**Melhorando a experiência do usuário**

Começando deste ponto? Você pode fazer o [DOWNLOAD](https://s3.amazonaws.com/caelum-online-public/angular-1/stages/04-alurapic.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

**A lista pode crescer, e agora?**

Muito bem, evoluímos ainda mais nossa aplicação sob o ponto de vista interno, porém podemos melhorar a experiência do usuário. Por enquanto temos poucas fotos, mas eu tenho certeza que assim que aprendermos a cadastrar novas fotos e agrupá-las, o número será muito, muito maior! Então, que tal já implementarmos um mecanismo de procura que permita o usuário exibir apenas fotos de acordo com algum critério?

**Que tal filtrarmos nossa lista?**

Primeiro, vamos adicionar o campo de entrada da pesquisa. Vamos aproveitar a adicionar mais uma linha usando o sistema de GRID do Bootstrap, inclusive utilizaremos algumas classes especiais para estilizar elementos de entrada:

<!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">

<script src="js/lib/angular.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 ng-controller="FotosController">

<div class="container">

<div class="jumbotron">

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

</div>

<!-- novidade, a row com o campo de busca -->

<div class="row">

<div class="col-md-12">

<form>

<input class="form-control" placeholder="filtrar pelo título da foto">

</form>

</div> <!-- fim col-md-12 -->

</div> <!-- fim row -->

<div class="row">

<meu-painel class="col-md-2" ng-repeat="foto in fotos" titulo="{{foto.titulo}}">

<img class="img-responsive center-block" src="{{foto.url}}" alt="{{foto.titulo}}">

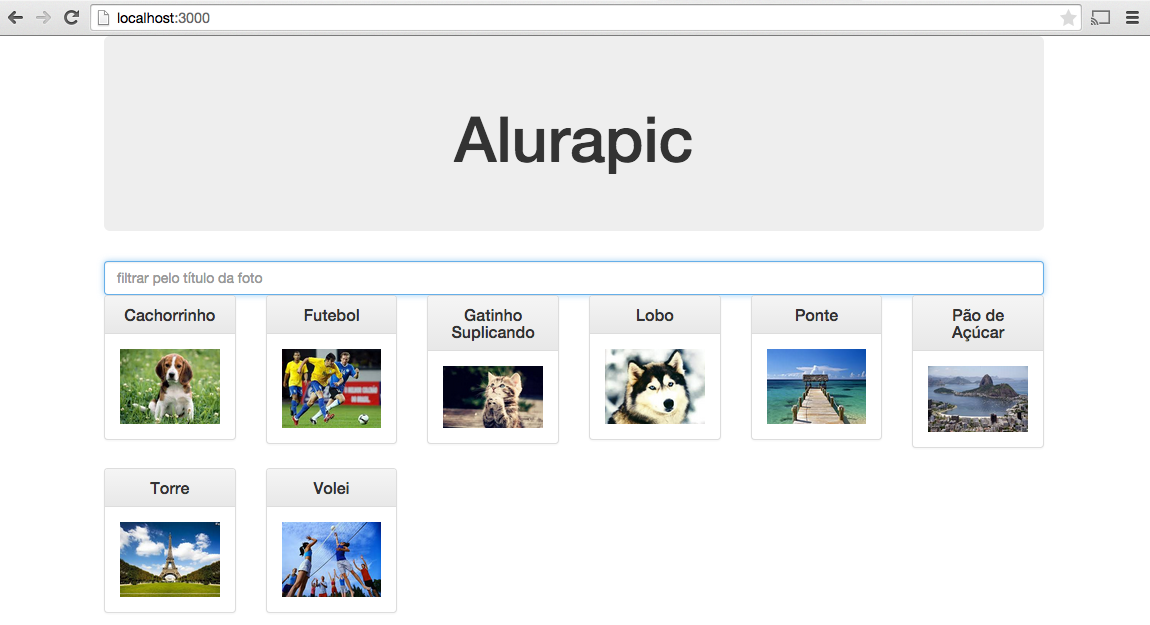
</meu-painel>

</div><!-- fim row -->

</div><!-- fim container -->

</body>

</html>



A ideia é simples: a medida que formos digitando, queremos exibir apenas as fotos que contenham em qualquer uma de suas propriedades o texto procurado.

Vamos adicionar em FotosController uma propriedade que guardará o texto digitado pelo usuário:

// public/js/controllers/fotos-controller.js

angular.module('alurapic').controller('FotosController', function($scope, $http) {

$scope.fotos = [];

$scope.filtro = '';

$http.get('/v1/fotos')

.success(function(retorno) {

$scope.fotos = retorno;

})

.error(function(erro) {

console.log(erro)

});

});

**Ah, então esse é o two-way data binding?**

Já sabemos acessar qualquer propriedade de $scope através de uma angular expression (AE), porém temos um problema: toda AE é somente leitura, isto é, não é capaz de atualizar $scope, que é justamente o que precisamos. É com base no que o usuário digitar que elaboraremos nossa estratégia de busca. Não queremos um data bindingunidirecional, queremos um bidirecional, aquele que é capaz de ler de $scope, inclusive atualizar seu valor de acordo com a entrada do usuário.

Felizmente o Angular suporta two-way data binging. A diferença é que não usamos AE, mas a diretiva **ng-model**. Vamos adicioná-la no input que recebe o filtro do usuário:

<!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">

<script src="js/lib/angular.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 ng-controller="FotosController">

<div class="container">

<div class="jumbotron">

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

</div>

<!-- novidade, a row com o campo de busca -->

<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" ng-repeat="foto in fotos" titulo="{{foto.titulo}}">

<img class="img-responsive center-block" src="{{foto.url}}" alt="{{foto.titulo}}">

</meu-painel>

</div><!-- fim row -->

</div><!-- fim container -->

</body>

</html>

**Eu não sabia que podemos filtrar com ng-repeat!**

E agora? Bem, toda vez que o usuário digitar neste campo, a propriedade $scope.filtro será atualizada! Excelente, mas como utilizaremos o valor corrente de $scope.filtro para filtrar a lista de fotos? A diretiva ng-repeat aceita receber um filtro através da propriedade filter, que deve ser adicionada imediatamente após um pipe |, sendo assim, ela ficará assim: **ng-repeat="foto in fotos | filter: filtro"**

<!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">

<script src="js/lib/angular.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 ng-controller="FotosController">

<div class="container">

<div class="jumbotron">

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

</div>

<!-- novidade, a row com o campo de busca -->

<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" ng-repeat="foto in fotos | filter: filtro" titulo="{{foto.titulo}}">

<img class="img-responsive center-block" src="{{foto.url}}" alt="{{foto.titulo}}">

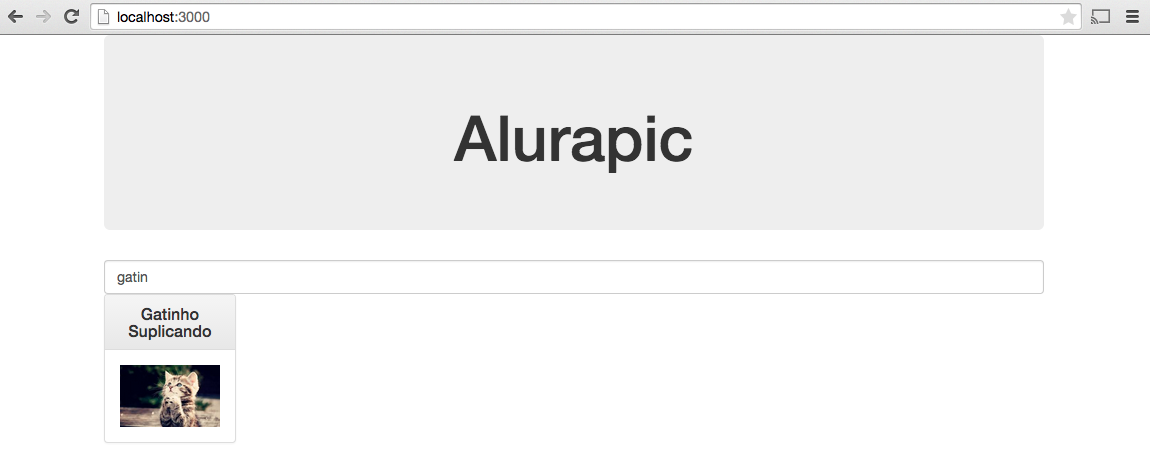
</meu-painel>

</div><!-- fim row -->

</div><!-- fim container -->

</body>

</html>



Inacreditável! Com pouquíssimo esforço, filtramos nossa lista, porém vale ressaltar que o filtro é aplicado em todas as propriedade do objeto foto de nossa lista, sendo assim, podemos encontrar uma foto pelo seu título, descrição, etc.

**Deixando o usuário ainda mais feliz animando nossa lista**

Você há de concordar que melhoramos bastante a aplicação, porém a aplicação do filtro é abrupta, instantânea, imediata! Para deixar o usuário feliz, que tal se animássemos a aplicação do filtro? Podemos fazer isso de várias maneiras, por exemplo, aplicado um efeito fadeout nas fotos que não atenderem nosso filtro.

Normalmente, esta é uma tarefa do mundo CSS, sendo assim, precisamos criar uma classe com o efeito desejado e aplicar essa classe via JavaScript para os elementos que deixarem nossa lista. Vamos criá-la, mas dentro do arquivopublic/css/efeitos.css:

// public/css/efeitos.css

.fade {

-moz-transform: scale(0.1);

-webkit-transform: scale(0.1);

-ms-transform: scale(0.1);

-o-transform: scale(0.1);

transform: scale(0.1);

}

Agora, vamos importar o CSS em index.html e aplicar a classe fade em nosso painel com a diretiva ng-repeat:

<!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/main.js"></script>

<script src="js/controllers/fotos-controller.js"></script>

<script src="js/directives/minhas-diretivas.js"></script>

</head>

<body ng-controller="FotosController">

<div class="container">

<div class="jumbotron">

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

</div>

<!-- novidade, a row com o campo de busca -->

<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 fade" ng-repeat="foto in fotos | filter: filtro" titulo="{{foto.titulo}}">

<img class="img-responsive center-block" src="{{foto.url}}" alt="{{foto.titulo}}">

</meu-painel>

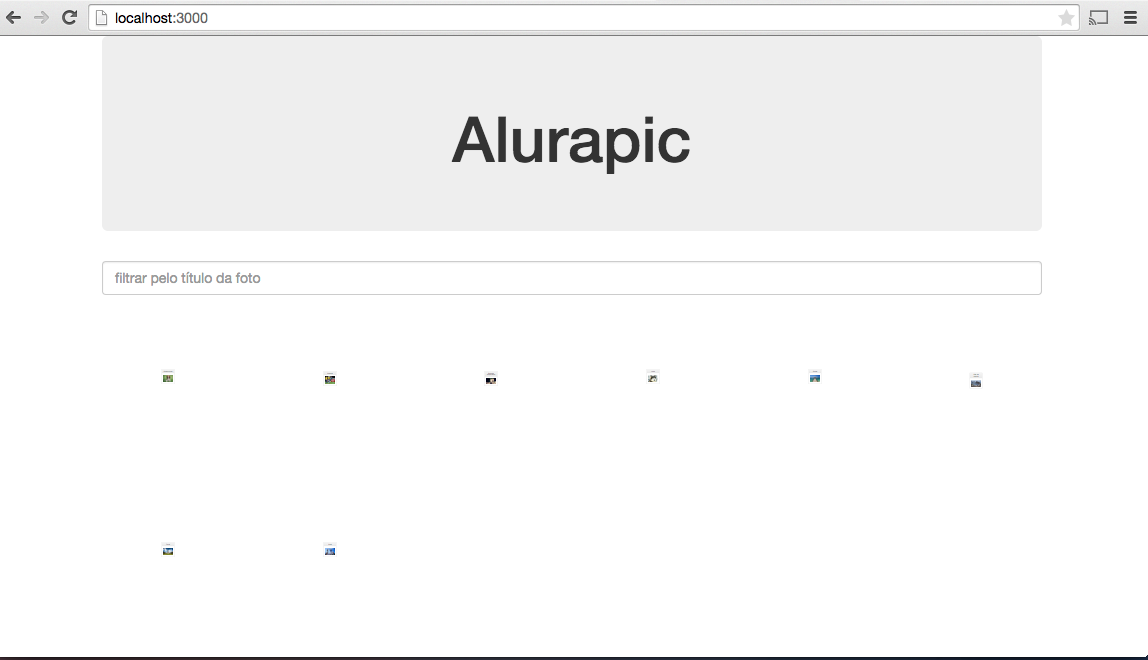
</div><!-- fim row -->

</div><!-- fim container -->

</body>

</html>

E testar logo em seguida:



Hum, não funcionou como esperado. Os elementos já começam invisíveis. A ideia é aplicar a classe painel-animadoapenas no elemento que **sair (leave)** da lista, isto é, para o elemento que não atender o critério do nosso filtro. E agora?

A equipe do Angular criou o módulo **ngAnimate**, que ataca justamente este problema. Antes de entrarmos nos detalhes de seu funcionamento, vamos importar seu script logo após o script *core* do Angular e adicioná-lo como dependência no 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>

<!-- importando o módulo ngAnimate -->

<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 ng-controller="FotosController">

<div class="container">

<div class="jumbotron">

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

</div>

<!-- novidade, a row com o campo de busca -->

<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">

<!-- adicionando a classe painel-animado -->

<meu-painel class="col-md-2 painel-animado" ng-repeat="foto in fotos | filter: filtro" titulo="{{foto.titulo}}">

<img class="img-responsive center-block" src="{{foto.url}}" alt="{{foto.titulo}}">

</meu-painel>

</div><!-- fim row -->

</div><!-- fim container -->

</body>

</html>

Agora, alterando a main.js:

// public/js/main.js

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

Pronto. Quando usamos o módulo ngAnimate, algumas diretivas do Angular passam a adicionar ou remover classes automaticamente sem a nossa ciência. São classes definidas pelo Angular, algo que lembra muito as pseudo classes do CSS3. Por exemplo, quando usamos a diretiva ng-repeat e um elemento sai da lista ele ganha a classeng-leave, e quando está para sair ng-leave-active. Existem outras classes e nem todas as diretivas suportam as mesmas classes, sendo necessário recorrer à documentação do Angular. Sendo assim, podemos resolver nosso problema aumentado a especificidade do nosso seletor .painel-animado:

// public/css/efeitos.css

.painel-animado.ng-leave-active {

-moz-transform: scale(0.1);

-webkit-transform: scale(0.1);

-ms-transform: scale(0.1);

-o-transform: scale(0.1);

transform: scale(0.1);

}

Testando mais uma vez, nossa lista é exibida mas nenhuma animação é efetuada quando digitamos algo no filtro, isto porque não usamos o poderoso transition do CSS. Por mais que estejamos aprendendo Angular, trabalhar com o módulo ngAnimate requer que você conheça os poderosos recursos do CSS3, a única coisa que o Angular faz é aplicar ou remover determinadas classes. Para efetuar a transição fazemos:

// public/css/efeitos.css

// novidade! Ativa a transição no elemento. Agora, quando o transform for aplicado apenas para os elementos que estão saindo da lista, a escala será animada!

.painel-animado {

-moz-transition:transform 0.8s;

-webkit-transition:transform 0.8s;

-ms-transition:transform 0.8s;

-o-transition:transform 0.8s;

transition:transform 0.8s;

}

.painel-animado.ng-leave-active {

-moz-transform: scale(0.1);

-webkit-transform: scale(0.1);

-ms-transform: scale(0.1);

-o-transform: scale(0.1);

transform: scale(0.1);

}

(CUIDADO: não é para dar um espaço entre as classes .painel-animado e .ng-leave-active , caso contrário o seletor se tornará hierárquico e o efeito não vai funcionar. Os dois devem estar grudados! Se quiser saber mais sobre esse tipo de seletor o Alura possui treinamentos de CSS)

Maravilha! Agora nossa transição funciona perfeitamente, ou quase perfeita (para perceber, é necessário ver o vídeo, não é possível através da foto ilustrar o problema a seguir)!

**O atraso pode ser a melhor opção!**

Repare que nossa lista é filtrada a cada tecla que pressionamos e os elementos que não condizem com nosso filtro são animados e retirados da lista, mas a coisa é feita tão instantaneamente que o efeito deixa um pouco a desejar. Para resolvermos isso, precisamos realizar um pequeno atraso (delay) na entrada do usuário. Por exemplo, a cada dígito do usuário, vamos aguardar meio segundo (500ms) para que o framework atualize o valor de $scope.filtroe filtre nossa lista. Fazemos isso através da diretiva **ng-model-options**:

<!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>

<!-- importando o módulo ngAnimte -->

<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 ng-controller="FotosController">

<div class="container">

<div class="jumbotron">

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

</div>

<!-- novidade, a row com o campo de busca -->

<div class="row">

<div class="col-md-12">

<form>

<input class="form-control" placeholder="filtrar pelo título da foto" ng-model="filtro" ng-model-options="{ debounce: 500 }">

</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" titulo="{{foto.titulo}}">

<img class="img-responsive center-block" src="{{foto.url}}" alt="{{foto.titulo}}">

</meu-painel>

</div><!-- fim row -->

</div><!-- fim container -->

</body>

</html>

Agora sim! Há um pequeno atraso na atualização do valor de $scope.filtro, por conseguinte um atraso nos elementos que deixaram a lista, deixando nossa animação mais bacana.

Melhoramos ainda mais a experiência do usuário e com certeza, só temos a ganhar com isso!

**O que aprendemos neste capítulo?**

* a diretiva ng-model e two-way data binding
* aplicação de filtro na diretiva ng-repeat
* ng-model-options e postergação do two-way data binding
* animações com o módulo ngAmimate
* animações requerem conhecimento sólido de CSS3