atributo **display** permite determinar se um elemento deve ser apresentado e, caso deva, permite determinar seu modo de apresentação. Os valores que podem ser utilizados com o atributo **display** são os seguintes:

Valores	Descrição
table	Determina que o elemento seja apresentado como uma tabela, sendo que há mudança de linha tanto antes do elemento quanto depois dele.
list-item	Determina que o elemento seja apresentado com sendo item de uma lista.
inline	Determina que o elemento seja apresentado sem que haja mudança de linha.
block	Determina que o elemento seja apresentado como um elemento de bloco, e, diferentemente do item anterior, existe a quebra de linha antes e depois do item.
none	Determina que o elemento não seja apresentado.

O exemplo descrito a seguir permite compreender de forma adequada a utilização dos valores que acabam de ser definidos:

- **style.css**

```css
.linha{display: inline;}
```

- **aula9_6.html**

```html
<!doctype HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <title>HTML5 Fundamentos - CSS</title>
        <link type="text/css" href="css/style.css"rel="StyleSheet" />
    </head>
    <body>
        <main role="main">
            <p class="linha"> PARAGRAFO 1 * PARA TESTE DO ATRIBUTO DISPLAY </p>
            <p class="linha"> PARAGRAFO 2* PARA TESTE DO ATRIBUTO DISPLAY </p>
            <p class="linha"> PARAGRAFO 3* PARA TESTE DO ATRIBUTO DISPLAY </p>
        </main>
    </body>
</html>
```

O resultado produzido será o seguinte:

PARAGRAFO 1 * PARA TESTE DO ATRIBUTO DISPLAY PARAGRAFO 2* PARA TESTE DO ATRIBUTO DISPLAY
PARAGRAFO 3* PARA TESTE DO ATRIBUTO DISPLAY

9.7.Tabindex

Por meio desse atributo, podemos atribuir uma ordem de navegação ao longo de todos os elementos existentes em um documento HTML5, como links e elementos de formulário. Sem o uso de **tabindex**, o usuário seria obrigado a percorrer todos esses elementos de acordo com a ordem em que eles aparecem no código.

É possível determinar o índice de tabulação em uma barra de navegação, em elementos pertencentes a um formulário ou outros itens que fazem parte do código XHTML. Para que o índice de tabulação seja definido em um campo de formulário e um link, por exemplo, devemos utilizar a sintaxe a seguir:

```
<input type="email" id="login" tabindex="1" />
```

9.8.Tipos de mídia

Alguns estilos podem ser aplicados somente em algumas situações especiais, por exemplo, a utilização de um determinado tipo de fonte assim que algum usuário requisitar a impressão da página. Por meio dos tipos de mídia (também chamados de media types), um arquivo CSS poderá ser criado apenas para um evento em particular.

Veja quais são os principais tipos de mídia para aplicação em uma folha de estilo CSS:

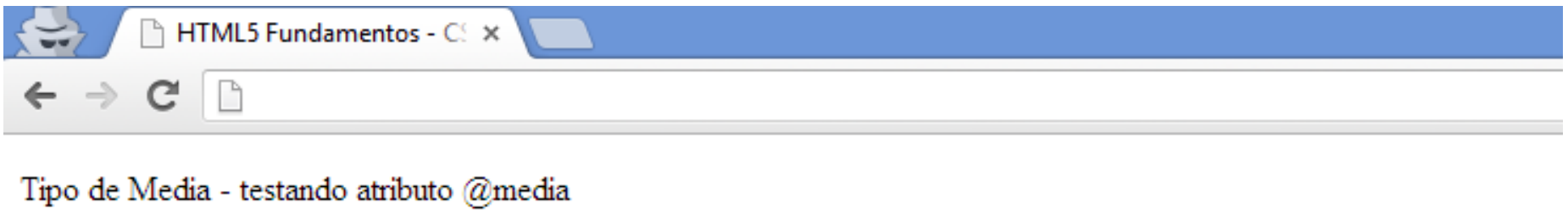
- **all**: Permite a definição de estilos para todos os tipos de mídia;
- **aural**: Os estilos são definidos para sintetizadores de voz;
- **braille**: Possibilita a aplicação de definições para textos em braile;
- **embossed**: Permite a impressão em impressoras próprias para a leitura braile;
- **handheld**: Os estilos são definidos para equipamentos móveis de mão, como celulares;
- **print**: Possibilita a aplicação de estilos para a impressão da página em papel;
- **projection**: Os estilos são definidos para a apresentação da página em um projetor;
- **screen**: Permite que os estilos sejam utilizados para que a página seja exibida na tela do computador;
- **tty**: Os estilos são aplicados para a apresentação da página em terminais com poucos recursos;
- **tv**: Permite a exibição da página em uma tela de TV.

Por meio da regra **@media**, é possível aplicar diversas propriedades relacionadas a diferentes tipos de mídia em uma só folha de estilo. No exemplo a seguir, temos uma demonstração dessa regra: a página será formatada para impressão em papel e exibição na tela. Para começar, será criado o arquivo **.css**:

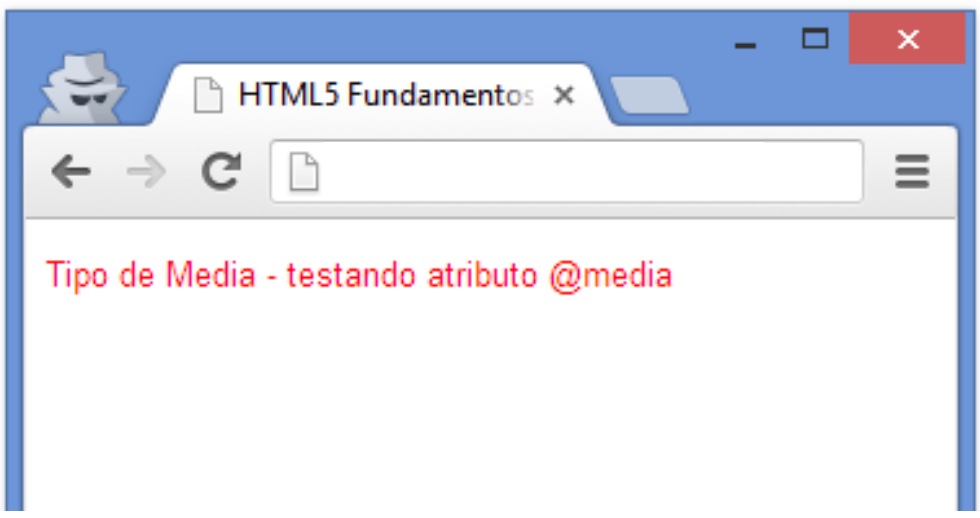
```
@media print {
 .midia {
 font-family: arial;
 font-size: 11px;
 color: black;
 text-align: left
 }
}
@media only screen and (max-width: 480px){
 .midia {
 font-family: arial;
 font-size: 13px;
 color: red;
 text-align: left
 }
}
```

No exemplo anterior, temos um estilo com cor preta para impressora e outro estilo para dispositivos que possuem no máximo 480px de largura.

Para realizar este teste podemos utilizar o redimensionar do navegador. Observe que com a janela completamente dimensionada, ultrapassamos o valor de 480px de largura, logo, o estilo não será aplicado:

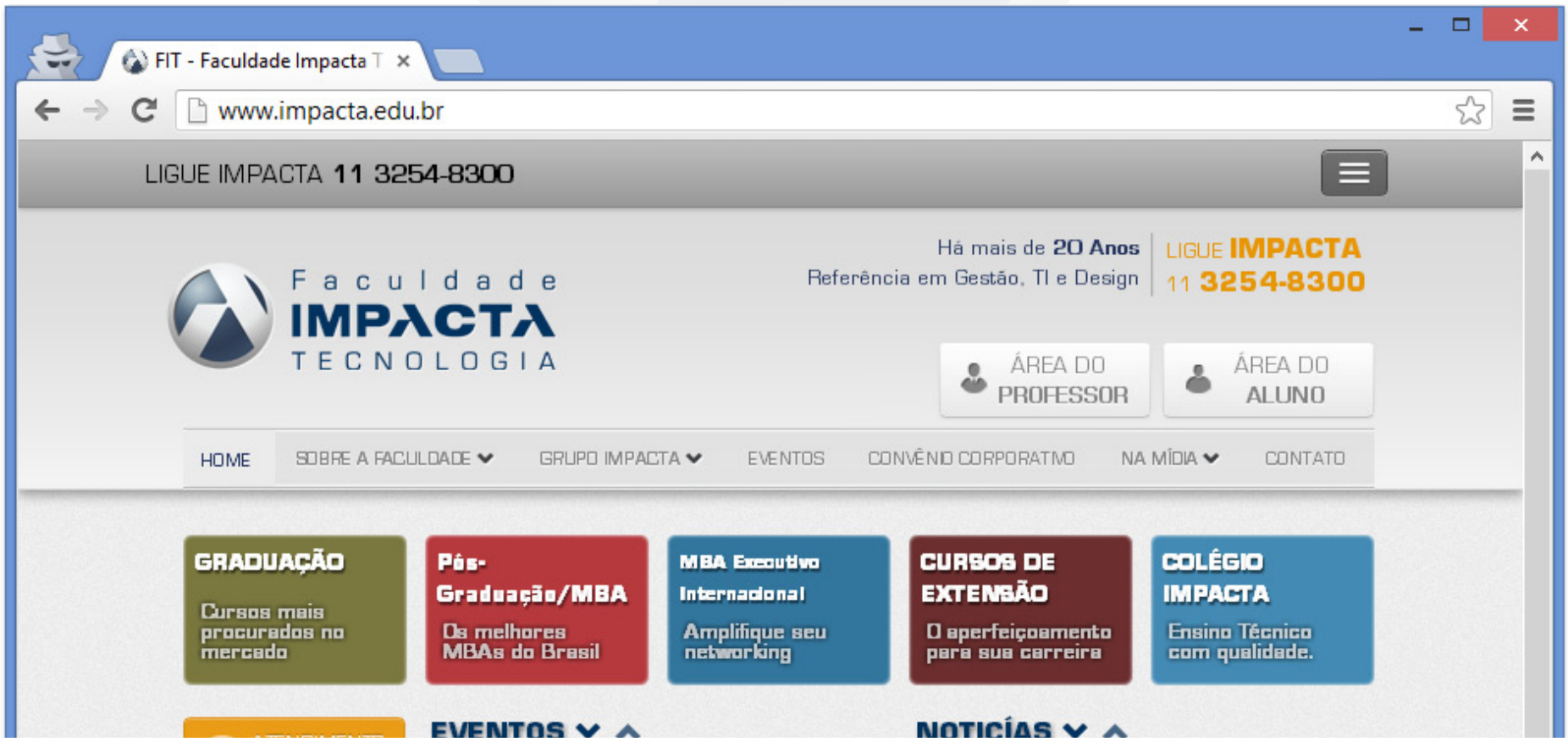


Agora, quando redimencionamos o navegador para uma resolução inferior a 480px o estilo é aplicado:



O layout do site da Faculdade Impacta Tecnologia utiliza esse recurso, que é denominado Responsive Web Design ou apenas Layout Responsivo:

Reduzindo a dimensão da janela, por exemplo, no uso em tablets, percebemos que o banner principal sumirá:



E com a tela utilizada para SmartPhone: