# Javascript教程：AngularJS的五个超酷特性

日期：2012-7-17　　来源：GBin1.com


AngularJS是一个超棒的javascript框架，不单单对于开发人员来说非常有吸引力，对于UI设计师来说也同样出色。在这篇教程中，我们将简单的介绍AngularJS几个重量级必备特性，并且介绍它如何能够让你的web应用更加强大！

## AugularJS简单介绍

> AngularJS是一个新出现的强大客户端技术，提供给大家的一种开发强大应用的方式。这种方式利用并且扩展HTML，CSS和javascript，并且弥补了它们的一些非常明显的不足。本应该使用HTML来实现而现在由它开发的动态一些内容。

在这篇文章中，我们讲述了一些最重要的AngularJS功能和特性。我们的目标在于阅读后，你可以开始自己开发一些有趣的应用。

## 特性一：双向的数据绑定

数据绑定可能是AngularJS最酷最实用的特性。它能够帮助你避免书写大量的初始代码从而节约开发时间。一个典型的web应用可能包含了80%的代码用来处理，查询和监听DOM。数据绑定是的代码更少，你可以专注于你的应用。

我们想象一下Model是你的应用中的简单事实。你的Model是你用来读取或者更新的部分。数据绑定指令提供了你的Model投射到view的方法。这些投射可以无缝的，毫不影响的应用到web应用中。

传统来说，当model变化了。 开发人员需要手动处理DOM元素并且将属性反映到这些变化中。这个一个双向的过程。一方面，model变化驱动了DOM中元素变化，另一方面，DOM元素的变化也会影响到Model。这个在用户互动中更加复杂，因为开发人员需要处理和解析这些互动，然后融合到一个model中，并且更新View。这是一个手动的复杂过程，当一个应用非常庞大的时候，将会是一件非常费劲的事情。

这里肯定有更好的解决方案！那就是AngularJS的双向数据绑定，能够同步DOM和Model等等。

这里有一个非常简单的例子，用来演示一个input输入框和<h1>元素的双向绑定：

```
<!doctype html>
```

```
<html ng-app>
  <head>
    <script src="http://code.angularjs.org/angular-1.0.0rc10.min.js"></script>
  </head>
  <body>
    <div>
      <label>Name:</label>
      <input type="text" ng-model="yourName" placeholder="Enter a name here">
      <hr>
      <h1>Hello, {{yourName}}!</h1>
    </div>
  </body>
</html>
```

## 特性二：模板

在AngularJS中，一个模板就是一个HTML文件。但是HTML的内容扩展了，包含了很多帮助你映射model到view的内容。

HTML模板将会被浏览器解析到DOM中。DOM然后成为AngularJS编译器的输入。AngularJS将会遍历DOM模板来生成一些指导，即，directive（指令）。所有的指令都负责针对view来设置数据绑定。

我们要理解AuguarJS并不把模板当做String来操作。输入AngularJS的是DOM而非string。数据绑定是DOM变化，不是字符串的连接或者innerHTML变化。使用DOM作为输入，而不是字符串，是AngularJS区别于其它的框架的最大原因。使用DOM允许你扩展指令词汇并且可以创建你自己的指令，甚至开发可重用的组件。

最大的好处是为设计师和开发者创建了一个紧密的工作流。设计师可以像往常一样开发标签，然后开发者拿过来添加上功能，通过数据绑定将会使得这个过程非常简单。

这里有一个例子，我们使用ng-repeat指令来循环图片数组并且加入img模板，如下：

```
function AlbumCtrl($scope) {
    scope.images = [
        {"image":"img/image_01.png", "description":"Image 01 description"},
        {"image":"img/image_02.png", "description":"Image 02 description"},
        {"image":"img/image_03.png", "description":"Image 03 description"},
        {"image":"img/image_04.png", "description":"Image 04 description"},
        {"image":"img/image_05.png", "description":"Image 05 description"}
    ];
}

<div ng-controller="AlbumCtrl">
  <ul>
    <li ng-repeat="image in images">
      <img ng-src="{{image.thumbnail}}" alt="{{image.description}}">
    </li>
  </ul>
```

```
</div>
```

这里还有一件事值得提一句，AngularJS并不强制你学习一个新的语法或者从你的应用中提出你的模板。

## 特性三：MVC

针对客户端应用开发AngularJS吸收了传统的MVC基本原则。MVC或者Model-View-Controll设计模式针对不同的人可能意味不同的东西。AngularJS并不执行传统意义上的MVC，更接近于MVVM（Moodel-View-ViewModel)。

### Model

model是应用中的简单数据。一般是简单的javascript对象。这里没有必要继承框架的classes，使用proxy对象封装或者使用特别的setter/getter方法来访问。事实上我们处理vanilla javascript的方法就是一个非常好的特性，这种方法使得我们更少使用应用的原型。

### ViewModel

viewmodel是一个用来提供特别数据和方法从而维护指定view的对象。

viewmodel是$scope的对象，只存在于AnguarJS的应用中。$scope只是一个简单的js对象，这个对象使用简单的API来侦测和广播状态变化。

### Controller

controller负责设置初始状态和参数化$scope方法用以控制行为。需要指出的controller并不保存状态也不和远程服务互动。

### View

view是AngularJS解析后渲染和绑定后生成的HTML 。这个部分帮助你创建web应用的架构。$scope拥有一个针对数据的参考，controller定义行为，view处理布局和互动。

## 特性四：依赖注入（Dependency Injection，即DI）

AngularJS拥有内建的依赖注入子系统，可以帮助开发人员更容易的开发，理解和测试应用。

DI允许你请求你的依赖，而不是自己找寻它们。比如，我们需要一个东西，DI负责找创建并且提供给我们。

为了而得到核心的AngularJS服务，只需要添加一个简单服务作为参数，AngularJS会侦测并且提供给你：

```
function EditCtrl($scope, $location, $routeParams) {
    // Something clever here...
}
```

你也可以定义自己的服务并且让它们注入：

```
angular.
    module('MyServiceModule', []).
```

```
    factory('notify', ['$window', function (win) {
    return function (msg) {
        win.alert(msg);
    };
}]);

function myController(scope, notifyService) {
    scope.callNotify = function (msg) {
        notifyService(msg);
    };
}

myController.$inject = ['$scope', 'notify'];
```

## 特性五：Directives（指令）

指令是我个人最喜欢的特性。你是不是也希望浏览器可以做点儿有意思的事情？那么AngularJS可以做到。

指令可以用来创建自定义的标签。它们可以用来装饰元素或者操作DOM属性。

这里是一个例子，它监听一个事件并且针对的更新它的$scope ，如下：

```
myModule.directive('myComponent', function(mySharedService) {
    return {
        restrict: 'E',
        controller: function($scope, $attrs, mySharedService) {
            $scope.$on('handleBroadcast', function() {
                $scope.message = 'Directive: ' + mySharedService.message;
            });
        },
        replace: true,
        template: '<input>'
    };
});
```

然后，你可以使用这个自定义的directive来使用：

```
<my-component ng-model="message"></my-component>
```

使用一系列的组件来创建你自己的应用将会让你更方便的添加，删除和更新功能。

## 额外的特性：测试

AngularJS内含了测试用例可以帮助你更方便的执行测试。为什么不用呢？

JS是一个动态的解析性语言，而不是编译类型的，因此非常的难写测试。

AngularJS被开成一个可测试的框架。它甚至包含了点对点的单元测试runner。如果你喜欢这个特性，看看这个项目：https://github.com/angular/angular-seed

一但你运行这个项目，你可以看到如下输出：


API文档是完整的点对点测试，说明了整个架构师如何工作的。通过查看这些测试，你会对AngularJS有更深刻的了解。

## 总结

在这篇教程中，我们总结了6个AngularJS的关键特性。如果大家对于AngularJS有兴趣，请访问 http://angularjs.org

我强烈建议大家加入maillist，你肯定能找到很多的有用信息。