

# 复习：第一天的内容

AngularJS的优势：

无DOM操作，而是利用双向绑定的形式，让数据直接与DOM相关联。实现程序员只操作数据就可以更新视图的效果。简化了开发的流程。DOM选择器越复杂，性能越低。

AngularJS核心概念：

1. 单页面应用特征：数据频繁发生变动，需要用JavaScript来更新界面的显示。在不使用框架时，这会要求程序员对DOM十分熟悉，编写非常复杂的代码，使用很多效率高的选择器。
2. 前端模板和双向数据绑定：
  - a. AngularJS把HTML文件视作模板，程序员用AngularJS指令的方式在HTML上打标记，然后把数据等内容交给AngularJS，AngularJS帮我们把数据填到对应的位置上、帮我们把各种行为的响应函数绑定到各种控件上。
  - b. AngularJS把作用域绑定到HTML元素上，在对应HTML元素上的各种指令就会与这个作用域进行关联，实现数据（或者样式）的绑定（ng-bind、{{}}、ng-style、ng-class）、双向绑定（ng-model）、事件绑定（ng-click）等等。
3. 依赖注入：在回调函数里面写指定的参数名就能获得指定的对象。（写\$scope就能获得\$scope，写\$http就能获得\$http，哪怕参数的位置改变都不会出问题，AngularJS知道调用这个函数时自己应该怎么传值）
4. MVC “程序三问”：

数据从哪儿来？用户输入、网络访问等等，Model。

数据去哪儿了？渲染到界面上了，View。

发生了什么？各种事件监听和事件处理函数，Controller。但是AngularJS的Controller和标准的MVC概念有些不同，AngularJS的Controller给人一种“专门用于组织\$scope的内容的”的感觉。
5. 指令：在HTML文本上打的各种供AngularJS识别并进行绑定的标记。

AngularJS常用指令：

1. 程序控制类：ng-app ng-controller
2. 数据绑定类：{{}} ng-bind ng-model
3. 事件绑定类：ng-click ng-dblclick ng-blur ng-focus ng-change ...
4. 状态绑定类：ng-class ng-style ng-readable ng-disable ng-hide ....
5. 流程控制类：ng-if ng-switch
6. ng-repeat循环生成（今天才讲）

# 模块与ng-app

★ 模块module的作用是存储一组AngularJS的功能组件，并可以被其他模块依赖、可以依赖其他模块。

模块的声明方法是：`var app = angular.module('demo.main',[])`

参数1：这个模块的名字

参数2：这个模块所依赖的其他模块。（一个数组，存储了所依赖的其他模块的名字）

在HTML标签上用 `ng-app='demo.main'` 指定一个模块的模块名，意味着：

1. 让AngularJS框架在这个标签上启动，并载入这个模块。
  2. 载入这个模块之后，在这个标签内部就可以使用这个模块上所挂载着的各种AngularJS功能组件。比如说用于把 `$scope`和HTML标签绑定到一起的controller。
- 

模块的名字 m1

```
var a = 1
var b = 2
var fn1 = function(){ ... }
```

exports.a = a;  
exports.b = b

模块 m2

```
var m1 = require('m1')
var a = m1.a
var b = m1.b
```

# 控制器与ng-controller

AngularJS的控制器，其最大的作用就是把一个作用域（\$scope）和模板上的一个HTML标签绑定到一起。然后在这个标签中的AngularJS表达式就可以依附于该作用域执行。

用法：

1. 创建控制器：

创建模块（假设我们已经用一个名为app的变量存储了一个模块），并在模块上创建一个AngularJS控制器

```
app.controller('mainController', function( $scope ){  
    // Todo: 为这个模块的$scope准备各种数据（ data, action等等 ）  
})
```

2. 与HTML上的标签相链接

```
<div ng-controller="mainController"> ..... </div>
```

# 细节：依赖注入

2016年5月12日 0:25

注意我们声明控制器时所做的事情：

```
app.controller('mainController',function($scope){  
    $scope.xx = xx;  
})
```

我们创建AngularJS控制器时，在第二个参数的回调函数里，写了一个\$scope参数。AngularJS框架读取了参数名“\$scope”，然后把名为“\$scope”的服务交给了我们。这个就是AngularJS里面的依赖注入。说白了就是获取AngularJS框架提供的各种功能，和jQuery的\$.ajax(...)是很像的，只不过看上去比jQuery更酷炫一点。有一种“你的函数想要什么就直接给你什么”的感觉。

# AngularJS表达式

关于表达式（来自百度百科）：

、数字分组符号（括号）、自由变量和约束变量等以能求得数值的有意义排列方法所得的组合。约束变量在表达式中已被指定数值，而自由变量则可以在表达式之外另行指定数值。

给与自由变量一些数值指定，可以给与一个表达式数值，即使对于一些自由变量的值，表示式或许没有定义。因此，一个表达式代表一个函数，其输入为自由变量的定值，而其输出则为表示式因之后所产生出的数值。

来自 [http://baike.baidu.com/link?url=-\\_Je10FARgpCixsSUdwLsmQyYYagbmsiOPp9Rr0E5gr6v5n0HijA7ziw9Op2WgOKmp9JLC3QJ6thzuKHfCRdcig](http://baike.baidu.com/link?url=-_Je10FARgpCixsSUdwLsmQyYYagbmsiOPp9Rr0E5gr6v5n0HijA7ziw9Op2WgOKmp9JLC3QJ6thzuKHfCRdcig)

什么是AngularJS表达式？

这样：{{ AngularJS Expression }}

或者这样：ng-click = "AngularJS Expression"

这样：ng-bind="AngularJS Expression"

这些指令内部输入的，其实都是AngularJS表达式。

- ★ 任何AngularJS表达式在执行之后都有值，所以才能进行绑定。
- ★ AngularJS表达式执行时，通常来说会需要一个作用域。（或者是表达式总是在作用域上执行）
- ★ AngularJS的表达式可以写什么？

```
比如说我们有 $scope = {  
  data:{  
    num : 1,  
    array:[1,2,3,4,5]  
  },  
  action:{  
    sendMsg : function( ... ){ ... },  
    getNum:function(){ return 1}  
  }  
}
```

表达式类型	书写方法	值
取值	data.num	1

取值	action.sendMsg	function( ... ){ ... }
算数	data.num+1	2
函数执行	action.getNum()	1 ( 函数执行之后返回的值 )
数组取值	data.array[3]	4
三联运算符	data.num == 1 ? 1 : 2	1
.....	.....	其他许多和JavaScript表达式类似的语法，但是不包括自增、if判断等

# AngularJS作用域

关于作用域（下面的内容来自百度百科）：

## 作用域（scope）

通常来说，一段程序代码中所用到的名字并不总是有效/可用的，而限定这个名字的可用性的代码范围就是这个名字的作用域。

作用域的使用提高了程序逻辑的局部性，增强程序的可靠性，减少名字冲突。

来自 <[http://baike.baidu.com/link?url=AAXe63jEQkUG4mv39EIIwt7QWjX83imVCr7uDqJw1D9R0rBLd-m7aCcAtTR\\_CQoQ3e1YoexwFQVVEUPBSMvRUa](http://baike.baidu.com/link?url=AAXe63jEQkUG4mv39EIIwt7QWjX83imVCr7uDqJw1D9R0rBLd-m7aCcAtTR_CQoQ3e1YoexwFQVVEUPBSMvRUa)>

- 💡 首先纠正一个误解：作用域并不是AngularJS的Controller独有的东西。实际上很多指令都有自己的作用域，只不过Controller专门用于把作用域和HTML标签绑定到一起。

那么，作用域到底是用来做什么的呢？我们需要把作用域与AngularJS表达式结合来看：

- ★ **AngularJS作用域，通常来说，它的作用就是给AngularJS表达式提供一个执行环境的。**
- 💡 实际上，无论是插值语法{{}}、AngularJS专有属性ng-bind ng-model ng-click，它们内部放着的都是AngularJS表达式，AngularJS在需要值的时候，会根据这些表达式的值做各种操作：替换、数据绑定、事件绑定等等。那么，这些AngularJS表达想要运行起来，必须怎样？当然是必须有一个它运行起来的环境了。这就是AngularJS的作用域。

# AngularJS与MVC、MVVM



AngularJS实质上更接近MVVM：

HTML的某个标签与某个\$scope互相连接

\$scope上的各种数据、动作都可以在那个标签内部的AngularJS表达式访问。

类似于\$scope这样的东西，就会被称作是VM（视图模型）。

换句话说也就是：构建视图所需要的数据。

那么VM和M之间的关系呢？

答案是，交集。



# 作用域：表达式求值

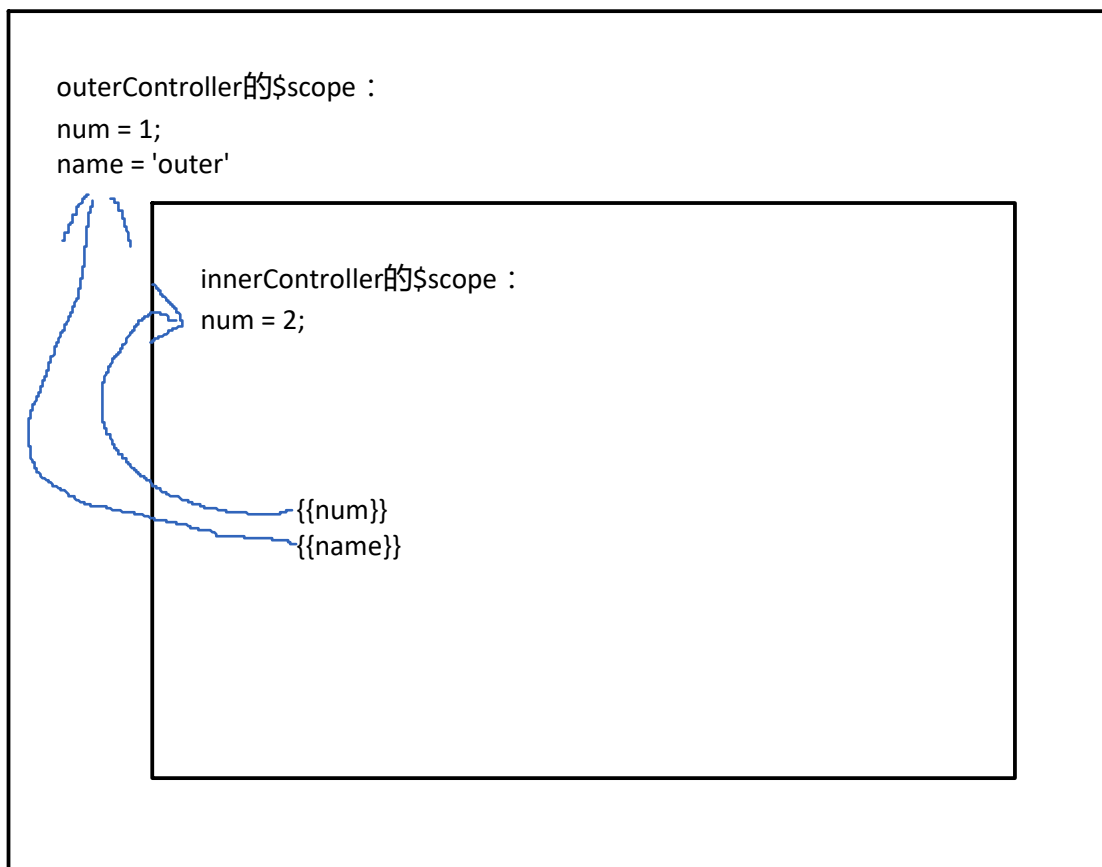
之前提到过AngularJS的表达式必须依附于作用域运行。那么AngularJS内部到底是怎么做的呢？可以关注一下AngularJS框架内部的\$parse服务，通过依赖注入拿到的\$parse实际上是一个函数。用法很简单：

```
var parseFn = $parse('damo.name')  
var nameOnScope = parseFn($scope)
```

# 作用域：嵌套问题

AngularJS的作用域是可以互相嵌套的，内部作用域可以访问外部的数据，当内部作用域和外部作用域名称冲突时，使用的是内部作用域上的数值。

观察作用域对象，你会发现它其实利用了JavaScript的原型机制。



# 作用域：通知作用域数据发生了改变

使用`$scope.$apply()`来通知AngularJS数据发生了变化，去更新视图

## 作用域：监视数据变化

```
var unregisterWatch = $scope.$watch(  
  'data.name',  
  function( newValue, oldValue, scope){  
    // Todo : 数据发生变化时做什么  
  })
```

这样可以监视这个作用域上的数据的变化。

- 💡 其本质是，监听这个作用域上执行这个表达式后，获得的值有没有发生变化。
- 小实验：用\$location获取当前的网址，并利用\$scope监视网址的变化。

watcher:

1. AngularJS表达式
2. 上一次对这个表达式求出来值
3. 如果和上一次表达式的值有变化，则执行我们传进去的回调函数。

## 指令：ng-repeat

▼ Models
<pre>\$id: 7 item:   name: "张三"   age: 18   sex: "男" \$index: 0 \$first: true \$last: false \$middle: false \$even: true \$odd: false</pre>

item: 数组里面的每一个元素（每一个ng-repeat生成\$scope中,item都不同）item来自于 "item in array"的写法，实际上item可以随便自定义它的名字。

\$index:这个元素在数组里的索引

\$first: 这个元素是不是数组里的第一个元素

\$last:这个元素是不是数组里最后一个元素

\$middle:.....是不是中间的元素

\$even:索引值是不是偶数

\$odd:索引值是不是奇数

```
<tr ng-repeat="item in infoes" ng-class="{ 'bg-blue': $even }">
```

绑定的数组，语法是 item in array

如果你的repeat所绑定的数组经常发生变化，最好也加一个track by \$index

```
<tr ng-repeat="(key, item) in infoes" ng-class="{ 'bg-blue': $even }">
```

绑定对象，语法是(key, value) in object

# AngularJS服务

可以配合依赖注入使用。

AngularJS服务应用场景：

1. 封装Model层和数据访问
2. 封装各种配置
3. 封装特殊功能

用app.factory自定义AngularJS服务。

💡 补充：用provider创建AngularJS服务，并在AngularJS模块中进行设置。

```
app.provider('demoService', function () {
    var demoStr = 'demo';
    return {
        setStr: function (val) {
            demoStr = val;
        },
        $get: function () {
            return demoStr;
        }
    }
});
```

```
app.config(function (demoServiceProvider) {
    demoServiceProvider.setStr('hello world')
});
```

# AngularJS的\$http服务

服务的名字叫\$http。

用法：

```
$http({
  method: "POST",
  url: "data.jsonl",
  params: {username: 'ted', age: 14},
  data: {password: "123456"}
}).success(function (data, status) {
  console.log('post', arguments);
}).error(function (data, status) {
  console.log('err', data, status)
})
```

VM - ViewModel



# 邀请名单的程序的程序的分析

## MVVM

1. M: Model , 模型 , 数据
  - a. 所有的被邀请人的数据。
2. V:View , 视图。
  - a. ng-model
  - b. ng-click
  - c. ng-repeat
3. VM:ViewModel , 视图模型 , 生成视图时所需要的数据
  - a. 要显示到屏幕上的人员信息的数组
  - b. 正在添加的被邀请人的姓名和电话