[HTML/CSS 7](#_Toc20526)

[1.HTML5 新的 DocType 和 Charset 是什么？ 7](#_Toc14227)

[2.如何在 HTML5 页面中嵌入音频? 7](#_Toc13893)

[3.如何在 HTML5 页面中嵌入视频？ 7](#_Toc27775)

[4.除了音频和视频，HTML5 还支持其他什么新的媒体元素？ 7](#_Toc12)

[5. HTML5 有哪些不同类型的存储？ 7](#_Toc31524)

[6.HTML5 引入什么新的表单属性？ 7](#_Toc8018)

[7. 与 HTML4 比较，HTML5 废弃了哪些元素？ 8](#_Toc3669)

[8. HTML5 标准提供了哪些新的 API？ 8](#_Toc6868)

[9. HTML5 应用缓存和常规的 HTML 浏览器缓存有何差别？ 8](#_Toc25648)

[10.Doctype 作用? 严格模式与混杂模式如何区分？它们有何意义? 9](#_Toc2373)

[11.行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？ 9](#_Toc28026)

[12.link 和@import 的区别是？ 9](#_Toc428)

[13.浏览器的内核分别是什么? 9](#_Toc4371)

[14.什么是盒子模型？ 9](#_Toc5014)

[16.列出display的值，说明他们的作用。position的值， relative和absolute定位原点是？ 10](#_Toc31702)

[17.CSS3有哪些新特性？ 11](#_Toc11755)

[18.为什么要初始化css样式。 11](#_Toc26779)

[19.权重 11](#_Toc20847)

[20.display:none和visibility:hidden的区别？ 12](#_Toc26939)

[21. display:none和visibility:hidden的区别？ 12](#_Toc5164)

[22.position:absolute和float属性的异同 12](#_Toc16583)

[23.介绍一下box-sizing属性？ 12](#_Toc25952)

[24.CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？CSS3新增伪类有那些？ 12](#_Toc8265)

[24.说说你对语义化的理解？ 13](#_Toc15778)

[25. Doctype作用? 严格模式与混杂模式如何区分？它们有何意义? 13](#_Toc25294)

[26. 你知道多少种Doctype文档类型？ 13](#_Toc1364)

[27.常见兼容性问题？ 14](#_Toc2703)

[28.解释下浮动和它的工作原理？清除浮动的技巧 15](#_Toc2987)

[29.浮动元素引起的问题和解决办法？ 15](#_Toc4195)

[30.清除浮动的几种方法： 15](#_Toc31428)

[31.请说出三种减少页面加载时间的方法。 15](#_Toc24562)

[32.有没有关注html5和CSS3?如有请简单说过一下对他们的了解 16](#_Toc22951)

[33.Doctype? 严格模式与混杂模式-如何触发这两种模式，区分它们有何意义? 16](#_Toc28216)

[34.前端页面由哪三层构成，分别是什么？作用是什么？ 16](#_Toc29482)

[35.如何居中一个浮动元素? 17](#_Toc1363)

[37.xhtml与html的区别 17](#_Toc28938)

[38.HTML含义 历史 17](#_Toc30657)

[39.CSS3 18](#_Toc13497)

[40.CSS和CSS3的区别 18](#_Toc15980)

[41.如果让你来制作一个访问量很高的大型网站，你会如何来管理所有CSS文件、JS与图片？ 18](#_Toc944)

[JavaScript 18](#_Toc24634)

[1.JavaScript原型，原型链 ? 有什么特点？ 19](#_Toc2891)

[2.eval是做什么的？ 19](#_Toc14075)

[3.介绍js的基本数据类型。 19](#_Toc8305)

[4.http状态码有那些？分别代表是什么意思？ 19](#_Toc1259)

[5.说说你对闭包的理解 19](#_Toc411)

[6.请你谈谈Cookie的弊端 19](#_Toc10019)

[7.浏览器本地存储 20](#_Toc22006)

[8.web storage和cookie的区别 20](#_Toc21675)

[9.cookie 和session 的区别： 21](#_Toc15028)

[10.DOM操作——怎样添加、移除、移动、复制、创建和查找节点。 21](#_Toc5903)

[11.null和undefined的区别？ 21](#_Toc30666)

[12. new操作符具体干了什么呢? 22](#_Toc20643)

[13.js延迟加载的方式有哪些？ 22](#_Toc32681)

[14.如何解决跨域问题? 22](#_Toc1496)

[15.documen.write和 innerHTML的区别 22](#_Toc14772)

[16.call() 和 .apply() 的区别和作用？ 22](#_Toc2644)

[17.哪些操作会造成内存泄漏？ 22](#_Toc21895)

[18.什么叫优雅降级和渐进增强？ 22](#_Toc30633)

[19.对前端界面工程师这个职位是怎么样理解的？它的前景会怎么样？ 23](#_Toc5445)

[20.你有哪些性能优化的方法？ 23](#_Toc17116)

[21.一个页面从输入 URL 到页面加载显示完成，这个过程中都发生了什么？ 23](#_Toc12168)

[22.平时如何管理你的项目？ 24](#_Toc12065)

[23.javascript对象的几种创建方式 24](#_Toc11693)

[24.javascript继承的6种方法 24](#_Toc3956)

[25.ajax过程 24](#_Toc4489)

[26.异步加载和延迟加载 25](#_Toc5624)

[27.前端安全问题？ 25](#_Toc9683)

[28.javascript里面的继承怎么实现，如何避免原型链上面的对象共享 25](#_Toc10046)

[29.Flash、Ajax各自的优缺点，在使用中如何取舍？ 25](#_Toc14014)

[30. 请解释一下 JavaScript 的同源策略。 25](#_Toc6374)

[31.GET和POST的区别，何时使用POST？ 25](#_Toc7332)

[32.嵌入JS应该放在什么位置？ 26](#_Toc15934)

[33.Javascript无阻塞加载具体方式 26](#_Toc4261)

[34.eval是做什么的？ 26](#_Toc31424)

[35.事件、IE与火狐的事件机制有什么区别？ 如何阻止冒泡？ 26](#_Toc17322)

[36.ajax 是什么?ajax 的交互模型?同步和异步的区别?如何解决跨域问题? 27](#_Toc4517)

[37.js对象的深度克隆 27](#_Toc12856)

[38.js数组去重 28](#_Toc25317)

[39.HTTP状态码 28](#_Toc17377)

[40.js操作获取和设置cookie 29](#_Toc2390)

[41.谈谈性能优化问题 30](#_Toc31591)

[42.移动端性能优化 30](#_Toc10968)

[43.栈和队列的区别? 30](#_Toc25799)

[44.栈和堆的区别？ 30](#_Toc4771)

[45.关于Http 2.0 你知道多少？ 30](#_Toc6341)

[浏览器兼容问题 31](#_Toc12078)

[Ø JavaScript 31](#_Toc10675)

[1.HTML对象获取问题 31](#_Toc6231)

[2. const问题 31](#_Toc11138)

[3. event.x与event.y问题 31](#_Toc10155)

[4. window.location.href问题 32](#_Toc22887)

[5. frame问题 32](#_Toc15354)

[6. 模态和非模态窗口问题 32](#_Toc11673)

[7. firefox与IE的父元素(parentElement)的区别 32](#_Toc19112)

[8. document.formName.item(”itemName”) 问题 33](#_Toc9192)

[9. 集合类对象问题 33](#_Toc13392)

[10. 自定义属性问题 33](#_Toc26186)

[11. input.type属性问题 33](#_Toc16311)

[12. event.srcElement问题 33](#_Toc23182)

[13. body载入问题 33](#_Toc29987)

[14. 事件委托方法 33](#_Toc715)

[15. Table操作问题 34](#_Toc9610)

[16. 对象宽高赋值问题 34](#_Toc28247)

[Ø CSS 34](#_Toc1090)

[1.cursor:hand   VS   cursor:pointer 34](#_Toc10043)

[2. innerText在IE中能正常工作，但在FireFox中却不行.    34](#_Toc12049)

[3. CSS透明 34](#_Toc10368)

[4. css中的width和padding 34](#_Toc8138)

[5. FF和IEBOX模型解释不一致导致相差2px 35](#_Toc3031)

[6. IE5 和IE6的BOX解释不一致 35](#_Toc7139)

[7. ul和ol列表缩进问题 35](#_Toc20823)

[8 元素水平居中问题 35](#_Toc6564)

[9. Div的垂直居中问题 35](#_Toc31344)

[10. margin加倍的问题 35](#_Toc4063)

[11. IE与宽度和高度的问题 36](#_Toc14667)

[12. 页面的最小宽度 36](#_Toc3151)

[13. DIV浮动IE文本产生3象素的bug 36](#_Toc16413)

[14.IE捉迷藏的问题 36](#_Toc3603)

[15.float的div闭合;清除浮动;自适应高度 37](#_Toc22597)

[16. 高度不适应 38](#_Toc19680)

[17. IE6下图片下有空隙产生 38](#_Toc26413)

[18. 对齐文本与文本输入框 38](#_Toc6877)

[19. LI中内容超过长度后以省略号显示 38](#_Toc30523)

[20. 为什么web标准中IE无法设置滚动条颜色了 39](#_Toc15943)

[21. 为什么无法定义1px左右高度的容器 39](#_Toc19807)

[22.怎么样才能让层显示在FLASH之上呢 39](#_Toc18767)

[23. 链接(a标签)的边框与背景 39](#_Toc6146)

[24. 超链接访问过后hover样式就不出现的问题 40](#_Toc27091)

[25. FORM标签 40](#_Toc27222)

[26. 属性选择器(这个不能算是兼容,是隐藏css的一个bug) 40](#_Toc14296)

[27. 为什么FF下文本无法撑开容器的高度 40](#_Toc3939)

[28. IE和FireFox 对空格的尺寸解释不同，FireFox为4px,IE为8px; FireFox对div与div 40](#_Toc249)

[29. 条件注释 41](#_Toc23111)

[30.强制渲染 41](#_Toc16417)

[31.js兼容文件 42](#_Toc1394)

[32. 浏览器识别符 42](#_Toc11345)

[33.div的垂直居中问题 42](#_Toc245)

[34.margin加倍的问题 43](#_Toc24118)

[35.浮动ie产生的双倍距离 43](#_Toc20507)

[36.DIV浮动IE文本产生3象素的bug 43](#_Toc19425)

[37.IE6下为什么图片下有空隙产生 43](#_Toc22751)

[38.如何对齐文本与文本输入框 43](#_Toc4838)

[39.为什么web标准中IE无法设置滚动条颜色了 44](#_Toc23550)

[40.为什么无法定义1px左右高度的容器 44](#_Toc3230)

[41.怎么样才能让层显示在FLASH之上呢 44](#_Toc17079)

[42.链接(a标签)的边框与背景 44](#_Toc17772)

[43.超链接访问过后hover样式就不出现的问题 44](#_Toc6013)

[44.游标手指cursor 44](#_Toc30512)

[45.UL的padding与margin 45](#_Toc26903)

[46.IE与宽度和高度的问题 45](#_Toc26703)

[47.页面的最小宽度 45](#_Toc5138)

[48.DIV浮动IE文本产生3象素的bug 45](#_Toc23542)

[49.属性选择器(这个不能算是兼容,是隐藏css的一个bug) 46](#_Toc3184)

[50.高度不适应 46](#_Toc25990)

[Javascript编程题 46](#_Toc20966)

[1.实现以下函数 46](#_Toc16458)

[2.写出注释处的a,b,c的值 47](#_Toc19267)

[3.请实现以下代码 47](#_Toc5600)

[5.请指出以下person1和person2的区别，以及this的指向： 48](#_Toc27040)

[6.如何用javascript获取所有选中的checkbox值 48](#_Toc12550)

[7.js继承有5种实现方式： 49](#_Toc14995)

# HTML/CSS

1.HTML5 新的 DocType 和 Charset 是什么？

HTML5 现在已经不是 SGML 的子集，DocType 简化为： <!doctype html>

HTML 5 指定 UTF-8 编码的方式如下：<meta charset="UTF-8"> / 定义编码格式

2.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>

3.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>

4.除了音频和视频，HTML5 还支持其他什么新的媒体元素？

HTML 5 对媒体支持很强，除了 audio 和 video 外，还提供：

<embed> 作为外部应用的容器

<track> 定义媒体的文本跟踪

<source> 对多种媒体源的支持很有帮助

5. HTML5 有哪些不同类型的存储？

HTML 5 支持本地存储，在之前版本中是通过 Cookie 实现的。HTML5 本地存储速度快而且安全。

有两种不同的对象可用来存储数据：

localStorage 适用于长期存储数据，浏览器关闭后数据不丢失

sessionStorage 存储的数据在浏览器关闭后自动删除

6.HTML5 引入什么新的表单属性？

HTML5 引入大量新的表单属性：

datalist

datetime

output

keygen

date

month

week

time

number

range

email

Url

7. 与 HTML4 比较，HTML5 废弃了哪些元素？

废弃的元素包括：

frame

frameset

noframe

applet

big

center

Basefront

8. HTML5 标准提供了哪些新的 API？

HTML 5 提供很多新的 API，包括：

Media API

Text Track API

Application Cache API

User Interaction

Data Transfer API

Command API

Constraint Validation API

History API

and many more....

9. HTML5 应用缓存和常规的 HTML 浏览器缓存有何差别？

HTML5 的应用缓存最关键的就是支持离线应用，可获取少数或者全部网站内容，包括 HTML、CSS、图像和 JavaScript 脚本并存在本地。该特性加速了网站的性能，可通过如下方式实现：

<!doctype html>

<html manifest="example.appcache">

.....

</html>

与传统的浏览器缓存比较，该特性并不强制要求用户访问网站。

10.Doctype 作用? 严格模式与混杂模式如何区分？它们有何意义?

1.<!DOCTYPE> 声明位于文档中的最前面，处于 <html> 标签之前。告知浏览器的解析器,用什么文档类型 规范来解析这个文档。

2.严格模式的排版和 JS 运作模式是 以该浏览器支持的最高标准运行。

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

4.DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。

11.行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？

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.

行内元素：a、b、span、img、input、strong、select、label、em、button、textarea

块级元素：div、ul、li、dl、dt、dd、p、h1-h6、blockquote

空元素：即系没有内容的HTML元素，例如：br、meta、hr、link

12.link 和@import 的区别是？

①．link属于XHTML标签，而@import是CSS提供的;

②．页面被加载的时，link会同时被加载，而@import引用的CSS会等到页面被加载完再加载;

③．import只在IE5以上才能识别，而link是XHTML标签，无兼容问题;

④．link方式的样式的权重 高于@import的权重.

13.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)

14.什么是盒子模型？

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

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

\* 1.id选择器（ # myid）

2.类选择器（.myclassname）

3.标签选择器（div, h1, p）

4.相邻选择器（h1 + p）

5.子选择器（ul < li）

6.后代选择器（li a）

7.通配符选择器（ \* ）

8.属性选择器（a[rel = "external"]）

9.伪类选择器（a: hover, li: nth - child）

\* 可继承的样式： font-size font-family color, UL LI DL DD DT;

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

优先级就近原则，同权重情况下样式定义最近者为准;

\* 载入样式以最后载入的定位为准;

\* 优先级为:

!important > id > class > tag important 比 内联优先级高

\* 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。

16.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。象行内元素类型一样显示。

inline-block 象行内元素一样显示，但其内容象块类型元素一样显示。

list-item 象块类型元素一样显示，并添加样式列表标记。

2.

\* absolute

生成绝对定位的元素，相对于 static 定位以外的第一个父元素进行定位。

\* 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。

17.CSS3有哪些新特性？

CSS3实现圆角（border-radius:8px），阴影（box-shadow:10px），对文字加特效（text-shadow、），线性渐变（gradient），旋转（transform）transform:rotate(9deg) scale(0.85,0.90) translate(0px,-30px) skew(-9deg,0deg);//旋转,缩放,定位,倾斜

18.为什么要初始化css样式。

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

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

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

淘宝的样式初始化：

body, h1, h2, h3, h4, h5, h6, hr, p, blockquote, dl, dt, dd, ul, ol, li, pre, form, fieldset, legend, button, input, textarea, th, td { margin:0; padding:0; }

body, button, input, select, textarea { font:12px/1.5tahoma,arial, \5b8b\4f53; }

h1, h2, h3, h4, h5, h6{ font-size:100%; }

address, cite, dfn, em, var { font-style:normal; }

code, kbd, pre, samp { font-family:couriernew, courier,monospace; }

small{ font-size:12px; }

ul, ol { list-style:none; }

a { text-decoration:none; }

a:hover { text-decoration:underline; }

sup { vertical-align:text-top; }

sub{ vertical-align:text-bottom; }

legend { color:#000; }

fieldset, img { border:0; }

button, input, select, textarea { font-size:100%; }

table { border-collapse:collapse; border-spacing:0; }

19.权重

从CSS代码存放位置看权重优先级：

内嵌样式 > 内部样式表 > 外联样式表。其实这个基本可以忽视之，大部分情况下CSS代码都是使用外联样式表。

从样式选择器看权重优先级：

important > 内嵌样式 > ID > 类 > 标签 | 伪类 | 属性选择 > 伪对象 > 继承 > 通配符。

important的权重为1,0,0,0

ID的权重为0,1,0,0

类的权重为0,0,1,0

标签的权重为0,0,0,1

伪类的权重为0,0,1,0

属性的权重为0,0,1,0

伪对象的权重为0,0,0,1

通配符的权重为0,0,0,0

20.display:none和visibility:hidden的区别？

display:none 隐藏对应的元素，在文档布局中不再给它分配空间，它各边的元素会合拢，就当他从来不存在。

visibility:hidden 隐藏对应的元素，但是在文档布局中仍保留原来的空间。

21. display:none和visibility:hidden的区别？

display:none 隐藏对应的元素，在文档布局中不再给它分配空间，它各边的元素会合拢，就当他从来不存在。

visibility:hidden 隐藏对应的元素，但是在文档布局中仍保留原来的空间。

22.position:absolute和float属性的异同

A：共同点：

对内联元素设置`float`和`absolute`属性，可以让元素脱离文档流，并且可以设置其宽高。

B：不同点：

float仍会占据位置，position会覆盖文档流中的其他元素。

23.介绍一下box-sizing属性？

box-sizing属性主要用来控制元素的盒模型的解析模式。默认值content-box。

content-box：让元素维持W3C的标准盒模型。元素的宽度/高度由border + padding + content的宽度/高度决定，设置width/height属性指的是content部分的宽/高

border-box：让元素维持IE传统盒模型（IE6以下版本和IE6~7的怪异模式）。设置width/height属性指的是border + padding + content

标准浏览器下，按照W3C规范对盒模型解析，一旦修改了元素的边框或内距，就会影响元素的盒子尺寸，就不得不重新计算元素的盒子尺寸，从而影响整个页面的布局。

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

1.id选择器（ # myid）

2.类选择器（.myclassname）

3.标签选择器（div, h1, p）

4.相邻选择器（h1 + p）

5.子选择器（ul > li）

6.后代选择器（li a）

7.通配符选择器（ \* ）

8.属性选择器（a[rel = "external"]）

9.伪类选择器（a: hover, li:nth-child）

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

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

优先级就近原则，同权重情况下样式定义最近者为准;

载入样式以最后载入的定位为准;

优先级为:

!important > id > class > tag

important 比 内联优先级高,但内联比 id 要高

CSS3新增伪类举例：

p:first-of-type 选择属于其父元素的首个 <p> 元素的每个 <p> 元素。

p:last-of-type 选择属于其父元素的最后 <p> 元素的每个 <p> 元素。

p:only-of-type 选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。

p:only-child 选择属于其父元素的唯一子元素的每个 <p> 元素。

p:nth-child(2) 选择属于其父元素的第二个子元素的每个 <p> 元素。

:enabled :disabled 控制表单控件的禁用状态。

:checked 单选框或复选框被选中。

24.说说你对语义化的理解？

(1) 去掉或者丢失样式的时候能够让页面呈现出清晰的结构

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

(3) 方便其他设备解析（如屏幕阅读器、盲人阅读器、移动设备）以意义的方式来渲染网页；

(4) 便于团队开发和维护，语义化更具可读性，是下一步吧网页的重要动向，遵循W3C标准的团队都遵循这个标准，可以减少差异化。

25. Doctype作用? 严格模式与混杂模式如何区分？它们有何意义?

（1）<!DOCTYPE> 声明位于文档中的最前面，处于 <html> 标签之前。告知浏览器以何种模式来渲染文档。

（2）严格模式的排版和 JS 运作模式是 以该浏览器支持的最高标准运行。

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

（4）DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。

26. 你知道多少种Doctype文档类型？

该标签可声明三种 DTD 类型，分别表示严格版本、过渡版本以及基于框架的 HTML 文档。

HTML 4.01 规定了三种文档类型：Strict、Transitional 以及 Frameset。

XHTML 1.0 规定了三种 XML 文档类型：Strict、Transitional 以及 Frameset。

Standards （标准）模式（也就是严格呈现模式）用于呈现遵循最新标准的网页，而 Quirks

（包容）模式（也就是松散呈现模式或者兼容模式）用于呈现为传统浏览器而设计的网页。

27.常见兼容性问题？

\* png24位的图片在iE6浏览器上出现背景，解决方案是做成PNG8.也可以引用一段脚本处理.

\* 浏览器默认的margin和padding不同。解决方案是加一个全局的\*{margin:0;padding:0;}来统一。

\* IE6双边距bug:块属性标签float后，又有横行的margin情况下，在ie6显示margin比设置的大。

\* 浮动ie产生的双倍距离（IE6双边距问题：在IE6下，如果对元素设置了浮动，同时又设置了margin-left或margin-right，margin值会加倍。）

#box{ float:left; width:10px; margin:0 0 0 100px;}

这种情况之下IE会产生20px的距离，解决方案是在float的标签样式控制中加入 ——\_display:inline;将其转化为行内属性。(\_这个符号只有ie6会识别)

\* 渐进识别的方式，从总体中逐渐排除局部。

首先，巧妙的使用“\9”这一标记，将IE游览器从所有情况中分离出来。

接着，再次使用“+”将IE8和IE7、IE6分离开来，这样IE8已经独立识别。

css

.bb{

background-color:#f1ee18;/\*所有识别\*/

.background-color:#00deff\9; /\*IE6、7、8识别\*/

+background-color:#a200ff;/\*IE6、7识别\*/

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

}

\* IE下,可以使用获取常规属性的方法来获取自定义属性,

也可以使用getAttribute()获取自定义属性;

Firefox下,只能使用getAttribute()获取自定义属性.

解决方法:统一通过getAttribute()获取自定义属性.

\* IE下,event对象有x,y属性,但是没有pageX,pageY属性;

Firefox下,event对象有pageX,pageY属性,但是没有x,y属性.

\* 解决方法：（条件注释）缺点是在IE浏览器下可能会增加额外的HTTP请求数。

\* Chrome 中文界面下默认会将小于 12px 的文本强制按照 12px 显示,

可通过加入 CSS 属性 -webkit-text-size-adjust: none; 解决.

\* 超链接访问过后hover样式就不出现了 被点击访问过的超链接样式不在具有hover和active了解决方法是改变CSS属性的排列顺序:

L-V-H-A : a:link {} a:visited {} a:hover {} a:active {}

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

\* 上下margin重合问题

ie和ff都存在，相邻的两个div的margin-left和margin-right不会重合，但是margin-top和margin-bottom却会发生重合。

解决方法，养成良好的代码编写习惯，同时采用margin-top或者同时采用margin-bottom。

\* ie6对png图片格式支持不好(引用一段脚本处理)

28.解释下浮动和它的工作原理？清除浮动的技巧

浮动元素脱离文档流，不占据空间。浮动元素碰到包含它的边框或者浮动元素的边框停留。

1.使用空标签清除浮动。

这种方法是在所有浮动标签后面添加一个空标签 定义css clear:both. 弊端就是增加了无意义标签。

2.使用overflow。

给包含浮动元素的父标签添加css属性 overflow:auto; zoom:1; zoom:1用于兼容IE6。

3.使用after伪对象清除浮动。

该方法只适用于非IE浏览器。具体写法可参照以下示例。使用中需注意以下几点。一、该方法中必须为需要清除浮动元素的伪对象中设置 height:0，否则该元素会比实际高出若干像素；

29.浮动元素引起的问题和解决办法？

浮动元素引起的问题：

（1）父元素的高度无法被撑开，影响与父元素同级的元素

（2）与浮动元素同级的非浮动元素（内联元素）会跟随其后

（3）若非第一个元素浮动，则该元素之前的元素也需要浮动，否则会影响页面显示的结构

解决方法：

使用CSS中的clear:both;属性来清除元素的浮动可解决2、3问题，对于问题1，添加如下样式，给父元素添加clearfix样式：

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

.clearfix{display: inline-block;} /\* for IE/Mac \*/

30.清除浮动的几种方法：

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

2，使用after伪类

#parent:after{

content:".";

height:0;

visibility:hidden;

display:block;

clear:both;

}

3,浮动外部元素

4,设置`overflow`为`hidden`或者auto

31.请说出三种减少页面加载时间的方法。

1.优化图片

2.图像格式的选择（GIF：提供的颜色较少，可用在一些对颜色要求不高的地方）

3.优化CSS（压缩合并css，如margin-top,margin-left...)

4.网址后加斜杠（如www.campr.com/目录，会判断这个“目录是什么文件类型，或者是目录。）

5.标明高度和宽度（如果浏览器没有找到这两个参数，它需要一边下载图片一边计算大小，如果图片很多，浏览器需要不断地调整页面。这不但影响速度，也影响浏览体验。

当浏览器知道了高度和宽度参数后，即使图片暂时无法显示，页面上也会腾出图片的空位，然后继续加载后面的内容。从而加载时间快了，浏览体验也更好了。）

6.减少http请求（合并文件，合并图片）。

32.有没有关注html5和CSS3?如有请简单说过一下对他们的了解

HTML5是用于取代1999年所制定的 HTML 4.01 和 XHTML 1.0 标准的 HTML 标准版本，现在仍处于发展阶段，但大部分浏览器已经支持某些 HTML5 技术。HTML 5有两大特点：首先，强化了 Web 网页的表现性能。其次，

据库等 Web 应用的功能。广义论及HTML5时，实际指的是包括HTML、CSS和JavaScript在内的一套技术组合。它希望能够减少浏览器对于需要插件的丰富性网络应用服务（plug-in-based rich internet application，RIA)，如Adobe Flash、Microsoft Silverlight，与Oracle JavaFX的需求，并且提供更多能有效增强网络应用的标准集。

CSS3特性(Class: CSS3)

在不牺牲性能和语义结构的前提下，CSS3中提供了更多的风格和更强的效果。此外，较之以前的Web排版，Web的开放字体格式（WOFF）也提供了更高的灵活性和控制性。

33.Doctype? 严格模式与混杂模式-如何触发这两种模式，区分它们有何意义?

Doctype声明位于文档中的最前面的位置，处于标签之前。此标签可告知浏览器文档使用哪种 HTML 或 XHTML 规范。该标签可声明三种DTD 类型，分别表示严格版本、过渡版本以及基于框架的 HTML 文档。

当浏览器厂商开始创建与标准兼容的浏览器时，他们希望确保向后兼容性。为了实现这一点，他们创建了两种呈现模式：标准模式和混杂模式（quirks mode）。在标准模式中，浏览器根据规范呈现页面；在混杂模式中，页面以一种比较宽松的向后兼容的方式显示。混杂模式通常模拟老式浏览器（比如Microsoft IE 4和Netscape Navigator 4）的行为以防止老站点无法工作。

浏览器根据DOCTYPE是否存在以及使用的哪种DTD来选择要使用的呈现方法。如果XHTML文档包含形式完整的DOCTYPE，那么它一般以标准模式呈现。对于HTML 4.01文档，包含严格DTD的DOCTYPE常常导致页面以标准模式呈现。包含过渡DTD和URI的DOCTYPE也导致页面以标准模式呈现，但是有过渡DTD而没有URI会导致页面以混杂模式呈现。DOCTYPE不存在或形式不正确会导致HTML和XHTML文档以混杂模式呈现。

34.前端页面由哪三层构成，分别是什么？作用是什么？

网页分成三个层次，即：结构层、表示层、行为层。

网页的结构层（structural layer）由 HTML 或 XHTML 之类的标记语言负责创建。标签，也就是那些出现在尖括号里的单词，对网页内容的语义含义做出了描述，但这些标签不包含任何关于如何显示有关内容的信息。例如，P 标签表达了这样一种语义：“这是一个文本段。”

网页的表示层（presentation layer） 由 CSS 负责创建。 CSS 对“如何显示有关内容”的问题做出了回答。

网页的行为层（behavior layer）负责回答“内容应该如何对事件做出反应”这一问题。这是 Javascript 语言和 DOM 主宰的领域。

35.如何居中一个浮动元素?

设置容器的浮动方式为相对定位，然后确定容器的宽高，比如宽500 高 300 的层，然后设置层的外边距。

div{Width:500px;height:300px;Margin: -150px 0 0 -250px;position:relative;left:50%;top:50%;}

36.对于元素的定位有哪些？其中对Z-index样式的理解及z-index不同浏览器默认的样式是什么？

答：定位有：relative absolute static fixed inherit

理解：Z-index属性决定了一个HTML元素的层叠级别。元素层叠级别是相对于元素在Z轴上（与X轴Y轴相对照）的位置而言。一个更高的Z-index值意味着这个元素在叠层顺序中会更靠近顶部。这个层叠顺序沿着垂直的线轴被呈现。

37.xhtml与html的区别

XHTML 要求正确嵌套

　XHTML 所有元素必须关闭

　XHTML 区分大小写

　XHTML 属性值要用双引号

　XHTML 用 id 属性代替 name 属性

　XHTML 特殊字符的处理

　XHTML 要求正确嵌套

38.HTML含义 历史

HTML的英文全称是hyptr text markup language，即超文本标记语言。HTML5是HTML的一个新版本。HTML不是一种编程语言，而是一种标记语言。

历史：HTML5草案的前身名为web applications 1.0.於2014年被WHATWG提出，於2004年被w3c接纳，并成立了新的HTML工作团队。在2008年1月22日，第一份正式草案已公布，2010年9月正式向公众推荐。

HTML5 将成为 HTML、XHTML 以及 HTML DOM 的新标准。

HTML 的上一个版本诞生于 1999 年。自从那以后，Web 世界已经经历了巨变。

HTML5 仍处于完善之中。然而，大部分现代浏览器已经具备了某些 HTML5 支持。

39.CSS3

CSS即层叠样式表（Cascading StyleSheet）。 在网页制作时采用层叠样式表技术，可以有效地对页面的布局、字体、颜色、背景和其它效果实现更加精确的控制。 只要对相应的代码做一些简单的修改，就可以改变同一页面的不同部分，或者页数不同的网页的外观和格式。

CSS3是CSS技术的升级版本，CSS3语言开发是朝着模块化发展的。以前的规范作为一个模块实在是太庞大而且比较复杂，所以，把它分解为一些小的模块，更多新的模块也被加入进来。这些模块包括： 盒子模型、列表模块、超链接方式 、语言模块 、背景和边框 、文字特效 、多栏布局等。

40.CSS和CSS3的区别

都是网页样式code，不同的在于css3比css多了一些样式设置而已。css3是向前兼容的，也就是说，css中有效的code在css3也有效。以上是纯code方面来看，而对于具体的显示效果，还要看浏览器对其的支持情况。

41.如果让你来制作一个访问量很高的大型网站，你会如何来管理所有CSS文件、JS与图片？

回答：涉及到人手、分工、同步；

先期团队必须确定好全局样式（globe.css），编码模式(utf-8) 等

编写习惯必须一致（例如都是采用继承式的写法，单样式都写成一行）；

标注样式编写人，各模块都及时标注（标注关键样式调用的地方）；

页面进行标注（例如 页面 模块 开始和结束）；

CSS跟HTML 分文件夹并行存放，命名都得统一（例如style.css）

JS 分文件夹存放 命民以该JS 功能为准英文翻译；

图片采用整合的 images.png png8 格式文件使用尽量整合在一起使用方便将来的管理

# JavaScript

1.JavaScript原型，原型链 ? 有什么特点？

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

\* 原型链是由一些用来继承和共享属性的对象组成的（有限的）对象链。

2.eval是做什么的？

它的功能是把对应的字符串解析成JS代码并运行；

应该避免使用eval，不安全，非常耗性能（2次，一次解析成js语句，一次执行）。

3.介绍js的基本数据类型。

number,string,boolean,object,undefined

4.http状态码有那些？分别代表是什么意思？

100-199 用于指定客户端应相应的某些动作。

200-299 用于表示请求成功。

300-399 用于已经移动的文件并且常被包含在定位头信息中指定新的地址信息。

400-499 用于指出客户端的错误。400 1、语义有误，当前请求无法被服务器理解。401 当前请求需要用户验证 403 服务器已经理解请求，但是拒绝执行它。

500-599 用于支持服务器错误。 503 – 服务不可用

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

5.说说你对闭包的理解

使用闭包主要是为了设计私有的方法和变量。闭包的优点是可以避免全局变量的污染，缺点是闭包会常驻内存，会增大内存使用量，使用不当很容易造成内存泄露。

闭包有三个特性:

1.函数嵌套函数

2.函数内部可以引用外部的参数和变量

3.参数和变量不会被垃圾回收机制回收

2.

6.请你谈谈Cookie的弊端

cookie虽然在持久保存客户端数据提供了方便，分担了服务器存储的负担，但还是有很多局限性的。

第一：每个特定的域名下最多生成20个cookie

1.IE6或更低版本最多20个cookie

2.IE7和之后的版本最后可以有50个cookie。

3.Firefox最多50个cookie

4.chrome和Safari没有做硬性限制

IE和Opera 会清理近期最少使用的cookie，Firefox会随机清理cookie。

cookie的最大大约为4096字节，为了兼容性，一般不能超过4095字节。

IE 提供了一种存储可以持久化用户数据，叫做userdata，从IE5.0就开始支持。每个数据最多128K，每个域名下最多1M。这个持久化数据放在缓存中，如果缓存没有清理，那么会一直存在。

优点：极高的扩展性和可用性

1.通过良好的编程，控制保存在cookie中的session对象的大小。

2.通过加密和安全传输技术（SSL），减少cookie被破解的可能性。

3.只在cookie中存放不敏感数据，即使被盗也不会有重大损失。

4.控制cookie的生命期，使之不会永远有效。

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。

7.浏览器本地存储

在较高版本的浏览器中，js提供了sessionStorage和globalStorage。在HTML5中提供了localStorage来取代globalStorage。

html5中的Web Storage包括了两种存储方式：sessionStorage和localStorage。

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。

8.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。浏览器的支持除了IE７及以下不支持外，其他标准浏览器都完全支持(ie及FF需在web服务器里运行)，值得一提的是IE总是办好事，例如IE7、IE6中的userData其实就是javascript本地存储的解决方案。通过简单的代码封装可以统一到所有的浏览器都支持web storage。

LocalStorage和sessionStorage都具有相同的操作方法，例如setItem、getItem和removeItem等

9.cookie 和session 的区别：

1、cookie数据存放在客户的浏览器上，session数据放在服务器上。

2、cookie不是很安全，别人可以分析存放在本地的COOKIE并进行COOKIE欺骗考虑到安全应当使用session。

3、session会在一定时间内保存在服务器上。当访问增多，会比较占用你服务器的性能考虑到减轻服务器性能方面，应当使用COOKIE。

4、单个cookie保存的数据不能超过4K，很多浏览器都限制一个站点最多保存20个 cookie。

5、所以个人建议：

将登陆信息等重要信息存放为SESSION

其他信息如果需要保留，可以放在COOKIE中

10.DOM操作——怎样添加、移除、移动、复制、创建和查找节点。

（1）创建新节点

createDocumentFragment() //创建一个DOM片段

createElement() //创建一个具体的元素

createTextNode() //创建一个文本节点

（2）添加、移除、替换、插入

appendChild()

removeChild()

replaceChild()

insertBefore() //在已有的子节点前插入一个新的子节点

（3）查找

getElementsByTagName() //通过标签名称

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

getElementById() //通过元素Id，唯一性

11.null和undefined的区别？

null是一个表示”无”的对象，转为数值时为0；undefined是一个表示”无”的原始值，转为数值时为NaN。

当声明的变量还未被初始化时，变量的默认值为undefined。

null用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象。

undefined表示”缺少值”，就是此处应该有一个值，但是还没有定义。典型用法是：

（1）变量被声明了，但没有赋值时，就等于undefined。

（2) 调用函数时，应该提供的参数没有提供，该参数等于undefined。

（3）对象没有赋值的属性，该属性的值为undefined。

（4）函数没有返回值时，默认返回undefined。

null表示”没有对象”，即该处不应该有值。典型用法是：

（1） 作为函数的参数，表示该函数的参数不是对象。

（2） 作为对象原型链的终点。

12. new操作符具体干了什么呢?

1、创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。

2、属性和方法被加入到 this 引用的对象中。

3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。

var obj = {};

obj.\_\_proto\_\_ = Base.prototype;

Base.call(obj);

13.js延迟加载的方式有哪些？

defer和async、动态创建DOM方式（创建script，插入到DOM中，加载完毕后callBack）、按需异步载入js

14.如何解决跨域问题?

jsonp、 document.domain+iframe、window.name、window.postMessage、服务器上设置代理页面

jsonp的原理是动态插入script标签

15.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

16.call() 和 .apply() 的区别和作用？

作用：动态改变某个类的某个方法的运行环境。

17.哪些操作会造成内存泄漏？

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

setTimeout 的第一个参数使用字符串而非函数的话，会引发内存泄漏。

闭包、控制台日志、循环（在两个对象彼此引用且彼此保留时，就会产生一个循环）

18.什么叫优雅降级和渐进增强？

优雅降级：Web站点在所有新式浏览器中都能正常工作，如果用户使用的是老式浏览器，则代码会检查以确认它们是否能正常工作。由于IE独特的盒模型布局问题，针对不同版本的IE的hack实践过优雅降级了,为那些无法支持功能的浏览器增加候选方案，使之在旧式浏览器上以某种形式降级体验却不至于完全失效.

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

19.对前端界面工程师这个职位是怎么样理解的？它的前景会怎么样？

前端是最贴近用户的程序员，比后端、数据库、产品经理、运营、安全都近。

1、实现界面交互

2、提升用户体验

3、有了Node.js，前端可以实现服务端的一些事情

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

参与项目，快速高质量完成实现效果图，精确到1px；

与团队成员，UI设计，产品经理的沟通；

做好的页面结构，页面重构和用户体验；

处理hack，兼容、写出优美的代码格式；

针对服务器的优化、拥抱最新前端技术。

20.你有哪些性能优化的方法？

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

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

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

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

（5） 少用全局变量、缓存DOM节点查找的结果。减少IO读取操作。

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

（7） 图片预加载，将样式表放在顶部，将脚本放在底部 加上时间戳。

21.一个页面从输入 URL 到页面加载显示完成，这个过程中都发生了什么？

分为4个步骤：

（1），当发送一个URL请求时，不管这个URL是Web页面的URL还是Web页面上每个资源的URL，浏览器都会开启一个线程来处理这个请求，同时在远程DNS服务器上启动一个DNS查询。这能使浏览器获得请求对应的IP地址。

（2）， 浏览器与远程Web服务器通过TCP三次握手协商来建立一个TCP/IP连接。该握手包括一个同步报文，一个同步-应答报文和一个应答报文，这三个报文在 浏览器和服务器之间传递。该握手首先由客户端尝试建立起通信，而后服务器应答并接受客户端的请求，最后由客户端发出该请求已经被接受的报文。

（3），一旦TCP/IP连接建立，浏览器会通过该连接向远程服务器发送HTTP的GET请求。远程服务器找到资源并使用HTTP响应返回该资源，值为200的HTTP响应状态表示一个正确的响应。

（4），此时，Web服务器提供资源服务，客户端开始下载资源。请求返回后，便进入了我们关注的前端模块,简单来说，浏览器会解析HTML生成DOM Tree，其次会根据CSS生成CSS Rule Tree，而javascript又可以根据DOM API操作DOM

22.平时如何管理你的项目？

先期团队必须确定好全局样式（globe.css），编码模式(utf-8) 等；

编写习惯必须一致（例如都是采用继承式的写法，单样式都写成一行）；

标注样式编写人，各模块都及时标注（标注关键样式调用的地方）；

页面进行标注（例如 页面 模块 开始和结束）；

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

JS 分文件夹存放 命名以该JS功能为准的英文翻译。

图片采用整合的 images.png png8 格式文件使用 尽量整合在一起使用方便将来的管理

23.javascript对象的几种创建方式

1，工厂模式

2，构造函数模式

3，原型模式

4，混合构造函数和原型模式

5，动态原型模式

6，寄生构造函数模式

7，稳妥构造函数模式

24.javascript继承的6种方法

1，原型链继承

2，借用构造函数继承

3，组合继承(原型+借用构造)

4，原型式继承

5，寄生式继承

6，寄生组合式继承

25.ajax过程

(1)创建XMLHttpRequest对象,也就是创建一个异步调用对象.

(2)创建一个新的HTTP请求,并指定该HTTP请求的方法、URL及验证信息.

(3)设置响应HTTP请求状态变化的函数.

(4)发送HTTP请求.

(5)获取异步调用返回的数据.

(6)使用JavaScript和DOM实现局部刷新.

26.异步加载和延迟加载

1.异步加载的方案： 动态插入script标签

2.通过ajax去获取js代码，然后通过eval执行

3.script标签上添加defer或者async属性

4.创建并插入iframe，让它异步执行js

5.延迟加载：有些 js 代码并不是页面初始化的时候就立刻需要的，而稍后的某些情况才需要的。

27.前端安全问题？

sql注入原理

就是通过把SQL命令插入到Web表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的SQL命令。

总的来说有以下几点：

1.永远不要信任用户的输入，要对用户的输入进行校验，可以通过正则表达式，或限制长度，对单引号和双"-"进行转换等。

2.永远不要使用动态拼装SQL，可以使用参数化的SQL或者直接使用存储过程进行数据查询存取。

3.永远不要使用管理员权限的数据库连接，为每个应用使用单独的权限有限的数据库连接。

4.不要把机密信息明文存放，请加密或者hash掉密码和敏感的信息。

28.javascript里面的继承怎么实现，如何避免原型链上面的对象共享

用构造函数和原型链的混合模式去实现继承，避免对象共享可以参考经典的extend()函数，很多前端框架都有封装的，就是用一个空函数当做中间变量

29.Flash、Ajax各自的优缺点，在使用中如何取舍？

1、Flash ajax对比

Flash适合处理多媒体、矢量图形、访问机器；对CSS、处理文本上不足，不容易被搜索。

Ajax对CSS、文本支持很好，支持搜索；多媒体、矢量图形、机器访问不足。

共同点：与服务器的无刷新传递消息、用户离线和在线状态、操作DOM

30. 请解释一下 JavaScript 的同源策略。

概念:同源策略是客户端脚本（尤其是Javascript）的重要的安全度量标准。它最早出自Netscape Navigator2.0，其目的是防止某个文档或脚本从多个不同源装载。

这里的同源策略指的是：协议，域名，端口相同，同源策略是一种安全协议。

指一段脚本只能读取来自同一来源的窗口和文档的属性。

31.GET和POST的区别，何时使用POST？

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

POST：一般用于修改服务器上的资源，对所发送的信息没有限制。

GET方式需要使用Request.QueryString来取得变量的值，而POST方式通过Request.Form来获取变量的值，

也就是说Get是通过地址栏来传值，而Post是通过提交表单来传值。

然而，在以下情况中，请使用 POST 请求：

无法使用缓存文件（更新服务器上的文件或数据库）

向服务器发送大量数据（POST 没有数据量限制）

发送包含未知字符的用户输入时，POST 比 GET 更稳定也更可靠

32.嵌入JS应该放在什么位置？

1、放在底部，虽然放在底部照样会阻塞所有呈现，但不会阻塞资源下载。

2、如果嵌入JS放在head中，请把嵌入JS放在CSS头部。

3、使用defer（只支持IE）

4、不要在嵌入的JS中调用运行时间较长的函数，如果一定要用，可以用`setTimeout`来调用

33.Javascript无阻塞加载具体方式

将脚本放在底部。<link>还是放在head中，用以保证在js加载前，能加载出正常显示的页面。<script>标签放在</body>前。

成组脚本：由于每个<script>标签下载时阻塞页面解析过程，所以限制页面的<script>总数也可以改善性能。适用于内联脚本和外部脚本。

非阻塞脚本：等页面完成加载后，再加载js代码。也就是，在window.onload事件发出后开始下载代码。

（1）defer属性：支持IE4和fierfox3.5更高版本浏览器

（2）动态脚本元素：文档对象模型（DOM）允许你使用js动态创建HTML的几乎全部文档内容。代码如下：

<script>

var script=document.createElement("script");

script.type="text/javascript";

script.src="file.js";

document.getElementsByTagName("head")[0].appendChild(script);

</script>

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

34.eval是做什么的？

它的功能是把对应的字符串解析成JS代码并运行；

应该避免使用eval，不安全，非常耗性能（2次，一次解析成js语句，一次执行）。

35.事件、IE与火狐的事件机制有什么区别？ 如何阻止冒泡？

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

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

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

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

1. 通过异步模式，提升了用户体验

2. 优化了浏览器和服务器之间的传输，减少不必要的数据往返，减少了带宽占用

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

Ajax的最大的特点是什么。

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

ajax的缺点

1、ajax不支持浏览器back按钮。

2、安全问题 AJAX暴露了与服务器交互的细节。

3、对搜索引擎的支持比较弱。

4、破坏了程序的异常机制。

5、不容易调试。

跨域： jsonp、 iframe、window.name、window.postMessage、服务器上设置代理页面

37.js对象的深度克隆

function clone(Obj) {

var buf;

if (Obj instanceof Array) {

buf = []; //创建一个空的数组

var i = Obj.length;

while (i--) {

buf[i] = clone(Obj[i]);

}

return buf;

}else if (Obj instanceof Object){

buf = {}; //创建一个空对象

for (var k in Obj) { //为这个对象添加新的属性

buf[k] = clone(Obj[k]);

}

return buf;

}else{

return Obj;

}

}

38.js数组去重

以下是数组去重的三种方法：

Array.prototype.unique1 = function () {

var n = []; //一个新的临时数组

for (var i = 0; i < this.length; i++) //遍历当前数组

{

//如果当前数组的第i已经保存进了临时数组，那么跳过，

//否则把当前项push到临时数组里面

if (n.indexOf(this[i]) == -1) n.push(this[i]);

}

return n;

}

Array.prototype.unique2 = function()

{

var n = {},r=[]; //n为hash表，r为临时数组

for(var i = 0; i < this.length; i++) //遍历当前数组

{

if (!n[this[i]]) //如果hash表中没有当前项

{

n[this[i]] = true; //存入hash表

r.push(this[i]); //把当前数组的当前项push到临时数组里面

}

}

return r;

}

Array.prototype.unique3 = function()

{

var n = [this[0]]; //结果数组

for(var i = 1; i < this.length; i++) //从第二项开始遍历

{

//如果当前数组的第i项在当前数组中第一次出现的位置不是i，

//那么表示第i项是重复的，忽略掉。否则存入结果数组

if (this.indexOf(this[i]) == i) n.push(this[i]);

}

return n;

}

39.HTTP状态码

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

200 OK 正常返回信息

201 Created 请求成功并且服务器创建了新的资源

202 Accepted 服务器已接受请求，但尚未处理

301 Moved Permanently 请求的网页已永久移动到新位置。

302 Found 临时性重定向。

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

304 Not Modified 自从上次请求后，请求的网页未修改过。（对于缓存来说非常重要）

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

401 Unauthorized 请求未授权。

403 Forbidden 禁止访问。

404 Not Found 找不到如何与 URI 相匹配的资源。

500 Internal Server Error 最常见的服务器端错误。

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

40.js操作获取和设置cookie

//创建cookie

function setCookie(name, value, expires, path, domain, secure) {

var cookieText = encodeURIComponent(name) + '=' + encodeURIComponent(value);

if (expires instanceof Date) {

cookieText += '; expires=' + expires;

}

if (path) {

cookieText += '; expires=' + expires;

}

if (domain) {

cookieText += '; domain=' + domain;

}

if (secure) {

cookieText += '; secure';

}

document.cookie = cookieText;

}

//获取cookie

function getCookie(name) {

var cookieName = encodeURIComponent(name) + '=';

var cookieStart = document.cookie.indexOf(cookieName);

var cookieValue = null;

if (cookieStart > -1) {

var cookieEnd = document.cookie.indexOf(';', cookieStart);

if (cookieEnd == -1) {

cookieEnd = document.cookie.length;

}

cookieValue = decodeURIComponent(document.cookie.substring(cookieStart + cookieName.length, cookieEnd));

}

return cookieValue;

}

//删除cookie

function unsetCookie(name) {

document.cookie = name + "= ; expires=" + new Date(0);

}

41.谈谈性能优化问题

代码层面：避免使用css表达式，避免使用高级选择器，通配选择器。

缓存利用：缓存Ajax，使用CDN，使用外部js和css文件以便缓存，添加Expires头，服务端配置Etag，减少DNS查找等

请求数量：合并样式和脚本，使用css图片精灵，初始首屏之外的图片资源按需加载，静态资源延迟加载。

请求带宽：压缩文件，开启GZIP，

42.移动端性能优化

尽量使用css3动画，开启硬件加速。适当使用touch事件代替click事件。避免使用css3渐变阴影效果。

尽可能少的使用box-shadow与gradients。box-shadow与gradients往往都是页面的性能杀手

43.栈和队列的区别?

栈的插入和删除操作都是在一端进行的，而队列的操作却是在两端进行的。

队列先进先出，栈先进后出。

栈只允许在表尾一端进行插入和删除，而队列只允许在表尾一端进行插入，在表头一端进行删除

44.栈和堆的区别？

栈区（stack）— 由编译器自动分配释放 ，存放函数的参数值，局部变量的值等。

堆区（heap） — 一般由程序员分配释放， 若程序员不释放，程序结束时可能由OS回收。

堆（数据结构）：堆可以被看成是一棵树，如：堆排序；

栈（数据结构）：一种先进后出的数据结构。

45.关于Http 2.0 你知道多少？

HTTP/2引入了“服务端推（serverpush）”的概念，它允许服务端在客户端需要数据之前就主动地将数据发送到客户端缓存中，从而提高性能。

HTTP/2提供更多的加密支持

HTTP/2使用多路技术，允许多个消息在一个连接上同时交差。

它增加了头压缩（header compression），因此即使非常小的请求，其请求和响应的header都只会占用很小比例的带宽。

46.jquery中的ajax如何去除缓存

$.ajaxSetup ({

cache: false //关闭AJAX相应的缓存

});

# 浏览器兼容问题

Ø JavaScript

1.HTML对象获取问题

FireFox：document.getElementById("idName");

ie:document.idname或者document.getElementById("idName").

解决办法：统一使用document.getElementById("idName");

2. const问题

说明:Firefox下,可以使用const关键字或var关键字来定义常量;

IE下,只能使用var关键字来定义常量.

解决方法：统一使用var关键字来定义常量.

3. event.x与event.y问题

说明:IE下,event对象有x,y属性,但是没有pageX,pageY属性;

Firefox下,event对象有pageX,pageY属性,但是没有x,y属性.

解决方法：使用mX(mX   =   event.x   ?   event.x   :   event.pageX;)来代替IE下的event.x或者Firefox下的event.pageX.

4. window.location.href问题

说明:IE或者Firefox2.0.x下,可以使用window.location或window.location.href;

Firefox1.5.x下,只能使用window.location.

解决方法：使用window.location来代替window.location.href.

5. frame问题

以下面的frame为例：

<frame   src="xxx.html"   id="frameId"   name="frameName"   />

(1)访问frame对象:

IE:使用window.frameId或者window.frameName来访问这个frame对象.   frameId和frameName可以同名。

Firefox:只能使用window.frameName来访问这个frame对象.

另外，在IE和Firefox中都可以使用window.document.getElementById("frameId")来访问这个frame对象.

(2)切换frame内容:

在 IE和Firefox中都可以使用window.document.getElementById("testFrame").src   =   "xxx.html"或window.frameName.location   =   "xxx.html"来切换frame的内容.

如果需要将frame中的参数传回父窗口(注意不是opener,而是parent   frame)，可以在frame中使用parent来访问父窗口。例如：parent.document.form1.filename.value="Aqing";

6. 模态和非模态窗口问题

说明:IE下,可以通过showModalDialog和showModelessDialog打开模态和非模态窗口;Firefox下则不能.

解决方法：直接使用window.open(pageURL,name,parameters)方式打开新窗口。

如果需要将子窗口中的参数传递回父窗口,可以在子窗口中使用window.opener来访问父窗口.

例如：var   parWin   =   window.opener;   parWin.document.getElementById("Aqing").value   =   "Aqing";

7. firefox与IE的父元素(parentElement)的区别

IE：obj.parentElement

firefox：obj.parentNode

解决方法:   因为firefox与IE都支持DOM,因此使用obj.parentNode是不错选择.

8. document.formName.item(”itemName”) 问题

问 题说明：IE下，可以使用document.formName.item(”itemName”) 或document.formName.elements ["elementName"]；Firefox 下，只能使用document.formName.elements["elementName"]。

解决方法：统一使用document.formName.elements["elementName"]。

9. 集合类对象问题

问题说明：IE下，可以使用 () 或 [] 获取集合类对象；Firefox下，只能使用 [ ]获取集合类对象。

解决方法：统一使用 [] 获取集合类对象。

10. 自定义属性问题

问题说明：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。

11. input.type属性问题

问题说明：IE下input.type属性为只读；但是Firefox下input.type属性为读写。

解决办法：不修改input.type属性。如果必须要修改，可以先隐藏原来的input，然后在同样的位置再插入一个新的input元素。

12. event.srcElement问题

问题说明：IE下，even对象有srcElement属性，但是没有target属性；Firefox下，even对象有target属性，但是没有srcElement属性。

解决方法：使用srcObj = event.srcElement ?event.srcElement : event.target;

如果考虑第8条问题，就改用myEvent代替event即可。

13. body载入问题

问题说明：Firefox的body对象在body标签没有被浏览器完全读入之前就存在；而IE的body对象则必须在body标签被浏览器完全读入之后才存在。

[注] 这个问题尚未实际验证，待验证后再来修改。

[注] 经验证，IE6、Opera9以及FireFox2中不存在上述问题，单纯的JS脚本可以访问在脚本之前已经载入的所有对象和元素，即使这个元素还没有载入完成。

14. 事件委托方法

问题说明：IE下，使用document.body.onload = inject; 其中function inject()在这之前已被实现；在Firefox下，使用document.body.onload = inject();

解决方法：统一使用document.body.onload=new Function(’inject()’); 或者document.body.onload = function(){}

[注意] Function和function的区别。

15. Table操作问题

问题说明：ie、firefox以及其它浏览器对于 table 标签的操作都各不相同，在ie中不允许对table和tr的innerHTML赋值，使用js增加一个tr时，使用appendChild方法也不管用。

解决方法：//向table追加一个空行：

var row = otable.insertRow(-1);var cell = document.createElement("td");cell.innerHTML = "";cell.className = "XXXX";row.appendChild(cell);[注] 由于俺很少使用JS直接操作表格，这个问题没有遇见过。建议使用JS框架集来操作table，如JQuery。

16. 对象宽高赋值问题

问题说明：FireFox中类似obj.style.height = imgObj.height的语句无效。

Ø CSS

1.cursor:hand   VS   cursor:pointer

firefox不支持hand，但ie支持pointer

解决方法:   统一使用pointer

2. innerText在IE中能正常工作，但在FireFox中却不行.

需用textContent。

解决方法:

if(navigator.appName.indexOf("Explorer")   >   -1){

        document.getElementById('element').innerText   =   "my   text";

}   else{

        document.getElementById('element').textContent   =   "my   text";

}

3. CSS透明

IE：filter:progid:DXImageTransform.Microsoft.Alpha(style=0,opacity=60)。

FF：opacity:0.6。

opacity 透明，子元素会继承透明属性。解决方式：1、使用 background:rgba(0,0,0,.6) //IE8及以下无效果。 2、使用定位，背景色与子元素处于同级关系。

4. css中的width和padding

在IE7和FF中width宽度不包括padding，在Ie6中包括padding.

5. FF和IEBOX模型解释不一致导致相差2px

box.style{width:100;border 1px;}

ie理解为box.width = 100

ff理解为box.width = 100 + 1\*2 = 102  //加上边框2px

解决方法：div{margin:30px!important;margin:28px;}

注意这两个margin的顺序一定不能写反， IE不能识别!important这个属性，但别的浏览器可以识别。所以在IE下其实解释成这样：div{maring:30px;margin:28px}

重复定义的话按照最后一个来执行，所以不可以只写margin:XXpx!important;

6. IE5 和IE6的BOX解释不一致

IE5下div{width:300px;margin:0 10px 0 10px;}

div 的宽度会被解释为300px-10px(右填充)-10px(左填充)，最终div的宽度为280px，而在IE6和其他浏览器上宽度则是以 300px+10px(右填充)+10px(左填充)=320px来计算的。这时我们可以做如下修改 div{width:300px!important;width :340px;margin:0 10px 0 10px}

7. ul和ol列表缩进问题

消除ul、ol等列表的缩进时，样式应写成：list-style:none;margin:0px;padding:0px;

经 验证，在IE中，设置margin:0px可以去除列表的上下左右缩进、空白以及列表编号或圆点，设置padding对样式没有影响；在 Firefox 中，设置margin:0px仅仅可以去除上下的空白，设置padding:0px后仅仅可以去掉左右缩进，还必须设置list- style:none才能去除列表编号或圆点。也就是说，在IE中仅仅设置margin:0px即可达到最终效果，而在Firefox中必须同时设置 margin:0px、 padding:0px以及list-style:none三项才能达到最终效果。

8 元素水平居中问题

FF: margin:0 auto;

IE: 父级{ text-align:center; }

9. Div的垂直居中问题

vertical-align:middle; 将行距增加到和整个DIV一样高：line-height:200px; 然后插入文字，就垂直居中了。缺点是要控制内容不要换行。

10. margin加倍的问题

设置为float的div在ie下设置的margin会加倍。这是一个ie6都存在的bug。解决方案是在这个div里面加上display:inline;

例如：

<div id=”imfloat”>

相应的css为

#imfloat{

float:left;

margin:5px;

display:inline;}

11. IE与宽度和高度的问题

IE不认得min-这个定义，但实际上它把正常的width和height当作有min的情况来使。这样问题就大了，如果只用宽度和高度，正常的浏览器里这两个值就不会变，如果只用min-width和min-height的话，IE下面根本等于没有设置宽度和高度。

比如要设置背景图片，这个宽度是比较重要的。要解决这个问题，可以这样：

#box{ width: 80px; height: 35px;}html>body #box{ width: auto; height: auto; min-width: 80px; min-height: 35px;}

12. 页面的最小宽度

如上一个问题，IE不识别min，要实现最小宽度，可用下面的方法：

#container{ min-width: 600px; width:expression(document.body.clientWidth＜ 600? "600px": "auto" );}

第一个min-width是正常的；但第2行的width使用了Javascript，这只有IE才认得，这也会让你的HTML文档不太正规。它实际上通过Javascript的判断来实现最小宽度。

13. DIV浮动IE文本产生3象素的bug

左边对象浮动，右边采用外补丁的左边距来定位，右边对象内的文本会离左边有3px的间距.

#box{ float:left; width:800px;}

#left{ float:left; width:50%;}

#right{ width:50%;}

\*html #left{ margin-right:-3px; //这句是关键}

<div id="box">

<div id="left">＜/div>

<div id="right">＜/div>

</div>

14.IE捉迷藏的问题

当div应用复杂的时候每个栏中又有一些链接，DIV等这个时候容易发生捉迷藏的问题。

有些内容显示不出来，当鼠标选择这个区域是发现内容确实在页面。

解决办法：对#layout使用line-height属性或者给#layout使用固定高和宽。页面结构尽量简单。

15.float的div闭合;清除浮动;自适应高度

① 例如：＜div id=”floatA”>＜div id=”floatB”>＜div id=”NOTfloatC”>

这里的NOTfloatC并不希望继续平移，而是希望往下排。(其中floatA、floatB的属性已经设置为float:left;)

这 段代码在IE中毫无问题，问题出在FF。原因是NOTfloatC并非float标签，必须将float标签闭合。在＜div class=”floatB”>＜div class=”NOTfloatC”>之间加上＜div class=”clear”>这个div一定要注意位置，而且必须与两个具有float属性的div同级，之间不能存在嵌套关系，否则会产生异常。 并且将clear这种样式定义为为如下即可：.clear{clear:both;}

②作为外部 wrapper 的 div 不要定死高度,为了让高度能自适应，要在wrapper里面加上overflow:hidden; 当包含float的box的时候，高度自适应在IE下无效，这时候应该触发IE的layout私有属性(万恶的IE啊！)用zoom:1;可以做到，这样 就达到了兼容。

例如某一个wrapper如下定义：

.colwrapper{overflow:hidden; zoom:1; margin:5px auto;}

③对于排版,我们用得最多的css描述可能就是float:left.有的时候我们需要在n栏的float div后面做一个统一的背景,譬如:

<div id=”page”>

<div id=”left”>＜/div>

<div id=”center”>＜/div>

<div id=”right”>＜/div>

</div>

比 如我们要将page的背景设置成蓝色,以达到所有三栏的背景颜色是蓝色的目的,但是我们会发现随着left center right的向下拉长,而page居然保存高度不变,问题来了,原因在于page不是float属性,而我们的page由于要居中,不能设置成 float,所以我们应该这样解决：

<div id=”page”>

<div id=”bg” style=”float:left;width:100%”>

<div id=”left”>＜/div>

<div id=”center”>＜/div>

<div id=”right”>＜/div>

</div>

</div>

再嵌入一个float left而宽度是100%的DIV解决之。

或者另一种方法：用选择器（：after）在page之后插入一个空标签,并清除浮动

.page:after {  content: ""; display: table; clear: both; }

④万能float 闭合(非常重要!)

关于 clear float 的原理可参见 [How To Clear Floats Without Structural Markup],将以下代码加入Global CSS 中,给需要闭合的div加上class="clearfix" 即可,屡试不爽。

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

.clearfix { display:inline-block; }

.clearfix {display:block;}

或者这样设置：.hackbox{ display:table; //将对象作为块元素级的表格显示}

16. 高度不适应

高度不适应是当内层对象的高度发生变化时外层高度不能自动进行调节，特别是当内层对象使用margin 或padding时。

例：

#box {background-color:#eee; }

#box p {margin-top: 20px;margin-bottom: 20px; text-align:center; }

<div id="box">

<p>p对象中的内容＜/p>

</div>

解决技巧：在P对象上下各加2个空的div对象CSS代码{height:0px;overflow:hidden;}或者为DIV加上border属性。

17. IE6下图片下有空隙产生

解决这个BUG的技巧有很多,可以是改变html的排版,或者设置img为display:block或者设置vertical-align属性为vertical-align:top/bottom/middle/text-bottom 都可以解决.

18. 对齐文本与文本输入框

加上vertical-align:middle;

<style type="text/css">

<!--

input {

width:200px;

height:30px;

border:1px solid red;

vertical-align:middle;

}

-->

</style>

经验证，在IE下任一版本都不适用，而ff、opera、safari、chrome均OK！

19. LI中内容超过长度后以省略号显示

此技巧适用与IE、Opera、safari、chrom浏览器，FF暂不支持。

<style type="text/css">

<!--

li {

width:200px;

white-space:nowrap;

text-overflow:ellipsis;

-o-text-overflow:ellipsis;

overflow: hidden;

}

-->

</style>

20. 为什么web标准中IE无法设置滚动条颜色了

解决办法是将body换成html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />

<style type="text/css">

<!--

html {

scrollbar-face-color:#f6f6f6;

scrollbar-highlight-color:#fff;

scrollbar-shadow-color:#eeeeee;

scrollbar-3dlight-color:#eeeeee;

scrollbar-arrow-color:#000;

scrollbar-track-color:#fff;

scrollbar-darkshadow-color:#fff;

}

-->

＜/style>

21. 为什么无法定义1px左右高度的容器

IE6下这个问题是因为默认的行高造成的,解决的技巧也有很多：

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

22.怎么样才能让层显示在FLASH之上呢

解决的办法是给FLASH设置透明

<param name="wmode" value="transparent" />

23. 链接(a标签)的边框与背景

a链接加边框和背景色，需设置 display: block, 同时设置 float: left 保证不换行。参照menubar, 给 a 和menubar设置高度是为了避免底边显示错位, 若不设 height, 可以在menubar中插入一个空格。

24. 超链接访问过后hover样式就不出现的问题

被点击访问过的超链接样式不在具有hover和active了,很多人应该都遇到过这个问题,解决技巧是改变CSS属性的排列顺序: L-V-H-A

Code:

<style type="text/css">

<!--

a:link {}

a:visited {}

a:hover {}

a:active {}

-->

</style>

25. FORM标签

这 个标签在IE中,将会自动margin一些边距,而在FF中margin则是0,因此,如果想显示一致,所以最好在css中指定margin和 padding,针对上面两个问题,我的css中一般首先都使用这样的样式ul,form{margin:0;padding:0;}。

26. 属性选择器(这个不能算是兼容,是隐藏css的一个bug)

p[id]{}div[id]{}

这个对于IE6.0和IE6.0以下的版本都隐藏,FF和OPera作用.属性选择器和子选择器还是有区别的,子选择器的范围从形式来说缩小了,属性选择器的范围比较大,如p[id]中,所有p标签中有id的都是同样式的.

27. 为什么FF下文本无法撑开容器的高度

标准浏览器中固定高度值的容器是不会象IE6里那样被撑开的,那我又想固定高度,又想能被撑开需要怎样设置呢？办法就是去掉height设置min-height:200px; 这里为了照顾不认识min-height的IE6 可以这样定义:

{

height:auto!important;

height:200px;

min-height:200px;

}

28. IE和FireFox 对空格的尺寸解释不同，FireFox为4px,IE为8px; FireFox对div与div

之间的空格是忽略的，但是IE是处理的。因此在两个相邻div之间不要有空格跟回车，否则可能造成不同浏览间之间格式不正 确，比如著名的3px偏差（多个img标签连着，然后定义float: left;结果在firefox里面正常，而IE里面显示的每个img都相隔了3px。我把标签之间的空格都删除都没有作用。解决方法是在img外面套 li，并且对li定义margin: 0; 避免方式：在必要的时候不要无视 list 标签）而且原因难以查明。

29. 条件注释

<link rel="stylesheet" type="text/css" href="css.css" />

<!--[if IE 7]>

<link rel="stylesheet" type="text/css" href="ie7.css" />

<![endif]-->

<!--[if lte IE 6]>

<link rel="stylesheet" type="text/css" href="ie.css" />

<![endif]-->

lte -- 小于等于

lt  -- 小于

gte --  大于等于

gt  --  大于

！ --  不等于

30.强制渲染

<meta http-equiv=X-UA-Compatible content=IE=EmulateIE7>    //这句话的意思是强制使用IE7模式来解析网页代码！

<meta http-equiv=“X-UA-Compatible” content=“IE=8″>

<meta http-equiv=“X-UA-Compatible” content=“chrome=1″ />    //Google Chrome Frame也可以让IE用上Chrome的引擎

<meta http-equiv=“X-UA-Compatible” content=“IE=EmulateIE7″><!– IE7 mode –> 或者 <meta http-equiv=“X-UA-Compatible” content=“IE=7″><!– IE7 mode –>       //强制IE8使用IE7模式来解析

<meta http-equiv=“X-UA-Compatible” content=“IE=6″><!– IE6 mode –>   <meta http-equiv=“X-UA-Compatible” content=“IE=5″><!– IE5 mode –>   //强制IE8使用IE6或IE5模式来解析

<meta http-equiv=“X-UA-Compatible” content=“IE=5; IE=8″ />   //一个特定版本的IE支持所要求的兼容性模式多于一种

31.js兼容文件

使IE5,IE6兼容到IE7模式（推荐）

<!–[if lt IE 7]>

<script src=”http://ie7-js.googlecode.com/svn/version/2.0(beta)/IE7.js” type=”text/javascript”></script>

<![endif]–>

使IE5,IE6,IE7兼容到IE8模式

<!–[if lt IE 8]>

<script src=”http://ie7-js.googlecode.com/svn/version/2.0(beta)/IE8.js” type=”text/javascript”></script>

<![endif]–>

使IE5,IE6,IE7,IE8兼容到IE9模式

<!–[if lt IE 9]>

<script src=”http://ie7-js.googlecode.com/svn/version/2.1(beta4)/IE9.js”></script>

<![endif]–>

32. 浏览器识别符

p{ \_color:red; }           IE6 专用

\*html p{ color:#red; }  IE6 专用

p{ +color:red; }           IE6,7 专用

p{ \*color:red; }           IE6,7 专用

\*html p{ color:red; }    IE6,7 专用

p{\*+color: red;}          IE7 专用

Body> p{ color: red; }  屏蔽 IE6

p{ color:red\9; }          IE8

Firefox: -moz-

Safari: -webkit-

Opera: -o-

IE: -ms-

33.div的垂直居中问题

vertical-align:middle; 将行距增加到和整个DIV一样高 line-height:200px; 然后插入文字，就垂直居中了。缺点是要控制内容不要换行 powered by 25175.net

34.margin加倍的问题

设置为float的div在ie下设置的margin会加倍。这是一个ie6都存在的bug。解决方案是在这个div里面加上display:inline;

例如：

<#div id=”imfloat”>

相应的css为

#IamFloat{

float:left;

margin:5px;/\*IE下理解为10px\*/

display:inline;/\*IE下再理解为5px\*/}

35.浮动ie产生的双倍距离

#box{ float:left; width:100px; margin:0 0 0 100px; //这种情况之下IE会产生200px的距离 display:inline; //使浮动忽略}

这里细说一下block与inline两个元素：block元素的特点是,总是在新行上开始,高度,宽度,行高,边距都可以控制(块元素);Inline元素的特点是,和其他元素在同一行上,不可控制(内嵌元素);

#box{ display:block; //可以为内嵌元素模拟为块元素 display:inline; //实现同一行排列的效果 diplay:table;

36.DIV浮动IE文本产生3象素的bug

左边对象浮动，右边采用外补丁的左边距来定位，右边对象内的文本会离左边有3px的间距.

#box{ float:left; width:800px;}

#left{ float:left; width:50%;}

#right{ width:50%;}

\*html #left{ margin-right:-3px; //这句是关键}

<div id="box">

<div id="left"></div>

<div id="right"></div>

</div>

37.IE6下为什么图片下有空隙产生

解 决这个BUG的方法也有很多,可以是改变html的排版,或者设置img 为display:block 或者设置vertical-align 属性为 vertical-align:top | bottom |middle |text-bottom 都可以解决.

38.如何对齐文本与文本输入框

加上 vertical-align:middle;

<style type="text/css">

<!--

input {

width:200px;

height:30px;

border:1px solid red;

vertical-align:middle;

}

-->

</style>

39.为什么web标准中IE无法设置滚动条颜色了

解决办法是将body换成html

40.为什么无法定义1px左右高度的容器

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

41.怎么样才能让层显示在FLASH之上呢

解决的办法是给FLASH设置透明

<param name="wmode" value="transparent" />

42.链接(a标签)的边框与背景

a 链接加边框和背景色，需设置 display: block, 同时设置 float: left 保证不换行。参照 menubar, 给 a 和 menubar 设置高度是为了避免底边显示错位, 若不设 height, 可以在 menubar 中插入一个空格。

43.超链接访问过后hover样式就不出现的问题

被点击访问过的超链接样式不在具有hover和active了,很多人应该都遇到过这个问题,解决方法是改变CSS属性的排列顺序: L-V-H-A

Code:

<style type="text/css">

<!--

a:link {}

a:visited {}

a:hover {}

a:active {}

-->

</style>

44.游标手指cursor

cursor: pointer 可以同时在 IE FF 中显示游标手指状， hand 仅 IE 可以

45.UL的padding与margin

ul标签在FF中默认是有padding值的,而在IE中只有margin默认有值,所以先定义 ul{margin:0;padding:0;}就能解决大部分问题

46.IE与宽度和高度的问题

IE不认得min-这个定义，但实际上它把正常的width和height当作有min的情况来使。这样问题就大了，如果只用宽度和高度，正常的浏览器里这两个值就不会变，如果只用min-width和min-height的话，IE下面根本等于没有设置宽度和高度。比如要设置背景图片，这个宽度是比较重要的。要解决这个问题，可以这样：

#box{ width: 80px; height: 35px;}html>body #box{ width: auto; height: auto; min-width: 80px; min-height: 35px;}

47.页面的最小宽度

min-width是个非常方便的CSS命令，它可以指定元素最小也不能小于某个宽度，这样就能保证排版一直正确。但IE不认得这个，而它实际上把 width当做最小宽度来使。为了让这一命令在IE上也能用，可以把一个<div> 放到 <body> 标签下，然后为div指定一个类：

然后CSS这样设计：

#container{

min-width: 600px;

width:e?xpression(document.body.clientWidth < 600? “600px”: “auto” );

}

第一个min-width是正常的；但第2行的width使用了Javascript，这只有IE才认得，这也会让你的HTML文档不太正规。它实际上通过Javascript的判断来实现最小宽度。

48.DIV浮动IE文本产生3象素的bug

左边对象浮动，右边采用外补丁的左边距来定位，右边对象内的文本会离左边有3px的间距.

#box{

float:left;

width:800px;}

#left{

float:left;

width:50%;}

#right{

width:50%;

}

\*html #left{

margin-right:-3px;

//这句是关键

}

HTML代码

<DIV id=box>

<DIV id=left></DIV>

<DIV id=right></DIV>

</DIV>

49.属性选择器(这个不能算是兼容,是隐藏css的一个bug)

p[id]{}div[id]{}

p[id]{}div[id]{}

这个对于IE6.0和IE6.0以下的版本都隐藏,FF和OPera作用

属性选择器和子选择器还是有区别的,子选择器的范围从形式来说缩小了,属性选择器的范围比较大,如p[id]中,所有p标签中有id的都是同样式的.

50.高度不适应

高度不适应是当内层对象的高度发生变化时外层高度不能自动进行调节，特别是当内层对象使用

margin 或paddign 时。例：

<div id=”box”>

<p>p对象中的内容</p>

</div>

CSS：

#box {background-color:#eee; }

#box p {margin-top: 20px;margin-bottom: 20px; text-align:center; }

解决方法：在P对象上下各加2个空的div对象CSS代码：.1{height:0px;overflow:hidden;}或者为DIV加上border属性。

# Javascript编程题

1.实现以下函数

add(2,5); //7

add(2) (5); //7

function add(a,b){

var sum;

sum = a+b;

return sum;

}

add(2,5);

(function()[

function add(a){

var sum;

sum = a+b;

return sum；

}

])(b)

add(2) (5)

2.写出注释处的a,b,c的值

var fun=function(){

var a=1,b=2;

var fun2=function(){

var b=3,c=4;

//a,b,c? 1,3,4

a+=b+c;

//a,b,c? 8,3,4

}

//a,b,c? 1,2

fun2();

//a,b,c? 8,2

}

fun();

3.请实现以下代码

[1,2,3,4,5].duplicator(); // [1,2,3,4,5,1,2,3,4,5]

var arr=[1,2,3,4,5]

Array.prototype.readd = function() {

return this.concat(this)

};

arr.readd()

4.请实现一个遍历至 100 的 for loop 循环，在能被 2 整除时输出 "two"，在能被 3 整除时输出 "three"，在能同时被 2 和 3 整除时输出 " twothree"。

for(var i=0;i<100;i++){

if(i/2 == 0){

return 'two' ;

}else if(i/3 == 0){

return 'three'

}else if(i/6 == 0){

return 'twothree'

}

}

5.请指出以下person1和person2的区别，以及this的指向：

function Person(){

this.name="jay";

}

var person1 = Person();//this? Window

var person2 = new Person();//this? Person

6.编写函数将URLhttp://163.com?key0=0&key1=1&key2=2的参数值截取出来，输出为json格式数据

var loc="http://163.com?key0=0&key1=1&key2=2"

var locarr = loc.split("&")

var str =""

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

str += locarr[i].replace(/=/g,":").replace(/\?/g,"")+","

}

str = "{"+str+"}"

console.log(str)

6.如何用javascript获取所有选中的checkbox值

原理是：当页面中有多个选择框，每个框的值都不一样，当我选择中其中若干选择框，点击按钮，则弹出这些选择框的值的和。重复是：通点击按钮，然后弹出这这些选中框的值的和。

html页面部分：

<input type="checkbox" value="1" name="check"/>1

<input type="checkbox" value="2" name="check"/>2

<input type="checkbox" value="3" name="check"/>3

<input type="checkbox" value="4" name="check"/>4

<input type="checkbox" value="5" name="check"/>5

<input type="button" value="求和" onclick="Math\_Click()"/>

JS部分：

<script type="text/javascript">

function Math\_Click(){

var rusult=0;

var check\_array=document.getElementsByName("check");

for(var i=0;i<check\_array.length;i++)

{

if(check\_array[i].checked==true)

{

rusult=parseInt(rusult)+parseInt(check\_array[i].value);

}

}

alert(rusult);

}

</script>

原理是把所有的checkbox的name值设为一样，然后查出所有的进行遍历，判断是否是选中状态然后再相加，需注意的是，页面其他标签如果有name值相同的可能会有影响，这个自己把握了

7.js继承有5种实现方式：

1、继承第一种方式：对象冒充

function Parent(username){

this.username = username;

this.hello = function(){

alert(this.username);

}

}

function Child(username,password){

//通过以下3行实现将Parent的属性和方法追加到Child中，从而实现继承

//第一步：this.method是作为一个临时的属性，并且指向Parent所指向的对象，

//第二步：执行this.method方法，即执行Parent所指向的对象函数

//第三步：销毁this.method属性，即此时Child就已经拥有了Parent的所有属性和方法

this.method = Parent;

this.method(username);//最关键的一行

delete this.method;

this.password = password;

this.world = function(){

alert(this.password);

}

}

var parent = new Parent("zhangsan");

var child = new Child("lisi","123456");

parent.hello();

child.hello();

child.world();

2、继承第二种方式：call()方法方式

call方法是Function类中的方法

call方法的第一个参数的值赋值给类(即方法)中出现的this

call方法的第二个参数开始依次赋值给类(即方法)所接受的参数

function test(str){

alert(this.name + " " + str);

}

var object = new Object();

object.name = "zhangsan";

test.call(object,"langsin");//此时，第一个参数值object传递给了test类(即方法)中出现的this，而第二个参数"langsin"则赋值给了test类(即方法)的str

function Parent(username){

this.username = username;

this.hello = function(){

alert(this.username);

}

}

function Child(username,password){

Parent.call(this,username);

this.password = password;

this.world = function(){

alert(this.password);

}

}

var parent = new Parent("zhangsan");

var child = new Child("lisi","123456");

parent.hello();

child.hello();

child.world();

3、继承的第三种方式：apply()方法方式

apply方法接受2个参数，

A、第一个参数与call方法的第一个参数一样，即赋值给类(即方法)中出现的this

B、第二个参数为数组类型，这个数组中的每个元素依次赋值给类(即方法)所接受的参数

function Parent(username){

this.username = username;

this.hello = function(){

alert(this.username);

}

}

function Child(username,password){

Parent.apply(this,new Array(username));

this.password = password;

this.world = function(){

alert(this.password);

}

}

var parent = new Parent("zhangsan");

var child = new Child("lisi","123456");

parent.hello();

child.hello();

child.world();

4、继承的第四种方式：原型链方式，即子类通过prototype将所有在父类中通过prototype追加的属性和方法都追加到Child，从而实现了继承

function Person(){

}

Person.prototype.hello = "hello";

Person.prototype.sayHello = function(){

alert(this.hello);

}

function Child(){

}

Child.prototype = new Person();//这行的作用是：将Parent中将所有通过prototype追加的属性和方法都追加到Child，从而实现了继承

Child.prototype.world = "world";

Child.prototype.sayWorld = function(){

alert(this.world);

}

var c = new Child();

c.sayHello();

c.sayWorld();

5、继承的第五种方式：混合方式

混合了call方式、原型链方式

function Parent(hello){

this.hello = hello;

}

Parent.prototype.sayHello = function(){

alert(this.hello);

}

function Child(hello,world){

Parent.call(this,hello);//将父类的属性继承过来

this.world = world;//新增一些属性

}

Child.prototype = new Parent();//将父类的方法继承过来

Child.prototype.sayWorld = function(){//新增一些方法

alert(this.world);

}

var c = new Child("zhangsan","lisi");

c.sayHello();

c.sayWorld();