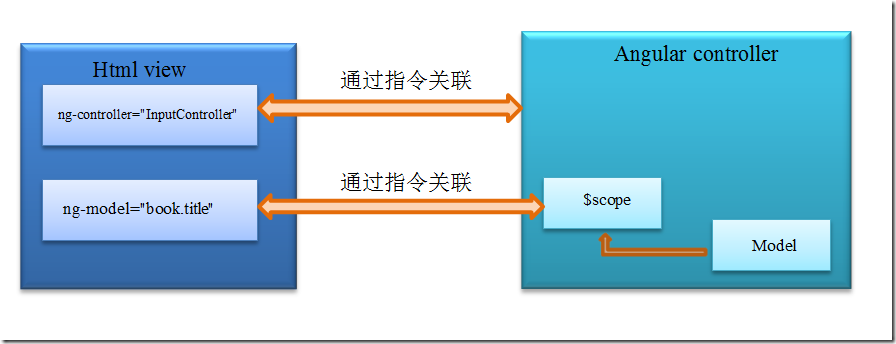
AngularJS

MVC（Model View Controller 模型-视图-控制器）View(视图), 即 HTML。Model(模型), 当前视图中可用的数据。Controller(控制器), 即 JavaScript 函数，可以添加或修改属性。



1、通过在div中添加属性ng-controller="InputController"，并设置属性值，关联到相关的控制器。

     也只有该div元素及其子元素，才能有权限使用InputController中的$scope对象上的模型数据。

2、ng-model="book.title"，通过angular解析，关联到其所处控制器中的$scope对象。

     根据指令的不同，关联到$scope对象上的方式也不同。

     ng-model指令将$scope对象与view对象的值进行双向绑定，犹如java中将对象的引用传给了view对象。

AngularJS 通过 [指令](http://www.runoob.com/angularjs/angularjs-directives.html)ng- 扩展了 HTML，且通过 [表达式](http://www.runoob.com/angularjs/angularjs-expressions.html){{}} 绑定数据到 HTML"输出"数据。

<div ng-app="">

<p>名字 : <input type="text" ng-model="name"></p>

<h1>Hello {{name}}</h1>

</div>

**ng-app** 指令告诉 AngularJS，<div> 元素是 AngularJS **应用程序** 的"所有者"。一个HTML文档只出现一次，如出现多次也只有第一个起作用，告诉子元素指令是属于angularJs。即定义了 AngularJS 应用程序的 **根元素，**

**ng-model** 指令把输入域的值绑定到变量 **name**。ng-model: 绑定元素值（比如输入域的值）

**ng-bind** 指令把应用程序变量 name 绑定到某个段落的 innerHTML ng-bind:数据绑定到 HTML 视图

ng-init一般不用，而使用更好的模块

指令 ng-

AngularJS 通过内置的指令来为应用添加功能。

AngularJS允许自定义指令 扩展（自制的）属性，以 data- 开头。 所以data-ng-

<div data-ng-app="" data-ng-init="firstName='John'">

<p>姓名为 <span data-ng-bind="firstName"></span></p>

</div>

表达式 {{}}

表达式写在双大括号内：{{ expression }} 并在表达式书写的位置"输出"数据

重复 HTML 元素

<div ng-app="" ng-init="names=['Jani','Hege','Kai']">   
<p>使用 ng-repeat 来循环数组</p>

<ul> <li ng-repeat="x in names"> {{ x }} </li> </ul>

</div>

应用

模块（Module） ng-app定义了应用, 模块定义了一个应用程序。所以模块是应用程序中不同部分的容器。

View: HTML

控制器（Controller） 用于控制 AngularJS 应用。ng-controller 定义了控制器。控制器通常属于一个模块。

<div ng-app="myApp" ng-controller="myCtrl">

名: <input type="text" ng-model="firstName"><br>

姓: <input type="text" ng-model="lastName"><br>

姓名: {{firstName + " " + lastName}}

</div>

<script>

var app = angular.module('myApp', []);创建模块 []为依赖 现表示无依赖

app.controller('myCtrl', function($scope) {

$scope.firstName= "John"; 相当于ng-init

$scope.lastName= "Doe";

});

</script>

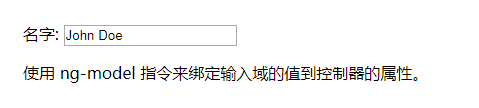


Scope(作用域)

应用在 HTML (视图) 和 JavaScript (控制器)之间的纽带

ng-model 指令

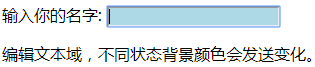
ng-model 指令可以将输入域的值与 AngularJS 创建的变量绑定

<div ng-app="myApp" ng-controller="myCtrl">  
    名字: <input ng-model="name">  
</div>  
  
<script>  
var app = angular.module('myApp', []);  
app.controller('myCtrl', function($scope) {  
    $scope.name = "John Doe";  
});  
</script>

CSS 类

ng-model 指令基于它们的状态为 HTML 元素提供了 CSS 类：

<style>

input.ng-invalid {  
    background-color: lightblue;  
}

</style>  
<body>  
  
<form ng-app="" name="myForm">  
    输入你的名字:  
    <input name="myAddress" ng-model="text" required>  
</form>

ng-invalid:未通过验证的表单

ng-valid:布尔型属性，它指示表单是否通过验证。如果表单当前通过验证，他将为true

ng-dirty:布尔值属性，表示用户是否修改了表单。如果为 ture，表示有修改过；false 表示修没有修改过

ng-touched:布尔值属性，表示用户是否和控件进行过交互

ng-pristine:布尔值属性，表示用户是否修改了表单。如果为ture，表示没有修改过；false表示修改过

验证用户输入

<form ng-app="" name="myForm">  
    Email:  
    <input type="email" name="myAddress" ng-model="text">  
    <span ng-show="myForm.myAddress.$error.email">不是一个合法的邮箱地址</span>  
</form>

应用状态

ng-model 指令可以为应用数据提供状态值(invalid, dirty, touched, error):

<form ng-app="" name="myForm" ng-init="myText = 'test@runoob.com'">  
    Email:  
    <input type="email" name="myAddress" ng-model="myText" required></p>  
    <h1>状态</h1>  
    {{myForm.myAddress.$valid}}  
    {{myForm.myAddress.$dirty}}  
    {{myForm.myAddress.$touched}}  
</form>

Scope(作用域)

Scope(作用域) 是应用在 HTML (视图) 和 JavaScript (控制器)之间的纽带。相当于this?

$scope 对象当作一个参数传递

<div ng-app="myApp" ng-controller="myCtrl">

<h1>{{carname}}</h1>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

$scope.carname = "Volvo";

});

</script>

scope 是模型scope 是一个 JavaScript 对象，带有属性和方法，这些属性和方法可以在视图和控制器中使用

<div ng-app="myApp" ng-controller="myCtrl">

<input ng-model="name">

<h1>{{greeting}}</h1>

<button ng-click='sayHello()'>点我</button>

</div>

<script>

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {

$scope.name = "Runoob";

$scope.sayHello = function() {

$scope.greeting = 'Hello ' + $scope.name + '!';

};

});

</script>