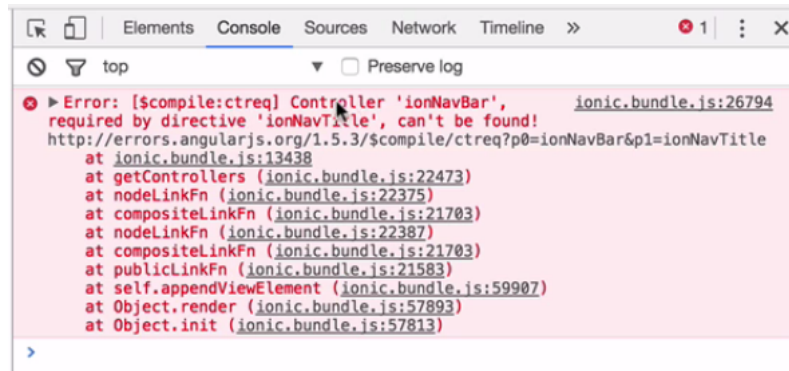


Refatoração e ajustes no botão voltar

Transcrição

Após fazer o login da minha aplicação no iOS, vamos testar a navegação. Se clicarmos na tela com a listagem de carros, quando clicarmos em alguma das opções, não conseguiremos avançar. Ao adicionarmos o plugin da câmera e quebramos a aplicação. Vamos acessar o Console nas Ferramentas do Desenvolvedor.



Nós trocamos o sistema de rota da aplicação, utilizando uma abstrata na listagem, mas não fizemos o mesmo com as outras telas: `carroescolhido` e `finalizarpedido`. Teremos que fazer esta refatoração. Atualmente a rota `carroescolhido` está assim no arquivo `routes.js`:

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

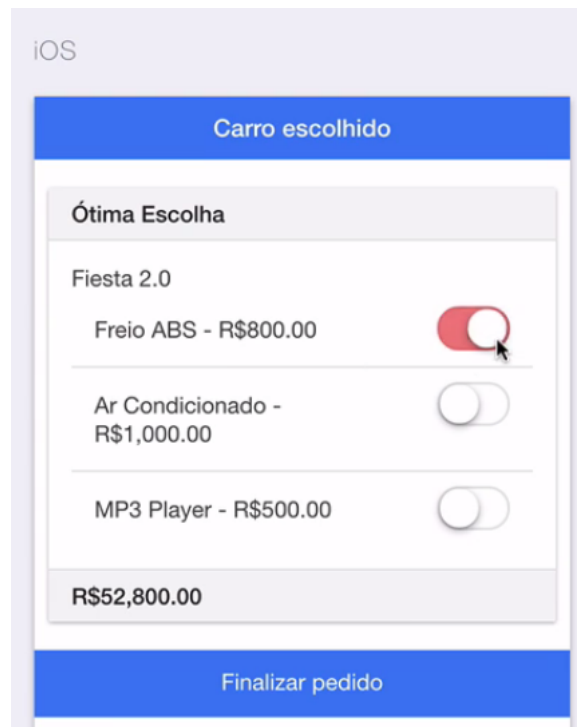
Com a refatoração, a rota ficará assim:

```
.state('app.carroescolhido', {
  url : '/carroescolhido/:carro',
  views : {
    'menuContent' : {
      templateUrl : 'templates/carroescolhido.html',
      controller: 'CarroEscolhidoController'
    }
  }
})
```

Vamos testar se tudo está funcionando. Se tentarmos fazer o login agora, a navegação voltará repetidamente para a tela de login. Isto está acontecendo, porque quando criamos o item, o `href` direciona da listagem para o `carrofinalizado`. No entanto, a rota mudou. Precisaremos mudar também o link. Adicionaremos o `/app/` no caminho passado no `href` do arquivo `listagem.html`:

```
<ion-item ng repeat="carro in listaDeCarros" href="#/app/carroescolhido/{{carro}}" >
  {{carro.nome}} - R{{carro.preco | currency }}
</ion-item>
```

Salvaremos os ajustes na rota, depois, conseguiremos fazer o login e ir para tela de listagem. Quando clicarmos em um dos modelos, por exemplo no Fiesta, ele seguirá com a navegação.



Porém, na tela de Carro escolhido, teremos problema com o link de "Finalizar pedido". Precisaremos alterar a rota de finalizapedido. Atualmente, ela está assim no routes.js :

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

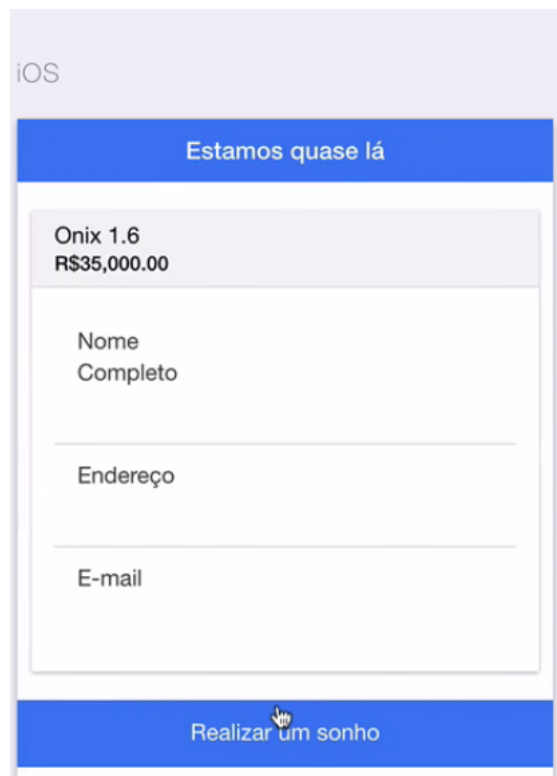
Com a refatoração, ele ficará assim:

```
.state('app.finalizapedido',{
  url : '/finalizapedido/:carro',
  views : {
    'menuContent' : {
      templateUrl : 'templates/finalizapedido.html',
      controller: 'FinalizarPedidoController'
    }
  }
})
```

No arquivo carroescolhido.html, modificaremos o caminho no href do botão:

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

Agora, ao clicarmos no botão "Finalizar pedido", seremos levados para a tela "Estamos quase lá":



Ao clicarmos no botão "Realizar um sonho", veremos a mensagem de que acabamos de comprar um carro. Mas a partir dela, deveríamos ser levados para a tela de listagem. No entanto, permanecemos na mesma tela. Teremos que fazer um conserto no código, lembrando que o botão "Realizar um sonho" foi feito pelo `controller`.

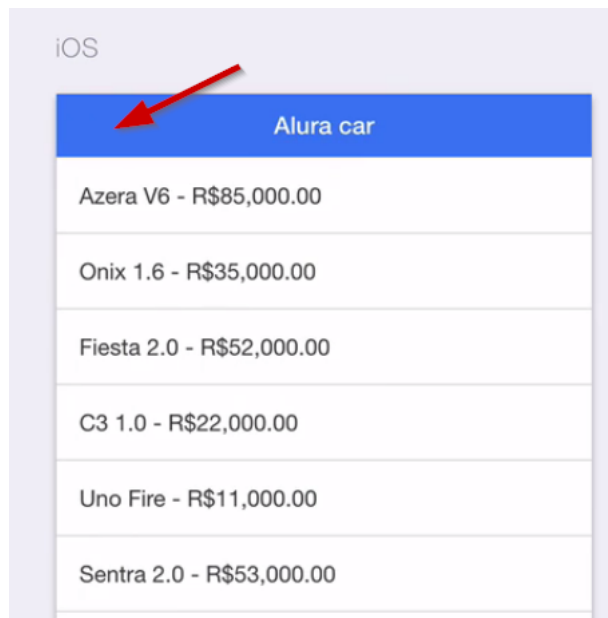
```
CarroService.salvarPedido(pedidoFinalizado).then(function(dados){

    $ionicPopup.alert({
        title: 'Parabéns',
        template: 'Você acaba de comprar um carro.'
    }).then(function(){
        $state.go('app.listagem');
    });

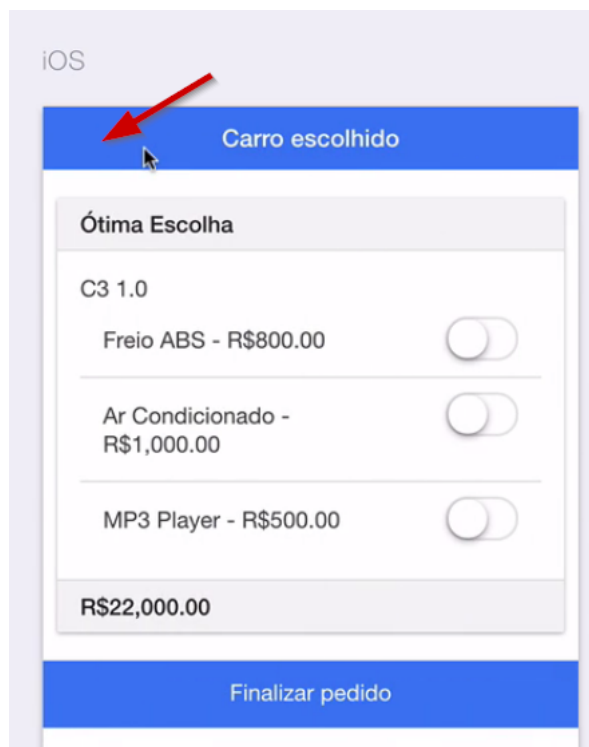
    //...
```

Observe que adicionamos o `app` juntamente ao `listagem`.

Agora, ao finalizarmos a compra compra, voltaremos para a tela de listagem. Mas se repararmos na parte superior da telas, não encontraremos o menu hambúrguer...



E nem o botão "Voltar"...

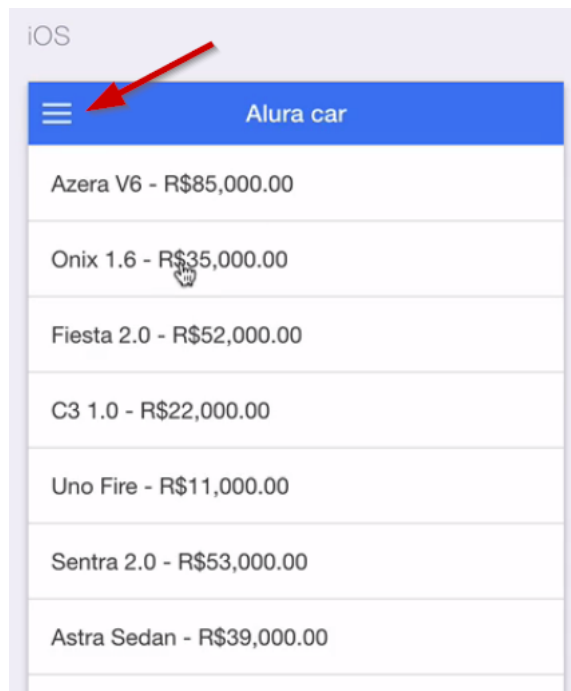


Eles não estão aparecendo, porque quando fizemos a aplicação, os arquivos ficavam no arquivo `index.html`. E nós retiramos a barra de navegação do arquivo. Nós adicionamos uma nova `nav-view`, mas não colocamos o botão. Precisaremos adicionar o botão no `<ion-nav-bar>` do arquivo `menu.html`:

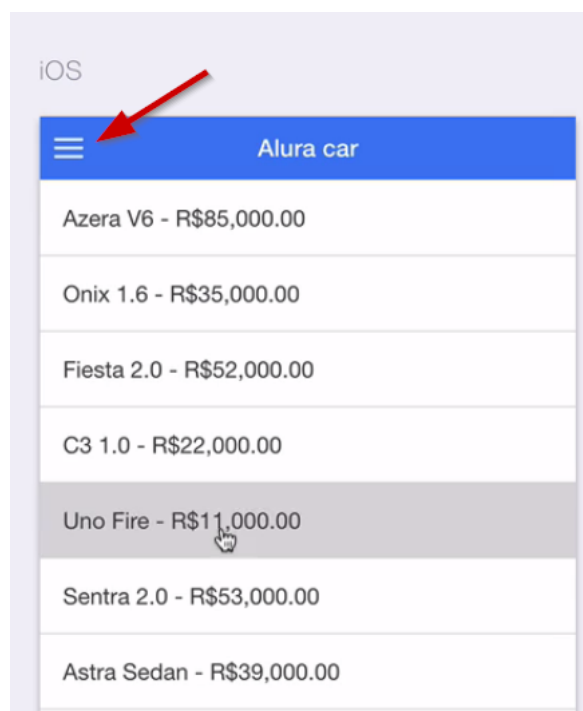
```
<ion-side-menus>
  <ion-side-menu-content>
    <ion-nav-bar class="bar bar-positive">
      <ion-nav-back-button>

      </ion-nav-back-button>
      <ion-nav-buttons side="left">
        <button class="button button-icon button-clear ion-navicon" menu-toogle="center"></button>
      </ion-nav-buttons>
    </ion-nav-bar>
  </ion-side-menu-content>
</ion-side-menus>
```

Agora, veremos o ícone de hamburguer...



E a tela com botão de "Back".



Mas agora, teremos que analisar a navegação. Após comprarmos o carro, voltamos para a tela de listagem, mas na barra aparece o botão de voltar. Isto está bem errado! O Ionic acredita que existe uma história sequencial. No entanto, queremos que quando ele retorne a listagem, o botão de voltar seja desabilitado. Para resolver o problema, podemos consultar a documentação na seção [Ionic History \(https://ionicframework.com/docs/api/service/\\$ionicHistory/\)](https://ionicframework.com/docs/api/service/$ionicHistory/). Nela, encontraremos a função `nextViewOptions()` que oferece opções para a próxima View. No caso, queremos que na próxima tela, seja desabilitado o botão de voltar, por isso, usaremos a opção `disableBack`. O padrão é `false`, teremos que habilitar o `true`. Vamos injetar o `$ionicHistory` no `controllers.js`.

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

```

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

$scope.pedido = {};
//...

```

Depois, iremos adicionar a função `nextViewOptions()` na controller do `finalizarPedido`

```

CarroService.salvarPedido(pedidoFinalizado).then(function(dados){

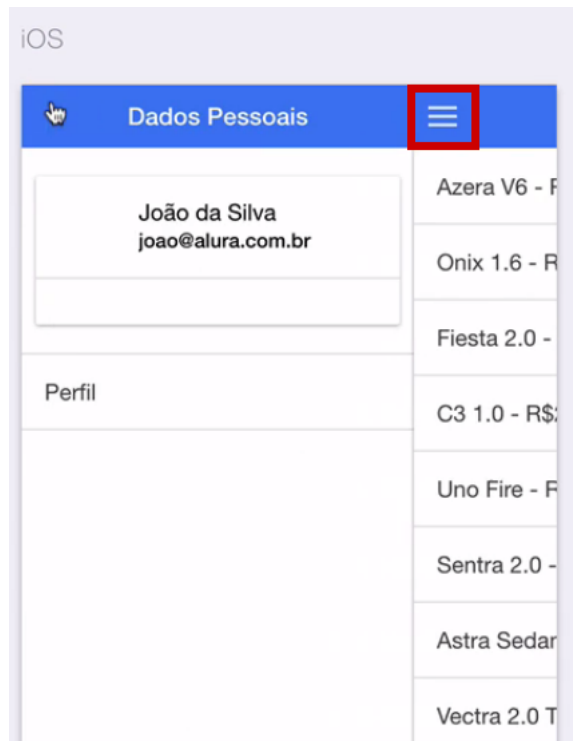
    $ionicHistory.nextViewOptions({
        disableBack : true
    })

    $ionicPopup.alert({
        title: 'Parabéns',
        template: 'Você acaba de comprar um carro.'
    }).then(function(){
        $state.go('app.listagem');
    });

    //...

```

O `nextViewOptions()` recebeu o atributo `disableBack` com o valor `true`. Vamos testar novamente o fluxo no app, do login até a compra finalizada.



Para finalizarmos, vamos alterar o botão "Back", pelo "Voltar" - considerando que toda a aplicação está em português. No arquivo `menu.html`, nós adicionamos uma tag de botão. Agora, vamos incluir o `Voltar`.

```

<ion-side-menus>
  <ion-side-menu-content>
    <ion-nav-bar class="bar bar-positive">

```

```

<ion-nav-back-button>Voltar

</ion-nav-back-button>
<ion-nav-buttons side="left">
  <button class="button button-icon button-clear ion-navicon" menu-toogle="center"></t
</ion-nav-buttons>

//...

```

Se testarmos, veremos que na barra superior, encontramos o texto "Voltar" ao lado da seta.



Agora, analisaremos o arquivo `controllers.js`. Temos muitas controllers no mesmo arquivo. Quando tínhamos apenas três, conseguíamos dar manutenção. Porém, o arquivo ficou muito grande. É uma boa prática separar cada controller em um arquivo. Assim faremos a manutenção mais facilmente, por exemplo, se quisermos alterar a controller `listagem`, iremos no arquivo `listagem.controller.js`. Lembrando que separar os arquivos é uma boa prática.

Começaremos criando uma nova pasta dentro da `js`, que chamaremos de `controllers`. Depois, separaremos todas as controllers, criando os seguintes arquivos:

- `listagem.controller.js`
- `carroEscolhido.controller.js`
- `finalizarPedido.controller.js`
- `login.controller.js`
- `menu.controller.js`
- `perfil.controller.js`

Em seguida, iremos carregá-los no `index.html` - que antes só tinha `controllers.js`.

```

<!-- your app's js -->
<script src="js/app.js"></script>

```

```
<script src="js/controllers/listagem.controller.js"></script>
<script src="js/controllers/carroEscolhido.controller.js"></script>
<script src="js/controllers/finalizarPedido.controller.js"></script>
<script src="js/controllers/login.controller.js"></script>
<script src="js/controllers/menu.controller.js"></script>
<script src="js/controllers/perfil.controller.js"></script>
<script src="js/routes.js"></script>
<script src="js/service.js"></script>
```

Após carregarmos os novos arquivos, podemos excluir o `controllers.js`, porque ele ficou vazio. Se fizermos o teste, veremos que conseguiremos seguir o fluxo corretamente. Se testarmos todos os botões, seremos levado para as telas corretas.

Nossa navegação funciona por completo, após refatoramos o código e seguir as boas práticas.

Até a próxima!