05

# Completando a lista de carros e a tela de finalização de compra

## Completando a lista de carros e a tela de finalização de compra

Nós já temos a lista de carros e os acessórios oferecidos, conseguimos adicionar ou diminuir os valores do preço final e depois, e depois, é possível chegar a tela de finalização de pedido.

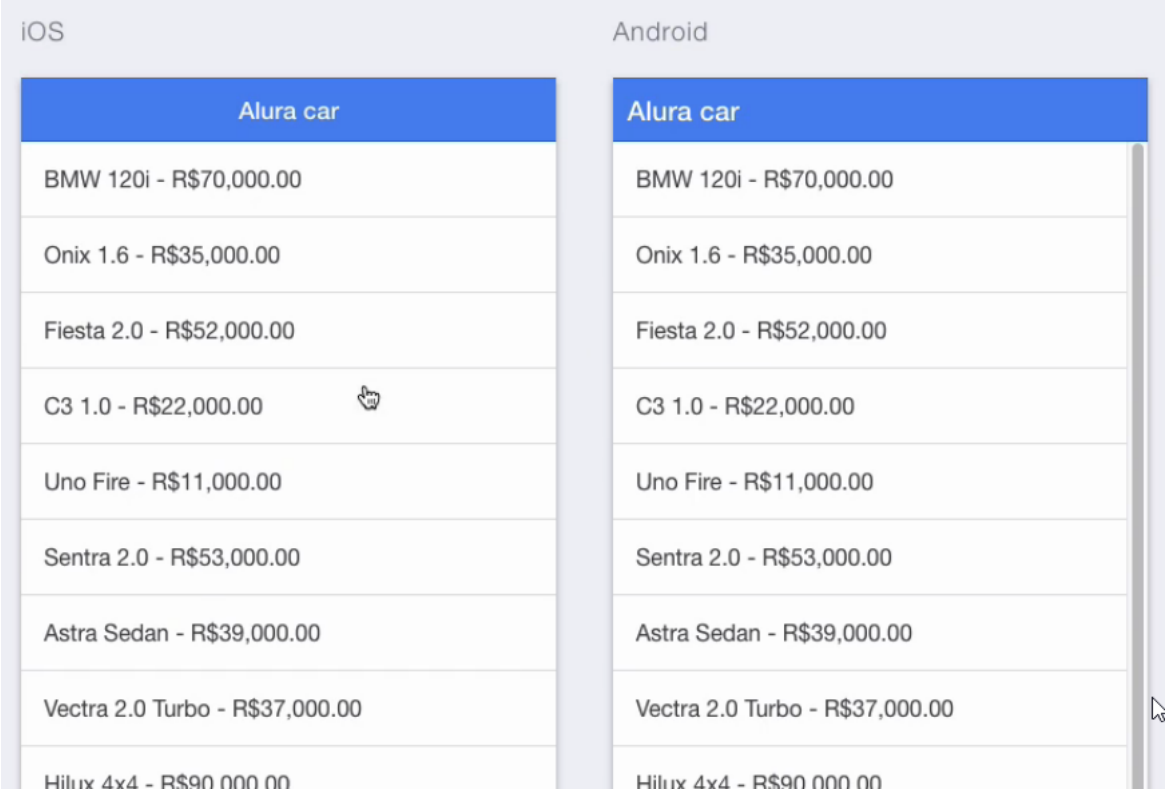

Temos um problema na nossa lista de carros, que agora só tem um item.

Lembre-se que Cada *view* possui uma `controller` .

A seguir, terminaremos de preencher a nossa lista.

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

    $scope.listaDeCarros = [{"nome" : "BMW 120i" , "preco" : 70000},
                            {"nome" : "Onix 1.6" , "preco" : 35000},
                            {"nome" : "Fiesta 2.0", "preco" : 52000},
                            {"nome" : "C3 1.0", "preco" : 22000},
                            {"nome" : "Uno Fire", "preco" : 11000},
                            {"nome" : "Sentra 2.0", "preco" : 53000},
                            {"nome" : "Astra Sedan", "preco" : 39000},
                            {"nome" : "Vectra 2.0", "preco" : 37000},
                            {"nome" : "Hilux 4x4", "preco" : 90000},
                            {"nome" : "Montana Cabine dupla", "preco" : 57000},
                            {"nome" : "Outlander 2.4", "preco" : 99000},
                            {"nome" : "Fusca 15000", "preco" : 6000}
                           ];
});
```

Já veremos a lista preenchida com os carros e preços.




Se testarmos selecionar outros carros, ao irmos para a tela de acessórios, veremos que o preço final é alterado corretamente. O que demonstra que a lista está funcionando, independente do veículo escolhido.

Agora, continuaremos criando a tela de finalização da compra do carro. Precisamos que seja visualizado, o carro escolhido e preço que será pago pelo comprador. Mas por enquanto, nenhum dos dois são visualizados na terceira tela. Faremos isto passando as informações pelo link. Na rota de `finalizarpedido`, do arquivo `routes.js`, não estamos recebendo nenhum parâmetro. Faremos isto adicionando o `/:carro`

```
.state('finalizarpedido',{
    url : '/finalizarpedido/:carro',
    templateUrl : 'templates/finalizarpedido.html',
    controller : 'FinalizarPedidoController'
})
```

Em seguida, no `carroescolhido.html`, vamos adicionar o link do carro escolhido no botão.

```
<a class="button button-positive button-full" href="#/finalizarpedido/{{carroEscolhido}}">Finalizar
```

Mas até aqui, não vamos conseguiremos ver o objeto na tela de finalização de produto. Faremos isto no `controller`, adicionando o parâmetro `$stateParams` e o escopo.

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

});
```

Criaremos a variável `carroFinalizado` dentro do escopo do Angular, seguido pelo parâmetro da outra tela, `$stateParams` .

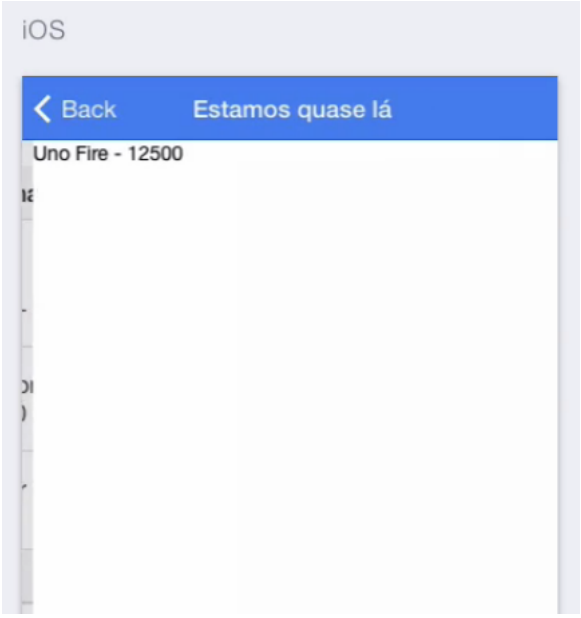
```
angular.module('starter')
.controller('FinalizarPedidoController', function($stateParams, $scope){

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

});
```

Observe que quando enviamos uma variável, ele chega em formato de *string*, por isso nós a transformamos uma variável Angular adicionando o `angular.fromJson` .

Depois, usaremos a variável `carroFinalizado` dentro do no arquivo `finalizarpedido.html` .

```
<ion-view>
    <ion-nav-title>Estamos quase lá</ion-nav-title>
    <ion-content>
        {{carroFinalizado.nome}} - {{carroFinalizado.preco}}
    <ion-content>
</ion-view>
```

Após salvarmos, veremos como ficou. Na aplicação, iremos selecionar o "Uno Fire", em seguida, adicionaremos o "Ar-condicionado" e o "MP3 Player" e finalizaremos o pedido. O valor total será "R$12.500".

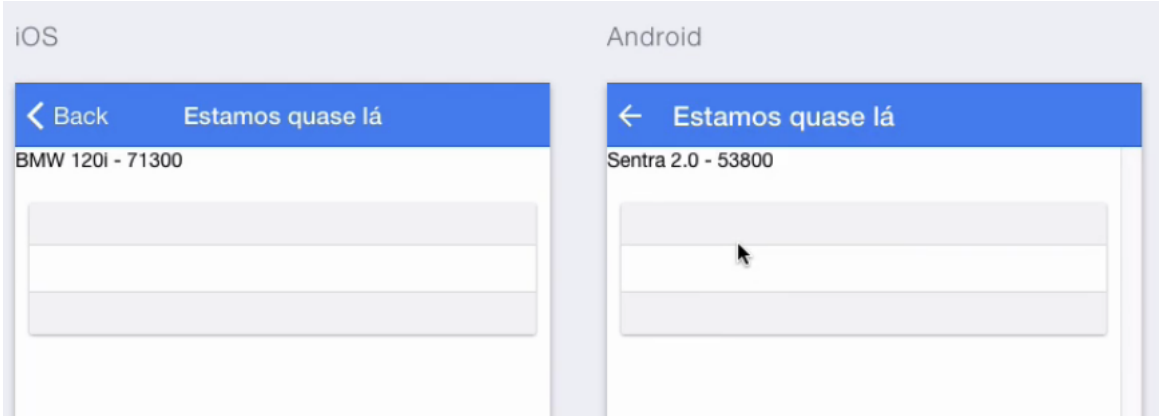



Já conseguimos passar o preço atualizado para a terceira tela. Mas nossa aplicação não tem um aspecto de mobile. Vamos voltar na nossa *view* e criaremos um card, usando o mesmo padrão da tela de acessórios.

Nós iremos aproveitar a estrutura do card do `carroescolhido.html` , que inclui o cabeçalho, o conteúdo e o rodapé.

```
<ion-view>
    <ion-nav-title>Estamos quase lá</ion-nav-title>
    <ion-content>
        {{carroFinalizado.nome}} - {{carroFinalizado.preco}}
        <div class="card">
```

```
            <div class="item item-divider">
            </div>
            <div class="item item-text-wrap">
            </div>
            <div clss="item item-divider">
            </div>
          </div>
      </ion-content>
  </ion-view>
```




Veremos que o card foi adicionado, mas ainda precisaremos fazer melhorias, como passar o texto para dentro do card. Iremos movê-lo para o cabeçalho e usaremos o `R` e o filtro `currency`.

```
<ion-view>
    <ion-nav-title>Estamos quase lá</ion-nav-title>
    <ion-content>

        <div class="card">
          <div class="item item-divider">
              {{carroFinalizado.nome}} - R{{carroFinalizado.preco | currency}}
          </div>
          <div class="item item-text-wrap">

          </div>
          <div clss="item item-divider">
          </div>
        </div>
    </ion-content>
</ion-view>
```

Daremos um destaque ao texto do cabeçalho, usando as *tags* `<h2>` e `<h2>`.

```
<div class="item item-divider">
    <h2>{{carroFinalizado.nome}}</h2>
    <h2>R{{carroFinalizado.preco | currency}}</h2>
</div>
```

Podemos ver a diferença no texto.

A concessionária tem um serviço de delivery. Se o cliente compra o carro pela aplicação, significa que ele quer receber o veículo sem precisar ir na loja física. Precisaremos criar um formulário para que o cliente preencha seus dados, como nome, endereço e email, por exemplo. Até aqui, não temos como ter acesso a estas informações.

O Ionic possui vários componentes de formulário. Podemos verificar algo a respeito na parte de Forms & Inputs (http://ionicframework.com/docs/components/#forms) na documentação do Ionic.

Nós adicionaremos o novo trecho na parte central da tela de de finalização de pedido. Vamos incluir o componente `label` na aplicação.

```
<div class="item item-text-wrap">
    <div class="item item-divider">
        <h2>{{carroFinalizado.nome}}</h2>
        <h3>R{{carroFinalizado.preco | currency}}</h3>

    </div>
    <div class="item item-text-wrap">
        <div class="list">
            <label class="item item-input item-stacked-label">
                <span class="input-label">First Name</span>
                <input type="text" placeholder="John">
            </label>
        </div>
    </div>
```

Mas veremos que no `span` será visualizado `First Name` e no `input` o usuário encontrará o texto `John`. Queremos substituir os termos em inglês, por `Nome Completo` e apagar o `placeholder` do código.

```
<label class="item item-input item-stacked-label">
    <span class="input-label">Nome Completo</span>
    <input type="text">
</label>
```

Vamos conferir o resultado:

Ele irá solicitar o nome completo do usuário, que será preenchido no campo logo abaixo. Em seguida, criaremos novas `label`s para os campos "endereço" e "email".

```
<div class="item item-text-wrap">
    <div class="list">
        <label class="item item-input item-stacked-label">
            <span class="input-label">Nome Completo</span>
            <input type="text">
        </label>
        <label class="item item-input item-stacked-label">
            <span class="input-label">Endereço</span>
            <input type="text">
        </label>
        <label class="item item-input item-stacked-label">
            <span class="input-label">Email</span>
```

```
            <input type="text">
          </label>
      </div>
  </div>
```

Não incluiremos o campo para o "CPF" porque este é um dado que precisa de validação.

Até aqui a nossa aplicação funcionará da seguinte maneira no iOS e no Android.

Este era o resultado que gostaríamos de alcançar.