HTML&css相关问题
1.XHTML和HTML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的置标语言
最主要的不同
XHTML元素必须被正确地嵌套。
XHTML元素必须被关闭
标签名必须用小写字母
XHTML文档必须拥有根元素
2.什么是语义化的HTML？
直观的认识标签对于搜索引擎的抓取有好处，用正确的标签做正确的事情！
HTML语义化就是让页面的内容结构化，便于对浏览器，搜索引擎解析；
在没有样式css情况下也以一种文档格式显示，并且是容易阅读。搜索引擎的爬虫依赖于标记来确定上下文和各个关键字的权重，利于SEO。
在阅读源代码的人对网站更容易将网站分块，便于阅读维护理解。
2（1）、简述一下你对HTML语义化的理解？
1、用正确的标签做正确的事情。
2、html语义化让页面的内容结构化，结构更清晰，便于对浏览器，搜索引擎解析；
3、即使在没有样式CSS情况下也以一种文档格式显示，并且是容易阅读的；
4、搜索引擎的爬虫也依赖于HTML标记确定上下文和各个关键字的权重，利用SEO；
5、使阅读源代码的人对网站更容易将网站分块，便于阅读维护理解。
3.常见的浏览器内核有哪些？
Trident内核：IE,MaxThon,TT,The Word,360,搜狗浏览器等。[又称为MSHTML]
Gecko内核：Netscape6及以上版本，FF,MozillaSuite/SeaMonkey等；
Presto内核：Opera7及以上。[Opera内核原为：Presto，现为：Blink]
Webkit内核：Safari,Chrome等。[Chrome的:Blink(Webkit的分支)]
3(1).常见哪几种浏览器测试？有哪些内核（Layout Engine）？
浏览器：IE、Chrome、FireFox、Safari、Opera
内核：Trident、Gecko、Presto、Webkit
4.HTML5有哪些新特性，移除了哪些元素？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。
绘画canvas
用于媒介回放的video和audio元素
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
语意化更好的内容元素，比如article,footer,header,nav,section
表单控件：calendar,date,time,email,url,search
新的技术webworker,websocktGeolocation
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。
5.请描述一下cookies,sessionStorage（会话存储）和localStrorage（本地存储）的区别？
cookie在浏览器和服务器间来回传递。sessionStorage和localStorage不会；
sessionStorage和localStorage的存储空间更大；
sessionStorage和localStorage有更多丰富易用的接口；

sessionStorage和localStorage各自独立的存储空间；

5、（1）请描述一下cookies、sessionStorage和localStorage区别？
cookie是网站为了标示用户身份而储存在用户本地终端（Client Side）上的数据（通常经过加密）。
cookie数据始终在同源的http请求中携带（即使不需要），即会在浏览器和服务器间来回传递。
sessionStorage和localStorage不会自动把数据发给服务器，仅在本地保存
存储大小：
cookie数据大小不能超过4K。
sessionStorage和localStorage虽然也有存储大小的限制，但比cookie大得多，可以达到5M或更大。
有期时间：
localStorage：存储持久数据，浏览器关闭后数据不丢失除非主动删除数据；
sessionStorage：数据在当前浏览器窗口关闭后自动删除
cookie：设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭
6.如何实现浏览器内多个标签页之间的通信？
调用localstorage,cookies等本地存储方式
WebSocket、SharedWorker
localstroge另一个浏览器上下文被添加、删除或修改时，它都会触发一个事件，我们通过监听事件，控制它的值来进行页面信息通信。
注意quirks:Safari在无痕迹模式下设置localstorge值抛出QuotExceededError的异常。
7.HTML5为什么只需要写！DOCTYPE HTML？
Html5不基于SGML,因此不需要对DTD进行引用，但是需要doctype来规范浏览器的行为(让浏览器按照它们应该的方式来运行)；而HTml4.01基于SGMA，所以需要对DTD进行引用，才能告知浏览器文档所使用的文档类型。
8.Doctype作用？标准模式与兼容模式各种什么区别？
！Doctype声明位于HTML文档的第一行，处于html标签之前。告知浏览器的解析器用什么文档标准解析这个文档。
doctype不存在或格式不正确会导致文档以兼容模式呈现。
标准模式的排版和JS运作模式都是该浏览器支持的最高标准运行。在兼容模式中，页面以宽松的向后兼容的方式来显示，模拟老式浏览器的行为以防止站点无法工作。
9.Doctype？严格模式与混杂模式如何触发这两种模式，区分它们有何意义？
用于声明文档使用哪种规范（html/Xhtml）一般为严格过渡基于框架的html文档。
加入XML声明可触发，解析方式更改为IE5.5拥有IE5.5的Bug。
9(1)、HTML5为什么只需要写<!DOCTYPE HTML>？
HTML5不基于SGML，因此不需要对DTD进行引用，但是需要doctype来规范浏览器的行为（让浏览器按照它们应该的方式来进行）
而HTML4.01基于SGML，所以需要对DTD进行引用，才能告知浏览器文档所使用的文档类型。
10、html document是干嘛的？
HTML即：超文本标记语言，标准通用标记语言的一个应用，“超文本”就是指页面内可以包含图片、链接、甚至音乐、程序等非文字元素。
HTML Document即：HTML Document对象，每个载入浏览器的HTML文档都会成为Document对象
由于Document对象是window对象的一部分，所以可通过window.document属性对其进行访问。
11、html5哪些操作可以SEO优化
title标签；meta标签；header标签；footer标签
元标签（meta标签）；导航标签（nav标签）；文章标签（article标签）；左或右侧标签（aside标签）
12、css盒模型有哪些及区别content-box border-box padding-box
Q1
IE盒子模型box-sizing:border-box;（怪异模式）
W3C标准盒子模型 box-sizing:content-box;（标准模式）默认模式

Q2
content-box:这是默认样式指定CSS标准。测量width和height属性只包括的内容，但不是border，margin，或者padding。
padding-box:width和height属性包括padding的大小，不包括border和margin
border-box:width和height属性包括padding和border，但不是margin。这是盒模型的文档时，Internet Explorer使用Quirks模式。

content-box不包含padding，border-box包含padding。所以如果你设置的大小是一样的，content-box看起来，会

比border-box大

13、行内元素和块状元素的区别？行内快元素的兼容性使用？（IE8以下）
行内元素：会在水平方向排列，不能包含快级元素，设置width无效，height无效（可以设置line-height），margin上下无效，padding上下无效
块级元素：各占据一行，垂直方向排列。从新行开始结束接着一个断行
兼容性：display:inline-block;display:inline;zoom:1;
14、页面导入样式时，使用link和@import有什么区别？
1.link属于HTML标签，除了加载CSS外，还能用于定义RSS，定义rel连接属性等作用；而@import是CSS提供，只能加载CSS;
2.页面被加载的时，link会同时被加载，而@import引用的CSS会等到页面被加载完再加载；
import是CSS2.1提出的，只在IE5以上才能被识别，而link是HTML标签，无兼容问题；
14(1).css引入的方式有哪些？link和@import的区别是？
内联，内嵌，外链，导入
区别：同时加载，
前者无兼容性，后者css2.1以下浏览器不支持
link支持使用javascript改变样式，后者不可。
15、介绍以下你对浏览器内核的理解？
1、主要分成两部分：渲染引擎（layout engineer或Rendering Engine）和JS引擎。
2、渲染引擎：负责取得网页的内容（HTML、XML、图像等等）、整理讯息（例如加入CSS等）、以及计算网页的显示方式、然后会输出至显示器或打印机。浏览器的内核的不同对于网页的语法解释会有不同、所以渲染的效果也不相同。所有网页浏览器、电子邮件客户端以及其他需要编辑、显示网络内容的应用程序都需要内核
3、JS引擎则：解析和执行javascript来实现网页的动态效果。
4、最开始渲染引擎和JS引擎并没有区分得很明确，后来JS引擎越来越独立，内核九倾向于只指渲染引擎。
16、box-sizing常用的属性有哪些？分别有什么作用？
box-sizing：content-box|border-box|inherit
content-box：宽度和高度分别应用到元素的内容框。
17、清楚浮动有哪些方式？比较好的方式是哪一种
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也浮动，需要定义宽度。
7、父级div定义display：table。
8、结尾处加br标签clear:both。
比较好的是第3种，好多网站都这样用
18、html5有哪些新特性？如何处理HTML5新标签的浏览器兼容问题？如何区分HTML和HTML5？
Q1
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；
1、绘画canvas；
2、用于媒介回放的video和auto元素；
3、本地离线存储localStorage长期存储数据，浏览器关闭后数据不丢失；
4、sessionStorage的数据在浏览器关闭后自动删除；
5、语意化更好的内容元素，比如article、footer、header、nav、section；
6、表单控件：calendar、date、time、url、search；
7、新的技术

Q2
IE8/IE7/IE6支持通过document.createElement方法产生的标签；

```
<!--[if lt IE 9]>

<script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
```

```
  <![endif]-->
```

19、iframe的作用？

用法

1、iframe是用来在网页中插入第三方页面，早期的页面使用iframe主要是用于导航栏这种很多页面都相同的部分，这样在切换页面的时候避免重复下载。

优点

1、便于修改，模拟分离，像一些信息管理系统会用到。

2、但现在基本不推荐使用。除非特殊需要，一般不推荐使用。

缺点

1、iframe的创建比一般的DOM元素慢了1-2个数量级

2、iframe标签会阻塞页面的的加载，如果页面的onload事件不能及时触发，会让用户觉得网页加载很慢，用户体验不好，在Safari和Chrome中可以通过js动态设置iframe的src属性来避免阻塞。

3、iframe对于SEO不友好，替换方案一般就是动态语言的Incude机制和ajax动态填充内容等。

20、box-sizing、transition、translate分别是什么？

1、box-sizing:用来指定模型的大小的计算方式。主要分为border-box(从边框固定盒子大小)、content-box(从内容固定盒子大小)两种计算方式。

2、transition:当前元素只要有"属性"发生变化时，可以平滑的进行过渡。通过transition-propety设置过渡属性；transition-duration设置过渡时间；transition-timing-function设置过渡速度；transition-delay设置过渡延时。

3、translate：通过移动改变元素的位置；有x,y,z三个属性

21、选择器优先级是怎样的？

1、 ！important>行内样式>id选择器>类选择器>标签选择器>通配符>继承

2、权重算法：(0,0,0,0)==》第一个0对应的是important的个数，第二个0对应的是id选择器的个数，第三个0对应的类选择器的个数，第四个0对应的是标签选择器的个数，就是当前选择器的权重

3、比较：先从第一个0开始比较，如果第一个0大，那么说明这个选择器的权重高，如果第一个相同，比较第二个，依次类推。

22、有一个导航栏在Chrome里面样式完好？在IE里文字都聚到一起了，是哪个兼容性问题？

用了display：flex属性，在IE10以下都是无效的。

23.CSS实现垂直水平居中

HTML结构：

```
<div class="wrapper">
    <div class="content"></div>
</div>
```

CSS：

```
.wrapper{position:relative;}
.content{
    background-color:#6699FF;
    width:200px;
    height:200px;
    position: absolute;         //父元素需要相对定位
    top: 50%;
    left: 50%;
    margin-top:-100px ;    //二分之一的height，width
    margin-left: -100px;
}
```

方法二

```
.content{
        position:absolute;
        left:50%;
        top:50%;
        width:200px;
        height:200px;
        background:red;
        -webkit-transform:translate(-50%,-50%);
        -moz-transform:translate(-50%,-50%);
        -o-transform:translate(-50%,-50%);
        -ms-transform:translate(-50%,-50%);
        transform:translate(-50%,-50%);
    }
```

24.display有哪些值？说明它们的作用。
block 块类型。默认宽度为父元素宽度，可设置宽高，换行显示。
none 缺省值。像行内元素类型一样显示。
inline 行内元素类型。默认宽度为内容宽度，不可设置宽高，同行显示。
inline-block 默认宽度为内容宽度，可以设置宽高，同行显示。
list-item 像块类型元素一样显示，并添加样式列表标记。
table 此元素会作为块级表格来显示。
inherit 规定应该从父元素继承display属性的值。
25.行内元素有哪些？块级元素有哪些？css的盒模型？
块级元素：div ,p,h1,form,ul,li
行内元素：span,a,label,input,img,strong,em;
css盒模型：内容，border,margin,padding;
26、写一下audio标签中，如何实现音乐的暂停和播放
play()开始,pause()暂停
26（1）、写出video标签中预加载视频用到的属性是什么
preload
27.css选择符有哪些？哪些属性可以继承？优先级算法如何计算？内联和important哪个优先级高？
标签选择符；类选择符；id选择符
id>class>标签选择
important优先级高
28.css清除浮动的几种方法？
使用带clear属性的空标签；
使用css的overflow属性；
使用css的：after伪元素；
同时为了兼容 IE6，7 同样需要配合zoom使用

```
.clear:after{content:'';display:block;clear:both;height:0;overflow:hidden;visibility:hidden;}
     .clear{zoom:1;}
```

使用邻接元素处理；
父级设置成inline-block；
br清浮动
以浮制浮（父级同时浮动）
给浮动元素父级设置高度
给父元素添加overflow：hidden 清除浮动方法；
问题：需要配合 宽度 或者 zoom 兼容IE6 IE7；

```
     overflow: hidden;

     *zoom: 1;
```

29、px、em、rem的区别？
1、px像素。绝对单位，像素px是相对于显示器屏幕分辨率而言的，是一个虚拟单位。是计算机系统的数字化图像长度单位，如果px要换算成物理长度，需要指定精度DPI。
2、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，则相对浏览器的默认字体尺寸。它会继承父级元素的字体大小，因此并不是一个固定的值。
rem是CSS3新增的一个相对单位(root em,根em),使用rem为元素设定字体大小事，仍然是相对大小但相对的只是HTML根元素。
4、区别：IE无法调用那些使用px作为单位的字体大小，而em和rem可以缩放，rem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。目前，除了IE8及更早版本外，所有浏览器已支持rem。
30、CSS3新特性有哪些？
1、颜色：新增RGBA、HSLA模式
2、文字阴影(text-shadow)
3、边框：圆角（border-radius）边框阴影：box-shadow
4、盒子模型：box-sizing
5、背景：background-size设置背景图片的尺寸，background-origin设置背景图片的原点，background-clip设置背景图片的裁剪区域，以“，”分隔可以设置多背景，用于自适应布局
6、渐变：linear-gradient、radial-gradient
7、过渡：transition可实现动画
8、自定义动画
9、在CSS3中唯一引入的伪元素是::selection
10、多媒体查询、多栏布局
11、border-image
12、2D转换：transform:translate(x,y)rotate(x,y)skew(x,y)scale(x,y)
13、3D转换
31、标签上title与alt属性的区别是什么？
Alt当图片不显示时，用文字代表
Title为该属性提供信息
32、描述css reset的作用和用途？
Reset重置浏览器的css默认属性，浏览器的品种不同，样式不同，然后重置，让他们统一。
33、解释css sprites,如何使用？
css 精灵图，把一堆小的图片整合到一张大的图片（png）上，减轻服务器对图片的请求数量。
33（1）、为什么要使用css sprites
css精灵 把一堆小的图片整合到一张大的图片上，减轻服务器对图片的请求数量
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-position"的组合进行背景定位，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，但是如果请求太多会给服务器增加很大的压力。
34、在新窗口打开链接的方法是？
target:_blank
35、合理的页面布局中常听过结构与表现分离，那么结构是什么？表现是什么？
结构是html,表现是css
36、简述对Web语义化的理解？
就是让浏览器更好的读懂你写的代码，在进行HTML结构、表现、行为设计时，尽量使用语义化的标签，使程序代码简洁明了，易于进行web操作和网站SEO，方便团队的一种标准，以图实现一种“无障碍”的web开发。
37、display:none;与visibility:hidden的区别是什么？
display:none;使用该属性后，HTML元素（对象）的宽高，高度等各种属性值都将“丢失”；
visibility:hidden;使用该属性后，HTML元素（对象）仅仅是在视觉上看不见（完全透明），而它所占据的空间位置仍然存在，也即是说它仍然具有高度，宽度等属性值。
38、请用css定义p标签，要求实现以下效果；字体颜色在IE6下为黑色（#000000）；IE7下为红色（#ff0000）;而其他浏览器下为绿色（#00ff00）
```
p{
    color:green;

    *color:blue;
```

```
    _color:black;
}
```
39、前端页面有哪三层构成，分别是什么？作用是什么？
结构层、表示层、行为层
结构层（structural layer）：由HTML或XHTML之类的标记语言负责创建。
表示层（presentation layer）:由css负责创建。
行为层（behaviorlayer）:负责回答“内容应该如何对事件做出反应”这一问题。这是 Javascript 语言和 DOM主宰的领域。
40、实现布局中间自适应宽度，左右两栏固定宽度
```
    <style>
        .box{
            display:flex;
        }
        .left{
            width: 200px;
            height: 600px;
            background:red;
        }
        .right{
            width: 200px;
            height: 600px;
            background:red;
        }
        .center{
            width: 100%;
            height:600px;
            background:green;
        }
    </style>
</head>
<body>
    <div class="box" >
        <div class="left"></div>
        <div class="center"></div>
        <div class="right"></div>
    </div>
```
41、如何在页面上实现一个圆形的可点击区域？
1、map+area或者svg
2、border+radius
3、纯js实现需要求一个点在不在圆上简单算法，获取鼠标坐标等等
42、介绍一下标准css的盒子模型？低版本IE的盒子模型有什么不同的？
1、有两种：IE盒子模型、W3c盒子模型
2、盒模型：内容(content)、填充(padding)、边界(margin)、边框(border)。
3、区别：IE的content部分把border和padding计算了进去
43、你如何对网站的文件和资源进行优化？期待的解决方案包括：
文件合并
文件最小化/文件压缩
使用CDN托管
缓存的使用
44、IE8以下版本的浏览器中盒模型有什么不同？
IE8以下浏览器的盒模型中定义的元素的宽高不包括内边剧和边距
45、写出几种IE6 BUG的解决方法

1、双边距BUG float引起的 使用display

2、3像素问题使用float引用的使用display:inline -3px;
3、超链接hover后点击失效，使用正确的书写顺序 link visited hover active
4、le z-index问题给父级添加position:relative
5、png 透明使用js代码改
6、min-height最小高度 ！important解决
7、select 在ie6下遮盖 使用iframe嵌套
8、为什么没有办法定义1px左右的宽度器（IE6默认的行高造成的，使用over:hidden,zoom:0.08,line-height:1px）
46、css选择符有哪些？哪些属性可以继承？优先级算法如何计算？内联和important哪个优先级高？
css选择符：类选择器、标签选择器、ID选择器、后代选择器（派生选择器）、群组选择器
可以继承：类选择器、标签名选择器、后代选择器（派生选择器）、群组选择器
优先级算法：
标签内直接定义：1000
ID选择器：100
类选择器：1
内联和important中，important优先级高
47、css的基本语句构成是？
选择符、属性、值

Html5 面试题汇总
1.HTML5 为什么只需要写 <!DOCTYPE HTML>？
答案解析：
Html5不基于SGML，因此不需要对DTD进行引用，但是需要DOCTYPE来规范浏览器的行为（让浏览器按照他们应该的方式来运行）而HTML4.01基于SGML，所以需要对DTD进行引用，才能告知浏览器文档所使用的文档类型。

2、行内元素有哪些？块级元素有哪些？空(void)元素有那些？
答案解析：
行内元素：a b span img input select strong
块级元素：div ul ol li dl dt dd h1 h2 h3 h4 p 等
空元素：
 <hr> <img> <link> <meta>

3、页面导入样式时，使用link和@import有什么区别？
答案解析：
1）link属于XHTML标签，而@import是css提供的；
2）页面被加载时，link会同时被加载，而@import引用的css会等到页面被加载完再加载；
3）@import只在IE5以上才能识别，而link是XHTML标签，无兼容问题；
4）link方式的样式的权重高于@import的权重。

4、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？
答案解析：
新特性，新增元素：
1）内容元素：article、footer、header、nav、section
2）表单控件：calendar、date、time、email、url、search
3）控件元素：webworker，websockt，Geolocation
移除元素：
1）显现层元素：basefont，big，center，font，s，strike，tt，u
2）性能较差元素：frame，frameset，noframes
处理兼容问题有两种方式：

1）IE6/IE7/IE8支持通过document方法产生的标签，利用这一特性让这些浏览器支持HTML5新标签。

2）使用是html5shim框架
另外，DOCTYPE声明的方式是区分HTML和HTML5标志的一个重要因素，此外，还可以根据新增的结构，功能元素来加以区分。

5、如何区分 HTML 和 HTML5？
答案解析：
1）在文档类型声明上不同：
HTML是很长的一段代码，很难记住，而HTML5却只有简简单单的声明，方便记忆。
2）在结构语义上不同：
HTML：没有体现结构语义化的标签，通常都是这样来命名的 ，这样表示网站的头部。
HTML5：在语义上却有很大的优势。提供了一些新的标签，比如：<header><article><footer>

6、简述一下你对HTML语义化的理解？
答案解析：
1）用正确的标签做正确的事情；
2）html语义化让页面的内容结构化，结构更清晰，便于对浏览器、搜索引擎解析；
3）即使在没有样式css情况下也以一种文档格式显示，并且是容易阅读的；
4）搜索引擎的爬虫也依赖于HTML标记来确定上下文和各个关键字的权重，利于SEO；
5）使于都源代码的人对网站更容易将网站分块，便于阅读维护理解。

7、HTML5的离线储存怎么使用，工作原理能不能解释一下？
答案解析：
localStorage 长期存储数据，浏览器关闭后数据不丢失；
sessionStorage 数据在浏览器关闭后自动删除。

8、iframe有那些缺点？
答案解析：
1）在网页中使用框架结构最大的弊病是搜索引擎的“蜘蛛”程序无法解读这种页面；
2）框架结构有时会让人感到迷惑，页面很混乱；

9、Doctype作用? 严格模式与混杂模式如何区分？它们有何意义?
答案解析：
1）<!Doctype>声明位于文档中的最前面，处于<html>标签之前。告知浏览器的解析器，用什么文档类型规范来解析这个文档。
2）严格模式的排版和JS运作模式是以该浏览器支持的最高标准运行。
3）在混杂模式中，页面以宽松的向后兼容的方式显示。模拟老式浏览器的行为以防止站点无法工作。
4）DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。

10、常见兼容性问题？
1）png24位的图片在IE6浏览器上出现背景；
解决方案是：做成PNG8；
2）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。
3）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 #box{float:left;width:10px;margin:0 0 0 100px;} 这种情况下IE6会产生200px的距离。
解决方法：加上_display：inline，使浮动忽略
4）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。
5）IE下，even对象有x，y属性，但是没有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。
6）Chrome中文界面下默认会将小于 12px 的文本强制按照 12px 显示

解决方法：可通过加入 CSS 属性 -webkt-text-size-adjust:none;解决

7）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 ；
解决方法：改变CSS属性的排列顺序：L-V-H-A: a:link{ } a:visited{ } a:hover{ } a:active{ }

11、如何实现浏览器内多个标签页之间的通信？
答案解析：
调用localstorge、cookies等本地存储方式

12、webSocket如何兼容低浏览器？
答案解析：
Adobe Flash Socket 、 ActiveX HTMLFile (IE) 、 基于 multipart 编码发送 XHR 、 基于长轮询的 XHR

13、支持HTML5新标签
答案解析：
1）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；
2）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]-->
```

14、如何区分：DOCTYPE 声明\新增的结构元素\功能元素，语义化的理解？
答案解析：
1）用正确的标签做正确的事情；
2）html语义化就是让页面的内容结构化，便于对浏览器、搜索引擎解析；
3）在没有样式 CSS 情况下也以一种文档格式显示，并且是容易阅读的；
4）搜索引擎的爬虫依赖于标记来确定上下文和各个关键字的权重，利用 SEO ；
5）使阅读源代码的人对网站更容易将网站分块，便于阅读维护理解。

15、介绍一下 CSS 的盒子模型？
答案解析：
1）有两种，IE 盒子模型、标准 W3C 盒子模型； IE 的 content 部分包含了 border 和 padding；
2）盒模型：内容（content）、填充（padding）、边界（margin）、边框（border）。

16、CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3 新增伪类有哪些？
答案解析：
1）id 选择器（#myid）
2）类选择器（.myclassname）
3）标签选择器（div，h1，p）
4）相邻选择器（h1 + p）
5）子选择器（ul > li）
6）后代选择器（li a）
7）通配符选择器（* ）
8）属性选择器（ a[rel = "external"]）
9）伪类选择器（a: hover, li: nth - child）

17、可继承的样式： font-size font-family color, UL LI DL DD DT

18、不可继承的样式：border padding margin width height

19、优先级就近原则，同权重情况下样式定义最近者为准

20、载入样式以最后载入的定位为准；

解析答案：优先级为: !important > id > class > tag ; important 比 内联优先级高

21、CSS3新增伪类举例：
答案解析：
p:first-of-type　选择属于其父元素的首个 <p> 元素的每个 <p> 元素；
p:last-of-type　选择属于其父元素的最后 <p> 元素的每个 <p> 元素；
p:only-of-type　选择属于其父元素唯一的 <p> 元素的每个 <p> 元素；
p:only-child　选择属于其父元素的唯一子元素的每个 <p> 元素；
p:nth-child(2)　选择属于其父元素的第二个子元素的每个 <p> 元素；
:enabled　:disabled 控制表单控件的禁用状态；
:checked　　单选框或复选框被选中。

22、如何居中div？如何居中一个浮动元素？
答案解析：
给div 设置一个宽度，然后添加 margin:0 auto 属性；div{width:200px; margin:0 auto; }

23、居中一个浮动元素
答案解析：
确定容器的宽高　宽500 高300的层，设置层的外边距
.div{width:500px;height:300px;margin:-150px 0 0 -250px;position:relative;background:green；
left：50%；头：50%}

24、css3有哪些新特性？
答案解析：
CSS3 实现圆角（border-radius:8px;），阴影（box-shadow:10px），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

25、为什么要初始化 CSS 样式
答案解析：
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。
当然，初始化样式会对 SEO 有一定的影响，但鱼和熊掌不可兼得，但力求影响最小的情况下初始化。
最简单的初始化方法是：*{padding:0;margin:0}（不建议）
淘宝的样式初始化：
```
body, h1, h2, h3, h4, h5, h6, hr, p, blockquote, dl, dt, dd, ul, ol, li, pre, form, fieldset, legend, button, input,
        textarea, th, td { margin:0; padding:0; }
        body, button, input, select, textarea { font:12px/1.5tahoma, arial, \5b8b\4f53; }
        h1, h2, h3, h4, h5, h6{ font-size:100%; }
        address, cite, dfn, em, var { font-style:normal; }
        code, kbd, pre, samp { font-family:couriernew, courier, monospace; }
        small{ font-size:12px; }
        ul, ol { list-style:none; }
        a { text-decoration:none; }
        a:hover { text-decoration:underline; }
        sup { vertical-align:text-top; }
        sub{ vertical-align:text-bottom; }
        legend { color:#000; }
        fieldset, img { border:0; }


        button, input, select, textarea { font-size:100%; } table { border-collapse:collapse;
```

```
border-spacing:0; }
```

26、display:inline-block 什么时候会显示间隙？
答案解析：
移除空格，使用margin 负值、使用 font-size:0、letter-spacing 、word-spacing

27、使用 CSS 预处理器吗？喜欢哪个？
答案解析：SASS

28、什么是盒子模型？

答案解析：

在网页中，一个元素占有空间的大小由几个部分构成，其中包括元素的内容（content），元素的内边距（padding），元素的边框（border），元素的外边距（margin）四个部分。这四个部分占有的空间中，有的部分可以显示相应的内容，而有的部分只用来分隔相邻的区域或区域。4个部分一起构成了css中元素的盒模型。

29、CSS实现垂直水平居中

答案解析：

一道经典的问题，实现方法有很多种，以下是其中一种实现：

HTML结构：

```
<div class="wrapper">
    <div class="content"></div>
</div>
```

CSS：

```
.wrapper{position:relative;}
    .content{
        background-color:#6699FF;
        width:200px;
        height:200px;
```

```
        position: absolute;         //父元素需要相对定位
        top: 50%;
        left: 50%;
        margin-top:-100px ;   //二分之一的height，width
        margin-left: -100px;
    }
```

30、简述一下src与href的区别

答案解析：

href 是指向网络资源所在位置，建立和当前元素（锚点）或当前文档（链接）之间的链接，用于超链接。

src是指向外部资源的位置，指向的内容将会嵌入到文档中当前标签所在位置；在请求src资源时会将其指向的资源下载并应用到文档内，例如js脚本，img图片和frame等元素。当浏览器解析到该元素时，会暂停其他资源的下载和处理，直到将该资源加载、编译、执行完毕，图片和框架等元素也如此，类似于将所指向资源嵌入当前标签内。这也是为什么将js脚本放在底部而不是头部。

31、简述同步和异步的区别

答案解析：

同步是阻塞模式，异步是非阻塞模式。

同步就是指一个进程在执行某个请求的时候，若该请求需要一段时间才能返回信息，那么这个进程将会一直等待下去，直到收到返回信息才继续执行下去；

异步是指进程不需要一直等下去，而是继续执行下面的操作，不管其他进程的状态。当有消息返回时系统会通知进程进行处理，这样可以提高执行的效率。

32、px和em的区别

答案解析：

px和em都是长度单位，区别是，px的值是固定的，指定是多少就是多少，计算比较容易。em得值不是固定的，并且em会继承父级元素的字体大小。

浏览器的默认字体高都是16px。所以未经调整的浏览器都符合: 1em=16px。那么12px=0.75em, 10px=0.625em

33、浏览器的内核分别是什么?

答案解析：

IE: trident内核

Firefox：gecko内核

Safari：webkit内核

Opera：以前是presto内核，Opera现已改用Google Chrome的Blink内核

Chrome：Blink(基于webkit，Google与Opera Software共同开发)

1、新的 HTML5 文档类型和字符集是？
HTML5 文档类型很简单：

```
1
<!doctype html>
```

HTML5 使用 UTF-8 编码示例：

```
1
<meta charset="UTF-8">
```

2、HTML5 中如何嵌入音频？
HTML5 支持 MP3、Wav 和 Ogg 格式的音频，下面是在网页中嵌入音频的简单示例：

```
1
2
3
4
<audio controls>
    <source src="jamshed.mp3" type="audio/mpeg">
    Your browser does'nt support audio embedding feature.
</audio>
```

3、HTML5 中如何嵌入视频？
和音频类似，HTML5 支持 MP4、WebM 和 Ogg 格式的视频，下面是简单示例：

```
1
2
3
4
<video width="450" height="340" controls>
  <source src="jamshed.mp4" type="video/mp4">

   Your browser does'nt support video embedding feature.
```

```
</video>
```
4、除了 audio 和 video，HTML5 还有哪些媒体标签？
HTML5 对于多媒体提供了强有力的支持，除了 audio 和 video 标签外，还支持以下标签：

<embed> 标签定义嵌入的内容，比如插件。

```
1
<embed type=”video/quicktime” src=”Fishing.mov”>
```
<source> 对于定义多个数据源很有用。

```
1
2
3
4
<video width=”450” height=”340” controls>
     <source src=”jamshed.mp4” type=”video/mp4”>
     <source src=”jamshed.ogg” type=”video/ogg”>
</video>
```
<track> 标签为诸如 video 元素之类的媒介规定外部文本轨道。 用于规定字幕文件或其他包含文本的文件，当媒介播放时，这些文件是可见的。

```
1
2
3
4
5
6
<video width=”450” height=”340” controls>
     <source src=”jamshed.mp4” type=”video/mp4”>
     <source src=”jamshed.ogg” type=”video/ogg”>
     <track kind=”subtitles” label=”English” src=”jamshed_en.vtt” srclang=”en” default></track>
      <track kind=”subtitles” label=”Arabic” src=”jamshed_ar.vtt” srclang=”ar”></track>
</video>
```
5、HTML5 Canvas 元素有什么用？
Canvas 元素用于在网页上绘制图形，该元素标签强大之处在于可以直接在 HTML 上进行图形操作，

```
1
2
<canvas id=”canvas1” width=”300” height=”100”>
</canvas>
```
不可思议的 HTML5 Canvas 应用试验
18个基于 HTML5 Canvas 的图表库
20个惊艳的 HTML5 Canvas 应用试验
16款 HTML5 Canvas 开发的网页游戏
推荐14款强大的HTML5素描及绘图工具
6、HTML5 存储类型有什么区别？
HTML5 能够本地存储数据，在之前都是使用 cookies 使用的。HTML5 提供了下面两种本地存储方案：

localStorage 用于持久化的本地存储，数据永远不会过期，关闭浏览器也不会丢失。
sessionStorage 同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此sessionStorage不是一种持久化的本地存储，仅仅是会话级别的存储
7、HTML5 有哪些新增的表单元素？

HTML5 新增了很多表单元素让开发者构建更优秀的 Web 应用程序。

```
datalist
datetime
output
keygen
date
month
week
time
color
number
range
email
url
```

8、HTML5 废弃了哪些 HTML4 标签？

HTML5 废弃了一些过时的，不合理的 HTML 标签：

```
frame
frameset
noframe
applet
big
center
basefront
```

9、HTML5 标准提供了哪些新的 API？

HTML5 提供的应用程序 API 主要有：

```
Media API
Text Track API
Application Cache API
User Interaction
Data Transfer API
Command API
Constraint Validation API
History API
```

10、HTML5 应用程序缓存和浏览器缓存有什么区别？

应用程序缓存是 HTML5 的重要特性之一，提供了离线使用的功能，让应用程序可以获取本地的网站内容，例如 HTML、CSS、图片以及 JavaScript。这个特性可以提高网站性能，它的实现借助于 manifest 文件，如下：

```
1
2
3
4
<!doctype html>
<html manifest=”example.appcache”>
…..
</html>
```

1.基础问题

1. = 和 == 和 === 的区别？

= ： 用于赋值
== ： 用于判断
=== ： 用于判断，必须类型和值同时相等才是等

2.有几种方式清除浮动？

父级元素 overflow: hidden;

浮动元素后面的元素 clear: both;

伪元素选择器: .clear:after { content: ''; display: block; clear: both; }

父级元素设置高度
父级元素 display: table;

3.请解释JS中的闭包？

闭包，一个概念，在一个函数中，返回另一个函数，里面的函数即成为闭包
闭包限制了一定的作用域，保证变量不被释放掉

```
// 示例代码
var add = (function () {
    var count = 0;
    return function () {
        return ++count;
    };
})();

add(); // 1
add(); // 2
add(); // 3
```

4.常见兼容性问题？

1.png24位的图片在iE6浏览器上出现背景，解决方案是做成PNG8
2.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。
3.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 #box{float:left;width:10px;margin:0 0 0 10px;} 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;/所有识别/.background-color:#00deff\9;/IE6、7、8识别/+background-color:#a200ff;/IE6、7识别/_background-color:#1e0bd1;/IE6识别/}
4.IE下,既可以使用获取常规属性的方法来获取自定义属性，又可以使用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.
5.IE下,even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。

6.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. 如果这一属性也不成 使用 transform:scale 来实施
7.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再具有hover和active样式了
解决方法是改变CSS属性的排列顺序:LoVe-HA:a:link{}a:visited{}a:hover{}a:active{}

5.如何实现浏览器内多个标签页之间同域页面的通信?

调用localstorge、cookies等本地存储方式

2.HTML相关问题
1.HTML5新标签？

nav / header / footer / section / article / aside / detail / summary
video / audio / source / canvas / figure / figcaption
time / progress / mark

2.HTML5新表单元素?

url / email / date / datetime / time / week / month / search / range / color

3.HTML5中有哪些新功能？

本地存储： localStorage / sessionStorage

离线缓存： Manifest

新标签： canvas / video / audio

拖放API： drag drop

地理位置： geolocation

浏览器后台执行：web worker

4.请解释Cookie、sessionStroage、localStroage的区别？

Cookie: 存储数据量小，可以设置存储时间，到期自动删除
sessionStroage: 将数据临时存储，离开当前页面就删除
localStroage: 将数据保存到设备上，可以实现永久存储，需要主动删除

5.描述application cache更新的过程?

第一次访问缓存manifest文件里列的文件，之后访问先加载缓存，在后台加载manifest文件按字节对比看是否有变化，如果没变化则说明缓存未失效，否则在后台按列表更新缓存，在下一次刷新页面的时候使用新的资源

6.iframe有那些缺点?

1.iframe会阻塞主页面的Onload事件
2.iframe和主页面共享连接池，而浏览器对相同域的连接有限制，所以会影响页面的并行加载。

使用iframe之前需要考虑这两个缺点。如果需要使用iframe，最好是通过javascript 动态给iframe添加src属性值，

这样可以可以绕开以上两个问题。
3.iframe 会降低页面整体性能， 之前QQ空间的数据是页面每新增一个iframe，性能下降10%。 更多详细内容， 参见知乎问答： http://www.zhihu.com/question/20653055

3.CSS相关问题
1.CSS3新属性？

更多的选择器
弹性布局： flex

帧动画： keyframes / animations

过渡动画： transition

多列布局： column

背景渐变： linear-gradient / radial-gradient

背景属性： background-image / background-size / background-clip / background-origin

边框、阴影： border-image / border-radius / box-shadow

2D、3D变换： transform / transform-origin

用户界面属性添加： box-sizing / outline-offset / cursor更多属性值

2.请简述flex布局？

CSS3中的新增的特性，也称为弹性布局。
使用flex布局，会更加快速与方便，但是有一定的兼容问题（display: -webkit-flex）。
主要含有主轴、交叉轴的概念，通过弹性布局，可以非常方便的设置元素的左对齐、右对齐、上对齐、下对齐、居中对齐等等各种对齐方式，快速实现页面效果。
常用属性有: flex-direction / flex-wrap / justify-content / align-items / align-content / flex-grow / flex-shrink / flex-basis / align-self / 等等

3.rem如何使用？

CSS3中的一个新单位
和em的区别是rem单位只针对HTML根元素，而em是根据父标签的大小进行计算
通常都是给html标签设置好font-size之后，然后其它的单位都换算为rem进行使用。这样，当根标签的font-size发生改变了之后，页面中所有根据rem设置的元素都会发生改变。
html的font-size会根据屏幕的宽度进行调整，可以使用媒体查询，根据屏幕宽度进行font-size的设置。
或者使用JS，根据屏幕宽度，计算出html的font-size的大小

4.Less和Sass分别是什么？

CSS预处理工具
使用编程的思路编写CSS代码，更加高效、快速、便捷
可以使用变量、嵌套规则、混入（函数）、参数混入、运算、颜色功能、作用域等等类似编程中的功能

Sass在市面上有一些成熟的框架，比如说Compass，而且有很多框架也在使用Sass，比如说Foundation

5.什么是样式重置文件（重置样式表）？

reset.css
由于各个浏览器对标签的默认样式支持方式不同，所以样子都不太一样。为了实现不同浏览器效果一致，开发者会主动清掉默认样式，这些代码就放到reset.css文件中，叫样式重置文件（重置样式表）。

6.em和rem的区别？

em是相对于父级标签字体大小
rem是CSS3新的单位，相对根标签(HTML)字体大小，通常用来做移动端适配

7.如何做水平垂直居中？

方案一：已知宽高，设置position: absolute;，然后left和top都设置为50%，再根据margin的负值来调正
方案二：类似方案一，最后一步用transform:translate(-50%,-50%);
方案三：绝对定位，top、bottom、left、right都设置为0，设置好宽高，然后margin: auto;
方案四：display:table-cell; + vertical-align:middle;
方案五：使用flex弹性盒模型

8.transition、animation、transform有什么区别?

transition: 处理过渡效果
animation: CSS3动画，和keyframes一起使用
transform: 处理元素的形变，位移、旋转、缩放、倾斜

9.display: inline-block;显示的缝隙如何去掉？

1.移除空格
2.使用margin负值
3.使用font-size:0
4.letter-spacing
5.word-spacing
6.float

10.rgba()和opacity的透明效果有什么不同？

rgba()和opacity都能实现透明效果
但最大的不同是opacity作用于元素，以及元素内的所有内容的透明度
而rgba()只作用于元素的颜色或其背景色。（设置rgba透明的元素的子元素不会继承透明效果！）

11.inline、block、inline-block的区别是什么？

inline元素和其他inline元素在同一行展示，宽度由内容决定，无法设置宽度
block元素在新行开始，默认宽度为容器的宽度，可以设置宽度
inline-block从外面看是inline，里面看是block，可以设置宽度

12.display:none与visibility:hidden的区别是什么？

display: 隐藏对应的元素但不挤占该元素原来的空间
visibility:隐藏对应的元素并且挤占该元素原来的空间

即是，使用CSS display:none; 属性后，HTML元素（对象）的宽度、高度等各种属性值都将“丢失”;而使用

visibility:hidden; 属性后，HTML元素（对象）仅仅是在视觉上看不见（完全透明），而它所占据的空间位置仍然存在

13.知道css有个content属性吗？有什么作用？有什么应用？

css的content属性专门应用在 before/after伪元素上，用于来插入生成内容
最常见的应用是利用伪类清除浮动
一种常见利用伪类清除浮动的代码

```
.clearfix::after {
    content: ".";//这里利用到了content属性
    display: block;
    height: 0;
    visibility: hidden;
    clear: both;
}
.clearfix {
    *zoom: 1;
}
```

14.有几种方式可以对一个DOM设置它的CSS样式？

外部样式表，引入一个外部css文件
内部样式表，将css代码放在 标签内部
内联样式，将css样式直接定义在 HTML 元素内部

15.什么是Css Hack？ie6,7,8的hack分别是什么？

针对不同的浏览器写不同的CSS code的过程，就是CSS hack

```
#test{
    width:300px;
    height:300px;
    background-color:blue; /*firefox*/
    background-color:red\9; /*all ie*/
    background-color:yellow; /*ie8*/
    +background-color:pink; /*ie7*/
    _background-color:orange; /*ie6*/
}
:root #test {
    background-color:purple\9;  /*ie9*/
}
@media all and (min-width:0px) {
    #test {
        background-color: black; /*opera*/
    }
}
@media screen and (-webkit-min-device-pixel-ratio:0) {
    #test {
        background-color: gray; /*chrome and safari*/
    }

}
```

16.CSS中link和@import的区别?

link属于html标签，而@import是CSS中提供的
在页面加载的时候，link会同时被加载，而@import引用的CSS会在页面加载完成后才会加载引用的CSS
@import只有在ie5以上才可以被识别，而link是html标签，不存在浏览器兼容性问题
link引入样式的权重大于@import的引用（@import是将引用的样式导入到当前的页面中）

4.JavaScript相关问题
1.说说你了解的ES6新特性？

let关键字
模板字符串
箭头函数
函数默认值的设置
for..of..
更多内容参照博客：http://es6.ruanyifeng.com/

2.JQ如何扩展插件？

jQuery.extend(): 给JQuery对象本身扩展方法

jQuery.fn.extend(): 给JQ元素扩展方法

3.AJAX兼容问题？

AJAX用于请求网络数据
正常情况下使用XMLHttpRequest对象即可，但是在IE浏览器下需要使用ActiveXObject对象处理兼容问题
如果使用JQ，则直接使用$.get() / $.post() / $.getJSON() / $.ajax() 即可处理网络请求问题

4.JQ和Zepto的区别？

Zepto更加的轻量级，专为移动端开发
Zepto并不是包含JQ所有的功能，只是封装JQ常用的一些方法
Zepto内部划分模块，不同的功能放到了不同的文件中，需要使用的时候引入，否则不引入。
JQ则是所有功能都放到一个文件中。JQ会更加占用项目体积，而Zepto的使用率更高

5.JQ的优势？

代码简介，语法简单
强大的选择器支持
封装了常用功能，例如：slideUp()、$.each()等等
很好的兼容处理
丰富强大的插件库

完善的AJAX

链式语法
完善的文档、开源

6.JavaScript严格模式是什么？

ECMAScript5添加了“严格模式”
这种模式下JavaScript在更严格的条件下运行
主要目的：
1.消除JavaScript语法的一些不合理，不严谨，减少一些怪异行为
2.消除代码运行的一些不安全之处，保证代码运行的安全
3.提高编译器效率，增加运行速度
4.为未来新版JavaScript做好铺垫
注意：
同样的代码，在“严格模式下”，可能会有不一样的运行结果；一些“正常模式”可以运行的语句，在严格模式下将不能运行
“use strict”;，进入严格模式的标志，老版本浏览器会当做一行普通字符串，加以忽略
将“use strict”;放到第一行，则整个脚本都将以严格模式运行。也可以放到函数中的第一行，则只作用当前函数，

```
// 优化写法
(function () {
    "use strict";
    // some code ...
})();
```

严格模式的一些限制：
1.全局变量必须显示声明
2.禁止this指向全局对象(构造函数，必须加new)
3.函数不能有重复的形参名
4.保留字(let、interface、package、private、protected、public、static、yield、implements)

7.JavaScript中常用的设计模式？

1.单例模式
2.原型模式
3.构造函数
4.工厂模式
5.适配器模式
6.策略模式

8.什么是作用域链？

作用域的范围是函数，函数嵌套函数，查找变量从内层函数依次向外层找，最后找不到在window上找

9.解释call、apply、bind的区别?

call和apply都是调用一个函数，并指定this和参数，call和apply第一个参数都是指定的this的值，区别在于call第二个参数开始的参数是替换的参数，而apply的第二个参数是一个数组。
bind是由一个函数创建一个新函数，并绑定this和部分参数，参数形式和call类似

10.描述事件委派(delegate)的原理和优点?

原理：在容器节点上绑定事件，利用冒泡，判断事件是否在匹配指定的选择器的节点上被触发
优点：只用绑定一次，不用对每个目标做绑定，还有对动态插入的节点也生效，无需重新绑定

11.有哪些原生触屏事件？

touchstart，touchmove，touchend，touchcancel

12.为什么老版本的webkit的click事件有300ms延迟？

为了支持双击放大，如果300ms以内有两次点击则触发放大操作，而不是click。
chromium较新版本在没有双击放大的页面去掉了click事件的300ms延迟

13.ajax和jsonp哪个可以跨域，原理是什么？

ajax默认无法跨域，xhr2新增的CORS让ajax也可以跨域，需要输出http头(Access-Control-Allow-Origin)
jsonp可以跨域，原理是script元素的src可以跨域

14.JavaScript基本数据类型？

基本数据类型： number, string, boolean, undefined, null
引用数据类型： Object(Array, Date, RegExp, Function)

15.谈谈This对象的理解?

this是JavaScript的一个关键字，随着函数使用场合不同，this的值会发生变化
但是有一个总原则，那就是this指的是调用函数的那个对象。
this一般情况下：是全局对象Global。
作为方法调用，那么this就是指这个对象

16.new操作符具体干了什么?

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

```
var obj = {};
obj.__proto__ = Base.prototype;
Base.call(obj);
```

5.Other问题
1.AngularJS中与项目压缩的问题？

在写控制器的时候，有一种写法，会在使用压缩工具的时候出问题

```
// 在进行代码压缩的时候，参数$scope/$css等服务名称会被压缩工具替换掉，就造成无法使用的错误
app.controller("homeCtrl", function ($scope, $css) {

});

// 换成如下写法即可：

app.controller('homeCtrl', ['$scope', '$css', function ($scope, $css) {
```

```
}]);
```

2.什么是模块化开发？AMD和CMD是什么？

NodeJS实现了JavaScript语言编写后台，其中使用CommonJS规范实现了模块化开发
模块化开发其实就是让JavaScript文件可以互相引用，每个文档代表一个模块，提高代码的使用频率。
在HTML中引入JS文件也会方便很多。
原生JavaScript中并不支持模块化开发，出现了一些规范，就是所说的AMD和CMD，这两种方式适合使用桌面端，可以进行异步加载，而在NodeJS中使用CommonJS是同步的，所以CommonJS并不适合桌面端
在require.js推广的过程中，产生了一种模块化规范，叫AMD
在sea.js推广的过程中，产生了一种模块化规范，叫CMD
sea.js是淘宝的前端开发工程师，网名叫玉伯的人编写的。
require.js而是国外朋友编写。
sea.js是根据CommonJS规范来编写的，和CommonJS语法类似，而require.js的写法则教特殊一些

3.微信JSSDK是什么？以及使用流程？

如果项目需要使用微信提供的一些功能，例如：分享、定位、扫一扫、支付、等，就需要借助微信JSSDK来实现这些功能
使用流程：
1.将自己项目的域名添加到微信公众账号中，设置为安全域名
2.在代码中引入JSSDK所需的js文件
3.进行配置: `wx.config({debug: true, appId: '', timestamp: '', nonceStr: '', signature: '', jsApiList: []});`
4.调用JSSDK提供的方法进行调用

4.什么是SPA？

Single Page Application 单页应用程序
打开SPA项目的时候，通常只有一个主页面，然后根据点击不同的按钮，切换主页面中显示的内容。
通过点击不同的按钮(a标签)，来修改锚点的值，然后使用路由根据锚点，实现加载不同的页面
不需要进行网页跳转，即可切换页面

5.什么是懒加载？什么是预加载?

懒加载
懒加载又称延迟加载，当图片需要显示到页面上的时候，才开始真正的发起图片的网络请求，否则不请求。
懒加载原理：
\1. 将图片地址不写入src属性，而是写入其它的属性(data-original)中
\2. 页面onload时候，根据图片的offsetTop值，判断哪些图片显示在用户视野范围内，然后把这些图片的地址从data-original中取出，放到src属性中，图片会自动请求
\3. 设置好onscroll函数，监测屏幕的滚动，如果有图片即将进入用户视野，则同样将图片的地址从data-original取出，放到src属性中取出即可

预加载
在页面正式显示之前，先把所需要的图片资源全部加载下来，然后再显示界面，用户就不会看到图片加载的过程，提高了用户体验
预加载原理
1.拿到所有图片的地址，分别创建Image对象，并赋值给src

2.在imgObj.onload方法中，判断是否所有图片都已经加载完毕

3.当所有图片加载完毕之后，调用回调函数，处理页面的显示

6.如何提高网页加载速度？（网站如何优化？）

上线的时候，使用压缩后三方库（带有min的文件：jquery.min.js）
尽可能使用CDN来加载你的三方库
选取图片的时候，能选择jpg就不选择png（只有镂空图才选择png图片）
图片搞定之后，使用网站进行二次压缩（TinyPNG）
减少网路请求，比如使用懒加载等等
优化代码

7.如何解决跨域问题？

由于浏览器同源策略的限制，访问非同源的数据需要进行跨域处理
实现跨域的几种方式：
\1. JSONP
\2. 后台设置访问源限制
\3. 使用后台代理

JSONP
实现跨域问题最常用的一种方式
利用了script标签src属性没有域的限制而完成
需要前端、后台配合完成
JSONP使用步骤：
\1. 动态创建script标签
\2. 将目标数据的地址放到script标签的src属性中，并在请求地址中拼接callback参数并携带回调函数名
\3. 定义回调函数，当数据请求成功之后，回调函数被调用
\4. 数据完成之后，移除script标签
注意：后台返回的数据不是真正的JSON数据，而是一个函数调用，数据为参数的一段字符串。（
callbackFun('jsonString'); ）
如果使用JQ，直接使用$.getJSON()函数即可进行跨域请求，JQ已经对跨域请求做了封装。

后台设置访问源限制：
只需后台做访问源(Access-Control-Allow-Origin)的设置，前端就可以正常访问跨域的数据。
NodeJS设置方式：
response.setHeader("Access-Control-Allow-Origin", "*");
*****号代表了所有域名，设置好之后代表任何域名都可以访问，也可以对域名进行单独控制
此种方式不够安全

使用后台代理：
前端不直接访问跨域的数据，而是交给服务器去做
服务器去请求别的站点的数据，后台服务器是没有域的限制的，所以很方便
当后台获取到数据之后，前端从自己的服务器请求到数据，无需做跨域处理
此种方式，主要内容都在后台，前端进行正常AJAX请求

8.请解释baiduTemplate.js模板引擎？

模板引擎
根据数据，快速生成HTML代码片段，插入到HTML结构中显示
首先编写结构，可以将HTML和JavaScript混写，然后赋值数据，就可以生成HTML代码片段

```
<!-- 模板 -->
<script type="text/template" id="li">
    <%for (var i = 0; i < data.length; i++) {%>
        <li><%= data[i] %></li>
    <%}%>
</script>

// 根据数据，生成HTML代码片段
var data = ['小明', '小红', '小黑'];
var html = baidu.template('li', {data: data});
document.getElementById("result").innerHTML = html;
```

和baiduTemplate相似模板引擎还有很多，还会使用artTemplate.js，语法相似
http://www.jq22.com/jquery-info1097

9.请解释MVC和MVVM设计模式？
10.XSS是什么？

跨站脚本攻击（Cross-site scripting，XSS）
是一种网站应用程序的安全漏洞攻击，是代码注入的一种。允许用户将恶意代码注入到网页上，影响网页。
这类攻击通常包含了HTML以及用户端脚本语言。

11.移动端性能优化？

加载优化
1.减少HTTP请求
2.压缩JS、CSS、HTML静态资源并在服务器设置gzip
3.首屏加载，不超过120k
4.压缩图片
5.避免重定向
6.异步加载第三方资源：async / 动态创建script / defer
7.oneRequest：首次内联CSS，JavaScript，存localStroage，第二次读取localStroage
8.按需加载：滚动加载、点击加载
CSS优化
1.层次不超过3
2.合并CSS规则，合并margin、background等属性
3.移除空的CSS规则
4.去掉0的单位
5.不要声明过多的font-size
脚本执行优化
1.避免iframe、img等src为空
2.图片尽量避免使用DataURI
3.避免重设图片大小
4.点击事件优化

5.注意scroll resize事件绑定时机

渲染优化
1.动画优化：使用CSS3动画、使用requestAnimationFrame替代setTimeout
2.高频事件优化：touch事件、scroll事件
3.图片优化：压缩图片、webp优于jpg、png8优于gif、对图片使用lazyload、避免使用DataURI
4.GPU加速：使用transition触发、tranform触发、opacity
5.DOM层次不宜过多
6.JS主动的释放内存

12.Tween.JS如何使用？

```
// 首先获取所需要的4个数字
var currentStep = 0; // 当前步数
var allStep = 100; // 总步数
var startValue = $(window).scrollTop(); // 开始的值
var changeValue = 0 - startValue; // 改变的值

// 使用定时器进行改变
var timer = setInterval(function () {
    // 当前步数增加
    currentStep++;
    // 判断结束时间
    if (currentStep > allStep) {
        clearInterval(timer);
    }
    // 四个参数（当前步数、开始值、变化值、总步数）
    var v = Tween.Bounce.easeIn(currentStep, startValue, changeValue, allStep);

    // 使用Tween.js实现效果
    $(window).scrollTop(v);
}, 10);
```

13.什么是Cordova？

hybird App开发平台
一个工具，正常编写HTML、CSS、JavaScript代码，即可通过Cordova生成各个平台的Native App
可以通过编写JavaScript代码调用设备的功能
常用命令

```
1.cordova create ProjectPath com.lidaze.www HelloCordova
2.cordova platform add ios
3.cordova platform rm ios
4.cordova build ios
5.cordova emulate ios
6.cordova plugin add cordova-plugin-dialogs
7.cordova plugin remove cordova-plugin-dialogs
```

14.什么是Web App？什么是hybird App？

Web App：移动App，通过编写HTML、CSS、JavaScript实现Native App的功能外观，还是运行在浏览器中。有时会被嵌入到Native App中
hybird App：混合App，通过编写HTML、CSS、JavaScript实现Navtive App的功能和外观，可直接生成各个操作系统的Native App，可直接安装到手机中

15.http状态码有那些？分别代表是什么意思？

100-199 用于指定客户端应相应的某些动作
200-299 用于表示请求成功
300-399 用于已经移动的文件并且常被包含在定位头信息中指定新的地址信息
400-499 用于指出客户端的错误。
400 语义有误，当前请求无法被服务器理解。
401 当前请求需要用户验证
403 服务器已经理解请求，但是拒绝执行它。
500-599 用于支持服务器错误。
503 – 服务不可用

16.你能描述一下渐进增强和优雅降级之间的不同吗?

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。

17.简述一下src与href的区别？

src用于替换当前元素，href用于在当前文档和引用资源之间确立联系。
src是source的缩写，指向外部资源的位置，指向的内容将会嵌入到文档中当前标签所在位置；在请求src资源时会将其指向的资源下载并应用到文档内，例如js脚本，img图片和frame等元素。
``
当浏览器解析到该元素时，会暂停其他资源的下载和处理，直到将该资源加载、编译、执行完毕，图片和框架等元素也如此，类似于将所指向资源嵌入当前标签内。这也是为什么将js脚本放在底部而不是头部。
href是Hypertext Reference的缩写，指向网络资源所在位置，建立和当前元素（锚点）或当前文档（链接）之间的链接，如果我们在文档中添加
``
那么浏览器会识别该文档为css文件，就会并行下载资源并且不会停止对当前文档的处理。这也是为什么建议使用link方式来加载css，而不是使用@import方式。

18.描述下重绘和回流是什么

浏览器在加载□个□□的时候，会绘制□个节点树，□□中的标签等元素都会称为节点树的□部分，当这些元素的规模尺□，布局，隐藏等改变□需要重新
构建。这就称之为回流。每个□□□少需要□次回流，就是在□□第□次加载的时候。在回流的时候，浏览器会使渲染树中受到影响的部分失效，并重新构造这

部分渲染树，完成回流后，浏览器会重新绘制受影响的部分到屏幕中，该过程称为重绘。

例如：当□□中的部分或者全部元素改变宽度和□度、或者位置发□变化、删除或者增加某个或者某些元素时、某个元素隐藏或者显示时，这时□□就需要重新
加载了，这就叫做回流。
当□□中元素的可□性发□变化时，□如：背景颜□，□字颜□等等，就会形成重绘。
需要注意的是回流必将引起重绘，□重绘不□定会引起回流

19. 对BFC理解吗？

应该是 Block Formatting Context信息来源□络，什么是Block Formatting Context？当涉及到可视化布局的时候，Block Formatting Contex
t提供了□个环境，HTML元素在这个环境中按照□定规则进□布局。□个环境中的元素不会影响到其它环境中的布局。为了让我们有个感性的认识，举个不太合
适的例□。你可以把□个□□想象成□的集装箱，这个集装箱□装的货物就是HTML元素。在现实□活中为了避免不同□的货物相互混淆，都是把货物打好包装
再装□集装箱，这样的话□论你包装□□的货物怎么摆放，都不会影响到其他□的货物。那么这个包装就可以被想象成Block Formatting Context。怎样
才能形成Block Formatting Context当□个HTML元素满□下□条件的任何□点，都可以产□Block Formatting Context
•float的值不为none。
•overflow的值不为visible。
•display的值为table-cell, table-caption, inline-block中的任何□个。
•position的值不为relative和static

20.请用js实现一个继承？

```
// □类
function Person(name, age) {
this.name = name;
this.age = age;
}
Person.prototype.say = function () {
console.log('my name is ' + this.name);
};
// □类
function Student(name, age, gender) {
Person.apply(this, arguments);
this.gender = gender;
}
Student.prototype = new Person;
Student.prototype.constructor = Student;
```

21.谈谈你对闭包的理解

闭包，□个概念，在□个函数中，返回另□个函数，□□的函数即成为闭包 闭包限制了□定的作□域，保证变量不被释放掉

```
// 示例代码
var add = (function () {
 var count = 0;
 return function () {
 return ++count;
 };
})();
```

```
add(); // 1
add(); // 2
add(); // 3
```

22.JS做动画的方式有哪些？

使□setInterval/setTimeout类的定时期，执□元素的状态改变
使□jQuery的animate/show/fade/slide等动画□法
使□requestAnimationFrame□法替代旧的定时器□法

5.非技术问题
1.使用过哪些开发工具？
编辑器:

Sublime Text
Atom (GitHub)
VS Code(微软)

IDE（集成开发环境）:

HBuilder(国内)
WebStorm(国外）

2.前公司技术团队多少人？H5开发有几个？
3.谈谈以前端角度出发做好SEO需要考虑什么？

了解搜索引擎如何抓取网页和如何索引网页
你需要知道一些搜索引擎的基本工作原理，各个搜索引擎之间的区别，搜索机器人（SErobot或叫webcrawler）如何进行工作，搜索引擎如何对搜索结果进行排序等等。
Meta标签优化
主要包括主题（Title），网站描述(Description)，和关键词（Keywords）。还有一些其它的隐藏文字比如Author（作者），Category（目录），Language（编码语种）等。
如何选取关键词并在网页中放置关键词
搜索就得用关键词。关键词分析和选择是SEO最重要的工作之一。首先要给网站确定主关键词（一般在5个上下），然后针对这些关键词进行优化，包括关键词密度（Density），相关度（Relavancy），突出性（Prominency）等等。
了解主要的搜索引擎
虽然搜索引擎有很多，但是对网站流量起决定作用的就那么几个。比如英文的主要有Google，Yahoo，Bing等；中文的有百度，搜狗，有道等。不同的搜索引擎对页面的抓取和索引、排序的规则都不一样。还要了解各搜索门户和搜索引擎之间的关系，比如AOL网页搜索用的是Google的搜索技术，MSN用的是Bing的技术。
主要的互联网目录 OpenDirectory自身不是搜索引擎，而是一个大型的网站目录，他和搜索引擎的主要区别是网站内容的收集方式不同。目录是人工编辑的，主要收录网站主页；搜索引擎是自动收集的，除了主页外还抓取大量的内容页面。
按点击付费的搜索引擎
搜索引擎也需要生存，随着互联网商务的越来越成熟，收费的搜索引擎也开始大行其道。最典型的有Overture和百度，当然也包括Google的广告项目GoogleAdwords。越来越多的人通过搜索引擎的点击广告来定位商业网站，这里面也大有优化和排名的学问，你得学会用最少的广告投入获得最多的点击。
搜索引擎登录
网站做完了以后，别躺在那里等着客人从天而降。要让别人找到你，最简单的办法就是将网站提交（submit）到搜索引擎。如果你的是商业网站，主要的搜索引擎和目录都会要求你付费来获得收录（比如Yahoo要299美元），但是好消息是（至少到目前为止）最大的搜索引擎Google目前还是免费，而且它主宰着60%以上的搜索市场。

链接交换和链接广泛度（LinkPopularity）

网页内容都是以超文本（Hypertext）的方式来互相链接的，网站之间也是如此。除了搜索引擎以外，人们也每天通过不同网站之间的链接来Surfing（“冲浪”）。其它网站到你的网站的链接越多，你也就会获得更多的访问量。更重要的是，你的网站的外部链接数越多，会被搜索引擎认为它的重要性越大，从而给你更高的排名。
合理的标签使用

---

1. javascript的typeof返回哪些数据类型.
答案：string,boolean,number,undefined,function,object
2. 例举3种强制类型转换和2种隐式类型转换?
答案：强制（parseInt,parseFloat,number）
隐式（== ===）
3. split() join() 的区别
答案：前者是将字符串切割成数组的形式，后者是将数组转换成字符串
4. 数组方法pop() push() unshift() shift()
答案：push()尾部添加 pop()尾部删除
unshift()头部添加 shift()头部删除
5. IE和标准下有哪些兼容性的写法
答案：
```
var ev = ev || window.event
document.documentElement.clientWidth || document.body.clientWidth
Var target = ev.srcElement||ev.target
```

6. ajax请求的时候get 和post方式的区别
答案：
一个在url后面 ，一个放在虚拟载体里面
get有大小限制(只能提交少量参数)
安全问题
应用不同 ，请求数据和提交数据
7. call和apply的区别
答案：
```
Object.call(this,obj1,obj2,obj3)
Object.apply(this,arguments)
```
8. ajax请求时，如何解析json数据
答案：使用JSON.parse
9. 事件委托是什么
答案: 利用事件冒泡的原理，让自己的所触发的事件，让他的父元素代替执行！
10. 闭包是什么，有什么特性，对页面有什么影响
答案：闭包就是能够读取其他函数内部变量的函数,使得函数不被GC回收，如果过多使用闭包，容易导致内存泄露
11. 如何阻止事件冒泡
答案：ie:阻止冒泡ev.cancelBubble = true;非IE ev.stopPropagation();
12. 如何阻止默认事件
答案：(1)return false；(2) ev.preventDefault();
13. 添加 删除 替换 插入到某个接点的方法
答案：
1）创建新节点
```
createElement()   //创建一个具体的元素
createTextNode()   //创建一个文本节点
```
2）添加、移除、替换、插入
```
appendChild()      //添加
removeChild()      //移除
```

```
replaceChild()      //替换
insertBefore()      //插入
```
3）查找
```
getElementsByTagName()    //通过标签名称
getElementsByName()     //通过元素的Name属性的值
getElementById()        //通过元素Id，唯一性
```
14. 解释jsonp的原理，以及为什么不是真正的ajax
答案：动态创建script标签，回调函数
Ajax是页面无刷新请求数据操作
15. document load 和document ready的区别
答案：document.onload 是在结构和样式,外部js以及图片加载完才执行js
document.ready是dom树创建完成就执行的方法，原生种没有这个方法，jquery中有 $().ready(function)
16. ”==”和“===”的不同
答案：前者会自动转换类型,再判断是否相等
后者不会自动类型转换，直接去比较
17. 函数声明与函数表达式的区别？
在Javscript中，解析器在向执行环境中加载数据时，对函数声明和函数表达式并非是一视同仁的，解析器会率先读取函数声明，并使其在执行任何代码之前可用（可以访问），至于函数表达式，则必须等到解析器执行到它所在的代码行，才会真正被解析执行。
18. 对作用域上下文和this的理解，看下列代码：
```
var User = {
 count: 1,
 getCount: function() {
  return this.count;
 }
};
console.log(User.getCount()); // what?
var func = User.getCount;
console.log(func()); // what?
```
问两处console输出什么？为什么？
答案:是1和undefined。
    func是在window的上下文中被执行的，所以不会访问到count属性。

19. 看下面代码，给出输出结果。
```
for(var i = 1; i <= 3; i++){  //建议使用let 可正常输出i的值
  setTimeout(function(){
      console.log(i);
  },0);
};
```
答案：4 4 4。
原因：Javascript事件处理器在线程空闲之前不会运行。

20. 当一个DOM节点被点击时候，我们希望能够执行一个函数，应该怎么做?
```
box.onlick= function(){}
box.addEventListener("click",function(){},false);
<button onclick="xxx()"></button>
```
21. Javascript的事件流模型都有什么?
“事件冒泡”：事件开始由最具体的元素接受，然后逐级向上传播
“事件捕捉”：事件由最不具体的节点先接收，然后逐级向下，一直到最具体的
“DOM事件流”：三个阶段：事件捕捉，目标阶段，事件冒泡
22. 看下列代码,输出什么?解释原因。
```
var a = null;

alert(typeof a);
```

答案：object
解释：null是一个只有一个值的数据类型，这个值就是null。表示一个空指针对象，所以用typeof检测会返回”object”。

23. 判断字符串以字母开头，后面可以是数字，下划线，字母，长度为6-30

```
var reg=/^[a-zA-Z]\w{5,29}$/;
```

24. 回答以下代码，alert的值分别是多少？

```
<script>
     var a = 100;
     function test(){
        alert(a);
     a = 10;  //去掉了var 就变成定义了全局变量了
        alert(a);
}
test();
alert(a);
</script>
```

正确答案是： 100， 10， 10

25. javaScript的2种变量范围有什么不同？
全局变量：当前页面内有效
局部变量：函数方法内有效
26. null和undefined的区别？
null是一个表示"无"的对象，转为数值时为0；undefined是一个表示"无"的原始值，转为数值时为NaN。
当声明的变量还未被初始化时，变量的默认值为undefined。 null用来表示尚未存在的对象
undefined表示"缺少值"，就是此处应该有一个值，但是还没有定义。典型用法是：
（1）变量被声明了，但没有赋值时，就等于undefined。
（2）调用函数时，应该提供的参数没有提供，该参数等于undefined。
（3）对象没有赋值的属性，该属性的值为undefined。
（4）函数没有返回值时，默认返回undefined。
null表示"没有对象"，即该处不应该有值。典型用法是：
（1） 作为函数的参数，表示该函数的参数不是对象。
（2） 作为对象原型链的终点。
27. new操作符具体干了什么呢?
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 。
28. js延迟加载的方式有哪些？
defer和async、动态创建DOM方式（创建script，插入到DOM中，加载完毕后callBack）、按需异步载入js
29. Flash、Ajax各自的优缺点，在使用中如何取舍？
Flash ajax对比
(1)Flash适合处理多媒体、矢量图形、访问机器；对CSS、处理文本上不足，不容易被搜索。
(2)ajax对CSS、文本支持很好，支持搜索；多媒体、矢量图形、机器访问不足。
共同点：与服务器的无刷新传递消息、用户离线和在线状态、操作DOM

30. 写一个获取非行间样式的函数

```
function getStyle(obj,attr) {
if(obj.currentStyle) {
return obj.currentStyle[attr];
}else{
getComputedStyle(obi,false)[attr]

}
```

```
}
```

31. 希望获取到页面中所有的checkbox怎么做？(不使用第三方框架)

```
var inputs = document.getElementsByTagName("input");//获取所有的input标签对象
var checkboxArray = [];//初始化空数组，用来存放checkbox对象。
for(var i=0;i<inputs.length;i++){
  var obj = inputs[i];
  if(obj.type=='checkbox'){
     checkboxArray.push(obj);
  }
}
```

32. 写一个function，清除字符串前后的空格。（兼容所有浏览器）

```
String.prototype.trim= function(){

return this.replace(/^\s+/,"").replace(/\s+$/,"");

}
```

33. javascript语言特性中，有很多方面和我们接触的其他编程语言不太一样,请举例

javascript语言实现继承机制的核心就是 1 （原型），而不是Java语言那样的类式继承。Javascript解析引擎在读取一个Object的属性的值时，会沿着 2 (原型链)向上寻找，如果最终没有找到，则该属性值为 3 undefined；如果最终找到该属性的值，则返回结果。与这个过程不同的是，当javascript解析引擎执行“给一个Object的某个属性赋值”的时候，如果当前Object存在该属性，则改写该属性的值，如果当前的Object本身并不存在该属性，则赋值该属性的值。

34. Cookie在客户机上是如何存储的

Cookies就是服务器暂存放在你的电脑里的文本文件，好让服务器用来辨认你的计算机。当你在浏览网站的时候，Web服务器会先送一小小资料放在你的计算机上，Cookies 会帮你在网站上所打的文字或是一些选择都记录下来。当下次你再访问同一个网站，Web服务器会先看看有没有它上次留下的Cookies资料，有的话，就会依据Cookie里的内容来判断使用者，送出特定的网页内容给你。

35. 如何获取javascript三个数中的最大值和最小值？

```
Math.max(a,b,c);//最大值
Math.min(a,b,c)//最小值
```

36. javascript是面向对象的，怎么体现javascript的继承关系？

使用prototype原型来实现。

37. .form中的input可以设置为readonly和disable，请问2者有什么区别？

readonly不可编辑，但可以选择和复制；值可以传递到后台

disabled不能编辑，不能复制，不能选择；值不可以传递到后台

38. 列举javaScript的3种主要数据类型，2种复合数据类型和2种特殊数据类型。

主要数据类型：string, boolean, number

复合数据类型：function, object

特殊类型：undefined，null

39. 程序中捕获异常的方法？

```
try{

}catch(e){

}finally{

}
```

40. Ajax原理

(1)创建对象

```
var xhr = new XMLHttpRequest();
```

(2)打开请求

```
xhr.open('GET', 'example.txt', true);
```

(3)发送请求

```
xhr.send(); 发送请求到服务器
```

(4)接收响应

```
xhr.onreadystatechange =function(){}
```

(1)当readystate值从一个值变为另一个值时，都会触发readystatechange事件。

(2)当readystate==4时，表示已经接收到全部响应数据。

(3)当status ==200时，表示服务器成功返回页面和数据。

(4)如果(2)和(3)内容同时满足，则可以通过xhr.responseText，获得服务器返回的内容。

41. 解释什么是Json:
(1)JSON 是一种轻量级的数据交换格式。
(2)JSON 独立于语言和平台，JSON 解析器和 JSON 库支持许多不同的编程语言。
(3)JSON的语法表示三种类型值，简单值(字符串，数值，布尔值，null),数组，对象
42. js中的3种弹出式消息提醒（警告窗口，确认窗口，信息输入窗口）的命令式什么？
alert
confirm
prompt
43. 以下代码执行结果

```
var uname = 'jack'
function change() {
    alert(uname) // ?
    var uname = 'lily'
    alert(uname)  //?
}
change()
```

分别alert出 undefined，lily，（变量声明提前问题）

44. 浏览器的滚动距离：
可视区域距离页面顶部的距离
scrollTop=document.documentElement.scrollTop||document.body.scrollTop
45. 可视区的大小：
(1)innerXXX（不兼容ie）
window.innerHeight 可视区高度，包含滚动条宽度
window.innerWidth 可视区宽度，包含滚动条宽度
(2)document.documentElement.clientXXX(兼容ie)
document.documentElement.clientWidth可视区宽度，不包含滚动条宽度
document.documentElement.clientHeight可视区高度，不包含滚动条宽度

46. 节点的种类有几种，分别是什么？

(1)元素节点：nodeType ===1;
(2)文本节点：nodeType ===3;
(3)属性节点：nodeType ===2;
47. innerHTML和outerHTML的区别
innerHTML(元素内包含的内容）
outerHTML(自己以及元素内的内容）
48. offsetWidth offsetHeight和clientWidth clientHeight的区别
(1)offsetWidth （content宽度+padding宽度+border宽度）
(2)offsetHeight（content高度+padding高度+border高度）
(3)clientWidth（content宽度+padding宽度）
(4)clientHeight（content高度+padding高度）
49. 闭包的好处
(1)希望一个变量长期驻扎在内存当中(不被垃圾回收机制回收)
(2)避免全局变量的污染
(3)私有成员的存在
(4)安全性提高
50. 冒泡排序算法
冒泡排序

```
var array = [5, 4, 3, 2, 1];
var temp = 0;
for (var i = 0; i <array.length; i++){
for (var j = 0; j <array.length - i; j++){
if (array[j] > array[j + 1]){
temp = array[j + 1];
array[j + 1] = array[j];
array[j] = temp;
 }
}
```

51、js 实现一个函数对javascript中json 对象进行克隆

```
var oldObject ="sdf";
   var newObject = JSON.parse(JSON.stringify(oldObject));
   console.log(newObject);
```

或者

```
    var a = 'dddd';
    function cp(a){return JSON.parse(JSON.stringify(a))}
    console.log(cp(a));
```

52、js 实现 ajax 请求或者submit请求时 锁屏功能以及开锁功能（请求时界面Loading以及元素不能点击，请求完成即消除Loading）

```
function(url, fn) {
var obj = new XMLHttpRequest(); // XMLHttpRequest对象用于在后台与服务器交换数据
obj.open('GET', url, true);
obj.onreadystatechange = function() {
if(obj.readyState == 4 && obj.status == 200||obj.status == 304) {
               loading.style.display = "none"

            } else {

               alert("不能点击,哈哈哈!");

            }
```

```
   };

obj.send(null);
}
```

53、js 实现一个函数 获得url参数的值

```
function getQueryString(name) {
  var reg = new RegExp("(^|&)" + name + "=([^&]*)(&|$)", "i");
  var r = window.location.search.substr(1).match(reg);
  if (r != null) return unescape(r[2]); return null;
}
```

54、请用js计算1-10000中出现的0 的次数

```
new Array(10000).fill('').map((_, index) => index + 1).filter(item =>
/0/.test(item)).reduce((count, item) => { return count + (String(item).match(/0/g) ||
[]).length}, 0)
```

55、写一个function，清除字符串前后的空格。（兼容所有浏览器）

```
function trim(str) {
    if (str & typeof str === "string") {
        return str.replace(/(^s*)|(s*)$/g,""); //去除前后空白符
    }
}
```

56、降维数组

```
var arr=[[1,2],[3,4]];
function Jw(obj){
     return Array.prototype.concat.apply([],obj);
}
Jw(arr);
```

57、将url的查询参数解析成字典对象

```
...
function getQueryObject(url) {
url = url == null ? window.location.href : url;
var search = url.substring(url.lastIndexOf("?") + 1);
var obj = {};
var reg = /([?&=]+)=([?&=]*)/g;
search.replace(reg, function (rs, 2) {
var name = decodeURIComponent(2);
val = String(val);
obj[name] = val;
return rs;
});
return obj;
}
...
```

58、判断一个字符串中出现次数最多的字符，统计这个次数

```
...
var str = 'asdfssaaasasasasaa';
var json = {};
for (var i = 0; i < str.length; i++) {
if(!json[str.charAt(i)]){

json[str.charAt(i)] = 1;
```

```
}else{
json[str.charAt(i)]++;
}
};
var iMax = 0;
var iIndex = '';
for(var i in json){
if(json[i]>iMax){
iMax = json[i];
iIndex = i;
}
}
alert('出现次数最多的是:'+iIndex+'出现'+iMax+'次');
...
59、编写一个方法 求一个字符串的字节长度;
...
//假设一个中文占两个字节
var str = '22两是';
alert(getStrlen(str))
function getStrlen(str){
var json = {len:0};
var re = /[\u4e00-\u9fa5]/;
for (var i = 0; i < str.length; i++) {
if(re.test(str.charAt(i))){
json['len']++;
}
};
return json['len']+str.length;
}
...
60、编写一个方法 去掉一个数组的重复元素
...
var arr = [1,2,3,1,43,12,12,1];
var json = {};
var arr2 = [];
for (var i = 0; i < arr.length; i++) {
if(!json[arr[i]]){
json[arr[i]] = true;
}else{
json[arr[i]] = false;
}
    if(json[arr[i]]){
            arr2.push(arr[i]);
    }

};
for (var i = 0; i < arr.length; i++) {
if(!aa(arr[i], arr2)){
arr2.push(arr[i])
}
};
function aa(obj, arr){

for (var i = 0; i < arr.length; i++) {
```

```
if(arr[i] == obj) return true;
else return false;
};
}
alert(arr2)
```
...
61、写出3个使用this的典型应用
事件： 如onclick this->发生事件的对象
构造函数 this->new 出来的object
call/apply 改变this
62、如何深度克隆
...
```
var arr = [1,2,43];
var json = {a:6,b:4,c:[1,2,3]};
var str = 'sdfsdf';
var json2 = clone(json);
alert(json['c'])
function clone(obj){
var oNew = new obj.constructor(obj.valueOf());
if(obj.constructor == Object){
for(var i in obj){
oNew[i] = obj[i];
if(typeof(oNew[i]) == 'object'){
clone(oNew[i]);
}
}
}
return oNew;
}
```
...
63、JavaScript中如何检测一个变量是一个String类型？请写出函数实现
...
```
typeof(obj) == 'string'
obj.constructor == String;
```
...
64、网页中实现一个计算当年还剩多少时间的倒数计时程序，要求网页上实时动态显示“××年还剩××天××时××分××秒”
...
```
var oDate = new Date();
var oYear = oDate.getFullYear();
var oNewDate = new Date();
oNewDate.setFullYear(oYear, 11, 31, 23, 59, 59);
var iTime = oNewDate.getTime()-oDate.getTime();
var iS = iTime/1000;
var iM = oNewDate.getMonth()-oDate.getMonth();
var iDate =iS
```
...
65、请解释一下什么是语义化的HTML。
内容使用特定标签，通过标签就能大概了解整体页面的布局分布
66、为什么利用多个域名来存储网站资源会更有效？
确保用户在不同地区能用最快的速度打开网站，其中某个域名崩溃用户也能通过其他郁闷访问网站
67、请说出三种减低页面加载时间的方法
1、压缩css、js文件

2、合并js、css文件，减少http请求

3、外部js、css文件放在最底下
4、减少dom操作，尽可能用变量替代不必要的dom操作
68、什么是FOUC？你如何来避免FOUC？
由于css引入使用了@import 或者存在多个style标签以及css文件在页面底部引入使得css文件加载在html之后导致页面闪烁、花屏
用link加载css文件，放在head标签里面
69、文档类型的作用是什么？你知道多少种文档类型？
影响浏览器对html代码的编译渲染
html2.0
xHtml
html5
70、浏览器标准模式和怪异模式之间的区别是什么？
盒模型解释不同
71、闭包
子函数能被外部调用到，则该作用连上的所有变量都会被保存下来。
72、请解释什么是Javascript的模块模式，并举出实用实例。
js模块化mvc（数据层、表现层、控制层）
seajs
命名空间
73、你如何组织自己的代码？是使用模块模式，还是使用经典继承的方法？
对内：模块模式
对外：继承
74、你如何优化自己的代码？
代码重用
避免全局变量（命名空间，封闭空间，模块化mvc..）
拆分函数避免函数过于臃肿
注释
75、你能解释一下JavaScript中的继承是如何工作的吗？
子构造函数中执行父构造函数，并用call\apply改变this
克隆父构造函数原型上的方法
76、请尽可能详尽的解释AJAX的工作原理。
创建ajax对象（XMLHttpRequest/ActiveXObject(Microsoft.XMLHttp)）
判断数据传输方式(GET/POST)
打开链接 open()
发送 send()
当ajax对象完成第四步（onreadystatechange）数据接收完成，判断http响应状态（status）200-300之间或者304（缓存）执行回调函数
77、最简单的一道题
...
```
var a = 2, b = 3;
var c = a+++b; // c = 5
```
...
78、var和function的预解析问题,以及变量和function的先后顺序的问题
...
```
// 以下代码执行输出结果是什么
function b () {
console.log(a);
var a = 10;
function a() {};
a = 100;
console.log(a);
}

b();
```

```
function c () {
    console.log(a);
    function a() {};
    var a = 10;
    a = 100;
    console.log(a);
}
c();

(function d (num) {
    console.log(num);
    var num = 10;
}(100))

(function e (num) {
    console.log(num);
    var num = 10;
    function num () {};
}(100))

(function f (num) {
    function num () {};
    console.log(num);
    var num =10
    console.log(num);
}(100))

//仍然是预解析(在与解析过程中还要考虑一下当前变量的作用于)
function m () {
    console.log(a1); // underfined
    console.log(a2); // underfined
    console.log(b1); // underfined
    console.log(b2); // underfined
    if(false) {
        function b1 (){};
        var a1 = 10;
    }
    if(true) {
        function b2 (){};
        var a2 = 10;
    }
    console.log(a1); // underfined
    console.log(a2); // 10
    console.log(b1); // underfined
    console.log(b2); // function
}
m();

function n() {
    if(2>1) {
        arr = 10;
        brr = 10;

        let arr;
```

```
        var brr;
        console.log(arr);
        console.log(brr);
    }
}
n(); // ReferenceError

...
```

79、dom事件委托什么原理，有什么优缺点

事件委托原理:事件冒泡机制

优点

1.可以大量节省内存占用，减少事件注册。比如ul上代理所有li的click事件就很不错。

2.可以实现当新增子对象时，无需再对其进行事件绑定，对于动态内容部分尤为合适

缺点

事件代理的常用应用应该仅限于上述需求，如果把所有事件都用事件代理，可能会出现事件误判。即本不该被触发的事件被绑定上了事件。

80、http的cache机制，以及200状态下怎么实现 from cache（表示接触最多的就是304的from cache）（用于优化，没有接触过，需要理解）

含义

定义：浏览器缓存（Browser Caching）是为了加速浏览，浏览器在用户磁盘上对最近请求过的文档进行存储，当访问者再次请求这个页面时，浏览器就可以从本地磁盘显示文档，这样就可以加速页面的阅览。

作用

cache的作用：

1、减少延迟，让你的网站更快，提高用户体验。

2、避免网络拥塞，减少请求量，减少输出带宽。

实现手段

Cache-Control中的max-age是实现内容cache的主要手段，共有3种常用策略：max-age和Last-Modified（If-Modified-Since）的组合、仅max-age、max-age和ETag的组合。

对于强制缓存，服务器通知浏览器一个缓存时间，在缓存时间内，下次请求，直接用缓存，不在时间内，执行比较缓存策略。

对于比较缓存，将缓存信息中的Etag和Last-Modified通过请求发送给服务器，由服务器校验，返回304状态码时，浏览器直接使用缓存。

81、一个原型链继承的问题

```
    // 有一个构造函数A，写一个函数B，继承A
    function A (num) {
        this.titileName = num;
    }
    A.prototype = {
        fn1: function () {},
        fn2: function () {}
    }
```

这个问题的关注点是B继承的A的静态属性，同时B的原型链中不存在A实例的titleName属性

82、什么是虚拟dom

React为啥这么大？因为它实现了一个虚拟DOM（Virtual DOM）。虚拟DOM是干什么的？这就要从浏览器本身讲起

如我们所知，在浏览器渲染网页的过程中，加载到HTML文档后，会将文档解析并构建DOM树，然后将其与解析CSS生成的CSSOM树一起结合产生爱的结晶——RenderObject树，然后将RenderObject树渲染成页面（当然中间可能会有一些优化，比如RenderLayer树）。这些过程都存在与渲染引擎之中，渲染引擎在浏览器中是于JavaScript引擎（JavaScriptCore也好V8也好）分离开的，但为了方便JS操作DOM结构，渲染引擎会暴露一些接口供JavaScript调用。由于这两块相互分离，通信是需要付出代价的，因此JavaScript调用DOM提供的接口性能不咋地。各种性能优化的最佳实践也都在尽可能的减少DOM操作次数。

而虚拟DOM干了什么？它直接用JavaScript实现了DOM树（大致上）。组件的HTML结构并不会直接生成DOM，而是映射生

成虚拟的JavaScript DOM结构，React又通过在这个虚拟DOM上实现了一个 diff 算法找出最小变更，再把这些变更写入实际的DOM中。这个虚拟DOM以JS结构的形式存在，计算性能会比较好，而且由于减少了实际DOM操作次数，性能会有较大提升

83、js基础数据类型和引用类型分别是什么？这个前提条件下写一个getType，返回相应的类型

1.基本数据类型（自身不可拆分的）：Undefined、Null、Boolean、Number、String

2.引用数据类型（对象）：Object （Array，Date，RegExp，Function）

ES6基本数据类型多了个symbol 据说这道题刷了百分之二十的人 感谢Abbyshen提出

```
function gettype(nm){
    return Object.prototype.toString.call(nm);
}
```

84、 dom选择器优先级是什么，以及权重值计算（一道老问题了）

1.行内样式 1000

2.id 0100

3.类选择器、伪类选择器、属性选择器[type="text"] 0010

4.标签选择器、伪元素选择器(::first-line) 0001

5.通配符*、子选择器、相邻选择器 0000

85、vue双向数据绑定的原理是什么

首先传输对象的双向数据绑定 Object.defineProperty(target, key, decription),在decription中设置get和set属性（此时应注意description中get和set不能与描述属性共存）

数组的实现与对象不同。

同时运用观察者模式实现wather，用户数据和view视图的更新

86、 react和vue比较来说有什么区别

1 component层面，web component和virtual dom

2 数据绑定（vue双向，react的单向）等好多

3 计算属性 vue 有，提供方便；而 react 不行

4 vue 可以 watch 一个数据项；而 react 不行

5 vue 由于提供的 direct 特别是预置的 directive 因为场景场景开发更容易；react 没有

6 生命周期函数名太长 directive

87、git使用过程中，如果你在开发着业务，突然另一个分支有一个bug要改，你怎么办

```
git stash       //将本次修改存到暂存区（紧急切换分支时）
git stash pop   //将所有暂存区的内容取出来
```

88、 网页布局有哪几种，有什么区别

静态、自适应、流式、响应式四种网页布局

静态布局：意思就是不管浏览器尺寸具体是多少，网页布局就按照当时写代码的布局来布置；

自适应布局：就是说你看到的页面，里面元素的位置会变化而大小不会变化；

流式布局：你看到的页面，元素的大小会变化而位置不会变化—这就导致如果屏幕太大或者太小都会导致元素无法正常显示。

自适应布局：每个屏幕分辨率下面会有一个布局样式，同时位置会变而且大小也会变。

89、执行下面代码

```
var a = {};
var b = {key: 'b'};
var c = {key: 'c'};
var d = [3,5,6];
a[b] = 123;
a[c] = 345;
a[d] = 333;
console.log(a[b]);  // 345
console.log(a[c]);  // 345
console.log(a[d]);  // 333
```

90、

```
    var R = (function() {
        var u = {a:1,b:2};
        var r = {
            fn: function(k) {
                return u[k];
            }
        }
        return r;
    }());
    R.fn('a');  // 1
```

上述代码中如何获取匿名函数中的u

91、不适用循环语句（包括map、forEach方法）实现一个100长度的数组，索引值和值相同的数组

```
[0,1,2,3,4,5........99]
var arr = new Array(100);
//方法1
[...arr.keys()];
//方法二
Array.from(arr.keys());

//方法三
Array.from({length: 100});

// 方法四 借助string
var arr1 = new Array(101);
var str = arr1.join('1,');
str = str.replace/(1\,)/g, function ($0, $1, index) {
    var start = '' + Math.ceil(index/2);
    if(index < str.length - 2) {
        start += ','
    }
    return start;
});
return str.split(',');

// 方法五（函数式，参考网络）
function reduce(arr, val) {
    if(Object.prototype.toString.apply(val)){
        return;
    }
    if(val >= 100) {
        return arr;
    }
    arr.push(val);
    return reduce(arr, val+1);
}
var res = reduce([], 0)
```

92、下面语句执行结果输出

```
var a = function (val, index) {
    console.log(index);

    return {
```

```
        fn: function (name) {
            return a(name, val);
        }
    }
}

var b = a(0); // underfined
b.fn(1); // 0
b.fn(2); // 0
b.fn(3); // 0
```

93、科普

dom节点的根节点是不是body
回答： 不是，dom节点的根节点是html(包含head和body，head中分为meta、title等。body又分为一组)

2）dom元素都会有offsetParent吗
回答： offsetParent属性返回一个对象的引用，这个对象是距离调用offsetParent的元素最近的（在包含层次中最靠近的），并且是已进行过CSS定位的容器元素。 如果这个容器元素未进行CSS定位，则offsetParent属性的取值为根元素(在标准兼容模式下为html元素；在怪异呈现模式下为body元素)的引用。 当容器元素的style.display 被设置为"none"时（译注：IE和Opera除外），offsetParent属性 返回 null。

[1,3,5]转译成字符串是什么
回答： '1,3,5'
调用toString方法，生成该字符串

4）li标签的祖级元素可以为li，父级元素也可以为例
回答： 错误
94、 jsonp原理，jquery是怎么实现的，这样实现有什么好处和坏处
原理
在同源策略下;在某个服务器下的页面是无法获取到该服务器以外的数据的;Jquery中ajax 的核心是通过XmlHttpRequest获取非本页内容，而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
当我们正常地请求一个JSON数据的时候，服务端返回的是一串 JSON类型的数据，而我们使用 JSONP模式来请求数据的时候服务端返回的是一段可执行的 JavaScript代码。因为jsonp 跨域的原理就是用的动态加载 script的src ，所以我们只能把参数通过 url的方式传递，所以jsonp的 type类型只能是get ！

```
$.ajax({
    url: 'http://192.168.1.114/yii/demos/test.php', //不同的域
    type: 'GET', // jsonp模式只有GET 是合法的
    data: {
        'action': 'aaron'
    },
    dataType: 'jsonp', // 数据类型
    jsonp: 'backfunc', // 指定回调函数名，与服务器端接收的一致，并回传回来
})
```

其实jquery 内部会转化成

```
http://192.168.1.114/yii/demos/test.php?
backfunc=jQuery203003857393935322761 5_1402643146875&action=aaron
```

然后动态加载

```
<script type="text/javascript"src="http://192.168.1.114/yii/demos/test.php?backfunc=>
```

然后后端就会执行backfunc(传递参数)，把数据通过实参的形式发送出去。

在jquery 源码中， jsonp的实现方式是动态添加<script>标签来调用服务器提供的 js脚本。jquery 会在window对

象中加载一个全局的函数，当 <script>代码插入时函数执行，执行完毕后就 <script>会被移除。同时jquery还对非跨域的请求进行了优化，如果这个请求是在同一个域名下那么他就会像正常的 Ajax请求一样工作。
95、 http协议属于七层协议中的哪一层，下一层是什么
七层结构：物理层、数据链路层、网络层、传输层、会话层、表示层、应用层
tcp属于传输层；http属于应用层。
表现层
96、 js垃圾回收机制知道哪些，v8引擎使用的哪一种
js的两种回收机制
1 标记清除（mark and sweep）
2 引用计数（reference counting）
javascript与V8引擎
垃圾回收机制的好处和坏处
好处：大幅简化程序的内存管理代码，减轻程序猿负担，并且减少因为长时间运转而带来的内存泄露问题。
坏处：自动回收意味着程序猿无法掌控内存。ECMAScript中没有暴露垃圾回收的借口，我们无法强迫其进行垃圾回收，更加无法干预内存管理。
V8 自动垃圾回收算法
https://segmentfault.com/a/11...
97、 作用域什么时候生成的？
页面加载-->创建window全局对象，并生成全局作用域-->然后生成执行上下文，预解析变量(变量提升)，生成全局变量对象；
$scope
98、 websocket长连接原理是什么
含义
Websocket是一个持久化的协议，相对于HTTP这种非持久的协议来说。
原理
类似长轮循长连接 ； 发送一次请求 ； 源源不断的得到信息
28. http缓存知道哪些
http://blog.csdn.net/yzf91321...
99、 讲一下事件循环机制
执行上下文(Execution context)
函数调用栈(call stack)
队列数据结构(queue)
Promise
https://zhuanlan.zhihu.com/p/...
100、 理解web安全吗？都有哪几种，介绍以及如何预防
1.XSS，也就是跨站脚本注入
攻击方法：
1\. 手动攻击:
编写注入脚本，比如”/><script>alert(document.cookie());</script><!--等，
手动测试目标网站上有的input, textarea等所有可能输入文本信息的区域
2\. 自动攻击
利用工具扫描目标网站所有的网页并自动测试写好的注入脚本，比如：Burpsuite等
防御方法：
1\. 将cookie等敏感信息设置为httponly，禁止Javascript通过document.cookie获得
2\. 对所有的输入做严格的校验尤其是在服务器端，过滤掉任何不合法的输入，比如手机号必须是数字，通常可以采用正则表达式
3\. 净化和过滤掉不必要的html标签，比如：<iframe>, alt,<script> 等
4\. 净化和过滤掉不必要的Javascript的事件标签，比如：onclick, onfocus等
5\. 转义单引号，双引号，尖括号等特殊字符，可以采用htmlencode编码 或者过滤掉这些特殊字符
6\. 设置浏览器的安全设置来防范典型的XSS注入

2.SQL注入

攻击方法：
编写恶意字符串，比如‘ or 1=1--等，
手动测试目标网站上所有涉及数据库操作的地方
防御方法：
1\. 禁止目标网站利用动态拼接字符串的方式访问数据库
2\. 减少不必要的数据库抛出的错误信息
3\. 对数据库的操作赋予严格的权限控制
4\. 净化和过滤掉不必要的SQL保留字，比如：where，or，exec 等
5\. 转义单引号，上引号，尖括号等特殊字符，可以采用htmlencode编码 或者过滤掉这些特殊字符

3.CSRF，也就是跨站请求伪造
就是攻击者冒用用户的名义，向目标站点发送请求
防范方法：
1\. 在客户端进行cookie的hashing，并在服务端进行hash认证
2\. 提交请求是需要填写验证码
3\. 使用One-Time Tokens为不同的表单创建不同的伪随机值

101、 sessionStorage和localstorage能跨域拿到吗？比如我在www.baidu.com设置的值能在m.baidu.com能拿到吗？为什么
localStorage会跟cookie一样受到跨域的限制，会被document.domain影响
102、 localstorage不能手动删除的时候，什么时候过期
除非被清除，否则永久保存 clear()可清楚
sessionStorage 仅在当前会话下有效，关闭页面或浏览器后被清除
103、cookie可以设置什么域？可以设置.com吗
可以通过设置domin来实现
104、登录状态的保存你认为可以保存在sessionstorage或者localstorage或者cookie或者你知道的哪种方式，存在了哪里？？为什么保存在那里
105、 flux -> redux -> mobx 变化的本质是什么
存储结构 将对象加工可观察 函数式 vs 面向对象
https://zhuanlan.zhihu.com/p/...
106、 按需加载路由怎么加载对应的chunk文件的？换句话说浏览器怎么知道什么时候加载这个chunk，以及webpack是怎么识别那个多个经过hash过的chunk文件
107、get和post有什么区别？get可以通过body传递数据吗
把数据放到 body 里面，必须用 POST 方式取，这是 HTTP 协议限制的。
108、右边宽度固定，左边自适应
第一种：

```
<style>
body{
    display: flex;
}
.left{
    background-color: rebeccapurple;
    height: 200px;
    flex: 1;
}
.right{
    background-color: red;
    height: 200px;
    width: 100px;
}

</style>
```

```
<body>
    <div class="left"></div>
    <div class="right"></div>
</body>
```

第二种

```
<style>
    div {
        height: 200px;
    }
    .left {
        float: right;
        width: 200px;
        background-color: rebeccapurple;
    }
    .right {
        margin-right: 200px;
        background-color: red;
    }
</style>
<body>
    <div class="left"></div>
    <div class="right"></div>
</body>
```

109、水平垂直居中

第一种

```
#container{
    position:relative;
}

#center{
    width:100px;
    height:100px;
    position:absolute;
    top:50%;
    left:50%;
    transform: translate(-50%,-50%);
}
```

第二种

```
#container{
    position:relative;
}

#center{
    width:100px;
    height:100px;
    position:absolute;
    top:50%;
    left:50%;
    margin:-50px 0 0 -50px;

}
```

第三种

```
#container{
    position:relative;
}

#center{
    position:absolute;
    margin:auto;
    top:0;
    bottom:0;
    left:0;
    right:0;
}
```

第四种 flex

```
#container{
    display:flex;
    justify-content:center;
    align-items: center;
}
```

109、.四种定位的区别

static 是默认值

relative 相对定位 相对于自身原有位置进行偏移，仍处于标准文档流中

absolute 绝对定位 相对于最近的已定位的祖先元素，有已定位(指position不是static的元素)祖先元素，以最近的祖先元素为参考标准。如果无已定位祖先元素，以body元素为偏移参照基准，完全脱离了标准文档流。

fixed 固定定位的元素会相对于视窗来定位,这意味着即便页面滚动，它还是会停留在相同的位置。一个固定定位元素不会保留它原本在页面应有的空隙。

110、封装一个函数，参数是定时器的时间，.then执行回调函数。

```
function sleep (time) {
  return new Promise((resolve) => setTimeout(resolve, time));
}
```

111、一行代码实现数组去重？

```
[...new Set([1,2,3,1,'a',1,'a'])]
```

112、使用addEventListener点击li弹出内容，并且动态添加li之后有效

```
<ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
    <li>4</li>
</ul>
var ulNode = document.getElementById("ul");
    ulNode.addEventListener('click', function (e) {
        if (e.target && e.target.nodeName.toUpperCase() == "LI") {
            alert(e.target.innerHTML);
        }
    }, false);
```

113、怎么判断两个对象相等

```
JSON.stringify(obj)==JSON.stringify(obj);//true
```

114、Vue router 除了 router-link 怎么实现跳转?
router.go(1)
router.push('/')

115、Vue router 跳转和 location.href 有什么区别？
router 是 hash 改变
location.href 是页面跳转，刷新页面
116、重排和重绘
部分渲染树（或者整个渲染树）需要重新分析并且节点尺寸需要重新计算。这被称为重排。注意这里至少会有一次重排-初始化页面布局。
由于节点的几何属性发生改变或者由于样式发生改变，例如改变元素背景色时，屏幕上的部分内容需要更新。这样的更新被称为重绘。
117、什么情况会触发重排和重绘
添加、删除、更新 DOM 节点
通过 display: none 隐藏一个 DOM 节点-触发重排和重绘
通过 visibility: hidden 隐藏一个 DOM 节点-只触发重绘，因为没有几何变化
移动或者给页面中的 DOM 节点添加动画
添加一个样式表，调整样式属性
用户行为，例如调整窗口大小，改变字号，或者滚动。
118、js bind 实现机制？手写一个 bind 方法？

```
function bind(fn, context){
  return function (){
     return fn.apply(context, arguments);
  }
}
// 柯理化函数思想   感谢pursuitTom的提出
Function.prototype.bind = function (context) {
    var args = Array.prototype.slice.call(arguments, 1);
    var _this = this;
    return function () {
        var thisArgs = [].slice.call(arguments);
        return _this.apply(context, args.concat(thisArgs))
    };
}
// ES6写法 感谢waterc的提出
Function.prototype.bind = function(context, ...res) {
    let self = this
    return function(...arg) {
        return self.apply(context, [...res,...arg])
    }
}
```

119、多个函数

```
var a = (function(){return '1';}, function(){return 1;})();
console.log(typeof a); //number
```

120、proto、prototype、Object.getPrototypeOf()
__proto__是指内部原型，和Object.getPrototypeOf()结果等价

```
function f(){}
f.__proto__ === Object.getPrototypeOf(f); //true
f.prototype === Object.getPrototypeOf(f); //false
```

121、浏览记录前后跳转（尚未试验）

```
<a href="javascript:history.go(-1)">backward</a>
<a href="javascript:history.go(1)">forward</a>
```

122、setTimeout 和 setInterval 细谈

常问的点，前者是在一定时间过后将函数添加至执行队列，执行时间=延迟时间+之前函数代码执行时间+执行函数时间。

后者是不管前一次是否执行完毕，每隔一定时间重复执行，用于精准执行互相没有影响的重复操作。

如果需要控制前后执行顺序，最好使用setTimeout模拟setInterval

```
var time = 400, times = 0, max = 10;
function func(){
  times++;
  if(times < max){
    //code here
    setTimeout(func, time);
  } else {
    console.log("finished");
  }
}
setTimeout(func, time);
```

123、判断多图片加载完毕

注：用jQueryObject.ready()只能判断dom结构加载完毕

好像描述的不是很清楚，这里写一下代码。

方法1：

```
var counter = 0;
var queryInterval = 30; //ms
var pics = document.getElementsByClassName("pics");

function singleQuery(i){
  if(pics[i].complete){
    counter++;
    console.log(i + " is loaded");
  } else {
    setTimeout(singleQuery, queryInterval, i);
  }
}

function allQuery(callback){
  if(counter < pics.length){
    console.log("current number of loaded pics: " + counter);
    setTimeout(allQuery, queryInterval, callback);
  } else {
    console.log("All pics are loaded.");
    callback();
  }
}

for(var i = 0; i < pics.length; i++){
  setTimeout(singleQuery, queryInterval, i);
}

setTimeout(allQuery, queryInterval, callback);
```

主要也是采用setTimeout模拟轮询，判断方式是img标签dom的complete属性（布尔值），缺点是定时器太多。
方法2：

```
var counter = 0, queryInterval = 50;
var pics = document.getElementsByClassName("pics");
for(var i = 0; i < pics.length; i++){
  pics[i].onload = function(){
    counter++;
    console.log(this.id + " is loaded");
  }
}

function queryPictures(callback){
  if(counter < pics.length){
    console.log("current number of loaded pics: " + counter);
    setTimeout(queryPictures, queryInterval, callback);
  } else {
    console.log("All pics are loaded");
    callback();
  }
}

setTimeout(queryPictures, queryInterval, callback);
```

利用onload绑定图片加载成功后的回调，通过计数器判断是否加载完毕。
124、CSS margin重叠问题
块元素在垂直方向上的margin是很奇怪的，会有重叠现象。
如果display都是block，有三种情况：
外间距均为正数，竖直方向上会选择最大的外边距作为间隔
一正一负，间距 = 正 - |负|
两个负，间距 = 0 - 绝对值最大的那个
设置display: inline-block的盒子不会有margin重叠，position: absolute的也不会出现。
125、CSS选择器优先级 && CSS选择器效率
ID > 类 > 标签 > 相邻 > 子选择器 > 后代选择器 > * > 属性 > 伪类
object.propertyIsEnumerable(xxx)
判断对象中是否有xxx属性，并且能通过for in枚举，如Array对象的length是不可枚举的
126、判断数组

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

127、git fetch && git pull
git pull自动完成了fetch最新远程版本，并且和本地进行merge
git fetch获得远程分支，要继续手动merge合并
128、WebSocket
HTML5带来的新协议，通过类似HTTP的请求建立连接。主要目的是可以获取服务端的推送。
原来的方式可能是使用long poll（即不中断连接一直等待数据），或者是ajax轮询的方式（每隔一段时间发送请求，建立连接，询问是否有新的数据）。这两种方式的缺点在于long poll的阻塞，以及ajax轮询的冗余连接。
WebSocket的设计思想有点类似于回调，在发送请求升级服务端的协议并收到确认信息后，服务端一有新的信息/数据就会主动推送给客户端，至于要一次HTTP握手便可以建立持久连接
129、跨域相关

只要协议、域名、端口有不同，则视为不同的域。（域名和域名对应的IP也是跨域）

1.CORS: Cross-Origin Resource Sharing
基于服务器支持的跨域，服务器设置Access-Control-Allow-Origin响应头，浏览器可允许跨域
2.设置domain
能从子域设到主域，如a.b.c.com—>b.c.com—>c.com
具体情况：在页面中用iframe打开了另一个页面（前提：两个页面主域是相同的）
利用frameElement.contentWindow.document.domain设置frame子页面的主域，document.domain设置主页面的主域，之后就能互相获取dom中的数据。
缺点是只能用于不同子域间的交互。
3.例如拥有src属性的img标签，每次改变src属性，都会发起http请求。

```
var img = new Image();
img.onload = function(){
  //code here
};
img.onerror = function(){
  //code here
};
img.src="http://server.com/data?query=3";
```

缺点是只能使用GET请求，不能获取数据，一般用于提交统计信息什么的。
script、link、iframe只有在添加到DOM中才会发起请求
4.HTML5 postMessage
支持IE8+和主流浏览器，写法也简单..

```
//source: http://test.org:4000
//get the window object of target origin
var win = window.open("http://target.com");
//or this, when a frame is used
var win = document.getElementById("targetId").contentWindow;
win.postMessage("data here", "http://target.com/rest");

//target: http://target.com/tiny
function handleMessage(event){
  if(event.orgin!="http://test.org:4000")
    return;
  var data = event.data;
  //code here

  //event.source is window.opener
  event.source.postMessage("response data here", event.origin);
}
window.addEventListener("message", handleMessage, false);
```

5.window.name
即使在页面打开多层iframe后，每个iframe中window.name 属性值都是相同的，以此用作数据传输的工具。
但由于跨域的限制，是无法获取另一个frame中的window.name数据，所以要使用一个同域的代理(proxy.html)：
6. jsonp
目前主流跨域方法
调用其他域的脚本获取数据，前提是另一个域能知道回调函数名，这个可以通过请求发送给目标域。
直接写jQuery封的jsonp

```
$.getJSON("http://target.com/data?callback=callbackFunctionName", function(data){});
```

$.getJSON会把获取的responseText转为json，如果url中有callback参数，数据会以script标签形式获取。
130、闭包相关

什么是闭包

闭包是指有权访问另一个函数作用域中变量的函数

怎么创建闭包

在函数内部嵌套使用函数

```
function fn() {
           for (var i = 0; i < 2; i++) {
               (function () {
                   var variate = i;
                   setTimeout(function () {
                       console.log("setTimeout执行后:"+variate);
                   }, 1000);
               })();//闭包,立即执行函数,匿名函数

           }
          console.log(i);//2
          console.log(variate);//variate is not defined
       }
       fn();
```

为什么用闭包

因为在闭包内部保持了对外部活动对象的访问,但外部的变量却无法直接访问内部,避免了全局污染;
可以当做私有成员,弥补了因js语法带来的面向对象编程的不足;
可以长久的在内存中保存一个自己想要保存的变量.

闭包的缺点

可能导致内存占用过多,因为闭包携带了自身的函数作用域
闭包只能取得外部包含函数中得最后一个值

详见https://segmentfault.com/a/11...
131、a:active 移动端实现
有时候一些按钮的简单点击交互可以通过css伪类来实现；必须点击了更改颜色；松开恢复；IOS手机会出现伪类无效的情况；iOS系统的移动设备中，需要在按钮元素或body/html上绑定一个touchstart事件才能激活:active状态。
document.body.addEventListener('touchstart', function () { //...空函数即可});

132、ios滑动卡顿
-webkit-overflow-scrolling:touch 可能会在IOS系统低的情况出现滚动条；尝试溢出解决
133、forEach和map的区别
相同点

都是循环遍历数组中的每一项
forEach和map方法里每次执行匿名函数都支持3个参数，参数分别是item（当前每一项）、index（索引值）、arr（原数组）
匿名函数中的this都是指向window
只能遍历数组
都有兼容问题

不同点

map速度比foreach快
map会返回一个新数组，不对原数组产生影响,foreach不会产生新数组，
map因为返回数组所以可以链式操作，foreach不能

134、浅拷贝和深拷贝
jQuery.extend第一个参数可以是布尔值，用来设置是否深度拷贝的
```
jQuery.extend(true, { a : { a : "a" } }, { a : { b : "b" } } );
jQuery.extend( { a : { a : "a" } }, { a : { b : "b" } } );
```

最简单的深拷贝
```
aa = JSON.parse( JSON.stringify(a) )
```

浅复制--->就是将一个对象的内存地址的“”编号“”复制给另一个对象。深复制--->实现原理，先新建一个空对象，内存中新开辟一块地址，把被复制对象的所有可枚举的(注意可枚举的对象)属性方法一一复制过来，注意要用递归来复制子对象里面的所有属性和方法，直到子子.....属性为基本数据类型。总结，深复制理解两点，1,新开辟内存地址，2,递归来刨根复制。
135、外边距合并

外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。
合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者。

136、js加载位置区别优缺点
html文件是自上而下的执行方式，但引入的css和javascript的顺序有所不同，css引入执行加载时，程序仍然往下执行，而执行到<script>脚本是则中断线程，待该script脚本执行结束之后程序才继续往下执行。
所以，大部分网上讨论是将script脚本放在<body>之后，那样dom的生成就不会因为长时间执行script脚本而延迟阻塞，加快了页面的加载速度。
但又不能将所有的script放在body之后，因为有一些页面的效果的实现，是需要预先动态的加载一些js脚本。所以这些脚本应该放在<body>之前。
其次，不能将需要访问dom元素的js放在body之前，因为此时还没有开始生成dom，所以在body之前的访问dom元素的js会出错，或者无效
script放置位置的原则“页面效果实现类的js应该放在body之前，动作，交互，事件驱动，需要访问dom属性的js都可以放在body之后

1.请你谈谈Cookie的弊端
cookie

1.IE6或更低版本最多20个cookie
2.IE7和之后的版本最后可以有50个cookie。
3.Firefox最多50个cookie
4.chrome和Safari没有做硬性限制
Opera 会清理近期最少使用的Firefox会随机清理 4096字节，为了兼容性，一般不能超过 IE 提供了一种存储可以持久化用户数据，叫做IE5.0就开始支持。每个数据最多128K，每个域名下最多1M。这个持久化数据放在缓存中，如果缓存没有清理，那么会一直存在。

优点：极高的扩展性和可用性
1.通过良好的编程，控制保存在cookie中的session对象的大小。
2.通过加密和安全传输技术（SSL），减少cookie被破解的可能性。
3.只在cookie中存放不敏感数据，即使被盗也不会有重大损失。
4.控制cookie的生命期，使之不会永远有效。偷盗者很可能拿到一个过期的cookie。
缺点：
1.`Cookie`数量和长度的限制。每个domain最多只能有20条cookie，每个cookie长度不能超过4KB，否则会被截掉。

2.安全性问题。如果cookie被人拦截了，那人就可以取得所有的session信息。即使加密也与事无补，因为拦截者并不需要知道cookie的意义，他只要原样转发cookie就可以达到目的了。

3.有些状态不可能保存在客户端。例如，为了防止重复提交表单，我们需要在服务器端保存一个计数器。如果我们把这个计数器保存在客户端，那么它起不到任何作用。
2.浏览器本地存储
在较高版本的浏览器中，sessionStorage和HTML5中提供了globalStorage。

Web Storage包括了两种存储方式：localStorage。

sessionStorage不是一种持久化的本地存储，仅仅是会话级别的存储。

而 3.web storage和cookie的区别

cookie相似，区别是它是为了更大容量存储设计的。Cookie都会被发送过去，这样无形中浪费了带宽，另外 除此之外，setItem,getItem,removeItem,clear等方法，不像setCookie，getCookie。

但是Cookie的作用是与服务器进行交互，作为Web Storage仅仅是为了在本地“存储”数据而生

浏览器的支持除了UserData其实就是web storage。

sessionStorage都具有相同的操作方法，例如removeItem等

CSS 相关问题
display:none和visibility:hidden的区别？
display:none 隐藏对应的元素，在文档布局中不再给它分配空间，它各边的元素会合拢，
就当他从来不存在。

visibility:hidden 隐藏对应的元素，但是在文档布局中仍保留原来的空间。
CSS中 link 和@import 的区别是？
A：(1) link属于HTML标签，而@import是CSS提供的; (2) 页面被加载的时，link会同时被加载，而@import引用的CSS会等到页面被加载完再加载;(3) import只在IE5以上才能识别，而link是HTML标签，无兼容问题; (4) link方式的样式的权重 高于@import的权重.
position的absolute与fixed共同点与不同点
A：共同点：
1.改变行内元素的呈现方式，display被置为block；2.让元素脱离普通流，不占据空间；3.默认会覆盖到非定位元素上

B不同点：
absolute的”根元素“是可以设置的，而fixed的”根元素“固定为浏览器窗口。当你滚动网页，fixed元素与浏览器窗口之间的距离是不变的。
介绍一下CSS的盒子模型？
1）有两种， IE 盒子模型、标准 W3C 盒子模型；IE的content部分包含了 border 和 pading;

2）盒模型： 内容(content)、填充(padding)、边界(margin)、 边框(border).

图片描述

CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3新增伪类有那些？

* 1.id选择器（ # myid）
2.类选择器（.myclassname）
3.标签选择器（div, h1, p）
4.相邻选择器（h1 + p）
5.子选择器（ul > li）
6.后代选择器（li a）
7.通配符选择器（ * ）
8.属性选择器（a[rel = "external"]）
9.伪类选择器（a: hover, li:nth-child）

* 可继承的样式： font-size font-family color, text-indent;

* 不可继承的样式：border padding margin width height ;

* 优先级就近原则，同权重情况下样式定义最近者为准;

* 载入样式以最后载入的定位为准;

优先级为:

!important > id > class > tag

important 比 内联优先级高,但内联比 id 要高

CSS3新增伪类举例：

p:first-of-type 选择属于其父元素的首个 <p> 元素的每个 <p> 元素。
p:last-of-type 选择属于其父元素的最后 <p> 元素的每个 <p> 元素。
p:only-of-type 选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。
p:only-child 选择属于其父元素的唯一子元素的每个 <p> 元素。
p:nth-child(2) 选择属于其父元素的第二个子元素的每个 <p> 元素。
:enabled :disabled 控制表单控件的禁用状态。
:checked 单选框或复选框被选中。
列出display的值，说明他们的作用。position的值， relative和absolute分别是相对于谁进行定位的？
1.
block 象块类型元素一样显示。
inline 缺省值。象行内元素类型一样显示。
inline-block 象行内元素一样显示，但其内容象块类型元素一样显示。
list-item 象块类型元素一样显示，并添加样式列表标记。

2.
*absolute
生成绝对定位的元素，相对于 static 定位以外的第一个祖先元素进行定位。

*fixed （老IE不支持）
生成绝对定位的元素，相对于浏览器窗口进行定位。

*relative

生成相对定位的元素，相对于其在普通流中的位置进行定位。

  * static 默认值。没有定位，元素出现在正常的流中
  *（忽略 top, bottom, left, right z-index 声明）。

  * inherit 规定从父元素继承 position 属性的值。

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

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。

当然，初始化样式会对SEO有一定的影响，但鱼和熊掌不可兼得，但力求影响最小的情况下初始化。

*最简单的初始化方法就是： * {padding: 0; margin: 0;} （不建议）

淘宝的样式初始化：

```
body, h1, h2, h3, h4, h5, h6, hr, p, blockquote, dl, dt, dd, ul, ol, li, pre, form, fieldset, legend, button, input, textarea, th, td { margin:0; padding:0; }
body, button, input, select, textarea { font:12px/1.5tahoma, arial, \5b8b\4f53; }
h1, h2, h3, h4, h5, h6{ font-size:100%; }
address, cite, dfn, em, var { font-style:normal; }
code, kbd, pre, samp { font-family:couriernew, courier, monospace; }
small{ font-size:12px; }
ul, ol { list-style:none; }
a { text-decoration:none; }
a:hover { text-decoration:underline; }
sup { vertical-align:text-top; }
sub{ vertical-align:text-bottom; }
legend { color:#000; }
fieldset, img { border:0; }
button, input, select, textarea { font-size:100%; }
table { border-collapse:collapse; border-spacing:0; }
```

对BFC规范的理解？

BFC，块级格式化上下文，一个创建了新的BFC的盒子是独立布局的，盒子里面的子元素的样式不会影响到外面的元素。在同一个BFC中的两个毗邻的块级盒在垂直方向（和布局方向有关系）的margin会发生折叠。

（W3C CSS 2.1 规范中的一个概念，它决定了元素如何对其内容进行布局，以及与其他元素的关系和相互作用。）

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`。

html部分

说说你对语义化的理解？

1，去掉或者丢失样式的时候能够让页面呈现出清晰的结构

2，有利于SEO：和搜索引擎建立良好沟通，有助于爬虫抓取更多的有效信息：爬虫依赖于标签来确定上下文和各个关键

字的权重；
3，方便其他设备解析（如屏幕阅读器、盲人阅读器、移动设备）以意义的方式来渲染网页；
4，便于团队开发和维护，语义化更具可读性，是下一步吧网页的重要动向，遵循W3C标准的团队都遵循这个标准，可以减少差异化。
Doctype作用? 严格模式与混杂模式如何区分？它们有何意义?
（1）、<!DOCTYPE> 声明位于文档中的最前面，处于 <html> 标签之前。告知浏览器以何种模式来渲染文档。

（2）、严格模式的排版和 JS 运作模式是 以该浏览器支持的最高标准运行。

（3）、在混杂模式中，页面以宽松的向后兼容的方式显示。模拟老式浏览器的行为以防止站点无法工作。

（4）、DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。
你知道多少种Doctype文档类型？
 该标签可声明三种 DTD 类型，分别表示严格版本、过渡版本以及基于框架的 HTML 文档。
 HTML 4.01 规定了三种文档类型：Strict、Transitional 以及 Frameset。
 XHTML 1.0 规定了三种 XML 文档类型：Strict、Transitional 以及 Frameset。
Standards （标准）模式（也就是严格呈现模式）用于呈现遵循最新标准的网页，而 Quirks （包容）模式（也就是松散呈现模式或者兼容模式）用于呈现为传统浏览器而设计的网页。
HTML与XHTML——二者有什么区别
区别：
1.所有的标记都必须要有一个相应的结束标记
2.所有标签的元素和属性的名字都必须使用小写
3.所有的XML标记都必须合理嵌套
4.所有的属性必须用引号""括起来
5.把所有<和&特殊符号用编码表示
6.给所有属性赋一个值
7.不要在注释内容中使“--”
8.图片必须有说明文字
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;/*所有识别*/
      .background-color:#00deff\9; /*IE6、7、8识别*/
      +background-color:#a200ff;/*IE6、7识别*/

      _background-color:#1e0bd1;/*IE6识别*/
```

```
    }

*  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)
解释下浮动和它的工作原理？清除浮动的技巧
浮动元素脱离文档流，不占据空间。浮动元素碰到包含它的边框或者浮动元素的边框停留。

1.使用空标签清除浮动。
    这种方法是在所有浮动标签后面添加一个空标签 定义css clear:both. 弊端就是增加了无意义标签。
2.使用overflow。
    给包含浮动元素的父标签添加css属性 overflow:auto; zoom:1; zoom:1用于兼容IE6。
3.使用after伪对象清除浮动。
    该方法只适用于非IE浏览器。具体写法可参照以下示例。使用中需注意以下几点。一、该方法中必须为需要清除浮动元素的伪对象中设置 height:0，否则该元素会比实际高出若干像素；
浮动元素引起的问题和解决办法？
浮动元素引起的问题：
（1）父元素的高度无法被撑开，影响与父元素同级的元素
（2）与浮动元素同级的非浮动元素会跟随其后
（3）若非第一个元素浮动，则该元素之前的元素也需要浮动，否则会影响页面显示的结构
解决方法：
使用clear:both;属性来清除元素的浮动可解决2、3问题，对于问题1，添加如下样式，给父元素添加.clearfix:after{content: ".";display: block;height: 0;clear: both;visibility: hidden;}
.clearfix{display: inline-block;} /* for IE/Mac */

清除浮动的几种方法：

1，额外标签法，<div style="clear:both;"></div>（缺点：不过这个办法会增加额外的标签使HTML结构看起来不够简洁。）

2，使用after伪类
```

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

3,浮动外部元素
4,设置`overflow`为`hidden`或者auto
IE 8以下版本的浏览器中的盒模型有什么不同
IE8以下浏览器的盒模型中定义的元素的宽高不包括内边距和边框
DOM操作—怎样添加、移除、移动、复制、创建和查找节点。
（1）创建新节点

```
createDocumentFragment()    //创建一个DOM片段

createElement()   //创建一个具体的元素

createTextNode()   //创建一个文本节点
```

（2）添加、移除、替换、插入

```
appendChild()

removeChild()

replaceChild()

insertBefore() //在已有的子节点前插入一个新的子节点
```

（3）查找

```
getElementsByTagName()    //通过标签名称

getElementsByName()    //通过元素的Name属性的值(IE容错能力较强，会得到一个数组，其中包括id等于name值的)

getElementById()    //通过元素Id，唯一性
```

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

iframe的优缺点？

1.<iframe>优点：

    解决加载缓慢的第三方内容如图标和广告等的加载问题
    Security sandbox
    并行加载脚本

2.<iframe>的缺点：

    *iframe会阻塞主页面的Onload事件；

    *即时内容为空，加载也需要时间
    *没有语意

如何实现浏览器内多个标签页之间的通信?

调用localstorge、cookies等本地存储方式

webSocket如何兼容低浏览器？

Adobe Flash Socket 、 ActiveX HTMLFile (IE) 、 基于 multipart 编码发送 XHR 、 基于长轮询的 XHR

线程与进程的区别

一个程序至少有一个进程,一个进程至少有一个线程.

线程的划分尺度小于进程，使得多线程程序的并发性高。

另外，进程在执行过程中拥有独立的内存单元，而多个线程共享内存，从而极大地提高了程序的运行效率。

线程在执行过程中与进程还是有区别的。每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口。但是线程不能够独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制。

从逻辑角度来看，多线程的意义在于一个应用程序中，有多个执行部分可以同时执行。但操作系统并没有将多个线程看做多个独立的应用，来实现进程的调度和管理以及资源分配。这就是进程和线程的重要区别。

你如何对网站的文件和资源进行优化？

期待的解决方案包括：

 文件合并
 文件最小化/文件压缩
 使用 CDN 托管
 缓存的使用（多个域名来提供缓存）
 其他

请说出三种减少页面加载时间的方法。

1.优化图片
2.图像格式的选择（GIF：提供的颜色较少，可用在一些对颜色要求不高的地方）
3.优化CSS（压缩合并css，如margin-top,margin-left...）
4.网址后加斜杠（如www.campr.com/目录，会判断这个“目录是什么文件类型，或者是目录。）
5.标明高度和宽度（如果浏览器没有找到这两个参数，它需要一边下载图片一边计算大小，如果图片很多，浏览器需要不断地调整页面。这不但影响速度，也影响浏览体验。
当浏览器知道了高度和宽度参数后，即使图片暂时无法显示，页面上也会腾出图片的空位，然后继续加载后面的内容。从而加载时间快了，浏览体验也更好了。）

6.减少http请求（合并文件，合并图片）。
你都使用哪些工具来测试代码的性能？
Profiler, JSPerf（http://jsperf.com/nexttick-vs-setzerotimeout-vs-settimeout）, Dromaeo
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。
解决方法简单的出奇，只要在<head>之间加入一个<link>或者<script>元素就可以了。
null和undefined的区别？
undefined是一个表示"无"的原始值，转为数值时为 当声明的变量还未被初始化时，变量的默认值为null用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象。

（1）变量被声明了，但没有赋值时，就等于undefined。 （2）调用函数时，应该提供的参数没有提供，该参数等于undefined。 （3）对象没有赋值的属性，该属性的值为undefined。 （4）函数没有返回值时，默认返回undefined。

（1） 作为函数的参数，表示该函数的参数不是对象。 （2） 作为对象原型链的终点。

new操作符具体干了什么呢?
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 。

```
var obj  = {};
obj.__proto__ = Base.prototype;
Base.call(obj);
```

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
{'age':'12', 'name':'back'}
js延迟加载的方式有哪些？
defer和async、动态创建DOM方式（创建script，插入到DOM中，加载完毕后callBack）、按需异步载入js
如何解决跨域问题?
jsonp、 document.domain+iframe、window.name、window.postMessage、服务器上设置代理页面

jsonp的原理是动态插入script标签
具体参见：详解js跨域问题

documen.write和 innerHTML的区别
document.write只能重绘整个页面

innerHTML可以重绘页面的一部分

.call() 和 .apply() 的区别和作用？
作用：动态改变某个类的某个方法的运行环境。
区别参见：JavaScript学习总结（四）function函数部分

哪些操作会造成内存泄漏？
内存泄漏指任何对象在您不再拥有或需要它之后仍然存在。
垃圾回收器定期扫描对象，并计算引用了每个对象的其他对象的数量。如果一个对象的引用数量为 0（没有其他对象引用过该对象），或对该对象的惟一引用是循环的，那么该对象的内存即可回收。

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）
详见：详解js变量、作用域及内存

JavaScript中的作用域与变量声明提升？
详见：详解JavaScript函数模式

如何判断当前脚本运行在浏览器还是node环境中？
通过判断Global对象是否为window，如果不为window，当前脚本没有运行在浏览器中
其他问题？
你遇到过比较难的技术问题是？你是如何解决的？
常使用的库有哪些？常用的前端开发工具？开发过什么应用或组件？
列举IE 与其他浏览器不一样的特性？
99%的网站都需要被重构是那本书上写的？
* 网站重构：应用web标准进行设计（第2版）
什么叫优雅降级和渐进增强？
优雅降级：Web站点在所有新式浏览器中都能正常工作，如果用户使用的是老式浏览器，则代码会检查以确认它们是否能正常工作。由于IE独特的盒模型布局问题，针对不同版本的IE的hack实践过优雅降级了,为那些无法支持功能的浏览器增加候选方案，使之在旧式浏览器上以某种形式降级体验却不至于完全失效.

渐进增强：从被所有浏览器支持的基本功能开始，逐步地添加那些只有新式浏览器才支持的功能,向页面增加无害于基础浏览器的额外样式和功能的。当浏览器支持时，它们会自动地呈现出来并发挥作用。
详见：css学习归纳总结（一）

WEB应用从服务器主动推送Data到客户端有那些方式？
对Node的优点和缺点提出了自己的看法？
*（优点）因为Node是基于事件驱动和无阻塞的，所以非常适合处理并发请求，
  因此构建在Node上的代理服务器相比其他技术实现（如Ruby）的服务器表现要好得多。
  此外，与Node代理服务器交互的客户端代码是由javascript语言编写的，
  因此客户端和服务器端都用同一种语言编写，这是非常美妙的事情。

*（缺点）Node是一个相对新的开源项目，所以不太稳定，它总是一直在变，
  而且缺少足够多的第三方库支持。看起来，就像是Ruby/Rails当年的样子。
除了前端以外还了解什么其它技术么？你最最厉害的技能是什么？
你常用的开发工具是什么，为什么？
对前端界面工程师这个职位是怎么样理解的？它的前景会怎么样？
前端是最贴近用户的程序员，比后端、数据库、产品经理、运营、安全都近。
    1、实现界面交互
    2、提升用户体验
    3、有了Node.js，前端可以实现服务端的一些事情

前端是最贴近用户的程序员，前端的能力就是能让产品从 90分进化到 100 分，甚至更好，

参与项目，快速高质量完成实现效果图，精确到1px；

与团队成员，UI设计，产品经理的沟通；

做好的页面结构，页面重构和用户体验；

处理hack，兼容、写出优美的代码格式；

针对服务器的优化、拥抱最新前端技术。
你在现在的团队处于什么样的角色，起到了什么明显的作用？
你认为怎样才是全端工程师（Full Stack developer）？
介绍一个你最得意的作品吧？
项目中遇到什么问题？如何解决？
你的优点是什么？缺点是什么？
如何管理前端团队?
最近在学什么？能谈谈你未来3，5年给自己的规划吗？
你有哪些性能优化的方法？
（详情请看雅虎14条性能优化原则）。

（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。
http状态码有那些？分别代表是什么意思？
100-199 用于指定客户端应相应的某些动作。
200-299 用于表示请求成功。
300-399 用于已经移动的文件并且常被包含在定位头信息中指定新的地址信息。
400-499 用于指出客户端的错误。400 1、语义有误，当前请求无法被服务器理解。401 当前请求需要用户验证
403 服务器已经理解请求，但是拒绝执行它。
500-599 用于支持服务器错误。 503 – 服务不可用
详情：http://segmentfault.com/blog/trigkit4/1190000000691919

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？
分为4个步骤：
（1），当发送一个URL请求时，不管这个URL是Web页面的URL还是Web页面上每个资源的URL，浏览器都会开启一个线程来处理这个请求，同时在远程DNS服务器上启动一个DNS查询。这能使浏览器获得请求对应的IP地址。
（2）， 浏览器与远程Web服务器通过TCP三次握手协商来建立一个TCP/IP连接。该握手包括一个同步报文，一个同步-应答报文和一个应答报文，这三个报文在 浏览器和服务器之间传递。该握手首先由客户端尝试建立起通信，而后服务器应答并接受客户端的请求，最后由客户端发出该请求已经被接受的报文。
（3），一旦TCP/IP连接建立，浏览器会通过该连接向远程服务器发送HTTP的GET请求。远程服务器找到资源并使用HTTP响应返回该资源，值为200的HTTP响应状态表示一个正确的响应。

（4），此时，Web服务器提供资源服务，客户端开始下载资源。

请求返回后，便进入了我们关注的前端模块
简单来说，浏览器会解析HTML生成DOM Tree，其次会根据CSS生成CSS Rule Tree，而javascript又可以根据DOM API操作DOM
详情：从输入 URL 到浏览器接收的过程中发生了什么事情？

平时如何管理你的项目？
先期团队必须确定好全局样式（globe.css），编码模式(utf-8) 等；

编写习惯必须一致（例如都是采用继承式的写法，单样式都写成一行）；

标注样式编写人，各模块都及时标注（标注关键样式调用的地方）；

页面进行标注（例如 页面 模块 开始和结束）；

CSS跟HTML 分文件夹并行存放，命名都得统一（例如style.css）；

JS 分文件夹存放 命名以该JS功能为准的英文翻译。

图片采用整合的 images.png png8 格式文件使用 尽量整合在一起使用方便将来的管理
说说最近最流行的一些东西吧？常去哪些网站？
Node.js、Mongodb、npm、MVVM、MEAN、three.js,React 。
网站：w3cfuns,sf,hacknews,CSDN,慕课，博客园，InfoQ,w3cplus等
javascript对象的几种创建方式
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
javascript继承的6种方法
1，原型链继承
2，借用构造函数继承
3，组合继承(原型+借用构造)
4，原型式继承
5，寄生式继承
6，寄生组合式继承
详情：JavaScript继承方式详解

ajax过程
(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.

详情：JavaScript学习总结（七）Ajax和Http状态字

异步加载和延迟加载
1.异步加载的方案： 动态插入script标签
2.通过ajax去获取js代码，然后通过eval执行
3.script标签上添加defer或者async属性
4.创建并插入iframe，让它异步执行js
5.延迟加载：有些 js 代码并不是页面初始化的时候就立刻需要的，而稍后的某些情况才需要的。
前端安全问题？
（XSS，sql注入，CSRF）
CSRF：是跨站请求伪造，很明显根据刚刚的解释，他的核心也就是请求伪造，通过伪造身份提交POST和GET请求来进行跨域的攻击。

完成CSRF需要两个步骤：

1.登陆受信任的网站A，在本地生成COOKIE

2.在不登出A的情况下，或者本地COOKIE没有过期的情况下，访问危险网站B。
ie各版本和chrome可以并行下载多少个资源
IE6 两个并发，iE7升级之后的6个并发，之后版本也是6个

Firefox，chrome也是6个
javascript里面的继承怎么实现，如何避免原型链上面的对象共享
用构造函数和原型链的混合模式去实现继承，避免对象共享可以参考经典的extend()函数，很多前端框架都有封装的，就是用一个空函数当做中间变量
grunt， YUI compressor 和 google clojure用来进行代码压缩的用法。
YUI Compressor 是一个用来压缩 JS 和 CSS 文件的工具，采用Java开发。

使用方法：

```
//压缩JS
java -jar yuicompressor-2.4.2.jar --type js --charset utf-8 -v src.js > packed.js
//压缩CSS
java -jar yuicompressor-2.4.2.jar --type css --charset utf-8 -v src.css > packed.css
```

详情请见：你需要掌握的前端代码性能优化工具

Flash、Ajax各自的优缺点，在使用中如何取舍？
1、Flash ajax对比
Flash适合处理多媒体、矢量图形、访问机器；对CSS、处理文本上不足，不容易被搜索。
Ajax对CSS、文本支持很好，支持搜索；多媒体、矢量图形、机器访问不足。
共同点：与服务器的无刷新传递消息、用户离线和在线状态、操作DOM
请解释一下 JavaScript 的同源策略。
概念:同源策略是客户端脚本（尤其是Netscape Navigator2.0，其目的是防止某个文档或脚本从多个不同源装载。

这里的同源策略指的是：协议，域名，端口相同，同源策略是一种安全协议。
指一段脚本只能读取来自同一来源的窗口和文档的属性。

为什么要有同源限制？
我们举例说明：比如一个黑客程序，他利用Javascript读取到你的表单中 什么是 "use strict"; ？使用它的好处和坏处分别是什么？

Javascript在更严格的条件下运行。

设立"严格模式"的目的，主要有以下几个：

- 消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为;
- 消除代码运行的一些不安全之处，保证代码运行的安全；
- 提高编译器效率，增加运行速度；
- 为未来新版本的Javascript做好铺垫。

注：经过测试 缺点：
现在网站的merge 后，这个串就到了文件的中间，不仅没有指示严格模式，反而在压缩后浪费了字节。

GET和POST的区别，何时使用POST？

GET：一般用于信息获取，使用URL传递参数，对所发送信息的数量也有限制，一般在2000个字符

POST：一般用于修改服务器上的资源，对所发送的信息没有限制。

GET方式需要使用Request.QueryString来取得变量的值，而POST方式通过Request.Form来获取变量的值，也就是说Get是通过地址栏来传值，而Post是通过提交表单来传值。

然而，在以下情况中，请使用 POST 请求：
无法使用缓存文件（更新服务器上的文件或数据库）
向服务器发送大量数据（POST 没有数据量限制）
发送包含未知字符的用户输入时，POST 比 GET 更稳定也更可靠
哪些地方会出现css阻塞，哪些地方会出现js阻塞？
js的阻塞特性：所有浏览器在下载JS下载、解析、执行完毕后才开始继续JS，但是 由于浏览器为了防止出现DOM树，需要重新构建 嵌入JS只会阻塞其后内容的显示，2种方式都会阻塞其后资源的下载。也就是说外部样式不会阻塞外部脚本的加载，但会阻塞外部脚本的执行。

CSS本来是可以并行下载的，在什么情况下会出现阻塞加载了(在测试观察中，CSS都是阻塞加载）

当JS的时候，该JS放到 根本原因：因为浏览器会维持css和JS会阻塞后面的资源加载，所以就会出现上面 嵌入 1、放在底部，虽然放在底部照样会阻塞所有呈现，但不会阻塞资源下载。 2、如果嵌入JS放在head中，请把嵌入JS放在CSS头部。 3、使用defer（只支持IE） 4、不要在嵌入的JS中调用运行时间较长的函数，如果一定要用，可以用 `setTimeout`来调用

Javascript无阻塞加载具体方式
将脚本放在底部。head中，用以保证在<script>标签放在前。
成组脚本：由于每个<script>总数也可以改善性能。适用于内联脚本和外部脚本。

前。
非阻塞脚本：等页面完成加载后，再加载window.onload事件发出后开始下载代码。
（1）fierfox3.5更高版本浏览器
（2）动态脚本元素：文档对象模型（DOM）允许你使用js动态创建

```
<script>
var script=document.createElement("script");
script.type="text/javascript";
script.src="file.js";
document.getElementsByTagName("head")[0].appendChild(script);
</script>
```

此技术的重点在于：无论在何处启动下载，文件额下载和运行都不会阻塞其他页面处理过程。即使在head里（除了用于下载文件的http链接）。

闭包相关问题？
详情请见：详解js闭包

js事件处理程序问题？

详情请见：JavaScript学习总结（九）事件详解

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）。
写一个通用的事件侦听器函数?

```
// event(事件)工具集，来源：github.com/markyun
    markyun.Event = {
        // 页面加载完成后
        readyEvent : function(fn) {
            if (fn==null) {
                fn=document;
            }
            var oldonload = window.onload;
            if (typeof window.onload != 'function') {
                window.onload = fn;
            } else {
                window.onload = function() {
                    oldonload();
                    fn();
                };
            }
        },
        // 视能力分别使用dom0||dom2||IE方式 来绑定事件
        // 参数： 操作的元素,事件名称 ,事件处理程序
        addEvent : function(element, type, handler) {
            if (element.addEventListener) {
                //事件类型、需要执行的函数、是否捕捉
                element.addEventListener(type, handler, false);
            } else if (element.attachEvent) {
                element.attachEvent('on' + type, function() {
                    handler.call(element);
                });
            } else {
                element['on' + type] = handler;
            }
        },
        // 移除事件
        removeEvent : function(element, type, handler) {
            if (element.removeEnentListener) {
                element.removeEnentListener(type, handler, false);
            } else if (element.datachEvent) {
                element.detachEvent('on' + type, handler);
            } else {
                element['on' + type] = null;
            }
        },
        // 阻止事件 (主要是事件冒泡，因为IE不支持事件捕获)
        stopPropagation : function(ev) {
            if (ev.stopPropagation) {
                ev.stopPropagation();
            } else {

                ev.cancelBubble = true;
```

```
            }
        },
        // 取消事件的默认行为
        preventDefault : function(event) {
            if (event.preventDefault) {
                event.preventDefault();
            } else {
                event.returnValue = false;
            }
        },
        // 获取事件目标
        getTarget : function(event) {
            return event.target || event.srcElement;
        },
        // 获取event对象的引用，取到事件的所有信息，确保随时能使用event；
        getEvent : function(e) {
            var ev = e || window.event;
            if (!ev) {
                var c = this.getEvent.caller;
                while (c) {
                    ev = c.arguments[0];
                    if (ev && Event == ev.constructor) {
                        break;
                    }
                    c = c.caller;
                }
            }
            return ev;
        }
    };
```

Node.js的适用场景？

高并发、聊天、实时消息推送

JavaScript原型，原型链 ？有什么特点？

* 原型对象也是普通的对象，是对象一个自带隐式的 __proto__ 属性，原型也有可能有自己的原型，如果一个原型对象的原型不为null的话，我们就称之为原型链。

* 原型链是由一些用来继承和共享属性的对象组成的（有限的）对象链。

页面重构怎么操作？

编写 CSS、让页面结构更合理化，提升用户体验，实现良好的页面效果和提升性能。

WEB应用从服务器主动推送Data到客户端有那些方式？

html5 websoket

WebSocket通过Flash

XHR长时间连接

XHR Multipart Streaming

不可见的Iframe

<script>标签的长时间连接(可跨域)

事件、IE与火狐的事件机制有什么区别？ 如何阻止冒泡？

1．我们在网页中的某个操作（有的操作对应多个事件）。例如：当我们点击一个按钮就会产生一个事件。是可以被JavaScript 侦测到的行为。

2．事件处理机制：IE是事件冒泡、firefox同时支持两种事件模型，也就是：捕获型事件和冒泡型事件。；

3． ev.stopPropagation();注意旧ie的方法 ev.cancelBubble = true;

ajax 是什么?ajax 的交互模型?同步和异步的区别?如何解决跨域问题?

详情请见：JavaScript学习总结（七）Ajax和Http状态字

1．通过异步模式，提升了用户体验

2．优化了浏览器和服务器之间的传输，减少不必要的数据往返，减少了带宽占用

3．Ajax在客户端运行，承担了一部分本来由服务器承担的工作，减少了大用户量下的服务器负载。

2．Ajax的最大的特点是什么。

Ajax可以实现动态不刷新（局部刷新）
readyState属性 状态 有5个可取值： 0=未初始化 ，1=启动 2=发送，3=接收，4=完成

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。

跨域： jsonp、 iframe、window.name、window.postMessage、服务器上设置代理页面
js对象的深度克隆

```
function clone(Obj) {
       var buf;
       if (Obj instanceof Array) {
           buf = [];  //创建一个空的数组
           var i = Obj.length;
           while (i--) {
               buf[i] = clone(Obj[i]);
           }
           return buf;
       }else if (Obj instanceof Object){
           buf = {};  //创建一个空对象
           for (var k in Obj) {  //为这个对象添加新的属性
               buf[k] = clone(Obj[k]);
           }
           return buf;
       }else{
           return Obj;
       }
   }
```

AMD和CMD 规范的区别？
详情请见：详解JavaScript模块化开发

网站重构的理解？
网站重构：在不改变外部行为的前提下，简化结构、添加可读性，而在网站前端保持一致的行为。也就是说是在不改变UI的情况下，对网站进行优化，在扩展的同时保持一致的UI。

对于传统的网站来说重构通常是：

表格(table)布局改为DIV+CSS
使网站前端兼容于现代浏览器(针对于不合规范的CSS、如对IE6有效的)
对于移动平台的优化
针对于SEO进行优化
深层次的网站重构应该考虑的方面

减少代码间的耦合
让代码保持弹性
严格按规范编写代码
设计可扩展的API
代替旧有的框架、语言(如VB)
增强用户体验
通常来说对于速度的优化也包含在重构中

压缩JS、CSS、image等前端资源(通常是由服务器来解决)
程序的性能优化(如数据读写)
采用CDN来加速资源加载
对于JS DOM的优化
HTTP服务器的文件缓存
如何获取UA？

```
<script>
    function whatBrowser() {
        document.Browser.Name.value=navigator.appName;
        document.Browser.Version.value=navigator.appVersion;
        document.Browser.Code.value=navigator.appCodeName;
        document.Browser.Agent.value=navigator.userAgent;
    }
</script>
```

js数组去重
以下是数组去重的三种方法：

```
Array.prototype.unique1 = function () {
  var n = []; //一个新的临时数组
  for (var i = 0; i < this.length; i++) //遍历当前数组
  {
    //如果当前数组的第i已经保存进了临时数组，那么跳过，
    //否则把当前项push到临时数组里面
    if (n.indexOf(this[i]) == -1) n.push(this[i]);
  }
  return n;
}

Array.prototype.unique2 = function()
{
    var n = {},r=[]; //n为hash表，r为临时数组
    for(var i = 0; i < this.length; i++) //遍历当前数组
    {
        if (!n[this[i]]) //如果hash表中没有当前项
        {
            n[this[i]] = true; //存入hash表
            r.push(this[i]); //把当前数组的当前项push到临时数组里面
        }

    }
```

```
    return r;
}

Array.prototype.unique3 = function()
{
    var n = [this[0]]; //结果数组
    for(var i = 1; i < this.length; i++) //从第二项开始遍历
    {
        //如果当前数组的第i项在当前数组中第一次出现的位置不是i，
        //那么表示第i项是重复的，忽略掉。否则存入结果数组
        if (this.indexOf(this[i]) == i) n.push(this[i]);
    }
    return n;
}
```

HTTP状态码

100 Continue 继续，一般在发送post请求时，已发送了http header之后服务端将返回此信息，表示确认，之后发送具体参数信息

200 OK 正常返回信息

201 Created 请求成功并且服务器创建了新的资源

202 Accepted 服务器已接受请求，但尚未处理

301 Moved Permanently 请求的网页已永久移动到新位置。

302 Found 临时性重定向。

303 See Other 临时性重定向，且总是使用 GET 请求新的 URI。

304 Not Modified 自从上次请求后，请求的网页未修改过。

400 Bad Request 服务器无法理解请求的格式，客户端不应当尝试再次使用相同的内容发起请求。

401 Unauthorized 请求未授权。

403 Forbidden 禁止访问。

404 Not Found 找不到如何与 URI 相匹配的资源。

500 Internal Server Error 最常见的服务器端错误。

503 Service Unavailable 服务器端暂时无法处理请求（可能是过载或维护）。

cache-control

网页的缓存是由HTTP消息头中的private、no-cache、max-age、must-revalidate等，默认为 max-age的效果。但是如果同时存在，则被max-age覆盖。

Expires = "Expires" ":" HTTP-date

例如

Expires: Thu, 01 Dec 1994 16:00:00 GMT （必须是GMT格式）

如果把它设置为 max-age都可以用来指定文档的过期时间，但是二者有一些细微差别

1.Expires在HTTP/1.0中已经定义，Cache-Control:max-age在HTTP/1.1中才有定义，为了向下兼容，仅使用max-age不够；

2.Expires指定一个绝对的过期时间(GMT格式),这么做会导致至少2个问题：1)客户端和服务器时间不同步导致Expires的配置出现问题。 2）很容易在配置后忘记具体的过期时间，导致过期来临出现浪涌现象；

3.max-age 指定的是从文档被访问后的存活时间，这个时间是个相对值(比如:3600s),相对的是文档第一次被请求时服务器记录的Request_time(请求时间)

4.Expires指定的时间可以是相对文件的最后访问时间(Atime)或者修改时间(MTime),而max-age相对对的是文档的请求时间(Atime)

如果值为no-cache,那么每次都会访问服务器。如果值为max-age,则在过期之前不会重复访问服务器。
js操作获取和设置cookie

```
//创建cookie
function setCookie(name, value, expires, path, domain, secure) {
    var cookieText = encodeURIComponent(name) + '=' + encodeURIComponent(value);
    if (expires instanceof Date) {
        cookieText += '; expires=' + expires;
    }
    if (path) {
        cookieText += '; expires=' + expires;
    }
    if (domain) {
        cookieText += '; domain=' + domain;
    }
    if (secure) {
        cookieText += '; secure';
    }
    document.cookie = cookieText;
}

//获取cookie
function getCookie(name) {
    var cookieName = encodeURIComponent(name) + '=';
    var cookieStart = document.cookie.indexOf(cookieName);
    var cookieValue = null;
    if (cookieStart > -1) {
        var cookieEnd = document.cookie.indexOf(';', cookieStart);
        if (cookieEnd == -1) {
            cookieEnd = document.cookie.length;
        }
```

1. jQuery 库中的 $() 是什么？（答案如下）
$() 函数是 jQuery() 函数的别称，乍一看这很怪异，还使 jQuery 代码晦涩难懂。一旦你适应了，你会爱上它的简洁。$() 函数用于将任何对象包裹成 jQuery 对象，接着你就被允许调用定义在 jQuery 对象上的多个不同方法。你甚至可以将一个选择器字符串传入 $() 函数，它会返回一个包含所有匹配的 DOM 元素数组的 jQuery 对象。这个问题我已经见过好几次被提及，尽管它非常基础，它经常被用来区分一个开发人员是否了解 jQuery。

2. 网页上有 5 个 元素，如何使用 jQuery来选择它们？（答案）
另一个重要的 jQuery 问题是基于选择器的。jQuery 支持不同类型的选择器，例如 ID 选择器、class 选择器、标签选择器。鉴于这个问题没提到 ID 和 class，你可以用标签选择器来选择所有的 div 元素。jQuery 代码：
$("div")，这样会返回一个包含所有 5 个 div 标签的 jQuery 对象。更详细的解答参见上面链接的文章。

3. jQuery 里的 ID 选择器和 class 选择器有何不同？（答案）

如果你用过 CSS，你也许就知道 ID 选择器和 class 选择器之间的差异，jQuery 也同样如此。ID 选择器使用 ID 来

选择元素，比如 #element1，而 class 选择器使用 CSS class 来选择元素。当你只需要选择一个元素时，使用 ID 选择器，而如果你想要选择一组具有相同 CSS class 的元素，就要用 class 选择器。在面试过程中，你有很大几率会被要求使用 ID 选择器和 class 选择器来写代码。下面的 jQuery 代码使用了 ID 选择器和 class 选择器：

```
$('#LoginTextBox')  // Returns element wrapped as jQuery object with id='LoginTextBox'
$('.active') // Returns all elements with CSS class active.
```

正如你所见，从语法角度来说，ID 选择器和 class 选择器的另一个不同之处是，前者用字符”#”而后者用字符”.”。更详细的分析和讨论参见上面的答案链接。

4. 如何在点击一个按钮时使用 jQuery 隐藏一个图片？
这是一个事件处理问题。jQuery为按钮点击之类的事件提供了很好的支持。你可以通过以下代码去隐藏一个通过ID或class定位到的图片。你需要知道如何为按钮设置事件并执行hide() 方法，代码如下所示：

```
$('#ButtonToClick').click(function(){
    $('#ImageToHide').hide();
});
```

我喜欢这个问题，因为很贴近实际使用，代码也不复杂。

5. $(document).ready() 是个什么函数？为什么要用它？(answer)
这个问题很重要，并且常常被问到。 ready() 函数用于在文档进入ready状态时执行代码。当DOM 完全加载（例如HTML被完全解析DOM树构建完成时），jQuery允许你执行代码。使用$(document).ready()的最大好处在于它适用于所有浏览器，jQuery帮你解决了跨浏览器的难题。需要进一步了解的用户可以点击 answer链接查看详细讨论。

6. JavaScript window.onload 事件和 jQuery ready 函数有何不同？（答案）
这个问答是紧接着上一个的。JavaScript window.onload 事件和 jQuery ready 函数之间的主要区别是，前者除了要等待 DOM 被创建还要等到包括大型图片、音频、视频在内的所有外部资源都完全加载。如果加载图片和媒体内容花费了大量时间，用户就会感受到定义在 window.onload 事件上的代码在执行时有明显的延迟。

另一方面，jQuery ready() 函数只需对 DOM 树的等待，而无需对图像或外部资源加载的等待，从而执行起来更快。使用 jQuery $(document).ready() 的另一个优势是你可以在网页里多次使用它，浏览器会按它们在 HTML 页面里出现的顺序执行它们，相反对于 onload 技术而言，只能在单一函数里使用。鉴于这个好处，用 jQuery ready() 函数比用 JavaScript window.onload 事件要更好些。

7. 如何找到所有 HTML select 标签的选中项？（答案如下）
这是面试里比较棘手的 jQuery 问题之一。这是个基础的问题，但是别期望每个 jQuery 初学者都知道它。你能用下面的 jQuery 选择器获取所有具备 multiple=true 的 <select > 标签的选中项：

```
$('[name=NameOfSelectedTag] :selected')
```

这段代码结合使用了属性选择器和 :selected 选择器，结果只返回被选中的选项。你可按需修改它，比如用 id 属性而不是 name 属性来获取 <select> 标签。

8. jQuery 里的 each() 是什么函数？你是如何使用它的？（答案如下）
each() 函数就像是 Java 里的一个 Iterator，它允许你遍历一个元素集合。你可以传一个函数给 each() 方法，被调用的 jQuery 对象会在其每个元素上执行传入的函数。有时这个问题会紧接着上面一个问题，举个例子，如何在 alert 框里显示所有选中项。我们可以用上面的选择器代码找出所有选中项，然后我们在 alert 框中用 each() 方法来一个个打印它们，代码如下：

```
$('[name=NameOfSelectedTag] :selected').each(function(selected) {
    alert($(selected).text());
});
```

其中 text() 方法返回选项的文本。

9. 你是如何将一个 HTML 元素添加到 DOM 树中的？（答案如下）

你可以用 jQuery 方法 appendTo() 将一个 HTML 元素添加到 DOM 树中。这是 jQuery 提供的众多操控 DOM 的方法中的一个。你可以通过 appendTo() 方法在指定的 DOM 元素末尾添加一个现存的元素或者一个新的 HTML 元素。

10. 你能用 jQuery 代码选择所有在段落内部的超链接吗？（答案略）
这是另一个关于选择器的 jQuery 面试题。就像其他问题那样，只需一行 jQuery 代码就能搞定。你可以使用下面这个 jQuery 代码片段来选择所有嵌套在段落（<p>标签）内部的超链接（<a>标签）……

11. $(this) 和 this 关键字在 jQuery 中有何不同？（答案如下）
这对于很多 jQuery 初学者来说是一个棘手的问题，其实是个简单的问题。$(this) 返回一个 jQuery 对象，你可以对它调用多个 jQuery 方法，比如用 text() 获取文本，用val() 获取值等等。而 this 代表当前元素，它是 JavaScript 关键词中的一个，表示上下文中的当前 DOM 元素。你不能对它调用 jQuery 方法，直到它被 $() 函数包裹，例如 $(this)。

12. 你如何使用jQuery来提取一个HTML 标记的属性 例如. 链接的href? (答案)
attr() 方法被用来提取任意一个HTML元素的一个属性的值. 你首先需要利用jQuery选择及选取到所有的链接或者一个特定的链接，然后你可以应用attr()方法来获得他们的href属性的值。下面的代码会找到页面中所有的链接并返回href值：

```
$('a').each(function(){
   alert($(this).attr('href'));
});
```

13. 你如何使用jQuery设置一个属性值? (答案)
前面这个问题之后额外的一个后续问题是，attr()方法和jQuery中的其它方法一样，能力不止一样. 如果你在调用attr()的同时带上一个值 例如. attr(name, value)，这里name是属性的名称，value是属性的新值。

14. jQuery中 detach() 和 remove() 方法的区别是什么? (答案)
尽管 detach() 和 remove() 方法都被用来移除一个DOM元素，两者之间的主要不同在于 detach() 会保持对过去被解除元素的跟踪，因此它可以被取消解除，而 remove() 方法则会保持过去被移除对象的引用. 你也还可以看看 用来向DOM中添加元素的 appendTo() 方法.

15. 你如何利用jQuery来向一个元素中添加和移除CSS类? (答案)
通过利用 addClass() 和 removeClass() 这两个 jQuery 方法。动态的改变元素的class属性可以很简单例如. 使用类“.active"来标记它们的未激活和激活状态，等等.

16. 使用 CDN 加载 jQuery 库的主要优势是什么 ? (答案)
这是一个稍微高级点儿的jQuery问题。好吧，除了报错节省服务器带宽以及更快的下载速度这许多的好处之外，最重要的是，如果浏览器已经从同一个CDN下载类相同的 jQuery 版本，那么它就不会再去下载它一次. 因此今时今日，许多公共的网站都将jQuery用于用户交互和动画，如果浏览器已经有了下载好的jQuery库，网站就能有非常好的展示机会。

17. jQuery.get() 和 jQuery.ajax() 方法之间的区别是什么?

ajax() 方法更强大，更具可配置性，让你可以指定等待多久，以及如何处理错误。get() 方法是一个只获取一些数据的专门化方法。

18. jQuery 中的方法链是什么？使用方法链有什么好处？

方法链是对一个方法返回的结果调用另一个方法，这使得代码简洁明了，同时由于只对 DOM 进行了一轮查找，性能方面更加出色。

19. 你要是在一个 jQuery 事件处理程序里返回了 false 会怎样？

这通常用于阻止事件向上冒泡。

20. 哪种方式更高效：document.getElementbyId("myId") 还是 $("#myId")？

第一种，因为它直接调用了 JavaScript 引擎。

JS设计模式
一:设计模式
复制代码
1:工厂模式
```
function createPerson(name,age,job){
    var obj = new Object();
    obj.name = name;
    obj.age = age;
    obj.job = job;
    obj.speak = function(){
        console.log(this.name);
    };
    return obj
}
var person1 = createPerson('panrui',20,'前端工程师')
```
注释:解决了创建多个对象的问题,但是没有结解决对象识别的问题(怎样知道一个对象的类型)
复制代码

复制代码
2:构造函数模式
```
function Person(name,age,job){
    this.name = name;
    this.age = age;
    this.job = job;
    this.speak = function(){
        console.log(this.name);
    };
}
var person2 = new Person('panrui',20,'前端工程师')
```
注释:没有显示的创建对象,没有返回语句,直接将属性赋给this对象,将Person的实例对象标识为一种特定的类型
缺点:每个方法在每个实例上面都需要重新定义一遍,
复制代码

复制代码
3:原型模式
```
function Person(){

}
Person.prototype.name = 'panrui';
Person.prototype.age = 23;
Person.prototype.job = '前端工程师';
Person.prototype.speak = function(){
    console.log(this.name)

}
```

```
var person3 = new Person()
```
注意:省略了为构造函数传递初始化参数,结果所有实例享有相同的属性(对于函数实用,但是对于那些基本属性也说的过去,但是对于引用类型的数据就麻烦了)
基本属性我们可以在实例当中添加一个同名属性,这样可以隐藏原型当中的对应的属性,但是引用类型的属性却会导致所有实例共享
复制代码

复制代码
4:组合使用构造函数与原型模式
构造函数用于定义实例属性,原型上面定义共享的属性和方法
```
function Person(name,age,job){
    this.name = name;
    this.age = age;
    this.job = job;
}
Person.prototype.speak = function(){
    console.log(this.name)
}
var person4 = new Person()
```
复制代码

复制代码
5:动态原型模式
```
function Person(name,age,job){
    this.name = name;
    this.age = age;
    this.job = job;
    if(typeof this.speak != "function"){
        Person.speak = function(){
            console.log(this.name);
        };
    }
}
```