

1、angular 内置过滤器一共有几种，分别是那些？

date：日期格式化

currency：货币

uppercase：大写

lowercase：小写

limitTo（限制数组或字符串长度）

orderBy（排序）

number（格式化数字，加上千位分隔符，并接收参数
限定小数点位数）

filter（处理一个数组，过滤出含有某个子串的元素）

json（格式化 json 对象）

2、angular 核心？

AngularJS 是为了克服 HTML 在构建应用上的不足而设计的。AngularJS 有着诸多特性，最为核心的是：

MVC

模块化

自动化双向数据绑定

语义化标签、依赖注入等等

3、angular 的数据绑定采用什么机制？详述原理

angularjs 的双向数据绑定采用脏检查 (dirty-checking) 机制。ng 只有在指定事件触发后，才进入 \$digest cycle：

- DOM 事件，譬如用户输入文本，点击按钮等。

(ng-click)

- XHR 响应事件 (\$http)

- 浏览器 Location 变更事件 (\$location)

- Timer 事件 (\$timeout , \$interval)

- 执行 \$digest() 或 \$apply()

4、依赖注入(DI)

让我们可以不用自己实例化就能创建依赖对象的方法。简单的来说，依赖是以注入的方式传递的。在 Web 应用中，Angular 让我们可以通过 DI 来创建像 Controllers 和 Directives 这样的对象。我们还可以创建自己的依赖对象，当我们要实例化它们时，Angular 能自动实现注入。

5、解释下什么是 \$rootScope 以及和 \$scope 的区别？

通俗的说 \$rootScope 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。

6、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。

7、factory 和 service, provider 是什么关系？

factory 把 service 的方法和数据放在一个对象里，并返回这个对象；service 通过构造函数方式创建 service，返回一个实例化对象；provider 创建一个可通过 config 配置的 service。

从底层实现上来看，service 调用了 factory，返回其实例；factory 调用了 provider，将其定义的内容放在 \$get 中返回。factory 和 service 功能类似，只不过 factory 是普通 function，可以返回任何东西（return 的都可以被访问，所以那些私有变量怎么写你懂的）；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。

8、ng-if 跟 ng-show/hide 的区别有哪些？

1. 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。
2. 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。

9、ng-repeat 迭代数组的时候，如果数组中有相同值，会有什么问题，如何解决？

会提示 Duplicates in a repeater are not allowed.

加 track by \$index 可解决。当然，也可以 trace by 任何一个普通的值，只要能唯一性标识数组中的每一项即可（建立 dom 和数据之间的关联）。

10、AngularJS 的数据双向绑定是怎么实现的？

1、每个双向绑定的元素都有一个 watcher

2、在某些事件发生的时候，调用 digest 脏数据检测。

这些事件有：表单元素内容变化、Ajax 请求响应、点击按钮执行的函数等。

3、脏数据检测会检测 rootscope 下所有被 watcher 的元素。

\$digest 函数就是脏数据监测

11、angular 的数据绑定采用什么机制？详述原理。

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。

12、单页应用有哪些优缺点？

单页 Web 应用（single page web application, SPA），就是只有一张 Web 页面的应用。单页应用程序 (SPA) 是加载单个 HTML 页面并在用户与应用程序交互时动态更新该页面的 Web 应用程序。浏览器一开始会加载必需的 HTML、CSS 和 JavaScript，所有的操作都在这张页面上完成，都由 JavaScript 来控制。因此，对单页应用来说模块化的开发和设计显得相当重要。

- 速度：更好的用户体验，让用户在 web app 感受 native app 的速度和流畅，
- MVC：经典 MVC 开发模式，前后端各负其责。
- ajax：重前端，业务逻辑全部在本地操作，数据都需要通过 AJAX 同步、提交。
- 路由：在 URL 中采用 # 号来作为当前视图的地址，改变 # 号后的参数，页面并不会重载。

单页 Web 应用（single page web application, SPA）是当今网站开发技术的弄潮儿，很多传统网站都在或者已经转型为单页 Web 应用，新的单页 Web 应用网站（包括移动平台上的）也如雨后春笋般涌现在人们的面前，如 Gmail、Evernote、Trello 等。如果你是一名 Web 开发人员，却还没开发过或者甚至是没有听说过单页应用，那你已经 Out 很久了。

单页 Web 应用和前端工程师们息息相关，因为主要的变革发生在浏览器端，用到的技术其实还是 HTML+CSS+JavaScript，所有的浏览器都原生支持，当然有的浏览器因为具备一些高级特性，从而使得单页 Web 应用的用户体验更上一层楼。关于单页应用的优点和缺点，网上讲解的文章有很多，这里就不展开论述了。单页 Web 应用，顾名思义，就是只有一张 Web 页面的应用。浏览器一开始会加载必需的 HTML、CSS 和 JavaScript，之后所有的操作都在这张页面上完成，这一切都由 JavaScript 来控制。因此，单页 Web 应用会包含大量的 JavaScript 代码，复杂度可想而知，模块化开发和设计的重要性不言而喻。

优点：

1. 分离前后端关注点，前端负责界面显示，后端负责数据存储和计算，各司其职，不会把前后端的逻辑混杂在一起；
2. 减轻服务器压力，服务器只输出数据就可以，不用管展示逻辑和页面合成，吞吐能力会提高几倍；
3. 同一套后端程序代码，不用修改就可以用于 Web 界面、手机、平板等多种客户端；

缺点：

1. SEO 问题，现在可以通过 Prerender 等技术解决一部分；
2. 前进、后退、地址栏等，需要程序进行管理；
3. 书签，需要程序来提供支持；

13、`{{now | 'yyyy-MM-dd'}}` 这种表达式里面，竖线和后面的参数通过什么方式可以自定义？

```
app.filter('过滤器名称',function(){
```

```
    return function(需要过滤的对象, 过滤器参数 1, 过滤器参数 2, ...){
```

```
        //...做一些事情
```

```
        return 处理后的对象;
```

```
    }
```

```
});
```

14、Angular Directive 中 restrict 中分别可以怎样设置？
scope 中 @,=,& 有什么区别？

restrict 中可以分别设置：

- A 匹配属性
- E 匹配标签
- C 匹配 class
- M 匹配注释

在 scope 中，@,=,&在进行值绑定时分别表示

- @获取一个设置的字符串，它可以自己设置的也可以使用{{yourModel}}进行绑定的；
- = 双向绑定，绑定 scope 上的一些属性；
- & 用于执行父级 scope 上的一些表达式，常见我们设置一些需要执行的函数



```
angular.module('docsIsolationExample', [])  
  
.controller('Controller', ['$scope', function($scope) {  
  
    $scope.alertName = function() {  
  
        alert('directive scope &');  
  
    }  
  
}])  
  
.directive('myCustomer', function() {  
  
    return {  
  
        restrict: 'E',  
  
        scope: {  
  
            clickHandle: '&  
  
        },  
  
    },  
  
}
```

```

    template: '<button ng-click="testClick()">Click
Me</button>',

    controller: function($scope) {

        $scope.testClick = function() {

            $scope.clickHandle();

        }

    }

};

});

```



```

<div ng-app="docsIsolationExample"> <div
ng-controller="Controller">

    <my-customer

click-handle="alertName()"></my-customer></div>

</div>

```

15、 有哪些措施可以改善 Angular 性能

使用一次绑定表达式即`{{::yourModel}}`

减少 watcher 数量

在无限滚动加载中避免使用 `ng-repeat`, 关于解决方法可以参考这篇文章

使用性能测试的小工具去挖掘你的 angular 性能问题, 我们可以使用简单的 `console.time()` 也可以借助开发者工具以及 Batarang

16、你认为在 Angular 中使用 jQuery 好么?

这是一个开放性的问题, 尽管网上会有很多这样的争论, 但是普遍还是认为这并不是一个特别好的尝试。其实当我们学习 Angular 的时候, 我们应该做到从 0 去接受 angular 的思想, 数据绑定, 使用 angular 自带的一些 api, 合理的路由组织和, 写相关指令和服务等等。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。

我们不妨再换个角度, 如果业务需求, 而对于一个新人 (比较熟悉 jQuery) 的话, 或许你引入 jQuery 可以让它在解决问题, 比如使用插件上有更多的选择, 当然这是通过影响代

码组织来提高工作效率，随着对于 angular 理解的深入，在重构时会逐渐摒弃掉当初引入 jquery 时的一些代码。

所以我觉得两种框架说完全不能一起用肯定是错的，但是我们还是应该尽力去遵循 angular 的设计。

17、列出至少三种实现不同模块之间通信方式？

Service

events, 指定绑定的事件

使用 \$rootScope

controller 之间直接使用 \$parent, \$\$childHead 等

directive 指定属性进行数据绑定

18、angular 中的 \$http

\$http 是 AngularJS 中的一个核心服务，用于读取远程服务器的数据。

我们可以使用内置的 \$http 服务直接同外部进行通信。

\$http 服务只是简单的封装了浏览器原生的 XMLHttpRequest 对象。

19、写 controller 逻辑的时候 你需要注意什么？

1. 简化代码（这个是所有开发人员都要具备的）

2. 坚决不能操作 dom 节点 这个时候可能会问 为什么

不能啊

你的回答是: DOM 操作只能出现在指令(directive)中。最不应该出现的位置就是服务(service)中。Angular 倡导以测试驱动开发, 在 service 或者 controller 中出现了 DOM 操作, 那么也就意味着测试是无法通过的。当然, 这只是一点, 重要的是使用 Angular 的其中一个好处是啥, 那就是双向数据绑定, 这样就能专注于处理业务逻辑, 无需关系一堆堆的 DOM 操作。如果在 Angular 的代码中还到处充斥着各种 DOM 操作, 那为什么不直接使用 jquery 去开发呢。

20、angular 和 jquery 的区别

angular 是基于数据驱动, 所以 angular 适合做数据操作比较繁琐的项目(这里可以再提一下单页面应用, 如果你不会福利又来

了 <http://www.zhihu.com/question/20792064>)

jquery 是基于 dom 驱动, jquery 适合做 dom 操作多的项目

21、阐述下你对 mvc 和 mvvm 的理解

随着代码规模越来越大, 切分职责是大势所趋, 还有为了后期维护方便, 修改一块功能不影响其他功能。还有为了

复用，因为很多逻辑是一样的。而 MVC 只是手段，终极目标是模块化和复用。

在 angular 中 MVVM 模式主要分为四部分：

View: 它专注于界面的显示和渲染，在 angular 中则是包含一堆声明式 Directive 的视图模板。

ViewModel: 它是 View 和 Model 的粘合体，负责 View 和 Model 的交互和协作，它负责给 View 提供显示的数据，以及提供了 View 中 Command 事件操作 Model 的途径；在 angular 中 \$scope 对象充当了这个 ViewModel 的角色；

Model: 它是与应用程序的业务逻辑相关的数据的封装载体，它是业务领域的对象，Model 并不关心会被如何显示或操作，所以模型也不会包含任何界面显示相关的逻辑。在 web 页面中，大部分 Model 都是来自 Ajax 的服务端返回数据或者是全局的配置对象；而 angular 中的 service 则是封装和处理这些与 Model 相关的业务逻辑的场所，这类的业务服务是可以被多个 Controller 或者其他 service 复用的领域服务。

Controller: 这并不是 MVVM 模式的核心元素，但它负责 ViewModel 对象的初始化，它将组合一个或者多个 service 来获取业务领域 Model 放在 ViewModel 对象上，使得应用界面在启动加载的时候达到一种可用的状态。

mvc 的界面和逻辑关联紧密，数据直接从数据库读取。
mvvm 的界面与 viewmode 是松耦合，界面数据从 viewmodel 中获取。所以 angularjs 更倾向于 mvvm

mvvm 的优点：

低耦合：View 可以独立于 Model 变化和修改，同一个 ViewModel 可以被多个 View 复用；并且可以做到 View 和 Model 的变化互不影响；

可重用性：可以把一些视图的逻辑放在 ViewModel，让多个 View 复用；

独立开发：开发人员可以专注与业务逻辑和数据的开发（ViewModemvvm 开发人员可以专注于 UI(View)的设计；