# 0.回顾

## 0.1. AngularJS概述

AngularJS是一款基于 MVC 设计模式的 JS 框架，适用于以数据操作为主的SPA应用。

## 0.2 四大特性之MVC复习：

* 采用MVC模式
  + Model：模型，即业务数据，AngularJS中由变量来表示数据；ng中保存在特定作用范围内的变量 $scope.xx=yy;
  + View：视图，负责数据的呈现， AngularJS中由HTML担当，但是HTML不能循环和选择，NG的指令可以增强HTML，也就是说HTML+指令
  + Controller：控制器，负责操作（CRUD Create Retrive Update Delete）数据， ng中function
* 双向数据绑定
* 依赖注入
* 模块化设计，即一个箱子/容器，可以用于盛放全局函数、控制器等。

学习angularJS之前，先学习其基本语法，主要包括：表达式和指令。

想在view中显示Model数据，必须有数据（ng-init/ng-controller）

Module<=Controller<=Model<=View

AngularJS使用MVC模式的步骤：

1. 创建一个自定义模块
2. 将自定义模块在当前应用中注册
3. 模块中创建控制器函数
4. 在View中使用ng-controller创建控制器实例
5. 使用控制器来声明Module模型数据
6. 在View中呈现Module数据

# 双向数据绑定

## 1.1 方向1

新建1.html

<!DOCTYPE html>

<html lang="en" ng-app="myModule">

<head>

<meta charset="UTF-8">

<title></title>

<link rel="stylesheet" href="css/bootstrap.css"/>

</head>

<body>

<h1>观察：Model的改变与View的更新间的关系</h1>

<div class="container" ng-controller="myCtrl">

<script>

var age1 = 10;

document.write(age1);

/\*age1++;

age1++;\*/

setInterval(function () {

age1++;

console.log('age1 is ' + age1)

}, 1000);

</script>

<p>{{age2}}</p>

<p ng-bind="age2"></p>

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myModule', [])

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

$scope.age2 = 20;

/\* $scope.age2++;

$scope.age2++;\*/

$scope.flagshow = true;

$interval(function () {

$scope.age2++;//只要在控制器中修改了模型数据，view会自动改变

}, 1000);

});

</script>

</body>

</html>

### 1.1.1. 【方向1】Model数据绑定到View

指令用于将模型数据绑定到view，进行显示。一旦绑定后，模型数据的修改，将自动更新到view并显示。

实现方法：{{Model变量名}}//（把Model变量绑定到View中） + 常用指令

常用的指令有：

ngApp：制定angularJS的范围

ngInit：定义变量

ngRepeat：设置循环

ngBind：将一个表达式的值绑定为当前元素的innerHTML

ngIf：设置条件判断

ngShow/ngHide/ngChecked

对比：

DOM操作方法：先查找元素，再操作元素

Angular：操作主要是放在Model数据：创建Model数据，修改Model数据

### 【方向1】 练习

1. 在View中显示一个一闪一闪的图片

//在HTML中加入下面一行代码

<img src="img/1.jpg" ng-if="flagshow" alt=""/>

//在控制器中加入以下代码

$scope.flagshow = true;//在定时器外设置

$scope.flagshow = !($scope.flagshow);

1. 实现一个点击次数记录程序

//在HTML中加入下边2行代码

<button ng-click="countNum()">统计点击次数</button>

<p>{{totalNum}}</p>

//在控制器中加入以下代码

$scope.totalNum = 0;

$scope.countNum = function ()

$scope.totalNum++;

}

1. 在一个ul中显示3个学生的成绩，点击“加载更多”按钮，则在View再显示3条数据

//HTML代码

<ul>

<li ng-repeat="num in list">{{num}}</li>

</ul>

<button ng-click="add()">添加更多</button>

//控制器代码

$scope.list = [10,20,30];

$scope.add = function () {

$scope.list.push(Math.round(Math.random()\*1000));

$scope.list.push(Math.round(Math.random()\*1000));

$scope.list.push(Math.round(Math.random()\*1000));

}

## 1.2. 方向2

### 1.2.1. 【方向2】View数据绑定到Model

所谓双向绑定，是指除了可以将 Model 中的数据绑定到 View，还可以将 View 中的数据绑定到 Model。比如，将文本框的值，绑定到模型变量中。

绑定：模型数据改变，view中的数据立即改变，所以称为绑定。Model中的数据映射到 view，称为方向1。

方向一：把Model数据绑定到View上——此后不论何时只要Model发生了改变，则View中的呈现会立即随之改变！实现方法：{{ }}、ngBind、ngRepeat、ngIf、ngSrc...都实现了方向1的绑定。

方向二：把View(表单控件)中修改绑定到Model上——此后不论任何时候，只要View中的数据一修改，Model中的数据会自动随之修改。实现方法：只有ngModel指令。

新建2.html

<!DOCTYPE html>

<html lang="en" ng-app="myApp">

<head>

<meta charset="UTF-8">

<title></title>

</head>

<body>

<h1>双向数据绑定方向2--View数据绑定到Model</h1>

<div ng-controller="myCtrl">

<h2>单行文本输入域的绑定</h2>

新闻搜索:<input type="text" ng-model="kw"/>//把当前输入域的值绑定到kw

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myApp',[]).controller('myCtrl',function ($scope) {

//监视Model数据的每一次修改

//注意，控制器中的kw是没有声明的

$scope.$watch('kw', function () {

console.log("keyword is " + $scope.kw);

})

})

</script>

</body>

</html>

**将文本框中的数据绑定到model，只使用 ngModel指令即可；也可以直接将多行文本框中的数据绑定到另一个控件，将单选按钮绑定到另一个控件；将下拉框的值绑定显示。**

### 1.2.2. 【方向2】$scope.$watch

可以使用$scope.$watch('模型变量名', fn)监视一个模型变量值的改变；

修改2.html

//HTML

<h2>多行文本输入域的绑定</h2>

您的留言内容：<br>

<textarea ng-model="userMsg"></textarea>

<h2>下拉框的绑定</h2>

所在城市：

<select ng-model="location">

<option value="bj">北京</option>

<option value="sh">上海</option>

<option value="hb">河北</option>

</select>

<h2>复选框的绑定</h2>

电视:<input type="checkbox" valute="tv" name='from' ng-model="fromTV"/>

<h2>单选框的绑定</h2>

性别

<br/>

男<input type="radio" name="sex" value="male" ng-model="userSex">

女<input type="radio" name="sex" value="female" ng-model="userSex">

//控制器JS

$scope.$watch('userMsg', function () {

console.log('userMsg is '+$scope.userMsg);

})

$scope.$watch('location', function () {

console.log('locaiton choose is '+$scope.location)

})

$scope.$watch('fromTV', function () {

console.log('fromTV choose is '+$scope.fromTV)

})

$scope.$watch('userSex', function () {

console.log('userSex is ' + $scope.userSex);

})

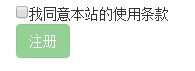
单行文本输入域、多行文本输入域、下拉框、单选按钮控件默认会把自己的value属性值绑定到一个Model变量；复选框会把一个true/false值绑定到一个Model变量。

$watch是一个scope函数，用于监听模型变化，当你的模型部分发生变化时它会通知你。事实上，angular中的每一个表达式，之所以能实现双向数据绑定，都是因为底层启动了一个watch队列，监听变化。

**使用watcher 监视变量。**

## 1.3 双向绑定练习

### 1.3.1. 只有勾选了“我同意。。” 才能点击“注册按钮”(ngIf ngDisabled)



### 1.3.2. 据性别选择，网页背景色呈现不同的变化(ngClass)



### 1.3.3. 选择头像，立即看到图片变化(ngSrc)



### 1.3.4. 全选/取消去选





新建3.html

<!DOCTYPE html>

<html lang="en" ng-app="myModule">

<head>

<meta charset="UTF-8">

<title></title>

<link rel="stylesheet" href="css/bootstrap.css"/>

<style>

.red {

background: #d43f3a;

}

.blue {

background: #00aafa;

}

</style>

</head>

<body ng-class='pageTheme' ng-controller="myCtrl">

<div class="container">

<table class="table">

<thead>

<tr>

<th>请选择</th>

<th>姓名</th>

<th>生日</th>

</tr>

</thead>

<tbody>

<tr>

<td>

<input type="checkbox" ng-checked="selectAll"/>

</td>

<td>

King

</td>

<td>

2015-12-22

</td>

</tr>

<tr>

<td>

<input type="checkbox" ng-checked="selectAll"/>

</td>

<td>

Zang

</td>

<td>

2015-03-11

</td>

</tr>

</tbody>

</table>

<input type="checkbox" ng-model="selectAll"/>全选/取消全选

<br/><br/><br/>

请选择头像：

<br/>

<select ng-model="photo">

<option value="1.jpg">大叔</option>

<option value="2.jpg">大婶</option>

<option value="3.jpg">萌妹</option>

<option value="4.jpg">鲜肉</option>

</select>

<br/>

<br/>

<img ng-src="img/{{photo}}"/>

<br/><br/>

请选择性别：

<br/>

<input type="radio" name='sex' value="blue" ng-model="pageTheme"/>男

<input type="radio" name='sex' value="red" ng-model="pageTheme"/>女

<br/>

<input type="checkbox" value="agree" ng-model="isAgree"/>我同意本站的使用条款

<br/>

<!-- 在这里开始讲 ng-hide ng-show }-->

<button ng-disabled='isShow' class="btn btn-success">注册</button>

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myModule', []).controller('myCtrl', function ($scope) {

$scope.$watch('isAgree', function () {

$scope.isShow = !$scope.isAgree;

})

$scope.$watch('pageTheme', function () {

console.log('sex choose is ' + $scope.pageTheme);

})

$scope.nowSrc = "img/1.jpg";

$scope.$watch('photo', function () {

})

})

</script>

</body>

</html>

# 2. 过滤器和函数

## 2.1. 过滤器

豆浆机的案例

过滤器可以用于表达式中，实现对表达式结果的筛选、过滤或者格式化，达到更好的表现效果。其本质也是一个函数，位于手册里的filter部分。

过滤器的语法如下：

{{expression|filter|filter:parameter1:parameter2}}

“|”表示传递数据的管道，可以把数据逐一传递，实现多种过滤。

新建4.html

<!DOCTYPE html>

<html ng-app="myModule1">

<head lang="en">

<meta charset="UTF-8">

<title>过滤器</title>

<link rel="stylesheet" href="css/bootstrap.css"/>

</head>

<body>

<div ng-controller="c1" ng-class="containerBg" class="panel panel-default">

<div class="panel-body" ng-controller="c1">

<h4>过滤器</h4>

数量：{{quantity|number:2}}<br>

价格：{{price|currency:'￥'}}<br>

公司：{{company|uppercase}}<br>

日期：{{pDate|date:'yyyy-MM-dd HH:mm:ss'}}

<hr>

按薪水排名

<ul>

<li ng-repeat="e in arr | orderBy : 'salary'">{{e}}</li>

</ul>

<br/>

按薪水排名且是降序

<ul>

<li ng-repeat="e in arr | orderBy : 'salary' : true">{{e}}</li>

</ul>

按姓名排序

<ul>

<li ng-repeat="e in arr | orderBy : 'name'">{{e}}</li>

</ul>

只显示前两个

<ul>

<li ng-repeat="e in arr |limitTo : 2">{{e}}</li>

</ul>

按元素的大小排名

<ul>

<li ng-repeat="e in arr1 |orderBy : e">{{e}}</li>

</ul>

</div>

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myModule1', ['ng']).controller('c1', function($scope){

$scope.quantity = 321579;

$scope.price = 4.5;

$scope.pDate = new Date();

$scope.company = "tarena";

$scope.arr = [

{'name':'mary','salary':3456},

{'name':'tom','salary':8902},

{'name':'jerry','salary':1000},

{'name':'heny','salary':3200}

];

$scope.arr1 = [12,43,1,678,9];

})

</script>

</body>

</html>

### 2.1.1. 【过滤器】number过滤器

格式化数字到文本，如果输入不是一个数字，返回空字符串。可以提供参数，实现带小数点的十进制数字，数字默认带有千分位的分隔符。

### 2.1.2. 【过滤器】currency过滤器

格式化数字为货币形式 (如 $1,234.56)。当提供的不是货币形式时，会使用本地化默认形式。参数为表示货币的符号字符。

### 2.1.3. 【过滤器】date过滤器

格式化date到字符串，基于format的要求。比如yyyy表示年份，MM表示月份，dd表示日期等。

### 2.1.4. 【过滤器】upperCase过滤器

### 2.1.5. 【过滤器】lowerCase过滤器

### 2.1.6. 【过滤器】orderBy过滤器

通过expression来排序指定的array 。字符串按字母顺序排序，数字按大小排序。注意：如果你发现数字没被正确排序，请确认它们保存的是数字而不是字符串。

## 2.2. 函数

函数是指NG提供的一些对应于常用 JS 操作的功能。

### 2.2.1. 【函数】forEach

实现循环操作。

### 2.2.2. 【函数】lowerCase 和 upperCase

实现字符串的大小写转换。

### 2.2.3. 【函数】toJson 和 fromJson

实现json串和对象的转换。

**新建5.html**

<!DOCTYPE html>

<html ng-app="myModule1">

<head lang="en">

<meta charset="UTF-8">

<title>函数</title>

<link rel="stylesheet" href="css/bootstrap.css"/>

</head>

<body>

<div ng-controller="c1" ng-class="containerBg" class="panel panel-default"></div>

<script src="js/angular.js"></script>

<script>

angular.module('myModule1', ['ng']).controller('c1', function($scope){

var values = [

{name: 'misko', gender: 'Male'},

{name: 'mary', gender: 'Female'}

];

angular.forEach(values, function(value, key) {

console.log(key + ': ' + angular.uppercase(value.name) + "," + angular.lowercase(value.gender));

});

console.log(angular.toJson({name: "misko", gender: "Male"}));

var obj = angular.fromJson('{"name":"mary"}');

console.log(obj.name);

})

</script>

</body>

</html>

# 3. 服务

## 3.1. 控制器和作用范围

新建6.html

<!DOCTYPE html>

<html ng-app="myModule1">

<head lang="en">

<meta charset="UTF-8">

<title>作用域范围</title>

<link rel="stylesheet" href="css/bootstrap.css"/>

</head>

<body>

<h2>控制器的作用域范围</h2>

<div ng-controller="c1" class="container">

{{name}}

</div>

<div ng-controller="c2" class="container">

{{name}}

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myModule1', ['ng']).controller('c1', function ($scope) {

$scope.name = "mary";

}).controller('c2', function ($scope) {

$scope.name = "jerry";

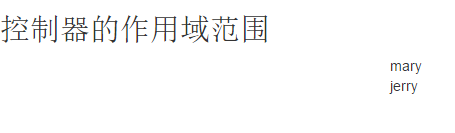
})

</script>

</body>

</html>

从上边的代码可以看到：



### 3.1.1. 【控制器和作用范围】$scope

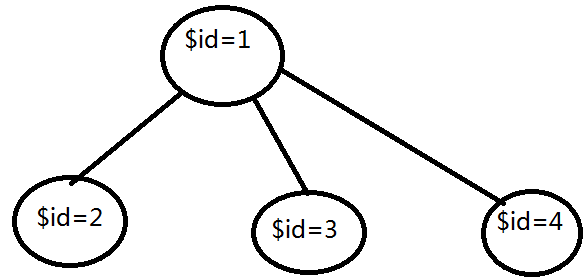
每个控制器的实例都对应一个作用范围对象，即$scope，在控制器中声明的Model数据，必须保存在一个作用范围内。

一个HTML中可以声明多个控制器实例，每个控制器都有自己的作用范围，这些范围内的数据彼此隔离不会互相影响，可以由不同开发人员来编写，保证良好的扩展性。

### 3.1.2. 【控制器和作用范围】$rootScope

每个控制器对象都有一个作用范围，每个作用范围都对应一个$scope对象，有着唯一的ID值，此 ID 值从 2 开始，ID值为1的是全局的 scope 对象。

若某个控制器想创建一个Model数据，让其它所有的控制器都可以共享，只需要保存在$id=1的那个根作用域中即可——$rootScope。整个AngularApp中有且只有一个$rootScope对象，且此对象是所有的$scope的父作用域对象。其结构如下图所示：



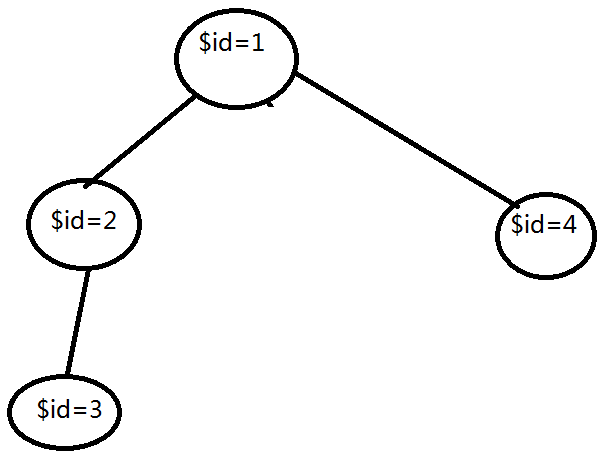
一般情况下，每个控制器作用范围内只能使用当前$scope中保存的Model数据。若需要让控制器2可以使用控制器1中的Model数据，只需要把控制器2声明在控制器1内部即可，代码如下所示：

<div ng-controller="c1">

<div ng-controller="c2"></div>

</div>

其结构如下图所示：



如果根范围和子范围中，有重复的变量，如何计算？私有的会覆盖父范围的变量。因此，对于这些作用域之间的关系而言，是属于可以相互包含的。作用域对象间可以实现继承，只需要将某个控制器实例声明在另一个控制器实例的有效作用域内部即可。

**面试题：$scope和$rootScope的区别。**

**6.html最终代码如下：**

<!DOCTYPE html>

<html ng-app="myModule1">

<head lang="en">

<meta charset="UTF-8">

<title>作用域范围</title>

<link rel="stylesheet" href="css/bootstrap.css"/>

</head>

<body>

<h2>控制器的作用域范围</h2>

<div ng-controller="c1" class="container">

姓名:{{name}}

<br/>

学校:{{school}}

</div>

<div ng-controller="c2" class="container">

姓名:{{name}}

<br/>

学校:{{school}}

<div ng-controller="c3">

{{name}}

</div>

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myModule1', ['ng']).controller('c1', function ($scope,$rootScope) {

$scope.name = "mary";

$rootScope.school='Tarena';

}).controller('c2', function ($scope) {

$scope.name = "jerry";

}).controller('c3', function ($scope) {

$scope.name='noJerry';

})

</script>

</body>

</html>

## 3.2. 定时器

### 3.2.1. 【定时器】$interval

周期性定时器。

### 3.2.2. 【定时器】$timeout

一次性定时器。

**新建7.html**

**先手工实现原生JS定时器（需要使用$scope.$apply）、ng的timeout定时器；**

**然后实现自增（$interval）**

**然后实现图片轮播（$interval）**

**时间允许的话，可以实现进度条（ng-style）**

<!DOCTYPE html>

<html lang="en" ng-app="myModule">

<head>

<meta charset="UTF-8">

<title></title>

<link rel="stylesheet" href="css/bootstrap.css"/>

<style>

[ng\:cloak], [ng-cloak], [data-ng-cloak], [x-ng-cloak], .ng-cloak, .x-ng-cloak {

display: none !important;

}

</style>

</head>

<body>

<h2>定时器</h2>

<div ng-controller="myCtrl">

<h2>原生JS操作</h2>

<button ng-click="showMsg1()">2s后显示数据</button>

<p>{{msg1}}</p>

<h2>angular定时器</h2>

<button ng-click="showMsg2()" >2s后显示数据</button>

<p ng-cloak>{{msg2}}</p>

<h2>用ng提供的周期性定时器，来实现一个num自增1，并显示在段落中</h2>

<p>{{num}}</p>

</div>

<div ng-controller="myCtrl2">

<img ng-src="img/{{src}}" alt=""/>

</div>

<h2>进度条 not needed</h2>

<div class='progress' ng-controller="myCtrl3">

<div ng-style="myStyle" class="progress-bar"></div>

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myModule', [])

.controller('myCtrl', function ($scope,$timeout,$interval) {

$scope.msg1 = '';

$scope.msg2 = '';

$scope.num = 0;

$scope.showMsg1 = function () {

setTimeout(function () {

$scope.msg1 = '我是在点击按钮2s后显示的'

console.log('原生JS操作后msg为:'+$scope.msg1);

// $scope.$apply();//刚开始不写这一行，发现定时器更改后没有结果。然后再写这样，发现就可以了

},2000);

}

$scope.showMsg2 = function () {

$timeout(function () {

$scope.msg2 = '我是在按钮点击2s后显示的';

console.log('msg changed');

},2000);

}

var t = $interval(function () {

$scope.num++;

if($scope.num >= 20)

{

$interval.cancel(t);

}

},1000);

}).controller('myCtrl2', function ($scope,$interval) {

list = ['1.jpg','2.jpg','3.jpg','4.jpg'];

$scope.src = '1.jpg';

var index = 0;

$interval(function () {

index++;

if(index == 4)

{

index = 0;

}

$scope.src = list[index];

console.log('src is '+$scope.src);

},1000);

}).controller('myCtrl3', function ($scope,$interval) {

var percentage = 0;

$scope.myStyle = { width:'0%' };

var t = $interval(function(){

percentage += 10;

$scope.myStyle.width = percentage+'%';

if(percentage>=100){

$interval.cancel(t);

}

}, 200)

});

</script>

</body>

</html>

### 3.2.3. 【控制器和作用范围】$scope.$digest

### 3.2.4. 【控制器和作用范围】$scope.$apply

用于实现手工触发$watch队列中的所有的监听函数。当在controller以外的范围修改模型后，需要手动调用此函数，激发绑定。推荐使用 apply 方法。

## 3.3. AJAX调用

### 3.3.1. 【AJAX调用】$http服务

ng模块中提供的 $http 服务可以用于向服务器发起AJAX请求，异步的获取服务器端返回的响应数据；声明为模型数据后，即可在视图中加以呈现

**新建文件夹data，然后新建a.json**

[

{

"name":"mary",

"age":20,

"birthday":"1989-03-22"

},

{

"name":"jack",

"age":29,

"birthday":"1988-03-22"

},

{

"name":"jerry",

"age":24,

"birthday":"1980-03-22"

}

]

**新建8.html**

<!DOCTYPE html>

<html lang="en" ng-app="myApp">

<head>

<meta charset="UTF-8">

<link rel="stylesheet" href="css/bootstrap.css"/>

<title></title>

</head>

<body>

<div ng-controller="myCtrl">

<h2>访问服务器的JSON文件</h2>

<button ng-click="getData()">加载数据</button>

<table class="table bg-info">

<thead>

<tr>

<th>姓名</th>

<th>年龄</th>

<th>生日</th>

</tr>

</thead>

<tbody>

<tr ng-repeat="d in arr | orderBy:'age':true">

<td>{{d.name}}</td>

<td>{{d.age}}</td>

<td>{{d.birthday}}</td>

</tr>

</tbody>

</table>

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myApp', []).controller('myCtrl', function ($scope, $http) {

$scope.arr = [];

$scope.getData = function () {

$http.get('data/a.json').success(function (data) {

list = data;

for (var k = 0;

k < data.length;

k++

) {

$scope.arr.push(list[k]);

console.log("length is "+$scope.arr.length)

}

});

}

})；

</script>

</body>

</html>

**时间允许的话，$http练习一下从php文件中读取数据。**

**新建server文件夹，然后新建1.php如下：**

<?php

header('Content-Type:application/json');

$stuArray = [];

$stuArray[] = [

'name'=>'mary',

'age'=>19

];

$stuArray[] = [

'name'=>'tom',

'age'=>20

];

$str = "";

$str = json\_encode($stuArray);

echo $str;

?>

**新建9.html（把8.html拷贝过来，稍作修改即可）**

<!DOCTYPE html>

<html lang="en" ng-app="myApp">

<head>

<meta charset="UTF-8">

<link rel="stylesheet" href="css/bootstrap.css"/>

<title></title>

</head>

<body>

<div ng-controller="myCtrl">

<h2>访问服务器的php文件</h2>

<button class='btn btn-success' ng-click="getData()">加载数据</button>

<table class="table">

<thead>

<tr>

<th>姓名</th>

<th>年龄</th>

</tr>

</thead>

<tbody>

<tr ng-repeat="obj in arrayList">

<td>{{obj.name}}</td>

<td>{{obj.age}}</td>

</tr>

</tbody>

</table>

</div>

<script src="js/angular.js"></script>

<script>

angular.module('myApp', []).controller('myCtrl', function ($scope, $http) {

$scope.arrayList = [];

$scope.getData = function () {

$http.get('server/1.php').success(function (data) {

console.log('data is '+data);

$scope.arrayList = data;

}).error(function (errMsg) {

console.log('sth is wrong:'+errMsg)

});

}

})

</script>

</body>

</html>

# 4.每日一练

获取json数据，初始化显示；  
添加数据：价格为100以内的随机小数，数量为10因为的随机整数  
删除：点击删除按钮，可以删除当前行 4、计算小计和总计，并格式化输出

