2016 年前端开发工程师面试题

HTML+CSS 部分

1.行内元素和块级元素?img 算什么?行内元素怎么转化为块级元素?

行内元素：和有他元素都在一行上，高度、行高及外边距和内边距都不可改变，文字图片的宽度不可改变，只能容纳文本或者其他行内元素；其中 img 是行元素

块级元素：总是在新行上开始，高度、行高及外边距和内边距都可控制，可以容纳内敛元素和其他元素；

行元素转换为块级元素方式：display：block；

2.将多个元素设置为同一行?清除浮动有几种方式?

将多个元素设置为同一行：float，inline-block

清除浮动的方式：方法一：添加新的元素 、应用 clear：both；

方法二：父级 div 定义 overflow: hidden；

方法三：利用:after 和:before 来在元素内部插入两个元素块，从面达到清除浮动的效果。

.clear{zoom:1;}

.clear:after{content:"";clear:both;display:block;height:0;overflow:hidden;visibility:hidden;}

3.怪异盒模型 box-sizing？弹性盒模型|盒布局?

在标准模式下的盒模型：盒子总宽度/高度=width/height+padding+border+margin

在怪异模式下的盒模型下，盒子的总宽度和高度是包含内边距 padding 和边框 border 宽度在内的，盒子总宽度/高度=width/height + margin = 内容区宽度/高度 + padding + border + margin;

box-sizing 有两个值一个是 content-box，另一个是 border-box。

当设置为 box-sizing:content-box 时，将采用标准模式解析计算；

当设置为 box-sizing:border-box 时，将采用怪异模式解析计算。

4.简述几个 css hack?

（1）图片间隙

在 div 中插入图片，图片会将 div 下方撑大 3px。hack1：将 与<img>写在同一行。hack2：给<img>添加 display：block；

dt li 中的图片间隙。hack：给<img>添加 display：block；

（2）默认高度，IE6 以下版本中，部分块元素，拥有默认高度（低于 18px）

hack1：给元素添加：font-size：0；

hack2：声明：overflow：hidden；

表单行高不一致

hack1：给表单添加声明：float：left；height： ；border： 0；

鼠标指针

hack：若统一某一元素鼠标指针为手型：cursor：pointer；

当 li 内的 a 转化块元素时，给 a 设置 float，IE 里面会出现阶梯状

hack1：给 a 加 display：inline-block；
hack2：给 li 加 float：left；
5.link import 两者之间区别?
（1）老祖宗的差别，@important 只能加载 css
（2）加载顺序的差别，最后加载 import
（3）兼容性的差别，老浏览器不兼容
（4）使用 dom 控制样式的差别
6.工作职责?公司人员?5：3 前后端比例?为甚离职?
7.href 和 src 区别?title 和 alt

href (Hypertext Reference)指定网络资源的位置（超文本引用），从而在当前元素或者当前文档和由当前属性定义的需要的锚点或资源之间定义一个链接或者关系，在 link 和 a 等元素上使用。

src (Source)属性仅仅嵌入当前资源到当前文档元素定义的位置，是页面必不可少的一部分，是引入。在 img、script、iframe 等元素上使用。

title：既是 html 标签，又是 html 属性，title 作为属性时，用来为元素提供额外说明信息.

alt：alt 是 html 标签的属性，alt 属性则是用来指定替换文字，只能用在 img、area 和 input 元素中（包括 applet 元素），用于网页中图片无法正常显示时给用户提供文字说明使其了解图像信息.

8.transform？animation？区别?animation-duration

Transform:它和 width、left 一样，定义了元素很多静态样式实现变形、旋转、缩放、移位及透视等功能，通过一系列功能的组合我们可以实现很炫酷的静态效果（非动画)。

Animation:作用于元素本身而不是样式属性,属于关键帧动画的范畴，它本身被用来替代一些纯粹表现的 javascript 代码而实现动画,可以通过 keyframe 显式控制当前帧的属性值.

animation-duration：规定完成动画所花费的时间，以秒或毫秒计。

9.nth-of-type | nth-child?
举例说明：

```
<ul>
<p>111</p>
<span>222<span>
<li>1</li>
<li>2</li>
<li>3</li>
</ul>
```

li:nth-of-type(2):表示 ul 下的第二个 li 元素
li:nth-child(2)：表示既是 li 元素又是在 ul 下的第二个元素（找不到）。
建议一般使用 nth-of-type，不容易出问题。
10 .:before 和 ::before 区别?
单冒号(:)用于 CSS3 伪类，双冒号(::)用于 CSS3 伪元素。
对于 CSS2 之前已有的伪元素，比如:before，单冒号和双冒号的写法::before 作用是一样的。
11.如何让一个 div 上下左右居中?
答：有三种方法。
方法 1：

```
.div1{
        width:400px;
```

```
        height:400px;
        border:#CCC 1px solid;
        background:#99f;
        position:absolute;
        left:50%;
        top:50%;
        transform: translate(-50%,-50%);
}
<div class="div1"></div>
```

方法 2：

```
.div2{
    width:400px;
    height:400px;
    border:#CCC 1px solid;
    background:#99f;
    position: absolute;
    left:0;
    top: 0;
    bottom: 0;
    right: 0;
    margin: auto;
}
<div class="div2"></div>
```

方法 3：

```
.div3{
    width:400px;
    height:400px;
    border:#CCC 1px solid;
    background:#9f9;
    position: absolute;
    left: 50%;
    top:50%;
    margin-left:-200px;
    margin-top: -200px;
}
<div class="div3"></div>
```

12.css2.0 和 css3.0

答：css3 加强了 css2 的功能，增加了新的属性和新的标签，并且删除了一些冗余的标签，在布局方面减少了代码量。以前比较复杂的布局现在一个属性就搞定了（columns 之类的属性）。在效果方面加入了更多的效果(圆角，动画之类的)，还有在盒子模型和列表模块都进行了改进。不过 CSS3 兼容性不好，只有一些高级版本的浏览器支持。

13.弹性盒子模型?flex|box 区别?

答：（1）引入弹性盒布局模型的目的是提供一种更加有效的方式来对一个容器中的条目进行排列、对齐和分配空白空间。即便容器中条目的尺寸未知或是动态变化的，弹性盒布局模

型也能正常的工作。在该布局模型中，容器会根据布局的需要，调整其中包含的条目的尺寸和顺序来最好地填充所有可用的空间。当容器的尺寸由于屏幕大小或窗口尺寸发生变化时，其中包含的条目也会被动态地调整。比如当容器尺寸变大时，其中包含的条目会被拉伸以占满多余的空白空间；当容器尺寸变小时，条目会被缩小以防止超出容器的范围。弹性盒布局是与方向无关的。在传统的布局方式中，block 布局是把块在垂直方向从上到下依次排列的；而 inline 布局则是在水平方向来排列。弹性盒布局并没有这样内在的方向限制，可以由开发人员自由操作。

（2）flex 和 box 的区别:

display：box 是老规范，要兼顾古董机子就加上它；

父级元素有 display:box;属性之后。他的子元素里面加上 box-flex 属性。可以让子元素按照父元素的宽度进行一定比例的分占空间。

flex 是最新的，董机老机子不支持的；

父元素设置 display:flex 后，子元素宽度会随父元素宽度的改变而改变，而 display:box 不会。

Android UC 浏览器只支持 display: box 语法；而 iOS UC 浏览器则支持两种方式。

14.viewport 所有属性 ？

答： (1)width :设置 layout viewport 的宽度，为一个正整数，或字符串'device-width';

(2)initial-scale:设置页面的初始缩放值，为一个数字，可以带小数。

(3)minimum-scale:允许用户的最小缩放值，为一个数字，可以带小数。

(4)maximum-scale:允许用户的最大缩放值，为一个数字，可以带小数。

(5)height:设置 layout viewport 的高度，这个属性对我们并不重要，很少使用

(6)user-scalable:是否允许用户进行缩放，值为‘no’或者‘yes’。

安卓中还支持：target-densitydpi，表示目标设备的密度等级，作用是决定 css 中的 1px 代表多少物理像素

(7)target-densitydpi:值可以为一个数值或者 high-dpi 、 medium-dpi、 low-dpi、device-dpi 这几个字符串中的一个

15. 如何理解 HTML 结构的语义化？

所谓标签语义化，就是指标签的含义。语义化的主要目的就是让大家直观的认识标签(markup)和属性(attribute)的用途和作用，对搜索引擎友好，有了良好的结构和语义我们的网页内容便自然容易被搜索引擎抓取，这种符合搜索引擎收索规则的做法，网站的推广便可以省下不少的功夫，而且可维护性更高，因为结构清晰,十分易于阅读。这也是搜索引擎优化 SEO 重要的一步。

16. 伪类选择器和伪元素？c3 中引入的伪类选择器有？c3 中伪元素有?

伪类用一个冒号来表示，而伪元素则用两个冒号来表示。

伪类选择器：

由于状态是动态变化的，所以一个元素达到一个特定状态时，它可能得到一个伪类的样式；当状态改变时，它又会失去这个样式。

伪元素选择器：

并不是针对真正的元素使用的选择器，而是针对 CSS 中已经定义好的伪元素使用的选择器；

c3 中引入的伪类选择器：

:root()选择器，根选择器，匹配元素 E 所在文档的根元素。在 HTML 文档中，根元素始终是<html>。:root 选择器等同于<html>元素。

:not()选择器称为否定选择器，和 jQuery 中的:not 选择器一模一样，可以选择除某个元素之外的所有元素。

:empty()选择器表示的就是空。用来选择没有任何内容的元素，这里没有内容指的是一点内容都没有，哪怕是一个空格。
:target()选择器来对页面某个 target 元素(该元素的 id 被当做页面中的超链接来使用)指定样式，该样式只在用户点击了页面中的超链接，并且跳转到 target 元素后起作用。
:first-child()选择器表示的是选择父元素的第一个子元素的元素 E。简单点理解就是选择元素中的第一个子元素，记住是子元素，而不是后代元素。
:nth-child()选择某个元素的一个或多个特定的子元素。
:nth-last-child()从某父元素的最后一个子元素开始计算，来选择特定的元素
:nth-of-type(n)选择器和:nth-child(n)选择器非常类似，不同的是它只计算父元素中指定的某种类型的子元素。
:only-child 表示的是一个元素是它的父元素的唯一一个子元素。
:first-line 为某个元素的第一行文字使用样式。
:first-letter 为某个元素中的文字的首字母或第一个字使用样式。
:before 在某个元素之前插入一些内容。
:after 在某个元素之后插入一些内容。
c3 中伪元素：
::first-line 选择元素的第一行，比如说改变每个段落的第一行文本的样式
::before 和::after 这两个主要用来给元素的前面或后面插入内容，这两个常用"content"配合使用，见过最多的就是清除浮动
::selection 用来改变浏览网页选中文的默认效果
17、html5 有哪些新特性、移除了那些元素？如何处理 HTML5 新标签的浏览器兼容问题？如何区分 HTML 和 HTML5？
* HTML5 现在已经不是 SGML 的子集，主要是关于图像，位置，存储，多任务等功能的增加。
* 拖拽释放(Drag and drop) API
语义化更好的内容标签（header,nav,footer,aside,article,section）
音频、视频 API(audio,video)
画布(Canvas) API
地理(Geolocation) API
本地离线存储 localStorage 长期存储数据，浏览器关闭后数据不丢失； sessionStorage 的数据在浏览器关闭后自动删除
表单控件，calendar、date、time、email、url、search
新的技术 webworker, websocket, Geolocation

* 移除的元素 纯表现的元素：basefont，big，center，font, s，strike，tt，u；
对可用性产生负面影响的元素：frame，frameset，noframes；

支持 HTML5 新标签：
* IE8/IE7/IE6 支持通过 document.createElement 方法产生的标签， 可以利用这一特性让这些浏览器支持 HTML5 新标签， 浏览器支持新标签后，还需要添加标签默认的样式：
* 当然最好的方式是直接使用成熟的框架、使用最多的是 html5shim 框架
<!--[if lt IE 9]>
<script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
<![endif]-->

* 如何区分： DOCTYPE 声明\新增的结构元素\功能元素

18. placeholder?如何在 ie8 上兼容 placeholder 这个效果

方法一：

首先判断浏览器是否支持 placeholder 属性，如果不支持的话，就遍历所有 input 输入框，获取 placeholder 属性的值填充进输入框作为提示信息，同时字体设置成灰色。

当输入框获得焦点(focus)同时输入框内文字等于设置的提示信息时，就把输入框内清空；

当输入框失去焦点(blur)同时输入框内文字为空时，再把获取的 placeholder 属性的值填充进输入框作为提示信息，同时字体设置成灰色；

当输入框内有输入(keydown)时，此时输入框内的提示信息已经由 focus 事件清除，此时只需要把字体再恢复成黑色即可

此方法的缺点是，不适用于加载完 DOM 时即获得焦点的输入框，因为在用户的角度，加载完页面时看到的获得焦点的那个输入框，它的提示文字是看不到的。

HTML：

```
<input type="text" id="uname" name="uname" placeholder="请输入用户名"/>
```

CSS：

```
.phcolor{ color:#999;}
```

JS

```
$(function(){
 //判断浏览器是否支持 placeholder 属性
 supportPlaceholder='placeholder'in document.createElement('input'),
 placeholder=function(input){
    var text = input.attr('placeholder'),
    defaultValue = input.defaultValue;
    if(!defaultValue){
      input.val(text).addClass("phcolor");
    }
    input.focus(function(){
      if(input.val() == text){
        $(this).val("");
      }
    });
    input.blur(function(){
      if(input.val() == ""){
        $(this).val(text).addClass("phcolor");
      }
    });
    //输入的字符不为灰色
    input.keydown(function(){
      $(this).removeClass("phcolor");
    });
 };
```

```
 //当浏览器不支持 placeholder 属性时，调用 placeholder 函数
 if(!supportPlaceholder){
    $('input').each(function(){
      text = $(this).attr("placeholder");
      if($(this).attr("type") == "text"){
        placeholder($(this));
      }
    });
 }
});
```

方法二：

此方法的思路是做一张含有提示文字的图片作为 input 输入框的背景图，初始时获得焦点同时加载背景图；

当输入框不为空时，去掉背景图；

当输入框为空时，加载背景图；

当用户键盘按键且输入框不为空(输入字符)时，去掉背景图；

当用户键盘按键且输入框为空(删除字符)时，加载背景图。

此方法的缺点是：需要为每一个提示文字不同的 input 制作背景图片，并且设置 input 的样式。

CSS：

```
.phbg{ background:url(img/bg.jpg) 0 0 no-repeat;}
```

JS

```
$(function(){
   //判断浏览器是否支持 placeholder 属性
   supportPlaceholder='placeholder' in document.createElement('input');
   if(!supportPlaceholder){
     //初始状态添加背景图片
     $("#uname").attr("class","phbg");
     //初始状态获得焦点
     $("#uname").focus;
     function destyle(){
       if($("#uname").val() != ""){
         $("#uname").removeClass("phbg");
       }else{
         $("#uname").attr("class","phbg");
         }
     }
     //当输入框为空时，添加背景图片；有值时去掉背景图片
     destyle();
     $("#uname").keyup(function(){
       //当输入框有按键输入同时输入框不为空时，去掉背景图片；有按键输入同时为空时（删除字符），添加背景图片
       destyle();
```

```
    });
    $("#uname").keydown(function(){
      //keydown 事件可以在按键按下的第一时间去掉背景图片
      $("#uname").removeClass("phbg");
    });
  }
});
```

方法三：
使用插件：Placeholders.js
19. 常见兼容性问题？
* png24 位的图片在 iE6 浏览器上出现背景，解决方案是做成 PNG8.也可以引用一段脚本处理.
* 浏览器默认的 margin 和 padding 不同。解决方案是加一个全局的*{margin:0;padding:0;}来统一。
* IE6 双边距 bug:块属性标签 float 后，又有横行的 margin 情况下，在 ie6 显示 margin 比设置的大。
* 浮动 ie 产生的双倍距离（IE6 双边距问题：在 IE6 下，如果对元素设置了浮动，同时又设置了 margin-left 或 margin-right，margin 值会加倍。）
 #box{ float:left; width:10px; margin:0 0 0 100px;}
这种情况之下 IE 会产生 20px 的距离，解决方案是在 float 的标签样式控制中加入——_display:inline;将其转化为行内属性。(_这个符号只有 ie6 会识别)

* 渐进识别的方式，从总体中逐渐排除局部。
 首先，巧妙的使用“\9”这一标记，将 IE 游览器从所有情况中分离出来。
 接着，再次使用“+”将 IE8 和 IE7、IE6 分离开来，这样 IE8 已经独立识别。
 css

```
.bb{
  background-color:#f1ee18;
.background-color:#00deff\9;
+background-color:#a200ff;
_background-color:#1e0bd1;
}
```

* IE 下,可以使用获取常规属性的方法来获取自定义属性,
 也可以使用 getAttribute()获取自定义属性;
 Firefox 下,只能使用 getAttribute()获取自定义属性.
 解决方法:统一通过 getAttribute()获取自定义属性.

* IE 下,event 对象有 x,y 属性,但是没有 pageX,pageY 属性;
 Firefox 下,event 对象有 pageX,pageY 属性,但是没有 x,y 属性.
* 解决方法：（条件注释）缺点是在 IE 浏览器下可能会增加额外的 HTTP 请求数。

* Chrome 中文界面下默认会将小于 12px 的文本强制按照 12px 显示,
 可通过加入 CSS 属性 -webkit-text-size-adjust: none; 解决.

* 超链接访问过后 hover 样式就不出现了 被点击访问过的超链接样式不在具有 hover 和 active 了解决方法是改变 CSS 属性的排列顺序:
L-V-H-A : a:link {} a:visited {} a:hover {} a:active {}

* 怪异模式问题：漏写 DTD 声明，Firefox 仍然会按照标准模式来解析网页，但在 IE 中会触发怪异模式。为避免怪异模式给我们带来不必要的麻烦，最好养成书写 DTD 声明的好习惯。现在可以使用[html5](http://www.w3.org/TR/html5/single-page.html)推荐的写法：
`<doctype html>`

* 上下 margin 重合问题
ie 和 ff 都存在，相邻的两个 div 的 margin-left 和 margin-right 不会重合，但是 margin-top 和 margin-bottom 却会发生重合。
解决方法，养成良好的代码编写习惯，同时采用 margin-top 或者同时采用 margin-bottom。
* ie6 对 png 图片格式支持不好(引用一段脚本处理)
20.简述前端优化的方式 旧的雅虎 34 条|h5 新添加的方式
1、尽量减少 HTTP 请求次数
2、减少 DNS 查找次数
3、避免跳转
4、可缓存的 AJAX
5、推迟加载内容
6、预加载
7、减少 DOM 元素数量
8、根据域名划分页面内容
9、使 iframe 的数量最小
10、不要出现 404 错误
11、使用内容分发网络
12、为文件头指定 Expires 或 Cache-Control 13、Gzip 压缩文件内容
14、配置 ETag
15、尽早刷新输出缓冲
16、使用 GET 来完成 AJAX 请求
17、把样式表置于顶部
18、避免使用 CSS 表达式（Expression）
19、使用外部 JavaScript 和 CSS
20、削减 JavaScript 和 CSS
21、用<link>代替@import
22、避免使用滤镜
23、把脚本置于页面底部
24、剔除重复脚本
21.jquery 版本?1.11 兼容?
Query 2.x 系列和 jQuery 1.x 拥有同样的 API，但是不再支持 IE6、7、8。
推荐使用 1.x version，除非你确定 IE6、7、8 用户不再访问网站。
jquery1.11 属于 1.x 版本，其兼容 IE6、7、8，所以也支持 IE9.
22.块级？行内？空元素?
答：行内元素：和有他元素都在一行上，高度、行高及外边距和内边距都不可改变，文字图

片的宽度不可改变，只能容纳文本或者其他行内元素
块级元素：总是在新行上开始，高度、行高及外边距和内边距都可控制，可以容纳内敛元素和其他元素；
空元素：在 HTML 元素中，没有内容的 HTML 元素被称为空元素。空元素是在开始标签中关闭的。
 就是没有关闭标签的空元素。
23.media 属性？screen？All？max-width?min-width?
答：media 属性规定被链接文档将显示在什么设备上。media 属性用于为不同的媒介类型规定不同的样式。Screen 计算机默认屏幕，all 适用于所有设备，max-width 超过最大宽度就不执行，min-width 必须大于最小宽度才执行。
24.meta 标签的 name 属性值？
答：name 属性主要用于描述网页，与之对应的属性值为 content，content 中的内容主要是便于搜索引擎机器人查找信息和分类信息用的。meta 标签的 name
属性语法格式是：＜meta name="参数" content="
具体的参数值"＞。其中 name 属性主要有以下几种参数：A 、Keywords(关键字)说明：keywords 用来告诉搜索引擎你网页的关键字是什么。B 、description(网站内容描述) 说明：description 用来告诉搜索引擎你的网站主要内容.
C robots(机器人向导)说明：
robots 用来告诉搜索机器人哪些页面需要索引，哪些页面不需要索引。content 的参数有 all,none,index,noindex,follow,nofollow
。默认是 all。举例：＜meta name="robots" content="none"＞D 、author(作者)
25.一般做手机页面切图有几种方式?
答：三种。响应式布局，弹性布局 display：flex，利用 js 编写设定比例，给根元素设定像素，使用 rem 为单位。
px/em/rem 有什么区别？ 为什么通常给 font-size 设置的字体为 62.5%
答：px 像素是相对长度单位。像素 px 是相对于显示器屏幕分辨率而言的。em 是相对长度单位。相对于当前对象内文本的字体尺寸。如当前对行内文本的字体尺寸未被人为设置，则相对于浏览器的默认字体尺寸。1、em 的值并不是固定的；2、em 会继承父级元素的字体大小。使用 rem 为元素设定字体大小时，仍然是相对大小，但相对的只是 HTML 根元素。这个单位可谓集相对大小和绝对大小的优点于一身，通过它既可以做到只修改根元素就成比例地调整所有字体大小，又可以避免字体大小逐层复合的连锁反应。
rem 是相对于浏览器进行缩放的。1rem 默认是 16px，在响应式布局中，一个个除来转换成 rem，太麻烦，所以重置 rem
body{font-size=62.5% } 此时 1rem = 10px;若是 12px 则是 1.2rem.
26.sass 和 scss 有什么区别?sass 一般怎么样编译的
答：文件扩展名不同，Sass 是以“.sass”后缀为扩展名，而 SCSS 是以“.scss”后缀为扩展名；语法书写方式不同，Sass 是以严格的缩进式语法规则来书写，不带大括号({})和分号(;)，而 SCSS 的语法书写和我们的 CSS 语法书写方式非常类似。
27.如果对 css 进行优化 如何处理？
答：压缩打包，图片整合，避免使用 Hack，解决兼容问题，使用简写，让 CSS 能保证日后的维护。
28.如何对 css 文件进行压缩合并? gulp ?
答：使用 gulp。
1、首页全局安装 gulp。
1 npm install --global gulp

2、其次局部安装 gulp。npm install gulp --save-dev
3、在项目根目录下创建一个名为 gulpfile.js 的文件
var gulp = require('gulp');
gulp.task('default', function() {
// 将你的默认的任务代码放在这});
4、运行 gulp。（默认的名为 default 的任务（task）将会被运行，想要单独执行特定的任务（task），请输入 gulp <task> <othertask>）
gulp
合并和压缩 JS、CSS 文件
压缩 JS，CSS 文件需要引用如下组件：
gulp-minify-css: 压缩 css
29.组件? 模块化编程?
组件化编程: 将 js css html 包装一起提供方法和效果；
模块化化: 将相同的功能抽取出来 存放在一个位置进行编程
30.图片和文字在同一行显示?
1 在 css 中给 div 添加上“vertical-align:middle”属性。
2 分别把图片和文字放入不同的 div 中，然后用“margin”属性进行定位，就可以使他们显示在同一行。
3 把图片设置为背景图片。
31.禁止事件冒泡
event.stopPropagation()
32.禁止默认事件
event.preventDefault()
33.a 标签中 active hover link visited 正确的设置顺序是什么?
a 标签的 link、visited、hover、active 是有一定顺序的
a:link
a:visited
a:hover
a:active
34.a 标签中 如何禁用 href 跳转页面 或 定位链接
e.preventDefault();
href="javascript:void(0);
35.手机端上 图片长时间点击会选中图片，如何处理?
onselect=function() {
return false
}
36.video 标签的几个个属性方法
src：视频的 URL poster：视频封面，没有播放时显示的图片 preload：预加载 autoplay：自动播放 loop：循环播放 controls：浏览器自带的控制条 width：视频宽度 height：视频高度
37.常见的视频编码格式有几种?视频格式有几种?
视频格式：MPEG-1、MPEG-2 和 MPEG4 、AVI 、RM、ASF 和 WMV 格式
视频编码格式：H.264、MPEG-4、MPEG-2、WMA-HD 以及 VC-1
38.canvas 在标签上设置宽高 和在 style 中设置宽高有什么区别？

canvas 标签的 width 和 height 是画布实际宽度和高度，绘制的图形都是在这个上面。而 style 的 width 和 height 是 canvas 在浏览器中被渲染的高度和宽度。如果 canvas 的 width 和 height 没指定或值不正确，就被设置成默认值 。

39.border-image？box-sizing?

Border-image： 图形化边框

Box-sizing： 属性允许您以特定的方式定义匹配某个区域的特定元素。

语法 ：box-sizing：content-box | border-box

Content-box : padding 和 border 不被包含在定义的 width 和 height 之内。对象的实际宽度等于设置的 width 值和 border、padding 之和，即 （Element width = width + border + padding）此属性表现为标准模式下的盒模型 .

Border-box : padding 和 border 被包含在定义的 width 和 height 之内。对象的实际宽度就等于设置的 width 值，即使定义有 border 和 padding 也不会改变对象的实际宽度，即 （Element width = width ）此属性表现为怪异模式下的盒模型。

40.渐进增强和优雅降级

渐进增强 progressive enhancement：针对低版本浏览器进行构建页面，保证最基本的功能，然后再针对高级浏览器进行效果、交互等改进和追加功能达到更好的用户体验。

优雅降级 graceful degradation：一开始就构建完整的功能，然后再针对低版本浏览器进行兼容。

区别：优雅降级是从复杂的现状开始，并试图减少用户体验的供给，而渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要。降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带。

“优雅降级”观点

“优雅降级”观点认为应该针对那些最高级、最完善的浏览器来设计网站。而将那些被认为“过时”或有功能缺失的浏览器下的测试工作安排在开发周期的最后阶段，并把测试对象限定为主流浏览器（如 IE、Mozilla 等）的前一个版本。

在这种设计范例下，旧版的浏览器被认为仅能提供“简陋却无妨 (poor, but passable)” 的浏览体验。你可以做一些小的调整来适应某个特定的浏览器。但由于它们并非我们所关注的焦点，因此除了修复较大的错误之外，其它的差异将被直接忽略。

“渐进增强”观点

“渐进增强”观点则认为应关注于内容本身。

内容是我们建立网站的诱因。有的网站展示它，有的则收集它，有的寻求，有的操作，还有的网站甚至会包含以上的种种，但相同点是它们全都涉及到内容。这使得“渐进增强”成为一种更为合理的设计范例。这也是它立即被 Yahoo! 所采纳并用以构建其“分级式浏览器支持 (Graded Browser Support)”策略的原因所在。

那么问题来了。现在产品经理看到 IE6,7,8 网页效果相对高版本现代浏览器少了很多圆角，阴影（CSS3），要求兼容（使用图片背景，放弃 CSS3），你会如何说服他？

41.解释在 ie 低版本下的怪异盒模型 和 c3 的怪异盒模型 和 弹性盒模型?

IE 当 padding+border 的值小于 width 或者 height:

盒模型的宽度=margin(左右)+width（width 已经包含了 padding 和 border 的值）

盒模型的高度=margin(上下)+height（height 已经包含了 padding 和 border 的值）

当 padding+border 的值大于 width 或者 height:

盒模型的宽度=margin(左右)+padding(左右)+border(左右)

盒模型的高度=margin(上下)+padding(上下)+border(上下)+19px （加一个默认行高 19px）

所以相当于是 padding+border 和 width 或者 height 比大小，谁大取谁。

以上几种 DOCTYPE 都是标准的文档类型，无论使用哪种模式完整定义 DOCTYPE，都会触发标准模式，而如果 DOCTYPE 缺失则在 ie6，ie7，ie8 下将会触发怪异模式（quirks 模式）
CSS3box-sizing 有两个值一个是 content-box，另一个是 border-box。
当设置为 box-sizing:content-box 时，将采用标准模式解析计算，也是默认模式；
当设置为 box-sizing:border-box 时，将采用怪异模式解析计算；
Css3 弹性盒模型引入了新的盒子模型—弹性盒模型，该模型决定一个盒子在其他盒子中的分布方式以及如何处理可用的空间。使用该模型，可以很轻松的创建自适应浏览器窗口的流动布局或自适应字体大小的弹性布局。
42.animation 对应的属性
写法：animation: name duration timing-function delay iteration-count direction;
下面是对应的属性的介绍
animation-name 规定需要绑定到选择器的 keyframe 名称。。
animation-duration 规定完成动画所花费的时间，以秒或毫秒计。
animation-timing-function 规定动画的速度曲线。
animation-delay 规定在动画开始之前的延迟。
animation-iteration-count 规定动画应该播放的次数。
animation-direction 规定是否应该轮流反向播放动画。
43.transition?
css 的 transition 允许 css 的属性值在一定的时间区间内平滑地过渡。这种效果可以在鼠标单击、获得焦点、被点击或对元素任何改变中触发，并圆滑地以动画效果改变 CSS 的属性值
注意区别 transform，transform 是进行 2D 转换的 如移动，比例化，反过来，旋转，和拉伸元素。
44.h5 新特性？
1、绘画的 canvas 元素
2、用于媒介回放的 video 和 audio 元素
3、对本地离线存储的更好的支持
4、新的特殊内容元素，比如 article、footer、header、nav、section
5、新的表单控件，比如 calendar、date、time、email、url、search
45.canvas 如何绘制一个三角形|正方形
moveto 是移动到某个坐标， lineto 是从当前坐标连线到某个坐标。这两个函数加起来就是画一条直线。 画线要用“笔”，那么 MoveTo()把笔要画的起始位置固定了（x,y）然后要固定终止位置要用到 LineTo 函数确定终止位置（xend,yend）,这样一条线就画出来了。 每次与前面一个坐标相连 。
stroke() 方法会实际地绘制出通过 moveTo() 和 lineTo() 方法定义的路径。默认颜色是黑色。

```
<!DOCTYPE HTML5>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>画布</title>
  </head>
<body>
```

```
    <canvas id="myCanvas" width="200" height="100" style="border:1px solid #c3c3c3;">
    Your browser does not support the canvas element.
    </canvas>
    <script type="text/javascript">
        var c=document.getElementById("myCanvas");
//三角形
        var cxt=c.getContext("2d");
        cxt.moveTo(10,10);
        cxt.lineTo(50,50);
        cxt.lineTo(10,50);
        cxt.lineTo(10,10);
        cxt.stroke();
//正方形
        var cxt2=c.getContext("2d");
        cxt2.moveTo(60,10);
        cxt2.lineTo(100,10);
        cxt2.lineTo(100,50);
        cxt2.lineTo(60,50);
        cxt2.lineTo(60,10);
        cxt2.stroke();
    </script>
</body>
</html>
```

46、所用 bootstap 版本？

3.0

47、css 清除浮动的几种方式？

1、父级 div 定义 height

2、结尾处加空 div 标签 clear:both

3、父级 div 定义 伪类:after 和 zoom

4、父级 div 定义 overflow:hidden

5、父级 div 定义 overflow:auto

6、父级 div 也一起浮动

7、父级 div 定义 display:table

48、为什么要初始化 CSS 样式。

因为浏览器的兼容问题，不同浏览器对有些标签的默认值是不同的，如果没对 CSS 初始化往往会出现浏览器之间的页面显示差异。

49、CSS3 有哪些新特性？

CSS3 实现圆角（border-radius），阴影（box-shadow），

对文字加特效（text-shadow、），线性渐变（gradient），旋转（transform）

transform:rotate(9deg) scale(0.85,0.90) translate(0px,-30px) skew(-9deg,0deg);//旋转,缩放,定位,倾斜

增加了更多的 CSS 选择器 多背景 rgba

在 CSS3 中唯一引入的伪元素是::selection.

媒体查询，多栏布局　　border-image
50、解释下 CSS sprites，以及你要如何在页面或网站中使用它。
CSS Sprites 其实就是把网页中一些背景图片整合到一张图片文件中，再利用 CSS 的“background-image”，“background- repeat”，“background-position”的组合进行背景定位，background-position 可以用数字能精确的定位出背景图片的位置。这样可以减少很多图片请求的开销，因为请求耗时比较长；请求虽然可以并发，但是也有限制，一般浏览器都是 6 个。对于未来而言，就不需要这样做了，因为有了`http2`。
51、什么是 FOUC（无样式内容闪烁）？你如何来避免 FOUC？
FOUC - Flash Of Unstyled Content 文档样式闪烁
<style type="text/css" media="all">@import "../fouc.css";</style>
而引用 CSS 文件的@import 就是造成这个问题的罪魁祸首。IE 会先加载整个 HTML 文档的 DOM，然后再去导入外部的 CSS 文件，因此，在页面 DOM 加载完成到 CSS 导入完成中间会有一段时间页面上的内容是没有样式的，这段时间的长短跟网速，电脑速度都有关系。
解决方法：只要在<head>之间加入一个<link>或者<script>元素就可以了。

JS 部分
1.几种基本数据类型?复杂数据类型?值类型和引用数据类型?堆栈数据结构?
基本数据类型：Undefined、Null、Boolean、Number、String
值类型：数值、布尔值、null、undefined。
引用类型：对象、数组、函数。
堆栈数据结构：是一种支持后进先出(LIFO)的集合,即后被插入的数据,先被取出!
js 数组中提供了以下几个方法可以让我们很方便实现堆栈：
shift:从数组中把第一个元素删除，并返回这个元素的值。
unshift: 在数组的开头添加一个或更多元素，并返回新的长度
push:在数组的中末尾添加元素，并返回新的长度
pop:从数组中把最后一个元素删除，并返回这个元素的值。
2.声明函数作用提升?声明变量和声明函数的提升有什么区别?
（1）变量声明提升：变量申明在进入执行上下文就完成了。
　　只要变量在代码中进行了声明，无论它在哪个位置上进行声明， js 引擎都会将它的声明放在范围作用域的顶部；
（2）函数声明提升：执行代码之前会先读取函数声明，意味着可以把函数申明放在调用它的语句后面。
　只要函数在代码中进行了声明，无论它在哪个位置上进行声明， js 引擎都会将它的声明放在范围作用域的顶部；
（3）变量 or 函数声明：函数声明会覆盖变量声明，但不会覆盖变量赋值。
　　同一个名称标识 a，即有变量声明 var a，又有函数声明 function a() {}，不管二者声明的顺序，函数声明会覆盖变量声明，也就是说，此时 a 的值是声明的函数 function a() {}。
注意：如果在变量声明的同时初始化 a，或是之后对 a 进行赋值，此时 a 的值变量的值。eg: var a; var c = 1; a = 1; function a() { return true; } console.log(a);
3.判断数据类型?
typeof 返回的类型都是字符串形式，可以判断 function 的类型；在判断除 Object 类型的对象时比较方便。
判断已知对象类型的方法： instanceof，后面一定要是对象类型，并且大小写不能错，该方法适合一些条件选择或分支。

4.异步编程？

方法 1：回调函数，优点是简单、容易理解和部署，缺点是不利于代码的阅读和维护，各个部分之间高度耦合（Coupling），流程会很混乱，而且每个任务只能指定一个回调函数。

方法 2：时间监听，可以绑定多个事件，每个事件可以指定多个回调函数，而且可以“去耦合”（Decoupling），有利于实现模块化。缺点是整个程序都要变成事件驱动型，运行流程会变得很不清晰。

方法 3：发布/订阅，性质与“事件监听”类似，但是明显优于后者。

方法 4：Promises 对象，是 CommonJS 工作组提出的一种规范，目的是为异步编程提供统一接口。

简单说，它的思想是，每一个异步任务返回一个 Promise 对象，该对象有一个 then 方法，允许指定回调函数。

5.事件流?事件捕获？事件冒泡？

事件流：从页面中接收事件的顺序。也就是说当一个事件产生时，这个事件的传播过程，就是事件流。

IE 中的事件流叫事件冒泡；事件冒泡：事件开始时由最具体的元素接收，然后逐级向上传播到较为不具体的节点（文档）。对于 html 来说，就是当一个元素产生了一个事件，它会把这个事件传递给它的父元素，父元素接收到了之后，还要继续传递给它的上一级元素，就这样一直传播到 document 对象（亲测现在的浏览器到 window 对象，只有 IE8 及下不这样）；事件捕获是不太具体的元素应该更早接受到事件，而最具体的节点应该最后接收到事件。他们的用意是在事件到达目标之前就捕获它；也就是跟冒泡的过程正好相反，以 html 的 click 事件为例，document 对象（DOM 级规范要求从 document 开始传播，但是现在的浏览器是从 window 对象开始的）最先接收到 click 事件的然后事件沿着 DOM 树依次向下传播，一直传播到事件的实际目标；

6.如何清除一个定时器?

window.clearInterval();

window.clearTimeout();

7.如何添加一个 dom 对象到 body 中?innerHTML 和 innerText 区别?

body.appendChild(dom 元素);

innerHTML:从对象的起始位置到终止位置的全部内容,包括 Html 标签。

innerText:从起始位置到终止位置的内容, 但它去除 Html 标签

分别简述五个 window 对象、属性

成员对象

window.event　window.document　window.history

window.screen　window.navigator　window.external

Window 对象的属性如下：

window //窗户自身

window.self //引用本窗户 window=window.self

window.name //为窗户命名

window.defaultStatus //设定窗户状态栏信息　window.location //URL 地址,配备布置这个属性可以打开新的页面

8.数据持久化技术(ajax)?简述 ajax 流程

1)客户端产生 js 的事件

2)创建 XMLHttpRequest 对象

3)对 XMLHttpRequest 进行配置

4)通过 AJAX 引擎发送异步请求
5)服务器端接收请求并且处理请求，返回 html 或者 xml 内容
6)XML 调用一个 callback()处理响应回来的内容
7)页面局部刷新
9.回调函数?

回调函数就是一个通过函数指针调用的函数。如果你把函数的指针（地址）作为参数传递给另一个函数，当这个指针被用来调用其所指向的函数时，我们就说这是回调函数。回调函数不是由该函数的实现方直接调用，而是在特定的事件或条件发生时由另外的一方调用的，用于对该事件或条件进行响应。

10.什么是闭包?* 堆栈溢出有什么区别？ 内存泄漏? 那些操作会造成内存泄漏？怎么样防止内存泄漏?
闭包：就是能够读取其他函数内部变量的函数。
堆栈溢出：就是不顾堆栈中分配的局部数据块大小，向该数据块写入了过多的数据，导致数据越界，结果覆盖了别的数据。经常会在递归中发生。
内存泄露是指：用动态存储分配函数内存空间，在使用完毕后未释放，导致一直占据该内存单元。直到程序结束。指任何对象在您不再拥有或需要它之后仍然存在。
造成内存泄漏：
setTimeout 的第一个参数使用字符串而非函数的话，会引发内存泄漏。
闭包、控制台日志、循环（在两个对象彼此引用且彼此保留时，就会产生一个循环）
防止内存泄露：
1、不要动态绑定事件；
2、不要在动态添加，或者会被动态移除的 dom 上绑事件，用事件冒泡在父容器监听事件；
3、如果要违反上面的原则，必须提供 destroy 方法，保证移除 dom 后事件也被移除，这点可以参考 Backbone 的源代码，做的比较好；
4、单例化，少创建 dom，少绑事件。
11.平时工作中怎么样进行数据交互?如果后台没有提供数据怎么样进行开发?mock 数据与后台返回的格式不同意怎么办?
由后台编写接口文档、提供数据接口实、前台通过 ajax 访问实现数据交互；
在没有数据的情况下寻找后台提供静态数据或者自己定义 mock 数据；
返回数据不统一时编写映射文件 对数据进行映射。
12 简述 ajax 执行流程
基本步骤：

```
var xhr =null;//创建对象
if(window.XMLHttpRequest){
        xhr = new XMLHttpRequest();
}else{
        xhr = new ActiveXObject("Microsoft.XMLHTTP");
}
xhr.open(“方式”,”地址”,”标志位”);//初始化请求
    xhr.setRequestHeader(“”,””);//设置 http 头信息
xhr.onreadystatechange =function(){}//指定回调函数
xhr.send();//发送请求
```

13.自执行函数?用于什么场景？好处
自执行函数:1、声明一个匿名函数 2、马上调用这个匿名函数。

作用：创建一个独立的作用域。
好处：防止变量弥散到全局，以免各种 js 库冲突。隔离作用域避免污染，或者截断作用域链，避免闭包造成引用变量无法释放。利用立即执行特性，返回需要的业务函数或对象，避免每次通过条件判断来处理
场景：一般用于框架、插件等场景
14.html 和 xhtml 有什么区别?
HTML 是一种基本的 WEB 网页设计语言，XHTML 是一个基于 XML 的标记语言。
1.XHTML 元素必须被正确地嵌套。2.XHTML 元素必须被关闭。3.标签名必须用小写字母。4.空标签也必须被关闭。5.XHTML 文档必须拥有根元素。
15. 什么是构造函数？与普通函数有什么区别?
构造函数：是一种特殊的方法、主要用来创建对象时初始化对象，总与 new 运算符一起使用，创建对象的语句中构造函数的函数名必须与类名完全相同。
与普通函数相比只能由 new 关键字调用，构造函数是类的标示
16. 通过 new 创建一个对象的时候，函数内部有哪些改变
```
function Person(){}
Person.prototype.friend = [];
Person.prototype.name = '';
// var a = new Person();
// a.friend[0] = '王琦';
// a.name = '程娇';
// var b = new Person();
// b.friend？
// b.name?
```
1、创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。
2、属性和方法被加入到 this 引用的对象中。
3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。
17.事件委托？有什么好处?
答：（1）利用冒泡的原理，把事件加到父级上，触发执行效果
（2）好处：新添加的元素还会有之前的事件；提高性能。
18.window.onload ==? DOMContentLoaded ?
答：一般情况下，DOMContentLoaded 事件要在 window.onload 之前执行，当 DOM 树构建完成的时候就会执行 DOMContentLoaded 事件，而 window.onload 是在页面载入完成的时候，才执行，这其中包括图片等元素。大多数时候我们只是想在 DOM 树构建完成后，绑定事件到元素，我们并不需要图片元素，加上有时候加载外域图片的速度非常缓慢。
19.节点类型?判断当前节点类型?
答：1 元素节点
2 属性节点
3 文本节点
8 注释节点
9 文档节点
通过 nodeObject.nodeType 判断节点类型：其中，nodeObject 为 DOM 节点（节点对象）。该属性返回以数字表示的节点类型，例如，元素节点返回 1，属性节点返回 2 。
20.如何合并两个数组？数组删除一个元素?
答：三种方法。

（1）var arr1=[1,2,3];
var arr2=[4,5,6];
arr1 = arr1.concat(arr2);
console.log(arr1);

（2）var arr1=[1,2,3];
var arr2=[4,5,6];
Array.prototype.push.apply(arr1,arr2);
console.log(arr1);

（3）var arr1=[1,2,3];
var arr2=[4,5,6];
for (var i=0; i < arr2.length; i++) {
arr1.push(arr2[i]);
}
console.log(arr1);

21.强制转换 显式转换 隐式转换?

答：(1)强制类型转换：

Boolean(0) // => false - 零
Boolean(new object()) // => true - 对象
Number(undefined) // => NaN
Number(null) // => 0
String(null) // => "null"
parseInt( )
parseFloat( )
JSON.parse( )
JSON.stringify ( )

(2)隐式类型转换：

在使用算术运算符时，运算符两边的数据类型可以是任意的，比如，一个字符串可以和数字相加。之所以不同的数据类型之间可以做运算，是因为 JavaScript 引擎在运算之前会悄悄的把他们进行了隐式类型转换的

（例如：x+"" //等价于 String(x)
+x //等价于 Number(x)
x-0 //同上
!!x //等价于 Boolean(x),是双叹号）

(3)显式转换：

如果程序要求一定要将某一类型的数据转换为另一种类型，则可以利用强制类型转换运算符进行转换，这种强制转换过程称为显示转换。

显示转换是你定义让这个值类型转换成你要用的值类型，是底到高的转换。例 int 到 float 就可以直接转，

int i=5,想把他转换成 char 类型，就用显式转换（char）i

22. var a = true;
typeof a++ //number
typeof true++ //错误！
typeof a+1 //number1
typeof (true)+1 //boolean1

23.jq 中如何实现多库并存?Noconfict
多库共存就是“$ ”符号的冲突。
方法一：
利用 jQuery 的实用函数$.noConflict();这个函数归还$的名称控制权给另一个库，因此可以在页面上使用其他库。这时，我们可以用"jQuery "这个名称调用 jQuery 的功能。

```
$.noConflict();
jQuery('#id').hide();
.....
//或者给 jQuery 一个别名
var $j=jQuery
$j('#id').hide();
.....
```

方法二：

```
(function($){

})(jQuery)
```

方法三：

```
jQuery(function($){

})
```

通过传递一个函数作为 jQuery 的参数，因此把这个函数声明为就绪函数。
我们声明$为就绪函数的参数，因为 jQuery 总是吧 jQuery 对象的引用作为第一个参数传递，所以就保证了函数的执行。

24.jq 中 get 和 eq 有什么区别？
get() :取得其中一个匹配的元素。num 表示取得第几个匹配的元素，get 多针对集合元素，返回的是 DOM 对象组成的数组
eq():获取第 N 个元素，下标都是从 0 开始，返回的是一个 JQuery 对象
25.如何通过原生 js 判断一个元素当前是显示还是隐藏状态?

```
if( document.getElementById("div").css("display")==='none')
if( document.getElementById("div").css("display")==='block')
$("#div").is(":hidden"); // 判断是否隐藏
$("#div").is(":visible")
```

26.jq 如何判断元素显示隐藏？
第一种：使用 CSS 属性

```
var display =$('#id').css('display');
if(display == 'none'){
   alert("我是隐藏的！ ");
}
```

第二种：使用 jquery 内置选择器

```
<div id="test">
<p>仅仅是测试所用</p>
```

```
</div>

if($("#test").is(":hidden")){
        $("#test").show();     //如果元素为隐藏,则将它显现
}else{
      $("#test").hide();     //如果元素为显现,则将其隐藏
}
```

第三种：jQuery 判断元素是否显示 是否隐藏

```
var node=$('#id');
if(node.is(':hidden')){      //如果 node 是隐藏的则显示 node 元素，否则隐藏
      node.show();
}else{
      node.hide();
}
```

27.移动端上什么是点击穿透?

点击穿透现象有 3 种：

点击穿透问题：点击蒙层（mask）上的关闭按钮，蒙层消失后发现触发了按钮下面元素的 click 事件

跨页面点击穿透问题：如果按钮下面恰好是一个有 href 属性的 a 标签，那么页面就会发生跳转

另一种跨页面点击穿透问题：这次没有 mask 了，直接点击页内按钮跳转至新页，然后发现新页面中对应位置元素的 click 事件被触发了

解决方案：

1、只用 touch

最简单的解决方案，完美解决点击穿透问题

把页面内所有 click 全部换成 touch 事件（ touchstart 、'touchend'、'tap'）

2、只用 click

下下策，因为会带来 300ms 延迟，页面内任何一个自定义交互都将增加 300 毫秒延迟

3、tap 后延迟 350ms 再隐藏 mask

改动最小，缺点是隐藏 mask 变慢了，350ms 还是能感觉到慢的

4、pointer-events

比较麻烦且有缺陷， 不建议使用

mask 隐藏后，给按钮下面元素添上 pointer-events: none; 样式，让 click 穿过去，350ms 后去掉这个样式，恢复响应

缺陷是 mask 消失后的的 350ms 内，用户可以看到按钮下面的元素点着没反应，如果用户手速很快的话一定会发现

28.jq 绑定事件的几种方式？on bind

jQuery 中提供了四种事件监听方式，分别是 bind、live、delegate、on，对应的解除监听的函数分别是 unbind、die、undelegate、off

Bind( )是使用频率较高的一种，作用就是在选择到的元素上绑定特定事件类型的监听函数;

Live( )可以对后生成的元素也可以绑定相应的事件,处理机制就是把事件绑定在 DOM 树的根节点上，而不是直接绑定在某个元素上;

Delegate( )采用了事件委托的概念，不是直接为子元素绑定事件，而是为其父元素（或祖先

元素也可）绑定事件，当在 div 内任意元素上点击时，事件会一层层从 event target 向上冒泡，直至到达你为其绑定事件的元素；
on( )方法可以绑定动态添加到页面元素的事件，on()方法绑定事件可以提升效率；
29.jq 中如何将一个 jq 对象转化为 dom 对象？
方法一：
jQuery 对象是一个数据对象，可以通过[index]的方法，来得到相应的 DOM 对象。
如：var $v =$("#v") ; //jQuery 对象
var v=$v[0]; //DOM 对象
alert(v.checked) //检测这个 checkbox 是否被选中
方法二：
jQuery 本身提供，通过.get(index)方法，得到相应的 DOM 对象
如：var $v=$("#v"); //jQuery 对象
var v=$v.get(0); //DOM 对象
alert(v.checked) //检测这个 checkbox 是否被选中
30.jq 中有几种选择器?分别是什么?
层叠选择器、基本过滤选择器、内容过滤选择器、可视化过滤选择器、属性过滤选择器、子元素过滤选择器、表单元素选择器、表单元素过滤选择器
31.jq 中怎么样编写插件?
第一种是类级别的插件开发：

1.1 添加一个新的全局函数
添加一个全局函数，我们只需如下定义：

```
jQuery.foo = function() {
  alert('This is a test. This is only a test.');
};
```

1.2 增加多个全局函数
添加多个全局函数，可采用如下定义：

```
jQuery.foo = function() {
     alert('This is a test. This is only a test.');
};
jQuery.bar = function(param) {
   alert('This function takes a parameter, which is "' + param + '".');
};
```

调用时和一个函数的一样的:jQuery.foo();jQuery.bar();或者$.foo();$.bar('bar');

1.3 使用 jQuery.extend(object);

```
jQuery.extend({
foo: function() {
     alert('This is a test. This is only a test.');
  },
bar: function(param) {
     alert('This function takes a parameter, which is "' + param +'".');
  }
```

```
});
```

1.4 使用命名空间

虽然在 jQuery 命名空间中，我们禁止使用了大量的 javaScript 函数名和变量名。但是仍然不可避免某些函数或变量名将于其他 jQuery 插件冲突，因此我们习惯将一些方法封装到另一个自定义的命名空间。

```
jQuery.myPlugin = {
foo:function() {
   alert('This is a test. This is only a test.');
 },
 bar:function(param) {
   alert('This function takes a parameter, which is "' + param + '".');
 }
};
```

采用命名空间的函数仍然是全局函数，调用时采用的方法：

```
$.myPlugin.foo();
$.myPlugin.bar('baz');
```

通过这个技巧（使用独立的插件名），我们可以避免命名空间内函数的冲突。

第二种是对象级别的插件开发

形式 1：

```
(function($){
   $.fn.extend({
     pluginName:function(opt,callback){
               // Our plugin implementation code goes here.
     }
   })
})(jQuery);
```

形式 2：

```
(function($) {
     $.fn.pluginName = function() {
          // Our plugin implementation code goes here.
     };
})(jQuery);
```

形参是$，函数定义完成之后,把 jQuery 这个实参传递进去.立即调用执行。这样的好处是,我们在写 jQuery 插件时,也可以使用$这个别名,而不会与 prototype 引起冲突

32.$('div+.ab')和$('.ab+div') 哪个效率高？

$('div+.ab')效率高

33.$.map 和$.each 有什么区别

map()方法主要用来遍历操作数组和对象，会返回一个新的数组。$.map()方法适用于将数组或对象每个项目新阵列映射到一个新数组的函数；

each()主要用于遍历 jquery 对象，返回的是原来的数组，并不会新创建一个数组。

34.编写一个 getElementsByClassName 封装函数?

```
<body>
  <input type="submit" id = "sub" class="ss confirm btn" value="提交"/>
  <script>
window.onload = function(){
//方法一
        var Opt = document.getElementById('sub');
        var getClass = function(className,tagName){
            if(document.getElementsByTagName){
                var Inp = document.getElementsByTagName(tagName);
                for(var i=0; i<Inp.length; i++){
                    if((new RegExp('(\\s|^)' +className +'(\\s|$)')).test(Inp[i].className)){
                        return Inp[i];
                    }
                }
            }else if(document.getElementsByClassName){
                return document.getElementsByClassName(className);
            }
        }
//方法二
        var aa = getClass("confirm", "input");
        function getClass(className, targetName){
            var ele = [];
            var all = document.getElementsByTagName(targetName || "*");
            for(var i=0; i<all.length; i++){
                if(all[i].className.match(new RegExp('(\\s|^)'+confirm+'(\\s|$)'))){
                    ele[ele.length] = all[i];
                }
            }
            return ele;
        }

//方法三
        function getObjsByClass(tagName, className){
            if(document.getElementsByClassName){
                alert("document.getElementsByClassName");
                return document.getElementsByClassName(className);
            }else{
                var el = [];
                var _el = document.getElementsByTagName(tagName);
                for(var i=0; i<_el.length; i++){
                    if(_el[i].className.indexOf(className) > -1){
                        alert(_el[i]);
                        el[_el.length] = _el[i];
```

```
                }
            }
            alert(el);
            return el;
        }
    }
  }
 </script>
</body>
```

35.简述下工作流程

我在之前的公司工作流程大概是这样的：公司定稿会结束以后，会进行简单的技术研讨，然后我们前端会进行先期的技术准备。前端切图人员会进行 psd 设计稿切图，并且将 css 文件进行整合。我们主要编写 JS 部分，其中包括搭建前端框架（大项目），编写 js 业务和数据持久化操作，我们也会编写 js 插件并且进行封装方便使用，还有就是编写 JS 前端组建和 JS 测试单元，最后将完成的 JS 部分与切图人员提供的 HTML 页面进行整合。最后对完成的页面进行功能测试、页面兼容、产品还原。然后对产品进行封存，提交测试。如果出现 BUG 会返回给我们开发人员进行修改，再提交测试，最后测试成功，进行版本封存。等到程序全部上线的时候进行线上测试。

36.一般使用什么版本控制工具?svn 如何对文件加锁

svn 加锁目的：为了避免多个人同一时间对同一个文件改动的相互覆盖，版本控制系统就必须有一套冲突处理机制。

svn 加锁两种策略：乐观加锁：所有签出的文件都是可读写的，对文件的修改不必获得文件的锁，当你修改完文件签入时，会首先要求你更新本地文件，版本控制系统不会覆盖你的本地修改，而是会让你自己合并冲突后签入。

严格加锁：所有签出的文件都是只读的，任何对文件的修改必须要获得文件的锁，如果其他人没有拥有该文件的锁，那么版本控制系统就会授权给你文件的锁，并将文件设置为可编辑的。

svn 两种加锁步骤：乐观加锁：选择你想要获取锁定的文件，然后右键菜单点击 TortoiseSVN 选取获取锁定。

严格加锁：在想要采取严格加锁的文件或目录上点击右键，使用 TortoiseSVN 属性菜单，点击新建属性，选择需要锁定。

37. git 和 svn 的区别?

SVN 是集中式版本控制系统，版本库是集中放在中央服务器的，而干活的时候，用的都是自己的电脑，所以首先要从中央服务器哪里得到最新的版本，然后干活，干完后，需要把自己做完的活推送到中央服务器。集中式版本控制系统是必须联网才能工作，如果在局域网还可以，带宽够大，速度够快，如果在互联网下，如果网速慢的话，就纳闷了。

Git 是分布式版本控制系统，那么它就没有中央服务器的，每个人的电脑就是一个完整的版本库，这样，工作的时候就不需要联网了，因为版本都是在自己的电脑上。既然每个人的电脑都有一个完整的版本库，那多个人如何协作呢？比如说自己在电脑上改了文件 A，其他人也在电脑上改了文件 A，这时，你们两之间只需把各自的修改推送给对方，就可以互相看到对方的修改了。

38. jquery 和 zepto 有什么区别?

1.针对移动端程序，Zepto 有一些基本的触摸事件可以用来做触摸屏交互（tap 事件、swipe

事件），Zepto 是不支持 IE 浏览器的，这不是 Zepto 的开发者 Thomas Fucks 在跨浏览器问题上犯了迷糊，而是经过了认真考虑后为了降低文件尺寸而做出的决定，就像 jQuery 的团队在 2.0 版中不再支持旧版的 IE（6 7 8）一样。因为 Zepto 使用 jQuery 句法，所以它在文档中建议把 jQuery 作为 IE 上的后备库。那样程序仍能在 IE 中，而其他浏览器则能享受到 Zepto 在文件大小上的优势，然而它们两个的 API 不是完全兼容的，所以使用这种方法时一定要小心，并要做充分的测试。
2.Dom 操作的区别：添加 id 时 jQuery 不会生效而 Zepto 会生效。
3.zepto 主要用在移动设备上，只支持较新的浏览器，好处是代码量比较小，性能也较好。jquery 主要是兼容性好，可以跑在各种 pc，移动上，好处是兼容各种浏览器，缺点是代码量大，同时考虑兼容，性能也不够好。
39. $(function(){})和 window.onload 和 $(document).ready(function(){})
window.onload:用于当页面的所有元素，包括外部引用文件，图片等都加载完毕时运行函数内的函数。load 方法只能执行一次，如果在 js 文件里写了多个，只能执行最后一个。
$(document).ready(function(){})和$(function(){})都是用于当页面的标准 DOM 元素被解析成 DOM 树后就执行内部函数。这个函数是可以在 js 文件里多次编写的，对于多人共同编写的 js 就有很大的优势，因为所有行为函数都会执行到。而且$(document).ready()函数在 HMTL 结构加载完后就可以执行，不需要等大型文件加载或者不存在的连接等耗时工作完成才执行，效率高。
40. jq 中 attr 和 prop 有什么区别
对于 HTML 元素本身就带有的固有属性，在处理时，使用 prop 方法。
对于 HTML 元素我们自己自定义的 DOM 属性，在处理时，使用 attr 方法。
41. 简述下 this 和 定义属性和方法的时候有什么区别?Prototype？
this 表示当前对象，如果在全局作用范围内使用 this，则指代当前页面对象 window； 如果在函数中使用 this，则 this 指代什么是根据运行时此函数在什么对象上被调用。 我们还可以使用 apply 和 call 两个全局方法来改变函数中 this 的具体指向。
prototype 本质上还是一个 JavaScript 对象。并且每个函数都有一个默认的 prototype 属性。在 prototype 上定义的属性方法为所有实例共享，所有实例皆引用到同一个对象，单一实例对原型上的属性进行修改，也会影响到所有其他实例。
42. 什么是预编译语音|预编译处理器?
Sass 是一种 CSS 预处理器语言，通过编程方式生成 CSS 代码。因为可编程，所以操控灵活性自由度高，方便实现一些直接编写 CSS 代码较困难的代码。
同时，因为 Sass 是生成 CSS 的语言，所以写出来的 Sass 文件是不能直接用的，必须经过编译器编译成 CSS 文件才能使用。
CSS 预处理器是一种语言用来为 CSS 增加一些编程的的特性，无需考虑浏览器的兼容性问题，例如你可以在 CSS 中使用变量、简单的程序逻辑、函数等等在编程语言中的一些基本技巧，可以让你的 CSS 更见简洁，适应性更强，代码更直观等诸多好处。最常用的 css 预处理器有 sass、less css、 stylus。
43.ajax 和 jsonp ？
ajax 和 jsonp 的区别：
相同点：都是请求一个 url
不同点：ajax 的核心是通过 xmlHttpRequest 获取内容
jsonp 的核心则是动态添加<script>标签来调用服务器 提供的 js 脚本。
44.ajax 执行流程？
1. 创建 XMLHttpRequest 对象,也就是创建一个异步调用对象

2. 创建一个新的 HTTP 请求,并指定该 HTTP 请求的方法、URL 及验证信息
3. 设置响应 HTTP 请求状态变化的函数
4. 发送 HTTP 请求
5. 获取异步调用返回的数据
6. 使用 JavaScript 和 DOM 实现局部刷新

45.xhr 对象 status ? readystate?

status 是 XMLHttpRequest 对象的一个属性，表示响应的 HTTP 状态码。

readyState 是 XMLHttpRequest 对象的一个属性，用来标识当前 XMLHttpRequest 对象处于什么状态。

46.readystate 0~4

0：未初始化状态：此时，已经创建了一个 XMLHttpRequest 对象

1： 准备发送状态：此时，已经调用了 XMLHttpRequest 对象的 open 方法，并且 XMLHttpRequest 对象已经准备好将一个请求发送到服务器端

2：已经发送状态：此时，已经通过 send 方法把一个请求发送到服务器端，但是还没有收到一个响应

3：正在接收状态：此时，已经接收到 HTTP 响应头部信息，但是消息体部分还没有完全接收到

4：完成响应状态：此时，已经完成了 HTTP 响应的接收

47.说出几个 http 协议状态码? 200 201 302 304 400 404 500

200：请求成功

201：请求成功并且服务器创建了新的资源

302：服务器目前从不同位置的网页响应请求，但请求者应继续使用原有位置来响应以后的请求。

304：自从上次请求后，请求的网页未修改过。服务器返回此响应时，不会返回网页内容。

400：服务器不理解请求的语法。

404：请求的资源（网页等）不存在

500： 内部服务器错误

48.上一个项目是什么？主要负责哪些？购物车流程?支付功能?

主要负责哪些就讲主要做哪些功能模块：

1）商品模块:

1、商品列表：商品排序 商品筛选 商品过滤 商品查询 商品推荐

2、商品详情:类型推荐 商品简介 商品详情 商品评价 售后维护

2)购物车模块：商品编号、数量、价格、总额、运费、运输选项、运费总计、从购物车删除选项、更新数量、结账、继续购物、商品描述、库存信息

49.sessionStorage 和 localstroage 与 cookie 之间有什么关联和区别，cookie 最大存放多少字节

三者共同点：都是保存在浏览器端，且同源的。

区别:

1、cookie 在浏览器和服务器间来回传递。而 sessionStorage 和 localStorage 不会自动把数据发给服务器，仅在本地保存

2、存储大小限制也不同，cookie 数据不能超过 4k，sessionStorage 和 localStorage 但比 cookie 大得多，可以达到 5M

3、数据有效期不同，sessionStorage：仅在当前浏览器窗口关闭前有效，自然也就不可能持久保持；localStorage：始终有效，窗口或浏览器关闭也一直保存，因此用作持久数

据；cookie 只在设置的 cookie 过期时间之前一直有效，即使窗口或浏览器关闭

4、作用域不同，sessionStorage 不在不同的浏览器窗口中共享，即使是同一个页面(即数据不共享)；localStorage 在所有同源窗口中都是共享的；cookie 也是在所有同源窗口中都是共享的（即数据共享）。

50.ajax 中 get 和 post 有什么区别?

get 和 post 都是数据提交的方式。

get 的数据是通过网址问号后边拼接的字符串进行传递的。post 是通过一个 HTTP 包体进行传递数据的。

get 的传输量是有限制的，post 是没有限制的。

get 的安全性可能没有 post 高，所以我们一般用 get 来获取数据，post 一般用来修改数据。

51.Gc 机制是什么？为什么闭包不会被回收变量和函数？

1、Gc 垃圾回收机制;

2、外部变量没释放，所以指向的大函数内的小函数也释放不了

52.简述下你理解的面向对象？

万物皆对象，把一个对象抽象成类,具体上就是把一个对象的静态特征和动态特征抽象成属性和方法,也就是把一类事物的算法和数据结构封装在一个类之中,程序就是多个对象和互相之间的通信组成的.

面向对象具有封装性,继承性,多态性。

封装:隐蔽了对象内部不需要暴露的细节,使得内部细节的变动跟外界脱离,只依靠接口进行通信.封装性降低了编程的复杂性. 通过继承,使得新建一个类变得容易,一个类从派生类那里获得其非私有的方法和公用属性的繁琐工作交给了编译器. 而 继承和实现接口和运行时的类型绑定机制 所产生的多态,使得不同的类所产生的对象能够对相同的消息作出不同的反应,极大地提高了代码的通用性.

总之,面向对象的特性提高了大型程序的重用性和可维护性.

53.this 是什么 在不同场景中分别代表什么

（1）function a(){ this ?}

This:指向 windows

（2） function b(){ return function(){ this ?}}

b()(); This:指向 windows

（3） function c(){ return {s:function(){this}}}

c().s(); This:指向 object

由于其运行期绑定的特性，JavaScript 中的 this 含义要丰富得多，它可以是全局对象、当前对象或者任意对象，这完全取决于函数的调用方式。

54.你对数据校验是怎么样处理的？jquery.validate？

通俗的说，就是为保证数据的完整性，用一种指定的算法对原始数据计算出的一个校验值。接收方用同样的算法计算一次校验值，如果和随数据提供的校验值一样，就说明数据是完整的。

用正则表达式来处理;

jquery.validate：为表单验证插件

55.如何对登录的账号密码进行加密?

Md5

56.在 jq 中 mouseover mouseenter mouseout mouseleave 和 hover 有什么关联?

mouseenter 与 mouseover：

不论鼠标指针穿过被选中元素或其子元素，都会触发 mouseover 事件。

只有在鼠标指针穿过被选元素时，才会触发 mouseentr 事件。

mouseout 与 mouseleave：

不论鼠标离开被选元素还是任何子元素，都会触发 mouseout 事件。

只有在鼠标指针离开被选元素时，才会触发 mouseleave 事件。

hover:

hover 是一个符合方法，相当于 mouseenter+mouseleave。

57.jsonp 原理？ 缺点?

工作原理：使用 script 标签实现跨域访问，可在 url 中指定回调函数，获取 JSON 数据并在指定的回调函数中执行 jquery 实现 jsop。

缺点：只支持 GET 方式的 jsonp 实现，是一种脚本注入行为存在一定的安全隐患。如果返回的数据格式有问题或者返回失败了，并不会报错。

58.除了 jsonp 还有什么跨域方式

javascript 跨域有两种情况：

1、基于同一父域的子域之间，如：a.c.com 和 b.c.com

2、基于不同的父域之间，如：www.a.com 和 www.b.com

3、端口的不同，如：www.a.com:8080 和 www.a.com:8088

4、协议不同，如：http://www.a.com 和 https://www.a.com

对于情况 3 和 4，需要通过后台 proxy 来解决，具体方式如下：

a、在发起方的域下创建 proxy 程序

b、发起方的 js 调用本域下的 proxy 程序

c、proxy 将请求发送给接收方并获取相应数据

d、proxy 将获得的数据返回给发起方的 js

代码和 ajax 调用一致，其实这种方式就是通过 ajax 进行调用的

而情况 1 和 2 除了通过后台 proxy 这种方式外，还可以有多种办法来解决：

1、document.domain+iframe（只能解决情况 1）：

a、在发起方页面和接收方页面设置 document.domain，并将值设为父域的主域名(window.location.hostname)

b、在发起方页面创建一个隐藏的 iframe，iframe 的源是接收方页面

c、根据浏览器的不同，通过 iframe.contentDocument || iframe.contentWindow.document 来获得接收方页面的内容

d、通过获得的接收方页面的内容来与接收方进行交互

这种方法有个缺点，就是当一个域被攻击时，另一个域会有安全漏洞出现。

59.如何使用 storage 对 js 文件进行缓存

由于 sessionStorage - 针对一个 session 的数据存储，所以我们一般利用 localStorage 储存 js 文件，只有在第一次访问该页面的时候加载 js 文件，以后在访问的时候加载本地 localStorage 执行

60.如何确保 ajax 或连接不走缓存路径

在 Ajax 中使用 Get 请求数据不会有页面缓存的问题，而使用 POST 请求可是有时候页面会缓存我们提交的信息，导致我们发送的异步请求不能正确的返回我们想要的数据

$.post(url,data ,ranNum:Math.random()} ,function(data){}）

ranNum : 这个是防止缓存的核心，每次发起请求都会用 Math.random()方法生成一个随机的数字，这样子就会刷新 url 缓存

61.split() join()?

前者是切割成数组的形式，

后者是将数组转换成字符串
62.slice() splice()?
slice() 方法可从已有的数组中返回选定的元素。
splice() 方法向/从数组中添加/删除项目，然后返回被删除的项目。
63.typeof？typeof [ ]返回数据类型是？
判断基本数据类型；var a=[];typeof a 输出 object；
本来判断一个对象类型用 typeof 是最好的，不过对于 Array 类型是不适用的，可以使用 instanceof 操作符：

```
var arrayStr=new Array("1","2","3","4","5");
alert(arrayStr instanceof Array);
```

当然以上在一个简单的页面布局里面是没有问题的，如果是复杂页面情况，入获取的是 frame 内部的 Array 对象，可以用这个函数判断：

```
function isArray(obj) {
    return Object.prototype.toString.call(obj) === '[object Array]';
}
```

64.disabled readyonly?
readonly 只针对 input(text / password)和 textarea 有效，
而 disabled 对于所有的表单元素都有效，当表单元素在使用了 disabled 后，当我们将表单以 POST 或 GET 的方式提交的话，这个元素的值不会被传递出去，而 readonly 会将该值传递出去。
65.同步异步?
1、进程同步：就是在发出一个功能调用时，在没有得到结果之前，该调用就不返回。也就是必须一件一件事做,等前一件做完了才能做下一件事
2、异步的概念和同步相对。当一个异步过程调用发出后，调用者不能立刻得到结果。实际处理这个调用的部件在完成后，通过状态、通知和回调来通知调用者。
66.promise
Promise 的构造函数接收一个参数，是函数，并且传入两个参数：resolve，reject，分别表示异步操作执行成功后的回调函数和异步操作执行失败后的回调函数。
67.函数 fn1 函数 fn2 函数 fn3，如果想在三个函数都执行完成后执行某一个事件应该如何实现?
1、设置事件监听。
2、回调函数：

```
function fn1(){
    console.log("执行 fn1");
    fn2();
}
function fn2(){
    console.log("执行 fn2");
    fn3();
}
function fn3(){
    console.log("执行 fn3");
    mou();
}
```

```
function mou(){
        console.log("执行某个函数");
}
fn1();
```

68.JavaScript 提供了哪几种“异步模式 ”?

1、回调函数（callbacks）

2、事件监听

3、Promise 对象

69.什么是移动端的 300ms 延迟？什么是点击穿透？解决方案?

移动端 300ms 延迟：假定这么一个场景。用户在 浏览器里边点击了一个链接。由于用户可以进行双击缩放或者双击滚动的操作，当用户一次点击屏幕之后，浏览器并不能立刻判断用户是确实要打开这个链接，还是想要进行双击操作。因此，浏览器 就等待 300 毫秒，以判断用户是否再次点击了屏幕。也就是说，当我们点击页面的时候移动端浏览器并不是立即作出反应，而是会等上一小会儿才会出现点击的效果。

点击穿透：假如页面上有两个元素 A 和 B。B 元素在 A 元素之上。我们在 B 元素的 touchstart 事件上注册了一个回调函数，该回调函数的作用是隐藏 B 元素。我们发现，当我们点击 B 元素，B 元素被隐藏了，随后，A 元素触发了 click 事件。这是因为在移动端浏览器，事件执行的顺序是 touchstart > touchend > click。而 click 事件有 300ms 的延迟，当 touchstart 事件把 B 元素隐藏之后，隔了 300ms，浏览器触发了 click 事件，但是此时 B 元素不见了，所以该事件被派发到了 A 元素身上。如果 A 元素是一个链接，那此时页面就会意外地跳转。

300ms 延迟解决方案：

（1）禁用缩放，在 html 文档头部加 meta 标签如下：

```
<meta name=”viewport” content=”user-scalable=no”/>
```

（2）更改默认的视口宽度 （响应式布局，消除了站点上可能存在的双击绽放的请求）

```
<meta name=”viewport” content=”width=device-width”/>
```

（3）Css touch-action

touch-action:none;在该元素上的操作不会触发用户代理的任何行为，无需进行 3000ms 延迟判断。

（4）FastClick 为解决移动端浏览器 300 毫秒延迟开发的一个轻量级的库

点击穿透解决方案：

（1）只用 touch

（2）只用 click

（3）tap 后延迟 350ms 再隐藏 mask

（4）pointer-events

70.变量作用域?

变量作用域：一个变量的作用域是程序源代码中定义这个变量的区域。全局变量拥有全局作用域，在 js 代码中任何地方都是有定义的。在函数内声明的变量只在函数体内有定义，它们是局部变量，作用域是局部性的。函数参数也是局部变量，它们只在函数体内有定义。

```
var a = "";
window.b="";
function(e) {
        var c= "";
        d="";
        e="";
```

```
}
function go() {
        console.info(this);//window
        return function() {
                console.info(this); // window
                return {
               b:function(){
                        console.info(this); //b 的父对象
                    }
            }
        }
}
go()().b();
```

71.call & apply 两者之间的区别

call 和 apply 都是改变 this 指向的方法，区别在于 call 可以写多个参数，而 apply 只能写两个参数，第二个参数是一个数组，用于存放要传的参数。

72.call 和 apply 有什么好处？

用 call 和 apply:实现更好的继承和扩展，更安全。

73.谁是 c 的构造函数?

```
function ab() {
        this.say = "";
}
ab.constructor = {}
ab.name = '';
var c = new ab();
```

构造函数默认指向函数本身,ab 是一个类,它的构造函数是它本身，然后 ab.constructor={};ab 的构造函数就指向{}了，c 是 ab 的实例化对象，c 的构造函数就是{}。

通过使用 new 的时候 创建对象 发生了那些改变?

当使用 new 操作时，会马上开辟一个块内存，创建一个空对象，并将 this 指向这个对象。接着，执行构造函数 ab()，对这个空对象进行构造（构造函数里有什么属性和方法都一一给这个空白对象装配上去，这就是为何它叫构造函数了）。

74.sass 和 less 有什么区别?

1.编译环境不一样

Sass 的安装需要 Ruby 环境，是在服务端处理的，而 Less 是需要引入 less.js 来处理 Less 代码输出 css 到浏览器，也可以在开发环节使用 Less，然后编译成 css 文件，直接放到项目中。

2.变量符不一相，less 是@，而 scss 是$,而且它们的作用域也不一样，less 是块级作用域

3.输出设置，Less 没有输出设置，sass 提供 4 种输出选项，nested,compact,compressed 和 expanded

nested：嵌套缩进的 css 代码(默认)

expanded：展开的多行 css 代码

compact：简洁格式的 css 代码

compressed：压缩后的 css 代码

4.sass 支持条件语句，可以使用 if{}else{},for{}循环等等，而 less 不行

5.引用外部 css 文件，sass 引用外部文件必须以_开头，文件名如果以下划线_形状，sass 会认为该文件是一个引用文件，不会将其编译为 css 文件。less 引用外部文件和 css 中的 @import 没什么差异。

6.sass 和 less 的工具库不同。sass 有工具库 Compass, 简单说，sass 和 Compass 的关系有点像 Javascript 和 jQuery 的关系,Compass 是 sass 的工具库。在它的基础上，封装了一系列有用的模块和模板，补充强化了 sass 的功能。less 有 UI 组件库 Bootstrap,Bootstrap 是 web 前端开发中一个比较有名的前端 UI 组件库，Bootstrap 的样式文件部分源码就是采用 less 语法编写。

总结：不管是 sass，还是 less，都可以视为一种基于 CSS 之上的高级语言，其目的是使得 CSS 开发更灵活和更强大，sass 的功能比 less 强大,基本可以说是一种真正的编程语言了，less 则相对清晰明了,易于上手,对编译环境要求比较宽松。考虑到编译 sass 要安装 Ruby,而 Ruby 官网在国内访问不了,个人在实际开发中更倾向于选择 less。

75.bootstrap 好处？

自适应和响应式布局，12 栅格系统，统一的界面风格和 css 样式有利于团队开发。编写灵活、稳定、高质量的 HTML 和 CSS 代码的规范。

76.开发时如何对项目进行管理?gulp?

答：本人开发时，利用 gulp 等前端工作流管理工具管理项目。

gulp 是新一代的前端项目构建工具，你可以使用 gulp 及其插件对你的项目代码（less,sass）进行编译，还可以压缩你的 js 和 css 代码，甚至压缩你的图片，能够合并文件，压缩文件，语法检查，监听文件变化，测试，转换二进制，转换图片等一系列功能。gulp 仅有少量的 API，所以非常容易学习。实现良好的项目管理。

77.压缩合并目的？http 请求的优化方式？

答：1）Web 性能优化最佳实践中最重要的一条是减少 HTTP 请求。而减少 HTTP 请求的最主要的方式就是，合并并压缩 JavaScript 和 CSS 文件。

CSS Sprites（CSS 精灵）：把全站的图标都放在一个图像文件中，然后用 CSS 的 background-image 和 background-position 属性定位来显示其中的一小部分。

合并脚本和样式表;

图片地图：利用 image map 标签定义一个客户端图像映射，（图像映射指带有可点击区域的一幅图像）具体看：http://club.topsage.com/thread-2527479-1-1.html

④图片 js/css 等静态资源放在静态服务器或 CDN 服时，尽量采用不用的域名，这样能防止 cookie 不会互相污染，减少每次请求的往返数据。

⑤css 替代图片

⑥缓存一些数据

⑦少用 location.reload()：使用 location.reload() 会刷新页面，刷新页面时页面所有资源（css，js，img 等）会重新请求服务器。建议使用 location.href="当前页 url" 代替 location.reload() ，使用 location.href 浏览器会读取本地缓存资源。

78.ajax 请求方式有几种（8 种）？

1）$.get(url,[data],[callback])

2）$.getJSON(url,[data],[callback])

3）$.post(url,[data],[callback],[type])

4）$.ajax(opiton)

5）$.getScript(url, [callback])

6）jquery 对象.load(url, [data], [callback])

7）serialize() 与 serializeArray()
79.如何 copy 一个 dom 元素？
原生 Js 方法：var div = document.getElementsByTagName('div')[0];
var clone = div.cloneNode();
Jquery 方法：$('div').clone();
在默认情况下，.clone()方法不会复制匹配的元素或其后代元素中绑定的事件。不过，可以为这个方法传递一个布尔值参数，将这个参数设置为 true，就可以连同事件一起复制，即.clone(true)。
80.数组的排序方法（sort）？排序？汉字排序？
1）数组的排序方法：reverse()和 sort()。reverse()方法会对反转数组项的顺序。
Eg:var values = [0, 1, 5, 10, 15];
values.sort();//0,1,10,15,5
var values = [1, 2, 3, 4, 5];
values.reverse();//5,4,3,2,1
js 中的排序（详情参考：http://www.tuicool.com/articles/IjInMbU）
利用 sort 排序
冒泡排序
快速排序
插入排序
希尔排序
选择排序
归并排序
localeCompare() 方法用于字符串编码的排序
localeCompare 方法：返回一个值，指出在当前的区域设置中两个字符串是否相同。
81.简述一下你理解的面向对象？
面向对象是基于万物皆对象这个哲学观点. 把一个对象抽象成类,具体上就是把一个对象的静态特征和动态特征抽象成属性和方法,也就是把一类事物的算法和数据结构封装在一个类之中,程序就是多个对象和互相之间的通信组成的.
面向对象具有封装性,继承性,多态性。
封装:隐蔽了对象内部不需要暴露的细节,使得内部细节的变动跟外界脱离,只依靠接口进行通信.封装性降低了编程的复杂性. 通过继承,使得新建一个类变得容易,一个类从派生类那里获得其非私有的方法和公用属性的繁琐工作交给了编译器. 而 继承和实现接口和运行时的类型绑定机制 所产生的多态,使得不同的类所产生的对象能够对相同的消息作出不同的反应,极大地提高了代码的通用性.
总之,面向对象的特性提高了大型程序的重用性和可维护性.
82.如何创建一个对象？
1. 工厂模式
2. 构造函数模式
3. 原型模式
4. 混合构造函数和原型模式
5. 动态原型模式
6. 寄生构造函数模式
7. 稳妥构造函数模式
程序的设计模式?工厂模式?发布订阅?

1）设计模式并不是某种语言的某块代码，设计模式是一种思想，提供给在编码时候遇到的各种问题是可以采取的解决方案，更倾向于一种逻辑思维，而不是万能代码块。
设计模式主要分三个类型:创建型、结构型和行为型。
创建型模式：单例模式，抽象工厂模式，建造者模式，工厂模式与原型模式。
结构型模式：适配器模式，桥接模式，装饰者模式，组合模式，外观模式，享元模式以及代理模式。
行为型模式：模板方法模式，命令模式，迭代器模式，观察者模式，中介者模式，备忘录模式，解释器模式，状态模式，策略模式，职责链模式和访问者模式。
2）与创建型模式类似，工厂模式创建对象（视为工厂里的产品）是无需指定创建对象的具体类。
工厂模式定义一个用于创建对象的接口，这个接口由子类决定实例化哪一个类。该模式使一个类的实例化延迟到了子类。而子类可以重写接口方法以便创建的时候指定自己的对象类型。
3）观察者模式又叫做发布订阅模式，它定义了一种一对多的关系，让多个观察者对象同时监听某一个主题对象，这个主题对象的状态发生改变时就会通知所有观察着对象。它是由两类对象组成，主题和观察者，主题负责发布事件，同时观察者通过订阅这些事件来观察该主体，发布者和订阅者是完全解耦的，彼此不知道对方的存在，两者仅仅共享一个自定义事件的名称。
（设计模式实在是太高深了，小伙伴门结合网上实例自行学习，我实在是无能为力啊）
83.commonjs?requirejs?AMD|CMD|UMD?
1.CommonJS 就是为 JS 的表现来制定规范，NodeJS 是这种规范的实现，webpack 也是以 CommonJS 的形式来书写。因为 js 没有模块的功能，所以 CommonJS 应运而生。但它不能在浏览器中运行。
CommonJS 定义的模块分为:{模块引用(require)} {模块定义(exports)} {模块标识(module)}
2.RequireJS 是一个 JavaScript 模块加载器。
RequireJS 有两个主要方法(method): define()和 require()。这两个方法基本上拥有相同的定义(declaration) 并且它们都知道如何加载的依赖关系，然后执行一个回调函数(callback function)。与 require()不同的是， define()用来存储代码作为一个已命名的模块。因此 define()的回调函数需要有一个返回值作为这个模块定义。这些类似被定义的模块叫作 AMD (Asynchronous Module Definition，异步模块定义)。
3.AMD 是 RequireJS 在推广过程中对模块定义的规范化产出
AMD 异步加载模块。它的模块支持对象 函数 构造器 字符串 JSON 等各种类型的模块。适用 AMD 规范适用 define 方法定义模块。

4.CMD 是 SeaJS 在推广过程中对模块定义的规范化产出
AMD 与 CDM 的区别：
（1）对于于依赖的模块，AMD 是提前执行(好像现在也可以延迟执行了)，CMD 是延迟执行。
（2）AMD 推崇依赖前置，CMD 推崇依赖就近。
（3）AMD 推崇复用接口，CMD 推崇单用接口。
（4）书写规范的差异。
5.umd 是 AMD 和 CommonJS 的糅合。
AMD 浏览器第一的原则发展 异步加载模块。
CommonJS 模块以服务器第一原则发展，选择同步加载，它的模块无需包装(unwrapped

modules)。
这迫使人们又想出另一个更通用的模式 UMD （Universal Module Definition）。希望解决跨平台的解决方案。
UMD 先判断是否支持 Node.js 的模块（exports）是否存在，存在则使用 Node.js 模块模式。
84、js 的几种继承方式？
1.使用对象冒充实现继承
2.采用 call、Apply 方法改变函数上下文实现继承
3.原型链方式继承
85、JavaScript 原型，原型链 ？有什么特点？
在 JavaScript 中,一共有两种类型的值,原始值和对象值.每个对象都有一个内部属性[[prototype]],我们通常称之为原型.原型的值可以是一个对象,也可以是 null.如果它的值是一个对象,则这个对象也一定有自己的原型.这样就形成了一条线性的链,我们称之为原型链.
访问一个对象的原型可以使用 ES5 中的 Object.getPrototypeOf 方法,或者 ES6 中的__proto__属性.
原型链的作用是用来实现继承,比如我们新建一个数组,数组的方法就是从数组的原型上继承而来的。
86、eval 是做什么的？
它的功能是把对应的字符串解析成 JS 代码并运行； 应该避免使用 eval，不安全，非常耗性能（2 次，一次解析成 js 语句，一次执行）。
87、null，undefined 的区别？
undefined 表示变量声明但未初始化的值，null 表示准备用来保存对象，还没有真正保存对象的值。从逻辑角度看，null 表示一个空对象指针。
88、JSON 的了解？
JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。 它是基于 JavaScript 的一个子集。数据格式简单, 易于读写, 占用带宽小。
89、js 延迟加载的方式有哪些？
defer 和 async、动态创建 DOM 方式（用得最多）、按需异步载入 js
90、ajax 是什么?
异步 javascript 和 XML，是指一种创建交互式网页应用的网页开发技术。通过后台与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。
91、同步和异步的区别?
javascript 同步表示 sync，指：代码依次执行
javascript 异步表示 async，指：代码执行不按顺序，‘跳过’执行，待其他某些代码执行完后再来执行，成为异步。
92、 如何解决跨域问题?
Jsonp、iframe、window.name、window.postMessage、服务器上设置代理页面

93、异步加载的方式有哪些？
(1) defer，只支持 IE
(2) async：true
(3) 创建 script，插入到 DOM 中，加载完毕后 callBack
94、jQuery 与 jQuery UI 有啥区别？
jQuery 是一个 js 库，主要提供的功能是选择器，属性修改和事件绑定等等。

jQuery UI 则是在 jQuery 的基础上，利用 jQuery 的扩展性，设计的插件。提供了一些常用的界面元素，诸如对话框、拖动行为、改变大小行为等等。

95、你有哪些性能优化的方法？

（1） 减少 http 请求次数：CSS Sprites, JS、CSS 源码压缩、图片大小控制合适；网页 Gzip，CDN 托管，data 缓存 ，图片服务器。

（2） 前端模板 JS+数据，减少由于 HTML 标签导致的带宽浪费，前端用变量保存 AJAX 请求结果，每次操作本地变量，不用请求，减少请求次数

（3） 用 innerHTML 代替 DOM 操作，减少 DOM 操作次数，优化 javascript 性能。

（4） 当需要设置的样式很多时设置 className 而不是直接操作 style。

（5） 少用全局变量、缓存 DOM 节点查找的结果。减少 IO 读取操作。

（6） 避免使用 CSS Expression（css 表达式)又称 Dynamic properties(动态属性)。

（7） 图片预加载，将样式表放在顶部，将脚本放在底部 加上时间戳。

（8） 避免在页面的主体布局中使用 table，table 要等其中的内容完全下载之后才会显示出来，显示比 div+css 布局慢。

96、一个页面从输入 URL 到页面加载显示完成，这个过程中都发生了什么？（流程说的越详细越好）

查找浏览器缓存

DNS 解析、查找该域名对应的 IP 地址、重定向（301）、发出第二个 GET 请求

进行 HTTP 协议会话

客户端发送报头(请求报头)

服务器回馈报头(响应报头)

html 文档开始下载

文档树建立，根据标记请求所需指定 MIME 类型的文件

文件显示

浏览器这边做的工作大致分为以下几步：

加载：根据请求的 URL 进行域名解析，向服务器发起请求，接收文件（HTML、JS、CSS、图象等）。

解析：对加载到的资源（HTML、JS、CSS 等）进行语法解析，建议相应的内部数据结构（比如 HTML 的 DOM 树，JS 的（对象）属性表，CSS 的样式规则等等）

97、ajax 的缺点

1、ajax 不支持浏览器 back 按钮。

2、安全问题 AJAX 暴露了与服务器交互的细节。

3、对搜索引擎的支持比较弱。

4、破坏了程序的异常机制。

5、不容易调试

argularJS

1.jquery 和 angularjs 的区别？

jQuery 在 DOM 上做得很好，可以根据用户交互，添加修改 DOM 元素。AngularJS 更关注数据展示本身，更新时会尽可能减少对 DOM 的破坏和重构。AngularJS 中很多特点的设计都是出于提高开发者效率的目的。 当一个项目的重点是数据展示和执行，而不是分析，此时可能 AngularJS 就会更胜一筹。

2.ng 的三种服务?哪个服务可以在 config 中进行调用?
Provider factory service
Provider
3.angular 一些优化手段
DOM 树要小，DOM 的访问要尽可能的少。尽量避免使用过滤器。当使用 ng-repeat 时要尽量避免对全局列表的刷新。尽量减少绑定。把变量作用范围限制地越紧密越好，这样垃圾回收器就可以更快地回收空间
4.ng 中如何进行项目结构的
将相同的数据功能单独的创建一个文件进行管理维护，方便调用。进行分层开发，分为 controller 控制层，filter 过滤层，service 服务层
5.ng 指令中 resctrict 是什么意思 ？EACM
生效指令
E:元素　　A：属性　　C：class　　M：注释
6.ng 指令中 scope ? true|false|{} ?分别代表什么意思?
不设置 scope 情况下，link 中 scope 与容器 controller 公用一块 scope
Scope 设置为 true 的情况下，继承容器的作用域，创建自己的作用域
Scope 设置为 false，link 中 scope 与容器 controller 公用一块 scope
Scope 设置为{}，继承 controller 容器 scope 作用域，但是会生成一块独立的作用域空间，不会自动调用父 scope 的变量或方法
7.ng 指令中 transclude 意义?
是否对指令内部包含的内容进行保存 配合 ng-transclude 使用
8.MVC?MVVM?
MVC :M 是指数据模型，V 是指用户界面，C 则是控制器。使用 MVC 的目的是将 M 和 V 的实现代码分离，从而使同一个程序可以使用不同的表现形式。C 存在的目的则是确保 M 和 V 的同步，一旦 M 改变，V 应该同步更新。
MVVM(Model View ViewModel) :是一种基于 MVC 和 MVP 的架构模式，它试图将用户界面（UI）从业务逻辑和行为中更加清晰地分离出来。为了这个目的，很多例子使用声明变量绑定来把 View 层的工作从其他层分离出来。
9.angular 版本?怎么去兼容 ie8？ng 中数据一般放在哪里?
目前我们使用的是 angularJS 1.5 版本，如何去兼容 IE8？
ng 中数据一般放在控制层。
10.ng 的三种服务?区别?
Provider:执行$get 方法，可以在 ng 的 config 配置方法中定义变量和参数；
Factory：对$get 进行封装，执行一个函数，返回一个对象；
Service：返回一个构造函数，在注入时进行实例化；
11.ng 编程时，你们的项目结构是什么样的 ?分别有什么特点 ？
分为控制层、过滤层、服务层
控制层：主要处理数据传输，页面渲染，数据绑定
过滤层：对数据进行过滤
服务层：ajax
12.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。
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。

```
<p>{{name}}</p>
<div ng-if="true">
        <input type="text" ng-model="name">
</div>
```

ng-show 不存在此问题，因为它不自带一级作用域。
避免这类问题出现的办法是，始终将页面中的元素绑定到对象的属性（data.x）而不是直接绑定到基本变量（x）上。
13.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）。
14.ng-click 中写的表达式，能使用 JS 原生对象上的方法吗？
不止是 ng-click 中的表达式，只要是在页面中，都不能直接调用原生的 JS 方法，因为这些并不存在于与页面对应的 Controller 的 $scope 中。
15.factory、service 和 provider 是什么关系？
共同点：都属于单例模式
Factory:把 service 的方法和数据放在一个对象里，并返回这个对象
Service:通过构造函数方式创建 service，返回一个实例化对象
Provider:创建一个可通过 config 配置的 service，$get 中返回的，就是用 factory 创建 service 的内容
从底层实现上来看，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 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。
16.angular 的数据绑定采用什么机制？详述原理
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 $watch，最后更新 dom。
17.如何在平级界面模块间进行通信
有两种方法，一种是共用服务，一种是基于事件。
共用服务:
在 Angular 中，通过 factory 可以生成一个单例对象，在需要通信的模块 a 和 b 中注入这个对象即可。
基于事件:
这个又分两种方式
第一种是借助父 controller。在子 controller 中向父 controller 触发（$emit）一个

事件，然后在父 controller 中监听（$on）事件，再广播（$broadcast）给子 controller ，这样通过事件携带的参数，实现了数据经过父 controller，在同级 controller 之间传播。

第二种是借助 $rootScope。每个 Angular 应用默认有一个根作用域 $rootScope，根作用域位于最顶层，从它往下挂着各级作用域。所以，如果子控制器直接使用 $rootScope 广播和接收事件，那么就可实现同级之间的通信。

18.一个 angular 应用应当如何良好地分层？

(1) 目录结构的划分

对于小型项目，可以按照文件类型组织

但是对于规模较大的项目，最好按业务模块划分

(2)逻辑代码的拆分

作为一个 MVVM 框架，Angular 应用本身就应该按照 模型，视图模型（控制器），视图来划分。

这里逻辑代码的拆分，主要是指尽量让 controller 这一层很薄。提取共用的逻辑到 service 中 （比如后台数据的请求，数据的共享和缓存，基于事件的模块间通信等），提取共用的界面操作到 directive 中（比如将日期选择、分页等封装成组件等），提取共用的格式化操作到 filter 中等等。

在复杂的应用中，也可以为实体建立对应的构造函数，比如硬盘（Disk）模块，可能有列表、新建、详情这样几个视图，并分别对应的有 controller，那么可以建一个 Disk 构造函数，里面完成数据的增删改查和验证操作，有跟 Disk 相关的 controller，就注入 Disk 构造器并生成一个实例，这个实例就具备了增删改查和验证方法。这样既层次分明，又实现了复用（让 controller 层更薄了）。

19.angular 应用常用哪些路由库，各自的区别是什么？

Angular1.x 中常用 ngRoute 和 ui.router，还有一种为 Angular2 设计的 new router（面向组件）。

无论是 ngRoute 还是 ui.router，作为框架额外的附加功能，都必须以 模块依赖 的形式被引入。

区别:

ngRoute 模块是 Angular 自带的路由模块，而 ui.router 模块是基于 ngRoute 模块开发的第三方模块。

ui.router 是基于 state （状态）的， ngRoute 是基于 url 的，ui.router 模块具有更强大的功能，主要体现在视图的嵌套方面。

使用 ui.router 能够定义有明确父子关系的路由，并通过 ui-view 指令将子路由模版插入到父路由模板的 <div ui-view> 中去，从而实现视图嵌套。而在 ngRoute 中不能这样定义，如果同时在父子视图中 使用了 <div ng-view> 会陷入死循环。

20.如果通过 angular 的 directive 规划一套全组件化体系，可能遇到哪些挑战？

组件如何与外界进行数据的交互，以及如何通过简单的配置就能使用

21.分属不同团队进行开发的 angular 应用，如果要做整合，可能会遇到哪些问题，如何解决？

可能会遇到不同模块之间的冲突。

比如一个团队所有的开发在 moduleA 下进行，另一团队开发的代码在 moduleB 下，会导致两个 module 下面的 serviceA 发生了覆盖。

在 Angular1.x 中并没有很好的解决办法，所以最好在前期进行统一规划，做好约定，严格按照约定开发，每个开发人员只写特定区块代码。

22.angular 的缺点有哪些？

（1）强约束

导致学习成本较高，对前端不友好。但遵守 AngularJS 的约定时，生产力会很高，对 Java 程序员友好。

（2）不利于 SEO

因为所有内容都是动态获取并渲染生成的，搜索引擎没法爬取。

一种解决办法是，对于正常用户的访问，服务器响应 AngularJS 应用的内容；对于搜索引擎的访问，则响应专门针对 SEO 的 HTML 页面。

（3）性能问题

作为 MVVM 框架，因为实现了数据的双向绑定，对于大数组、复杂对象会存在性能问题。

23.优化 Angular 应用的性能的办法？

（1）减少监控项，即减少 watcher 数量（比如对不会变化的数据采用单向绑定）

（2）主动设置索引（指定 track by，简单类型默认用自身当索引，对象默认使用 $$hashKey，比如改为 track by item.id）

（3）降低渲染数据量（比如分页，或者每次取一小部分数据，根据需要再取）

（4）数据扁平化（比如对于树状结构，使用扁平化结构，构建一个 map 和树状数据，对树操作时，由于跟扁平数据同一引用，树状数据变更会同步到原始的扁平数据）

24.如何看待 angular 1.2 中引入的 controller as 语法？

最根本的好处

在 angular 1.2 以前，在 view 上的任何绑定都是直接绑定在 $scope 上的，

使用 controllerAs，不需要再注入 $scope，controller 变成了一个很简单的 javascript 对象（POJO），一个更纯粹的 ViewModel。

25.详述 angular 的“依赖注入”

依赖注入是一种软件设计模式，目的是处理代码之间的依赖关系，减少组件间的耦合。

原理：

AngularJS 是通过构造函数的参数名字来推断依赖服务名称的，通过 toString() 来找到这个定义的 function 对应的字符串，然后用正则解析出其中的参数（依赖项），再去依赖映射中取到对应的依赖，实例化之后传入。

通常会使用下面两种方式注入依赖（对依赖添加的顺序有要求）。

（1）数组注释法

（2）显式 $inject

26.如何看待 angular2？

（1）相比 Angular1.x，Angular2 的改动很大，几乎算是一个全新的框架。

（2）基于 TypeScript（可以使用 TypeScript 进行开发），在大型项目团队协作时，强语言类型更有利。

（3）组件化，提升开发和维护的效率。

（4）还有 module 支持动态加载，new router，promise 的原生支持等等。

（5）迎合未来标准，吸纳其他框架的优点，值得期待，不过同时要学习的东西也更多了（ES next、TS、Rx 等）。

27.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。

28.Angular 中的 digest 周期是什么？

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。
29.如何取消 $timeout, 以及停止一个$watch()?
（1）停止 $timeout 我们可以用 cancel：
```
var customTimeout = $timeout(function () {
// your code
}, 1000);
$timeout.cancel(customTimeout);
```
停掉一个$watch：
```
// .$watch() 会返回一个停止注册的函数
var deregisterWatchFn = $rootScope.$watch('someGloballyAvailableProperty', function (newVal) {
if (newVal) {
  // we invoke that deregistration function, to disable the watch
  deregisterWatchFn();
  ...
}
});
```
30.列出至少三种实现不同模块之间通信方式？
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
31.解释下什么是$rootScrope 以及和$scope 的区别？
通俗的说$rootScrope 是页面所有$scope 的父亲。
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。

其它
1.restful 规范
认为一切接口都是资源，每个资源和每个资源都可以通过一定的关系进行连接或管理。
rest 对接口进行简单的规范定义，即 post，delete,put/patch,get
2、react？
提供虚拟 dom 节点
3、webpack，gulp？
Gulp，文件的压缩打包合并，是一个纯粹的工具，并不能将你的 css 等非 js 资源模块化，但是 webpack 可以做到这些。
webpack 模块化加载器兼打包工具，可以把各种资源都作为模块来使用和处理，具有前端构建的功能，只不过通过插件实现了构建工具的一些功能。
webpack 的优点如下：
1. webpack 遵循 commonJS 的形式，但对 AMD/CMD 的支持也很全面，方便旧项目进行

代码迁移。
2. 能被模块化的不仅仅是 JS ，所有的静态资源，例如 css，图片等都能模块化，即以 require 的方式引入。
3. 开发便捷，能替代部分 grunt/gulp 的工作，比如打包、压缩混淆、图片转 base64 等。
4、函数的两个内置对象？
This,arguments
5、布局方式？
rem,百分比，Flex,meta 标签实现响应式
6、js 中创建对象的几种方式？
构造函数方式；工厂方式；混合模式；代理模式；动态原型方式（推荐使用这种模式）
7、确保浏览器不走缓存路线？
1、在 ajax 发送请求前加上 anyAjaxObj.setRequestHeader("If-Modified-Since","0")
2、在 ajax 发送请求前加上 anyAjaxObj.setRequestHeader("Cache-Control","no-cache")
3、在 URL 后面加上一个随机数： "fresh=" + Math.random();
4、在 URL 后面加上时间搓："nowtime=" + new Date().getTime();
5、如果是使用 jQuery，直接这样就可以了$.ajaxSetup({cache:false})，这样页面的所有 ajax 都会执行这条语句就是不需要保存缓存记录。
8、移动端开发，会发现 input 在 iscroll 中不能输入
使用了 iscroll 之后，你会发现点击输入框时不灵敏，经常无法聚焦；页面文字也无法选择和复制。这是由于 iscroll 要监听鼠标事件和触摸事件来进行滚动，所以禁止了浏览器的默认行为，导致上述问题。所以我们需要稍作修改：

```
onBeforeScrollStart: function (e) {
    var target = e.target;
    while (target.nodeType != 1)
        target = target.parentNode;
    if (target.tagName != ‘SELECT’ && target.tagName != ‘INPUT’ && target.tagName != ‘TEXTAREA’)
        e.preventDefault();
}
```

9、angularjs 中的自定义指令的 scope 有几个值，分别代表什么？
1.scope 设置为 true 的情况下，继承容器的作用域，创建自己的作用域
2.设置 scope 为 false 时，link 中 scope 与容器 controller 公用一块 scope
3.如果 scope 设置为{}对象，继承 controller 容器 scope 作用域,但是会生成一块独立的作用域空间，不会自动调用父 scope 的变量或方法
当 scope 设置为{}时：可以声明给 scope 传递变量和值
3 种方式传递 = & @
@通过在标签上定义属性,在 scope 使用@属性名传递变量
&通过在标签上定义属性，在 scope 中使用&传递进来一个'函数',且只能是函数
=通过在标签上定义属性，在父 scope 上寻找到匹配的变量和函数为当前的函数属性进行复制。
10、Node.js 的适用场景？
高并发、聊天、实时消息推送