AngularJS (一)

讲师：文元

无限互联
wuxianedu.com
泛IT职业教育机构

走进无限互联

无限互联
wuxianedu.com

信·机构

- 慧科教育集团旗下泛IT职业教育培训机构。
- 与找座儿形成国内第一家O2O混合式教学落地基地。
- 良心教学，可信赖大机构！

新·模式

- 同时采用最合理的 TS 教学模式和最合理教学时间安排.
- 全程手把手教学、边学边练，7*12小时答疑指导，保证学员即学即会。

心·服务

- 多年教学经验资深讲师倾心教学
- 配备专门的生活导师，提供入学接送、精选宿舍、生活指导、学习督促等一条龙服务。
- 提供专业就业指导服务，简历指导+模拟面试，保证快速就业。

薪·未来

- 无限互联学生毕业平均薪资12755元，比同行业机构平均高出12%。
- 98%的无限互联学生在毕业4周内成功就业！

本章知识点

- AngularJS 入门
- AngularJS 指令
- AngularJS 控制器
- AngularJS 模块
- AngularJS MVC
- $scope、$rootScope

一、AngularJS 入门

1. AngularJS 简介
2. AngularJS 功能
3. 开发工具和调试工具的使用
4. 第一个 AngularJS 实例

AngularJS 简介

- AngularJS 最初由 Misko Hevery 和 Adam Abrons 于 09年开发,后成为了 Google 公司的项目,弥补了 HTML 在构建应用方面(数据处理)的不足,通过使用标识符 (directives) 结构,来扩展 web 应用中的 HTML 词汇,使开发者可以使用HTML来声明动态内容,从而使得 web 开发和测试工作变得更加容易

AngularJS 简介

- 通过指令扩展了 HTML，且通过表达式绑定数据到 HTML
- AngularJS 是一个 JavaScript 框,它可通过 <script> 标签添加到 HTML 页面
- 下载地址一: http://www.bootcdn.cn/angular.js/
- 下载地址二: http://www.ngnice.com/
- 资源网: http://www.runoob.com/angularjs/angularjs-tutorial.html

AngularJS 简介

- AngularJS 特性
  - MVC 架构模式
  - 模块化与依赖注入
  - 双向数据绑定
  - 指令

AngularJS 功能

- AngularJS 把应用程序数据绑定到 HTML 元素
- 可以克隆和重复 HTML 元素
- 可以隐藏和显示 HTML 元素
- 可以在 HTML 元素"背后"添加代码
- 支持输入验证

AngularJS 功能

- 传统的计数器是这样实现的

```
<input id="txt_value" type="number">
<input id="btn_add" type="button" value="增加">
<script>
  (function(window, document) {
    var txt = document.getElementById('txt_value');
    var btn = document.getElementById('btn_add');
    btn.addEventListener('click', function(e) {
      var now = txt.value - 0;
      now = now + 1;
      txt.value = now;
    });
  })(window, document);
</script>
```

传统计数器的使用

AngularJS 功能

- AngularJS 是这样实现的
- 代码量显著减少,实现的功能是一样的

```
<body ng-app>
  <input type="number" ng-model="value">
  <input type="button" ng-click="value=value+1" value="增加">
  <script src="../bower_components/angular/angular.js"></script>
</body>

</html>
```

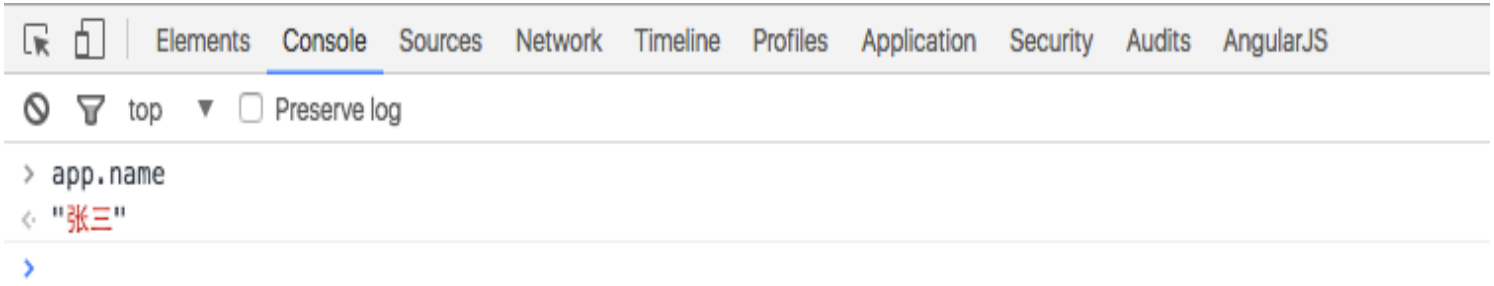
AngularJS实现计数器

开发工具和调试工具的使用

- 开发工具 sublime 下载地址:
  http://c758482.r82.cf2.rackcdn.com/Sublime Text 2.0.2 x64 Setup.exe
- 开发工具 WebStorm 下载地址:
  http://download-cf.jetbrains.com/webstorm/WebStorm-9.0.3.exe

开发工具和调试工具的使用

- 调试采用 Google Chrome 进行调试



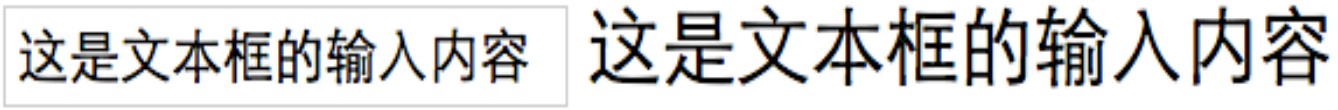
第一个 AngularJS 实例

- 导入 angular.js 库,建议把脚本放在 <body> 元素的底部。
  这会提高网页加载速度，因为 HTML 加载不受制于脚本加载。
- 使用 ng-app 指令告诉 angular 应该管理 DOM 的哪一部分

```
<body>
    <div ng-app = "">
        <input type="text" ng-model="name">
        {{name}}
    </div>
</body>
<script src="angular/angular.js"></script>
```

第一个 AngularJS 实例

- 此实例运行的效果为:在输入框中输入文本的时候,在文本框后面同步显示出输入的内容



二、AngularJS 指令(Directive)

1. AngularJS 指令简介
2. AngularJS 常用指令
3. AngularJS 表达式

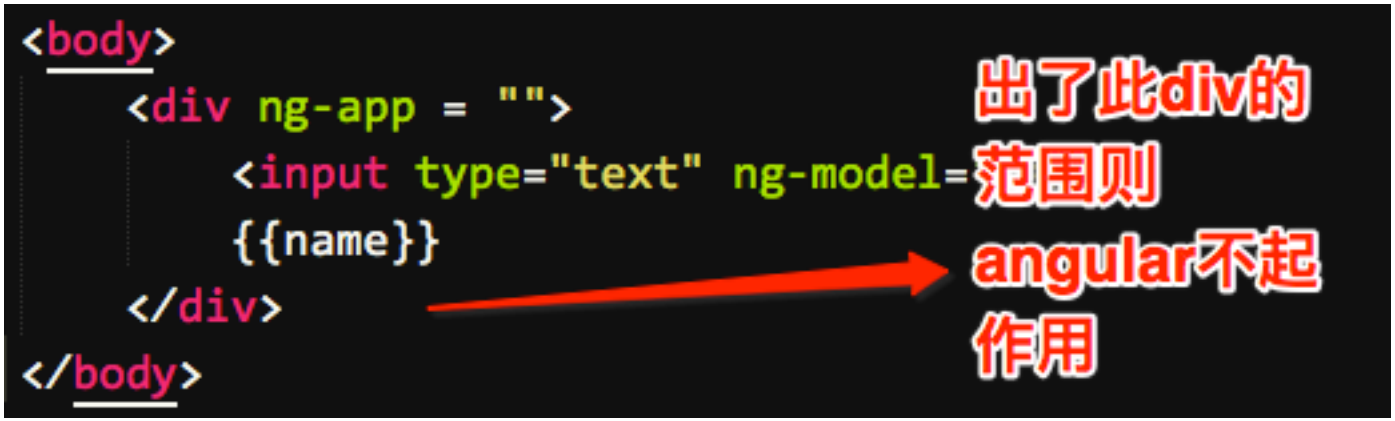
AngularJS 指令简介

- 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
- 带有前缀 ng-

```
</head>
<body>
    <div ng-app = "">
        <input type="text" ng-model="name">
        {{name}}
    </div>
</body>
<script src="angular/angular.js"></script>
```

AngularJS 常用指令

- HTML5 允许扩展的属性,以 data- 开头,AngularJS 指令以 ng- 开头,但是可以使用 data- ng- 来让网页对HTML有效 (ng-app == data-ng-app)
- ng-app 指令: 初始化一个 AngularJS 应用程序,指明 angular 应该管理 DOM 中的哪一部分
- 可以在同一个页面创建多个 ng-app 节点,但是一般只创建一个



AngularJS 常用指令

- ng-init 指令: 初始化应用程序数据,多个数据中间使用；隔开
- ng-model 指令: 指令把元素值(输入框的值)绑定到应用程序
- ng-bind 指令: 等同于{{}},绑定 HTML 元素到应用程序数据,相当于将应用程序作为某个元素的innerHTML

AngularJS 常用指令

- 以下代码中,初始化了两个应用程序数据 lastname , firstname, 通过 {{}} 在网页中显示,输入框的值绑定到了应用程序中,名字为 name,再使用ng-bind 指令将输入框的值作为 span 标签 innerHTML 在网页中显示出来

```
<div ng-app = "" ng-init="lastname = 'a';firstname = 'b'">
    <p>在输入框中尝试输入:</p>
    <input type="text" ng-model="name"><br><br>
    初始化的数据:<div>{{lastname}}</div><br><br>
    bind:<span ng-bind="name"></span>
</div>
```

AngularJS 常用指令

- 其他指令(先了解,后面的课程将会一 一讲到)

指令	描述
ng-click	定义元素被点击时的行为
ng-controller	为应用程序定义控制器对象
ng-disabled	绑定应用程序数据到 HTML 的 disabled 属性
ng-repeat	对于集合中(数组中)的每个项会克隆一次 HTML 元素
ng-show	值为true的时候显示元素
ng-hide	值为true的时候隐藏元素
ng-if	值为true的时候从DOM中移除元素

AngularJS 表达式

- AngularJS 表达式写在双大括号内：**{{ expression }}**
- AngularJS 表达式把数据绑定到 HTML，这与 **ng-bind** 指令有异曲同工之妙。
- AngularJS 将在表达式书写的位置"输出"数据。
- AngularJS 表达式很像 JavaScript 表达式：它们可以包含文字、运算符和变量。

AngularJS 表达式

- AngularJS 数字就像 JavaScript 数字

```
<div ng-app="" ng-init="quantity=1;cost=5">
    <p>总价： {{ quantity * cost }}</p>
</div>
```

- AngularJS 字符串就像 JavaScript 字符串

```
<div ng-app="" ng-init="firstName='John';lastName='Doe'">
    <p>姓名： {{ firstName + " " + lastName }}</p>
</div>
```

AngularJS 表达式

- AngularJS 对象就像 JavaScript 对象

```
<div ng-app="" ng-init="person={firstName:'John',lastName:'Doe'}">
    <p>姓为 {{ person.lastName }}</p>
</div>
```

- AngularJS 数组就像 JavaScript 数组

```
<div ng-app="" ng-init="points=[1,15,19,2,40]">
    <p>第三个值为 <span ng-bind="points[2]"></span></p>
</div>
```

三、AngularJS 控制器

1. AngularJS 控制器简介
2. AngularJS 控制器的创建
3. ng-repeat 结合控制器

AngularJS 控制器简介

- AngularJS 控制器 控制 AngularJS 应用程序的数据
- AngularJS 控制器是常规的 JavaScript 对象
- AngularJS 应用程序被控制器控制
- ng-controller 指令定义了应用程序控制器
- 控制器是 JavaScript 对象,由标准的 JavaScript 对象的构造函数创建

AngularJS 控制器的创建

- AngularJS 控制器的创建方式如图所示:

```
    <div ng-app="">
        <div ng-controller="firstController">
            {{title}}<br>
            {{person.name}}
        </div>
    </div>
</body>
<script src="angular-1.3.0.js"></script>
<script>
    function firstController($scope) {
        $scope.title = '男神';
        $scope.person = {
            name : 'hyh'
        };
    }
```

AngularJS 控制器的创建

- ng-controller=“firstController”,定义了一个名为 firstController 的控制器
- firstController 函数是一个 JavaScript 函数,对应以上控制器名
- AngularJS 使用**$**scope 对象来调用控制器
- 控制器的 $scope (相当于作用域、控制范围)用来保存 AngularJS Model(模型)的对象,达到双向绑定的效果
- 控制器在作用域中创建了两个属性 (title 和 person)

ng-repeat 结合控制器

- ng-repeat 指令对于集合中(数组中)的每个项会克隆一次 HTML 元素
- ng-repeat 指令会重复一个 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>
```

ng-repeat 结合控制器

- ng-repeat 作用于对象数组

```
<div ng-app="" ng-init="names=[
        {name:'zs',age:11},
        {name:'li',age:12},
        {name:'ww',age:13},
        {name:'tq',age:14}
    ];">
    <p>使用 ng-repeat 来循环对象数组</p>
    <ul>
        <li ng-repeat="x in names">{{ x.name + ',' + x.age }}</li>
    </ul>
</div>
```

ng-repeat 结合控制器

- ng-repeat 作用于对象

```
<div ng-app="" ng-init="names={name:'zs',age:21,height:171,weight:250}">
    <p>使用 ng-repeat 来循环对象数</p>
    <ul>
        <li ng-repeat="(key,value) in names">{{ key + ',' + value }}</li>
    </ul>
</div>
```

课堂练习

- 使用 ng-repeat 与控制器结合的方式,输出一个表格,表格样式如下图所示

人员信息表

zs	11
li	12
ww	13
tq	14

四、AngularJS 模块

1. 使用模块的意义
2. 模块功能
3. 模块的使用
4. 应用程序文件

使用模块的意义

- 控制器污染了全局命名空间
- 在所有应用程序中,都应该尽量避免使用全局函数和全局变量
- 全局值(变量或函数)可被脚本重写或破坏
- 为了解决以上问题, AngularJS 使用了模块

模块的功能

- 模块定义了一个应用程序
- 模块是应用程序中不同部分的容器
- 模块是应用控制器的容器
- 控制器通常属于一个模块
- 可以在模块中添加控制器

模块的使用

- 通过 ng-app 指定AngularJS的应用程序,来实现myApp这个模块启动应用的
- 通过 angular 对象调用 module 方法,创建,注册或者检索一个Angular应用程序
- angular 对象可看成是 AngularJS 的内置对象,可以调用许多方法及属性

```
    <div ng-app="myApp">
    </div>
</body>
<script src="angular-1.3.0.js"></script>
<script>
    //创建,注册或者检索一个Angular应用程序
    var app = angular.module('myApp',[]);

</script>
```

模块的使用

- module 方法中第一个参数与 ng-app 的值一直
- 在模块中定义 [] 参数用于定义模块的依赖关系
- 在此处表示该模块没有依赖,如果有依赖的话则在 [] 中写上依赖的模块名字

```
    <div ng-app="myApp">
    </div>
</body>
<script src="angular-1.3.0.js"></script>
<script>
    //创建,注册或者检索一个Angular应用程序
    var app = angular.module('myApp',[]);

</script>
```

模块的使用

- 前面所学的创建控制器的方式是使用一个全局函数,污染了全局命名空间,所以一般在模块中添加控制器
- 通过获取的应用程序调用 controller 方法来创建控制器
- 第一个参数是控制器的名字,第二个参数与前面所学的全局函数一样,只是不再是全局函数了

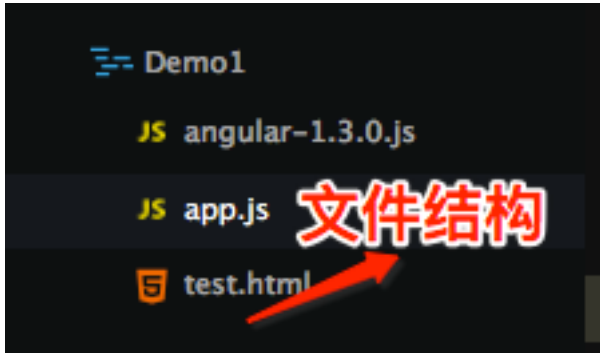
```
//创建,注册或者检索一个Angular应用程序
var app = angular.module('myApp',[]);

//创建控制器
app.controller('firstCtrl',function  ($scope) {
    // body...
    $scope.title = '男神';
});
```

应用程序文件

- 现在已经知道模块是什么以及它们是如何工作的,可以尝试创建自己的应用程序文件
- 应用程序至少应该有一个模块文件,一个控制器文件
- 模块文件名为: myApp.js ,在此文件中,创建模块
- 控制器名称为: ctrls.js ,在此文件中,创建控制器
- 在 html 文件最后引入文件
- 注: 当模块、控制器较少时, 可以考虑单独建立一个 app.js 文件,将模块以及控制器的创建都放入此文件中

应用程序文件




```
//创建,注册或者检索一个Angular应用程序
    var app = angular.module('myApp',[]);

    //创建控制器
    app.controller('firstCtrl',function  ($scope) {
        // body...
        $scope.title = '男神';
    });
```

文件内容

五、AngularJS MVC

1. MVC 介绍
2. MVC 优缺点
3. MVC Controller 的实现方式
4. Controller 注意点

MVC 介绍

- MVC 是一种经典的架构模式,不是设计模式
- MVC 模式代表 Model-View-Controller（模型-视图-控制器） 模式。这种模式用于应用程序的分层开发。

MVC 介绍

- Model（模型） - 模型代表一个存取数据的对象,它也可以带有逻辑，在数据变化时更新控制器。
- View（视图） - 视图代表模型包含的数据的可视化
- Controller（控制器） - 控制器作用于模型和视图上。它控制数据流向模型对象，并在数据变化时更新视图。它使视图与模型分离开

MVC 优缺点

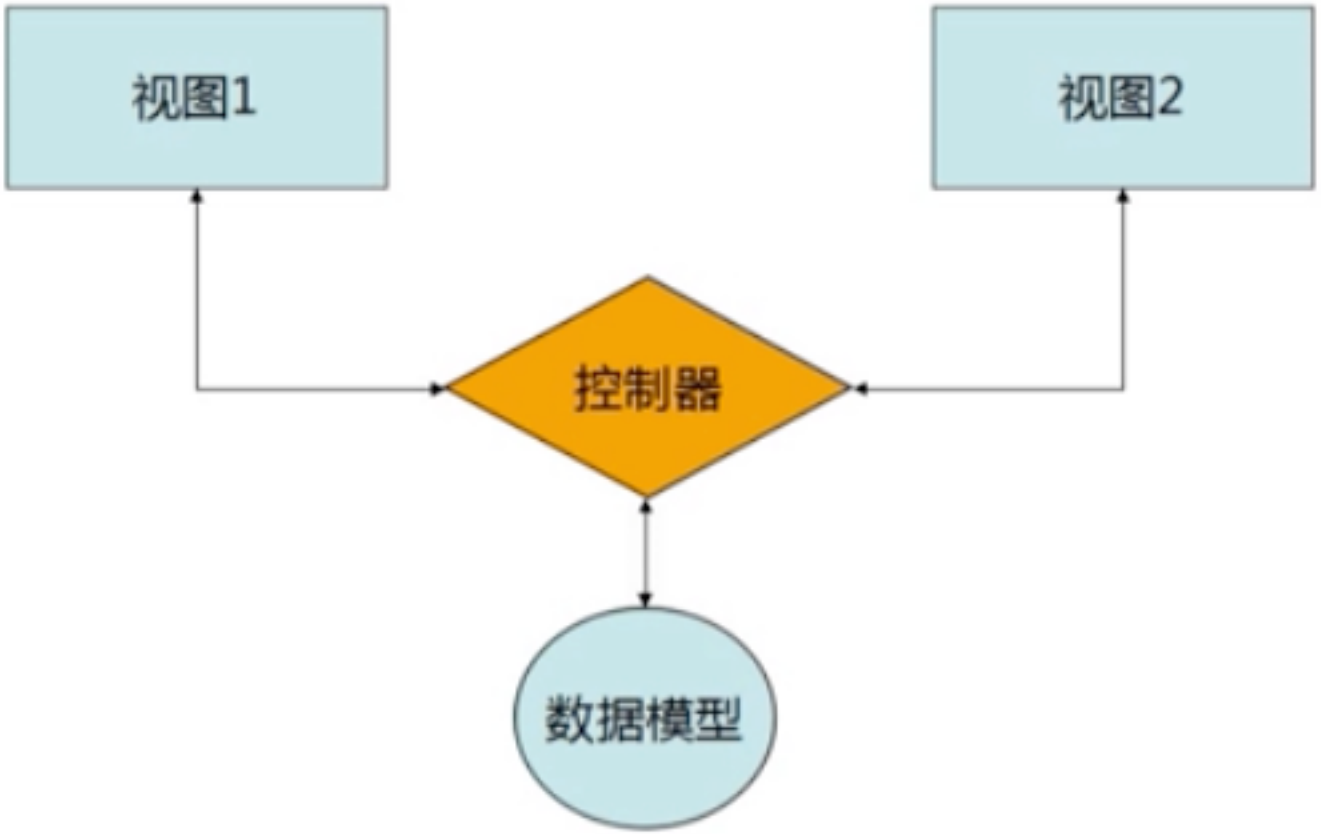
- 优点
  - 代码逻辑比较清晰、可移值性高
  - 后期维护方便、可以代码复用
  - 代码规模越来越大的时候,切分职责是大势所趋
  - 每个部件有自己明确的职责，相互之间没有依赖
  - 解决应用程序展示结构，业务逻辑之间的紧耦合关系

MVC 优缺点

- 缺点
  - 运行效率相对较低,但是可以忽略不计
  - 多个 js 文件之间如果出现互相依赖,程序员必须自己解决

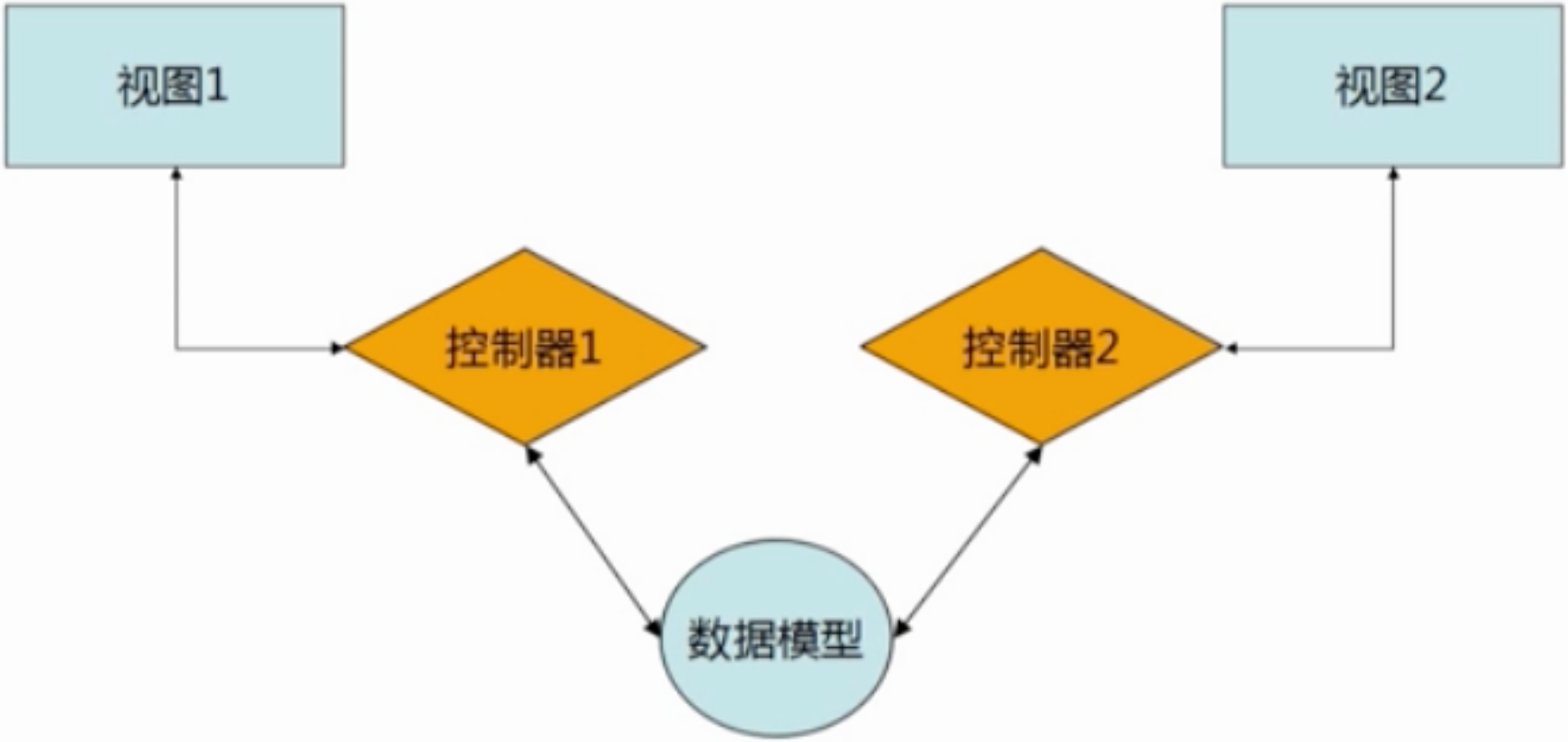
MVC 只是手段,终极目标是模块化和复用

MVC Controller 的实现方式



问题: 如果视图1和视图2根本没有任何逻辑关系,控制器的角色就会很尴尬

MVC Controller 的实现方式

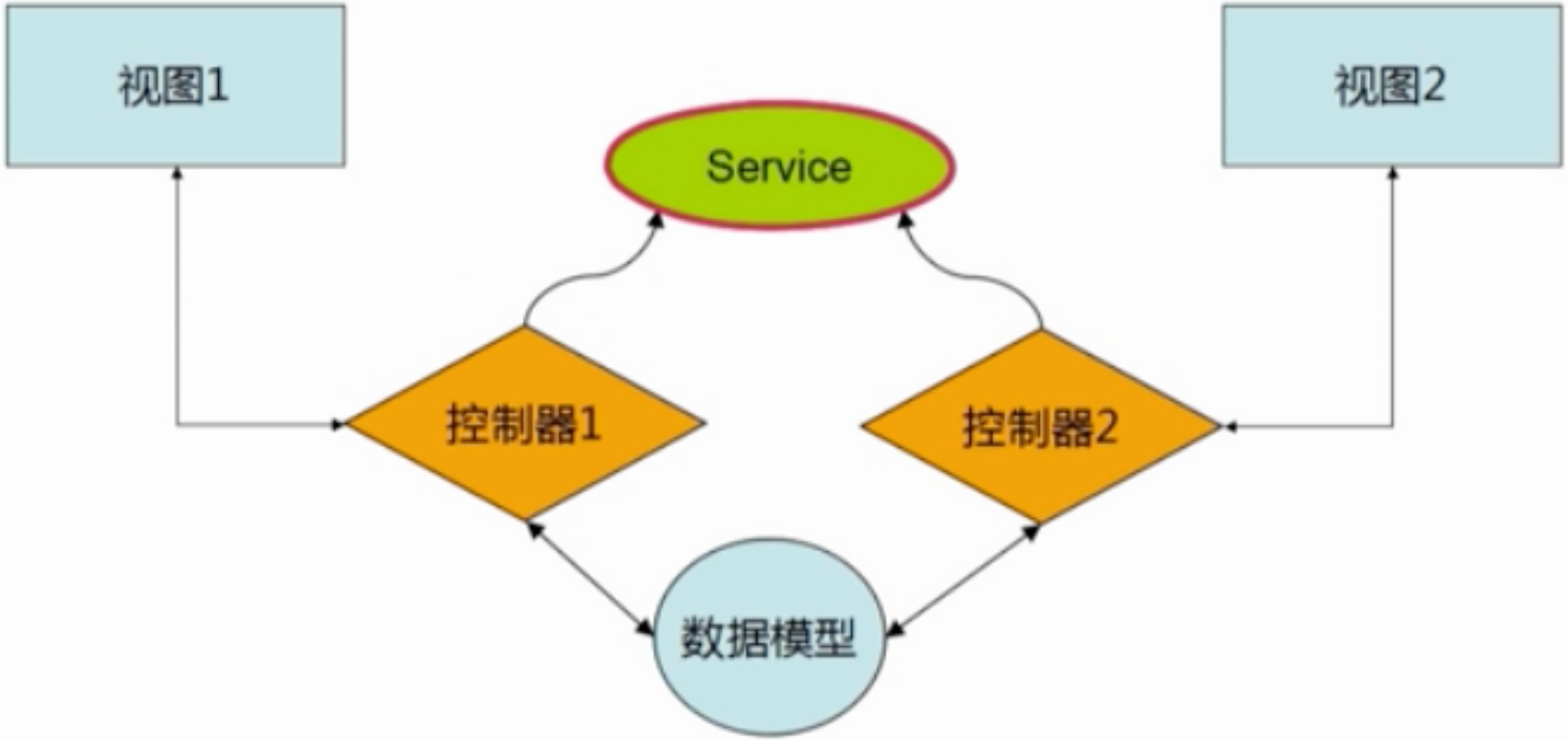


问题: 如果控制器1和控制器2里面有2个方法是一模一样的,那应该怎么办?

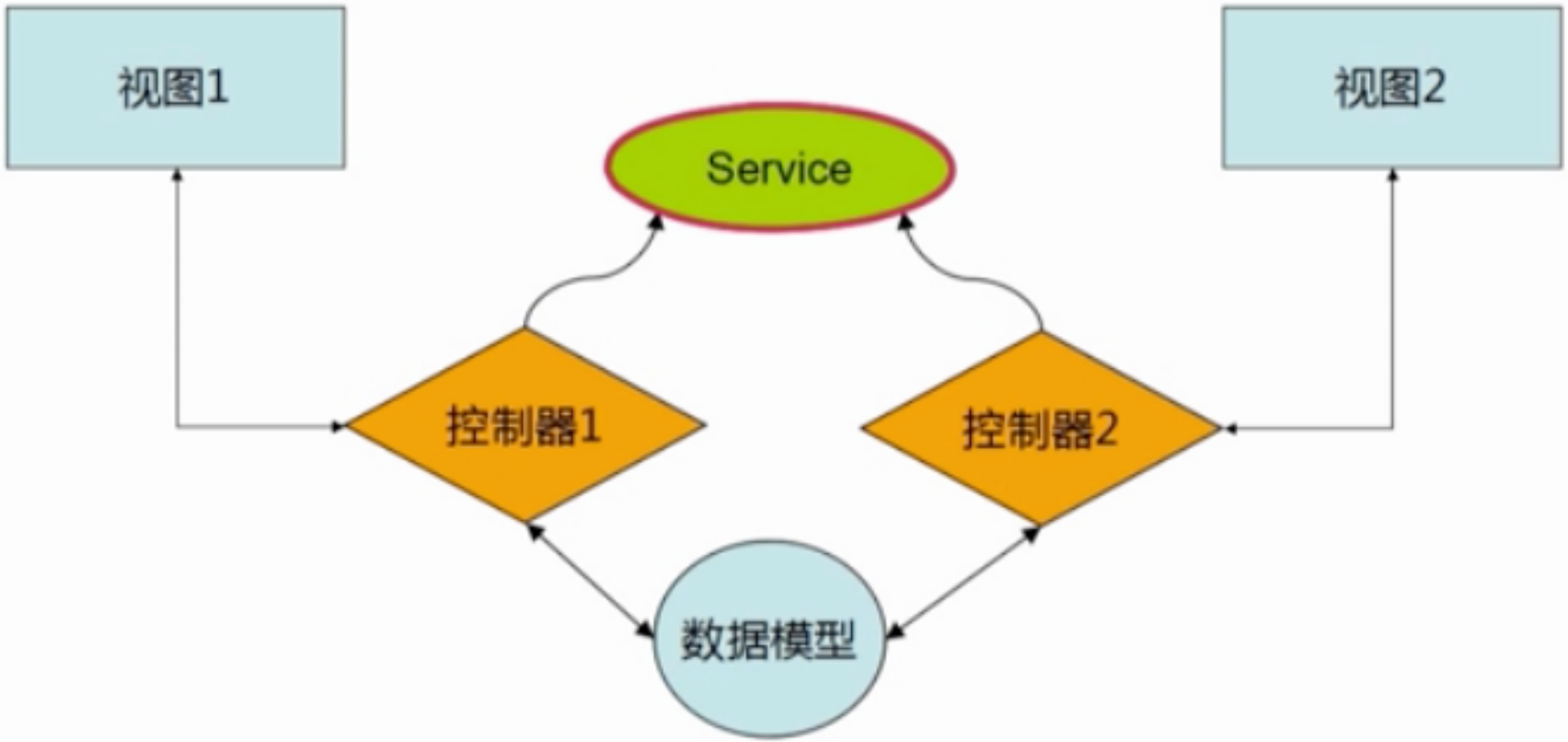
MVC Controller 的实现方式

这是设计时候的一个坑,在 AngularJS 里面不推荐使用此种方式来设计,而是希望将那些相同的数据或者方法都写到一个服务 (Service) 里面,再进行调用

MVC Controller 的实现方式



这是最常用的设计



这是最常用的设计

Controller 注意点

- 不要试图复用 Controller ,一个控制器一般只负责一块视图
- 不要在 Controller 中操作 DOM,这不是控制器的职责
- 不要在 Controller 中做数据格式化,ng 有很好用的表单控件
- 不要在 Controller 中做数据过滤操作,ng 有 $filter 服务
- 一般来说, Controller 是不会互相调用的,控制器之间的交互会通过事件进行
- 利用 自定义Directive(指令) 实现 View 的复用

六、$scope 、$rootScope

1. $scope 简介
2. 如何使用 $scope
3. $rootScope 简介
4. 如何使用 $rootScope
5. 依赖注入中代码压缩问题

$scope 简介

- $scope 是表达式的执行环境,或者成为作用域
- 子 $scope 会继承父 $scope 上的属性方法
- $Scope(作用域) 是应用在 HTML (视图) 和 JavaScript (控制器)之间的纽带
- $Scope 是一个对象，有可用的方法和属性
- $Scope 可应用在视图和控制器上
- 每个控制器的 $scope 作用范围只在此控制器
- 所有的 $scope 都是直接或者间接继承于 $rootScope,拥有 $rootScope 的属性和方法

如何使用 $scope

- 当创建控制器时,可以将 $scope 对象当作一个参数传递
- 当在控制器中添加 $scope 对象时，视图 (HTML) 可以获取了这些属性
- 视图中，你不需要添加 $scope 前缀, 只需要添加属性名即可，如： {{title}}

```
<div ng-controller = "firstCtrl">
    {{title}}
</div>
```

view中使用属性

```
//创建控制器
app.controller('firstCtrl',function  ($scope) {
    // 设置属性
    $scope.title = '男神';
});
```

设置属性

$rootScope 简介

- 所有的应用都有一个 $rootScope (根作用域)
- 可以做用在 ng-app 指令包含的所有HTML 元素中
- 是各个 controller 中的桥梁
- 用 $rootScope 定义的值,可以在各个 controller 中使用
- $rootScope 是所有 $scope 的直接或间接父 $scope

如何使用 $rootScope

- 用法与 $scope 类似,一般用在 run() 方法中
- run() 方法初始化全局的数据,只对全局作用域起作用

```
app.run(['$rootScope','$scope',function(rootScope,scope) {
    rootScope.person = '华杨海';
    // scope.person2 = '男神';  错误,只对全局的作用域起作用
}]);
```

依赖注入中的代码压缩问题

- 当用户浏览网页时,需要加载 js 文件,而这个加载过程是从服务器下载的 js 文件进行加载的,出于为用户流量考虑的角度来看,一般需将 js 代码进行压缩
- 由于 AngularJS 是通过控制器里面的实现函数的参数名字来推断依赖服务名称的。所以如果你要压缩控制器的JS代码，它所有的参数也同时会被压缩，这时候依赖注入系统就不能正确的识别出服务了

依赖注入中的代码压缩问题

- 这个时候要来处理这产生的代码压缩问题
- 使用Javascript数组方式构造控制器：把要注入的服务放到一个字符串数组里，数组最后一个元素是控制器的方法函数,字符串是不会被压缩的

```
//创建控制器
app.controller('firstCtrl',['$scope',function ($scope) {
    $scope.title = '男神';
}]);
```

依赖注入中的代码压缩问题

- 数组里面的字符串必须是各种服务 (Service) 的名字,并且得和控制器函数里面的形参一一对应
- 当解决了代码压缩问题的时候,字符串必须是 Service 的名字,而对应的形参的名字可以随便起,在控制器中就可以用此形参代替 $scope,功能还是一样的

```
//创建控制器
app.controller('firstCtrl',['$scope',function (ss) {
    ss.title = '男神';
}]);
```

本章总结

- AngularJS 的四大特性:MVC 架构模式、模块化与依赖注入、双向数据绑定、指令
- 要使用 AngularJS 必须导入 js 文件
- 必须设置 ng-app 指令,告诉 angular 应该管理 DOM 中的哪一部分
- {{}} 可以在网页中 ” 输出 ”
- 控制器的创建应该避免使用全局函数来创建

课后作业

- 练习控制器的使用
- 练习 $scope、$rootScope 的使用
- 练习常用指令的使用
- 理解 MVC 架构模式

无限互联
wuxianedu.com
慧科集团旗下企业

官方微信

官方网站： http://www.wuxianedu.com/

无限智造 互联精英