|  |
| --- |
| 广西广电网络.信息化部 |
| Angular.JS |
| 快速入门 |
|  |
| **黄少文** |
| **2015/10/8** |

|  |
| --- |
| Angular.JS的快速入门介绍文档，用于快速了解架构的基本概念。 |

文档目录

[文档目录 1](#_Toc432082564)

[一、基础概念 2](#_Toc432082565)

[二、数据模板绑定 4](#_Toc432082566)

[三、Bootstrap静态页面 6](#_Toc432082567)

[四、MVC代码结构 9](#_Toc432082568)

[五、搜索编辑框 14](#_Toc432082569)

[六、列表排序 16](#_Toc432082570)

[七、动态获取后台数据 20](#_Toc432082571)

[八、添加图片和链接 24](#_Toc432082572)

# 一、基础概念

今天开始一步一步学习 AngularJS 的教程之旅。

在这一小节中，我们将学习最基础、同时也是重要的代码结构和数据绑定。

在开始之前，我们先来看看 [演示](http://wenzhixin.net.cn/demos/angular-scutech/step0)， 也可以通过链接下载 [zip](http://wenzhixin.net.cn/demos/angular-scutech/step0.zip) 包。

可能没有让你觉得很兴奋，不过一切都只是开始。

可以看到页面显示“现在还没有任何内容！”，通过查看 html 的代码， 可以看到一些我们所需要的 Angular 的简单的关键字。

文件index.html

**<!doctype html>**

<html lang="en" ng-app>

<head>

 <meta charset="utf-8">

 <title>Angular Scutech</title>

 <link rel="stylesheet" href="lib/bootstrap/css/bootstrap.css">

 <script src="lib/angular/angular.js"></script>

</head>

<body>

<div class="navbar navbar-inverse" role="navigation">

 <div class="container">

 <div class="navbar-header">

 <a class="navbar-brand" href="#">项目名称</a>

 </div>

 <div class="collapse navbar-collapse">

 <ul class="nav navbar-nav">

 <li class="active"><a href="#">产品列表</a></li>

 </ul>

 </div>*<!--/.nav-collapse -->*

 </div>

</div>

<div class="container">

 <p>现在还没有任何{{'内容' + '！'}}</p>

</div>

</body>

</html>

**这些代码都做了什么事情呢？**

指令ng-app：

<html ng-app>

ng-app标签代表 Angular 中名叫ngApp的指令 （Angular 使用连接符-来定义它的标签，如ng-app， 与其相对应的规则为驼峰命名，如ngApp），这里表示我们的应用程序是作用于根元素 html下，对开发者而言可以将 angular 应用于整个页面中， 也可以作为你的应用的一部分，例如用到了其他的框架。

样式标签：

<link rel="stylesheet" href="lib/bootstrap/css/bootstrap.css">

表示我们的应用是基于 [bootstrap](http://getbootstrap.com/) 的，我们增加了一个基本的框架， 可以通过 [案例](http://v3.bootcss.com/getting-started/#examples) 查看相对应的例子。

脚本标签：

<script src="lib/angular/angular.js"></script>

引用 angular 的脚本，当脚本加载完成之后，Angular 会寻找npApp指令， 假如找到这个标签，Angular 会为项目解析该标签下的元素，并启用应用程序。

# 二、数据模板绑定

绑定表达式的双括号行：

现在还没有任何{{'内容' + '！'}}

这一行演示了 Angular 的核心功能——数据模板绑定，通过{{}}来绑定我们的内容。 绑定标签告诉 Angular 运行括号内的表达式并插入到相对应的 DOM 中，当表达式进行改变的时候， DOM 中的内容会相对应的作出改变，这就是所谓的实时动态绑定。

这里的表达式可以是一个变量，也可以是一个运算符，在插入数据之前， Angular 会先解析并运行想对应的表达式。

**启用 AngularJS 应用程序**

在上面的内容中，我们知道，使用ngApp标签可以自动启用我们的程序。 当然，你也可以通过[手动](http://docs.angularjs.org/guide/bootstrap)来启用 Angular。

在启用的过程中，Angular 主要做了三件事情：

* [injector](http://docs.angularjs.org/api/auto/service/%24injector) 会被自动创建用于依赖注入
* 接下来 injector 会创建根变量root scope作为应用程序的数据模块
* Angular 会从ngApp根元素开始，“编译”解析，处理所有的指令和绑定。

在本小节中，我们的应用程序很简单，我们只是添加了简单的静态绑定，数据模型是空的。



**扩展练习**

试着添加其他的绑定元素，并查看结果，例如：

<p>1 + 2 = {{ 1 + 2 }}</p>

# 三、Bootstrap静态页面

在这一小节，为了说明 Angular 是如何增强标准 HTML 的，这一节我们来创建静态的 HTML 页面， 顺便也会用到 bootstrap 相关的组件，然后下一节我们将使用 Angular，通过创建模板动态生成和显示相同的页面。

在开始之前，我们先来看看 [演示](http://wenzhixin.net.cn/demos/angular-scutech/step1)， 也可以通过链接下载 [zip](http://wenzhixin.net.cn/demos/angular-scutech/step1.zip) 包。

页面现在显示鼎甲科技的6个产品列表以及简单的产品介绍。

文件index.html

<div class="container">

 <ul class="list-group">

 <li class="list-group-item">

 <h4 class="list-group-item-heading">鼎甲信息仓（Infokist）</h4>

 <p class="list-group-item-text">

 一款将自主研发的一系列数据软件与云存储硬件深度整合为一体的数据保护系统。

 </p>

 </li>

 <li class="list-group-item">

 <h4 class="list-group-item-heading">鼎甲迪备（DBackup）</h4>

 <p class="list-group-item-text">

 一款分布式运维级数据库存储备份管理软件，广泛支持主流的操作系统平台以及数据库类型。

 </p>

 </li>

 <li class="list-group-item">

 <h4 class="list-group-item-heading">鼎甲安备（AnBay）</h4>

 <p class="list-group-item-text">

 一款为文件系统提供全面的SaaS（Software-as-a-Service）数据保护的“云备份”产品。

 </p>

 </li>

 <li class="list-group-item">

 <h4 class="list-group-item-heading">鼎甲奥备（OBackup）</h4>

 <p class="list-group-item-text">

 一款通过简单的图形化界面操作就可迅速完成的Linux操作系统备份和裸机恢复软件。

 </p>

 </li>

 <li class="list-group-item">

 <h4 class="list-group-item-heading">鼎甲智备（WBackup）</h4>

 <p class="list-group-item-text">

 一款基于云存储的远程异的移动终端备份软件。

 </p>

 </li>

 <li class="list-group-item">

 <h4 class="list-group-item-heading">鼎甲数据卫士（GBackup）</h4>

 <p class="list-group-item-text">

 一款专门用于Windows系统的各种类型文件、文件夹以及程序的单机备份和恢复的软件。

 </p>

 </li>

 </ul>

</div>

其中，list-group是bootstap 提供的列表组件（也提供了其他的一些组件）， 详细见[列表](http://v3.bootcss.com/components/#list-group)， 通过增加list-group-item、list-group-item-heading、list-group-item-text 这些 class 类来显示我们的列表。

**扩展练习**

添加静态内容到index.html中，例如：

<p>产品个数：6</p>

# 四、MVC代码结构

在这一小节，我们使用 Angular 来动态生成上一节的页面。

在开始之前，我们先来看看 [演示](http://wenzhixin.net.cn/demos/angular-scutech/step2)， 也可以通过链接下载 [zip](http://wenzhixin.net.cn/demos/angular-scutech/step2.zip) 包。

一个应用可以有很多种代码结构，对于 Angular 来说，我们推荐使用 [Model–View–Controller（MVC）](http://en.wikipedia.org/wiki/Model%E2%80%93View%E2%80%93Controller) 来组织代码结构。现在，让我们使用 Angular 和 JavaScript 代码来为我们的应用程序添加 model（模型）、view（视图）和 controller（控制器）。

查看演示，可以看到跟上一节我们使用静态页面一样的页面。

**View（视图）和 Template（模板）**

在 Angular 中，View（视图）是通过 HTML 的 Template（模板）来显示 Model（模型）的。 也就是说，无论在任何时候，模型发生了变化，Angular 会刷新相对应的绑定节点， 从而更新视图。

文件index.html：

**<!doctype html>**

<html lang="en" ng-app="scutechApp">

<head>

 <meta charset="utf-8">

 <title>Angular Scutech</title>

 <link rel="stylesheet" href="lib/bootstrap/css/bootstrap.css">

 <script src="lib/angular/angular.js"></script>

 <script src="js/controllers.js"></script>

</head>

<body ng-controller="ListCtrl">

<div class="navbar navbar-inverse" role="navigation">

 <div class="container">

 <div class="navbar-header">

 <a class="navbar-brand" href="#">项目名称</a>

 </div>

 <div class="collapse navbar-collapse">

 <ul class="nav navbar-nav">

 <li class="active"><a href="#">产品列表</a></li>

 </ul>

 </div>*<!--/.nav-collapse -->*

 </div>

</div>

<div class="container">

 <ul class="list-group">

 <li class="list-group-item" ng-repeat="product in products">

 <h4 class="list-group-item-heading">{{product.name}}</h4>

 <p class="list-group-item-text">{{product.info}}</p>

 </li>

 </ul>

</div>

</body>

</html>

这里我们使用ng-repeat指令来代替产品列表，然后使用两个表达式：{{product.name}} 和{{product.info}}来绑定我们的产品信息。

<li ng-repeat="product in products">

li 标签中使用了重复器，告诉 Angular 去对每个 product 创建一个 li 标签，使用第一个 li 标签的模板。

我们新增一个新的指令，叫做ng-controller，并且设置为 ListCtrl，这里对应模型中的控制器。

**Model（模型） 和 Controller（控制器）**

控制器ListCtrl现在加入了数据模型（一个简单的产品列表数组），这里的控制器只是简单的拥有 $scope 属性的一个方法：

文件js/controllers.js：

**var** scutechApp = angular.module('scutechApp', []);

scutechApp.controller('ListCtrl', **function** ($scope) {

 $scope.products = [

 {

 'name': '鼎甲信息仓（Infokist）',

 'info': '一款将自主研发的一系列数据软件与云存储硬件深度整合为一体的数据保护系统。'

 },

 {

 'name': '鼎甲迪备（DBackup）',

 'info': '一款分布式运维级数据库存储备份管理软件，广泛支持主流的操作系统平台以及数据库类型。'

 },

 {

 'name': '鼎甲安备（AnBay）',

 'info': '一款为文件系统提供全面的SaaS（Software-as-a-Service）数据保护的“云备份”产品。'

 },

 {

 'name': '鼎甲奥备（OBackup）',

 'info': '一款通过简单的图形化界面操作就可迅速完成的Linux操作系统备份和裸机恢复软件。'

 },

 {

 'name': '鼎甲智备（WBackup）',

 'info': '一款基于云存储的远程异的移动终端备份软件。'

 },

 {

 'name': '鼎甲数据卫士（GBackup）',

 'info': '一款专门用于Windows系统的各种类型文件、文件夹以及程序的单机备份和恢复的软件。'

 }

 ];

});

我们定义了ListCtrl控制器，并且注册到 Angular 的 module：scutechApp中， 注意到我们在上面的 html 中设置了ng-app="scutechApp"，与这里的 module 相对应， 从而让 Angular 知道如何启用我们的应用程序。

即使控制器没有做任何事情，但它起了决定性的作用。通过提供我们的模型数据， 控制器允许我们建立模型和视图之间的双向绑定。我们可以看到：

* 指令ngController是位于body标签中的，与之相对应的ListCtrl是位于controllers.js中， 表示我们的控制器范围是作用于 body 下的。
* ListCtrl控制器将数据作为$scope的一个属性值products。这里的 scope 继承于 root scope， 当应用程序启动的时候自动被创建。

**Scope**

在 Angular 中，Scope 的概念是至关重要的，Scope 将视图、模板、模型和控制器紧紧结合在了一起。 同时，使用 Scope 让视图、模型和控制器起到了解藕的效果，但是又做到了同步。 即数据模型与视图的双向绑定，所以任何数据的改变都会在视图中显示出来，反之亦然。

**扩展练习**

在index.html中添加新的绑定：

<p>产品个数：{{products.length}}</p>

使用表格显示我们的产品：

<table class="table table-bordered">

 <tr>

 <th>编号</th>

 <th>名称</th>

 <th>描述</th>

 </tr>

 <tr ng-repeat="product in products">

 <td>{{$index + 1}}</td>

 <td>{{product.name}}</td>

 <td>{{product.info}}</td>

 </tr>

</table>

# 五、搜索编辑框

在这一小节，我们增加一个简单的input输入框，用于搜索。

在开始之前，我们先来看看 [演示](http://wenzhixin.net.cn/demos/angular-scutech/step3)， 也可以通过链接下载 [zip](http://wenzhixin.net.cn/demos/angular-scutech/step3.zip) 包。

**控制器**

我们没有改变控制器的任何代码。

**模板**

文件：index.html

<div class="container">

 <div class="col-md-3">

 <input type="text" class="form-control" placeholder="搜索" ng-model="query" />

 </div>

 <ul class="col-md-9 list-group">

 <li class="list-group-item" ng-repeat="product in products | filter:query">

 <h4 class="list-group-item-heading">{{product.name}}</h4>

 <p class="list-group-item-text">{{product.info}}</p>

 </li>

 </ul>

</div>

我们修改我们的模板代码，col-md-3、col-md-9 都是 bootstrap 提供的类，表示将搜索栏和列表分别放在左右两边，form-control 将 input 设置为 bootstrap 的风格。

看到为 input 输入框增加了ng-model指令，表示我们的模型是query， 而在ng-repeat中加入了 filter 函数用于处理输入框的模型。

当你在输入框中输入内容的时候会立即看到列表的变化。

这些新代码表明了：

* 数据绑定：这是 AngularJS 的一个核心功能，当页面加载后，AngularJS 在模型中绑定了相同的名字的变量， 从而保证视图和模型之间的同步。
* 使用filter函数过滤内容：filter函数使用query参数去创建新的只包含输入框内容的列表。 从而自动更新列表的内容，达到搜索的功能。

**扩展练习**

增加一个绑定显示当前的搜索内容：

<p>输入的内容：{{query}}</p>

假如我想把搜索内容显示在页面的标题上呢？

是不是通过设置：

<title>Angular Scutech：{{query}}</title>

就可以了呢？通过实践我们可以知道，并不能达到我们想要的结果，这是为什么呢？

因为我们的控制器是放在 body 标签上的，这样使用 query 变量的话就超出我们的作用域了：

<body ng-controller="ListCtrl">

我们可以把控制移动到 html 标签上，就可以了：

<html lang="en" ng-app="scutechApp" ng-controller="ListCtrl">

但是这里有个小问题，就是使用{{}}的时候在最初加载的时候，显示为Angular Scutech：{{query}}，很不美观。

在 AngularJS 中，我们可以使用ngBind（用于绑定变量，例如ng-bind="query"） 或者ngBindTemplate（用于绑定模块，例如ng-bind-template="Angular Scutech：{{query}}"） 来代替双括号进行数据绑定，例如我们用下面的代码就可以了：

<title ng-bind="Angular Scutech：{{query}}">Angular Scutech</title>

到这里，你是不是对 AngularJS 的数据绑定有所理解了呢:-)

# 六、列表排序

在这一小节，我们增加一个功能可以用来对列表进行排序。

在开始之前，我们先来看看 [演示](http://wenzhixin.net.cn/demos/angular-scutech/step4)， 也可以通过链接下载 [zip](http://wenzhixin.net.cn/demos/angular-scutech/step4.zip) 包。

可以看到在搜索框下面，我们用一个下拉列表来显示排序的信息。

**模板**

文件index.html：

<div class="container">

 <div class="col-md-3">

 <div class="form-group">

 <input type="text" class="form-control" placeholder="搜索" ng-model="query" />

 </div>

 <div class="form-group">

 <select class="form-control" ng-model="order">

 <option value="name">名称</option>

 <option value="age">发布时长</option>

 </select>

 </div>

 </div>

 <ul class="col-md-9 list-group">

 <li class="list-group-item" ng-repeat="product in products | filter:query | orderBy:order">

 <h4 class="list-group-item-heading">{{product.name}} - {{product.age}}年</h4>

 <p class="list-group-item-text">{{product.info}}</p>

 </li>

 </ul>

</div>

在这个页面中，我们做了一些修改：

* 首先，我们增加了模型名称为order的select下拉列表，可以选择按照不同的条件进行排序。
* 我们在过滤器filter的后面增加了orderBy函数，从而更进一步去处理列表。orderBy 是一个接受数组，并对数组进行拷贝排序后返回想要的数组。

Angular 创建了一个从选择框到模型order之间的双向绑定，并且作为orderBy函数的参数。

**控制器**

文件js/controllers.js：

**var** scutechApp = angular.module('scutechApp', []);

scutechApp.controller('ListCtrl', **function** ($scope) {

 $scope.products = [

 {

 'name': '鼎甲信息仓（Infokist）',

 'info': '一款将自主研发的一系列数据软件与云存储硬件深度整合为一体的数据保护系统。',

 'age': 2

 },

 {

 'name': '鼎甲迪备（DBackup）',

 'info': '一款分布式运维级数据库存储备份管理软件，广泛支持主流的操作系统平台以及数据库类型。',

 'age': 4

 },

 {

 'name': '鼎甲安备（AnBay）',

 'info': '一款为文件系统提供全面的SaaS（Software-as-a-Service）数据保护的“云备份”产品。',

 'age': 3.5

 },

 {

 'name': '鼎甲奥备（OBackup）',

 'info': '一款通过简单的图形化界面操作就可迅速完成的Linux操作系统备份和裸机恢复软件。',

 'age': 3

 },

 {

 'name': '鼎甲智备（WBackup）',

 'info': '一款基于云存储的远程异的移动终端备份软件。',

 'age': 1.5

 },

 {

 'name': '鼎甲数据卫士（GBackup）',

 'info': '一款专门用于Windows系统的各种类型文件、文件夹以及程序的单机备份和恢复的软件。',

 'age': 1

 }

 ];

 $scope.order = 'age';

});

我们修改了products列表的数据，为每条记录增加了age属性，用于排序，对应于上面的：

<option value="age">发布时长</option>

然后增加了：

$scope.order = 'age';

表示我们将 age 作为默认的排序值，假如没有为它设置值，下拉列表将会为空，这个可以在扩展练习中尝试下。

这里涉及到了一个很重要的双向绑定的概念。可以看到，当页面加载完成的时候，下拉列表会自动选择“发布时长”， 这是因为我们在控制器中将order默认设置为age，这里就是从模型（model）到视图（view）的绑定。 而当你改变下拉列表的值（名称）后，模型（model）将会相对应的更新为name，列表也会进行重新排序， 这是从视图（view）到模型（model）的绑定。

**拓展练习**

删除控制器中的：

$scope.order = 'age';

通过F12，可以看到在下拉列表增加了：

<option value="? undefined:undefined ?"></option>

假如要改变排序的顺序，可以通过加上-号，不加的话相当于“asc”，加了就变成了“desc”：

$scope.order = '-age';

<option value="-age">发布时长</option>

# 七、动态获取后台数据

在之前几个小节中，我们的列表都是使用固定的数据。 在这一小节，我们使用 $http 请求从后台获取数据。

在开始之前，我们先来看看 [演示](http://wenzhixin.net.cn/demos/angular-scutech/step5)， 也可以通过链接下载 [zip](http://wenzhixin.net.cn/demos/angular-scutech/step5.zip) 包。

**数据**

我们在当前文件夹下增加了 data 文件夹，并将我们的列表保存为 JSON 格式放到data/products.json文件中：

[

 {

 "name": "鼎甲信息仓（Infokist）",

 "info": "一款将自主研发的一系列数据软件与云存储硬件深度整合为一体的数据保护系统。",

 "age": 2

 },

 {

 "name": "鼎甲迪备（DBackup）",

 "info": "一款分布式运维级数据库存储备份管理软件，广泛支持主流的操作系统平台以及数据库类型。",

 "age": 4

 },

 {

 "name": "鼎甲安备（AnBay）",

 "info": "一款为文件系统提供全面的SaaS（Software-as-a-Service）数据保护的“云备份”产品。",

 "age": 3.5

 },

 {

 "name": "鼎甲奥备（OBackup）",

 "info": "一款通过简单的图形化界面操作就可迅速完成的Linux操作系统备份和裸机恢复软件。",

 "age": 3

 },

 {

 "name": "鼎甲智备（WBackup）",

 "info": "一款基于云存储的远程异的移动终端备份软件。",

 "age": 1.5

 },

 {

 "name": "鼎甲数据卫士（GBackup）",

 "info": "一款专门用于Windows系统的各种类型文件、文件夹以及程序的单机备份和恢复的软件。",

 "age": 1

 }

]

**控制器**

我们 Angular 提供的使用 [$http](http://docs.angularjs.org/api/ng/service/%24http) 服务来发送 http 请求， 类似于 jQuery 的 ajax 请求，例如 $http.get 相当于 jQuery 的 $.get，$http.post 相当于 jQuery 的 $.post。

文件js/controllers.js：

**var** scutechApp = angular.module('scutechApp', []);

scutechApp.controller('ListCtrl', **function** ($scope, $http) {

 $http.get('data/products.json').success(function(data) {

 $scope.products = data;

 });

 $scope.order = 'age';

});

可以看到，我们通过 http 的 GET 方法，发送请求到地址为 data/products.json（这里相当于我们上面新建的文件， 跟在服务器去获取是一样的道理，但是涉及到跨域的问题，所以这里用本地文件作为演示），服务器返回了一个 JSON 格式的数据，即我们的列表数据。

$http 服务返回了一个 [promise](http://docs.angularjs.org/api/ng/service/%24q) 对象，这个 promise 对象有一个success的方法，我们使用这个方法来处理异步返回的数据，并且将返回的数据绑定到$scope.products 变量中，从而通知我们的视图做出相对应的更新。

$http 服务的使用和 $scope 一样，我们只需要通过注入的方式即可，使用起来很简单。

scutechApp.controller('ListCtrl', **function** ($scope, $http) {...}

Angular 的依赖注入提供了一系列的服务给我们使用，例如后面会讲到的 $resource、$animate也是一样的道理。 需要注意的是，参数的名称必须是有意义的，因为 Angular 就是通过名字进行依赖查找的。

**$前缀的命名规范**

我们可以创建属于自己的服务，后面应该会提到。根据 Angular 的命名规范，提供的服务都是以 $ 为前缀进行命名的， 例如 $scope、$http、$resource 等等。

也就是说，$前缀在 Angular 中代表提供的服务的命名空间，为了避免与 Angular 提供的服务进行冲突， 我们不建议使用 $前缀进行自定义服务的命名。

**极简化改进**

上面提到，Angular 是通过服务名进行查找，然后依赖注入的，但是这样子就相当于将所有的参数都传递进来了。 这里涉及到一个概念，就是 [极简化](http://zh.wikipedia.org/wiki/%E6%A5%B5%E7%B0%A1%E5%8C%96)，意思是说， 我们只需要传递我们需要的参数就可以了，这样就可以优化我们的程序。

那么，我们应该如何改进我们的程序呢？

有两种方法：

* 可以为控制器方法创建一个$inject属性，这个属性是一个保存我们所需要注入的服务的名称，例如：

funciton ListCtrl($scope, $http) {...}

ListCtrl.$inject = ['$scope', '$http'];

scutechApp.controller('ListCtrl', ListCtrl);

* 使用单行括号来声明我们需要的服务，然后最后一个参数是我们的控制器方法：

funciton ListCtrl($scope, $http) {...}

scutechApp.controller('ListCtrl', ['$scope', '$http', ListCtrl]);

在实际使用中，我们会使用第二种方法，并且写成了一行，改进后的代码为：

**var** scutechApp = angular.module('scutechApp', []);

scutechApp.controller('ListCtrl', ['$scope', '$http',

 **function** ($scope, $http) {

 $http.get('data/products.json').success(function(data) {

 $scope.products = data;

 });

 $scope.order = 'age';

 }

]);

**扩展练习**

在index.html的最后面，我们可以通过增加{{products | json}}来查看我们的列表信息（JSON格式）。

在控制器js/controllers.js中，我们可以通过预处理，只显示两条记录（实际项目中可能会对列表做一些处理）：

$scope.products = data.**splice**(0, 2);

# 八、添加图片和链接

在这一小节，我们为列表中的产品添加图片和链接，下一小节将使用链接跳转到详细的页面。

在开始之前，我们先来看看 [演示](http://wenzhixin.net.cn/demos/angular-scutech/step6)， 也可以通过链接下载 [zip](http://wenzhixin.net.cn/demos/angular-scutech/step6.zip) 包。

现在可以看到产品图片和链接了，现在点击链接还没有任何的反应，因为我们还没进行任何的处理。

**数据**

我们增加了 images 文件夹用于存放我们的图片，并且为产品信息添加了id和image 两个属性，可以看到文件data/products.json（这里只列举了一条记录， 图片路径对应我们的实际的图片文件）：

[

 {

 "id": 1,

 "name": "鼎甲信息仓（Infokist）",

 "info": "一款将自主研发的一系列数据软件与云存储硬件深度整合为一体的数据保护系统。",

 "age": 2,

 "image": "images/infokist.png"

 },

 ...

]

**模板**

文件index.html：

<ul class="col-md-9 list-group">

 <li class="list-group-item" ng-repeat="product in products | filter:query | orderBy:order">

 <a href="#/product/{{product.id}}">

 <h4 class="list-group-item-heading">

 <img ng-src="{{product.image}}" />

 {{product.name}} - {{product.age}}年

 </h4>

 </a>

 <p class="list-group-item-text">

 {{product.info}}

 </p>

 </li>

</ul>

我们增加了超链接并且设置 href 为#/product/{{product.id}}，以 id 作为我们的标志， 用于显示更详细的产品内容。

然后我们增加了图片标签，并且设置了 Angular 的ngSrc指令。这里你可能会有疑问，为什么不能用 src 属性呢？ 需要注意的是，假如我们使用了 src 属性，那么浏览器在加载的时候会发送请求 到http://localohost/images/{{product.image}}，使用 F12 可以看到会导致 404 Not Found 的错误， 所以 Angular 提供了ng-src属性来避免这种错误。

**扩展练习**

将ng-src属性替换为原来的src属性，然后通过 F12（Chrome 的审查元素或者 Firefox 的 Firebug） 查看，会发现错误信息，这里可以加深理解 Angular 的工作原理。