**中软国际考题-笔试**

1. 单选题

1、在scope中，如下那个符号用于绑定scope中的属性（D）

A.@ B.& C.$ D.=

2、关于angular的http服务的快捷方法错误的是（D）

A.$http.get(url字符串，config可选的配置-对象类型) 返回HttpPromise对象

B.$http.delete(url字符串，config可选的配置-对象类型) 返回HttpPromise对象

C.$http.head(url字符串，config可选的配置-对象类型) 返回HttpPromise对象

D.$http.json(url字符串，config可选的配置-对象类型) 返回HttpPromise对象

3、关于AngularJS，描述错误的是（D）

A. AngularJS 是一个 JavaScript 框架

B. 在 AngularJS 中，你可以在 HTML 中包含 HTML 文件

C. 控制器是 JavaScript 对象，由标准的 JavaScript 对象的构造函数 创建

D. AngularJS 使用$scope 对象来定义控制器

4、angular2中父组件如何向子组件传递数据（B）

A.@Output B.@Input C.#Output D.#Input

5、在AngularJS 使用动画需要引入？（A）

A. angular-animate.min.js B. myChange r-animate.min.js

C. transition -animate.min.js D. ngAnimate -animate.min.js

6、我们可以使用哪个属性让HTML元素产生动画效果？（A）

A. transition B. animate C. active D. show

7、设置隐藏 HTML 元素？（A）

A. ng-hide="true"

B. ng-hide="false"

C. ng-click="count = count + 1">

D. ng-click="toggle()"

8、下面哪一个绑定下拉菜单（）

A．ng-options

B．ng-repeat

C．ng-for

D．ng-select

1. 多选题
2. 在AngularJS中两个平级界面块 a 和 b，如果 a 中触发一个事件，有哪些方式能让 b 知道？（B、C、E）
3. 通过发送http请求
4. 通过 factory 可以生成一个单例对象，在需要通信的模块 a 和 b 中注入这个对象即可
5. 借助父 controller。在子 controller 中向父 controller 触发（ $emit ）一个事件，然后在父 controller 中监听（ $on ）事件，再广播（ $broadcast ）给子 controller
6. 将信息存入sessionStorage中共享信息。
7. 借助 $rootScope

2、下列是内置 filter的是（A、B、C、D、E）

A. date（日期） B. limitTo（限制数组或字符串长度） C. orderBy（排序）

D. json（格式化 json 对象） E. currency（货币）

3、ng内置的filter包括（A、B、C、D、F）

A．date B. currency C. limitTo D. orderBy E. jsonp F.number

4、Angular Directive中 restrict的设置以下正确的是?（ABC）

A. A匹配属性 B. E匹配标签 C. C匹配class D. N匹配注释

5、AngularJS 表达式可以包含文字、运算符和变量？（A、B、D、E）

A. 文字 B. 事件 C. 类

D. 运算符 E. 样式

E. 变量

6、 哪些可以绑定div中的内容()

A． {{}}

B． ng-bind

C． ng-model

D． ng-show

1. 填空题（每空1分，共10分）
2. \_\_\_\_\_\_\_\_\_模块是 Angular 自带的路由模块，\_\_\_\_\_\_\_\_\_ 模块是基于 ngRoute模块开发的第三方模块。(ngRoute 、ui.router)
3. 实现不同模块之间通信方式有\_\_\_\_\_\_\_\_\_\_，\_\_\_\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_\_\_\_\_。(Service, events指定绑定的事件, 使用 $rootScope, controller之间直接使用$parent或$$childHead, directive 指定属性进行数据绑定)
4. Angular采用\_\_\_\_\_\_\_\_\_\_机制实现数据的绑定。(脏检查机制)
5. ng内置的filter有九种：分别是： \_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 。( date、currency、limitTo、orderBy、lowercase、 uppercase、number、filter、 json )
6. Mvvm : \_\_\_\_\_\_\_\_\_\_ ( 双向数据绑定 )
7. 下拉框隐藏空白选项方法是（ng-show="false"）：
8. 使用\_\_\_ng-click\_\_\_\_\_\_指令绑定点击事件
9. 重复的DOM使用\_\_\_ng-repeat\_\_\_\_指令
10. 使用\_\_\_ng-app\_\_\_\_绑定一个应用程序
11. 使用\_\_\_ng-controller\_\_\_\_\_绑定一个控制器
12. 使用\_\_ng-model\_\_\_绑定表单元素
13. 判断题
14. 每个digest周期中，angular总会对比scope上model的值，一般digest周期都是自动触发的，我们也可以使用$apply进行手动触发。（ T ）
15. angular提供了强大的扩展能力，即使分属不同团队进行开发的 angular 应用也不会遇到不同模块之间的冲突。（ F ）
16. ng-repeat迭代数组的时候，如果数组中有相同值时会提示 Duplicates in a repeater are not allowed. 加 track by $index 并且不可解决。（ F ）
17. AngularJS 通过 指令 扩展了 HTML，且通过 表达式 绑定数据到 HTML。（ T ）
18. ng-bind 指令把元素值绑定至应用程序（F）
19. AngularJS 指令均是以 ng 作为前缀的 HTML 属性（T）
20. AngularJS控制器(controller)用于定义AngularJS应用（F）
21. ng-show/ng-hide实际上是通过控制dom节点的增删除来实现的。（ F ）
22. HTML 表单通常与 HTML 控件同时存在。（ T ）
23. angularjs是优秀的前端框架(正确)
24. angularjs比vue更好（错误）

五、简答题

1、angular 的数据绑定采用什么机制？详述原理?

答案：

脏检查机制。

双向数据绑定是 AngularJS 的核心机制之一。当 view 中有任何数据变化时，会更新到 model ，当 model 中数据有变化时，view 也会同步更新，显然，这需要一个监控。

原理就是，Angular 在 scope 模型上设置了一个 监听队列，用来监听数据变化并更新 view 。每次绑定一个东西到 view 上时 AngularJS 就会往 $watch 队列里插入一条 $watch ，用来检测它监视的 model 里是否有变化的东西。当浏览器接收到可以被 angular context 处理的事件时， $digest 循环就会触发，遍历所有的 $watch ，最后更新 dom。

* 1. ng-show/ng-hide 与 ng-if的区别？

答案：

ng-show/ng-hide实际上是通过display来进行隐藏和显示的。而ng-if实际上控制dom节点的增删除来实现的。因此如果我们是根据不同的条件来进行dom节点的加载的话，那么ng-if的性能好过ng-show.

### 3、ng-if 跟 ng-show/hide 的区别有哪些？（5分）

第一点区别是， ng-if 在后面表达式为 true 的时候才创建这个 dom 节点， ng-show 是初始时就创建了，用 display:block 和 display:none 来控制显示和不显示。

第二点区别是， ng-if 会（隐式地）产生新作用域， ng-switch 、 ng-include 等会动态创建一块界面的也是如此。

这样会导致，在 ng-if 中用基本变量绑定 ng-model ，并在外层 div 中把此 model 绑定给另一个显示区域，内层改变时，外层不会同步改变，因为此时已经是两个变量了。

4、angular 的数据绑定采用什么机制？（5分）

答案：采用了脏检查机制。Angular 在 scope 模型上设置了一个监听队列，这个监听队列可以用来监听数据变化并更新 view 。每次绑定一个东西到 view 上时， AngularJS 就会往 $watch 队列里插入一条 $watch ，用来检测它监视的 model 里是否有变化的东西。当浏览器接收到可以被 angular context 处理的事件时， $digest 循环就会触发，遍历所有的 $watch，最后更新 dom。

1. ng-click 中写的表达式，能使用 JS 原生对象上的方法吗？

答案：不仅是 ng-click 中，只要是在页面中，都是无法直接调用原生的 JS 方法的，因为这些并不存在于与页面对应的 Controller 的 $scope 中。

1. factory、service 和 provider 是什么关系？

答案：从底层实现上来看，三者的关系是：service 调用了 factory，返回其实例；factory 调用了 provider，返回其 $get 中定义的内容。factory 和 service 的功能类似，但是 factory 是普通 function，可以返回任何东西；service 是构造器，可以不返回（绑定到 this 的都可以被访问）；provider 是加强版 factory，返回一个可配置的 factory。

1. 列出至少三种实现不同模块之间通信方式。

答案：

(1)Service

(2)events,指定绑定的事件

(3)使用 $rootScope

(4)controller之间直接使用$parent, $$childHead等

(5)directive 指定属性进行数据绑定

1. 解释下什么是$rootScrope以及和$scope的区别？

答案：Angular解析ng-app然后在内存中创建$rootScope -> Angular会继续解析，找到{{}}表达式，并解析成变量。 -> 接着会解析带有ng-controller的div然后指向到某个controller函数。这个时候在这个controller函数变成一个$scope对象实例。

1. 表达式 {{yourModel}}是如何工作的？

答案： 它依赖于 $interpolation服务，在初始化页面html后，它会找到这些表达式，并且进行标记，于是每遇见一个{{}}，则会设置一个$watch。而$interpolation会返回一个带有上下文参数的函数，最后该函数执行，则算是表达式$parse到那个作用域上。

13. 一个 angular 应用在目录结构上应当如何良好地分层？

答案：

对于小型项目，可以按照文件类型组织，比如：css、js（controllers、models、services、filters）、templates

对于规模较大的项目，最好按业务模块划分，比如：

|-css

|-modules

|----- account

|----------- controllers

|-----------models

|-----------services

|-----------filters

|-----------templates

|-----disk

|-----------controllers

|-----------models

|-----------services

|-----------filters

|-----------templates

### 14、列出至少三种实现不同模块之间通信方式？

答案：Service; events指定绑定的事件; 使用$rootScope; controlle之间直接使用$parent, $$childHead等; directive指定属性进行数据绑定;

15、angularjs是什么?

AngularJS 是一个 JavaScript 框架。它可通过 <script> 标签添加到 HTML 页面。

AngularJS 通过 指令 扩展了 HTML，且通过 表达式 绑定数据到 HTML。

16、 解释下什么是$rootScrope以及和$scope的区别？

通俗的说$rootScrope 页面所有$scope的父亲。

我们来看下如何产生$rootScope和$scope吧。

step1:Angular解析ng-app然后在内存中创建$rootScope。

step2:angular回继续解析，找到{{}}表达式，并解析成变量。

step3:接着会解析带有ng-controller的div然后指向到某个controller函数。这个时候在这个controller函数变成一个$scope对象实例。

17、 ng-show/ng-hide 与 ng-if的区别？

我们都知道ng-show/ng-hide实际上是通过display来进行隐藏和显示的。而ng-if实际上控制dom节点的增删除来实现的。因此如果我们是根据不同的条件来进行dom节点的加载的话，那么ng-if的性能好过ng-show.

18、 表达式 {{yourModel}}是如何工作的？

它依赖于 $interpolation服务，在初始化页面html后，它会找到这些表达式，并且进行标记，于是每遇见一个{{}}，则会设置一个$watch。而$interpolation会返回一个带有上下文参数的函数，最后该函数执行，则算是表达式$parse到那个作用域上。

19、你认为在Angular中使用jQuery好么？

angular自带了很多api可以完全替代掉jQuery中常用的api，我们可以使用angular.element，$http,$timeout,ng-init等。但是两个框架一起使用时完全可以的，框架共存本身是各个框架稳定发展的前提。

六、程序实现题

1、自定义指令：调用ng的基本api，声明一个自定义指令。

（10分）

|  |
| --- |
| <html>  <script src='angular.js'></script>  <script type="text/javascript">  angular.module('myapp',[])  .directive('myCnblogs',function(){  return {  restrict: 'E',  replace: false,  template:'<a href="http://www.cnblogs.com/">自定义指令</a>'  }  })  </script>  <head>  <title></title>  <div ng-app='myapp'>  <my-Cnblogs></my-Cnblogs>  </div>  </head>  <body>  </body>  </html> |

2、使用angular实现购物车全选/取消全选

（10分）

|  |
| --- |
| <!DOCTYPE html>  <html lang="en" ng-app="testMo">  <head>  <meta charset="UTF-8">  <title></title>  <link rel="stylesheet" href="css/bootstrap.css">  <style>  .div1{  margin: 20px;  }  </style>  </head>  <body>  <div ng-controller="testCtrl" class="div1">  <h4>angularJS--购物车实现全选/取消全选</h4>  <button type="button" class="btn btn-info" ng-click="addProduct()">添加商品</button>  <button type="button" class="btn btn-danger" ng-click="deleteProduct()">删除商品</button>  <br><br>  <table class="table table-bordered table-responsive" >  <thead>  <td>操作</td>  <td>check状态</td>  <td>商品名称</td>  <td>单价</td>  <td>数量</td>  <td>小计</td>  </thead>  <tr ng-repeat="p in cart" >  <td><input type="checkbox" ng-checked="p.checked" ng-click="echoChange(p.id,p.checked,selectAll)"></td>  <td>{{p.checked}}||{{p.checked}}</td>  <td>{{p.name}}</td>  <td>单价：￥{{p.price}}</td>  <td>数量：<input type="number" ng-model="p.count" min="0" value="p.count"></td>  <td>小计：￥{{p.sum}}</td>  </tr>  </table>  <br>  <input type="checkbox" ng-model="selectAll" ng-click="selectAllClick(selectAll)"><span ng-hide="selectAll" >全选</span><span ng-show="selectAll">取消全选</span>  <br><br>  已选择<span>{{jishuqi}}</span>件商品，总金额：<span>￥{{ sumTotal }}</span>  </div>  <script src="../js/angular.js"></script>  <script>  angular.module('testMo',['ng']).controller('testCtrl',function($scope){  // $scope.p1=new Object();  // $scope.p1.price=10;  // $scope.p1.count=1;  //购物车应该是一个数组  $scope.selectAll=false;//全选默认为false  $scope.cart=[{id:0,name:'商品0',price:10,count:5,sum:10,checked:false}];  $scope.addProduct= function (){  var p=new Object();  p.id=$scope.cart.length;  p.name='商品'+ p.id  p.price=Math.floor(Math.random()\*100);//对数值向下取整  p.count=1;  p.sum= p.price\* p.count;  p.checked=false;  $scope.cart.push({id: p.id,name: p.name,price:p.price,count: p.count,sum: p.sum,checked: p.checked});  console.log($scope.cart);  }  //删除商品  $scope.deleteProduct= function (){  $scope.cart.pop();//删除数组中的最后的一个元素，并且返回这个元素，会改变数组里的元素  }  //全选按钮check的点击事件  $scope.selectAllClick= function (sa) {  for(var i=0;i<$scope.cart.length;i++){  $scope.cart[i].checked=sa;  }  }  //单个数据的check事件  $scope.echoChange=function(id,ch,se){  $scope.cart[id].checked=!ch;  //当所有都选中时，全选也要被勾选  var cc=0;//计算当前数组中checked为真的数目  for(var i=0;i<$scope.cart.length;i++){  // if($scope.cart[i].checked==true){  // cc++;  // }  $scope.cart[i].checked?cc++:cc;  }  $scope.selectAll=(cc==$scope.cart.length);//当为真的数目=数组长度时，证明全部勾选  // console.log($scope.selectAll);  }  //监控数据  $scope.$watch('cart',function(newValue,oldValue,scope){  $scope.sumTotal=0; //总计  $scope.jishuqi=0; //计数器  for(var i in newValue) {  var sumN = newValue[i].count \* newValue[i].price; //计算出新的结果  $scope.cart[i].sum = sumN.toFixed(2); //保留两位小数并且把它赋值给元数据;  if (newValue[i].checked) {  $scope.sumTotal += sumN;  $scope.jishuqi++;  // console.log($scope.sumTotal);  // console.log($scope.jishuqi);  }  }  },true);  /\*$watch简介:在digest执行时，如果watch观察的的value与上一次执行时不一样时，就会被触发。  AngularJS内部的watch实现了页面随model的及时更新。  $watch方法在用的时候主要是手动的监听一个对象，但对象发生变化时触发某个事件。  $watch(watchFn,watchAction,deepWatch);  如果不加第三个参数，那么只会监听cart数组，只有当cart引用改变时才会触发,因此当需要监听一些引用对象时需要把第三个参数设置成true。  \*/  });  </script>  </body>  </html> |

3、分别实现取消 $timeout, 以及停止一个$watch()?。（10分）

var customTimeout = $timeout(function () {

// your code

}, 1000);

$timeout.cancel(customTimeout);

function that we store to a variable

var deregisterWatchFn = $rootScope.$watch(‘someGloballyAvailableProperty’, function (newVal) {

if (newVal) {

// we invoke that deregistration function, to disable the watch

deregisterWatchFn();

...

}

});

4、使用AngularJS创建表格

（10分）

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <script src="https://cdn.bootcss.com/angular.js/1.6.3/angular.min.js"></script>  </head>  <body>  <div ng-app="myApp" ng-controller="customersCtrl">  <table>  <tr ng-repeat="x in names">  <td>{{ x.Name }}</td>  <td>{{ x.Country }}</td>  </tr>  </table>  </div>  <script>  var app = angular.module('myApp', []);  app.controller('customersCtrl', function($scope, $http) {  $http.get("/try/angularjs/data/Customers\_JSON.php")  .then(function (result) {  $scope.names = result.data.records;  });  });  </script> |

5、使用angularjs完成购物车程序

（10分）

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <title>angular购物车练习</title>  <link rel="stylesheet" href="../../vendor/bootstrap3/css/bootstrap.css">  <script src="../../vendor/angular/angular.min.js"></script>  <script src="app/index.js"></script>  </head>  <body ng-app="myApp" ng-controller="myCtrl">  <div class="container">  <table class="table" ng-show="cartList.length > 0">  <thead>  <tr>  <th>产品编号</th>  <th>产品名称</th>  <th>购买数量</th>  <th>产品单价</th>  <th>产品总价</th>  <th>操作</th>  </tr>  </thead>  <tbody>  <tr ng-repeat="item in cartList">  <td>{{item.id}}</td>  <td>{{item.name}}</td>  <td>  <button ng-click="reduceOne(item.id)" class="btn btn-primary">-</button>  <input type="text" ng-model="item.quantity">  <button ng-click="addOne(item.id)" class="btn btn-primary">+</button>  </td>  <td>{{item.price}}</td>  <td>{{item.quantity \* item.price}}</td>  <td><button ng-click="remove(item.id)" class="btn btn-danger">移除</button></td>  </tr>  <tr>  <td>总购买价</td>  <td>{{totalCost()}}</td>  <td>总购买数量</td>  <td>{{totalCount()}}</td>  <td colspan="3"><button ng-click="cartList=[]" class="btn btn-danger">清空购物车</button></td>  </tr>  </tbody>  </table>  <h4 ng-show="cartList.length < 1">您的购物车暂无商品</h4>  </div>    </body>  </html>  Index.js  var app=angular.module("myApp",[]);  app.controller("myCtrl",function($scope){  $scope.cartList=[  {id:1000,name:"iphone5s",quantity:3,price:4300},  {id:1001,name:"iphone5",quantity:30,price:3300},  {id:1002,name:"imac",quantity:3,price:3000},  {id:1003,name:"ipad",quantity:5,price:6900}  ];    var findIndex=function(id){  var index=-1;  angular.forEach($scope.cartList,function(item,key){  if(item.id == id){  index=key;  return;  }  });return index;  };    //从cartList数组中删除一项产品  $scope.remove=function(id){  var index=findIndex(id);  if(index != -1){  $scope.cartList.splice(index,1);  }  };    //为某个商品添加一个数量  $scope.addOne=function(id){  var index=findIndex(id);  if(index != -1){  $scope.cartList[index].quantity ++;  }  };  //位某个商品减少一个数量  $scope.reduceOne=function(id){  var index=findIndex(id);  if(index != -1){  var item=$scope.cartList[index];  if(item.quantity > 1){  item.quantity --;  }else{  var returnKey=confirm("删除该商品？");  if(returnKey){  $scope.remove(item.id);  }  }  }  };    //总购买价  $scope.totalCost=function(){  var total=0;  angular.forEach($scope.cartList,function(item,key){  total += item.quantity \* item.price;  });return total;  };  //总购买数量  $scope.totalCount=function(){  var count=0;  angular.forEach($scope.cartList,function(item,key){  count += item.quantity;  });return count;  };    //监听输入框更改事件避免输入负数或字符串  $scope.$watch('cartList',function(newValue,oldValue){  console.log( "$scope.cartList === newValue "+ ($scope.cartList === newValue) ); //永远为ture newValue指向cartList  console.log( "$scope.cartList === oldValue "+ ($scope.cartList === oldValue) ); //页面初始化后为true 一旦改动永远为false  angular.forEach(newValue,function(item,key){  if( isNaN(item.quantity) ){  item.quantity=oldValue[key].quantity;  }  else if( item.quantity <= 0 ){  var returnKey=confirm("删除该商品？");  if(returnKey){  $scope.remove(item.id);  }else{  item.quantity=oldValue[key].quantity;  }  }  });  },true);    }); |