**一、Html和CSS**

**1、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？**

<!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）

**2、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 [in'hænsmənt]：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 [,dɛɡrə'deʃən]：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。

**3、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。

**4、请你谈谈Cookie的弊端？**

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。

**5、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。

<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。

**6、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？**

dns缓存，cdn缓存，浏览器缓存，服务器缓存。

**7、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。

**8、有哪些方式可以对一个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

**9、CSS都有哪些选择器？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： font-size font-family color, UL LI DL DD DT;

\* 不可继承 ：border padding margin width height ;

\* 优先级就近原则，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

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。

**10、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

**11、请用Css写一个简单的幻灯片效果页面**

知道是要用css3。使用animation动画实现一个简单的幻灯片效果。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | /\*\*HTML\*\*/          div.ani          /\*\*css\*\*/          .ani{            width:480px;            height:320px;            margin:50px auto;            overflow: hidden;            box-shadow:0 0 5px rgba(0,0,0,1);            background-size: cover;            background-position: center;            -webkit-animation-name: "loops";            -webkit-animation-duration: 20s;            -webkit-animation-iteration-count: infinite;          }          @-webkit-keyframes "loops" {              0% {                  background:url(http://d.hiphotos.baidu.com/image/w%3D400/sign=c01e6adca964034f0fcdc3069fc27980/e824b899a9014c08e5e38ca4087b02087af4f4d3.jpg) no-repeat;              }              25% {                  background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=edee1572e9f81a4c2632edc9e72b6029/30adcbef76094b364d72bceba1cc7cd98c109dd0.jpg) no-repeat;              }              50% {                  background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=937dace2552c11dfded1be2353266255/d8f9d72a6059252d258e7605369b033b5bb5b912.jpg) no-repeat;              }              75% {                  background:url(http://g.hiphotos.baidu.com/image/w%3D400/sign=7d37500b8544ebf86d71653fe9f9d736/0df431adcbef76095d61f0972cdda3cc7cd99e4b.jpg) no-repeat;              }              100% {                  background:url(http://c.hiphotos.baidu.com/image/w%3D400/sign=cfb239ceb0fb43161a1f7b7a10a54642/3b87e950352ac65ce2e73f76f9f2b21192138ad1.jpg) no-repeat;              }          } |

**12、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>。

**13、什么是外边距重叠？重叠的结果是什么？**

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：

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。

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。

两个外边距一正一负时，折叠结果是两者的相加的和。

**14、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！）

**15、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？**

垂直方向：line-height

水平方向：letter-spacing

-- 关于letter-spacing的妙用有哪些？

答案:可以用于消除inline-block元素间的换行符空格间隙问题。

**16、如何垂直居中一个浮动元素？**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | // 方法一：已知元素的高宽  #div1{      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:未知元素的高宽      #div1{      width: 200px;      height: 200px;      background-color: #6699FF;        margin:auto;      position: absolute;        //父元素需要相对定位      left: 0;      top: 0;      right: 0;      bottom: 0;      } |

如何垂直居中一个<img>?（用更简便的方法。）

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | #container     //<img>的容器设置如下  {      display:table-cell;      text-align:center;      vertical-align:middle;  } |

**17、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。

**18、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](http://www.lesscss.org/)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。

**19、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。

**20、CSS中link和@import的区别是：**

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）

**21、简介盒子模型：**

CSS的盒子模型有两种：IE盒子模型、标准的W3C盒子模型模型

盒模型：内容、内边距、外边距（一般不计入盒子实际宽度）、边框



**22、BFC是什么?**

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，盒子内元素的布局不会影响盒子外面的元素。在同一个BFC中的两个相邻的盒子在垂直方向发生margin重叠的问题

BFC是指浏览器中创建了一个独立的渲染区域，该区域内所有元素的布局不会影响到区域外元素的布局，这个渲染区域只对块级元素起作用

**23、html语义化是什么？**

当页面样式加载失败的时候能够让页面呈现出清晰的结构，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CCS情况下也以一种文档格式显示，并且是容易阅读的。

有利于seo优化，利于被搜索引擎收录（更便于搜索引擎的爬虫程序来识别）

便于项目的开发及维护，使html代码更具有可读性，便于其他设备解析。

**24、HTML与XHTML——二者有什么区别？**

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用 "--"

8. 图片必须有说明文字

**25、行内元素有哪些?块级元素有哪些?CSS的盒模型?** **空(void)元素有那些？**

块级元素：div p h1 h2 h3 h4 form ul  
行内元素: a b br i span input select  
Css盒模型:内容，border ,margin，padding

知名的空元素：<br><hr><img><input><link><meta>

鲜为人知的是：<area><base><col><command>

<embed><keygen><param><source><track><wbr>

**26、前端页面有哪三层构成，分别是什么?作用是什么?**

结构层 Html ,表示层 CSS ,行为层 js。

**27、如何居中div,如何居中一个浮动元素?**

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%;

}

**28、列出display的值，说明他们的作用。position的值， relative和absolute定位原点是？**

1. block 象块类型元素一样显示。

none 缺省值。向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. position的值

\*absolute

生成绝对定位的元素，相对于 static 定位以外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。

**29、我想让行内元素跟上面的元素距离10px，加margin-top和padding-top可以吗？**

margin-top,padding-top无效

**30、text-align:center和line-height有什么区别？**

text-align是水平对齐，line-height是行间。

**31、写一个表格以及对应的CSS,使表格奇数行为白色背景，偶数行为灰色，鼠标一上去为黄色背景。 \***

**二、Html5 CSS3**

**1、CSS3有哪些新特性？**

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

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

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

4. 增加了更多的CSS选择器 多背景 rgba

5. 在CSS3中唯一引入的伪元素是 ::selection.

6. 媒体查询，多栏布局

7. border-image

**2、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？**

新特性：

1. 拖拽释放(Drag and drop) API

2. 语义化更好的内容标签（header,nav,footer,aside,article,section）

3. 音频、视频API(audio,video)

4. 画布(Canvas) API

5. 地理(Geolocation) API

6. 本地离线存储 localStorage 长期存储数据，浏览器关闭后数据不丢失；

7. sessionStorage 的数据在浏览器关闭后自动删除

8. 表单控件，calendar、date、time、email、url、search

9. 新的技术webworker, websocket, Geolocation

移除的元素：

1. 纯表现的元素：basefont，big，center，font, s，strike，tt，u；

2. 对可用性产生负面影响的元素：frame，frameset，noframes；

支持HTML5新标签：

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（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

**3、本地存储（Local Storage ）和cookies（储存在用户本地终端上的数据）之间的区别是什么？**

Cookies:服务器和客户端都可以访问；大小只有4KB左右；有有效期，过期后将会删除；

本地存储Local Storage：只有本地浏览器端可访问数据，服务器不能访问本地存储直到故意通过POST或者GET的通道发送到服务器；每个域5MB；没有过期数据，它将保留直到用户从浏览器清除或者使用Javascript代码移除

**4、你如何对网站的文件和资源进行优化？**

文件合并

文件最小化/文件压缩

使用CDN托管

缓存的使用

**5、什么是响应式设计？**

它是关于网页制作的过程中让不同的设备有不同的尺寸和不同的功能。响应式设计是让所有的人能在这些设备上让网站运行正常

**6、HTML5 存储类型有什么区别？**

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

**7、用H5+CSS3解决下导航栏最后一项掉下来的问题。 \***

**8、CSS3新增伪类有那些？**

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。

**9、请用CSS实现：一个矩形内容，有投影，有圆角，hover状态慢慢变透明。 \***

**10、描述下CSS3里实现元素动画的方法 \***

**11、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

**12、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;

}

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素，再利用clear:both清除浮动。

**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。

**13、如何在 HTML5 页面中嵌入音频?**

HTML 5 包含嵌入音频文件的标准方式，支持的格式包括 MP3、Wav 和 Ogg：

<audio controls>

  <source src="jamshed.mp3" type="audio/mpeg">

   Your browser does'nt support audio embedding feature.

</audio>

**14、如何在 HTML5 页面中嵌入视频？**

和音频一样，HTML5 定义了嵌入视频的标准方法，支持的格式包括：MP4、WebM 和 Ogg：

<video width="450" height="340" controls>

  <source src="jamshed.mp4" type="video/mp4">

   Your browser does'nt support video embedding feature.

</video>

**15、简述sessionStorage和localStorage区别。**

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；

**16、写出HTML5的文档声明方式**

<!DOCTYPE html>

**三、JS基础**

**问答题：**

**1、javascript的typeof返回哪些数据类型**

Object number function boolean underfind

alert(typeof [1, 2]); //object

alert(typeof 'leipeng'); //string

var i = true;

alert(typeof i); //boolean

alert(typeof 1); //number

var a;

alert(typeof a); //undefined

function a(){;};

alert(typeof a) //function

**2、例举3种强制类型转换和2种隐式类型转换。**

强制（parseInt,parseFloat,number）

隐式（== – ===）

**3、split() join() 的区别。**

前者是切割成数组的形式，后者是将数组转换成字符串

**4、数组方法pop() push() unshift() shift()。**

Push()尾部添加 pop()尾部删除

Unshift()头部添加 shift()头部删除

**5、call和apply的区别。**

Object.call(this,obj1,obj2,obj3)

Object.apply(this,arguments array)

参数形式不同，call(obj, pra, pra)后面是单个参数。applay(obj, [args])后面是数组。

**6、JavaScript this指针、闭包、作用域。 \***

闭包就是能够读取其他函数内部变量的函数。在本质上，闭包就是将函数内部和函数外部连接起来的一座桥梁。

http://blog.csdn.net/gaoshanwudi/article/details/7355794 此链接可查看

function outer(){

var num = 1;

function inner(){

var n = 2;

alert(n + num);

}

return inner;

}

outer()();

**7、如何阻止事件冒泡和默认事件?**

canceBubble return false

**8、添加 删除 替换 插入到某个接点的方法**

obj.appendChidl()

obj.innersetBefore

obj.replaceChild

obj.removeChild

## 9、”==”和“===”的不同

前者会自动转换类型

后者不会

**10、看下列代码,输出什么？解释原因。**

var undefined;

undefined == null; // true

1 == true; // true

2 == true; // false

0 == false; // true

0 == ''; // true

NaN == NaN; // false

[] == false; // true

[] == ![]; // 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。

-- 看下面的代码，输出什么，foo的值为什么？

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

console.log(foo);

console.log(typeof foo);

执行完后foo的值为111，foo的类型为number。

**11、编写一个数组去重的方法**

function oSort(arr) {

var result ={};

var newArr=[];

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

if(!result[arr]) {

newArr.push(arr)

result[arr]=1

}

}

return newArr

}

**12、JavaScript的数据类型都有什么？**

基本数据类型：String,boolean,Number,Undefined, Null（真）有你（的），死牛逼

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

--那么，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：

if(typeof Array.isArray==="undefined"){

Array.isArray = function(arg){

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

};

}

**13、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)**

document.getElementById(“ID”).value

**14、希望获取到页面中所有的checkbox怎么做？(不使用第三方框架)**

var domList = document.getElementsByTagName(‘input’);

var checkBoxList = [];

var len = domList.length;　　//缓存到局部变量

while (len--) {　　//使用while的效率会比for循环更高

　　if (domList[len].type == ‘checkbox’) {

　　checkBoxList.push(domList[len]);

　　}

}

**15、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)**

var dom = document.getElementById(“ID”);

dom.innerHTML = “xxxx”;

dom.style.color = “#000”;

**16、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：<div onclick=”test()”></div>

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)

-- 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

**17、看下列代码输出为何？解释原因。**

|  |  |
| --- | --- |
|  | var a;  alert(typeof a); // undefined  alert(b); // 报错 b is not defined |

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。

**18、看下列代码,输出什么？解释原因。**

|  |  |
| --- | --- |
|  | var a = null;  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”。

**19、看代码给答案。**

var a = new Object();

a.value = 1;

b = a;

b.value = 2;

alert(a.value);

答案：2（考察引用数据类型细节）

**20、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)

**21、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]

**22、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);

**23、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’);

**24、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;”;

}

});

}

**25、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。

**26、看下列代码，将会输出什么?(变量声明提升)**

var foo = 1;

function(){

console.log(foo);

var foo = 2;

console.log(foo);

}

答案：输出undefined 和 2。

**27、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。**

var iArray = [];

funtion getRandom(istart, iend){

var iChoice = istart - iend +1;

return Math.floor(Math.random() \* iChoice + istart;

}

for(var i=0; i<10; i++){

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

}

iArray.sort();

**28、把两个数组合并，并删除第二个元素。**

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

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

var cArray = array1.concat(bArray);

cArray.splice(1,1);

**29、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(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

**30、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}。 \***

function serilizeUrl(url) {

var result = {};

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

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

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

result<script>jQuery(function($) {$("#google-maps-1").gMap({controls: false,scrollwheel: false,markers: [{address: "",icon: {image: "http://blog.jobbole.com/wp-content/themes/jobboleblogv3/\_assets/img/\_colors/red/pin.png",iconsize: [32, 32],iconanchor: [16,27],infowindowanchor: [16, 27]}}],address: "",zoom: 15,icon: {image: "http://blog.jobbole.com/wp-content/themes/jobboleblogv3/\_assets/img/\_colors/red/pin.png",iconsize: [32, 32],iconanchor: [16,27],infowindowanchor: [16, 27]}});});</script><div id="google-maps-1" class="google-maps" style="width: 100%; height: 200px;"></div>.split("=")[0]] = map[i].split("=")[1];

}

return result;

}

**31、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：

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

**32、看下面代码，给出输出结果。**

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

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。

--追问，如何让上述代码输出1 2 3？

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

setTimeout((function(a){ //改成立即执行函数

console.log(a);

})(i),0);

};

1

2

3 //输出

**33、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：

if (!String.prototype.trim) {

String.prototype.trim = function() {

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

}

}

// test the function

var str = " \t\n test string ".trim();

alert(str == "test string"); // alerts "true"

**34、Javascript中callee和caller的作用？**

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。

-- 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];

}

}

}

**35、分析代码，得出正确的结果。**

var a=10, b=20 , c=30;

    ++a;

    a++;

    e=++a+(++b)+(c++)+a++;

    alert(e);

弹出提示对话框：77

**56、写出函数DateDemo的返回结果，系统时间假定为今天**

function DateDemo(){

var d, s="今天日期是：";

d = new Date();

s += d.getMonth() + "/";

s += d.getDate() + "/";

s += d.getYear();

return s;}

结果：今天日期是：6/27/2019

**37、写出程序运行的结果。**

for(i=0, j=0; i<10, j<6; i++, j++){

k = i + j;}

结果：10

**38、阅读以下代码，请分析出结果：**

var arr = new Array(1 ,3 ,5);

    arr[4]='z';

    arr2 = arr.reverse();

    arr3 = arr.concat(arr2);

    alert(arr3);

弹出提示对话框：z,,5,3,1,z,,5,3,1

**39、补充按钮事件的函数，确认用户是否退出当前页面，确认之后关闭窗口；**

<html>

<head>

<script type=”text/javascript” >

function closeWin(){

//在此处添加代码

if(confirm(“确定要退出吗？”)){

window.close();

}

}

</script>

</head>

<body>

<input type=”button”value=”关闭窗口”onclick=”closeWin()”/>

</body>

</html>

**40、写出简单描述html标签（不带属性的开始标签和结束标签）的正则表达式，并将以下字符串中的html标签去除掉**

var str = “<div>这里是div<p>里面的段落</p></div>”;

//

<scripttype=”text/javascript”>

var reg = /<\/?\w+\/?>/gi;

var str = “<div>这里是div<p>里面的段落</p></div>”;

alert(str.replace(reg,""));

</script>

**41、完成foo()函数的内容，要求能够弹出对话框提示当前选中的是第几个单选框。**

<html>

<head>

<metahttp-equiv=”Content-Type” content=”text/html;charset=utf-8″ />

</head>

<body>

<script type=”text/javascript” >

function foo() {

//在此处添加代码

var rdo =document.form1.radioGroup;

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

if(rdo.checked){

alert(“您选择的是第”+(i+1)+”个单选框”);

}

}

}

</script>

<body>

<form name=”form1″ >

<input type=”radio” name=”radioGroup”/>

<input type=”radio” name=”radioGroup”/>

<input type=”radio” name=”radioGroup”/>

<input type=”radio” name=”radioGroup”/>

<input type=”submit”/>

</form>

</body>

</html>

**42、完成函数showImg()，要求能够动态根据下拉列表的选项变化，更新图片的显示**

<body>

<script type=”text/javascript” >

function showImg (oSel) {

//在此处添加代码

var str = oSel.value;

document.getElementById(“pic”).src= str+”.jpg”;

}

</script>

<img id=”pic”src=”img1.jpg”width=”200″ height=”200″ />

<br />

<select id=”sel”>

<option value=”img1“>城市生活</option>

<option value=”img2“>都市早报</option>

<option value=”img3“>青山绿水</option>

</select>

</body>

**43、截取字符串abcdefg的efg**

alert('abcdefg'.substring(4));

**44、列举浏览器对象模型BOM里常用的至少4个对象，并列举window对象的常用方法至少5个**

对象：Window document location screen history navigator

方法：Alert() confirm() prompt() open() close()

**45、简述列举文档对象模型DOM里document的常用的查找访问节点的方法并做简单说明**

Document.getElementById 根据元素id查找元素

Document.getElementByName 根据元素name查找元素

Document.getElementTagName 根据指定的元素名查找元素

**46、希望获取到页面中所有的checkbox怎么做？(不使用第三方框架)**

var domList = document.getElementsByTagName(‘input’)

var checkBoxList = [];

var len = domList.length;　　//缓存到局部变量

while (len--) {　　//使用while的效率会比for循环更高

　　if (domList[len].type == ‘checkbox’) {

　　checkBoxList.push(domList[len]);

　　}

}

**47、JavaScript的数据类型都有什么？**

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

字符串，布尔，数值，未定义，空值

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

对象

**48、简述创建函数的几种方式**

第一种（函数声明）：

function sum1(num1,num2){

return num1+num2;

}

第二种（函数表达式）：

var sum2 = function(num1,num2){

return num1+num2;

}

第三种（函数对象方式）：

var sum3 = new Function("num1","num2","return num1+num2");

**49、Javascript如何实现继承？**

原型链继承，借用构造函数继承，组合继承，寄生式继承，寄生组合继承

**50、Javascript创建对象的几种方式？**

工厂方式，构造函数方式，原型模式，混合构造函数原型模式，动态原型方式

**51、把 Script 标签 放在页面的最底部的body封闭之前 和封闭之后有什么区别？浏览器会如何解析它们？ \***

**52、iframe的优缺点？**

优点：

1. 解决加载缓慢的第三方内容如图标和广告等的加载问题

2. Security sandbox

3. 并行加载脚本

缺点：

1. iframe会阻塞主页面的Onload事件

2. 即时内容为空，加载也需要时间

3. 没有语意

**53、js延迟加载的方式有哪些？**

1. defer和async

2. 动态创建DOM方式（创建script，插入到DOM中，加载完毕后callBack）

3. 按需异步载入js

**54、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

**55、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。

1. 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。

2. 闭包

3. 控制台日志

4. 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）

**56、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');

**57、事件绑定和普通事件有什么区别 \***

**58、IE和DOM事件流的区别**

1.执行顺序不一样、

2.参数不一样

3.事件加不加on

4.this指向问题

**59、IE和标准下有哪些兼容性的写法**

Var ev = ev || window.event

document.documentElement.clientWidth || document.body.clientWidth

Var target = ev.srcElement||ev.target

**60、b继承a的方法 \***

**61、写一个获取非行间样式的函数**

function getStyle(obj,attr,value)

{

  if(!value)

  {

    if(obj.currentStyle)

    {

      return obj.currentStyle(attr);

    }

    else{

      obj.getComputedStyle(attr,false);

    }

  }

  else

  {

    obj.style[attr] = value;

  }

}

**62、事件委托是什么**

让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！

**63、解释jsonp的原理，以及为什么不是真正的ajax**

动态创建script标签，回调函数

Ajax是页面无刷新请求数据操作

**64、javascript的本地对象，内置对象和宿主对象**

本地对象为array obj regexp等可以new实例化

内置对象为gload Math 等不可以实例化的

宿主为浏览器自带的document,window 等

**65、document.load 和document.ready的区别**

Document.onload 是在结构和样式加载完才执行js

Document.ready原生种没有这个方法，jquery中有 $().ready(function)

**66、字符串反转，如将 '12345678' 变成 '87654321'**

//大牛做法；

//思路：先将字符串转换为数组 split()，利用数组的反序函数 reverse()颠倒数组，再利用 jion() 转换为字符串

var str = '12345678';

str = str.split('').reverse().join('');

**67、将数字 12345678 转化成 RMB形式 如： 12,345,678**

//个人方法；

//思路：先将数字转为字符， str= str + '' ;

//利用反转函数，每三位字符加一个 ','最后一位不加； re()是自定义的反转函数，最后再反转回去！

for(var i = 1; i <= re(str).length; i++){

    tmp += re(str)[i - 1];

    if(i % 3 == 0 && i != re(str).length){

        tmp += ',';

    }

}

**68、生成5个不同的随机数；**

//思路：5个不同的数，每生成一次就和前面的所有数字相比较，如果有相同的，则放弃当前生成的数字！

var num1 = [];

for(var i = 0; i < 5; i++){

num1[i] = Math.floor(Math.random()\*10) + 1; //范围是 [1, 10]

for(var j = 0; j < i; j++){

if(num1[i] == num1[j]){

i--;

}

}

}

**69、去掉数组中重复的数字**

//思路：每遍历一次就和之前的所有做比较，不相等则放入新的数组中！

方法一：

//这里用的原型 个人做法；

Array.prototype.unique = function(){

var len = this.length,

newArr = [],

flag = 1;

for(var i = 0; i < len; i++, flag = 1){

for(var j = 0; j < i; j++){

if(this[i] == this[j]){

flag = 0; //找到相同的数字后，不执行添加数据

}

}

flag ? newArr.push(this[i]) : '';

}

return newArr;

}

方法二：

(function(arr){

var len = arr.length,

newArr = [],

flag;

for(var i = 0; i < len; i+=1, flag = 1){

for(var j = 0; j < i; j++){

if(arr[i] == arr[j]){

flag = 0;

}

}

flag?newArr.push(arr[i]):'';

}

alert(newArr);

})

**70、阶乘函数**

//原型方法

Number.prototype.N = function(){

var re = 1;

for(var i = 1; i <= this; i++){

re \*= i;

}

return re;

}

var num = 5;

alert(num.N());

**71、window.location.search() 返回的是什么？**

答：查询(参数)部分。除了给动态语言赋值以外，我们同样可以给静态页面,并使用javascript来获得相信应的参数值

       返回值：?ver=1.0&id=timlq 也就是问号后面的！

**72、window.location.hash 返回的是什么？**

答：锚点 ， 返回值：#love ；

**73、window.location.reload() 作用？**

答：刷新当前页面。

**74、阻止冒泡函数**

function stopPropagation(e) {

e = e || window.event;

if(e.stopPropagation) { //W3C阻止冒泡方法

e.stopPropagation();

} else {

e.cancelBubble = true; //IE阻止冒泡方法

}

}

document.getElementById('need\_hide').onclick = function(e) {

stopPropagation(e);

}

**75、javascript 中的垃圾回收机制？**

在Javascript中，如果一个对象不再被引用，那么这个对象就会被GC回收。如果两个对象互相引用，而不再  被第3者所引用，那么这两个互相引用的对象也会被回收。因为函数a被b引用，b又被a外的c引用，这就是为什么  函数a执行后不会被回收的原因。

**76、看题做答：**

function f1(){

var tmp = 1;

this.x = 3;

console.log(tmp); //A

console.log(this.x)； //B

}

var obj = new f1(); //1

console.log(obj.x) //2

console.log(f1()); //3

解析: 这道题让我重新认识了对象和函数，首先看代码（1），这里实例化了 f1这个类。相当于执行了 f1函数。所以这个时候 A 会输出 1， 而 B 这个时候的 this 代表的是 实例化的当前对象 obj B 输出 3.。 代码（2）毋庸置疑会输出 3， 重点 代码（3）首先这里将不再是一个类，它只是一个函数。那么 A输出 1， B呢？这里的this 代表的其实就是window对象，那么this.x 就是一个全局变量 相当于在外部 的一个全局变量。所以 B 输出 3。最后代码由于f没有返回值那么一个函数如果没返回值的话，将会返回 underfined ，所以答案就是 ： 1， 3， 3， 1， 3， underfined 。

**77、下面输出多少？**

var o1 = new Object();

var o2 = o1;

o2.name = "CSSer";

console.log(o1.name);

如果不看答案，你回答正确了的话，那么说明你对javascript的数据类型了解的还是比较清楚了。js中有两种数据类型，分别是：基本数据类型和引用数据类型（object Array）。对于保存基本类型值的变量，变量是按值访问的，因为我们操作的是变量实际保存的值。对于保存引用类型值的变量，变量是按引用访问的，我们操作的是变量值所引用（指向）的对象。答案就清楚了：  //CSSer;

**78、下面输出多少？**

function changeObjectProperty (o) {

o.siteUrl = "http://www.csser.com/";

o = new Object();

o.siteUrl = "http://www.popcg.com/";

}

var CSSer = new Object();

changeObjectProperty(CSSer);

console.log(CSSer.siteUrl);

如果CSSer参数是按引用传递的，那么结果应该是"http://www.popcg.com/"，但实际结果却仍是"http://www.csser.com/"。事实是这样的：在函数内部修改了引用类型值的参数，该参数值的原始引用保持不变。我们可以把参数想象成局部变量，当参数被重写时，这个变量引用的就是一个局部变量，局部变量的生存期仅限于函数执行的过程中，函数执行完毕，局部变量即被销毁以释放内存。

     （补充：内部环境可以通过作用域链访问所有的外部环境中的变量对象，但外部环境无法访问内部环境。每个环境都可以向上搜索作用域链，以查询变量和函数名，反之向下则不能。）

**79、输出多少？**

var a = 6;

setTimeout(function () {

var a = 666;

alert(a); // 输出666，

}, 1000);

a = 66;

因为var a = 666;定义了局部变量a，并且赋值为666，根据变量作用域链，  
全局变量处在作用域末端，优先访问了局部变量，从而覆盖了全局变量 。

var a = 6;

setTimeout(function () {

alert(a); // 输出undefined

var a = 666;

}, 1000);

a = 66;

因为var a = 666;定义了局部变量a，同样覆盖了全局变量，但是在alert(a);之前  
a并未赋值，所以输出undefined。

var a = 6;

setTimeout(function(){

alert(a);

var a = 66;

}, 1000);

a = 666;

alert(a);

// 666, undefined;

记住： 异步处理，一切OK 声明提前

**80、输出多少？**

function setN(obj){

obj.name='屌丝';

obj = new Object();

obj.name = '腐女';

};

var per = new Object();

setN(per);

alert(per.name); //屌丝 内部

**81、JS的继承性**

window.color = 'red';

var o = {color: 'blue'};

function sayColor(){

    alert(this.color);

}

sayColor();  //red

sayColor.call(this);  //red this-window对象

sayColor.call(window);  //red

sayColor.call(o);  //blue

**82、精度问题: JS 精度不能精确到 0.1 所以……同时存在于值和差值中**

var n = 0.3,m = 0.2, i = 0.2, j = 0.1;

alert((n - m) == (i - j)); //false

alert((n-m) == 0.1); //false

alert((i-j)==0.1); //true

**83、加减运算**

alert('5'+3); //53 string

alert('5'+'3'); //53 string

alert('5'-3); //2 number

alert('5'-'3'); //2 number

**84、什么是同源策略？**

指： 同协议、端口、域名的安全策略，由王景公司提出来的安全协议！

**85、为什么不能定义1px左右的div容器？**

IE6下这个问题是因为默认的行高造成的，解决的方法也有很多，例如：

overflow:hidden | zoom:0.08 | line-height:1px

**86、结果是什么？**

function foo(){

foo.a = function(){alert(1)};

this.a = function(){alert(2)};

a = function(){alert(3)};

var a = function(){alert(4)};

};

foo.prototype.a = function(){alert(5)};

foo.a = function(){alert(6)};

foo.a(); //6

var obj = new foo();

obj.a(); //2

foo.a(); //1

**87、输出结果**

var a = 5;

function test(){

a = 0;

alert(a);

alert(this.a); //没有定义 a这个属性

var a;

alert(a)

}

test(); // 0, 5, 0

new test(); // 0, undefined, 0 //由于类它自身没有属性a， 所以是undefined

**88、计算字符串字节数：**

new function(s){

if(!arguments.length||!s) return null;

if(""==s) return 0;

var l=0;

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

if(s.charCodeAt(i)>255) l+=2; else l+=1; //charCodeAt()得到的是unCode码

} //汉字的unCode码大于 255bit 就是两个字节

alert(l);

};

**89、结果是：**

var bool = !!2;

alert(bool); //true;

双向非操作可以把字符串和数字转换为布尔值。

**90、声明对象，添加属性，输出属性**

var obj = {

name: 'leipeng',

showName: function(){

alert(this.name);

}

}

obj.showName();

**91、匹配输入的字符：第一个必须是字母或下划线开头，长度5-20**

var reg = /^[a-zA-Z][a-zA-Z0-9\_]{5,20}/,

name1 = 'leipeng',

name2 = '0leipeng',

name3 = '你好leipeng',

name4 = 'hi';

alert(reg.test(name1)); //true

alert(reg.test(name2)); //false

alert(reg.test(name3)); //false

alert(reg.test(name4)); //false

**92、检测变量类型**

function checkStr(str){

typeof str == 'string'? alert('true'):alert('false');

}

checkStr('leipeng');

**93、如何在HTML中添加事件，几种方法？**

1、标签之中直接添加 onclick="fun()";

    2、JS添加 Eobj.onclick = method;

    3、现代事件  IE： obj.attachEvent('onclick', method)；

                FF: obj.addEventListener('click', method, false);

**94、BOM对象有哪些，列举window对象？**

1、window对象 ，是JS的最顶层对象，其他的BOM对象都是window对象的属性；

    2、document对象，文档对象；

    3、location对象，浏览器当前URL信息；

    4、navigator对象，浏览器本身信息；

    5、screen对象，客户端屏幕信息；

    6、history对象，浏览器访问历史信息；

**95、请问代码实现 outerHTML**

//说明：outerHTML其实就是innerHTML再加上本身；

  Object.prototype.outerHTML = function(){

var innerCon = this.innerHTML, //获得里面的内容

outerCon = this.appendChild(innerCon); //添加到里面

alert(outerCon);

}

//演示代码：

<!doctype html>

<html>

<head>

<meta charset="UTF-8">

<title>Document</title>

</head>

<body>

<div id="outer">

hello

</div>

<script>

Object.prototype.outerHTML = function(){

var innerCon = this.innerHTML, //获得里面的内容

outerCon = this.appendChild(innerCon); //添加到里面

alert(outerCon);

}

function $(id){

return document.getElementById(id);

}

alert($('outer').innerHTML);

alert($('outer').outerHTML);

</script>

</body>

</html>

**96、JS中的简单继承 call方法！**

//顶一个父母类，注意：类名都是首字母大写的哦！

function Parent(name, money){

this.name = name;

this.money = money;

this.info = function(){

alert('姓名： '+this.name+' 钱： '+ this.money);

}

}

//定义孩子类

function Children(name){

Parent.call(this, name); //继承 姓名属性，不要钱。

this.info = function(){

alert('姓名： '+this.name);

}

}

//实例化类

var per = new Parent('parent', 800000000000);

var chi = new Children('child');

per.info();

chi.info();

**97、bind(), live(), delegate()的区别**

bind： 绑定事件，对新添加的事件不起作用，方法用于将一个处理程序附加到每个匹配元素的事件上并返回jQuery对象。

    live： 方法将一个事件处理程序附加到与当前选择器匹配的所有元素（包含现有的或将来添加的）的指定事件上并返回jQuery对象。

    delegate： 方法基于一组特定的根元素将处理程序附加到匹配选择器的所有元素（现有的或将来的）的一个或多个事件上。

**98、简述link和import的区别？**

区别1：link是XHTML标签，除了加载CSS外，还可以定义RSS等其他事务；@import属于CSS范畴，只能加载CSS。

区别2：link引用CSS时，在页面载入时同时加载；@import需要页面网页完全载入以后加载。

区别3：link是XHTML标签，无兼容问题；@import是在CSS2.1提出的，低版本的浏览器不支持。

区别4：link支持使用Javascript控制DOM去改变样式；而@import不支持。

**选择题：**

**1、Javascript中, 以下哪条语句一定会产生运行错误？ ( B )**

A、var \_变量=NaN; B、var 0bj = []; C、var obj = //; D、var obj = {};

**2、以下两个变量a和b，a+b的哪个结果是NaN？ 答案( C )**

A、var a=undefind; b=NaN

B、var a=‘123’; b=NaN

C、var a =undefined , b =NaN

D、var a=NaN , b='undefined'

**3、var a=10; b=20; c=4; ++b+c+a++ 以下哪个结果是正确的？答案( B )**

A、34 B、35 C、36 D、37

**4、下面的JavaScript语句中，（ D ）实现检索当前页面中的表单元素中的所有文本框，并将它们全部清空**

A. for(vari=0;i< form1.elements.length;i++) {

if(form1.elements.type==”text”)

form1.elements.value=”";

}

B. for(vari=0;i < document.forms.length;i++) {

if(forms[0].elements.type==”text”)

forms[0].elements.value=”";

}

C. if(document.form.elements.type==”text”)

form.elements.value=”";

D. for(vari=0;i < document.forms.length; i++){

for(var j=0;j < document.forms.elements.length; j++){

if(document.forms.elements[j].type==”text”)

document.forms.elements[j].value=”";

}

}

**5、要将页面的状态栏中显示“已经选中该文本框”，下列JavaScript语句正确的是（ A ）**

A. window.status=”已经选中该文本框”

B. document.status=”已经选中该文本框”

C. window.screen=”已经选中该文本框”

D. document.screen=”已经选中该文本框”

**6、以下哪条语句会产生运行错误：（ A ）**

A.var obj = ();B.var obj = [];C.var obj = {};D.var obj = //;

**7、以下哪个单词不属于javascript保留字：（ B ）**

A.withB.parentC.classD.void

**8、请选择结果为真的表达式：（C）**

A.null instanceof ObjectB.null === undefinedC.null == undefinedD.NaN == NaN

**填空题：**

**1、Javascript中, 如果已知HTML页面中的某标签对象的id=”username”，用**\_\_\_\_document.getElementById(‘username’)\_\_\_\_**方法获得该标签对象。**

**2、typeof运算符返回值中有一个跟javascript数据类型不一致，****它是\_\_\_\_\_\_\_\_**”function”**\_\_\_\_\_\_\_\_\_。**

**3、定义了一个变量，但没有为该变量赋值，如果alert该变量，javascript弹出的对话框中显示\_\_\_**undefined**\_\_\_\_\_\_ 。**