Disciplina de Desenvolvimento Web

PROFESSOR JEFFERSON CHAVES

jefferson.chaves@ifpr.edu.br

ATIVIDADE AVALIATIVA

O objetivo desta atividade é reforçar o uso de floats e media queries.

Float

Float é uma propriedade que controla o posicionamento de um elemento **em relação ao seu container** e aos outros elementos na página. Originalmente, a propriedade float foi criada para possibilitar a inserção de texto ao redor de imagens, mas acabou sendo amplamente utilizada no layout de páginas.

Como funciona o float?

Quando um elemento é flutuado, ele é removido do fluxo normal do documento, ou seja, ele não "empurra" os outros elementos ao seu redor, mas flutua à esquerda ou à direita do seu container, permitindo que o conteúdo ao redor se organize de acordo com esse posicionamento.

Valores de float:

- none (padrão): O elemento segue o fluxo normal do documento.
- left: O elemento flutua à esquerda do container e o conteúdo ao redor se ajusta ao seu lado direito.

- right: O elemento flutua à direita do container e o conteúdo ao redor se ajusta ao seu lado esquerdo.

```
<!DOCTYPE html>
<html>
<head>

<style>
 .imagem {
   float: left;
   margin: 10px;
 }
</style>

</head>
<body>
 <img src="imagem.jpg" class="imagem" alt="Exemplo de imagem flutuante">
 <p>Este é um exemplo de texto que será ajustado ao lado de uma imagem flutuante. O texto contorna a imagem porque ela foi flutuada à esquerda usando o `float`.</p>
</body>
</html>
```

Passo 0: Preparando o projeto

Crie uma pasta chamada float-e-media-queries. Faça o download dos arquivos iniciais do projeto em neste repositório.

Passo 1: Estrutura HTML

Vamos analisar o código HTML.

Tanto os menus quanto ao conteúdo central, estão envolvidos por uma uma div com uma classe chamada de “container”:

```
<div class="container">
    //conteudo
</div>
```

Embora o uso do nome seja uma convenção, usado por 99% dos programadores, essa classe poderia ser chamada de container, contêiner, caixa, wrapper, e etc. O mais importante aqui é o papel que ela tem: deixar a página HTML centralizada.

A razão por trás disso é permitir que mesmo que você **não saiba** quantos pixels possuem uma determinada tela, é possível determinar uma largura conhecida, com o objetivo de acomodar os componentes de uma página.

Adicione o trecho de código ao arquivo style.css:

```
.container  {
   width: 1200px;
   margin: 0 auto;
   padding: 15px;
}
```

Neste ponto estamos determinando uma largura conhecida de 1200px para nosso container, nos permitindo o cálculo mais adequado dos componentes do layout. O uso de `margin: 0 auto;` define uma margem de `0px` para cima e para baixo do container e uma margem automaticamente calculada, o que faz com que o espaço disponível seja igualmente distribuída entre as margens esquerda e direita, deixando o container ao centro. Já o uso de `padding: 15px;` indica a margem interna do container, fazendo com que os componentes internos não fiquem “grudados” nas bordas do container.

O uso de container é tão comum que podemos criar contêineres para outras coisas. O objetivo é o mesmo: organizar um conjunto de componentes;

O leiaute proposto apresenta um contêiner para cada produto apresentado.

```
<div class="caixa-container">
    <div class="caixa">
        <h1>Produto 1</h1>
    </div>
</div>
```

O comportamento padrão de uma caixa (<header>, <aside>, <main>, <footer>, e etc) é ocupar toda a largura de seu container. Vamos adicionar estilos para modificar o comportamento componente:

```
.caixa-container {
   width: 25%;
}
```

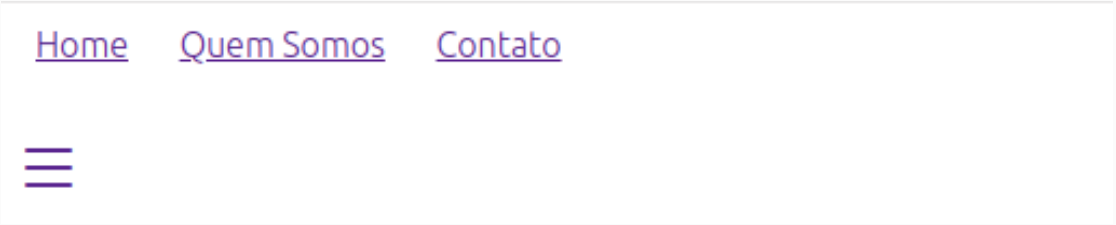
Mesmo havendo espaço disponível, o comportamento de uma caixa ainda faz com que ele ocupe 100% do tamanho do contêiner.

Vamos adicionar um novo comportamento a essa : float! O float faz com que os componentes “flutuem” e se encaixem à direita ou à esquerda. Adicione mais esse estilo a declaração anterior:

```
.caixa-container {
    width: 25%;
    float: left
}
```

Passo 2: Ajustando o menu

O Leiaute ainda está com um problema no menu.



Como ajuste inicial, vamos ocultar o menu “sanduíche” da visualização, adicionando a propriedade `display: none;`. Esse menu deverá aparecer somente em resoluções menores, tais como celulares ou tablets.

```
.menu-responsivo {
    …
    display: none;
}
```

Passo 3: Ajustando a responsividade.

O objetivo agora é fazer com que o leiaute apresentado se ajuste a diferentes resoluções. Sua tarefa é:

a) Para resoluções de até 992px:
   i) O body deve apresentar a cor ametist-color;
   ii) O classe .container deve apresentar largura de 768px;
   iii) E a classe .caixa-container, largura de 33.33%.

b) Para resoluções de até 768px:
   i) O body deve apresentar a cor sunflower-color;
   ii) O classe .container deve apresentar largura de 576px;
   iii) E a classe .caixa-container, largura de 50%.

c) Para resoluções de até 576px:
   i) O body deve apresentar a cor carrot-color;
   ii) O classe .container deve apresentar largura de 100%;
   iii) A classe .caixa-container, largura de 100%;
   iv) O menu deve ser ocultado e o menu sanduíche deve ser exibido;