01

Dividir para conquistar!

Transcrição

Começando deste ponto? Você pode fazer o DOWNLOAD (https://s3.amazonaws.com/caelum-online-public/angular-1/stages/05-alurapic.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

Desmistificando Single Page Applications

Agora que a nossa aplicação ficou mais atraente, permitiremos que o usuário cadastre e edite fotos. Nada mais justo do que criarmos uma página de cadastro exclusiva. Porém, o Angular tem como foco a criação de Single Page Applications (Aplicação de única página), aquele tipo de aplicação que não recarrega durante uso. Então, será que a solução é colocar o HTML da página de cadastro e de todas as outras dentro de `index.html` ? Com certeza não, já demos um duro para tornar nossa página mais fácil de manter. Então que diabos é esse de "não recarrega a página durante seu uso"?

Então, realmente a página `index.html` não é recarregada, é a mesma durante todo o ciclo de vida da aplicação. Porém, é através de URL's especiais chamadas de **rotas** que o Angular busca a página desejada, inclusive ele já faz o trabalho de atualizar `index.html` automaticamente para nós com o conteúdo da página. É como se `index.html` fosse um template com uma única lacuna que recebe o conteúdo de outras páginas, mas uma por vez! Essas páginas não devem conter as tags `head` e `body` , pois serão inseridas dentro da página principal da aplicação que já os tem. É por isso que são chamadas de **páginas parciais** , ou *partial pages*, no inglês.

Nossas primeiras views parciais

Antes de avançarmos, vamos criar a parcial: `public/partials/principal.html` . Ela conterá a cópia da marcação que lista nossas fotos lá em `index.html` :

```
<!-- public/partials/principal.html -->

<div class="jumbotron">
    <h1 class="text-center">Alurapic</h1>
</div>

<div class="row">
    <div class="col-md-12">
    <form>
         <input class="form-control" placeholder="filtrar pelo título da foto" ng-model="filtro'
    </form>
    </div> <!-- fim col-md-12 -->
</div> <!-- fim row -->

<div class="row">
    <meu-painel class="col-md-2 painel-animado" ng-repeat="foto in fotos | filter: filtro" titu
        <img class="img-responsive center-block" src="{{foto.url}}" alt="{{foto.titulo}}">
    </meu-painel>
</div>
```

Então, em `index.html` vamos substituir esta marcação pela diretiva **ng-view**. Vamos aproveitar e **remover a diretiva ng-controller da tag body**:

```
<!-- public/index.html -->
<!DOCTYPE html>
<html lang="pt-br" ng-app="alurapic">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width">
        <title>Alurapic</title>
        <link rel="stylesheet" href="css/bootstrap.min.css">
        <link rel="stylesheet" href="css/bootstrap-theme.min.css">
        <link rel="stylesheet" href="css/efeitos.css">
        <script src="js/lib/angular.min.js"></script>
        <script src="js/lib/angular-animate.min.js"></script>
        <script src="js/main.js"></script>
        <script src="js/controllers/fotos-controller.js"></script>
        <script src="js/directives/minhas-diretivas.js"></script>
    </head>
    <body>
        <div class="container">

            <!-- opa! abri uma mega lacuna! -->
            <ng-view></ng-view>

        </div><!-- fim container -->
    </body>
</html>
```

O módulo ngRoute e configurações de rotas

Você não precisa ser um vidente para saber que, assim que abrirmos o endereço `http://localhost:3000` , veremos uma página em branco, porém já sabe que precisamos configurar as tais **rotas** do Angular.

Rotas são configuradas com auxílio do módulo **ngRoute**. Como sempre, isso envolve a importação de um script e adição do módulo como dependência do módulo principal da aplicação:

```
<!-- public/index.html -->
<!DOCTYPE html>
<html lang="pt-br" ng-app="alurapic">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width">
        <title>Alurapic</title>
        <link rel="stylesheet" href="css/bootstrap.min.css">
        <link rel="stylesheet" href="css/bootstrap-theme.min.css">
        <link rel="stylesheet" href="css/efeitos.css">
        <script src="js/lib/angular.min.js"></script>
        <script src="js/lib/angular-animate.min.js"></script>

        <!-- Importando o script do módulo ngRoute -->

        <script src="js/lib/angular-route.min.js"></script>
        <script src="js/main.js"></script>
```

```
        <script src="js/controllers/fotos-controller.js"></script>
        <script src="js/directives/minhas-diretivas.js"></script>
    </head>
    <body>
        <div class="container">
            <ng-view></ng-view>
        </div><!-- fim container -->
    </body>
</html>
```

Agora, em `main.js`:

```
// public/js/main.js

angular.module('alurapic', ['minhasDiretivas', 'ngAnimate', 'ngRoute']);
```

Pronto, agora que temos encaixadas as peças do quebra-cabeças, vamos às configurações! Bem, poderíamos até criar um módulo exclusivo com as configurações das rotas da aplicação, mas não é incomum essa configuração ser feita diretamente no módulo principal da aplicação através do serviço **$routeProvider**, que obtemos através do sistema de injeção de dependências do Angular, dentro da função **config**:

```
// public/js/main.js

angular.module('alurapic', ['minhasDiretivas', 'ngAnimate', 'ngRoute'])
    .config(function($routeProvider) {

    });
```

Usamos `$routeProvider` da seguinte maneira. Quando (**when**) o usuário acessar determinada rota, enviaremos para ele uma view parcial (**templateUrl**) e também indicamos qual é o controlador (**controller**) para aquela parcial. Definir o controller nos dá flexibilidade de usar uma mesma parcial com controllers diferentes:

```
// public/js/main.js

angular.module('alurapic', ['minhasDiretivas', 'ngAnimate', 'ngRoute'])
    .config(function($routeProvider) {

        $routeProvider.when('/fotos', {
            templateUrl: 'partials/principal.html',
            controller: 'FotosController'
        });

    });
```

Se abrirmos o endereço `http://localhost:3000` nada será exibido, inclusive se tentarmos acessar `http://localhost:3000/fotos`. O Angular precisa de alguma maneira saber que o endereço que estamos tentando acessar não é um endereço do nosso servidor, mas uma de suas rotas. Fazemos isso adicionando um `#` (hash) antes da rota:

```
http://localhost:3000/#/fotos
```

Rotas ainda mais elegantes com Html5Mode

Agora sim! Nossa view parcial `principal.html` é exibida! Mas um olhar atento revela que essa URL é um tanto estranha, pela presença do `#`, mas esse é um subterfúgio muito utilizado para que seja possível favoritar em seu navegador a parcial, mas, ainda assim, quando alguém abrir o endereço, quem será carregada é a view principal `index.html`, que buscará a parcial através do sistema de rotas do Angular. Com o advento do HTML5 foi criada a History API, que permite conseguirmos o mesmo resultado, mas sem o uso do `#`. O Angular suporta a History API e podemos ativá-la através do serviço **$locationProvider**, que também podemos pedir através do sistema de injeção de dependências:

```
// public/js/main.js

angular.module('alurapic', ['minhasDiretivas', 'ngAnimate', 'ngRoute'])
    .config(function($routeProvider,  $locationProvider) {

       $locationProvider.html5Mode(true);

        $routeProvider.when('/fotos', {
            templateUrl: 'partials/principal.html',
            controller: 'FotosController'
        });

    });
```

Além dessa configuração, precisamos adicionar a tag `<base href="/">`:

```
<!DOCTYPE html>
<html lang="pt-br" ng-app="alurapic">
    <head>
        <base href="/">
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width">
        <title>Alurapic</title>
        <link rel="stylesheet" href="css/bootstrap.min.css">
        <link rel="stylesheet" href="css/bootstrap-theme.min.css">
        <link rel="stylesheet" href="css/efeitos.css">
        <script src="js/lib/angular.min.js"></script>
        <script src="js/lib/angular-animate.min.js"></script>
        <script src="js/lib/angular-route.min.js"></script>
        <script src="js/main.js"></script>
        <script src="js/controllers/fotos-controller.js"></script>
        <script src="js/directives/minhas-diretivas.js"></script>
    </head>
    <body>
        <div class="container">
            <ng-view></ng-view>
        </div><!-- fim container -->
    </body>
</html>
```

Agora, podemos remover o hash e acessar a URL diretamente:

```
http://localhost:3000/fotos
```

Outro ponto importante é que se o seu navegador não suportar este modo, automaticamente o Angular adotará a estratégia com `#` (hash).

ATENÇÃO: para usar html5Mode seu servidor precisa estar preparado. Esta configuração está fora do escopo deste treinamento e você deve consultar a documentação do seu servidor favorito.

Excelente, mas o que acontece se agora acessarmos um endereço que não existe? Nada será exibido, porém podemos indicar uma rota alternativa caso o endereço acessado pelo usuário não exista:

```
// public/js/main.js

angular.module('alurapic', ['minhasDiretivas', 'ngAnimate', 'ngRoute'])
    .config(function($routeProvider, $locationProvider) {

        $locationProvider.html5Mode(true);

        $routeProvider.when('/fotos', {
            templateUrl: 'partials/principal.html',
            controller: 'FotosController'
        });

        $routeProvider.otherwise({redirectTo: '/fotos'});

    });
```

Fantástico! Agora vamos criar a parcial do cadastro de fotos, apenas com um título e sua respectiva rota:

```
<!-- public/partials/foto.html -->
<h1>Cadastro</h1>
```

E agora vamos a rota `/fotos/new`, porém ainda sem definirmos um controller:

```
angular.module('alurapic', ['minhasDiretivas', 'ngAnimate', 'ngRoute'])
    .config(function($routeProvider, $locationProvider) {

        $locationProvider.html5Mode(true);

        $routeProvider.when('/fotos', {
            templateUrl: 'partials/principal.html',
            controller: 'FotosController'
        });

       // foto no singular!

        $routeProvider.when('/fotos/new', {
            templateUrl: 'partials/foto.html'
        });

        $routeProvider.otherwise({redirectTo: '/fotos'});
```

```
    });
```

Agora, basta acessarmos:

```
http://localhost:3000/fotos/new
```

Agora sim! Temos duas parciais com responsabilidades diferentes.

O que aprendemos neste capítulo?

- criação de parciais
- a diretiva ng-view
- o módulo ngRoute
- $routeProvider e configurações de rota
- rota padrão
- hash
- $LocationProvider e html5Mode