AngularJS 高级(二)

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

- Directive 的使用
- 常用服务

一、Directive(指令) 的使用(一)

1. Directive 的介绍
2. Directive 的使用
3. restrict 、template
4. replace 、templateUrl,$templateCache

Directive 的介绍

- Directive 是 AngularJS 的一个特殊标志,也是有别于其他框架的一个重要特征, AngularJS 之所以功能强大,在很大程度上得益于它拥有大量内置的指令
- 前面所学习的指令都属于 AngularJS 内置的指令,除此之外, AngularJS 也能通过语法自定义指令
- 通过自定义指令,使Html更具语义化，不需要深入研究代码和逻辑即可知道页面的大致逻辑
- 通过 Directive，可以封装很多公共指令,也能达到重用的效果

Directive 的使用

- 在 Angular 中,定义一个新的指令方法非常简单,只需要由模块调用 directive 方法即可,该方法接收两个参数
- name 表示指令的名称
- fn 是一个函数,返回一个对象,此对象中,定义了新增指令的全部行为

```
var app1 = angular.module('myApp',[]);
//自定义指令
app1.directive(name,fn);
```

restrict 、 template


```
//自定义指令
app1.directive('nsHello',function () {
    return {
        restrict: 'ECMA',
        template: '<div>hello,男神</div>'
    };
});
```

js 文件

```
<div ng-app = "myApp">
    <ns-hello></ns-hello>
</div>
```

html文件

运行结果:

hello,男神

restrict 、 template

- 在 html 文件中,使用了一个名称为 ns-hello 的新指令,此指令为用户自定义指令,调用了 directive 方法
- 方法中的第一个参数就是新指令的名称,参数中的名称必须使用驼峰命名风格,一次,名称分为两个部分,前部分是指令前缀,例如内置的指令为 ng ,为了避免冲突,可以使用项目名,公司名的缩写字母;后面部分是指令的作用,一般说明指令当前的功能

restrict

- 在第二个参数函数的返回对象中,添加了两个简单的属性,restrict 属性之处指令在 HTML 中的使用方式,共有 E、 C 、 M 、 A 四种,分别代表标签、属性、类和注释,属性值可以是单个字母,也可以是多个字母组合,默认值为 A

```
<div ng-app = "myApp">
    <!-- 标签元素使用 -->
    <ns-hello></ns-hello>
    <!-- 类使用 -->
    <div class="ns-hello"></div>
    <!-- 属性使用 -->
    <div ns-hello></div>
    <!-- 注释使用 -->
    <!--  directive: ns-hello -->
</div>
```

template

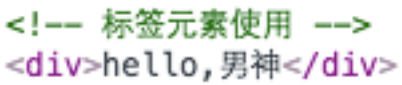
- 第二个属性名为 template(模板),表示指令在编译和链接后生成的 HTML 标记,即在调用新创建的指令后,在 HTML 中展现出来的 DOM 元素内容
- template 的属性值可以是字符串,可以是很复杂的字符集合,还可以包含大括号,动态获取变量的值
- template 的属性值最常见的是一段 HTML 文本 或者 一段函数,当值为函数的时候,可以接受两个参数,ele 和 attr ,ele 指的是使用此指令的元素,attr 指的是实例的属性,是一个由元素上所有的属性组合成的对象

```
template: function(ele,attr){
        console.log(ele);
         var _html = '';
         _html += '<div>' +'hello '+attr.title+'</div>';
         return _html;
        } ,
```

replace

- 此返回对象可以设置 replace 属性,属性值为 true 或者 false ,默认为 false ,当设置为 true 时,可以隐藏指令标签

replace 为 false



replace 为 true

templateUrl

- 此返回对象可以设置 templateUrl 属性,属性值为 字符串 或者 函数,当为字符串的时候,一般代表 HTML 文件路径的字符串,函数与 template 属性值类似
- 当传入的值为 HTML 文件路径时,html 文件必须有标签
- 在使用 chrome 浏览器时,同源策略会阻止chrome从file://中加载模板,要将项目放到服务器中去运行(直接在 WebStorm 运行即可)
- 也可以使用 script 标签设置模板,使用时只需要将 templateUrl 值设置为 script 标签里面的 id 属性值 (一定要将此标签的内容放置在 ng-app 做用范围内)

```
<script type="text/ng-template" id = "test2">
    <div>这是标签里面的模板</div>
</script>
```

$templateCache

- 可以将一段 HTML 文本通过 $templateCache 缓存起来,需要使用的时候直接取出来使用
- 如果是用 templateUrl 取出缓存内容,可直接根据缓存名称取
- 如果是用 template 取出缓存内容,需要注入 $templateCache 服务,使用 $tempalteCache.get(缓存名称) 取出缓存内容

```
//初始化全局数据
app1.run(function ($templateCache) {
    //缓存模板
    $templateCache.put('cache','<h2>模板内容来源于缓存</h2>');
});
```

缓存名称

内容

一、Directive(指令) 的使用(二)

1. transclude、priority、terminal
2. link
3. 指令的执行机制
4. compile

transclude

- 当自定义指令作为标签使用时,是允许标签嵌套的,但是模板里面的内容会覆盖嵌套标签里面的内容,可以通过设置 transclude 属性来保留嵌套标签里面的内容
- transclude 属性默认为 false,当其设置为 true 之后,要想显示嵌套标签里面的内容,必须培训 ng-transclude 指令使用

```
template: '<div>hello,男神<div ng-transclude></div></div>',
```

```
<!-- 标签元素使用 -->
▼ <div>
    "hello,男神"
  ▼ <div ng-transclude>
      <span class="ng-
      scope">嵌套标签内
      容</span>
    </div>
  </div>
```

priority、terminal

- priority :指明指令的优先级，若在单个DOM上有多个指令，则优先级高的先执行,默认为0,哪个先定义哪个先使用
- 设置优先级的情况较少,比如 ng-repeat,在遍历元素的过程中,需要 Angular 先拷贝生成的元素,再应用其他指令 (类似于 ng-click),所以 ng-repeat 默认的 priority 是1000
- 在同一个标签中,不能同时使用多个指令作为属性
- terminal :可以被设置为true或false，若设置为true，则优先级低于此指令的其他指令则无效，不会被调用(优先级相同的还是会执行),一般也不用

link

- 与指令对象中的 transclude 属性不同,link 属性的值是一个函数,在该函数中可以操控 DOM 元素对象,包括绑定元素的各类事件,定义事件触发时执行的内容
- link 函数包含3个主要的参数,scope 参数表示指令所在的作用域,它的功能与页面中控制器注入的作用域是相同的
- ele 参数表示指令中的元素,该元素可以通过 Angular 内部封装的 jqLite 框架进行调用
- attr 参数表示指令元素的属性集合,通过这个参数可以获取元素中的各类属性

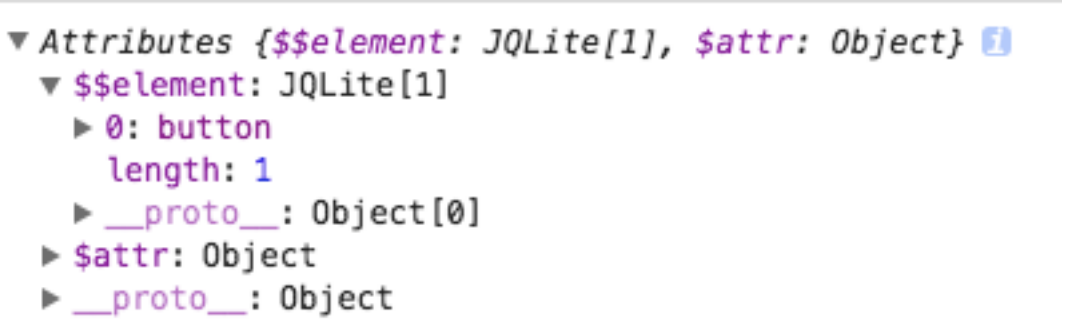
```
link: function(scope,ele,attr){

}
```

link

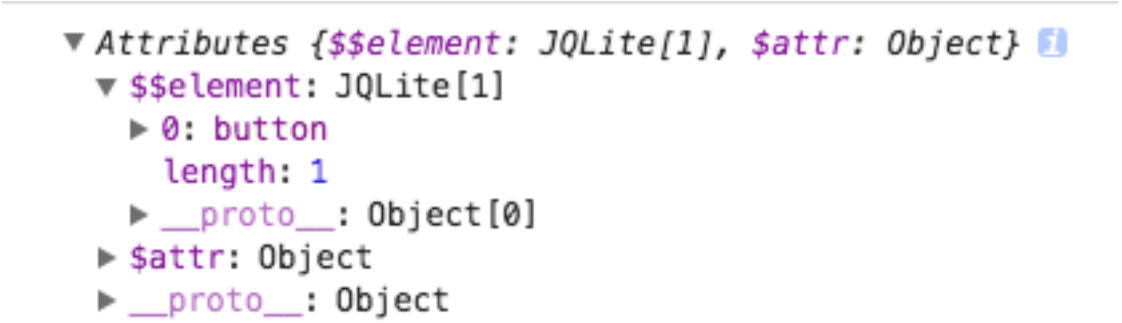

```
//自定义指令
app1.directive('nsButton',function () {
    return {
        restrict: 'ECMA',
        template: '<button>单击显示更多内容</button>',
        replace: true,
        link: function(scope,ele,attr){
            ele.bind('click',function () {
                scope.$apply(function () {
                    scope.content = '这是点击后显示的内容';
                    console.log(attr);
                    attr.$$element[0].disabled = true;
                });
            });
        }
    };
});
```

link

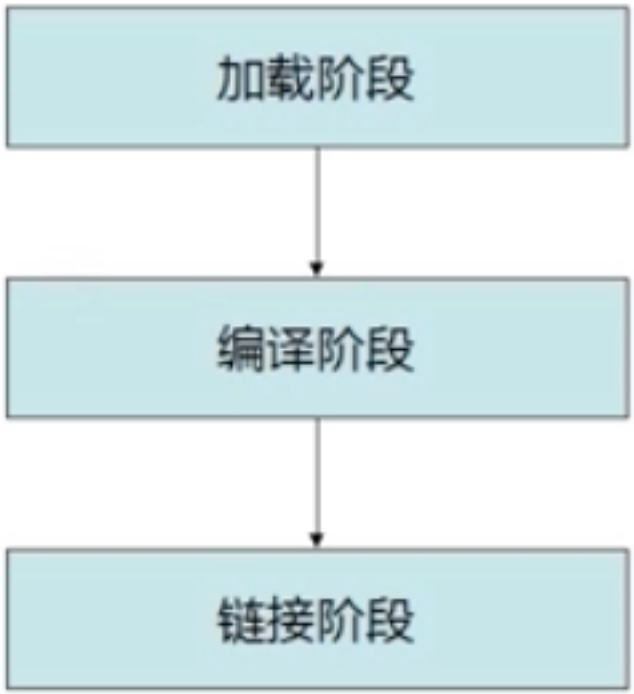
- 在此代码中,指令返回的对象中添加了 link 属性,用于绑定和执行 DOM 元素的各类事件,在指令的执行过程中,由于指令中并没有定义 scope 属性,所以 scope 参数就是元素外层的父级 scope 属性,即控制器注入的 $scope 属性
- ele 参数是被指令模板替换后的 <button> 元素,因为 Angular 里面内置了 jqLite,所以可以调用 jQuery 的方法,绑定了一个点击方法,点击过后通过执行 scope 的 $apply 方法重新渲染页面视图,显示值
- Attr 参数是指令元素的属性集合,如图所示,所以可以通过 attr.$$element[0] 获取 <button> 这个元素,再将他的 disabled 属性值设置为 tur



- Attr 参数是指令元素的属性集合,如图所示,所以可以通过 attr.$$element[0] 获取 <button> 这个元素,再将他的 disabled 属性值设置为 ture,则点击一次后就不能再进行点击(或者是 this.disabled = true 实现的功能也是一样的)



指令的执行机制



- 加载 angular.js,找到 ng-app 指令,确定应用的边界
- 遍历 DOM,找到所有的指令
- 根据指令代码中的 template、replace、transclude 转换 DOM 结构,如果存在 compile 函数则调用
- 对每一条指令运行 link 函数
- Link 函数一般用来操作 DOM,绑定事件

compile

- compile函数用来对模板自身进行转换,而 link 函数负责在模型和视图之间进行动态关联
- 作用域在链接阶段才会被绑定到编译之后的 link 函数上
- compile函数仅仅在编译阶段运行一次,而对于指令的每个实例,link 函数都会执行一遍
- compile函数可以返回 preLink 和 postLink 函数,而 link 函数只会返回 postLink 函数,preLink 和 postLink 函数名是系统提供的,不能修改

compile

- 当添加 compile 属性时,不能同时再添加 link 属性,因为 compile 属性创建了一个名为 postLink 的函数,因此,在编译的过程中,将会自动忽略其他的链接函数
- 如果要修改 DOM 结构,应该在 postLink 中来做这件事情,如果在 preLink 函数中做这件事情会导致错误
- 大部分指令是不用修改 DOM 结构的,compile 函数基本上不需要用到

三、Directive(指令) 的使用(三)

1. scope 的属性值
2. scope 的绑定策略

scope 的属性值

- 指令返回对象的 scope 属性,代表的是指令的作用域,为了便于理解,一般将指令的作用域称之为子作用域,把指令元素所在的作用域称为父作用域
- scope 有三种取值,false (默认),true , { }(JSON 对象)
- scope 的取值默认为 false,此时父子作用域的数据完全相同,一方如果发生变化,另一方将会自动发生变化
- scope 的取值如果是 true,继承父作用域,并创建子作用域,表示子作用域是独立创建的,当它的内容发生变化时,并不会修改父作用域中的内容
- scope 的取值如果是 {},则表示创建一个全新的作用域,父子作用域完全不相干

scope 的绑定策略

- 当 scope 的取值是 JSON 对象的时,父作用域与子作用域是完全独立的,不存在任何关联
- 如果子作用域需要添加属性,必须先添加指令中的 link 函数,再通过函数职工的 scope 对象进行添加
- 如果在子作用域中,要绑定或调用父作用域中的属性和方法,则需要再 scope 属性对应的 JSON 对象值中添加绑定策略
- 在 JSON 对象中添加的有 3 种绑定策略,分别是 “@” 绑定,” = “绑定和 “&” 绑定,绑定符号不同,执行的功能也是有区别的

scope 的绑定策略

- @ 绑定的功能与将 scope 属性设置为 true 时有许多相同的地方,表现为:在子作用域修改属性内容之前,父作用域的属性内容修改了,子作用域对应的属性内容也会随之修改,并且子作用域的属性内容变化时,不会影响到父作用域中对应的属性内容
- 两者唯一的不同在于,@ 绑定的功能在子作用域中修改属性内容之后,再返回父作用域中对应属性内容时,子作用域的属性内容同样还是随之修改

scope 的绑定策略

- = 绑定的功能是创建一个父与子作用域可以同时共享的属性,即当父作用域修改了该属性,子作用域也随之变化;子作用域修改时,父作用域也会跟着变化,两个作用域之间的属性内容是互通的,是完全共享和同步的
- & 绑定的功能是可以在独立的子作用域中直接调用父作用域的方法,在调用时可以向函数传递实参数,好处在于,避免重复编写功能相同的代码,只需要进行简单的绑定设置,就可以使指令执行后,轻松调用元素控制器中的方法
- 由于在指令中的绑定策略不同,在指令元素中,属性绑定属性值也会有些变化,使用 @ 绑定方式绑定的属性,绑定属性值的方式为 {{}},而使用 = 绑定的属性,绑定属性的方式为 = 等号,在使用时需要注意

四、Directive(指令) 的使用(四)

1. require
2. controller

require

- require 和 controller 这两个属性常用于多个自定义指令元素嵌套时使用,即当一个子元素指令需要与父元素指令通信时,就需要添加并使用这两个属性值
- require 属性在创建子元素指令时添加,它的属性值用于描述与父元素指令通信时的方式
- “^”符号表示向外城寻找指定名称的指令
- “?”符号表示即使没有找到,也不会出现异常

controller

- controller 属性值是一个构造函数,在创建父元素指令时添加,可以在该该函数中添加多个方法或属性
- 添加的方法或属性会被实例的对象所继承,而这个实例对象则是子元素指令中 “link” 函数的第4个参数
- 当在子元素指令中添加了 require 属性,并通过属性值指定父元素指令的名称,那么,就可以通过子元素指令中 link 函数的第4个参数来访问父元素指令中 controller 属性添加的方法

五、内置服务(一)

1. $http 的介绍
2. $http 的请求类型
3. $http success 方法
4. $http 函数式写法

$http 的介绍

- 在 Angular 中,页面与服务器端交互的主要方式是调用 $http 服务,属于内置服务,可以直接注入使用
- 该服务的底层封装了 js 中的 XMLHttpRequest 对象,并只接收一个对象作为参数,用于收集生成 HTTP 请求的配置内容,同时返回一个 promise 对象,该对象拥有名为 success 和 error 的两个回调方法
- 根据请求类型的不同,$http 服务提供了不同的调用方式

```
$http.请求类型(url,[data],[config])
.success(function (data,stastus,headers,config) {
})
.error(function(data,stastus,headers,config){
});
```

$http 的请求类型

- 请求类型包括 POST、GET 、JSONP 、DELETE 、PUT 、HEAD,其中 POST 和 PUT 类型请求可以通过可选项参数 data 来发送数据,除发送数据外,还可以通过可选项参数 config 来设置请求时传递的数据
- 参数 url 表示一个相对或绝对的服务器端请求路径
- 当 $http 请求成功时,可以在回调的 success 方法中获取服务器端返回的数据和相关信息

```
$http.请求类型(url,[data],[config])
.success(function (data,stastus,headers,config) {
})
.error(function(data,stastus,headers,config){
});
```

$http success 方法参数

- 当 $http 请求成功时,可以在回调的 success 方法中获取服务器端返回的数据和相关信息
- Data 参数表示返回体,通常是请求返回的结果集
- Status 表示请求后返回的状态值
- Headers 表示请求后返回的头函数,用来显示返回请求的头部信息
- Config 是一个对象,通过该对象,可以获取发送 HTTP 请求时完成的配置信息

```
$http.请求类型(url,[data],[config])
.success(function (data,stastus,headers,config) {
})
.error(function(data,stastus,headers,config){
});
```

$http 函数式写法

- 前面所学习的 $http 方法虽然简单,但配置时缺少灵活性,代码量也不少,针对这种情况,可以将 $http 服务当成函数来使用,将构造 XHR 对象的所有配置项作为一个对象,并将对象定义为函数的形参,调用时,只需修改形参对象中的各属性即可

```
$http ({
    method:
    url:
    data:
    params:
    transformRequest:
    transformResponse:
    cache:
    timeout:
})
```

$http 函数式写法

- url : 表示向服务器请求的地址
- data : 是一个对象,该对象将作为消息体的一部分发送给服务端,常用语 POST 或 PUT 数据时使用
- params : 是一个字符串或者对象,当发送 HTTP 请求时,如果是对象,将自动按 JSON 格式进行序列化,并追加到 URL 的后面,作为发送数据的一部分,传递给服务器
- transformRequest：用于对请求体头信息和请求体进行序列化转换,并生成一个数组发送给服务端

$http 函数式写法

- transformResponse：用于对响应体头信息和响应体进行反序列化的转换,其实质就是解析服务器发送来的被序列化后的数据
- cache：表示是否对 HTTP 请求返回的数据进行缓存,如果为 true ,表示需要缓存,否则不缓存
- timeout：表示延迟发送 HTTP 请求的时间,单位为毫秒

六、内置服务(二)

1. $resource 服务
2. promise 对象

$resource 的介绍

- 与 $http 服务相比而言,$resource 的服务功能更为强大,不仅如此,它的核心价值在于能为支持 RESTful 的服务器进行无缝隙的数据交互,而这种交互是数据模型方式的对接,通过它抽象剥离出来的方法,无需开发者编写大量代码,就可以实现许多复杂的功能
- 在调用 $resource 后,返回的 $resource 对象中包含了多种与服务端进行交互的 API,像 get、save和 query 等,开发人员只需调用就可以实现对数据的基本操作功能
- 在请求 $resource 时,允许自定义请求的方法,使用非常灵活

$resource 的介绍

- $resource 服务本身是一个可选性的模块,它并没有包含在 Angular 中,如果需要使用该模块,需要再页面中先通过 <script> 元素进行文件的导入,然后在应用模型中注入 ngResource

promise

- 在操作 AJAX 异步请求时,必须添加一个 callback 函数,用于处理请求成功后的逻辑,但这种方式是以牺牲控制流、异步处理为代价,并且还有可能陷入 callback 函数嵌套中,流程复杂,代码臃肿
- promise 是一种处理异步编程的模式,可以有效解决回调的烦琐,并以一种同步的方式去处理业务流程,同时允许在回调中采用链式写法,处理的代码更加优雅

本章总结

- 自定义 directive 可以达到重用的效果
- 自定义 directive 最好使用驼峰命名标志,在 html 中用指令的时候用 – 隔开使用
- E C M A 代表指令的四种用法(元素,类,注释,属性)
- 使用 templateUrl ,当传入的值为 HTML 文件路径时,html 文件必须有标签
- link 函数绑定元素的各类事件,定义事件触发时执行的内容
- 最好不要使用 compile 函数,compile 函数与 link 函数不能共存

本章总结

- Scope 有三种绑定策略: @ 绑定 ,= 绑定 , & 绑定
- require 和 controller 这两个属性常用于多个自定义指令元素嵌套时使用

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