# Angular基础学习

### 简介

AngularJS 是一个为动态WEB应用设计的结构框架，提供给大家一种新的开发应用方式，这种方式可以让你扩展HTML的语法，以弥补在构建动态WEB应用时静态文本的不足，从而在web应用程序中使用HTML声明动态内容。

AngularJS有五个主要核心特性，如下介绍：

****双向数据绑定**** —— 实现了把model与view完全绑定在一起，model变化，view也变化，反之亦然。

****模板**** —— 在AngularJS中，模板相当于HTML文件被浏览器解析到DOM中，AngularJS遍历这些DOM，也就是说AuguarJS把模板当做DOM来操作，去生成一些指令来完成对view的数据绑定。

****MVVM**** —— 吸收了传统的MVC设计模式针但又并不执行传统意义上的MVC，更接近于MVVM(Moodel-View-ViewModel)。

****依赖注入**** —— 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。

****指令**** —— 可以用来创建自定义的标签，也可以用来装饰元素或者操作DOM属性。

认识了AngularJS框架，我们开始创建第一个AngularJS应用。

AngularJS是以一个JavaScript文件形式发布的，可通过script标签载入AngularJS脚本，如下所示：

1. <script src="http://www.hubwiz.com/javascripts/angular.min.js"></script>

复制以下代码至右边栏，做好准备工作。

1. <div ng-app="" ng-init="name='World'">
2. Hello {{ name }}！
3. </div>

点击运行结果查看是否为“Hello World”，如正确则表示AngularJS脚本成功引入。

### 指令介绍

AngularJS有一套完整的、可扩展的、用来帮助web应用开发的指令集，它使得HTML可以转变成“特定领域语言(DSL)”，是用来扩展浏览器能力的技术之一，在DOM编译期间，和HTML关联着的指令会被检测到，并且被执行，这使得指令可以为DOM指定行为，或者改变它。

AngularJS通过称为指令的新属性来扩展的HTML，带有前缀ng-，我们也可以称之为“指令属性”，它就是绑定在DOM元素上的函数，可以调用方法、定义行为、绑定controller及$scope对象、操作DOM，等等。

AngularJS指令指示的是“当关联的HTML结构进入编译阶段时应该执行的操作”，它本质上只是一个当编译器编译到相关DOM时需要执行的函数，可以写在元素的名称里，属性里，css类名里，注释里。

当浏览器启动、开始解析HTML时，DOM元素上的指令属性就会跟其他属性一样被解析，也就是说当一个Angular.js应用启动，Angular编译器就会遍历DOM树来解析HTML，寻找这些指令属性函数，在一个DOM元素上找到一个或多个这样的指令属性函数，它们就会被收集起来、排序，然后按照优先级顺序被执行。

Angular.js应用的动态性和响应能力，都要归功于指令属性，常见的有：ng-app、ng-init、ng-model、ng-bind、ng-repeat等等。

关于“指令属性”就先暂时介绍这些，相信你已经对它有初步的了解，接下来我们就开始逐一的深入的学习它们。

### 指令：ng-app

ng-app指令来标明一个AngularJS应用程序，并通过AngularJS完成自动初始化应用和标记应用根作用域，同时载入和指令内容相关的模块，并通过拥有ng-app指令的标签为根节点开始编译其中的DOM。

引用方法很简单，如下所示：

1. <div ng-app="">
3. </div>

如上引用，一个AngularJS应用程序初始化就完成了并标记了作用域，也就是div元素就是AngularJS应用程序的"所有者"，在它里面的指令也就会被Angular编译器所编译、解析了。

### 指令：ng-init

ng-init指令初始化应用程序数据，也就是为AngularJS应用程序定义初始值，通常情况下，我们会使用一个控制器或模块来代替它，后面我们会介绍有关控制器和模块的知识。

如下所示，我们为应用程序变量name赋定初始值。

1. <div ng-app="" ng-init="name='Hello World'">
3. </div>

我们不仅可以赋值字符串，也可以赋值为数字、数组、对象，而且可以为多个变量赋初始值，如下所示：

1. <div ng-app="" ng-init="quantity=1;price=5">
3. </div>
4. //或者
5. <div ng-app="" ng-init="names=['Tom','Jerry','Gaffey']">
7. </div>

后面我们还会学习使用控制器来初始化数据的方式，接着我们来学习如何来调用这些已经初始化了的值。

### AngularJS表达式

AngularJS框架的核心功能之一 —— 数据绑定，由两个花括号{{}}组成，可以把数据绑定到HTML，类似Javascript代码片段，可以包含文字、运算符和变量，通常在绑定数据中用到，表达式可以绑定数字、字符串、对象、数组，写在双大括号内：{{ expression }}。

1. 如前面的示例，我们就可以使用表达式这样调用初始化的变量值，如下。
2. <div ng-app="" ng-init="name='Hello World'">
3. {{ name }}
4. </div>
5. 当然我们也可以使用表达式输出数字、数组等等，如下所示：

2.1. 输出数字，如下示例：

1. <div ng-app="" ng-init="quantity=12;price=5">
3. 总价： {{ quantity \* price }}
5. </div>

2.2. 输出对象，如下示例：

1. <div ng-app="" ng-init="names=['Tom','Jerry','Gaffey']">
3. 名字为： {{ names[0] }}
5. </div>

后面我们还会学习另一种数据绑定的方式 —— 通过指令ng-bind来实现

### 指令：ng-model

在AngularJS中，只需要使用ng-model指令就可以把应用程序数据绑定到HTML元素，实现model和view的双向绑定。

如下示例，使用ng-model指令对数据进行绑定。

1. <div ng-app="">
3. 请输入任意值：<input type="text" ng-model="name" />
5. 你输入的为： {{ name }}
7. </div>

ng-model把相关处理事件绑定到指定标签上，这样我们就可以不用在手工处理相关事件(比如change等)的条件下完成对数据的展现需求。

### 指令：ng-bind

指令ng-bind和AngularJS表达式{{}}有异曲同工之妙，但不同之处就在于ng-bind是在angular解析渲染完毕后才将数据显示出来的。

如下使用ng-bind指令绑定把应用程序数据。

1. <div ng-app="">
2. 请输入一个名字：<input type="text" ng-model="name" />
3. Hello <span ng-bind="name"></span>
4. </div>

PS：使用花括号语法时，因为浏览器需要首先加载页面，渲染它，然后AngularJS才能把它解析成你期望看到的内容，所以对于首个页面中的数据绑定操作，建议采用ng-bind，以避免其未被渲染的模板被用户看到。

### 指令：ng-click

AngularJS也有自己的HTML事件指令,比如说通过ng-click定义一个AngularJS单击事件。

对按钮、链接等，我们都可以用ng-click指令属性来实现绑定，如下简单示例：

1. <div ng-app="" ng-init="click=false">
2. <button ng-click="click= !click">隐藏/显示</button>
3. <div ng-hide="click">
4. 请输入一个名字：<input type="text" ng-model="name" />
5. Hello <span ng-bind="name"></span>
6. </div>
7. </div>

PS：ng-hide="true"，设置HTML元素不可见。

ng-click指令将DOM元素的鼠标点击事件(即mousedown)绑定到一个方法上，当浏览器在该DOM元素上鼠标触发点击事件时，Angular就会调用相应的方法，是不是很简单方便呢！

### MVVM简介

针对客户端应用开发AngularJS吸收了传统的MVC基本原则。MVC(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。

MVVM模式是Model-View-ViewMode模式的简称。由视图(View)、视图模型(ViewModel)、模型(Model)三部分组成，通过这三部分实现UI逻辑、呈现逻辑和状态控制、数据与业务逻辑的分离。

Model将和ViewModel互动(通过$scope对象)，将监听Model的变化。这些可以通过View来发送和渲染，由HTML来展示你的 代码。View可以通过$routeProvider对象来支配，所以你可以深度的链接和组织你的View和Controller，将他们变成导航 URL。AngualrJS同时提供了无状态的Controller，可以用来初始化和控制$scope对象。

Model与MVC模式一样，Model用于封装与应用程序的业务逻辑相关的数据以及对数据的处理方法。它具有对数据直接访问的权利，例如对数据库的访问，Model不依赖于View和ViewModel，也就是说，模型不关心会被如何显示或是如何被操作，模型也不能包含任何用户使用的与界面相关的逻辑。

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。

使用MVVM模式有几大好处：

1. 低耦合：View可以独立于Model变化和修改，一个ViewModel可以绑定到不同的View上，当View变化的时候Model可以不变，当Model变化的时候View也可以不变。

可重用性：可以把一些视图的逻辑放在ViewModel里面，让很多View重用这段视图逻辑。

独立开发：开发人员可以专注与业务逻辑和数据的开发(ViewModel)。设计人员可以专注于界面(View)的设计。

可测试性：可以针对ViewModel来对界面(View)进行测试。

### 控制器

AngularJS控制器控制AngularJS应用程序的数据，是常规的JavaScript对象。

ng-controller指令就是用来定义应用程序控制器的，并且同时创建了一个新的作用域关联到相应的DOM元素上。

所谓作用域就是一个指向应用模型的对象，它是表达式的执行环境，作用域有层次结构，这个层次和相应的DOM几乎是一样的，作用域能监控表达式和传递事件并且可以从父作用域继承属性。

每一个AngularJS应用都有一个绝对的根作用域。但也可能有多个子作用域。 一个应用可以有多个作用域，因为有一些指令会生成新的子作用域，当新作用域被创建的时候，他们会被当成子作用域添加到父作用域下，这使得作用域会变成一个和相应DOM结构一个的树状结构。

### 控制器

现在我们就用ng-controller指令来创建一个简单的控制器定义，如下所示：

1. <div ng-app="" ng-controller="MyController">
3. 请输入一个名字：<input type="text" ng-model="person.name">
5. Hello <span ng-bind="person.name"></span>
7. </div>
9. <script>
10. function MyController($scope) {
11. $scope.person = {
12. name: "World"
13. };
14. }
15. </script>

如上所述，我们通过ng-controller指令创建了一个JavaScript对象 —— MyController并带有name属性，那参数$scope是什么呢，代表什么意思呢。

现在我们就来解答MyController对象参数 —— $scope。

$scope就是把一个DOM元素连结到控制器上的对象，它提供一个绑定到DOM元素(以及其子元素)上的执行上下文。它也是一个JavaScript对象，指向应用程序作用域内的所有HTML元素和执行上下文。作用域呢，就是作为$scope的数据属性关联到DOM上的，并且能在需要调试的时候被获取到。

要明确创建一个$scope对象，我们就要给DOM元素安上一个controller对象，使用的是ng-controller 指令属性。

所有$scope都遵循原型继承，这意味着它们都能访问父$scope们,对任何属性和方法，如果AngularJS在当前$scope上找不到，就会到父$scope上去找，如果在父$scope上也没找到，就会继续向上回溯，一直到$rootScope上，这个$rootScope是最顶级的$scope，它对应着含有 ng-app指令属性的那个DOM元素，也就是说根作用域关联的DOM就是ng-app指令定义的地方。

也就是说，拥有了$scope，我们就可以操作作用域内任何我们想要获取的对象数据。

控制器不仅声明属性也可以声明方法，如下所示：

1. <div ng-app="" ng-controller="MyController">
2. Your name:
3. <input type="text" ng-model="username">
4. <button ng-click="sayHello()">打招呼</button>
5. <hr />
6. {{greeting}}
7. </div>
9. <script>
10. function MyController($scope) {
11. $scope.username = 'World';
12. $scope.sayHello = function() {
13. $scope.greeting= 'Hello ' + $scope.username + '!';
14. };
15. }
16. </script>

### 常用指令

ng-repeat指令，遍历一个数据集合中的每个数据元素，并且加载HTML模版把数据渲染出来，当我们要向HTML容器节点中添加更多类似DOM元素的时候，使用ng-repeat是再好不过了。

使用方法，如下示例：

1. <div ng-app="" ng-init="friends = [
2. {name:'Tom', age:25},
3. {name:'Jerry', age:28},
4. {name:'Tom', age:25},
5. {name:'Jerry', age:28}]">
7. <table>
8. <tr ng-repeat="x in friends">
9. <td> {{ 'Name：'+ x.name +' ，Age：'+ x.age}} </td>
10. </tr>
11. </table>
12. </div>

ng-repeat指令对于集合中(数组中)的每一项都会渲染一次HTML元素。

### 过滤器

使用AngularJS过滤器可以实现对字符串的大小写转换、货币格式的转换、数组的过滤等等。

用法：管道字符(|)+过滤器名。

1.过滤器uppercase、lowercase对字符串转换大、小写，如下所示：

1. <div ng-app="">
2. 请输入: <input type="text" ng-model="name">
3. 结果为： {{ name | uppercase}}
4. </div>

使用方法很简单吧，在试试过滤器lowercase的效果吧！

1. filter过滤器可以过滤数组并从中选择出一个子集出来，用法是“filter:模型名称”，示例如下：
2. <div ng-app="" ng-init="friends = [
3. {name:'tom', age:16},
4. {name:'jerry', age:20},
5. {name:'garfield', age:22}]">
7. 输入过滤:<input type="text" ng-model="name" >
8. <ul style="list-style-type:none">
9. <li> 姓名，年龄</li>
10. <li ng-repeat="x in friends | filter:name">
11. {{ x.name + ' , ' + x.age }}
12. </li>
13. </ul>
14. </div>

看到输出结果了吧，filter过滤器是不是很强大而且用法超简单就实现了对数据的简单过滤，如果不使用它，那我们想要实现这个数据过滤效果是不是要相当麻烦(可以自己试试如何实现)，而现在只需那么简单，足以彰显AngularJS是多么的给力呀！