04

Desenvolvimento da tela de acessórios

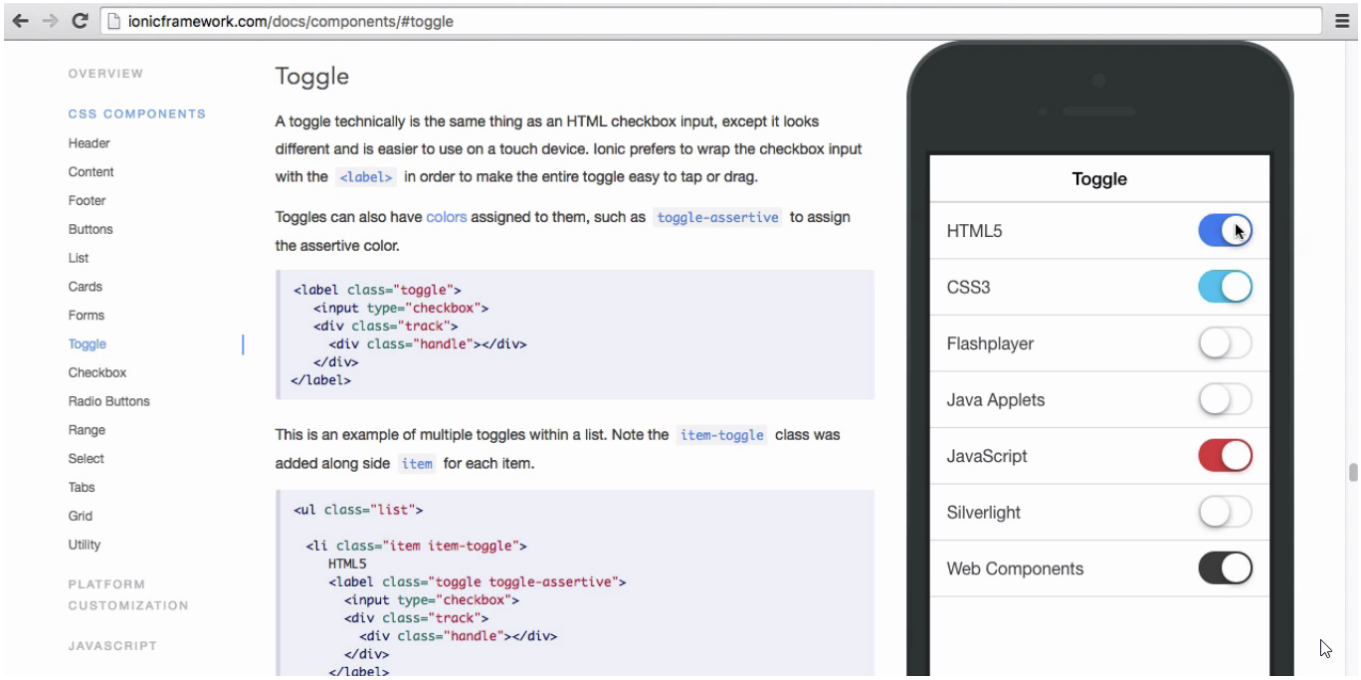
Desenvolvimento da tela de acessórios

Vamos continuar com o desenvolvimento da tela de acessórios. O nosso cliente especificou que cada carro pode ter até três acessórios: freios ABS, ar-condicionado e MP3 Player. Eles serão selecionados pelo usuário. Temos espaço na tela para adicionar os componentes, mas como isto será feito? Vamos voltar ao template do projeto.

No arquivo `carroescolhido.html` , usaremos a tag `<ul>` para construir uma lista dentro do "miolo". Após consultarmos a documentação

```
<div class="item item-divider">
    Ótima Escolha
</div>
<div class="item item-text-wrap">
  {{carroEscolhido}}
  <ul>
  </ul>
</div>
<div class="item item-divider">
    R$50.000
</div>
```

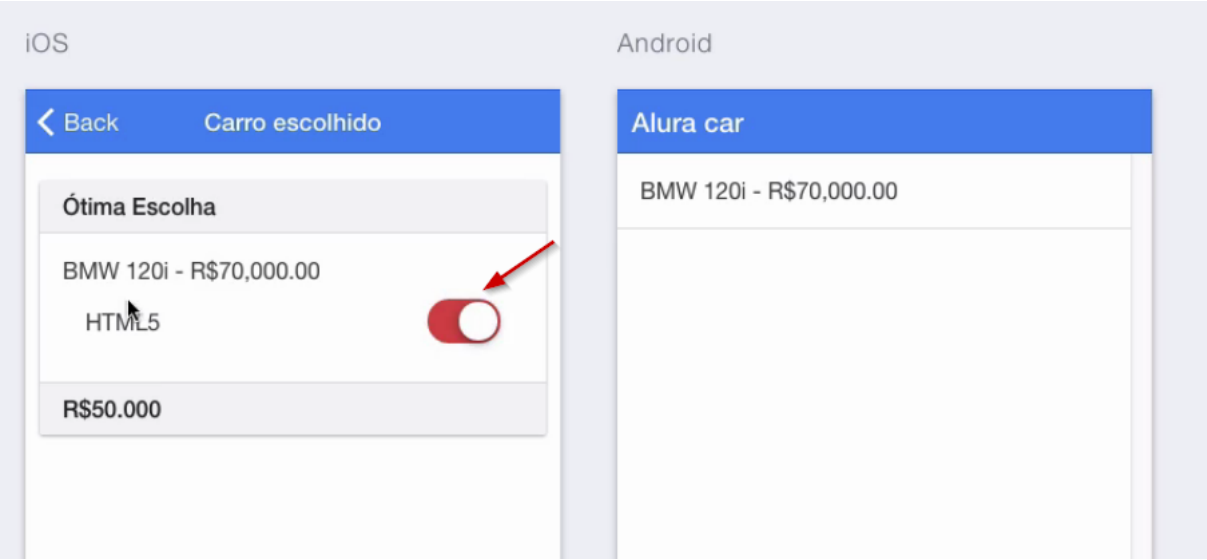
Depois, consultaremos a documentação e veremos como adicionar um botão.



Selecionaremos no Menu de componentes o "Toggle", que funcionará como um chave de alternância. Queremos adicioná-lo na parte de acessórios, para escolher quais itens serão selecionados. Na documentação, vemos o exemplo de como adicioná-lo dentro de uma lista. Nela, veremos que além da tag `<ul>` , usaremos o `class = "lista"` . Também precisaremos adicionar a `<li>` . Já entendemos que não podemos copiar o código html da documentação, mas que precisamos saber o que fazer. Nós criaremos um componente do `carroescolhido.html` , dentro da `ul` que adicionamos anteriormente. .

```
<ul class="list">
    <li class="item item-toggle">
        HTML5
        <label class="toggle toggle-assertive">
          <input type="checkbox">
          <div class="track">
            <div class="handle"></div>
          </div>
        </label>
      </li>
</ul>
```

Agora, vamos testar no navegador.



Podemos ver que o botão já foi adicionado e está funcionando. Mas ainda precisaremos trocar o texto. Não faz sentido deixar a lista de acessórios fixa como um *string* no código, a melhor opção é criar uma lista de objetos. No arquivo `controllers.js`, temos uma `controller` referente ao `CarroEscolhidoController`, criaremos nossa lista aqui. Primeiramente, chamaremos o escopo do Angular e criaremos o `listaDeAcessorios`. Em seguida, criaremos um *array* de objetos.

```
$scope.listaDeAcessorios = [{"nome" : "Freio ABS", "preco" : 800}];
```

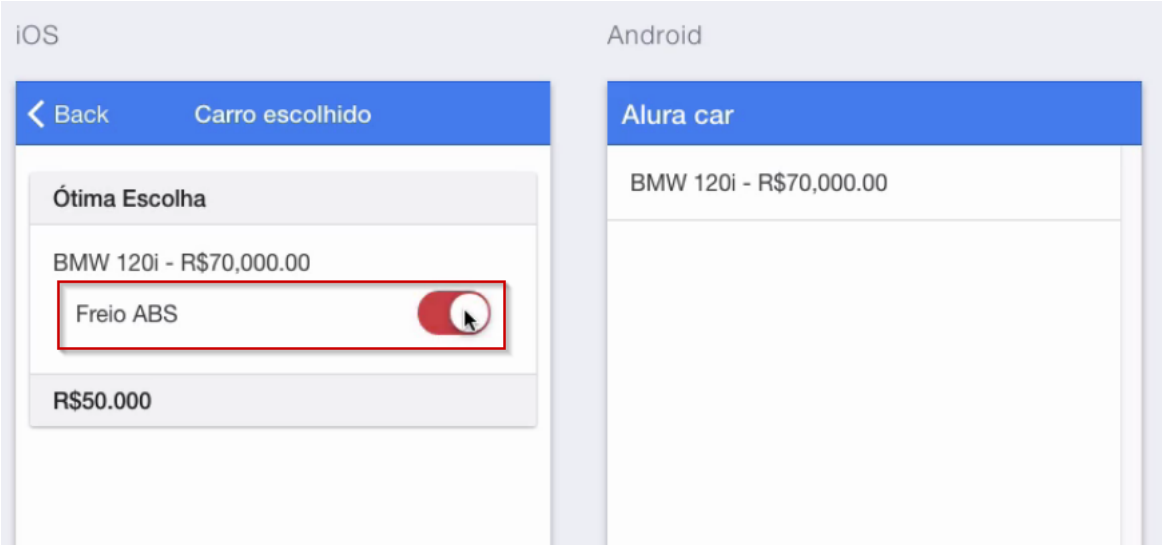
Observe que adicionamos como segundo argumento o `preco`. O trecho do código ficou assim:

```
angular.module('starter')
.controller('CarroEscolhidoController', function($stateParams, $scope){

    $scope.carroEscolhido = angular.fromJson($stateParams.carro);

    $scope.listaDeAcessorios = [{"nome" : "Freio ABS", "preco" : 800}];

});
```

Mas se testarmos na aplicação, ainda não veremos nenhuma diferença. Nós adicionamos no escopo do Angular, mas não usamos a variável `listaDeAcessorios` dentro da *View*. Precisaremos percorrer a lista dentro do `carroescolhido.html`. Para isto, adicionaremos o `ng-repeat` dentro da tag `<li>`.

```
<ul class="list">
    <li class="item item-toggle" ng-repeat="acessorio in listaDeAcessorios">
        {{acessorio.nome}}
        <label class="toggle toggle-assertive">

          <input type="checkbox">
          <div class="track">
            <div class="handle"></div>
          </div>
        </label>
      </li>
</ul>
```

Para cada elemento da nossa lista, nós criamos o `acessorio in` e em vez de usarmos `HTML5`, usamos o escopo do objeto `{{acessorio.nome}}`. Veremos se agora, o item aparece na lista de acessório.

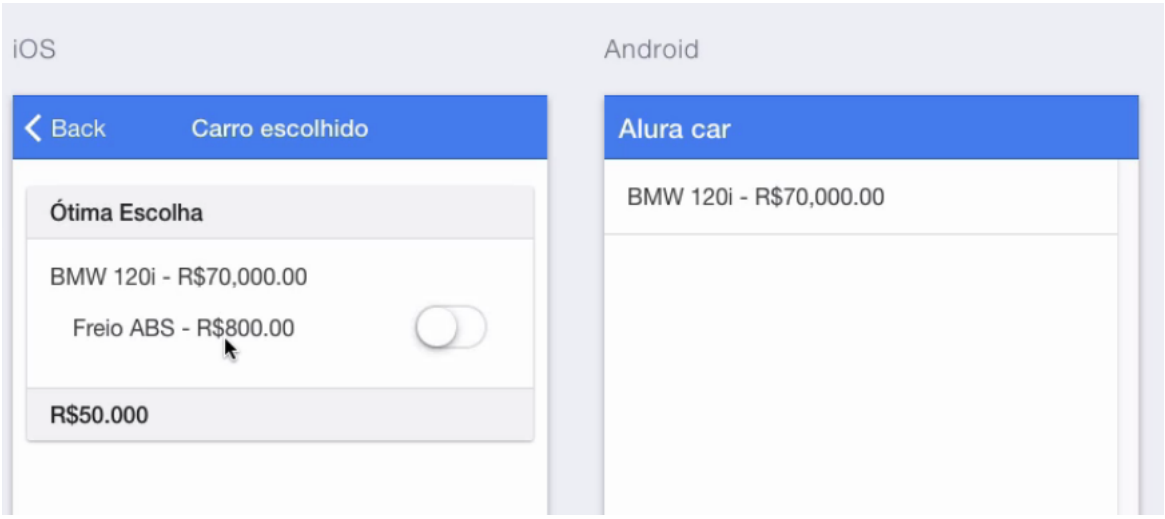


Agora, adicionaremos o objeto e atributo do preço, seguido pelo filtro do `currency`.

```
{{acessorio.nome}} - R{{acessorio.preco | currency }}
```

Com as alterações, o código ficou assim:

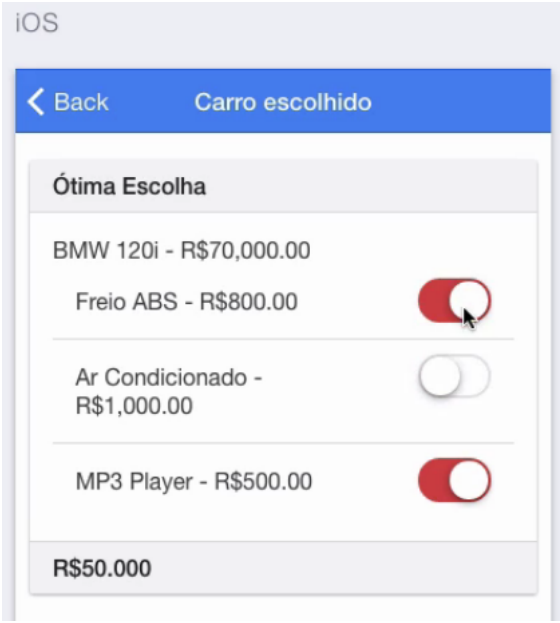
```
<ul class="list">
    <li class="item item-toggle" ng-repeat="acessorio in listaDeAcessorios">
        {{acessorio.nome}} - R{{acessorio.preco | currency }}
        <label class="toggle toggle-assertive">
          <input type="checkbox">
          <div class="track">
            <div class="handle"></div>
          </div>
        </label>
      </li>
</ul>
```



O preço do acessório foi adicionando. Em seguida, iremos incluir os demais objetos da nossa lista, no `controllers.js` .

```
angular.module('starter')
.controller('CarroEscolhidoController', function($stateParams.carro){

  $scope.carroEscolhido = angular.fromJson($stateParams.carro);

  $scope.listaDeAcessorios = [{"nome" : "Freio ABS", "preco" : 800},
                              {"nome" : "Ar-condicionado", "preco" : 1000},
                              {"nome" : "MP3 Player", "preco" : 500}];
});
```

Podemos visualizar os preços dos três itens:



Agora, precisaremos encontrar uma forma de adicionar o preço dos acessórios selecionados ao valor do carro. Por enquanto, o preço do veículo está estático. Nós usaremos o Angular para nos auxiliar nesta parte. Ao selecionarmos um item da lista o valor deve receber um acréscimo e diminuir, caso seja desmarcado. Para isto, faremos alterações no acessório. Dentro da tag `<input>` do arquivo `carroescolhido.html` , adicionaremos o `ng-model` que indicará se o item foi marcado ou não.

```
<input type="checkbox" ng-model='isMarcado' >
```

Mas com estas alterações, o valor do carro não será modificado ainda. Para solucionarmos isto, usaremos o Angular. Nós já temos o valor do objeto, sabemos quando ele é selecionado, agora, criaremos uma função para que o valor seja modificado ao ser clicado. A função será criada dentro da `controller` e receberá o nome de `mudou`. Ela receberá uma `function` e dentro dos parênteses colocaremos o `acessorio` e a variável `isMarcado`. Em seguida, criaremos a lógica: usaremos um `if`.

```
$scope.mudou = function(acessorio, isMarcado) {

  if (isMarcado) {
      $scope.carroEscolhido.preco =
                  $scope.carroEscolhido.preco + acessorio.preco;
  }
}
```

Quando o `carroEscolhido` for marcado, somaremos o `carro.Escolhido.preco` com o `acessorio.preco`.

Mas, no caso em que o item seja desmarcado cairemos no `else` e iremos subtrair o valor do acessório do preço atual visualizado na tela.

```
$scope.mudou = function(acessorio, isMarcado) {

  if (isMarcado) {
      $scope.carroEscolhido.preco =
                  $scope.carroEscolhido.preco + acessorio.preco;
  } else {
      $scope.carroEscolhido.preco =
                  $scope.carroEscolhido.preco - acessorio.preco;
  }
}
```

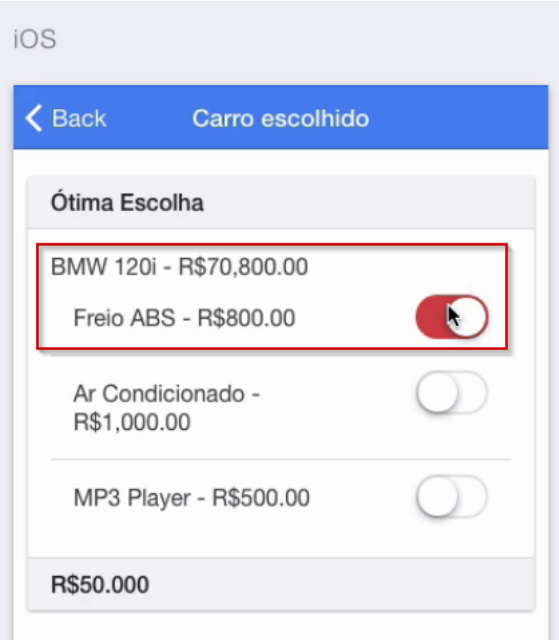
No aplicativo veremos que o valor do carro ainda não será alterado quando selecionarmos um dos acessórios. Criamos a função, mas ela ainda não foi chamada. Faremos isto dentro do botão no HTML `carroEscolhido.html`. Com o `ng-click` chamaremos a função `mudou` e passaremos como parâmetro o `acessorio` e o `isMarcado`.

```
<input type="checkbox" ng-model='isMarcado' ng-click="mudou(acessorio, isMarcado)" >
```

O trecho do código ficará assim:

```
<ul class="list">
    <li class="item item-toggle" ng-repeat="acessorio in listaDeAcessorios">
        {{acessorio.nome}} - R{{acessorio.preco | currency }}
        <label class="toggle toggle-assertive">
          <input type="checkbox" ng-model='isMarcado' ng-click="mudou(acessorio, isMarcado)" >
          <div class="track">
            <div class="handle"></div>
          </div>
        </label>
      </li>
</ul>
```

Agora, os valores já poderão ser alterados.



Conseguimos somar ou subtrair os valores dos acessórios do preço final do carro. Agora, queremos que o valor total fique no rodapé da lista e substitua o valor fixo `R$50.000` .

Atualmente, o trecho do código referente ao rodapé está assim:

```
<div class="item item-divider">
  R$50.000
</div>
```

Vamos substituir o valor fixo por `R{{acessorio.preco | currency }}` .

```
<div class="item item-divider">
  R{{carroEscolhido.preco | currency }}
</div>
```

Em seguida, iremos retirar o `R{{carroEscolhido.preco | currency }}` da parte de cima do código, que ficará assim:

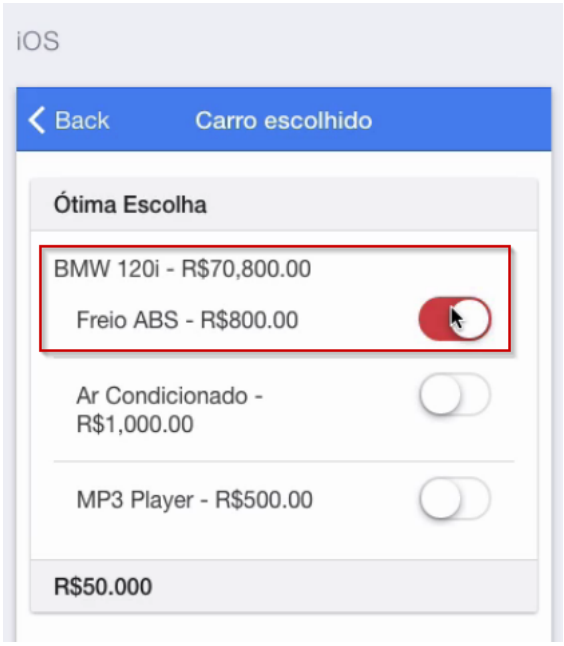
```
<div class="item item-text-wrap">
      {{carroEscolhido.nome}}

//...
```

O arquivo ficou assim:

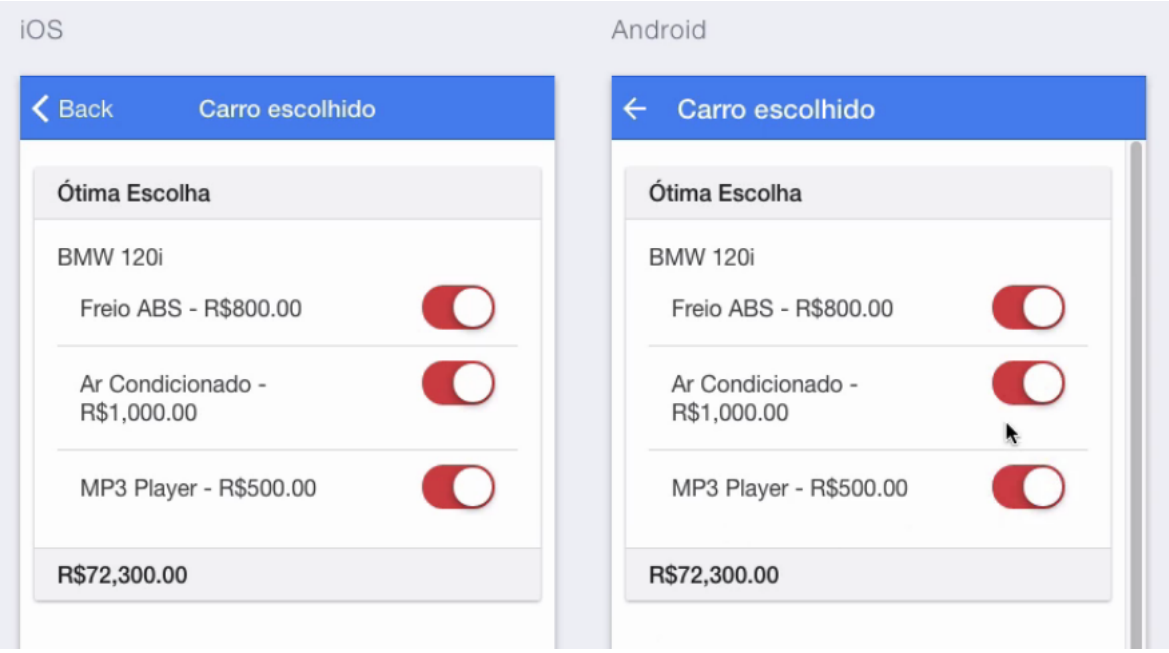
```
<ion-view>
    <ion-nav-title>Carro escolhido</ion-nav-title>
    <ion-content>
    <div class="card">
        <div class="item item-divider">
            Ótima Escolha
        </div>
        <div class="item item-text-wrap">
              {{carroEscolhido.nome}}
```

```
            <ul class="list">
                <li class="item item-toggle" ng-repeat="acessorio in listaDeAcessorios">
                    {{acessorio.nome}} - R{{acessorio.preco | currency }}
                    <label class="toggle toggle-assertive">
                      <input type="checkbox" ng-model='isMarcado' ng-click="mudou(acessorio, isMar
                      <div class="track">
                        <div class="handle"></div>
                      </div>
                    </label>
                  </li>
            </ul>

        </div>
        <div class="item item-divider">
          R{{carroEscolhido.preco | currency }}
        </div>
    </div>
```

Ao testarmos a aplicação no navegador, veremos que o valor total do carro aparecerá modificado no rodapé da lista.



Na lista de acessórios, visualizamos o nome do carro, os itens e o preço final. O mesmo acontecerá se rodarmos a aplicação no Android.



É recomendável que sejam feito testes nas duas plataformas, um dos benefícios oferecidos pelo Ionic-lab.