02

Implementando o Garconapp

Transcrição

O primeiro passo será ir no ***site do Materialize*** (http://materializecss.com/) e podemos baixar padrões de estilo e animação escritos em CSS e JavaScript. Além disso, precisaremos fazer o download do jQuery e de fontes que trazem os ícones do Material Design, no site do Google.

Disponibilizei no meu ***Github*** (https://github.com/sergiolopes/garconapp), a estrutura básica com o Materialize, as fontes, o jQuery. Ele irá baixar um arquivo .zip, que irá facilitar o processo. Dentro dele, teremos a pasta `www`, que contem o `css` do Materialize, fonte, ícone, JS.

Em seguida, iremos para o projeto `garconapp`, que possui a pasta `www`, com todo o conteúdo padrão criado pelo Cordova e não está relacionado com nosso aplicativo. Nós vamos substituir esse material pelo conteúdo que veio no arquivo .zip baixado no Github.

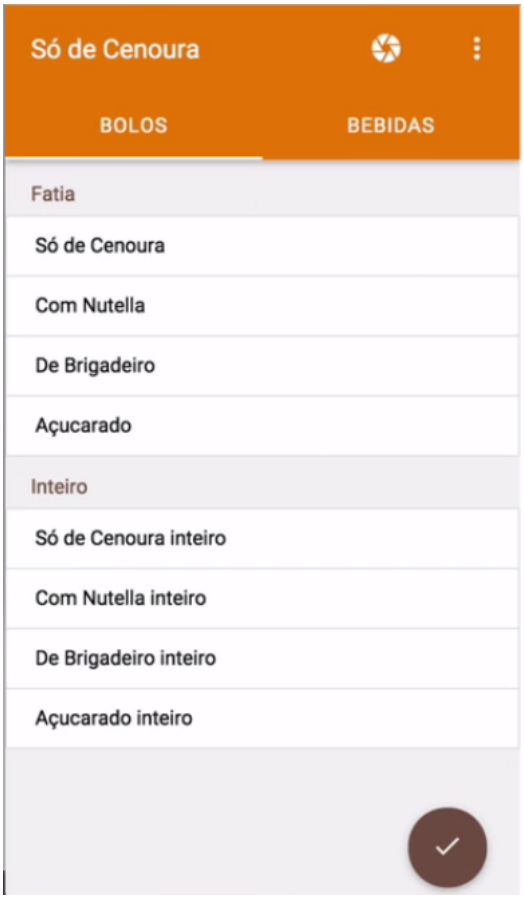
O nosso material traz um `index.html` bem simples, mas você também poderá criá-lo desde o começo. Observe que o nosso arquivo tem um `viewport`, título, os ícones do Material Digital, o `css` do Materialize, além do JS do jQuery e o JS do Materialize.

```
<!DOCTYPE html
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta charset="utf-8">
    <title> Só de Cenoura Garçom</title>
    <link rel="stylesheet" href="icons/material.css"
    <link rel="stylesheet" href="css/materialize.min.css">
</head>
<body>

    <script src="js/jquery.min.js"></script>
    <script src="js/materialize.min.js"></script>
</body>
</html>
```

Se abrirmos o código no navegador, ele ficará em branco, porque ele ainda não faz nada. Vamos deixar o modo de visualização como de um dispositivo, como já utilizamos anteriormente.

Nós iremos implementar o aplicativo que terá o resultado final assim:



O app terá um título do topo ("Só de Cenoura"), com duas abas que podemos alternar ("Bolos" e "Bebidas") e uma lista de opções para cada categoria.

Vamos implementar primeiro as duas abas, com o *Materialize*. Para isto, precisarei de uma lista com a *tag* `ul` com as abas. Em seguida, adicionaremos a classe `tabs` - que veio do Materialize.

```
<body>
    <ul class="tabs">

    </ul>

    <script src="js/jquery.min.js"></script>
    <script src="js/materialize.min.js"></script>
</body>
```

Depois, incluiremos *tag*s `li` e cada uma delas irá conter uma classe `tab` . Ela terá um link que irá apontar para o conteúdo da `tab` . Por exemplo, teremos as abas "Bolos" e "Bebidas". Iremos especificar isto no código.

```
<body>
    <ul class="tabs">
    <li class="tab"><a href="">Bolos</a></li>
    <li class="tab"><a href=""> Bebidas</a></li>
    </ul>

    <script src="js/jquery.min.js"></script>
    <script src="js/materialize.min.js"></script>

</body>
```

Com este trecho de código, se visualizarmos no navegador, veremos que já aparecem as abas. Ainda não estão da cor que queremos, porque estão com o design padrão do Materialize, mas resolveremos isto mais adiante.

Como iremos implementar o conteúdo de cada aba? Usaremos a *tag* `div`, que terá a classe `section`. Nós podemos incluir qualquer conteúdo dentro, como por exemplo, "meus bolos". Criaremos um `div` diferente para minhas "minhas bebidas".

```
<div class="section">
    meus bolos
</div>

<div class="section">
    meus bebidas
</div>
```

Em seguida, queremos associar cada `div` com as abas criadas acima. Faremos isso com HTML, através de âncora e `ids`.

```
<div class="section" id="bolos">
    meus bolos
</div>

<div class="section" id="bebidas">
    meus bebidas
</div>
```

E no `href`, que estava vazio, iremos apontar o `id` "bolos" e "bebidas".

```
<ul class="tabs">
    <li class="tab"><a href="#bolos">Bolos</a></li>
    <li class="tab"><a href="#bebidas"> Bebidas</li>
</ul>
```

Assim indicamos que cada uma das abas está associado com as *tag*s `div`.

Quando fizermos um novo teste no navegador, ao selecionarmos a aba "Bolos", aparecerá "meus bolos" e ao selecionarmos a aba "Bebidas", aparecerá "minhas bebidas".

Agora, iremos implementar a lista do app. Para isto, adicionaremos um `div` que irá receber a classe `collection` - referente lista no *Materialize*. Depois, iremos incluir alguns links, porque queremos que os itens sejam clicáveis, adicionado a classes especiais `collection-item`.

```
<div class="section" id="bolos">
    <div class="collection">
        <a class="collection-item">Só de Cenoura</a>
        <a class="collection-item">Com Nutella</a>
        <a class="collection-item">De Brigadeiro</a>
        <a class="collection-item">Açucarado</a>
```

```
    </div>
</div>
```

No navegador, a aba "Bolos" já terá as opções clicáveis, mas com o design padrão.

Faremos algo parecido com a aba "Bebidas", apenas incluindo os itens do cardápio de bebidas.

```
<div class="section" id="bebidas">
    <div class="collection">
        <a class="collection-item">Espresso</a>
        <a class="collection-item">Capuccino</a>
        <a class="collection-item">Mocaccino</a>
    </div>
</div>
```

No navegador, os itens da aba "Bebidas" serão clicáveis também.

Para incluir outras bebidas, vamos adicionar separadores do *Materialize* para termos categorias. Podemos fazer isto, com a classe `container` . Criaremos duas: "Cafés" e " Refrigerantes".

```
<div class="section" id="bebidas">
    <h6 class="container">Cafés</h6>
        <div class="collection">
        <a class="collection-item">Espresso</a>
        <a class="collection-item">Capuccino</a>
        <a class="collection-item">Mocaccino</a>
    </div>

    <h6 class="container">Refrigerantes</h6>
        <div class="collection">
        <a class="collection-item">Coca</a>
        <a class="collection-item">Soda</a>
        <a class="collection-item">Guaraná</a>
    </div>
</div>
```

Agora, na aba "Bebidas" teremos duas categorias diferentes. É possível incluir categorias na aba "Bolos". Vamos adicionar as categorias "Em pedaços" e "Inteiro".

```
<div class="section" id="bolos">
    <h6 class="container">Em pedaços</h6>
    <div class="collection">
        <a class="collection-item">Só de Cenoura</a>
        <a class="collection-item">Com Nutella</a>
        <a class="collection-item">De Brigadeiro</a>
        <a class="collection-item">Açucarado</a>
    </div>

    <h6 class="container">Inteiro</h6>
    <div class="collection">
        <a class="collection-item">Só de Cenoura inteiro</a>
        <a class="collection-item">Com Nutella inteiro</a>
```

```
        <a class="collection-item">De Brigadeiro inteiro</a>
        <a class="collection-item">Açucarado inteiro</a>
    </div>
</div>
```

Após adicionar os separadores na aba "Bolos", iremos adicionar efeitos nos botões. Vimos que o *Materialize* disponibiliza um efeito de onda (`waves`), bastante usado em outros aplicativos. Como podemos adicioná-lo ao nosso app?

De volta ao editor de texto, na aba HTML, vamos adicionar a classe `waves-effect` em todos os `collection-item` .

```
<div class="section" id="bolos">
    <h6 class="container">Em pedaços</h6>
    <div class="collection">
        <a class="collection-item waves-effect">Só de Cenoura</a>
        <a class="collection-item waves-effect">Com Nutella</a>
        <a class="collection-item waves-effect">De Brigadeiro</a>
        <a class="collection-item waves-effect">Açucarado</a>
    </div>

    <h6 class="container">Inteiro</h6>
    <div class="collection">
        <a class="collection-item waves-effect">Só de Cenoura inteiro</a>
        <a class="collection-item waves-effect">Com Nutella inteiro</a>
        <a class="collection-item waves-effect">De Brigadeiro inteiro</a>
        <a class="collection-item waves-effect">Açucarado inteiro</a>
    </div>
</div>

/...
```

Se testarmos os botões de todos os itens, nas abas "Bolos" e "Bebidas", terão o efeito de ondas de cor, quando clicado.

Podemos também mudar a cor do texto. No design padrão, o texto está escrito em verde. Iremos mudá-lo para preto. Algumas classes no *Materialize* que têm cores, no nosso caso, usaremos a `black-text` . Você também pode criar o seu arquivo CSS e adicionar a cor preta. Nós utilizaremos o recurso do *Materialize*. Para os títulos, usaremos a cor marrom (`brown-text`) . Veremos como ficará o trecho referente à bebidas

```
<div class="section" id="bebidas">
    <h6 class="container brown-text">Cafés</h6>
        <div class="collection">
        <a class="collection-item waves-effect black-text">Espresso</a>
        <a class="collection-item waves-effect black-text">Capuccino</a>
        <a class="collection-item waves-effect black-text">Mocaccino</a>
    </div>

    <h6 class="container brown-text">Refrigerantes</h6>
        <div class="collection">
        <a class="collection-item waves-effect black-text">Coca</a>
        <a class="collection-item waves-effect black-text">Soda</a>
        <a class="collection-item waves-effect black-text">Guaraná</a>
    </div>
</div>
```

Podemos utilizar diversas cores. Se acessarmos o site do Materialize (http://materializecss.com/color.html), encontraremos uma paleta de cores, com as classes que precisaremos usar.

Vamos alterar as cores das abas e deixá-las laranjas. Nós iremos até a parte das `tabs` no código e adicionaremos a classe `yellow darken-4` . Para aumentar o contraste do texto, vamos alterá-lo para branco com uma classe nova: `white-text` . Também incluiremos os efeitos de ondas nas duas abas, mas em vez de escuras, serão claras com `waves-light` .

```
<ul class="tabs yellow darken-4">
    <li class="tab"><a href="#bolos" class="white-text waves-effect waves-light">Bolos</a></li>
    <li class="tab"><a href="#bebidas" class="white-text waves-effect waves-light">Bebidas</a><,
</ul>
```

Existe um detalhe que iremos alterar, um traço que aparece abaixo da aba selecionada e ainda tem a cor anterior. Queremos que ele fique branco. O Materialize não tem uma classe fácil para isso, mas podemos mudá-lo criando um arquivo novo chamado `estilo.css` . Iremos importá-lo para o arquivo HMTL.

```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1"
    <meta charset="utf-8">
    <title>Só de Cenoura Garçom</title>
    <link rel="stylesheet" href="icons/material.css">
    <link rel="stylesheet" href="css/materialize.min.css">
    <link rel="stylesheet" type="text/css" href="css/estilos.css">
    </head>
</html>
```

Para mudar a cor do traço da aba, teremos que adicionar o seletor `.tabs .indicator` com o `background-color:white` .

```
.tabs .indicator {
    background-color: white;
}
```

Se testarmos no navegador, veremos o traço abaixo da aba selecionada, branco.

Ainda no arquivo CSS, vamos mudar o fundo do corpo (`body`) do app. Vamos usar uma cor cinza claro (`#f2f2f2`).

```
.tabs .indicator {
    background-color: white;
}
body {
    background-color: #f2f2f2
}
```

Agora, o fundo das categorias contrasta com o branco do resto da lista.

Para terminarmos o design final, faltou o título do app "Só de Cenoura". Vamos implementá-lo acima das abas. No arquivo HTML, vamos acrescentar um `div` no topo do `body` .

```
<body>

<div>
    <h5>Só de Cenoura</h5>

</div>

</body>
```

Vamos customizar as cores com o uso de classes do *Materialize*. Vamos acrescentar uma classe `titulo`, escrita em português, porque não faz nada. Queremos que o fundo tenha o mesmo laranja das abas (`yellow darken-4`), o texto seja branco (`white-text`) e que ele esteja alinhado verticalmente `valign-wrapper`. Vamos acrescentar as classes adiante do `div`.

```
    <div class="yellow darken-4 white-text valign-wrapper">
    <h5 class="titulo">Só de Cenoura</h5>
    </div>
```

Quando visualizarmos, o aplicativo terão título, mas o tamanho da fonte está desproporcional. Faremos os ajustes adiante.

Em seguida, iremos adicionar alguns ícones ao lado do título. Vamos criar `div` abaixo do título. Dentro dele, irão os ícones. Para isto, adicionaremos `i` com a classe `material-icons`, que veio com o arquivo baixado do Material Design do Google. Ele utiliza *icon fonts*, com vários ícones possíveis para trabalhar. Vamos incluir um efeito no botão `waves-effect waves-light waves-circle`. Depois, incluiremos ícone `more_vert`.

```
 <div class="yellow darken-4 white-text valign-wrapper">
   <h5 class="titulo">Só de Cenoura</h5>
   <div>
   <i class="material-icons waves-effect waves-light waves-circle">more_vert</i>
   </div>
</div>
```

Temos a opção de outros ícones, no [site do Material Design (https://design.google.com/icons/#ic_invert_colors_off)](https://design.google.com/icons/#ic_invert_colors_off) encontramos algumas delas. Temos que pesquisar e ver qual é o texto referente ao ícone para ser acrescentado no código. O ícone `more-vert` é o que contem três pontinhos alinhados verticalmente. No aplicativo, ao clicarmos, o botão terá o efeito wave circular.

Para ajustar o título e o novo ícone, iremos para o arquivo CSS e usaremos a classe `titulo` que criamos anteriormente. Vamos utilizar o tamanho de fonte `1.3rem`, e faremos ajustes nas margens da esquerda e da direita.

```
.titulo {
    font-size: 1.3rem;
    margin-left: 5%;
    margin-right: auto;
}
```

Estas medidas podem ser ajustadas da forma como você achar mais apropriada.

Ao testamos o app, o tamanho da fonte do título está menor e o ícone `more-vert` está posicionado à direita da barra.

Queremos agora que a barra com o título fique fixa quando fizermos um *scroll* na tela. Na aba HTML, iremos incluir um `div` que irá incluir o corpo inteiro - das linhas do título até as listas de abas. Depois, iremos incluir uma classe `topo-fixo`

```
<div>

<div class="topo-fixo">
    <div class="yellow darken-4 white-text valign-wrapper">
        <h5 class="titulo">Só de Cenoura</h5>

        <div>
        <i class="material-icons waves-effect waves-light waves-circle">more_vert</i>
        </div>
    </div>

    <ul class="tabs yellow darken-4">
    <li class="tab"><a href="#bolos" class="white-text waves-effect waves-ligth">Bolos</a></li>
    <li class="tab"><a href="#bebidas" class="white-text waves-effect waves-light">Bebidas</a></
    </ul>
</div>
```

No arquivo CSS, usaremos a classe `topo-fixo` e vamos configurar o `position: fixed`.

```
.topo-fixo {
    position: fixed;
    top: 0;
    width: 100%;

}
```

Para fazer com que o topo fique fixo acima da lista de itens, adicionaremos um `z-index`.

```
.topo-fixo {
    position: fixed;
    top: 0;
    width: 100%;
    z-index: 2;
}
```

Porém, o topo ficará sobreposto aos primeiros itens da lista. No CSS, vamos dar um `padding` em cima das seções (`section`) de "Bolos" e "Bebibas" e o tamanho do topo será considerado.

```
.section {
    padding-top: 125px;
}
```

Assim, o topo não irá mais tampar as categorias e podemos fazer o *scroll* sem problemas.

Iremos acrescentar um último detalhe. No topo da *tag* `body` do HTML, vamos adicionar a classe `z-depth-1` do *Materialize*.

```
<body>

    <div class="topo-fixo z-depth-1">
        <div class="yellow darken-4 white-text valign-wrapper">
        <h5 class="titulo">Só de Cenoura</h5>

        <i class="material-icons waves-effect waves-light waves-circle">more_vert</i>
        </div>
    </div>

    /...
</body>
```

Com isto, o topo irá ganhar uma impressão de profundidade com o sombreamento (como se fossem folhas sobrepostas) e ficará visualmente separado da lista.

Nós já criamos o design do app. Em seguida, iremos implementar as funcionalidades do aplicativo.