【摘客】2016 精选“前端开发”笔试题

官方网址：zkread.com

更多学习资讯请关注：

官方公众号

APP 下载

—— HTML ——

1.Doctype 作用？标准模式与兼容模式各有什么区别？（百度 2015 面试题）

（1）、<!DOCTYPE>声明位于位于 HTML 文档中的第一行，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。

（2）、标准模式的排版 和 JS 运作模式都是以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显示,模拟老式浏览器的行为以防止站点无法工作。

2.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 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。

3.行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？（百度 2015 面试题）

首先：CSS 规范规定，每个元素都有 display 属性，确定该元素的类型，每个元素都有默认的 display 值，如 div 的 display 默认值为“block”，则为“块级”元素；span 默认 display 属性值为“inline”，是“行内”元素。
（1）行内元素有：a b span img input select strong（强调的语气）
（2）块级元素有：div ul ol li dl dt dd h1 h2 h3 h4…p

常见的空元素：
 <hr> <img> <input> <link> <meta>
鲜为人知的是：<area> <base> <col> <command> <embed> <keygen> <param> <source> <track> <wbr>

4.页面导入样式时，使用 link 和@import 有什么区别？（百度 2015 面试题）

（1）link 属于 XHTML 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 提供的，只能用于加载 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 XHTML 标签，无兼容问题

5.介绍一下你对浏览器内核的理解？（百度 2015 面试题）

主要分成两部分：渲染引擎(layout engineer 或 Rendering Engine)和 JS 引擎。

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它需要编辑、显示网络内容的应用程序都需要内核。

JS 引擎则：解析和执行 javascript 来实现网页的动态效果。

最开始渲染引擎和 JS 引擎并没有区分的很明确，后来 JS 引擎越来越独立，内核就倾向于只指渲染引擎。

6.常见的浏览器内核有哪些？

Trident 内核：IE,MaxThon,TT,The World,360,搜狗浏览器等。[又称 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）]

7.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。
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, 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。

当然也可以直接使用成熟的框架、比如 html5shim;

```
<!--[if lt IE 9]>
   <script>
src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
<![endif]-->
```

* 如何区分 HTML5： DOCTYPE 声明\新增的结构元素\功能元素

8.简述一下你对 HTML 语义化的理解？

用正确的标签做正确的事情。
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;
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;
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;
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。

9.HTML5 的离线储存怎么使用，工作原理能不能解释一下？

在用户没有与因特网连接时，可以正常访问站点或应用，在用户与因特网连接时，更新用户机器上的缓存文件。
原理：HTML5 的离线存储是基于一个新建的.appcache 文件的缓存机制(不是存储技术)，通过这个文件上的解析清单离线存储资源，这些资源就会像 cookie 一样被存储了下来。之后当网络在处于离线状态下时，浏览器会通过被离线存储的数据进行页面展示。

如何使用：
1、页面头部像下面一样加入一个 manifest 的属性；
2、在 cache.manifest 文件的编写离线存储的资源；

```
CACHE MANIFEST
#v0.11
CACHE:
js/app.js
css/style.css
NETWORK:
resourse/logo.png
FALLBACK:
/ /offline.html
```

3、在离线状态时，操作 window.applicationCache 进行需求实现。

10.浏览器是怎么对 HTML5 的离线储存资源进行管理和加载的呢？

在线的情况下，浏览器发现 html 头部有 manifest 属性，它会请求 manifest 文件，如果是第一次访问 app，那么浏览器就会根据 manifest 文件的内容下载相应的资源并且进行离线存储。如果已经访问过 app 并且资源已经离线存储了，那么浏览器就会使用离线的资源加载页面，然后浏览器会对比新的 manifest 文件与旧的 manifest 文件，如果文件没有发生改变，就不做任何操作，如果文件改变了，那么就会重新下载文件中的资源并进行离线存储。

离线的情况下，浏览器就直接使用离线存储的资源。

11.请描述一下 cookies,sessionStorage 和 localStorage 的区别？

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），记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

12.iframe 有那些缺点？（百度 2015 面试题）

*iframe 会阻塞主页面的 Onload 事件；
*搜索引擎的检索程序无法解读这种页面，不利于 SEO;

*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，这样可以绕开以上两个问题。

13.Label 的作用是什么？是怎么用的？

label 标签来定义表单控制间的关系,当用户选择该标签时，浏览器会自动将焦点转到和标签相关的表单控件上。

```
<label for="Name">Number:</label>
```

```
<input type=“text“name="Name" id="Name"/>

<label>Date:<input type="text" name="B"/></label>
```

14.HTML5 的 form 如何关闭自动完成功能？

给不想要提示的 form 或某个 input 设置为 autocomplete=off。

15.如何实现浏览器内多个标签页之间的通信？(阿里)

WebSocket、SharedWorker；
也可以调用 localstorge、cookies 等本地存储方式；

localstorge 另一个浏览上下文里被添加、修改或删除时，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 在无痕模式下设置 localstorge 值时会抛出 QuotaExceededError 的异常

16.webSocket 如何兼容低浏览器？(阿里)

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

17.页面可见性（Page Visibility API） 可以有哪些用途？

通过 visibilityState 的值检测页面当前是否可见，以及打开网页的时间等；
在页面被切换到其他后台进程的时候，自动暂停音乐或视频的播放；

18.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

19.实现不使用 border 画出 1px 高的线，在不同浏览器的标准模式与怪异模式下都能保持一致的效果。

```
<div style="height:1px;overflow:hidden;background:red"></div>
```

20.网页验证码是干嘛的，是为了解决什么安全问题。

区分用户是计算机还是人的公共全自动程序。可以防止恶意破解密码、刷票、论坛灌水；
有效防止黑客对某一个特定注册用户用特定程序暴力破解方式进行不断的登陆尝试。

21.title 与 h1 的区别、b 与 strong 的区别、i 与 em 的区别？

title 属性没有明确意义只表示是个标题，H1 则表示层次明确的标题，对页面信息的抓取也有很大的影响；

strong 是标明重点内容，有语气加强的含义，使用阅读设备阅读网络时：会重读，而<B>是展示强调内容。

i 内容展示为斜体，em 表示强调的文本；

Physical Style Elements -- 自然样式标签
b, i, u, s, pre
Semantic Style Elements -- 语义样式标签
strong, em, ins, del, code
应该准确使用语义样式标签，但不能滥用，如果不能确定时首选使用自然样式标签。

—— CSS ——

1.介绍一下标准的 CSS 的盒子模型？低版本 IE 的盒子模型有什么不同的？（百度 2015 面试题）

（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；

2.CSS 选择符有哪些？哪些属性可以继承？（百度 2015 面试题）

* 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, UL LI DL DD DT;

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

3.CSS 优先级算法如何计算？

* 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

4.CSS3 新增伪类有那些？

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。

:after 在元素之前添加内容,也可以用来做清除浮动。
:before 在元素之后添加内容
:enabled
:disabled 控制表单控件的禁用状态。
:checked 单选框或复选框被选中。

5.如何居中 div？如何居中一个浮动元素？如何让绝对定位的 div 居中？

给 div 设置一个宽度，然后添加 margin:0 auto 属性

```
div{
    width:200px;
    margin:0 auto;
 }
```

居中一个浮动元素

```
 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;//高度可以不设
     margin: -150px 0 0 -250px;
     position:relative;         //相对定位
     background-color:pink;     //方便看效果
     left:50%;
     top:50%;
}
```

让绝对定位的 div 居中

```
position: absolute;
width: 1200px;
background: none;
margin: 0 auto;
top: 0;
left: 0;
bottom: 0;
right: 0;
```

6.display 有哪些值？说明他们的作用。（百度 2015 面试题）

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。象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	象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。

来源：

7.position 的值 relative 和 absolute 定位原点是？

absolute
生成绝对定位的元素，相对于值不为 static 的第一个父元素进行定位。
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，相对于其正常位置进行定位。
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。

8.CSS3 有哪些新特性？

新增各种 CSS 选择器 （: not(.input)：所有 class 不是“input”的节点）
圆角 （border-radius:8px）
多列布局 （multi-column layout）
阴影和反射 （Shadow\Reflect）
文字特效 （text-shadow、）
文字渲染 （Text-decoration）
线性渐变 （gradient）
旋转 （transform）
增加了旋转,缩放,定位,倾斜,动画，多背景
transform:\scale(0.85,0.90)\ translate(0px,-30px)\
skew(-9deg,0deg)\Animation:

9.用纯 CSS 创建一个三角形的原理是什么？

把上、左、右三条边隐藏掉（颜色设为 transparent）

```
#demo {
  width: 0;
  height: 0;
  border-width: 20px;
  border-style: solid;
  border-color: transparent transparent red transparent;
}
```

10.一个满屏“品”字布局 如何设计?

简单的方式：
上面的 div 宽 100%，
下面的两个 div 分别宽 50%，
然后用 float 或者 inline 使其不换行即可

11.经常遇到的浏览器的兼容性有哪些？原因,解决方法是什么，常用 hack 的技巧 ？

* png24 位的图片在 iE6 浏览器上出现背景，解决方案是做成 PNG8.

* 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 #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,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。

* 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 {}

12.li 与 li 之间有看不见的空白间隔是什么原因引起的？有什么解决办法？

行框的排列会受到中间空白（回车\空格）等的影响，因为空格也属于字符,这些空白也会被应用样式，占据空间，所以会有间隔，把字符大小设为 0，就没有空格了。

13.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： `* {padding: 0; margin: 0;}` （强烈不建议）

淘宝的样式初始化代码：

```
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; }
```

14.absolute 的 containing block(容器块)计算方式跟正常流有什么不同？

无论属于哪种，都要先找到其祖先元素中最近的 position 值不为 static 的元素，然后再判断：

1、若此元素为 inline 元素，则 containing block 为能够包含这个元素生成的第一个和最后一个 inline box 的 padding box (除 margin, border 外的区域) 的最小矩形；
2、否则,则由这个祖先元素的 padding box 构成。
如果都找不到，则为 initial containing block。

补充：

1. static(默认的)/relative：简单说就是它的父元素的内容框（即去掉 padding 的部分）
2. absolute: 向上找最近的定位为 absolute/relative 的元素
3. fixed: 它的 containing block 一律为根元素(html/body)，根元素也是 initial containing block

15.对 BFC 规范(块级格式化上下文:block formatting context)的理解？

（W3C CSS 2.1 规范中的一个概念,它是一个独立容器，决定了元素如何对其内容进行定位,以及与其他元素的关系和相互作用。）
一个页面是由很多个 Box 组成的,元素的类型和 display 属性,决定了这个 Box 的类型。
不同类型的 Box,会参与不同的 Formatting Context（决定如何渲染文档的容器）,因此 Box 内的元素会以不同的方式渲染,也就是说 BFC 内部的元素和外部的元素不会互相影响。

16.css 定义的权重（百度 2015 面试题）

以下是权重的规则：标签的权重为 1，class 的权重为 10，id 的权重为 100，以下例子是演示各种定义的权重值：

```
/*权重为 1*/
div{
}
/*权重为 10*/
.class1{
}
/*权重为 100*/
#id1{
}
/*权重为 100+1=101*/
#id1 div{
}
/*权重为 10+1=11*/
.class1 div{
```

```
}
/*权重为 10+10+1=21*/
.class1 .class2 div{
}
```

如果权重相同，则最后定义的样式会起作用，但是应该避免这种情况出现

17. 使用 CSS 预处理器吗？喜欢那个？

SASS（SASS、LESS 没有本质区别，只因为团队前端都是用的 SASS）

让页面里的字体变清晰，变细用 CSS 怎么做？（-webkit-font-smoothing: antialiased;）

position:fixed;在 android 下无效怎么处理？

如果需要手动写动画，你认为最小时间间隔是多久，为什么？（阿里）

多数显示器默认频率是 60Hz，即 1 秒刷新 60 次，所以理论上最小间隔为 1/60＊1000ms ＝ 16.7ms

display:inline-block 什么时候会显示间隙？(携程)

移除空格、使用 margin 负值、使用 font-size:0、letter-spacing、word-spacing

png、jpg、gif 这些图片格式解释一下，分别什么时候用。有没有了解过 webp？

什么是 Cookie 隔离？（或者说：请求资源的时候不要让它带 cookie 怎么做）

如果静态文件都放在主域名下，那静态文件请求的时候都带有的 cookie 的数据提交给 server 的，非常浪费流量，
所以不如隔离开。

因为 cookie 有域的限制，因此不能跨域提交请求，故使用非主要域名的时候，请求头中就不会带有 cookie 数据，

这样可以降低请求头的大小，降低请求时间，从而达到降低整体请求延时的目的。

同时这种方式不会将 cookie 传入 Web Server，也减少了 Web Server 对 cookie 的处理分析环节，
提高了 webserver 的 http 请求的解析速度。

什么是 CSS 预处理器 / 后处理器？

- 预处理器例如：LESS、Sass、Stylus，用来预编译 Sass 或 less，增强了 css 代码的复用性，
还有层级、mixin、变量、循环、函数等，具有很方便的 UI 组件模块化开发能力，极大的提高工作效率。

- 后处理器例如：PostCSS，通常被视为在完成的样式表中根据 CSS 规范处理 CSS，让其更有效；目前最常做的
是给 CSS 属性添加浏览器私有前缀，实现跨浏览器兼容性的问题。

—— JavaScript ——

1.介绍 JavaScript 的基本数据类型。（百度 2015 面试题）

1. Number 数字类型
2. String 字符串类型
3. Boolean 布尔类型
4. Function 函数
5. Object 对象
6. Null
7. Undefined 没有定义类型

2.说说写 JavaScript 的基本规范？

1.不要在同一行声明多个变量。
2.请使用 ===/!==来比较 true/false 或者数值
3.使用对象字面量替代 new Array 这种形式
4.不要使用全局函数。
5.Switch 语句必须带有 default 分支
6.函数不应该有时候有返回值，有时候没有返回值。
7.For 循环必须使用大括号
8.If 语句必须使用大括号
9.for-in 循环中的变量 应该使用 var 关键字明确限定作用域，从而避免作用域污染。
10 使用规范有意义的变量名

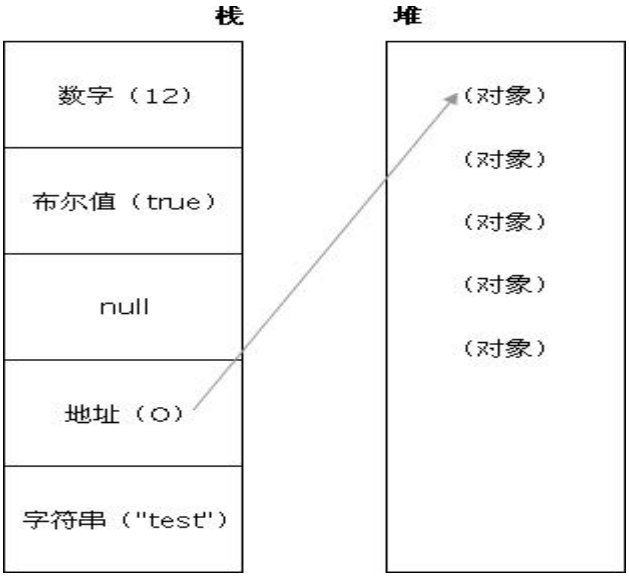
3.JavaScript 原型，原型链 ？有什么特点？

1.JS 中每个函数都存在有一个原型对象属性 prototype。并且所有函数的默认原型都是 Object 的实例。
2.每个继承父函数的子函数的对象都包含一个内部属性_proto_。该属性包含一个指针，指向父函数的 prototype。若父函数的原型对象的_proto_属性为再上一层函数。在此过程中就形成了原型链。
3.原型链实现了继承。
原型链存在两个问题：a 包含引用类型值的原型属性会被所有实例共享。

b 在创建子类型时，无法向超类型的构造函数中传递参数。
特点：JavaScript 对象是通过引用来传递的，我们创建的每个新对象实体中并没有一份属于自己的原型副本。
当我们修改原型时，与之相关的对象也会继承这一改变。当我们需要一个属性的时，Javascript 引擎会先看当前对象中是否有这个属性， 如果没有的话，就会查找他的 Prototype 对象是否有这个属性，如此递推下去，一直检索到 Object 内建对象。

4. JavaScript 有几种类型的值？（堆：原始数据类型和 栈：引用数据类型），你能画一下他们的内存图吗？（百度 2015 面试题）

基本数据类型存储在栈中，
引用数据类型（对象）存储在堆中，指针放在栈中。
两种类型的区别是：存储位置不同；原始数据类型直接存储在栈(stack)中的简单数据段，占据空间小、大小固定，属于被频繁使用数据，所以放入栈中存储；
引用数据类型存储在堆(heap)中的对象,占据空间大、大小不固定,如果存储在栈中，将会影响程序运行的性能；
引用数据类型在栈中存储了指针，该指针指向堆中该实体的起始地址。当解释器寻找引用值时，会首先检索其在栈中的地址，取得地址后从堆中获得实体。



来源：

5. Javascript 如何实现继承？

1、构造继承法
2、原型继承法
3、实例继承法
4、拷贝继承法

1、构造继承法

```
//定义一个 Collection 类型
function Collection(size)
{
    this.size = function() {return size}; //公有方法，可以被继承
}
this.base = Collection;
```

2、原型继承法：

```
function Point(dimension)  //定义一个 Point 类
{
    this.dimension = dimension;
}
Point2D.prototype = new Point(2);  //运行“原型继承法”使 Point2D 继承 Point
```

3、实例继承法

```
function MyDate()
{
    var instance = new Date();  //instance 是一个新创建的日期对象
    instance.printDate = function()  //对日期对象 instance 扩展 printDate()方法
    {
        dwn(instance.toLocaleString());
    }
    return instance;  //将 instance 作为构造函数的返回值返回
}

var date = new MyDate();
dwn(date.toGMTString());
date.printDate();
dwn(date instanceof MyDate); //false
dwn(date instanceof Date);   //true
//对象的构造函数将会是实际构造的对象的构造函数（new Date()），而不是类型本身的构造函数（new MyDate()）
```

4、拷贝继承法

拷贝继承法是通过对象属性的拷贝来实现继承。

```
<script language="JavaScript">
function Point(dimension)
{
```

```
4            this.dimension = dimension;
5        }
6
7        var Point2D = function(x,y)
8        ···{
9            this.x = x;
10           this.y = y;
11       }
12
13     Point2D.extend = function()
14     ···{
15         var p = new Point(2);
16         for(var each in p)   //将对象的属性进行一对一的复制。
17         ···{
18             //this[each] = p[each];
19             this.prototype[each] = p[each];
20         }
21     }
22     Point2D.extend();
23       //alert(Point2D.dimension);
24       alert(Point2D.prototype.dimension);
25   </script>
```

比较项	构造继承	原型继承	实例继承	拷贝继承
静态属性继承	N	Y	Y	Y
内置（核心）对象继承	N	部分	Y	Y
多参多重继承	Y	N	Y	N
执行效率	高	高	高	低
多继承	Y	N	N	Y
instanceof	false	true	false	false

6. Javascript 创建对象的几种方式？

1. 工厂方法:能创建并返回特定类型对象的工厂函数(factory function).

```
function createCar(sColor){
var car = new Object(); //  或者  var car = new Object ;
```

```
// 对象属性
car.color = sColor ;
// 对象方法
car.showColor = function (){
alert(123);
} ; // 记住，这里一定要用 ； 表示结束

return car; // 这里是 return car ； 而不是 return this.car ； 因为 this.car 为 undefined
} //
```

2. 构造函数方式

```
function Car(sColor){
this.color = sColor;
this.showColor = function(){
alert(this.color);
};
}

var car1 = new Car('red');
car1.showColor();

/*
```

你可能已经注意到第一个的差别了，在构造函数内部无创建对象，而是使用 this 关键字，使用 new
运算符调用构造函数，在执行第一行代码前先创建一个对象，只有用 this 才能访问该对象。然后可以
直接赋予 this 属性，默认情况下是构造函数的返回值，（不必明确使用 return 运算符）。
这种方式在管理函数方面与工厂方法一样都存在相同的问题.

3. 原型方式

```
function PCar(){
}

PCar.prototype.color = "blue";
var pcar1 = new PCar();

/*
```

调用 new Car()时，原型的所有属性都被立即赋予要创建的对象，意味着所有的 PCar 实例存放的是指向
showColor() 函数的指针，从语义看起来都属于一个对象，因此解决了前面两种方式存在的问题。此外使用
该方法，还能使用 instanceof 运算符检查给定变量指向的对象类型。因此下面的代码将输出 true:

```
*/
alert(pcar1 instanceof PCar); // output "true"

/*
```

这个方法看起来不错，遗憾的是，它并不尽人意。

1. 首先这个构造函数没有参数。使用原型方式时，不能给构造函数传递参数初始化属性值，因为 pcar1 和

pcar2 的属性都等于 "blue"

2. 真正的问题出现在属性指向的对象,而不是函数时，函数共享不会造成任何问题，但是对象却是很少被多个

实例共享的。

4. 混合的构造函数/原型方式(推荐)

```
/*
```

联合使用构造函数和原型方式,就可像使用其他程序设计语言一样创建对象,这种概念非常简单，即用构造函数

定义对象的所有非函数属性，用原型方式定义对象的函数属性(方法)。

```
*/

function hCar(sColor){
this.color = sColor;
this.drivers = new Array('Mike','Sue');
}

hCar.prototype.showColor = function(){
alert(this.color);
}

var hcar1 = new hCar('y color');
var hcar2 = new hCar('r color');

hcar1.drivers.push('Matt');

alert(hcar1.drivers); // output "Mike,Sue,Matt"
alert(hcar2.drivers
```

5. 动态原型方式 (推荐)

```
/*
```

对于习惯使用其他开发语言的开发者来说，使用混合构造函数/原型方式感觉不那么和谐。批评构造函数/原型方式的人

认为，在构造函数内找属性，在外部找方法的做法不合理。所以他们设计了动态原型方式，以供

更友好的编码风格。

动态原型方式的基本想法与混合构造函数/原型方式 相同，即在构造函数内定义非函数的属性，而函数的属性则利用
原型属性定义。唯一的区别是赋予对象方法的位置。下面是使用动态原型方法重写的 Car 类:
*/

```
function DCar(sColor){
this.color = sColor;
this.drivers = new Array('Mike','Sue');

if(typeof DCar._initialized == 'undefined'){

DCar.prototype.showColor = function(){
alert(this.color);
}
}

DCar._initialized = true;
}


var dcar1 = new DCar('y dcar');
var dcar2 = new DCar('b dcar');

dcar1.showColor();
dcar2.showColor();



alert(DCar._initialized); // output "true"
alert(dcar1._initialized); // output "undefined"); // output "Mike,Sue"
//
```

6、对象直接量、new 创建
创建对象最简单的方法是你的 javascript 代码中包含对象直接量，也可以通过运算符 new 创建。

```
var empty = {}; // An object with no properties
var point = { x:0, y:0 };
var circle = { x:point.x, y:point.y+1, radius:2 };
var homer = {
"name": "Homer Simpson",
"age": 34,
```

```
"married": true,
"occupation": "plant operator",
'email': homer@example.com
};
var a = new Array(); // Create an empty array
var d = new Date(); // Create an object representing the current date and time
var r = new RegExp("javascript", "i"); // Create a pattern-matching object
```

创建对象后，我们可以通过"."运算符，在对象中创建新属性、引用已有属性、设置属性值等。

```
var book = new Object();         //创建对象
book.title="JavaScript: The Definitive Guide";
book.chapter1=new Object(); //作为对象属性的，嵌套对象
book.chapter1.title = "Introduction to JavaScript";
book.chapter1.pages = 11;
book.chapter2 = { title: "Lexical Structure", pages: 6 };
alert("Outline: " + book.title + "\n\t" +
        "Chapter 1 " + book.chapter1.title + "\n\t" +
        "Chapter 2 " + book.chapter2.title); //  从对象中读取一些属性.
```

在上例中，需注意，可以通过把一个值赋给对象的一个新属性来创建它.

7.谈谈 This 对象的理解。

this 是 js 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

8.eval 是做什么的？

eval() 函数可计算某个字符串，并执行其中的的 JavaScript 代码

```
eval("x=10;y=20;document.write(x*y)")
document.write(eval("2+2"))
```

9.什么是 window 对象？什么是 document 对象？

Window -- 代表浏览器中一个打开的窗口：

document 对象 -- 代表整个 HTML 文档,可用来访问页面中的所有元素：

10.null，undefined 的区别？

Undefined 类型只有一个值，即 undefined。当声明的变量还未被初始化时，变量的默认值为 undefined。
Null 类型也只有一个值，即 null。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

11.写一个通用的事件侦听器函数(机试题)。（百度 2015 面试题）

```
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;
    }
};
```

12.["1","2","3"].map(parseInt) 答案是多少？（百度 2015 面试题）

1 Nan Nan

13.关于事件，IE 与火狐的事件机制有什么区别？ 如何阻止冒泡？（百度 2015 面试题）

响应顺序
ie 从下到上 body
firefox 从上到下
停止冒泡：
firefox：
e.stopPropagation();
ie：
window.event.cancelBubble = true;

14 什么是闭包（closure），为什么要用它？（腾讯 2015 面试题）

函数作为返回值，相关参数和变量都保存在返回的函数中，
借助闭包，同样可以封装一个私有变量.
，闭包就是携带状态的函数，并且它的状态可以完全对外隐藏起来。
返回一个函数然后延迟执行.

15.javascript 代码中的"use strict";是什么意思 ？使用它区别是什么？

严格模式

use strict”是一个指令，指示解释器用更严格的方式检查代码。

16.如何判断一个对象是否属于某个类？

1、typeof 操作符。对于 Function、String、Number、Undefined 这几种类型的对象来说，不会有什么问题，但是针对 Array 的对象就没什么用途了： alert(typeof null); // "object" alert(typeof []); // "object"

2、instanceof 操作符。此操作符检测对象的原型链是否指向构造函数的 prototype 对象，恩，听起来不错，应该可以解决我们的数组检测问题： var arr = []; alert(arr instanceof Array); // true

3、对象的 constructor 属性。除了 instanceof，我们还可以利用每个对象都具有 constructor 的属性来判断其类型，于是乎我们可以这样做： var arr = []; alert(arr.constructor == Array); //

17.new 操作符具体干了什么呢?（腾讯 2015 面试题）

(1) 首先，当我们使用 new 操作符时，js 会先创建一个空的对象;
(2) 然后，构造函数中的 this 指向该空对象;
(3) 其次，在构造函数中通过操作 this，来给这个空对象赋予相应的属性;
(4) 最后，返回这个经过处理的"空对象"(此时，对象已经不是空的了).

18.对 JSON 的了解？（百度 2015 面试题）

JavaScript 对象表示法（JavaScript Object Notation）。
JSON 是存储和交换文本信息的语法。类似 XML。
JSON 比 XML 更小、更快，更易解析。
JSON 值
JSON 值可以是：
数字（整数或浮点数）
字符串（在双引号中）
逻辑值（true 或 false）
数组（在方括号中）
对象（在花括号中）
null

19.js 延迟加载的方式有哪些？（腾讯 2015 面试题）

1、延迟加载 js 代码使用 setTimeout 延迟方法的加载时间

```
<script type=”text/javascript” src=”" id=”my”></script>
<script type=”text/javascript”>
```

```
setTimeout(“ document.getElementById(‘ my').src='include/php100.php';
“,3000);//延时 3 秒
</script>
```

2.让 js 最后加载
3.
例如引入外部 js 脚本文件时，如果放入 html 的 head 中,则页面加载前该 js 脚本就会被加载入页面,而放入 body 中,则会按照页面从上倒下的加载顺序来运行 javascript 的代码~~~ 所以我们可以把 js 外部引入的文件放到页面底部，来让 js 最后引入，从而加快页面加载速度.

20.同步和异步的区别?

javascript 异步表示 async，指：代码执行不按顺序，‘跳过’执行，待其他某些代码执行完后，再来执行，称为“异步”。

javascript 同步表示 sync，指：代码依次执行。

21.如何解决跨域问题?（百度 2015 笔试题）

1> document.domain + iframe （只有在主域相同的时候才能使用该方法）
2> 动态创建 script
3> location.hash + iframe
4> CORS
5> JSONP
JSONP 包含两部分：回调函数和数据。
回调函数是当响应到来时要放在当前页面被调用的函数。
数据就是传入回调函数中的 json 数据，也就是回调函数的参数了。

22.DOM 操作——怎样添加、移除、移动、复制、创建和查找节点?

添加：

```
haskell = document.createElement('p');
haskell.id = 'haskell';
haskell.innerText = 'Haskell';
list.appendChild(haskell);
```

insertBefore
创建新节点

```
      createDocumentFragment()    //创建一个 DOM 片段
移除
removeChild()
查找：
getElementById()
document.getElementsByClassName("class");//通过 class 查找，返回节点数组
document.getElementsByTagName("div");
复制：
//复制节点
var  cEle  =  oldEle.cloneNode(true);//深度复制，复制节点下面所有的子节点
cEle  =  oldEle.cloneNode(false);//只复制当前节点，不复制子节点

//移动节点
var  cloneEle  =  oldEle.cloneNode(true);//被移动的节点
document.removeChild(oldEle);//删除原节点
document.insertBefore(cloneEle,newEle);//插入到目标节点之前
//替换
ele.replaceChild(newEle,oldEle);
```

23.call() 和 .apply() 的含义和区别？（腾讯 2015 面试题）

call 方法:
语法：call(thisObj，Object)
定义：调用一个对象的一个方法，以另一个对象替换当前对象。
说明：
call 方法可以用来代替另一个对象调用一个方法。call 方法可将一个函数的对象上下文从初始的上下文改变为由 thisObj 指定的新对象。
如果没有提供 thisObj 参数，那么 Global 对象被用作 thisObj。

apply 方法：
语法：apply(thisObj，[argArray])
定义：应用某一对象的一个方法，用另一个对象替换当前对象。
如果 argArray 不是一个有效的数组或者不是 arguments 对象,那么将导致一个 TypeError。
如果没有提供 argArray 和 thisObj 任何一个参数，那么 Global 对象将被用作 thisObj， 并且无法被传递任何参数

```
function Animal(name) {
    this.name = name;
    this.showName = function() {
        console.log(this.name);
    };
}
function Cat(name) {
```

```
    Animal.call(this, name);
}
Cat.prototype = new Animal();
function Dog(name) {
    Animal.apply(this, name);
}
Dog.prototype = new Animal();

var cat = new Cat("Black Cat"); //call 必须是 object
var dog = new Dog(["Black Dog"]); //apply 必须是 array
cat.showName();
dog.showName();
console.log(cat instanceof Animal);
console.log(dog instanceof Animal);
```

24.JS 怎么实现一个类。怎么实例化这个类

在 Javascript 中，一切都是对象，包括函数。在 Javascript 中并没有真正的类，不能像 C#,PHP 等语言中用 class xxx 来定义。但 Javascript 中提供了一种折中的方案：把对象定义描述为对象的配方（先看一下例子会比较容易理解）。

定义类的方法有很多种，这里有两中较为通用的方法，大家参考一下。

这两种方法均可以解决构造函数会重复生成函数，为每个对象都创建独立版本的函数的问题。

解决了重复初始化函数和函数共享的问题。

1、混合的构造函数/原型方式

```
//混合的构造函数/原型方式
//创建对象
function Card(sID,ourName){
        this.ID = sID;
        this.OurName = ourName;
        this.Balance = 0;
}
Card.prototype.SaveMoney = function(money){
        this.Balance += money;
};
Card.prototype.ShowBalance = function(){
        alert(this.Balance);
};
//使用对象
var cardAA = new Card(1000,'james');
var cardBB = new Card(1001,'sun');
cardAA.SaveMoney(30);
```

```
cardBB.SaveMoney(80);
cardAA.ShowBalance();
cardBB.ShowBalance();
```

2、动态原型方法

```
//动态原型方法
//创建对象
function Card(sID,ourName){
	this.ID = sID;
	this.OurName = ourName;
	this.Balance = 0;
	if(typeof Card._initialized == "undefined"){
		Card.prototype.SaveMoney = function(money){
			this.Balance += money;
		};
		Card.prototype.ShowBalance = function(){
			alert(this.Balance);
		};
		Card._initialized = true;
	}
}
//使用对象
var cardAA = new Card(1000,'james');
var cardBB = new Card(1001,'sun');
cardAA.SaveMoney(30);
cardBB.SaveMoney(80);
cardAA.ShowBalance();
```

25.JavaScript 中的作用域与变量声明提升？

1.全局作用域;2 函数作用域（立即执行时）；3:eval 作用域；切记无块级作用域。
它会先扫描整个函数体的语句，把所有申明的变量“提升”到函数顶部

26 如何编写高性能的 Javascript？

1.尽量不要用 for-in 循环去访问数组，建议用 for 循环进行循环：
2.建议将对象进行缓存处理，特别是 DOM 访问是比较消耗资源的：
3.建议不要在函数内进行过深的嵌套判断：
4.建议避免在函数内返回一个未声明的变量，会污染外部变量：

27.jquery 中如何将数组转化为 json 字符串,然后再转化回来？

serializeArray

JSON.stringify 把一个对象转换成 json 字符串，
JSON.parse 把一个 json 字符串解析成对象

28.jquery.extend 与 jquery.fn.extend 的区别？

jQuery.extend(): 把两个或者更多的对象合并到第一个当中。扩展实例方法；jQuery.fn.extend(): 把对象挂载到 jQuery 的 prototype 属性，来扩展一个新的 jQuery 实例方法,实例化的对象才能用。扩展静态方法；jQuery.extend()是直接用$调用，jQuery.fn.extend()是用$().调用，另外 jQuery.extend()接收多个对象作为参数，如果只有一个参数，则把这个对象的属性方法附加到 jQuery 上，如果含有多个参数，则把后面的对象的属性和方法附加到第一个对象上

29.谈一下 Jquery 中的 bind(),live(),delegate(),on()的区别？

$("a").bind("click",function(){alert("ok");});
live(type,[data],fn) 给所有匹配的元素附加一个事件处理函数，即使这个元素是以后再添加进来的
$("a").live("click",function(){alert("ok");});
delegate(selector,[type],[data],fn) 指定的元素（属于被选元素的子元素）添加一个或多个事件处理程序，并规定当这些事件发生时运行的函数
$("#container").delegate("a","click",function(){alert("ok");})
on(events,[selector],[data],fn) 在选择元素上绑定一个或多个事件的事件处理函数
差别：
.bind()是直接绑定在元素上
.live()则是通过冒泡的方式来绑定到元素上的。更适合列表类型的，绑定到 document DOM 节点上。和.bind()的优势是支持动态数据。
.delegate()则是更精确的小范围使用事件代理，性能优于.live()
.on()则是最新的 1.9 版本整合了之前的三种方式的新事件绑定机制

30.针对 jQuery 性能的优化方法？（百度 2015 面试题）

使用最新版本的 jQuery

用对选择器

理解子元素和父元素的关系

这条是最快的语句。.find()方法会调用浏览器的原生方法

不要过度使用 jQuery

做好缓存

选择作用域链最短的方法

31.Jquery 与 jQuery UI 有啥区别？

Jquery 是一个前端的主开发库，jQuery UI 是基于主库基础之上开发的专门针对页面 UI 效果的框架，jQuery UI 必须在 Jquery 的基础之上，否则无法实现相关的功能。

1、 jQuery 是一个 js 库，主要提供的功能是选择器，属性修改和事件绑定等等。

2、jQuery UI 则是在 jQuery 的基础上，利用 jQuery 的扩展性，设计的插件。提供了一些常用的界面元素，

诸如对话框、拖动行为、改变大小行为等等。

3、jQuery 本身注重于后台，没有漂亮的界面，而 jQuery UI 则补充了前者的不足，他提供了华丽的展示界面，使人更容易接受。既有强大的后台，又有华丽的前台。

32.jQuery 和 Zepto 的区别？各自的使用场景

jQuery 是在 Web 上应用很广泛的 JavaScript 库，它提供了大量的工具和 API 函数，使用它的人相当普遍，使用的门槛也比较低.

Zepto 最初是为移动端开发的库，是 jQuery 的轻量级替代品，因为它的 API 和 jQuery 相似，而文件更小，对任何项目都是个不错的选择。Zepto 最大的优势是它的文件大小，只有 8k 多一点，是目前功能完备的库中最小的一个，尽管不大，Zepto 所提供的工具足以满足开发程序的需要。大多数在 jQuery 中·常用的 API 和和方法 Zepto 都有，Zepto 中还有一些 jQuery 中没有的，而用一些定制的 JavaScript 就很容易做出来的 API 和方法。另外，因为 Zepto 的 API 大部分都能和 jQuery 兼容.

针对移动端程序，Zepto 还有一些基本的触摸事件可以用来做触摸屏交互,但是有一点，Zepto 是不支持 IE 浏览器的

33.我们给一个 dom 同时绑定两个点击事件，一个用捕获，一个用冒泡，你来说下会执行几次事件，然后会先执行冒泡还是捕获

从上往下，如有捕获事件，则执行；一直向下到目标元素后，从目标元素开始向上执行冒泡元素，即第三个参数为 false 的绑定事件的元素。(在向上执行过程中，已经执行过的捕获事件不再执行，只执行冒泡事件。)