Html 篇：

1.你做的页面在哪些流览器测试过？这些浏览器的内核分别是什么？

- 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)

2.每个 HTML 文件里开头都有个很重要的东西，Doctype，知道这是干什么的吗？

答案：<!DOCTYPE> 声明位于文档中的最前面的位置，处于 <html> 标签之前。此标签可告知浏览器文档使用哪种 HTML 或 XHTML 规范。（重点：告诉浏览器按照何种规范解析页面）

3.Quirks 模式是什么？它和 Standards 模式有什么区别

答案：

从 IE6 开始，引入了 Standards 模式，标准模式中，浏览器尝试给符合标准的文档在规范上的正确处理达到在指定浏览器中的程度。

在 IE6 之前 CSS 还不够成熟，所以 IE5 等之前的浏览器对 CSS 的支持很差， IE6 将对 CSS 提供更好的支持，然而这时的问题就来了，因为有很多页面 是基于旧的布局方式写的，而如果 IE6 支持 CSS 则将令这些页面显示不正常，如何在即保证不破坏现有页面，又提供新的渲染机制呢？

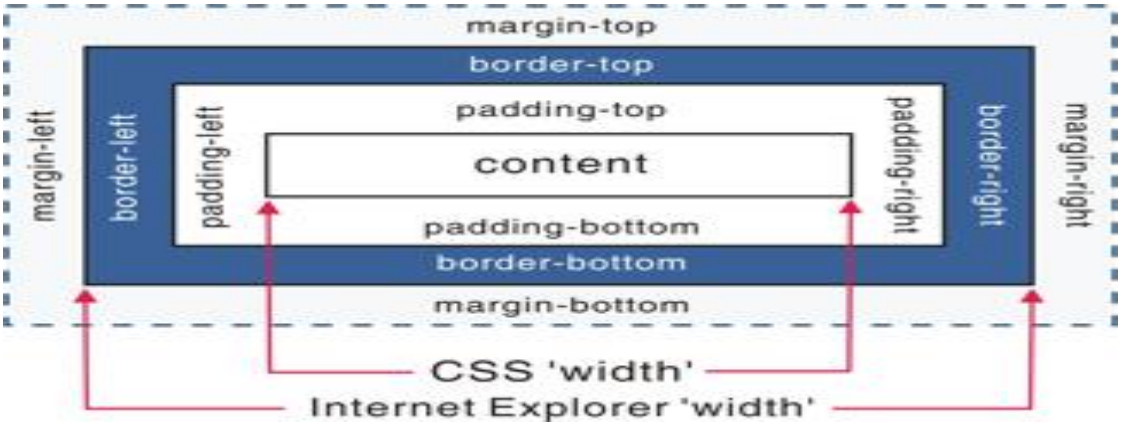
在写程序时我们也会经常遇到这样的问题，如何保证原来的接口不变，又提供更强大的功能，尤其是新功能不兼容旧功能时。遇到这种问题时的一个常见做法是增加 参数和分支，即当某个参数为真时，我们就使用新功能，而如果这个参数 不为真时，就使用旧功能，这样就能不破坏原有的程序，又提供新功能。IE6 也是类似 这样做的，它将 DTD 当成了这个“参数”，因为以前的页面大家都不会去

写 DTD，所以 IE6 就假定 如果写了 DTD，就意味着这个页面将采用对 CSS 支持 更好的布局，而如果没有，则采用兼容之前的布局方式。这就是 Quirks 模式（怪癖模式，诡异模式，怪异模式）。

区别：

总体会有布局、样式解析和脚本执行三个方面的区别。

盒模型：在 W3C 标准中，如果设置一个元素的宽度和高度，指的是元素内容的宽度和高度，而在 Quirks 模式下，IE 的宽度和高度还包含了 padding 和 border。



设置行内元素的高宽：在 Standards 模式下，给等行内元素设置 wdith 和 height 都不会生效，而在 quirks 模式下，则会生效。

设置百分比的高度：在 standards 模式下，一个元素的高度是由其包含的内容来决定的，如果父元素没有设置百分比的高度，子元素设置一个百分比的高度 是无效的用 margin:0 auto 设置水平居中：使用 margin:0 auto 在 standards 模式下可以使元素水平居中，但在 quirks 模 式下却会失效。

（还有很多，答出什么不重要，关键是看他答出的这些是不是自己经验遇到的，还是说都是看文章看的，甚至完全不知道。）

4.div+css 的布局较 table 布局有什么优点？

- 改版的时候更方便 只要改 css 文件。
- 页面加载速度更快、结构化清晰、页面显示简洁。

- 表现与结构相分离。
- 易于优化（seo）搜索引擎更友好，排名更容易靠前。

5.a：img 的 alt 与 title 有何异同？b：strong 与 em 的异同？

答案：

a:

- alt(alt text):为不能显示图像、窗体或 applets 的用户代理（UA），alt 属性用来指定替换文字。

  替换文字的语言由 lang 属性指定。(在 IE 浏览器下会在没有 title 时把 alt 当成 tool tip 显示)

- title(tool tip):该属性为设置该属性的元素提供建议性的信息。

b:

- strong:粗体强调标签，强调，表示内容的重要性
- em:斜体强调标签，更强烈强调，表示内容的强调点

6.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。

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

（自由发挥）

7.为什么利用多个域名来存储网站资源会更有效？

- CDN 缓存更方便
- 突破浏览器并发限制
- 节约 cookie 带宽
- 节约主域名的连接数，优化页面响应速度
- 防止不必要的安全问题

8.请谈一下你对网页标准和标准制定机构重要性的理解。

（无标准答案）网页标准和标准制定机构都是为了能让 web 发展的更‘健康’，开发者遵循统一的标准，降低开发难度，开发成本，SEO 也会更好做，也不会因为滥用代码导致各种 BUG、安全问题，最终提高网站易用性。

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

sessionStorage 用于本地存储一个会话（session）中的数据，这些数据只有在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。而 localStorage 用于持久化的本地存储，除非主动删除 数据，否则数据是永远不会过期的。

web storage 和 cookie 的区别

- Web Storage 的概念和 cookie 相似，区别是它是为了更大容量存储设计的。Cookie 的大小是受限的，并且每次你请求一个新的页面的时候 Cookie 都会被发送过去，这样无形中浪费了带宽，另外 cookie 还需要指定作用域，不可以跨域调用。

- 除 此之外，Web Storage 拥有 setItem,getItem,removeItem,clear 等方法，不像 cookie 需要前端开发者自己封装 setCookie，getCookie。但是 Cookie 也是不可以或缺的：Cookie 的作用是与服务器进行交互，作为 HTTP 规范的一部分而存 在 ，而 Web Storage 仅仅是为了在本地 “存储” 数据而生。

10.简述一下 src 与 href 的区别。

答案：

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。

<script src =” js.js” ></script>

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

<link href=”common.css” rel=”stylesheet”/>

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。

11.知道的网页制作会用到的图片格式有哪些？

答案：

png-8，png-24，jpeg，gif，svg。

但是上面的那些都不是面试官想要的最后答案。面试官希望听到是 Webp,Apng。（是否有关注新技术，新鲜事物）

科普一下 Webp：WebP 格式，谷歌（google）开发的一种旨在加快图片加载速度的图片格式。图片压缩体积大约只有 JPEG 的 2/3，并能节省大量的服务器带宽资源和数据空间。Facebook Ebay 等知名网站已经开始测试并使用 WebP 格式。

在质量相同的情况下，WebP 格式图像的体积要比 JPEG 格式图像小 40%。

Apng：全称是“Animated Portable Network Graphics”，是 PNG 的位图动画扩展，可以实现 png 格式的动态图片效果。04 年诞生，但一直得不到各大浏览器厂商的支持，直到日前得到 iOS safari 8 的支持，有望代替 GIF 成为下一代动态图标准。

12.知道什么是微格式吗？谈谈理解。在前端构建中应该考虑微格式吗？

答案：

微格式（Microformats）是一种让机器可读的语义化 XHTML 词汇的集合，是结构化数据的开放标准。是为特殊应用而制定的特殊格式。

优点：将智能数据添加到网页上，让网站内容在搜索引擎结果界面可以显示额外的提示。（应用范例：豆瓣，有兴趣自行 google）

13.在 css/js 代码上线之后开发人员经常会优化性能，从用户刷新网页开始，一次 js 请求一般情况下有哪些地方会有缓存处理？

答案：dns 缓存，cdn 缓存，浏览器缓存，服务器缓存。

14.一个页面上有大量的图片（大型电商网站），加载很慢，你有哪些方法优化这些图片的加载，给用户更好的体验。

- 图片懒加载，在页面上的未可视区域可以添加一个滚动条事件，判断图片位置与浏览器顶端的距离与页面的距离，如果前者小于后者，优先加载。
- 如果为幻灯片、相册等，可以使用图片预加载技术，将当前展示图片的前一张和后一张优先下载。
- 如果图片为 css 图片，可以使用 CSSsprite，SVGsprite，Iconfont、Base64 等技术。
- 如果图片过大，可以使用特殊编码的图片，加载时会先加载一张压缩的特别厉害的缩略图，以提高用户体验。
- 如果图片展示区域小于图片的真实大小，则因在服务器端根据业务需要先行进行图片压缩，图片压缩后大小与展示一致。

15.你如何理解 HTML 结构的语义化？

- 去掉或样式丢失的时候能让页面呈现清晰的结构：

html 本身是没有表现的，我们看到例如<h1>是粗体，字体大小 2em，加粗；是加粗的，不要认为这是 html 的表现，这些其实 html 默认的 css 样式在起作用，所以去掉或样式丢失的时候能让页面呈现清晰的结构不是语义化的 HTML 结构的优点，但是浏览 器都有有默认样式，默认样式的目的也是为了更好的表达 html 的语义，可以说浏览器的默认样式和语义化的 HTML 结构是不可分割的。

- 屏幕阅读器（如果访客有视障）会完全根据你的标记来“读”你的网页.

例如,如果你使用的含语义的标记,屏幕阅读器就会“逐个拼出”你的单词,而不是试着去对它完整发音.

- PDA、手机等设备可能无法像普通电脑的浏览器一样来渲染网页（通常是因为这些设备对 CSS 的支持较弱）

使用语义标记可以确保这些设备以一种有意义的方式来渲染网页.理想情况下,观看设备的任务是符合设备本身的条件来渲染网页.

语义标记为设备提供了所需的相关信息,就省去了你自己去考虑所有可能的显示情况（包括现有的或者将来新的设备）.例如,一部手机可以选择使一段标记了标题 的文字以粗体显示.而掌上电脑可能会以比较大的字体来显示.无论哪种方式一旦你对文本标记为标题,您就可以确信读取设备将根据其自身的条件来合适地显示页 面.

- 搜索引擎的爬虫也依赖于标记来确定上下文和各个关键字的权重

过去你可能还没有考虑搜索引擎的爬虫也是网站的“访客”,但现在它们他们实际上是极其宝贵的用户.没有他们的话,搜索引擎将无法索引你的网站,然后一般用户将很难过来访问.

- 你的页面是否对爬虫容易理解非常重要,因为爬虫很大程度上会忽略用于表现的标记,而只注重语义标记.

因此,如果页面文件的标题被标记,而不是,那么这个页面在搜索结果的位置可能会比较靠后.除了提升易用性外,语义标记有利于正确使用 CSS 和 JavaScript,因为其本身提供了许多“钩钩”来应用页面的样式与行为.

SEO 主要还是靠你网站的内容和外部链接的。

- 便于团队开发和维护

W3C 给我们定了一个很好的标准，在团队中大家都遵循这个标准，可以减少很多差异化的东西，方便开发和维护，提高开发效率，甚至实现模块化开发。

16.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（SE robot 或叫 web crawler）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

Open Directory 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 Google Adwords。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（Link Popularity）

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

Css 篇：

1.有哪项方式可以对一个 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 <head> 标签内部
- 内联样式，将 css 样式直接定义在 HTML 元素内部

2.CSS 都有哪些选择器？

- 派生选择器（用 HTML 标签申明）
- id 选择器（用 DOM 的 ID 申明）
- 类选择器（用一个样式类名申明）
- 属性选择器（用 DOM 的属性申明，属于 CSS2，IE6 不支持，不常用，不知道就算了）

除了前 3 种基本选择器，还有一些扩展选择器，包括

- 后代选择器（利用空格间隔，比如 div .a{ }）
- 群组选择器（利用逗号间隔，比如 p,div,#a{ }）

那么问题来了，CSS 选择器的优先级是怎么样定义的？

基本原则：

一般而言，选择器越特殊，它的优先级越高。也就是选择器指向的越准确，它的优先级就越高。

复杂的计算方法：

- 用 1 表示派生选择器的优先级
- 用 10 表示类选择器的优先级
- 用 100 标示 ID 选择器的优先级
    - div.test1 .span var 优先级 1+10 +10 +1
    - span#xxx .songs li 优先级 1+100 + 10 + 1
    - #xxx li 优先级 100 +1

那么问题来了，看下列代码，<p>标签内的文字是什么颜色的？。

```
<style>
.classA{ color:blue;}

.classB{ color:red;}
</style>

<body>
```

```
 8
 9 <p class='classB classA'> 123 </p>
10
11 </body>
```

答案：red。与样式定义在文件中的先后顺序有关，即是后面的覆盖前面的，与在<p class=’ classB classA’ >中的先后关系无关。

3.CSS 中可以通过哪些属性定义，使得一个 DOM 元素不显示在浏览器可视范围内？

最基本的：

设置 display 属性为 none，或者设置 visibility 属性为 hidden

技巧性：

设置宽高为 0，设置透明度为 0，设置 z-index 位置在-1000

4.超链接访问过后 hover 样式就不出现的问题是什么？如何解决？

答案：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（link,visited,hover,active）

5.什么是 Css Hack？ie6,7,8 的 hack 分别是什么？

答案：针对不同的浏览器写不同的 CSS code 的过程，就是 CSS hack。

示例如下：

```
1  #test      {
2        width:300px;
3        height:300px;
4
5        background-color:blue;      /*firefox*/
```

```
6        background-color:red\9;     /*all ie*/
7        background-color:yellow\0;    /*ie8*/
8        +background-color:pink;       /*ie7*/
9        _background-color:orange;      /*ie6*/    }
10       :root #test { background-color:purple\9; }  /*ie9*/
11    @media all and (min-width:0px){ #test {background-color:black\0;} }  /*opera*/
12    @media screen and (-webkit-min-device-pixel-ratio:0){ #test {background-color:gra
y;} }      /*chrome and safari*/
```

6.请用 Css 写一个简单的幻灯片效果页面

答案：知道是要用 css3。使用 animation 动画实现一个简单的幻灯片效果。

更多资料欢迎加群领取！！！

```
1        /**HTML**/
2        div.ani
3
4        /**css**/
5        .ani{
```

```
 6          width:480px;
 7          height:320px;
 8          margin:50px auto;
 9          overflow: hidden;
10          box-shadow:0 0 5px rgba(0,0,0,1);
11          background-size: cover;
12          background-position: center;
13          -webkit-animation-name: "loops";
14          -webkit-animation-duration: 20s;
15          -webkit-animation-iteration-count: infinite;
16        }
17        @-webkit-keyframes "loops" {
18            0% {
19                background:url(http://d.hiphotos.baidu.com/image/w%3D400/sign=c01e6adc
a964034f0fcdc3069fc27980/e824b899a9014c08e5e38ca4087b02087af4f4d3.jpg) no-repeat;
20            }
21            25% {
22                background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=edee1572
e9f81a4c2632edc9e72b6029/30adcbef76094b364d72bceba1cc7cd98c109dd0.jpg) no-repeat;
23            }
24            50% {
25                background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=937dace2
552c11dfded1be2353266255/d8f9d72a6059252d258e7605369b033b5bb5b912.jpg) no-repeat;
26            }
27            75% {
28                background:url(http://g.hiphotos.baidu.com/image/w%3D400/sign=7d37500b
8544ebf86d71653fe9f9d736/0df431adcbef76095d61f0972cdda3cc7cd99e4b.jpg) no-repeat;
29            }
```

```
30          100% {

31              background:url(http://c.hiphotos.baidu.com/image/w%3D400/sign=cfb239ce
b0fb43161a1f7b7a10a54642/3b87e950352ac65ce2e73f76f9f2b21192138ad1.jpg) no-repeat;

32          }

33      }
```

7.行内元素和块级元素的具体区别是什么？行内元素的 padding 和 margin 可设置吗？

块级元素(block)特性：

- 总是独占一行，表现为另起一行开始，而且其后的元素也必须另起一行显示;
- 宽度(width)、高度(height)、内边距(padding)和外边距(margin)都可控制;

内联元素(inline)特性：

- 和相邻的内联元素在同一行;
- 宽 度(width)、高度(height)、内边距的 top/bottom(padding-top/padding-bottom)和外边距的 top /bottom(margin-top/margin-bottom)都不可改变（也就是 padding 和 margin 的 left 和 right 是可以设置 的），就是里面文字或图片的大小。

那么问题来了，浏览器还有默认的天生 inline-block 元素（拥有内在尺寸，可设置高宽，但不会自动换行），有哪些？
答案：<input> 、<img> 、<button> 、<texterea> 、<label>。

8.什么是外边距重叠？重叠的结果是什么？

答案：

外边距重叠就是 margin-collapse。

在 CSS 当中，相邻的两个盒子（可能是兄弟关系也可能是祖先关系）的外边距可以结合成一个单独的外边距。这种合并外边距的方式被称为折叠，并且因而所结合成的外边距称为折叠外边距。

折叠结果遵循下列计算规则：

1. 两个相邻的外边距都是正数时，折叠结果是它们两者之间较大的值。
2. 两个相邻的外边距都是负数时，折叠结果是两者绝对值的较大值。
3. 两个外边距一正一负时，折叠结果是两者的相加的和。

9.rgba()和 opacity 的透明效果有什么不同？

答案：

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！）

10.css 中可以让文字在垂直和水平方向上重叠的两个属性是什么？

答案：

垂直方向：line-height

水平方向：letter-spacing

那么问题来了，关于 letter-spacing 的妙用知道有哪些么？

答案:可以用于消除 inline-block 元素间的换行符空格间隙问题。

11.如何垂直居中一个浮动元素？

```
1 // 方法一：已知元素的高宽
2
3 #div1{
4     background-color:#6699FF;
5     width:200px;
6     height:200px;
7
8     position: absolute;        //父元素需要相对定位
```

```
 9    top: 50%;
10    left: 50%;
11    margin-top:-100px ;   //二分之一的height，width
12    margin-left: -100px;
13    }
14
15 //方法二:未知元素的高宽
16
17   #div1{
18    width: 200px;
19    height: 200px;
20    background-color: #6699FF;
21
22    margin:auto;
23    position: absolute;       //父元素需要相对定位
24    left: 0;
25    top: 0;
26    right: 0;
27    bottom: 0;
28    }
```

那么问题来了，如何垂直居中一个<img>?（用更简便的方法。）

```
1 #container     //<img>的容器设置如下
2 {
3    display:table-cell;
4    text-align:center;
5    vertical-align:middle;
```

```
6 }
```

12.px 和 em 的区别。

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。

13.描述一个”reset”的 CSS 文件并如何使用它。知道 `normalize.css` 吗？你了解他们的不同之处？

重置样式非常多，凡是一个前端开发人员肯定有一个常用的重置 CSS 文件并知道如何使用它们。他们是盲目的在做还是知道为什么这么做呢？原因是不同的浏览器对一些元素有不同的默认样式，如果你不处理，在不同的浏览器下会存在必要的风险，或者更有戏剧性的性发生。

你可能会用 Normalize 来代替你的重置样式文件。它没有重置所有的样式风格，但仅提供了一套合理的默认样式值。既能让众多浏览器达到一致和合理，但又不扰乱其他的东西（如粗体的标题）。

在这一方面，无法做每一个复位重置。它也确实有些超过一个重置，它处理了你永远都不用考虑的怪癖，像 HTML 的 `audio` 元素不一致或 `line-height` 不一致。

14.Sass、LESS 是什么？大家为什么要使用他们？

他们是 CSS 预处理器。他是 CSS 上的一种抽象层。他们是一种特殊的语法/语言编译成 CSS。

例如 Less 是一种动态样式语言. 将 CSS 赋予了动态语言的特性，如变量，继承，运算， 函数. LESS 既可以在客户端上运行 (支持 IE 6+, Webkit, Firefox)，也可一在服务端运行 (借助 Node.js)。

为什么要使用它们?

- 结构清晰，便于扩展。
- 可以方便地屏蔽浏览器私有语法差异。这个不用多说，封装对浏览器语法差异的重复处理，减少无意义的机械劳动。
- 可以轻松实现多重继承。

- 完全兼容 CSS 代码，可以方便地应用到老项目中。LESS 只是在 CSS 语法上做了扩展，所以老的 CSS 代码也可以与 LESS 代码一同编译。

15.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。

16.知道 css 有个 content 属性吗？有什么作用？有什么应用？

答案：

知道。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。

```
1 //一种常见利用伪类清除浮动的代码
2  .clearfix:after {
3     content:".";       //这里利用到了 content 属性
4     display:block;
5     height:0;
6     visibility:hidden;
7     clear:both; }
8
9 .clearfix {
10     *zoom:1;
11 }
```

after 伪元素通过 content 在元素的后面生成了内容为一个点的块级元素，再利用 clear:both 清除浮动。

那么问题继续还有，知道 css 计数器（序列数字字符自动递增）吗？如何通过 css content 属性实现 css 计数器？

答案：css 计数器是通过设置 counter-reset 、counter-increment 两个属性 、及 counter()/counters()一个方法配合 after / before 伪类实现。

初级 Javascript：

1.JavaScript 是一门什么样的语言，它有哪些特点？

没有标准答案。

2.JavaScript 的数据类型都有什么？

基本数据类型：String,boolean,Number,Undefined, Null

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

那么问题来了，如何判断某变量是否为数组数据类型？

- 方法一.判断其是否具有“数组性质”，如 slice()方法。可自己给该变量定义 slice 方法，故有时会失效
- 方法二.obj instanceof Array 在某些 IE 版本中不正确
- 方法三.方法一二皆有漏洞，在 ECMA Script5 中定义了新方法 Array.isArray()，保证其兼容性，最好的方法如下：

```
1 if(typeof Array.isArray==="undefined")

2 {

3   Array.isArray = function(arg){

4       return Object.prototype.toString.call(arg)==="[object Array]"
```

```
5    };
6 }
```

3.已知 ID 的 Input 输入框，希望获取这个输入框的输入值，怎么做？(不使用第三方框架)

```
1 document.getElementById("ID").value
```

4.希望获取到页面中所有的 checkbox 怎么做？(不使用第三方框架)

```
1 var domList = document.getElementsByTagName('input')
2 var checkBoxList = [];
3 var len = domList.length;    //缓存到局部变量
4 while (len--) {    //使用 while 的效率会比 for 循环更高
5    if (domList[len].type == 'checkbox') {
6        checkBoxList.push(domList[len]);
7    }
8 }
```

5.设置一个已知 ID 的 DIV 的 html 内容为 xxxx，字体颜色设置为黑色(不使用第三方框架)

```
1 var dom = document.getElementById("ID");
2 dom.innerHTML = "xxxx"
3 dom.style.color = "#000"
```

6.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？

- 直接在 DOM 里绑定事件：
- 在 JS 里通过 onclick 绑定：xxx.onclick = test
- 通过事件添加进行绑定：addEventListener(xxx, 'click', test)

那么问题来了，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

7.什么是 Ajax 和 JSON，它们的优缺点。

Ajax 是异步 JavaScript 和 XML，用于在 Web 页面中实现异步数据交互。

优点：

- 可以使得页面不重载全部内容的情况下加载局部内容，降低数据传输量
- 避免用户不断刷新或者跳转页面，提高用户体验

缺点：

- 对搜索引擎不友好（
- 要实现 ajax 下的前后退功能成本较大
- 可能造成请求数的增加
- 跨域问题限制

JSON 是一种轻量级的数据交换格式，ECMA 的一个子集

优点：轻量级、易于人的阅读和编写，便于机器（JavaScript）解析，支持复合数据类型（数组、对象、字符串、数字）

8.看下列代码输出为何？解释原因。

```
1 var a;

2 alert(typeof a); // undefined

3 alert(b); // 报错
```

解释：Undefined 是一个只有一个值的数据类型，这个值就是“undefined”，在使用 var 声明变量但并未对其赋值进行初始化时，这个变量的值就是 undefined。而 b 由于未声明将报错。注意未申明的变量和声明了未赋值的是不一样的。

9.看下列代码,输出什么？解释原因。

```
1 var a = null;

2 alert(typeof a); //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”。

10.看下列代码,输出什么？解释原因。

```
1 var undefined;

2 undefined == null; // true

3 1 == true;   // true

4 2 == true;   // false

5 0 == false;  // true

6 0 == '';     // true

7 NaN == NaN;  // false

8 [] == false; // true

9 [] == ![];   // true
```

- undefined 与 null 相等，但不恒等（===）
- 一个是 number 一个是 string 时，会尝试将 string 转换为 number
- 尝试将 boolean 转换为 number，0 或 1
- 尝试将 Object 转换成 number 或 string，取决于另外一个对比量的类型

- 所以，对于 0、空字符串的判断，建议使用 “===” 。“===”会先判断两边的值类型，类型不匹配时为 false。

那么问题来了，看下面的代码，输出什么，foo 的类型为什么？

```
1 var foo = "11"+2-"1";

2 console.log(foo);

3 console.log(typeof foo);
```

执行完后 foo 的值为 111，foo 的类型为 Number。

```
1 var foo = "11"+2+"1";    //体会加一个字符串'1' 和 减去一个字符串'1'的不同

2 console.log(foo);

3 console.log(typeof foo);
```

执行完后 foo 的值为 113，foo 的类型为 String。

11.看代码给答案。

```
1 var a = new Object();

2 a.value = 1;

3 b = a;

4 b.value = 2;

5 alert(a.value);
```

答案：2（考察引用数据类型细节）

12.已知数组 var stringArray = [“This”, “is”, “Baidu”, “Campus”]，Alert 出”This is Baidu Campus”。

答案：alert(stringArray.join(“”))

已知有字符串 foo=”get-element-by-id”,写一个 function 将其转化成驼峰表示

法”getElementById”。

```
function combo(msg){
    var arr=msg.split("-");
    for(var i=1;i<arr.length;i++){
        arr[i]=arr[i].charAt(0).toUpperCase()+arr[i].substr(1,arr[i].length-1);
    }
    msg=arr.join("");
    return msg;
}
```

(考察基础 API)

13.var numberArray = [3,6,2,4,1,5]; （考察基础 API）

1) 实现对该数组的倒排，输出[5,1,4,2,6,3]

2) 实现对该数组的降序排列，输出[6,5,4,3,2,1]

```
var numberArray = [3,6,2,4,1,5];

numberArray.reverse(); // 5,1,4,2,6,3

numberArray.sort(function(a,b){  //6,5,4,3,2,1
   return b-a;
})
```

14.输出今天的日期，以 YYYY-MM-DD 的方式，比如今天是 2014 年 9 月 26 日，则输出 2014-09-26

```
var d = new Date();
// 获取年，getFullYear()返回 4 位的数字
var year = d.getFullYear();
// 获取月，月份比较特殊，0 是 1 月，11 是 12 月
var month = d.getMonth() + 1;
// 变成两位
month = month < 10 ? '0' + month : month;
// 获取日
var day = d.getDate();
day = day < 10 ? '0' + day : day;
alert(year + '-' + month + '-' + day);
```

15.将字符串” <tr><td>{$id}</td><td>{$name}</td></tr>” 中的{$id}替换成 10，{$name}替换成 Tony （使用正则表达式）

答案：” <tr><td>{$id}</td><td>{$id}_{$name}</td></tr>”.replace(/{\$id}/g, ’10’).replace(/{\$name}/g, ‘Tony’);

更多资料欢迎加群领取！！！

16.为了保证页面输出安全，我们经常需要对一些特殊的字符进行转义，请写一个函数 escapeHtml，将<, >, &, “进行转义

```
function escapeHtml(str) {
return str.replace(/[<>”&]/g, function(match) {
    switch (match) {
                case “<”:
                   return “&lt;”;
                case “>”:
                   return “&gt;”;
                case “&”:
                   return “&amp;”;
                case “\””:
                   return “&quot;”;
     }
  });
}
```

17.foo = foo||bar ，这行代码是什么意思？为什么要这样写？

答案：if(!foo) foo = bar; //如果 foo 存在，值不变，否则把 bar 的值赋给 foo。

短路表达式：作为”&&”和”||”操作符的操作数表达式，这些表达式在进行求值时，只要最终的结果已经可以确定是真或假，求值过程便告终止，这称之为短路求值。

18.看下列代码，将会输出什么?(变量声明提升)

```
var foo = 1;
```

```
2 function(){
3     console.log(foo);
4     var foo = 2;
5     console.log(foo);
6 }
```

答案：输出 undefined 和 2。上面代码相当于：

```
1 var foo = 1;
2 function(){
3     var foo;
4     console.log(foo); //undefined
5     foo = 2;
6     console.log(foo); // 2;
7 }
```

函数声明与变量声明会被 JavaScript 引擎隐式地提升到当前作用域的顶部，但是只提升名称不会提升赋值部分。

19.用 js 实现随机选取 10–100 之间的 10 个数字，存入一个数组，并排序。

```
1 var iArray = [];
2 funtion getRandom(istart, iend){
3         var iChoice = istart - iend +1;
4         return Math.floor(Math.random() * iChoice + istart;
5 }
6 for(var i=0; i<10; i++){
```

```
7         iArray.push(getRandom(10,100));

8 }

9 iArray.sort();
```

20.把两个数组合并，并删除第二个元素。

```
1 var array1 = ['a','b','c'];

2 var bArray = ['d','e','f'];

3 var cArray = array1.concat(bArray);

4 cArray.splice(1,1);
```

21.怎样添加、移除、移动、复制、创建和查找节点（原生 JS，实在基础，没细写每一步）

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() //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

22.有这样一个 URL：http://item.taobao.com /item.htm?a=1&b=2&c=&d=xxx&e，请写一段 JS 程序提取 URL 中的各个 GET 参数(参数名和 参数个数不确定)，将其按 key-value 形式返回到一个 json 结构中，如{a:’ 1’, b:’ 2’ , c:”, d:’ xxx’ , e:undefined}。

答案：

```
1 function serilizeUrl(url) {

2     var result = {};

3     url = url.split("?")[1];

4     var map = url.split("&");

5     for(var i = 0, len = map.length; i < len; i++) {

6         result[map[i].split("=")[0]] = map[i].split("=")[1];

7     }

8     return result;

9 }
```

23.正则表达式构造函数 var reg=new RegExp(“xxx”)与正则表达字面量 var reg=//有什么不同？匹配邮箱的正则表达式？

答案：当使用 RegExp()构造函数的时候，不仅需要转义引号（即\”表示”），并且还需要双反斜杠（即\\表示一个\）。使用正则表达字面量的效率更高。

邮箱的正则匹配：

```
1 var regMail = /^([a-zA-Z0-9_-])+@([a-zA-Z0-9_-])+((.[a-zA-Z0-9_-]{2,3}){1,2})$/;
```

24.看下面代码，给出输出结果。

```
1 for(var i=1;i<=3;i++){

2   setTimeout(function(){

3       console.log(i);

4   },0);
```

```
5 };
```

答案：4 4 4。

原因：Javascript 事件处理器在线程空闲之前不会运行。那么问题来了，如何让上述代码输出 1 2 3？

```
1 for(var i=1;i<=3;i++){
2    setTimeout((function(a){  //改成立即执行函数
3        console.log(a);
4    })(i),0);
5 };
6
7 1          //输出
8 2
9 3
```

25.写一个 function，清除字符串前后的空格。（兼容所有浏览器）

使用自带接口 trim()，考虑兼容性：

```
1 if (!String.prototype.trim) {
2  String.prototype.trim = function() {
3  return this.replace(/^\s+/, "").replace(/\s+$/,"");
4  }
5 }
6
7 // test the function
8 var str = " \t\n test string ".trim();
9 alert(str == "test string"); // alerts "true"
```

26.Javascript 中 callee 和 caller 的作用？

答案：

caller 是返回一个对函数的引用，该函数调用了当前函数；

callee 是返回正在被执行的 function 函数，也就是所指定的 function 对象的正文。

那么问题来了？如果一对兔子每月生一对兔子；一对新生兔，从第二个月起就开始生兔子；假定每对兔子都是一雌一雄，试问一对兔子，第 n 个月能繁殖成多少对兔子？（使用 callee 完成）

```
var result=[];
function fn(n){  //典型的斐波那契数列
   if(n==1){
        return 1;
   }else if(n==2){
          return 1;
   }else{
        if(result[n]){
              return result[n];
        }else{
              //argument.callee()表示 fn()
              result[n]=arguments.callee(n-1)+arguments.callee(n-2);
              return result[n];
        }
   }
}
```

中级 Javascript：

1.实现一个函数 clone，可以对 JavaScript 中的 5 种主要的数据类型（包括 Number、String、Object、Array、Boolean）进行值复制

- 考察点 1：对于基本数据类型和引用数据类型在内存中存放的是值还是指针这一区别是否清楚
- 考察点 2：是否知道如何判断一个变量是什么类型的
- 考察点 3：递归算法的设计

```
// 方法一：
Object.prototype.clone = function(){
        var o = this.constructor === Array ? [] : {};
        for(var e in this){
                o[e] = typeof this[e] === "object" ? this[e].clone() : this[e];
        }
        return o;
}

//方法二：
  /**
     * 克隆一个对象
     * @param Obj
     * @returns
     */
    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{                         //普通变量直接赋值
            return Obj;
        }
    }
```

2.如何消除一个数组里面重复的元素？

```
var arr=[1,2,3,3,4,4,5,5,6,1,9,3,25,4];
        function deRepeat(){
            var newArr=[];
            var obj={};
            var index=0;
            var l=arr.length;
            for(var i=0;i<l;i++){
                if(obj[arr[i]]==undefined)
                  {
                    obj[arr[i]]=1;
                    newArr[index++]=arr[i];
```

```
12                }
13              else if(obj[arr[i]]==1)
14                continue;
15           }
16           return newArr;
17
18       }
19       var newArr2=deRepeat(arr);
20       alert(newArr2); //输出1,2,3,4,5,6,9,25
```

3.小贤是一条可爱的小狗(Dog)，它的叫声很好听(wow)，每次看到主人的时候就会乖乖叫一声(yelp)。

从这段描述可以得到以下对象：

```
1 function Dog() {
2       this.wow = function() {
3              alert('Wow');
4      }
5       this.yelp = function() {
6             this.wow();
7      }
8 }
```

小芒和小贤一样，原来也是一条可爱的小狗，可是突然有一天疯了(MadDog)，一看到人就会每隔半秒叫一声(wow)地不停叫唤(yelp)。请根据描述，按示例的形式用代码来实。（继承，原型，setInterval）

答案：

```
1 function MadDog() {
2    this.yelp = function() {
3         var self = this;
```

```
        setInterval(function() {
            self.wow();
        }, 500);
    }
}
MadDog.prototype = new Dog();

//for test
var dog = new Dog();
dog.yelp();
var madDog = new MadDog();
madDog.yelp();
```

4.下面这个 ul，如何点击每一列的时候 alert 其 index?（闭包）

```
<ul id="test">
<li>这是第一条</li>
<li>这是第二条</li>
<li>这是第三条</li>
</ul>
```

答案：

```
// 方法一：
var lis=document.getElementById('2223').getElementsByTagName('li');
for(var i=0;i<3;i++)
{
    lis[i].index=i;
    lis[i].onclick=function(){
```

```
        alert(this.index);
    };
}

//方法二：
var lis=document.getElementById('2223').getElementsByTagName('li');
for(var i=0;i<3;i++)
{
    lis[i].index=i;
    lis[i].onclick=(function(a){
        return function() {
            alert(a);
        }
    })(i);
}
```

5.编写一个 JavaScript 函数，输入指定类型的选择器(仅需支持 id，class，tagName 三种简单 CSS 选择器，无需兼容组合选择器)可以返回匹配的 DOM 节点，需考虑浏览器兼容性和性能。

/*** @param selector {String} 传入的 CSS 选择器。* @return {Array}*/

答案：（过长，点击打开）

6.请评价以下代码并给出改进意见。

```
if(window.addEventListener){
    var addListener = function(el,type,listener,useCapture){
        el.addEventListener(type,listener,useCapture);
  };
}
```

```
6 else if(document.all){
7    addListener = function(el,type,listener){
8        el.attachEvent("on"+type,function(){
9          listener.apply(el);
10      });
11   }
12 }
```

评价：

- 不应该在 if 和 else 语句中声明 addListener 函数，应该先声明；
- 不需要使用 window.addEventListener 或 document.all 来进行检测浏览器，应该使用能力检测；
- 由于 attachEvent 在 IE 中有 this 指向问题，所以调用它时需要处理一下

改进如下：

```
1 function addEvent(elem, type, handler){
2    if(elem.addEventListener){
3        elem.addEventListener(type, handler, false);
4    }else if(elem.attachEvent){
5        elem['temp' + type + handler] = handler;
6        elem[type + handler] = function(){
7        elem['temp' + type + handler].apply(elem);
8    };
9    elem.attachEvent('on' + type, elem[type + handler]);
10  }else{
11    elem['on' + type] = handler;
12    }
```

```
13 }
```

7.给 String 对象添加一个方法，传入一个 string 类型的参数，然后将 string 的每个字符间价格空格返回，例如：

addSpace（“hello world”）// -> ‘h e l l o w o r l d’

```
1  String.prototype.spacify = function(){

2      return this.split('').join(' ');

3    };
```

接着上述答题，那么问题来了

1）直接在对象的原型上添加方法是否安全？尤其是在 Object 对象上。(这个我没能答出？希望知道的说一下。)

2）函数声明与函数表达式的区别？

答案：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。（函数声明提升）

8.定义一个 log 方法，让它可以代理 console.log 的方法。

可行的方法一：

```
1 function log(msg)  {

2    console.log(msg);

3 }

4

5 log("hello world!") // hello world!

6
```

如果要传入多个参数呢？显然上面的方法不能满足要求，所以更好的方法是：

```
1 function log(){
```

```
2    console.log.apply(console, arguments);

3 };
```

那么问题来了，apply 和 call 方法的异同？

答案：

对于 apply 和 call 两者在作用上是相同的，即是调用一个对象的一个方法，以另一个对象替换当前对象。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。

但两者在参数上有区别的。对于第一个参数意义都一样，但对第二个参数： apply 传入的是一个参数数组，也就是将多个参数组合成为一个数组传入，而 call 则作为 call 的参数传入（从第二个参数开始）。 如 func.call(func1,var1,var2,var3)对应的 apply 写法为：func.apply(func1, [var1,var2,var3]) 。

9.在 Javascript 中什么是伪数组？如何将伪数组转化为标准数组？

答案：

伪数组（类数组）：无法直接调用数组方法或期望 length 属性有什么特殊的行为，但仍可以对真正数组遍历方法来遍历它们。典型的是函数的 argument 参数，还有像调用 getElementsByTagName,document.childNodes 之类的,它们都返回 NodeList 对象都属于伪数组。可以使用 Array.prototype.slice.call(fakeArray)将数组转化为真正的 Array 对象。

假设接第八题题干，我们要给每个 log 方法添加一个”(app)”前缀，比如’ hello world!’ ->’(app)hello world!’ 。方法如下：

```
1  function log(){

2      var args = Array.prototype.slice.call(arguments);  //为了使用 unshift 数组方法，将 argument 转化为真正的数组

3      args.unshift('(app)');
```

```
4
5      console.log.apply(console, args);
6    };
```

10.对作用域上下文和 this 的理解，看下列代码：

```
 1 var User = {
 2   count: 1,
 3
 4   getCount: function() {
 5     return this.count;
 6   }
 7 };
 8
 9 console.log(User.getCount());  // what?
10
11 var func = User.getCount;
12 console.log(func());  // what?
```

问两处 console 输出什么？为什么？

答案是 1 和 undefined。

func 是在 winodw 的上下文中被执行的，所以会访问不到 count 属性。

那么问题来了，如何确保 Uesr 总是能访问到 func 的上下文，即正确返回 1。

答案：正确的方法是使用 Function.prototype.bind。兼容各个浏览器完整代码如下：

```
 1 Function.prototype.bind = Function.prototype.bind || function(context){
```

```
 2    var self = this;
 3
 4    return function(){
 5       return self.apply(context, arguments);
 6    };
 7 }
 8
 9 var func = User.getCount.bind(User);
10 console.log(func());
```

11.原生 JS 的 window.onload 与 Jquery 的$(document).ready(function(){})有什么不同？如何用原生 JS 实现 Jq 的 ready 方法？

window.onload()方法是必须等到页面内包括图片的所有元素加载完毕后才能执行。

$(document).ready()是 DOM 结构绘制完毕后就执行，不必等到加载完毕。

```
 1 /*
 2  * 传递函数给 whenReady()
 3  * 当文档解析完毕且为操作准备就绪时，函数作为 document 的方法调用
 4  */
 5 var whenReady = (function() {              //这个函数返回 whenReady()函数
 6     var funcs = [];             //当获得事件时，要运行的函数
 7     var ready = false;          //当触发事件处理程序时,切换为 true
 8
 9     //当文档就绪时,调用事件处理程序
10     function handler(e) {
11         if(ready) return;       //确保事件处理程序只完整运行一次
12
```

```
13        //如果发生 onreadystatechange 事件，但其状态不是 complete 的话,那么文档尚未准备好
14        if(e.type === 'onreadystatechange' && document.readyState !== 'complete') {
15            return;
16        }
17
18        //运行所有注册函数
19        //注意每次都要计算 funcs.length
20        //以防这些函数的调用可能会导致注册更多的函数
21        for(var i=0; i<funcs.length; i++) {
22            funcs[i].call(document);
23        }
24        //事件处理函数完整执行,切换 ready 状态, 并移除所有函数
25        ready = true;
26        funcs = null;
27    }
28    //为接收到的任何事件注册处理程序
29    if(document.addEventListener) {
30        document.addEventListener('DOMContentLoaded', handler, false);
31        document.addEventListener('readystatechange', handler, false);          //IE9
+
32        window.addEventListener('load', handler, false);
33    }else if(document.attachEvent) {
34        document.attachEvent('onreadystatechange', handler);
35        window.attachEvent('onload', handler);
36    }
37    //返回 whenReady()函数
38    return function whenReady(fn) {
```

```
39        if(ready) { fn.call(document); }
40        else { funcs.push(fn); }
41    }
42 })();
```

如果上述代码十分难懂，下面这个简化版：

```
 1 function ready(fn){
 2    if(document.addEventListener) {       //标准浏览器
 3        document.addEventListener('DOMContentLoaded', function() {
 4            //注销事件, 避免反复触发
 5            document.removeEventListener('DOMContentLoaded',arguments.callee, false);
 6            fn();           //执行函数
 7        }, false);
 8    }else if(document.attachEvent) {       //IE
 9        document.attachEvent('onreadystatechange', function() {
10            if(document.readyState == 'complete') {
11                document.detachEvent('onreadystatechange', arguments.callee);
12                fn();       //函数执行
13            }
14        });
15    }
16 };
```

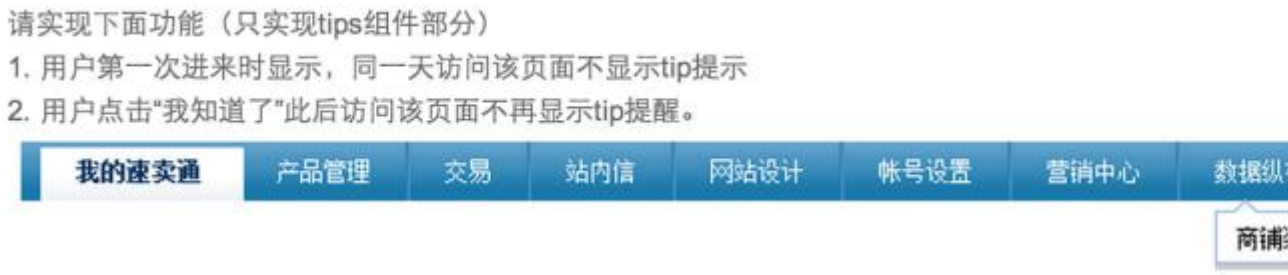
12.（设计题）想实现一个对页面某个节点的拖曳？如何做？（使用原生 JS）

回答出概念即可，下面是几个要点

1. 给需要拖拽的节点绑定 mousedown, mousemove, mouseup 事件
2. mousedown 事件触发后，开始拖拽

3. mousemove 时，需要通过 event.clientX 和 clientY 获取拖拽位置，并实时更新位置
4. mouseup 时，拖拽结束
5. 需要注意浏览器边界的情况

13.




```
function setcookie(name,value,days){  //给 cookie 增加一个时间变量
    var exp = new Date();
    exp.setTime(exp.getTime() + days*24*60*60*1000); //设置过期时间为 days 天
    document.cookie = name + "="+ escape (value) + ";expires=" + exp.toGMTString();
}
function getCookie(name){
    var result = "";
    var myCookie = ""+document.cookie+";";
    var searchName = "+name+"=";
    var startOfCookie = myCookie.indexOf(searchName);
    var endOfCookie;
    if(satrtOfCookie != -1){
        startOfcookie += searchName.length;
        endOfCookie = myCookie.indexOf(";",startOfCookie);
        result = (myCookie.substring(startOfCookie,endOfCookie));
    }
    return result;
```

```
}
(function(){
    var oTips = document.getElementById('tips');//假设 tips 的 id 为 tips
    var page = {
    check: function(){//检查 tips 的 cookie 是否存在并且允许显示
        var tips = getCookie('tips');
        if(!tips || tips == 'show') return true;//tips 的 cookie 不存在
        if(tips == "never_show_again") return false;
    },
    hideTip: function(bNever){
        if(bNever) setcookie('tips', 'never_show_again', 365);
        oTips.style.display = "none";//隐藏
    },
    showTip: function(){
    oTips.style.display = "inline";//显示，假设 tips 为行级元素
    },
    init: function(){
        var _this = this;
        if(this.check()){
        _this.showTip();
        setcookie('tips', 'show', 1);
    }
    oTips.onclick = function(){
        _this.hideTip(true);
    };
    }
    };
```

```
45   page.init();
46 })();
```

14.说出以下函数的作用是？空白区域应该填写什么？

```
 1 //define
 2 (function(window){
 3     function fn(str){
 4         this.str=str;
 5     }
 6
 7     fn.prototype.format = function(){
 8         var arg = ______;
 9         return this.str.replace(_____,function(a,b){
10             return arg[b]||"";
11       });
12     }
13     window.fn = fn;
14 })(window);
15
16 //use
17 (function(){
18     var t = new fn('<p><a href="{0}">{1}</a><span>{2}</span></p>');
19     console.log(t.format('http://www.alibaba.com','Alibaba','Welcome'));
20 })();
```

答案：访函数的作用是使用 format 函数将函数的参数替换掉{0}这样的内容，返回一个格式化后的结果：

第一个空是：arguments

第二个空是：/\{(\d+)\}/ig

15.用面向对象的 Javascript 来介绍一下自己。（没答案哦亲，自己试试吧）

答案： 对象或者 Json 都是不错的选择哦

欢迎加群领取资料！！！！