



## 第五讲 控制器和作用域

# 教学目标

---

1. 使用控制器
2. 控制器的作用域
3. 控制器继承

1

使用控制器

# ①

## 控制器

为什么及何时使用控制器？

```
<body ng-init="arrs=[{name:'张三',age:22},{name:'李四',age:16},{name:'赵五',age:30},{name:'郑六',age:10}]" >  
<ul>  
  <li ng-repeat="arr in arrs">{{arr}}</li>  
</ul>
```

编写大量的业务逻辑时，使用控制器是不合适的


# ②


## 定义控制器


定义控制器:

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

```
app . controller ( ' ctrlName ' , function( $scope ) {  
    //somecode;  
} );
```

 **定义控制器**

 **控制器名称**

 **控制器处理函数**

[Demo:ex01.html](#)

# 3

## 调用控制器

**ng-controller** 指令——应用控制器

**ng-controller** 指令必须在父级元素上添加

```
<ul ng-controller="ctrlName">  
  <li ng-repeat="arr in arrs">{{arr}}</li>  
</ul>
```

The diagram illustrates the mapping of two AngularJS directives to their functional components. A red line connects the `ng-controller` directive in the `<ul>` tag to the text '应用控制器' (Application Controller). A blue line connects the `ng-repeat` directive in the `<li>` tag to the text '控制器名称' (Controller Name).

应用控制器

控制器名称

# 4

## 外部文件中控制器

在大型的应用程序中，通常是把控制器存储在外部文件中。

- 1、创建外部js文件，所有控制器代码写在文件里。
- 2、引入外部js文件既可。

```
<script src=controller.js"> </script>
```

Demo:ex02.html

# 2

## 控制器的作用域



# ①

## 什么是作用域

- ◆ 作用域是一个指向应用模型的**对象**。
- ◆ 它是表达式的**执行环境**。
- ◆ 作用域**有层次结构**，这个层次和相应的DOM几乎是一样的。
- ◆ 作用域**能监控表达式和传递事件**。

`ng-controller`和`ng-repeat`会创建新的作用域，并关联到相应的DOM元素上，可以使用`angular.element(aDomElement).scope()`方法来获得某一个DOM元素相关的作用域。

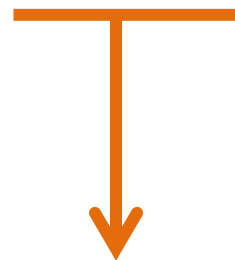
# ①

## 作用域——\$scope

在创建控制器时，\$scope 对象当作一个参数传递

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

```
app . controller ( ' ctrlName ' , function( $scope ) {  
    //somecode;  
} );
```



控制器的作用域对象

## ②

# 设置作用域

**\$scope** 是应用在视图(html)和控制器(js)之间的纽带。

```
var app=angular.module('test',[]);  
app.controller('ctrl1',function($scope){  
    $scope.username='用户名';  
})
```

```
<form ng-controller="ctrl1">  
    <input type="text" value="{{username}}">  
</form>
```

Demo:ex03.html

## ②

# 设置作用域

**\$scope** 是一个对象，作用域内的变量都是它的属性。

```
var app=angular.module('test',[]);
app.controller('ctrl1',function($scope){
    alert($scope.aa);
})
```

```
<body ng-init="aa=12" >
<form ng-controller="ctrl1">
    <input type="text" value="username">
</form>
</body>
```

Demo:ex03.html

## ②

# 设置作用域

在控制器中添加 `$scope` 对象时，视图可以获取了这些属性。

视图中不需要添加 `$scope` 前缀，只需要添加属性名即可。

如：`{{username}}`。

Demo:ex04.html

# 3

## 修改作用域

模型：

```
$scope.cities = ["London", "New York", "Paris"];  
$scope.city = "London";  
$scope.getCountry = function (city) {  
    switch (city) {  
        case "London":  
            return "UK";  
        case "New York":  
            return "USA";  
    }  
}
```

Demo:ex05.html

# 3

## 修改作用域

视图：

```
<div class="well">
  <label>Select a City:</label>
  <select ng-options="city for city in cities"
ng-model="city">
    </select>
</div>
<div class="well">
  <p>The city is: {{city}}</p>
  <p>The country is: {{getCountry(city) ||
"Unknown"}}</p>
</div>
```

Demo:ex05.html

## 4

# 组织作用域

## 1、使用一个单块控制器

```
angular.module("exampleApp", [])  
  .controller("simpleCtrl", function ($scope) {  
    $scope.addresses = {};  
    $scope.setAddress = function (type, zip) {  
      console.log("Type: " + type + " " + zip);  
      $scope.addresses[type] = zip;  
    }  
    $scope.copyAddress = function () {  
      $scope.shippingZip = $scope.billingZip;  
    }  
  });
```

Demo:ex06.html



## 4

# 组织控制器

## 2、复用控制器

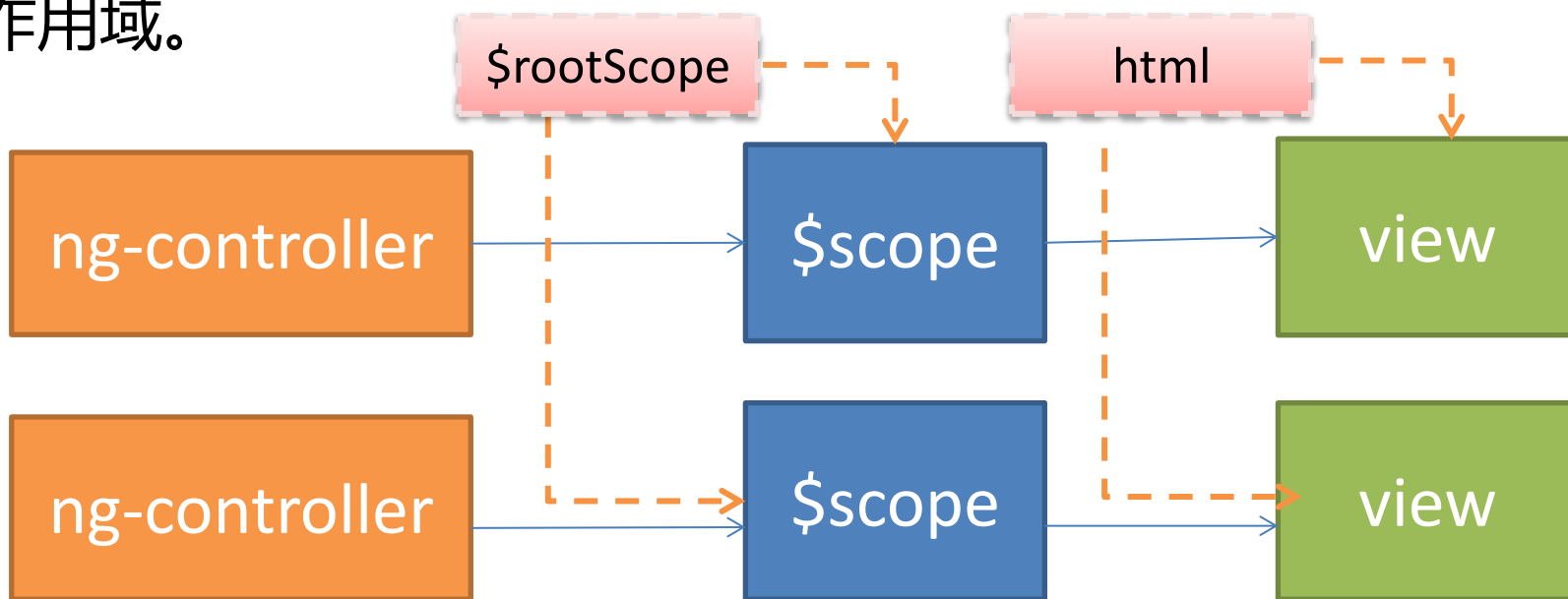
Demo:ex07.html

```
<div class="well" ng-controller="simpleCtrl">
  <h4>邮政编码</h4>
  <div class="form-group">
    <input class="form-control" ng-model="zip">
  </div>
  <button class="btn btn-primary" ng-click="setAddress('邮政编码', zip)">
    保存邮政编码
  </button>
</div>
<div class="well" ng-controller="simpleCtrl">
  <h4>航运邮编</h4>
  <div class="form-group">
    <input class="form-control" ng-model="zip">
  </div>
  <button class="btn btn-primary" ng-click="copyAddress()">
    使用邮编
  </button>
  <button class="btn btn-primary" ng-click="setAddress('航运邮编', zip)">
    保存航运邮编
  </button>
</div>
```

# 4

## 作用域层级

作用域层级结构是组织应用的数据的一种方式。应用拥有一个绝对的根作用域(ng-app)。但是可以有多个子作用域。



ng-controller创建\$scope，每个控制器都会赋予一个新的作用域，该作用域作为根作用域的一个子作用域。

## 4

## 作用域之间的通信

根作用域可以作为一个服务被使用，控制器中使用`$rootScope`声明对它的依赖，所有的作用域都定义了可以用于发送和接收事件的方法。

方法	描述
<code>\$broadcast(name,args)</code>	向当前作用域下的所有子作用域发送一个事件。参数是事件名称以及一个用于向事件提供额外数据的对象
<code>\$emit(name,args)</code>	向当前作用域的父作用域发送一个事件，直至根作用域
<code>\$on(name,handler)</code>	注册一个事件处理函数，该函数在特定的事件被当前作用域收到时将会被调用。

# 4

## 作用域之间的通信

Demo:ex08\_a.html

- **\$emit** 只能向parent controller传递event与data
- **\$broadcast** 只能向child controller传递event与data
- **\$on** 用于接收event与data

\$emit和\$broadcast可以传多个参数（使用json），\$on也可以接收多个参数。

## 4

## 范例： Demo:ex08\_b.html

```
angular.module("exampleApp", [])  
  .controller("simpleCtrl", function ($scope, $rootScope) {  
    $scope.$on("zipCodeUpdated", function (event, args) {  
      $scope[args.type] = args.zipCode;  
    });  
    $scope.setAddress = function (type, zip) {  
      $rootScope.$broadcast("zipCodeUpdated", {  
        type: type, zipCode: zip  
      });  
      console.log("Type: " + type + " " + zip);  
    }  
    $scope.copyAddress = function () {  
      $scope.zip = $scope.billingZip;  
    }  
  });
```

## 5

# 作用域的更新

\$scope提供\$apply( )方法传递Model的变化。

\$scope提供\$digest( )方法触发循环队列。

\$scope提供\$watch( )方法监视Model的变化。



Demo:ex09.html

# 5

## 作用域的更新

`$scope.$apply()` 会自动地调用 `$rootScope.$digest()`

什么时候手动调用 `$apply()` 方法？

使用 JavaScript 来更新一个scope model时，AngularJS就不能获取更改了，这种情况下需要调用 `$apply()` 方法。

```
setTimeout(function() {  
    $scope.message = '3秒后执行';  
    console.log('信息:'+$scope.message);  
    $scope.$apply(); //这里触发了一个$digest  
}, 3000);
```

Demo:ex10.html

# 6

## AngularJS计时器

试一试：

- 1、AngularJS中创建和取消延迟执行？
- 2、AngularJS中创建和取消周期执行？

**var id1=\$timeout( )** —————> **\$timeout.cancel( id1)**

**var id2=\$interval( )** —————> **\$interval.cancel( id2)**

Demo:ex11.html



# 3

## 控制器继承

1

# 使用控制器继承

一个按钮控制三个输入框

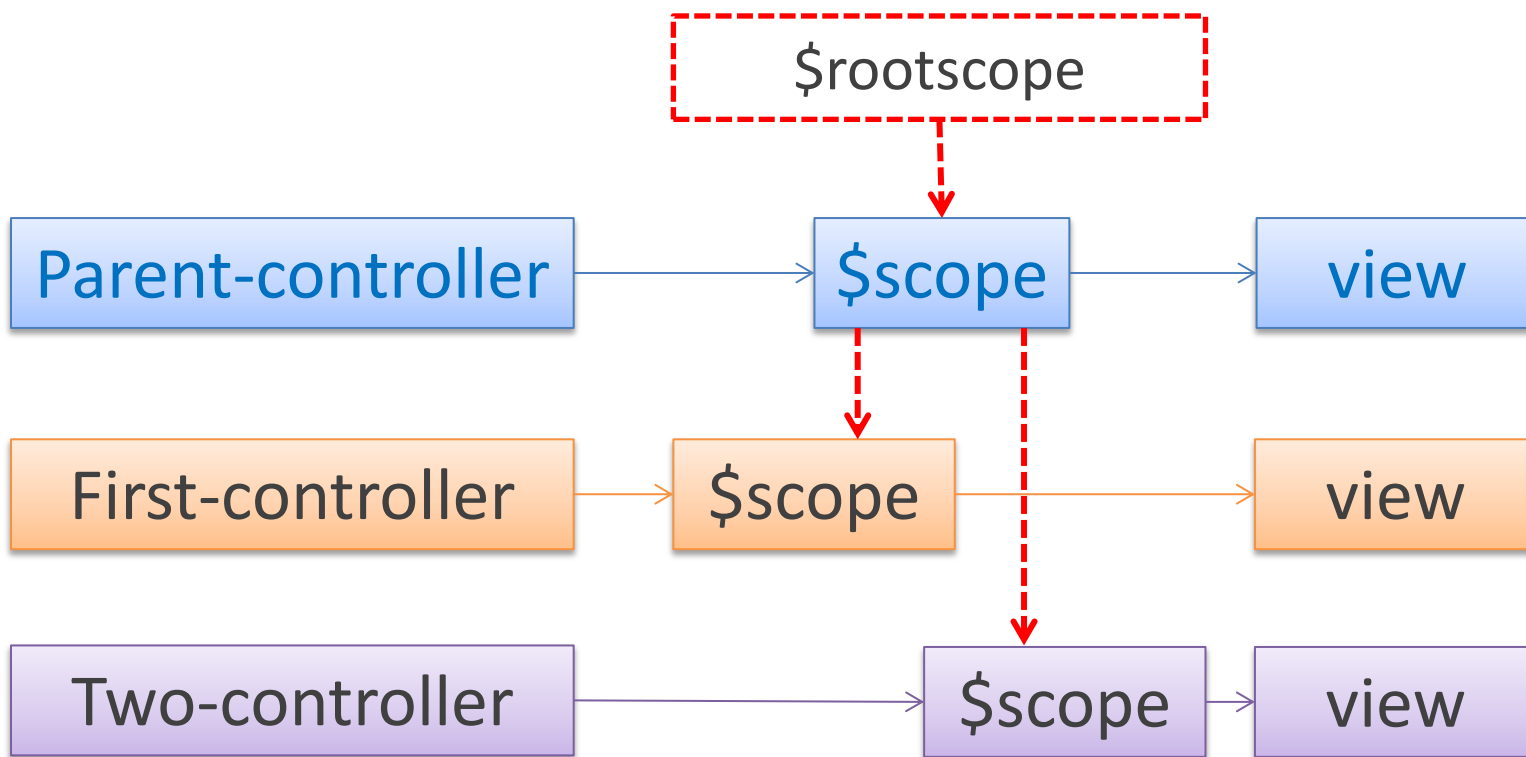


Demo:ex12.html

# 1

## 使用控制器继承

子控制器继承父控制器中的功能。作用域层次如下：

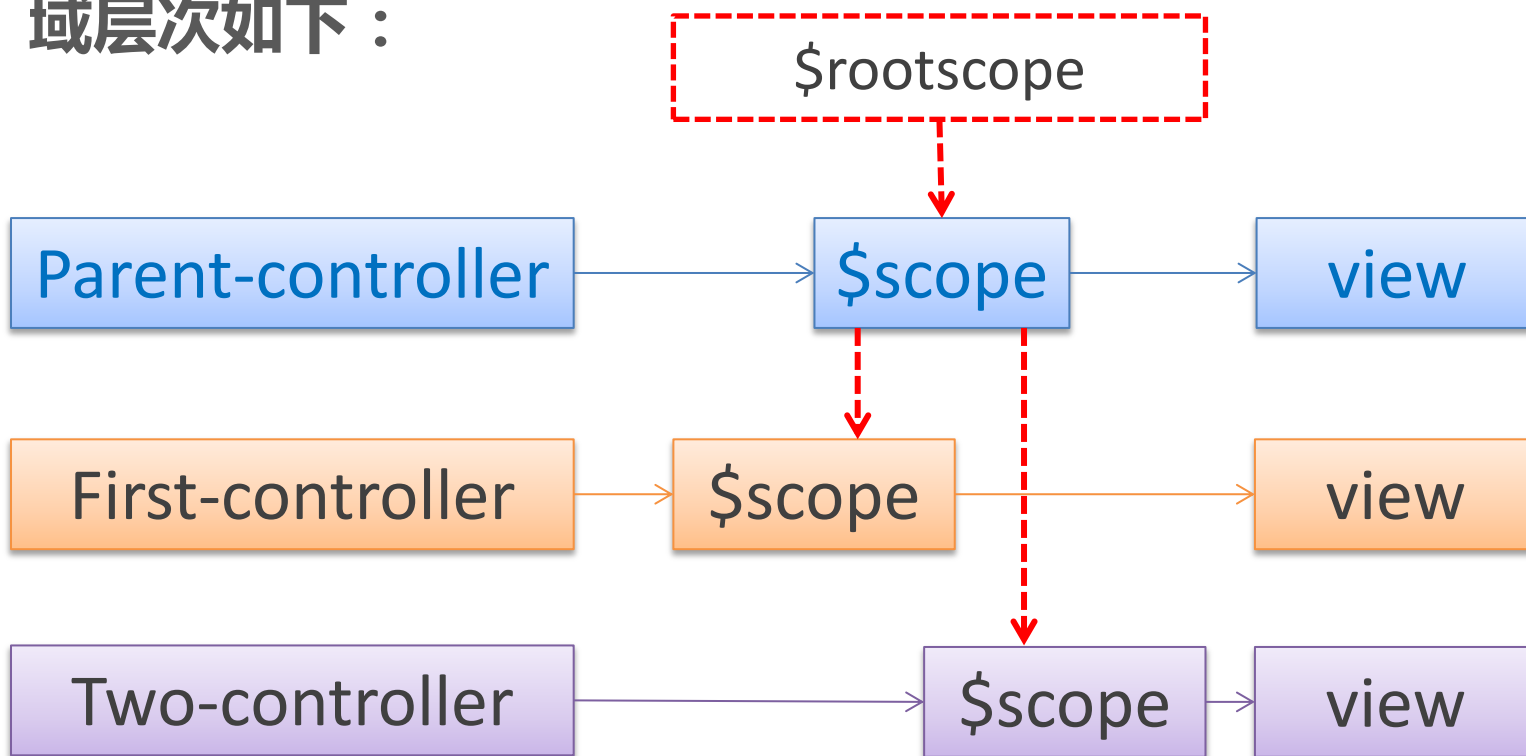


Demo:ex12.html

# 1

## 使用控制器继承 ???

子控制器以**原型**方式继承父控制器中的功能。作用域层次如下：

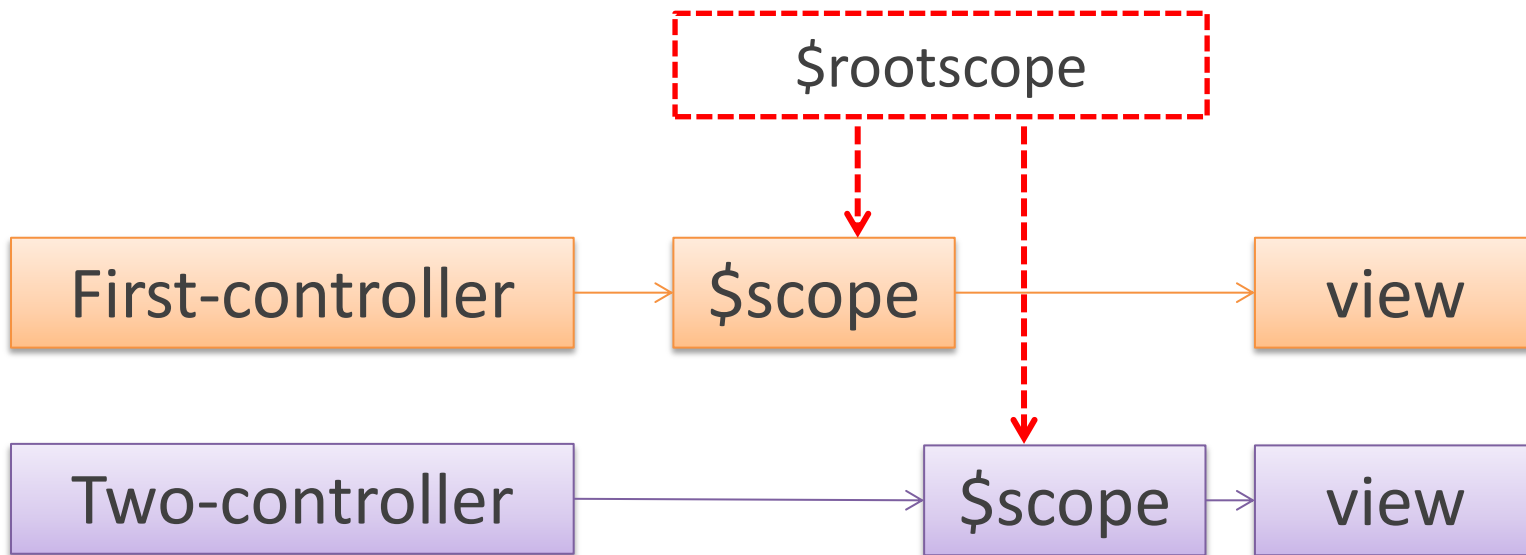


Demo:ex13.html

## ②

# 使用多控制器

在AngularJS中可以定义多个控制器



Demo:ex14.html

# 3

## 无作用域的控制器

若应用程序无需继承，也无需在控制器间通信，那么可以使用无作用域控制器。

```
var app = angular.module("exampleApp", [])
    .controller("simpleCtrl", function () {
        this.dataValue = "Hello, Adam";
        this.reverseText = function () {
            this.dataValue =
this.dataValue.split("").reverse().join("");
        }
    });
```

Demo:ex15.html

# 本课小结

---

## 1. 使用控制器

ng-controller

## 2. 控制器的作用域—— \$scope ( 层级 )

作用域通信 : \$on(), \$emit(), \$broadcast()

作用域更新 : \$apply() , \$digest() , \$watch()

定时器 : \$interval() , \$timeout()

## 3. 控制器继承

原型继承

# TNAKS

主讲：王智娟

QQ: 24132228

Email: wangzhijuan@onest.net