# 3.AngularJS 指令

AngularJS 通过被称为 **指令** 的新属性来扩展 HTML。

AngularJS 通过内置的指令来为应用添加功能。

AngularJS 允许你自定义指令。

## AngularJS 指令

AngularJS 指令是扩展的 HTML 属性，带有前缀 **ng-**。

**ng-app** 指令初始化一个 AngularJS 应用程序。

**ng-init** 指令初始化应用程序数据。

**ng-model** 指令把元素值（比如输入域的值）绑定到应用程序。

完整的指令内容可以参阅 [AngularJS 参考手册](https://www.runoob.com/angularjs/angularjs-reference.html)。

[3-1]

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body>  
  
<div ng-app="" ng-init="firstName='John'">  
  
 <p>在输入框中尝试输入:</p><p>姓名: <input type="text" ng-model="firstName"></p>  
 <p>你输入的为: {{ firstName }}</p>  
  
</div>  
  
</body>  
</html>

**ng-app** 指令告诉 AngularJS，<div> 元素是 AngularJS **应用程序** 的"所有者"。

## 数据绑定

上面实例中的 **{{ firstName }}** 表达式是一个 AngularJS 数据绑定表达式。

AngularJS 中的数据绑定，同步了 AngularJS 表达式与 AngularJS 数据。

**{{ firstName }}** 是通过 **ng-model="firstName"** 进行同步。

在下一个实例中，两个文本域是通过两个 ng-model 指令同步的：

[3-2]

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body>  
  
<div data-ng-app="" data-ng-init="quantity=1;price=5">  
 <h2>价格计算器</h2>  
 数量: <input type="number" ng-model="quantity">  
 价格: <input type="number" ng-model="price">  
 <p><b>总价:</b> {{quantity \* price}}</p>  
</div>  
  
</body>  
</html>

**使用 ng-init 不是很常见。您将在控制器一章中学习到一个更好的初始化数据的方式。**

## 重复 HTML 元素

**ng-repeat** 指令会重复一个 HTML 元素：

[3-3]

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body>  
  
<div data-ng-app="" data-ng-init="names=['Jani','Hege','Kai']">  
 <p>使用 ng-repeat 来循环数组</p>  
 <ul>  
 <li data-ng-repeat="x in names">  
 {{ ***x*** }}  
 </li>  
 </ul>  
</div>  
  
</body>  
</html>

**ng-repeat** 指令用在一个对象数组上：

[3-4]

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body>  
  
<div ng-app="" ng-init="names=[  
{name:'Jani',country:'Norway'},  
{name:'Hege',country:'Sweden'},  
{name:'Kai',country:'Denmark'}]">  
  
 <p>循环对象:</p>  
 <ul>  
 <li ng-repeat="x in names">  
 {{ ***x***.name + ', ' + ***x***.country }}</li>  
 </ul>  
  
</div>  
  
</body>  
</html>

**AngularJS 完美支持数据库的 CRUD（增加Create、读取Read、更新Update、删除Delete）应用程序。  
把实例中的对象想象成数据库中的记录。**

## ng-app 指令

**ng-app** 指令定义了 AngularJS 应用程序的 **根元素**。

**ng-app** 指令在网页加载完毕时会**自动引导**（自动初始化）应用程序。

稍后您将学习到 **ng-app** 如何通过一个值（比如 ng-app="myModule"）连接到代码模块。

## ng-init 指令

**ng-init** 指令为 AngularJS 应用程序定义了 **初始值**。

通常情况下，不使用 ng-init。您将使用一个控制器或模块来代替它。

稍后您将学习更多有关控制器和模块的知识。

## ng-model 指令

**ng-model** 指令 **绑定 HTML 元素** 到应用程序数据。

**ng-model** 指令也可以：

* 为应用程序数据提供类型验证（number、email、required）。
* 为应用程序数据提供状态（invalid、dirty、touched、error）。
* 为 HTML 元素提供 CSS 类。
* 绑定 HTML 元素到 HTML 表单。

## ng-repeat 指令

**ng-repeat** 指令对于集合中（数组中）的每个项会 **克隆一次 HTML 元素**。

## 创建自定义的指令

除了 AngularJS 内置的指令外，我们还可以创建自定义指令。

你可以使用 **.directive** 函数来添加自定义的指令。

要调用自定义指令，HTML 元素上需要添加自定义指令名。

使用驼峰法来命名一个指令， **runoobDirective**, 但在使用它时需要以 **-** 分割, **runoob-directive**:

[3-5]

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body ng-app="myApp">  
  
<runoob-directive></runoob-directive>  
  
<script>  
 var ***app*** = angular.module("myApp", []);  
 ***app***.directive("runoobDirective", function() {  
 return {  
 template : "<h1>自定义指令!</h1>"  
 };  
 });  
</script>  
  
</body>  
</html>

你可以通过以下方式来调用指令：

* 元素名
* 属性
* 类名
* 注释

以下实例方式也能输出同样结果:

元素名

<runoob-directive></runoob-directive>

属性

<div runoob-directive></div>

类名

[3-6]

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body ng-app="myApp">  
  
<div class="runoob-directive"></div>  
  
<script>  
 var ***app*** = angular.module("myApp", []);  
 ***app***.directive("runoobDirective", function() {  
 return {  
 restrict : "C",  
 template : "<h1>自定义指令!</h1>"  
 };  
 });  
</script>  
  
<p><strong>注意：</strong> 你必须设置 <b>restrict</b> 的值为 "C" 才能通过类名来调用指令。</p>  
  
</body>  
</html>

注释

[3-7]

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body ng-app="myApp">  
  
*<!-- directive: runoob-directive -->*<script>  
 var ***app*** = angular.module("myApp", []);  
 ***app***.directive("runoobDirective", function() {  
 return {  
 restrict : "M",  
 replace : true,  
 template : "<h1>自定义指令!</h1>"  
 };  
 });  
</script>  
  
<p><strong>注意：</strong> 我们需要在该实例添加 <strong>replace</strong> 属性， 否则评论是不可见的。</p>  
  
<p><strong>注意：</strong> 你必须设置 <b>restrict</b> 的值为 "M" 才能通过注释来调用指令。</p>  
  
</body>  
</html>

## 限制使用

你可以限制你的指令只能通过特定的方式来调用。

**通过添加 restrict 属性,并设置值为 "A", 来设置指令只能通过属性的方式来调用:**

[3-8]

<!DOCTYPE html>  
<html><head>  
 <meta charset="utf-8">  
 <script src="js/angular.min.js"></script>  
</head>  
<body ng-app="myApp">  
  
<runoob-directive></runoob-directive>  
  
<div runoob-directive></div>  
  
<script>  
 var ***app*** = angular.module("myApp", []);  
 ***app***.directive("runoobDirective", function() {  
 return {  
 restrict : "A",  
 template : "<h1>自定义指令!</h1>"  
 };  
 });  
</script>  
  
<p><strong>注意：</strong> 通过设置 <strong>restrict</strong> 属性值为 "A" 来设置指令只能通过 HTML 元素的属性来调用。</p>  
  
</body>  
</html>

**restrict** 值可以是以下几种:

* E 作为元素名使用
* A 作为属性使用
* C 作为类名使用
* M 作为注释使用

**restrict** 默认值为 EA, 即可以通过元素名和属性名来调用指令。