# AngularJS总结

总的来说，AngularJS的核心就是**数据绑定**，也就是这句话：

- **“作用域对象通过ng-app、ng-controller、ng-repeat、ng-view等等指令，被绑定到HTML标签上。而对应的HTML标签内部的各个AngularJS指令里的AngularJS表达式，就会依附于这个作用域执行、求值并进行对应的绑定。”**

依据“数据绑定”这个核心，AngularJS分成了以下的几个部分：

1. **AngularJS内置指令**（如：ng-app, ng-controller, ng-bind, ng-model, {{}}, ng-class等），AngularJS内置指令用于在文档上打标记，示意AngularJS如何把作用域上的数据绑定到页面上
2. **AngularJS控制器**，用于创建一个自带作用域（$scope）的AngularJS功能组件，需要时我们可以用ng-controller指令，把AngularJS控制器的作用域和一个html标签进行绑定。
3. **AngularJS自定义指令**，用于扩展AngularJS的指令，不仅仅可以使用内置指令，我们还可以自定义指令。自定义指令允许拥有自己的作用域，指令的作用域和外部作用域进行交互时，使用scope字段进行设置。而且还可以使用controller字段和link字段取到作用域对象，对作用域对象进行编辑。
4. **AngularJS服务和依赖注入**，用于提供AngularJS的各种功能。我们还可以自定义一些通用的功能作为服务。然后用依赖注入的方式使用这些功能。
    a. AngularJS内置了一些服务，比如$http可以用来做ajax访问、$parse可以解析表达式等等。
    b. 我们还可以自定义AngularJS服务，常用的自定义方法是app.factory(服务名,回调函数)。
5. **AngularJS作用域**，AngularJS作用域会和某个HTML标签进行绑定，为这个标签内部的各种指令提供数据内容。除此之外，它还提供两个特殊的方法：
    a. 当我们人为修改数据但是AngularJS不知道时，我们可以通过$scope.$apply()通知AngularJS；
    b. 当AngularJS监控到数据改变，但是我们有可能不知道时，可以通过定义$scope.$watch()方法让AngularJS通知我们。这两个方法都不算常用，但是有必要了解。
    c. 当页面上的$watch超过两千个后，页面可能就会变得卡起来。

做个总结：

1. 内置指令是用于帮我们在HTML文档上打标记，告知AngularJS框架如何帮我们做绑定的。
2. 作用域是AngularJS框架做绑定时的数据源。
3. 控制器是用来提供$scope对象的。
4. 模块是用来提供包括控制器在内的各种AngularJS功能组件的。
5. 自定义指令是帮我们扩展在HTML文档上打标记的方法的，是内置指令的补充。与此同时，自定义指令也可以帮助我们打标记。
6. 服务和自定义服务通过依赖注入的方式为框架提供了各种功能，让我们在创建控制器、自定义指令和其他服务时能使用这些通用的功能完成更强大的想、更好的组织代码。

换句话说就是，整个AngularJS框架，一部分是用来做绑定的，一部分是用来更好的做绑定的，还有一部分是用来整理代码结构的。

# AngularJS自我评判标准

如何判断自己算不算“会用”AngularJS了呢？

评判标准很简单，能不能用双向绑定的机制完成一个MVVM的小应用。大的应用和小的应用之间最大的差别就是规模的差别，知识量上的差别可能就小一些了。最多也就是更多的各种各样的小工具。

我认为，我们可以关注于下面几个点：

1. 入门（了解，能够使用）
    a. 使用AngularJS的内置指令和AngularJS控制器完成各种数据绑定，使用依赖注入的方式取得AngularJS提供的各种功能。
    b. 使用$http服务访问网络。（这个没怎么讲，但是你们会发现它和jQuery的ajax大部分地方都是类似的，网上资料也有太多太多。）
2. 基础（能够使用，熟悉）
    a. 能够使用factory创建自定义的服务，并使用依赖注入的方式获取这些服务。
    b. 理解AngularJS表达式需要执行才能得到值，表达式的执行需要一个上下文或者说是“作用域”，作用域绑定在各个HTML标签上，而ng-app、ng-controller等指令，可以把作用域和HTML标签绑定起来。理解这个就能够理解AngularJS这个框架的很多原理性的东西。
    c. 能说的清楚MVVM是什么，为什么要使用MVVM的设计思路进行开发。
3. 熟练（熟练）
    a. 能够创建自定义指令，使用template、replace、transclude等等字段配置模板相关的内容，使用scope字段来配置指令内部作用域和外部作用域的关联关系，使用link字段来配置一个用于操作自定义指令作用域对象、操作DOM的函数。注意，自定义指令的link函数是通常情况下，开发中唯一推荐做DOM操作的地方。
    b. 会使用ngRoute等第三方模块。尤其是需要会使用一种路由模块。ngRoute是一种，课下有精力还可以去学习使用angular-ui-router。
    c. 能说的清楚MVVM是什么，为什么要使用MVVM的设计思路进行开发。
4. 熟练+
    a. 对课上讲过的各种细节都有所理解，甚至于有自己的看法。（比如说：什么时候AngularJS可能会发现不了数据的改变，为什么。=> 因为AngularJS的watch机制，当有AngularJS监控的事件发生时，AngularJS会检查所有作用域上的所有watcher，看它们监控的表达式的值是不是发生了变化。这导致了一个什么问题？watcher太多了、性能会变差一些。如果你来写可能用什么办法来改进？我可能会使用Object.DefineProperties之类的api为$scope对象设置getter和setter进行监听……实际上、很多框架、比如vue、avalon等，都使用了这个办法……）
    b. 理解AngularJS整个渲染流程（编译compile阶段做模板的字符串替换，然后是创建

作用域对象，然后执行连接函数link、通过dom操作对文档进行修改完成数据绑定）。甚至打断点看过源码怎么跑的。

c. 了解更多的第三方库，比如ui-router，比如ui-bootstrap，又比如说官方出品的那一大堆。
d. 了解其他的数据绑定框架，比如React、Vue、Avalon，Backbone、Ember等等.....
e. 了解脏值检测、虚拟dom等等可能用得到的算法。

# AngularJS资料参考

在精不在多，我就推荐几个：

1. 《AngularJS权威教程》，上课时多次提过，很好的一本书。
2. AngularJS沉思录。非常深入的一系列博客文章，讲得非常深入。
3. Vue框架、或者Avalon框架，去学习一下这个框架，就可以对比着来理解各种类似框架是，这样是很有好处的。这两个框架都有不错的官方文档，比较适合学习。

# AngularJS的优势

1. 可读性高、表现力强
2. 开发迅捷、效率高
3. 相比起程序员自己写选择器，性能可能会相对高一些。（这个是不一定的）

# 关于依赖注入的小细节

# 关于$digest

# 关于几个数组函数

array.filter()

arr.map()

arr.reduce()

# 关于promise

2016年5月17日 0:29