Webstorm使用

|  |  |
| --- | --- |
| Ctrl+/ 或 Ctrl+Shift+/ | 注释（// 或者/\*…\*/ ） |
| Shift+F6 | 重构-重命名 |
| Ctrl+X | 删除行 |
| Ctrl+D | 复制行 |
| Ctrl+G | 查找行 |
| Ctrl+Shift+Up/Down | 代码向上/下移动。 |
| F2 或Shift+F2 | 高亮错误或警告快速定位 |
| 写代码，按Tab | 生成代码 |
| 选中文本，按Ctrl+Shift+F7 | 高亮显示所有该文本，按Esc高亮消失。(因为这个功能我就可以发放心的放弃sublime了) |
| Ctrl+B或Ctrl+鼠标左键单击 | 快速打开光标处的类或方法，（NB的功能） |
| Ctrl + Alt + B | Go to implementation(s) 跳转方法实现处 |
| Ctrl + Shift + I | Open quick definition lookup 打开定义快速查找 |
| Alt + Up/Down | Go to previous/next method 跳转到上一个/下一个方法 |
| Ctrl+E | 最近打开的文件 |
| Alt+F1 | 查找代码所在位置 |
| Ctrl+Alt+L | 格式化代码 |
| Ctrl+R | 替换文本 |
| Ctrl+F | 查找文本 |
| Ctrl+P | 方法参数提示 |
| F3 | 查找下一个 |
| Shift+F3 | 查找上一个 |
| alt+Shift+F | 将当前文件加入收藏夹 |
| ctrl+alt+s | 打开配置窗口 |
| ctrl+Shift+N | 通过文件名快速查找工程内的文件（必记） |
| ctrl+Shift+alt+N | 通过一个字符快速查找位置（必记） |
| Shift+enter | 重新开始一行（无论光标在哪个位置） |
| Ctrl + Alt + T | with…（if, else, try, catch, for, etc）用 \* 来围绕选中的代码行，（ \* 包括 if 、 while 、 try catch 等） |
| Ctrl + Shift + U | Toggle case for word at caret or selected block 光标所在位置大小写 |
| Ctrl + Delete | Delete to word end 删除文字结束 |
| Ctrl + Backspace | Delete to word start 删除文字开始 |
| Ctrl + E | Recent files popup 弹出最近打开的文件 |
| F11 | Toggle bookmark 切换标记，我觉得叫书签更好，就是sublime text 的F2 |
| Ctrl + Shift + F12 | Toggle maximizing editor 切换最大化编辑器 |
| Alt + Shift + F | Add to Favorites 添至收藏夹 |

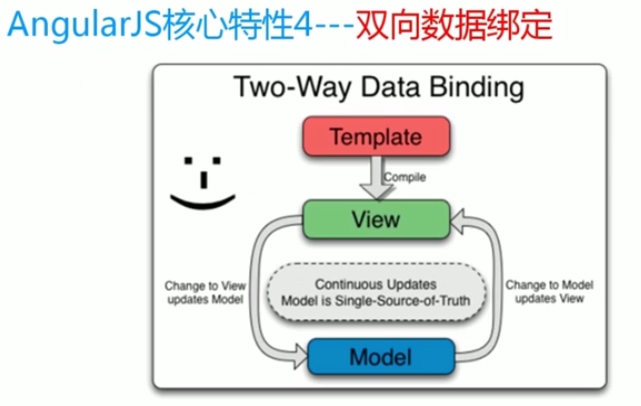
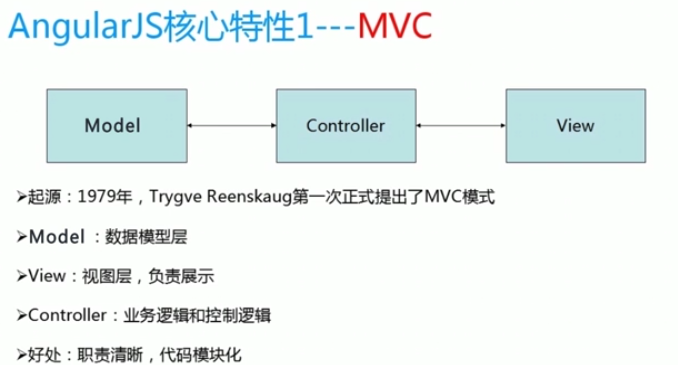
Angular 4大核心特性

1）MVC

2）模块化

3）指令系统

4）双向数据绑定



ng-app 只能有一个，ng-app="MyModule"告诉浏览器要执行那个模块，

相当于c中的main函数

问题1. 为什么大多数框架不实现双向数据绑定

绝大部分操作只需要单向绑定就能实现，双向绑定在一定程度上不利于管理数据源

单向数据绑定：指的是我们先把模板写好，然后把模板和数据（数据可能来自后台）整合到一起形成HTML代码，然后把这段HTML代码插入到文档流里面。

单向数据绑定缺点：HTML代码一旦生成完以后，就没有办法再变了，如果有新的数据来了，那就必须把之前的HTML代码去掉，再重新把新的数据和模板一起整合后插入到文档流中。

双向数据绑定：数据模型（Module）和视图（View）之间的双向绑定。

用户在视图上的修改会自动同步到数据模型中去，同样的，如果数据模型中的值发生了变化，也会立刻同步到视图中去。

双向数据绑定的优点是无需进行和单向数据绑定的那些CRUD（Create，Retrieve，Update，Delete）操作

双向数据绑定最经常的应用场景就是表单了，这样当用户在前端页面完成输入后，不用任何操作，我们就已经拿到了用户的数据存放到数据模型中了。

目前。实现双向数据绑定的前端框架主要有**[AngularJS](http://lib.csdn.net/base/angularjs" \t "_blank" \o "AngularJS知识库)**，VueJS等

原理：

我们需要一个UI元素和属性相互绑定的方法  
我们需要监视属性和UI元素的变化  
我们需要让所有绑定的对象和元素都能感知到变化

1）代码编辑工具

sublime Text（轻量级）

webstorm(强大，集成很多功能，神器)

2）断点调试工具

chrome插件Batarang

firebug

3）版本管理工具

git

git小乌龟

4）开发和调试工具

Nodejs

5）代码合并和混淆工具

grunt

js文件合并/代码自动压缩

Ctrl+s自动执行以上动作

Ctrl+s自动运行单元测试、集成测试

http://www.gruntjs.org/

6）依赖管理工具

bower

轻量级Server------Http-server(模拟测试数据)

7）单元测试工具/集成测试工具

Karma

Jasmine(分组、用例、期望、匹配)

Protractor(专门为angularjs定制的测试工具)

为什么需要 MVC ？

1、代码规模越来越大，切分职责是大势所趋

2、为了复用：很多逻辑是一模一样的

3、为了后期维护方便：修改一块功能不影响其他功能

MVC 只是手段，终极目标是模块化和复用

MVC的困难

1、操作dom的代码必须等待整个页面全部加载完成

2、多个js文件之间如果出现互相依赖，程序员必须自己解决

3、js的原型继承也给前端编程带来了很多困难

浏览器加载脚本是以字符串形式加载的，然后通过内部的JIT进行编辑

A：AngularJS的Controller--MVC基本思想：

1）正确：一个控制器只负责跟一个视图发生交互，多个视图共用一个数据模型，

通过Service，把控制器中通用的东西抽成一个服务，让控制器去调用它；

2）错误：控制器中有通用内容，抽出来通用控制器后继承它；

B：Controller中避免以下操作，注意第五点：

1）不要复用Controller，一个Controller只负责一个视图；

2）不要在Controller中操作DOM；

3）不要在Controller中对数据格式化，AngularJS有表单控件；

4）不要在Controller中对数据进行过滤，ng有$filter服务；

5）Controller不会相互调用，避免强耦合，用数据模型事件交互，Controller内部监控事件进行操作；

有时，标签中只要写ng-app，而有时却要用ng-app=”MyApp”之类的带具体值的写法，那两者区别在哪呢？

如果在js文件中的写法是直接用函数定义controller，例如：

var mycontroller = function($scope){

$scope.msg = 'hello';

}

那么在html中只需要用ng-app声明即可；

如果在js文件中是用module的方式来声明controller，例如：

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

app.controller('mycontroller ',function($scope){

$scope.msg = 'hello';

});

那么在html中就要声明ng-app=”MyApp”，否则是无法使用mycontroller的。

$scope生命周期

1 创建

2 注册监控

3 检测模型变化

4 观察模型脏值

神奇的$scope

$scope是一个POJP （Plain Old JavaScript Object）

$scope提供了一些工具方法$watch()/$apply()

$scope是表达式的执行环境(或者叫作用域)

$scope是一个树形结构，与DOM标签平行

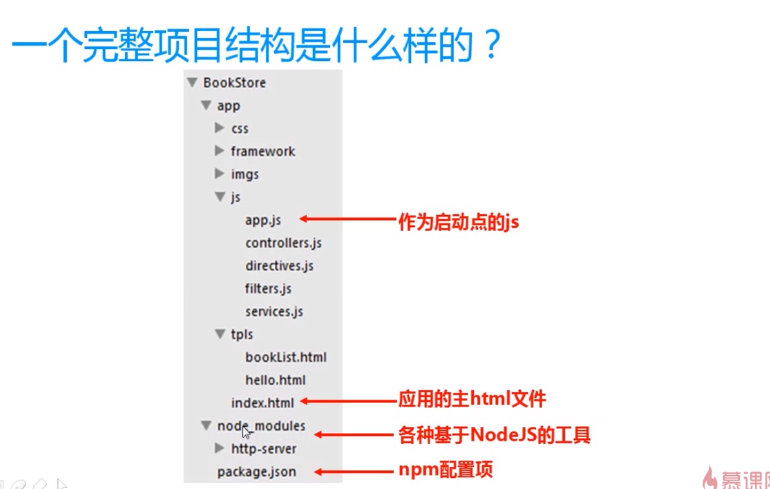
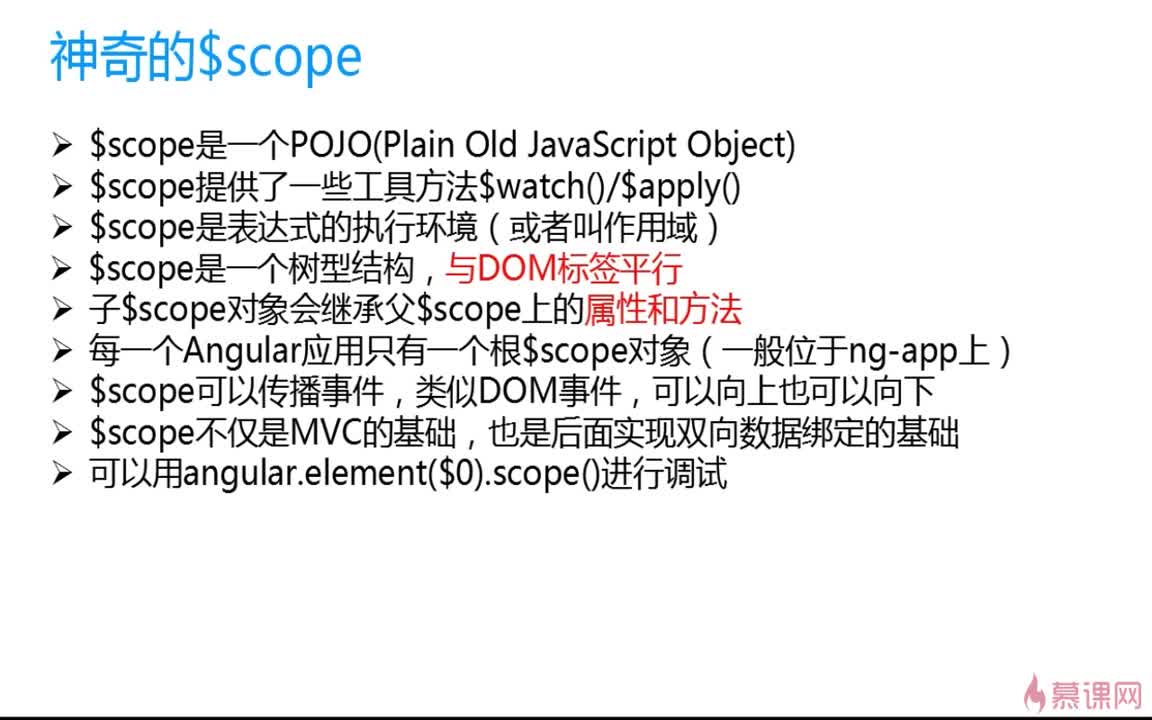
可以用anfular.element($0).scope()进行调试

子$scope对象会继承父$scope上的属性和方法

么一个Angular应用只有一根$scope对象(一般位于ng-app上)

$scope可以传播事件，类似DOM事件，可以向上也可以向下

$scope不仅是MVC的基础，也是后面实现双向数据绑定的基础



url地址中的#:防止浏览器向后台提交请求，代表内部锚点，不会向服务端发起请求

通过"#/目录名"方式告诉浏览器在页面内进行跳转，angular会拦截到这个url地址，把"#"后面的内容取出来，和routeProvider进行匹配，展示相应的视图；所以可以把不同的视图交给不同的控制器处理，把视图之间的职能分开，用多个控制器处理不同的视图内容首页或者第一个页面有数据绑定的话，用ng-bind，只要angular.js加载完后其他视图用{{}}绑定数据

ng-bind和{{}}作用是一样的，只不过好看一些

一般情况下：

- index用ng-bind

- 其他页面用双花括号