一、软件工程知识点

1、软件设计原则/优化原则

(1)DRY原则

不要重复代码(Ctrl+C/V)

(2)KISS原则

代码越简单越好

(3)YAGNI原则

不要写不需要的代码

(4)OCP原则

对扩展持"开放"态度；对修改持"封闭"态度

(5)SRP原则

单一责任原则

例如 <div class="col-sm-8 col-md-6 alert alert-danger modal"></div>

此写法违反单一责任原则，不要将所有的功能放在同一个div上

所以要如下写法：

<div class="col-sm-8 col-md-6">

<div class="alert alert-danger">

<div class="modal"></div>

</div>

</div>

(6)High Cohesion,Low Coupling高内聚低耦合原则

(7)迪米特法则(也叫最少知识法则)

让一个对象知道的越少越好

2、软件设计模式

总共有23+1种设计模式

+1模式 -- MVC模式(分为三个层次)

(1)Model，模型，指业务数据

Web项目中由Js中的变量(数字、字符串、对象、数组等)来担当

(2)View，视图，指业务数据在用户面前的呈现

Web项目中由HTML(增强型)来担当

(3)Controller，控制器，负责获取、删除、更新模型数据

Web项目中由Js中的function来担当

二、AngularJS概述

1、它是一个纯Js框架，基于JQuery对DOM操作做了进一步的封装

使用MVC操作代替所有的DOM操作

\*适用于以数据操作为主的SPA(SinglePageApplication)应用

2、四大特性

(1)采用MVC模式，页面中再也无需出现DOM操作

(2)双向数据绑定

(3)依赖注入

(4)模块化设计

3、i18n

internationalization，国际化，即一个项目可以根据客户端的不同，呈现出不同的语言来

三、Angular表达式

1、语法：

{{ 表达式 }}

2、作用：

在当前位置"输出"该表达式的值

3、表达式中可以执行哪些式子

(1)算术运算

+ - \* /都可以，++和--不行

(2)比较运算

> < >= <= == ===都可以

(3)逻辑运算

且、或、非都可以

(4)三目运算

可以

(5)赋值运算

直接赋值(=)可以；+=、-=等运算赋值表达式不可以

(6)特殊运算

不可以

(7)调用string对象的成员方法和属性

都可以

(8)创建新对象

直接量语法可以，但不能使用new关键字

(9)创建数组

直接量语法可以，但不能使用new关键字

四、Angular中ng模块提供的指令(Directive)

1、ngApp，自动载入/启动一个Angular应用

语法：

<div ng-app="xxx"></div>

或<div data-ng-app="xxx"></div>

或<div ng-app></div>

\*一个HTML页面中只允许使用一次ngApp指令，用于确定AngularJS应用的范围，只有ngApp范围内的Angular表达式才会被计算出对应的值

如果有两个元素同时需要ngApp，则可以将它放在这两个元素的父元素上来解决，甚至可以放在<html>元素里来解决

2、ngInit，用于声明Model变量

语法：

<div ng-init="变量名=值;变量名=值; ... "></div>

\*Model变量声明时不能使用var关键字

3、ngController，创建一个控制器对象的实例(即调用Controller函数)

语法：

<div ng-controller="控制器名"></div>

4、ngBind，在当前元素的innerHTML上输出指定的表达式的值

语法：

<div ng-bind="表达式"></div>

此指令的作用跟{{}}一样，但是可以防止刷新闪动问题

例如 <p ng-bind="stu.score">成绩：</p>

显示的效果中"成绩："并不显示，因为是innerHTML上输出，它已经被值覆盖掉

→所以要<p ng-bind="'成绩：'+stu.score"></p>

5、ngRepeat，为HTML增加循环功能，循环输出当前元素

语法：

<div ng-repeat="变量名 in Model数组/对象">{{变量名}}</div>

<div ng-repeat="(下标,值) in Model数组/对象">{{下标}}或{{值}}</div>

6、ngIf，为HTML增加选择功能

只有在if表达式值为true时，当前元素才添加到DOM树；否则就从DOM树上删除

语法：

<div ng-if="表达式"></div>

7、ngSrc，解决img等标签的src属性中包含{{}}产生的404问题

语法：

<img ng-src="路径/{{表达式}}">

8、ngClick，为元素绑定监听函数(不是全局函数，而是Model函数)

语法：

<div ng-click="模型函数名()"></div>

onclick绑定的是全局函数，不能使用

\*在Angular中必须使用：

$scope.模型函数名=function(){}格式来声明模型函数

传统的onXxx事件句柄都有一个对应的ngXxx版本

9、ngStyle，赋值为一个对象，用于为当前元素指定样式

语法：

<div ng-style="模型对象"></div>

10、ngShow/ngHide，通过display：none/block来控制当前元素是否显示

语法：

<div ng-show/hide="表达式"></div>

11、ngDisabled，赋值为true/false，可以控制当前元素是否禁用

语法：

<div ng-disabled="表达式"></div>

12、ngChecked，赋值为true/false，可以控制当前元素是否选中

语法：

<div ng-checked="表达式"></div>

五、Angular中声明变量 -- Model数据(两种方式)

1、使用ngInit指令声明

(1)作为HTML元素属性来使用

例如 <span ng-init="price=10.5"></span>

<p>价格1：{{price}}</p>

(2)作为HTML元素样式来使用

例如 <span class="ng-init:price=15;"></span>

<p>价格2：{{price}}</p>

\*ngInit指令声明的Model变量可以先使用再声明

ngInit指令可以一次声明多个Model变量，用分号隔开即可

ngInit指令可以声明哪些类型的Model变量：

number、string、boolean、对象、数组、对象的数组

\*使用ngInit定义Model变量时的缺点：

(1)不能使用new关键字

(2)此方法把View和Model混杂在一起了

所以使用ngInit指令声明变量【不推荐使用】！

2、使用Controller创建Model变量

过程：

创建Module→创建Controller→创建Model变量

Angular要求控制器必须声明在一个模块中

具体步骤：

(1)声明一个自定义的模块Module

angular.module('模块名',[])

(2)在当前AngularJS应用中注册自定义模块

ng-app="模块名"

(3)在自定义模块中创建Controller函数，其中声明Model数据

$scope.模型变量名=值;

(4)在View中创建Controller对象的实例，指定其作用范围

<div ng-controller='控制器名'>

...

此处即就是控制器的有效范围

...

</div>

(5)在控制器的作用范围内输出Model变量

可以使用{{}}输出Model变量的值

六、$scope和$rootScope

1、每个控制器的实例都对应一个作用范围对象，即$scope

2、在控制器中声明的Model数据，必须保存在一个作用范围内

3、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

4、为了在多个控制器间共享数据，可以将Model数据保存在一个"全局作用范围内" -- 即$rootScope

\*整个AngularJS中【有且只有一个】$rootScope对象，且此对象是所有$scope的父作用域对象

5、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即可

七、双向数据绑定

1、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、ngStyle等

(2)方向二：把View中的(指表单控件)修改绑定到Model上

此后不论何时只要View中的数据改变了，则Model中的数据会自动随之改变

实现方法：

ngModel指令

①只要input的value改变，Model数据的userName会自动随之改变

②可以使用$scope.$watch('模型变量名',function(){})监视一个模型变量值的改变

③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

2、双向数据绑定的实现原理，以及AngularJS最大的缺陷

每一次方向一的绑定都会在$digest队列中生成一个执行DOM操作的函数(若一个ngApp中有n次数据绑定，就会生成n个这样的函数)，只要某一个Model数据发生了值的改变，立即会自动执行$digest队列中的每一个函数，进行View的更新

\*\*setInterval()和$interval()的不同

(1)window.setInterval()只会执行指定的任务，即使修改了Model数据也不会自动轮询$digest队列

(2)$interval()的执行体中会在最后自动执行$scope.$digest()或者$scope.$apply()

即轮询$digest队列，执行其中的每一个函数

$interval()=setInterval()+$scope.$digest()

\*见Unit3实例2

八、ng模块中提供的服务(service)

(1)$rootScope，用于在不同的控制器间共享数据

(2)$interval，提供周期性定时器服务

(3)$timeout，提供一次性定时器服务

(4)$http，发起Ajax请求

九、如何压缩css/js文件，得到xx.min.css/js

1、YUI框架：Yahoo User Interface，是由雅虎的前端工程师将自己的工作经验整理出来的前端工具集合(百度"YUI军规")

其中有一个小工具：yui-compressor，可用于压缩css和js文件

2、cmd使用步骤：

(1)下载并安装java语言的运行环境 -- JDK

(2)在cmd中输入java.exe测试是否安装成功

(3)在java命令下运行yui-compressor工具

java.exe -jar c:/yui-compressor-2.4.2.jar e:/1.js > e:/1.min.js

e:/1.js，是文件的路径位置

e:/1.min.js，是将要生成的min文件的路径位置

3、将此工具配置为WebStorm中的文件监视器FileWatcher

实现功能与less文件类似，具体步骤：

(1)下载并安装java语言的运行环境 -- JDK

(2)在cmd中输入java.exe测试是否安装成功

(3)Settings→Tools→File Watchers

(4)右上角加号→YUI Compressor CSS/YUI Compressor JS

(5)Program: 设置路径yuicompressor-2.4.2.jar

4、压缩效果

(1)删除所有的注释(不可恢复)

(2)删除没有语义的空白字符

(3)尽可能的简化变量名、函数名、形参名(此操作称之为“混淆”)(不可恢复)

(4)所有的数据值、关键字(如function、window)不会做任何改变

十、依赖注入(DI)

1、什么是依赖(Dependency)

例如 function Driver(car){

car.start();

car.run();

car.stop();

}

Driver对象的创建和运行必需一个car对象，这称之为Driver对象“依赖于”car对象

2、依赖对象的解决方法

(1)主动创建

var c1=new Car(); //创建被依赖的对象

var d1=new Driver(c1);

(2)被动注入(inject)

一般由特定框架来创建Driver对象，发现其依赖于一个car对象，框架自动创建被依赖的car对象

例如 module.contorller('控制器名',function($scope){...})

$scope不用创建，angular框架会自动创建并调用

注意：

①控制器对象的构造函数是由AngularJS来调用的，不能手动调用

②Angular会根据控制器对象的构造函数的【形参名】来创建依赖的参数对象，例如$scope，形参名不能随意编写，否则创建失败报错

③若控制器对象未声明形参，则Angular不会传递任何实参进来

④控制器对象的形参名必须是Angular可识别的，但是数量和顺序都没有限制

⑤Angular会根据每一个形参的名称来查找创建被依赖的对象，并自动注入进来

\*若使用了JS的压缩功能(min.js)，会自动将依赖对象的形参名进行精简混淆，则Angular就无法再根据形参名实现依赖注入了！

解决方法：

module.controller('c4',['$scope','$http',function (a,b) {

... ...

}])

其中a对应$scope，b对应$http

3、依赖注入体现着"最少知识法则"

十一、模块化设计

1、模块化设计体现着"高内聚低耦合"设计原则

项目中，可以根据功能的不同，将不同的组件放置在不同的模块中

例如用户管理相关内容全部放在userModule，商品相关的内容全部放在productModule

2、AngularJS中有两种模块

(1)AngularJS官方提供的模块

ng、ngRoute、ngAnimate、ngTouch等

(2)用户自定义的模块

userModule、productModule、orderModule等

3、一个AngularJS的模块中可以包含哪些组件

(1)controller组件，用于维护模型数据

(2)directive组件，用于在View中输出/绑定Model数据

(3)service组件，用于在不同的控制器中提供某种函数服务

(4)filter组件，用于对View中输出的数据进行格式化

(5)function组件

(6)object组件

(7)provider组件

(8)type组件

十二、Angular模块中的常用组件 -- filter过滤器

用于Model数据在View中呈现时进行某种格式的筛选/过滤/格式化

在View中使用过滤器时，需要借助于管道 "|"

\*ng模块中提供的过滤器

(1)lowercase，把数据格式化为小写

语法：{{表达式 | lowercase}}

(2)uppercase，把数据格式化为大写

语法：{{表达式 | uppercase}}

(3)number，把数字型数据格式化为三位一个逗号的字符串(如果是小数且小数位数超过三位，默认保留三位小数)

例如 123,456.656

语法：{{表达式 | number}}

同时可以指定小数点位数

{{表达式 | number : 2}}，表示小数点保留两位(并四舍五入)

(4)currency，把数字型数据格式化为三位一个逗号的货币格式的字符串(如果是小数且小数位数超过两位，默认保留两位小数)

例如 $123,456.66

语法：{{表达式 | currency}}

同时可以指定货币符号

{{表达式 | currency : '￥'}}

→￥123,456.66

(5)date，把数字/Date型数据格式化为特定日期时间格式的字符串

真实项目中往往不使用Date类型表示日期时间，而使用【长整形的数字】来代替

所有的编程语言/数据库系统都支持长整形数字，都可以把数字和日期时间随意转换

语法：{{表达式 | date}}

{{表达式 | date:'yyyy-MM-dd'}}

{{表达式 | date:'yyyy年MM月dd日'}}

{{表达式 | date:'yyyy-MM-dd HH:mm:ss'}}

十三、Web项目中"单页应用"和"多页应用"的比较

1、多页应用

(1)一个项目中由多个完整的.html页面

(2)多个页面间跳转可以使用超链接、表单提交、JS(location.href="xxx.html")

(3)页面切换是同步请求

客户端先删除第一个页面的DOM结构，发起HTTP请求，等待服务器给第二个页面的响应数据(等待过程中页面一片空白)

(4)每个页面都是一个完整的DOM树

(5)页面切换时控制权在浏览器中，不可能添加任何的过场动画效果

2、单页应用

(1)只有一个.html是完整的(缺少body主体)，其他.html都是不完整的(可能只是一个div而已)

(2)多个"伪页面"间的跳转可以使用超链接、JS()

(3)伪页面切换是异步请求

客户端首先请求一个完整的页面，然后再发起异步Ajax请求，获取不同的模板页面，插入在当前的DOM树上

(4)整个项目只有一个完整的DOM树

(5)伪页面切换的本质是一颗DOM树上的两个DIV在切换，可以很容易的添加各种过场动画

总结：

单页应用完全可以实现传统的多页页面的各种效果，同时还可以降低服务器和客户端数据传输量、加快页面显示速度、添加丰富的过场动画效果

3、如何仿照AngularJS实现SPA的页面切换效果

(1)页面URL须知：http://127.0.0.1/index.html#/路由地址

(2)浏览器首先请求基础页面(index.html)，再解析URL中的路由地址

(3)查找路由字典，形如：

#/start =》 tpl/start.html

#/main =》 tpl/main.html

... ...

获取当前URL中路由地址所对应的真实模板页面的地址

(4)客户端发起异步Ajax请求，获取目标模板页面，将服务器返回HTML片段(只含有几个div)，插入到当前的DOM树上

十四、ngRoute模块

ngRoute模块可以让用户自定义“路由字典”，自动解析请求URL中的路由地址，查找路由字典，自动发起异步Ajax请求，把获取的结果放在当前页面中

1、使用ngRoute模块的步骤：

(1)创建唯一完整的页面：index.html，引入angular.js和angular-route.js

(2)在index.html中的body中使用ngView指令声明一个容器元素，用于盛放模板页面

(3)创建自定义模块，声明依赖于ng和ngRoute两个模块

(4)在当前模块中使用ngRoute提供的对象配置路由字典

module.config(function($routeProvider){

$routeProvider.

when('/路由地址',{

templateRrl:'模板页面URL'

}).

when('/路由地址',{

templateRrl:'模板页面URL'

}).

... ...

otherwise({

redirectTo:'/路由地址'

})

})

(5)再创建几个模板页面，只需要有div元素即可

(6)测试路由字典的配置是否正确

http://127.0.0.1/index.html#/路由地址

2、使用ngRoute模块需要注意的问题

(1)由于模板页面被客户端请求后挂载在index.html DOM树上，所以其中所有的图片等外部资源文件的路径必须相对于index.html，而不是模板页面

(2)使用ngRoute模块时，无需为模板页面中的某个元素单独设置ngController，只需要在声明路由字典时设置整个模板页面所需要的控制器即可

when('/路由地址',{

templateUrl:'xxx.html',

controller:'控制器名'

})

(3)在不同的模板页面间跳转可以采用两种方式：

①超链接方式

<a href="#/路由地址"></a>

②js编码方式

页面中：

<button ng-click="jump('/路由地址')"></button>

js中：

$scope.jump=function(routeUrl){

$location.path(routeUrl);

}

(4)index.html中，由于所有模板页面都是在ngView中切换，ngView外面的内容始终会呈现

十五、ngAnimate模块

ngAnimate可以支持js、Transition、Keyframes动画，但它本身未提供任何的动画效果，而是为上面三种技术提供了相应的"动画钩子(Hooks)"

1、默认提供了动画钩子的ng模块中的指令：

ngRepeat、ngInclude、ngIf、ngSwitch、ngShow、ngHide、ngView、ngClass

2、使用ngAnimate模块调用其CSS Transition动画钩子的步骤：

(1)index.html引入angular.js和angular-animate.js

(2)自定义模块中声明依赖于ngAnimate模块

特点的ng指令就会产生动画钩子

(3)为ngView声明class，样式中指定transition动画，就可以查看到动画钩子

可以看到ngAnimate为即将要离开的ngView添加了ng-leave和ng-leave-active两个class；为即将要进入的ngView添加了ng-enter和ng-enter-active两个class

为这四个class编写特定的样式即可

详见Unit 4/my.css

十六、$http如何修改请求消息头部

$http.post()方法提交数据时，AngularJS默认设置请求消息头部：Content-Type:application/json;此格式是不符合HTTP协议规定的请求主体数据格式的。

HTTP协议规定的请求主体数据只能是下述三种格式之一：

<form action=”” method=”post” enctype=”三种格式之一”>

(1)

(2)

(3)