基础部分

1. 浏览器工作原理（事件队列循环）

答：意思也就是说当我们设置一个延迟函数的时候，当前脚本并不会阻塞，它只是会在浏览器的**事件表**中进行记录，程序会继续向下执行。当延迟的时间结束之后，事件表会将回调函数添加至**事件队列（task queue）**中，事件队列拿到了任务过后便将任务压入**执行栈（stack）**当中，执行栈执行任务，事件队列是一个存储着待执行任务的队列，其中的任务严格按照时间先后顺序执行，排在队头的任务将会率先执行，而排在队尾的任务会最后执行。事件队列每次仅执行一个任务，在该任务执行完毕之后，再执行下一个任务。执行栈则是一个类似于函数调用栈的运行容器，当执行栈为空时，JS 引擎便检查事件队列，如果不为空的话，事件队列便将第一个任务压入执行栈中运行。

事件循环：每次执行代码时，都会去事件队列里看有没有要执行的代码。

1. Es6语法中的数组去重？

答：先用set方法去重，由于这个方法返回的是一个对象，然后再用Array方法将其转化为数组。Var newArr=Array.from(new set(arr));console.log(newArr).

1. 水平居中（字，盒子，越多越好）？

答：盒子水平居中：1.有宽度就margin：0 auto；，

2.position：absolute；left：50%，transform :-50%

3.利用css3的新增属性table-cell, vertical-align:middle;

4.直接再父元素上使用flexbox伸缩布局。

5.再父元素上设置text-align:center,然后再利用inline-block；

文字水平居中：text-align:center;line-height:等于高；

1. Js隐士转换？

答：1.利用运算符方法进新房转换；

1. 继承相关问题？
2. 介绍meta标签有什么用，主要设置哪些东西？

答：<meta> **标签提供关于 HTML 文档的元数据。它不会显示在页面上，但是对于机器是可读的。可用于浏览器（如何显示内容或重新加载页面），搜索引擎（关键词），或其他 web 服务。**meta里的数据是供机器解读的，告诉机器该如何解析这个页面，还有一个用途是可以添加服务器发送到浏览器的http头部内容可以设置移动端的适配问题等；

1. Css的calc方法？

答：calc()能让你给元素的做计算，你可以给一个div元素，使用百分比、em、px和rem单位值计算出其宽度或者高度，比如说“width:calc(50% + 2em)”，这样一来你就不用考虑元素DIV的宽度值到底是多少，而把这个烦人的任务交由浏览器去计算。可以设置响应式布局。

1. 怎么判断两个对象是否相等？
2. 发送方Ajax的时候，怎么判断是使用异步还是同步？

答; 当把asyn设为false时，这时ajax的请求是同步的，也就是说，这个时候ajax块发出请求后，他会等待在function1（）这个地方，不会去执行function2()，知道function1()部分执行完毕。当设置成true时，代表的是异步。

1. 数据请求失败，页面怎么显示？

答:直接在页面上显示loading font404；

1. Nodejs反项代理？

答：首先有一个liux服务器。然后xshell的工具连接服务器下载nginx，然后使用命令找到配置文件，修改头部允许跨域，这样就可以反项代理了。

1. 搜索框输入关键字，得到搜索结果，需要给搜索框绑定什么建通事件？
2. 一个不规则图形，上面的文字也根据图形排列，怎么实现？
3. 用jQuery深拷贝和浅拷贝？
4. 怎么优化页面？就是性能优化的问题
5. 影响页面加载速度的因素有哪些？

答：1.**页面的HTML代 码量。**

2. **对图片的处理。**

3 **CSS/JS的 代码量和引入方式。**

4. **HTML制作采用的技术。**

5. **页面是动态还是静态**

1. 页面中有很多图片该怎么优化才能让页面的加载速度最快？

答：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。

1. 如何引入外部文件能使页面的加载速度达到最优的程度？
2. 你觉得vue开发和传统开发有什么不同？

答：前端框架，这种框架的开发可以把前后台完全分离，前端人员只负责前端开发，服务端人员负责服务端开发。vue有自己的请求处理方式，现在前后端都是用rest风格，前后台只在接口上有交集，代码上完全独立。还有就是Vue同angualrjs类似都是用我们所谓的MVC或者称为MVVC的方式，是的前端代码也区分出model，controller and view层，代码做到更好的控制和流畅

1. Nodejs和js的不同？

答：NodeJS是以ECMASCRIPT为基础，然后在这个基础上扩展出来的一些操作其他东西（即操作非浏览器）的一些方法，比如操作OS操作系统、file文件系统、net网络、database数据库等。也就是说nodejs和JS在底层都是一样的，都是ecmascript，但是扩展出来的功能却是不一样的。一个是偏向于前端浏览器，一个是偏向于后端。

在ecmascript部分node和JS其实是一样的，比如与数据类型的定义、语法结构，内置对象。但在js中的顶层对象是window对象，但是在node中没有什么window对象，node中的顶层对象是global对象。这就是二者的差异性。

1. 为什么出现跨域的问题？

答：1.web中浏览器设置的相关策略，

2. 应用程序本身设置的禁止跨域限制

3.同源策略规定，Ajax请求只能请求同源的网址。

1. 怎么实现一个元素中的内容滚动？

答：

1. Css3动画怎么实现？
2. 自己做一个弹层怎么去实现？
3. 如果弹层的高度超出了页面的高度怎么解决？
4. 怎么使用地图接口的？

答：1. // 创建Map实例，

2. 初始化地图,设置中心点坐标和地图级别

3. 开启鼠标滚轮缩放

4. 设置地图模板

5.成功后的回调，设置经纬度。

1. 怎么遍历一个数组？
2. 介绍一下你的项目并说说你主要做的哪块功能以及实现它的逻辑
3. 用户登录注册中通过session，cookie存储，怎么样保证它的安全性
4. 当你从服务器获取数据时怎么来保证你书写代码的健壮性
5. 你会做文件的上传和下载吗
6. 给你一个需求，风险评估的界面，比如有几种风险，每一个风险里有风险的状态，

还有风险的内容，我要实时的知道每个风险的是什么样一个状态。你觉得你需要多长的时间完成，还有哪一块你觉得时比较难做的，说明他的原因

33原型链

34vue传参问题

35生命周期

36浏览器缓存

37map

38孟老师讲过的promise的问题解决的异步问浏览器兼容问题

39字符串和数组常用的方法

40如何替换字符串中的某个字符

41对象中新增一个属性 深拷贝浅拷贝

42自己介绍一些vue的东西

43ajax的参数 如何判断ajax请求超时处理 ajax请求出问题怎么判断的；

44ajax请求后怎么清楚浏览器缓存

45.怎么理解前后端分离？

46vue的原理是什么，它是如何实现dom渲染的？

47.

1.什么是HTML？

答：HTML并不是真正的的程序语言，他是一种 标 记 语 言 ，用来结构化和语义化你想要放在web 网站上的那些内容。它由一系列的元素（elements）所组成，这些元素可以用来封装你的内容中担任不同工作的各部分和各个角色。

2．什么是CSS？

答： 就像 HTML，CSS 也不是真正的编程语言。它是样式表语言，也就是说，它允许你有选择性的为 HTML 文档的元素添加样式。

3.行内元素和块级元素的具体区别是什么？

答：块级元素特点：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、高度、内边距和外边距都可以控制。

行内元素特点：和相邻的内连元素在同一行，宽度、高度、内边距和外边距的top/bottom都不可改变（也就是padding和margin的left和right是可以设置的），就是里面文字或者图片的大小。

4.简述一下你对HTML语意化的理解？

答：a: HTML语义化让页面的内容结构化，结构更清晰，便于对浏览器、搜素引擎解析；

b:即使在没有样式CSS的情况下也能以一种文档格式显示，并且是容易阅读的；

c：搜素引擎的爬虫也依赖于HTML标记来确定上下文和个个关键字的权重，有利于SEO；

d:使阅读源代码的人更容易将网站分块，便于阅读、维护和理解；

5.rgba和opacity设置透明度的区别是什么？

答：两者都能实现透明效果，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颜色，也就是说opacity有继承性而rgba没有。

6.DOCTYPE的作用？

答：a:<!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。

b:严格模式的排版和JS运作模式是以浏览器支持的最高标准运行。

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

D:DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。

7.介绍一下你对浏览器内核的理解？都有哪些常见的浏览器内核？

答：负责对网页语法的理解（如标准通用标记语言下的一个应用HTML.JAVASCRIPT）并渲染（显示）网页，所以，通常所谓的浏览器内核也就是浏览器所采用的渲染引擎，渲染引擎决定了浏览器如何显示网页的内容以及网页的格式信息，不同的浏览器内核对网页辨析的语法的解释也有不同，因此同一网页在不同的内核的浏览器里的渲染效果也可能不同，这也是网页编写者需要在不同内核的浏览器中测试网页显示效果的原因。

常见浏览器内核：Trident内核：IE，MAXThon,TT,360,搜狗浏览器等。

Gecko内核：Netscape6及以上版本，FF等

Presto内核：opera7及以上（原来内核是这个，但现在更换成Blink）

Webkit内核：Safari，Chrome等

blink内核：chrome。

8.css选择器权重如何计算？

答：页面显示样式的优先级取决于其特殊性，特殊性越高，就显示最高的，当特殊新性相等时，显示后者。

！importent：无限大

内联式特殊性权重为1000

Id选择器为100

属性选择器，class或伪类为10

对于标签选择器或伪元素为1

通配符为0

9.对WEB标准以及W3C的理解与认识？

答：1>：标签闭合、标签小写、不乱嵌套、提高搜索机器人搜素几率、使用外链css和js脚本、结构行为表现的分离。

2>.文件下载与页面速度更快.内容能被更多的用户所访问.内容能被更广泛的设备所访问,更少的代码和组件.

3>容易维护,改版方便,不需要变动页面内容,提供打印版本而不需要复制内容,提高网站易用性.

10. CSS中优雅降级和渐进增强有什么区别？

答: 优雅降级和渐进增强印象中是随着css3 流出来的一个概念。由于低级浏览器不支持css3，但 css3 的效果又太优秀不忍放弃，所以在高级浏览中使用 css3 而低级浏览器只保证最基本的功能。咋一看两个概念差不多，都是在关注不同浏览器下的不同体验，关键的区别是他们所侧重的内容，以及这种不同造成的工作流程的差异。

> “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。

> “渐进增强”观点则认为应关注于内容本身。

11.  对BFC规范的理解有哪些？

答: BFC(Block formatting context)直译为"块级格式化上下文"。它是一个独立的渲染区域，只有 Block-level box 参与，它规定了内部的Block-level Box 如何布局，并且与这个区域外部毫不相干。

布局规则:

A. 内部的Box 会在垂直方向，一个接一个地放置。

B. Box垂直方向的距离由 margin决定。属于同一个 BFC的两个相邻 Box的 margin 会发生重叠。

C. 每个元素的 margin box 的左边，与包含块border box 的左边相接触(对于从左往右的格式化，否则相反)。即使存在浮动也是如此。

D.BFC 的区域不会与 float box 重叠。

E.BFC 就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素。反之也如此。

F.计算 BFC 的高度时，浮动元素也参与计算。

哪些元素会产生BFC:

A. 根元素

B. float 属性不为 none

C. position为 absolute 或 fixed

D. display为inline-block， table-cell， table-caption， flex， inline-flex

F. overflow 不为 visible

12有多少种清除浮动的方法?

答:a:父级div定义高

原理：父级 div手动定义 height，就解决了父级 div无法自动获取到高度的问题。 简单、代码少、容易掌握 ，但只适合高度固定的布局.

B:结尾处加空div标签clear:both

原理：在浮动元素的后面添加一个空div兄弟元素，利用 css 提高的clear：both清除浮动，让父级div能自动获取到高度 ，如果页面浮动布局多，就要增加很多空 div，让人感觉很不好.

C: 父级div定义 伪类：after 和 zoom

 /\*清除浮动代码\*/

> .clearfix：after{

> content：""；

> display：block；

> visibility：hidden；

> height：0；

> line-height：0；

> clear：both；

> }

> .clearfix{zoom：1}

原理：IE8以上和非 IE浏览器才支持：after，原理和方法 2有点类似，zoom(IE专有属性)可解

D:父级div定义overflow:hieen:超出盒子部分会被隐藏,不推荐使用.

E:双伪元素:  .clearfix：before，.clearfix：after {

> content： ""；

> display： block；

> clear：both；

> }

> .clearfix {

> zoom： 1；}

13.css外观属性

答:color:文本颜色(预定义的颜色值,十六进制,rgb代码)

Text-align:文本水平对齐方式,还可以让行内元素居中对齐

Line-height:行间距，可以让行内元素水平居中。

Text-indent:首行缩进,建议使用em作为设置单位

Text-decoration:文本装饰(none underline下划线 overline上条线)

14.css字体样式

答:Font-family:设置字体,中文字体用双引号,尽量使用系统默认字体

Font-weight:字体粗细(normal默认不加粗,bold加粗)

Font-style:字体风格(默认normal,italic斜体)

Font-size:字体大小

连写font:font-style font-weight font-size font-family,不能更换顺序.

15.css复合选择器

答:a:后代选择器

B:子元素选择器

C:交集选择器

D:并集选择器

E:连接伪类选择器

16.嵌套元素垂直外边距的合并(塌陷)

答:产生原因:a如果父元素没有上内边距及边框

B:父元素的上外边距会与子元素的上外边距发生合并

C:合并后的外边距为两者中的最大者.

解决方案:a可以为父元素定义上边框

B:可以为父元素定义上内边距

C:可以为父元素添加overflow:hidden

17.浮动的特点?

答:a:加了浮动的盒子是不占位置的,原来的位置让给了标准流的盒子

B:浮动元素会改变display属性,类似转化了行内快,但是元素之间没有空白缝隙

C:子盒子的浮动参照父盒子对齐,不会与父盒子的边框重叠,也不会超出父盒子的内边距

D:在一个父级盒子中,如果前一个盒子浮动的,那么当前盒子会与之前的盒子的顶部对齐,如果之前的盒子是标准流的,那么当前的盒子会显示在之前盒子的下方.

注意:两盒浮动为了不影响下面的元素,需要添加一个父盒子,且父盒子需要设置高.

18.常见的图片格式?

答:jpg图片格式:对色彩的信息保留较好

Gif图片格式:最多只能存贮256色,所以通常用来显示简单图形及字体,但是可以保存透明背景和动画效果.

Png图片格式:存储形式丰富,能够保持透明背景

Psd图像格式:是Photoshop的专用格式,可以放图层.通道.遮盖等多种设计.

Webp图像格式，是chrome新有的。

19.定位?

答:Fixed:固定定位,完全托标,不占位置,只认浏览器的可视窗口.跟父元素没有任何关系,单独使用,不随滚动条滚动.

Relative:相对定位,相对于自己原来在标准流中的位置来移动的,原来在标准流的区域继续占有,后面的盒子任然以标准流变得方式对待.

Absolute:绝对定位,如果父级没有定位,以我们浏览器为主(根据最近的祖先元素定位),完全托标,不保留原来位置.

Static:静态定位,不托标不占位,跟标准流没有区别.

扩展:绝对定位和相对定位和浮动元素相似,会改变display特性,将元素默认转化位行内元素,所以,定位的盒子需要单写宽度.

20. 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？

答:  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。

21.  精灵图(CSS Sprites)的优点和缺点?

答:定义: 精灵图是一种网页图片应用处理方式。就是把网页中很多小背景图片整合到一张图片文件中，再利用CSS的“background-image”，“background-repeat”，“background-position”的组合进行背景图显示及定位，达到显示某一部分背景图的效果。

优点: 1. 减少图片的体积，因为每个图片都有一个头部信息，把多个图片放到一个图片里，就会共用同一个头部信息，从而减少了字节数。

2. 减少了网页的http请求次数，从而加快了网页加载速度，提高用户体验。

3. 解决了网页设计师在图片命名上的困扰，只需对一张集合的图片上命名就可以了，不需要对每一个小元素进行命名，从而提高了网页的制作效率。

4. 更换风格方便，只需要在一张或少张图片上修改图片的颜色或样式，整个网页的风格就可以改变。维护起来更加方便。

缺点: 1. 在图片合并的时候，你要把多张图片有序的合理的合并成一张图片，还要留好足够的空间，防止板块内出现不必要的背景；这些还好，最痛苦的是在宽屏，高分辨率的屏幕下的自适应页面，你的图片如果不够宽，很容易出现背景断裂；

2. 在开发的时候比较麻烦，你要通过photoshop或其他工具测量计算每一个背景单元的精确位置，这是针线活，没什么难度，但是很繁琐；

3. 在维护的时候比较麻烦，如果页面背景有少许改动，一般就要改这张合并的图片，无需改的地方最好不要动，这样避免改动更多的css，如果在原来的地方放不下，又只能（最好）往下加图片，这样图片的字节就增加了，还要改动css。

4. 精灵图不能随意改变大小和颜色。精灵图改变大小会失真模糊，降低用户体验，css3新属性可以改变精灵图颜色，但是比较麻烦，并且新属性有兼容问题。现在一般都是用web字体(图标字体)来代替精灵图。

22. HTML兼容性问题？

答：1.png24位的图片再ie6浏览器上出现背景，解决方案：做成png8，也可以引用一段脚本处理。

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

3.ie6双边距bug:再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。解决方案：再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）。

4. ie下，可以使用获取常规属性的方法来获取自定义属性，也可以使用getAttribute()获取自定义属性；firefox下，只能使用getAttribute()获取自定义属性。解决方案：统一使用getAttribute()获取属性。

23.css兼容性问题？

答：1.不同浏览器水平居中问题：ie6-7文本居中，嵌套的块元素也会居中，ff/opera/safari/ie8文本会居中，嵌套块不会居中；

解决方案：块元素： maegin-left:auto;margin-right:auto或margin:0 auto;或外层嵌套div设置<div align=”center”></div>

浮动块元素的居中：<div style=”margin:0px auto;”.<div style=”float:left”></div></div>

2.不同浏览垂直居中问题：再浏览器中想要垂直居中，设置vertical-align:middle;不起作用；

解决方案：给容器设置一个与其高度相同的行高line-height:与容器的height一样。

3.ie6-7图片下面有空隙的问题：块元素中含有图片，ie6-7中会出现图片下有空隙。

解决：1.再源代码中让</div>和<img>再同一行。

2.将图片转化为块级对象display:block;

3.设置图片的垂直对齐方式 vartical-align:top/middle/bottom

4．改变父对象的属性，如果父对象的宽、高固定，图片大小随父对象而定，那么可以对父元素设置：overflow:hidden;

5.设置图片的浮动属性：float:left。

4.ie6子元素绝对定位的问题：父级元素使用padding后，子元素使用绝对定位，不能精确定位；

解决：再子元素中设置：\_left:-20px;\_top:-1px;

5.ie6两个层之间3px的问题：左边层采用浮动，右边没有采用浮动，这时再ie6 中两层之间就会产生3像素的间距；

解决：1.右边层也采用浮动 2、左边层添加margin-right:-3px;

24.如何对网站的文件和资源进行优化？

答：html优化：多使用语义化标签，尽量使用h5的新特性，但需要注意兼容性问题；

CSS优化：对于重复使用的样式，最好统一写完后直接写类名即可；

Js优化：兼容性处理、逻辑的整理使代码不要太臃肿、方法的封装方便之后多次调用，尽量使用es6语法；

可以使用cdn网络文件，不需要下载下来，而是使用网上的路径即可。

25.为什么利用多个域名来存储网站资源会更有效？

答：并发，这个意思是在很多人同时上网的时候，因为网络服务器被同时请求，会导流量下载变慢、服务器卡顿或崩溃。那么这个时候，一般将图片放在一个服务器中，其他的资源放在另一个服务器中，这样可以适量的减少同一个服务器的压力。

26.固定宽高、不固定顶宽高的垂直水平居中？

答：1.利用定位的方式：position：absolute；left：50%；transform：translate（-50%）；

2.比较适用模态框，背景遮罩可以始终覆盖全屏，里面的居中方式也可以用以上的方式。

3.css3中：justify-content:center=>子元素水平居中，

Align-items:center=>子元素垂直居中

Display：webkit-flex.

27.表格自动换行怎么实现？

答：word-break:normal使用浏览器默认的换行规则。

Break-all:允许单词内换行

Keep-all：只能在半角空格或连字符处换行。

Word-wrap:normal使用浏览器默认的换行规则。

Break-word：在长单词或url地址内部进行换行。

28.初始化CSS？

答：CSS初始化是指重设浏览器的样式，不同的浏览器默认的样式可能不尽相同，所以开发时的第一件事就是尽可能使他们统一。

腾讯官网：body,ol,ul,h1,h2,h3,h4,h5,h6,p,th,td,dl,dd,form,fieldset,legend,input,textarea,select{margin:0;padding:0}   
    body{font:12px"宋体","Arial Narrow",HELVETICA;background:#fff;-webkit-text-size-adjust:100%;}   
    a{color:#2d374b;text-decoration:none}   
    a:hover{color:#cd0200;text-decoration:underline}   
    em{font-style:normal}   
    li{list-style:none}   
    img{border:0;vertical-align:middle}   
    table{border-collapse:collapse;border-spacing:0}   
    p{word-wrap:break-word}

29.说说z-index的工作原理及使用范围？

答：原理：z-index这个属性控制着元素在z轴上的变现形式，该属性仅适用于定位元素，即拥有relative、absolute、fixed属性的position元素。堆叠顺序，是当前元素位于z轴上的值。数值越大表明元素的堆叠顺序越高，越靠近屏幕。未定义时，后来居上。如果未指定z-index的属性，元素的堆叠顺序基于他所在的文档树。默认情况下，文档中后面声明的元素具有更高的堆叠顺序。当父元素的堆叠顺序备设置的时候，这也意味着，它的子元素的堆叠顺序不能高于或低于这一顺序（相对于父元素的堆叠上下文）。

适用范围：网页两侧浮动窗口（播放器、置顶按钮、浮动广告、功能按钮等）

导航栏浮动置顶

隐藏div实现弹窗功能（通过设置div的定位和z-index控制div的位置和出现隐藏）。

30.display和visibility的区别？

答：display通常可以设置为None、inline、block;

Visibility通常设置为hidden、visible

当display为none，visibility为hidden时，元素都会不见。不过其还有不同之处。Display会将元素隐藏掉，并且位置不再被占据，而visibility则是占据原来的位置。

31.CSS中几种设置隐藏的方法？

答：1.display：none；

2.visible：hidden；

3.opcity：0;

4.position:absolute，就是把top值和left值写的绝对大的负值。

5．Clip-path；

6.z-index：绝对大值。

32.怎么给加载不出来的图片添加一个默认的图片，点击刷新该图片？

答：可以给img的3个时间属性做一下控制；

1. onError：当图片加载出现错误，会触发经常再这里事件里头写入将图片导向默认报错图片，以免页面上出现红色的叉叉；
2. onload：事件是当图片加载完成之后触发。
3. onAbort：图片加载的时候，用户通过点击停止加载（浏览器上的红色叉叉）时触发，通常这里触发一个提示：“图片正在加载”；
4. 在onload和onerror事件中添加js代码比如在onload事件中检查settime事件30秒，在显示一个远程图片，onerror方法检查图片没有的话就在js中处理显示默认图片。

33.块级元素和行内元素？

答：块级元素：div, p, form, ul, li, ol, dl, address, hr, menu, table;

行内元素：span, strong, em, br, img, input, lable, select, textarea, cite.

**JS和jQuery部分**

1.DOM对象？

答：DOM实际上是以面向对象方式描述的文档模型。DOM定义了表示和修改文档所需的对象、这些对象的行为和属性以及这些对象之间的关系。可以把DOM认为是页面上数据和结构的一个树形表示，不过页面当然可能并不是以这种树的方式具体实现。

通过 JavaScript，您可以重构整个 HTML 文档。您可以添加、移除、改变或重排页面上的项目。

要改变页面的某个东西，JavaScript 就需要获得对 HTML 文档中所有元素进行访问的入口。这个入口，连同对 HTML 元素进行添加、移动、改变或移除的方法和属性，都是通过文档对象模型来获得的（DOM）。

DOM 可被 JavaScript 用来读取、改变 HTML、XHTML 以及 XML 文档。

DOM的优势主要表现在：易用性强，使用DOM时，将把所有的XML文档信息都存于内存中，并且遍历简单，支持XPath，增强了易用性。

DOM的缺点主要表现在：效率低，解析速度慢，内存占用量过高，对于大文件来说几乎不可能使用。另外效率低还表现在大量的消耗时间，因为使用DOM进行解析时，将为文档的每个element、attribute、processing-instruction和comment都创建一个对象，这样在DOM机制中所运用的大量对象的创建和销毁无疑会影响其效率。

2.BOM对象？

答：BOM 主要处理浏览器窗口和框架，不过通常浏览器特定的JavaScript 扩展都被看做 BOM 的一部分。这些扩展包括：

1.弹出新的浏览器窗口

2.移动、关闭浏览器窗口以及调整窗口大小

3.提供 Web 浏览器详细信息的定位对象

4.提供用户屏幕分辨率详细信息的屏幕对象

5.对 cookie 的支持

6.IE 扩展了 BOM，加入了 ActiveXObject 类，可以通过 JavaScript 实例化 ActiveX 对象

1.js再html中的几中使用方式？

答：1外部引用远程js文件<script src=””></script>。

2直接写在页面上<script>document.write()</script>

3在js代码中引用外部js文件<script>window.onload=function(){script.src=’’}</script>

2.变量的声明方式？

答：Var a=100; let a=100; const a=100

let只能在定义它的范围内使用，不能在范围外使用；

const表示定义一个定量，只能在初始化时赋值一次，不能再次被赋值。一般定量用大写字母表示

Var可以在预解析的时候进行变量提升

3.交换两个变量的值，不使用第三方变量？

答：Var a=10; var b=5;a =a+b; b=a-b;a=a-b 分别给a和b重新赋值

4.js的基本数据类型？

答：五种：1数据型nubmer

2字符串string

3布尔Bloor

4空型null

5未定义型undefind

5.js的数据类型转换？

答： 隐士转换：123+1=》124，‘123’+1=》124；！‘123’=》false

显示转换：var a =123;

a.toString=>’123’

6. 一元运算符++、

答：A++ 先运用后计算

++A 先计算后运用

7. 逻辑运算符的优先级（

答：！>&& >||

&&:找错，若内容全是对的时候返回最后一个

||:找对的，只要有对直接输出，如果内容全是错的返回最后一个

8.关系运算符==和===的区别？

答：==是双等，是指值相等。

===是三等，是指不仅指值相等以外，还要求类型也要相等。

9. 三元运算符 和If () {} else {}的语法结构

答：条件表达式？结果1：结果2；

If（条件）{

满足输出}else{

否则输出}

10. switch 和 while 和 do...while

答：switch多分支判断语句switch(n){case1:执行代码1 break;……default：剩余的代码}

while（条件表达式）{语句}当条件为真时重复循环，否则退出循环

do….while是do{语句}while(条件表达式)先执行后判断，至少执行一次语句

11．break、continue、return的区别

答：break语句使得循环从for或者while中跳出终止循环，continue使得跳过循环内剩余的语句而进入下一次循环。

return：函数中使用，返回一个值能够在函数外面使用，并且不执行后面的代码，最常用的是返回值，本意是把一个值返回到函数以外。

12.return和return false的区别？

答： 1.return返回null，起到中断方法执行的效果，只要不return false事件处理函数将会继续执行，表单将提交

2.return false，事件处理函数会取消事件，不再继续向下执行。比如表单将终止提交。

12.数组的用法？

答：1>通过构造函数：Var arr=new Array();

2>通过字面量：Var arr=[];

3>赋值：arr[索引号]=1；

数组的索引是从0开始，可以访问某个特定的元素数组名[索引号]；

数组常用方法：push（）向数组尾部添加

Pop()向数组尾部删除一个

Unshift()向数组的头部添加元素

Shift()向头部删除

Reverse（）翻转数组，不会修改原数组，翻转之后返回一个新数组

Sort（）数组排序，1>必须写两形参2>看返回值return，当返回值为负数时，那么前面的数放在前面；当返回值为正数时，那么后面的数在前。3>为0，不动。

直接在原数组上进行排序（默认情况下是对字符串编码，才从小到大排序）var arr=[25,10,18,100];arr.sort(function(a,b){return a-b})console.log(arr)

Splice（a,b,c）删除或替换当前数组的某项，第一个参数是开始索引，第二个参数是截止索引，不包括本身，第三个参数是替换文本（删除的内容），a可为负数，-1则代表从倒数第一位开始。

Slice(a,b)从当前数组中截取一个新的数组，不影响原来的数组，参数start从0开始，end从1开始，1>填两个参数，slice（从该位截取，截取到该位）；2>如果填一个参数，从第几位开始截取，一直截取到最后；3>不写参数就是整个截取数组。

Concat()把参数拼接到当前数组，不改变原数组。

toString（）是把数组当作字符串展示出来。

Join()括号里面需要用字符串形式（标准语法规定），就会用加的东西链接起数组，将数组中所有元素组成一个字符串。

Search（）查找指定字符，可以直接用于正则表达式

改变原数组（在原数组基础上改变）：reverse,sort,push,pop,unshift,shift,splice.

不改变原数组：slice,concat,join,split,toString

13.函数、函数的形参，函数的作用域、函数的return？

答：函数的声明：1.function（）{}

2.var fn =function(){}

3.function fun(形参){}fun(实参)；

函数的形参：参数传递：本质赋值；

形参中多个变量用逗号隔开，当形参大于实参时，后面取得形参会被赋值undefined。实参把具体的值传递给形参，只匹配形参的个数，多余忽略。

函数的return：函数执行过程中只要遇到return，后面的代码就不会再执行，当函数中不写return时，返回的是undefined，或者return后面什么都不跟会返回undefind。

14. arguments的用法（有形参和没有形参的用法）（js基础04-01）★★★

答：类数组：1.可以利用属性名模拟数组的特性；

2.可以动态的增长length属性；

3.如果强行让类数组调用push方法，则会根据length属性值的位置进行属性的扩充。

15. Math数学对象有哪些常用的？

答：Math对象是一个静态对象

Math.PI 圆周率。

Math.abs() 绝对值。

Math.ceil() 向上取整(整数加 1，小数去掉)。

Math.floor() 向下取整(直接去掉小数)。

Math.round() 四舍五入。

Math.pow(x，y) 求 x的y次方。

Math.sqrt() 求平方根。

16. Number数值对象？

答:>Number.MAX\_VALUE 最大数值

> Number.MIN\_VALUE 最小数值

> Number.NaN 特殊的非数字值

> Number.NEGATIVE\_INFINITY 负无穷大

> Number.POSITIVE\_INFINITY 正无穷大

> Number.toExponential( ) 用指数计数法格式化数字

> Number.toFixed( ) 采用定点计数法格式化数字

> Number.toLocaleString( ) 把数字转换成本地格式的字符串

> Number.toPrecision( ) 格式化数字的有效位

> Number.toString( ) 将—个数字转换成字符串

> Number.valueOf( ) 返回原始数值

17.事件那一大块重点？

答：事件冒泡：结构上嵌套关系的元素，会存在事件冒泡的功能，即同一事件，自子元素冒泡向父元素。（自内向外）与位置无关，与嵌套有关。

事件的三个阶段：捕获阶段=>从外向内；

当前目标阶段=>执行当前点击的元素

冒泡阶段=>从内向外

浏览器先去执行捕获状态，只是不显示，捕获阶段判断不了。

触发顺序：先捕获，后冒泡。同一个对象的一个事件处理类型，上面绑定了两个事件处理，分别执行事件冒泡和事件执行。

取消冒泡：W3C标准event.stopProagation()，但不支持IE9以下版本。

event.preventDefault()W3C标注，IE9以下不兼容

取消默认事件：return false，兼容性非常好，以对象属性的方式注册的事件才生效，需要注意的是在js中式阻止默认行为，但在JQUERY中 是取消冒泡。

IE独有event.canceLBubble=true;[实际上谷歌也实现了]

Event.returnValue=false兼荣IE；

Foucus,blur,change,submit,reset,select等事件不冒泡。

事件对象:非IE浏览器会把事件对象（记载了数据发生时的状态和信息）打包传到参数里面去，IE浏览器在window.event里面存储事件对象。

Event ||window.event用于IE，window.event用于IE，event只能用于非IE浏览器。

事件源对象：（找事件源对象的方法）event.target火狐独有的，event.srcElement IE独有的，但是这两个谷歌都有。

事件委托：利用事件冒泡，和事件源对象进行处理。

优点：1性能 不需要循环所有的元素一个个绑定事件

2灵活 当有新的子元素时不需要重新绑定事件

18.全局作用域、局部作用域、块级作用域？

答：作用域：变量或者函数可以起作用的访问。

全局作用域：在script或者一个独立的js文件中，在全局作用域中定义的变量，全局变量，在任何位置都可以访问。

局部作用域：任何一个函数的内部都有一个局部作用域，在局部作用域中定义的变量，局部变量只有在定义该变量的函数中可以访问。

生存周期：局部变量会在函数运行以后被删除，全局变量会在页面关闭后被删除。

注意：只要声明函数就会产生局部作用域，局部作用域可以访问全局作用域。

19. 变量、函数声明提升

答：预解析：解析器在执行当前作用域下的代码之前有一个预解析的过程。预解析的时候，会将变量和函数的声明提升到当前作用域的顶部然后才执行。

声明提升：分为变量提升和函数提升。

变量提升：解析器会将变量提升到作用域的最上面，但是变量的赋值不会提升。

函数提升：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。

另外，解析器会先找到var再找function，因此如果变量和函数重名之后找到的函数会把先找到的变量 覆盖（如果代码执行后有赋值语句，该值又会把函数覆盖）

变量作用域：最外层的使全局变量

函数内部的使局部变量

特殊：函数内部不加var的是全局变量。

注意：全局变量无法直接访问局部变量，但局部变量使可以向上级访问它的父级的变量的，先找局部的，如果局部的没有找到则会向上找他的父级变量。

20. 对象的几种创建方法

答：1.对象字面量：var obj={}

2.构造函数 var obj=new.Object()

3.工厂方法创建 function createHero(name,age){var hero=new objecr(); hero.name=name, hero.age=age}return hero; var hero1=createHero(‘周杰伦’,40)

4.自定义方法创建 function hero(name,age){this.name=name;this.age=age} var hero1=new hero(‘周杰伦’,40);

var obj=new Object({name:’周杰伦’})

21. this的指向：

答：1函数中：this指向window

2方法中：this指向的使这个方法所属对象

3构造函数中：指向构造函数创建的对象

4事件处理函数中：指向事件源，谁调用的该事件this指向谁。

5计时器中：this指向的使window

22. 深度克隆的步骤 ？浅克隆

答：1、先把所有的值都遍历一遍（看是引用值和原始值） 用 for ( var prop in obj )，对象和数组都可以使用

2、判断是原始值，还是引用值？用 typeof 判断是不是 object 1）如果是原始值就直接拷贝 2）如果是引用值，判断是数组还是对象

3、判断是数组还是对象？（方法 instanceof【看a 的原型链上有没有 b的原型】、 toString、constructor，建议用 toString,另外两个有个小 bug——跨父子域不行） 1）如果是数组，就新建一个空数组； 2）如果是对象，就新建一个空对象。

4、建立了数组以后，如果是挨个看原始对象里面是什么，都是原始值就可以直接考 过来了；或者，建立了对象以后，挨个判断对象里面的每一个值，看是原始值还是 引用值

5、递归

23. 简述创建函数的几种方法？

答：1.函数声明：function sum1(num1,num2){

Return num1+num2}

2.函数表达式：var sum2=function(num1,num2){

Return num1+num2}

3.函数对象方式：var sum3=new Function(“num1”,”num2”,:return num1+num2”);

24.Error异常对象？

答：Error.message:可以读取的错误消息。

Error.name:错误的类型

Error.toString():把Error对象转化成字符串

EvalError:在不正确使用eval（）时抛出

SyntaxError:抛出该错误用来通知语法错误

ReferenceError:在读取不存在的变量时抛出

RangeError:在数字超出合法范围时抛出

TypeError:当一个值的类型错误时，抛出该异常

URIError:由URI的编码和解码方法抛出

25.Function函数构造器？

答：Function.apply( ) 将函数作为一个对象的方法调用

Function.arguments[] 传递给函数的参数

Function.call( ) 将函数作为对象的方法调用

Function.caller 调用当前函数的函数

Function.length 已声明的参数的个数

Function.prototype 对象类的原型

Function.toString( ) 把函数转换成字符串

26.String字符串对象？

答：> Length 获取字符串的长度。

> toLowerCase() 将字符串中的字母转成全小写。

> toUpperCase() 将字符串中的字母转成全大写。

> charAt(index) 返回指定下标位置的一个字符。如果没有找到，则返回空字符串

> substr() 在原始字符串，返回一个子字符串

> substring() 在原始字符串，返回一个子字符串。

> split() 将一个字符串转成数组。

> charCodeAt( ) 返回字符串中的第 n个字符的代码

> concat( ) 连接字符串

> fromCharCode( ) 从字符编码创建—个字符串

> indexOf( ) 返回一个子字符串在原始字符串中的索引值(查找顺序从左往右查找)。如果没 有找到，则返回-1。

> lastIndexOf( ) 从后向前检索一个字符串

> localeCompare( ) 用本地特定的顺序来比较两个字符串

> match( ) 找到一个或多个正则表达式的匹配

> replace( ) 替换一个与正则表达式匹配的子串

> search( ) 检索与正则表达式相匹配的子串

> slice( ) 抽取一个子串

> toLocaleLowerCase( ) 把字符串转换小写

> toLocaleUpperCase( ) 将字符串转换成大写

> toLowerCase( ) 将字符串转换成小写

> toString( ) 返回字符串

> toUpperCase( ) 将字符串转换成大写

> valueOf( ) 返回字符串

27．什么是闭包？

答：简单的说，在一个作用域中可以访问另一个局部作用域的变量，延展了函数的作用域范围，使用的变量不会被自动回收，而是会一直占用内存空间，

闭包的作用，就是保存自己私有的变量，通过提供的接口（方法）给外部使用，，但外部不能直接访问该变量。

28.什么是原型链？

答：Javascript是面向对象的，每一个实例对象都有一个\_proto\_属性，该属性指向它原型对象，这个实例对象的构造函数有一个原型属性prototype，与实例的\_proto\_属性指向同一个对象。当一个对象在查找一个属性的时候，自身没有就会根据\_proto\_向它的原型进行查找，如果都没有，则向它的原型的原型继续查找，知道查到Object.prototype.\_proto\_为null，这样也就形成了原型链。

29.有哪些方式继承？

答：> 1. 借用构造函数。也叫伪造对象或经典继承。 思路：在子类构造函数的内部调用超类型构造函数。可以通过使用 apply()和call()方法 在新创建的对象上执行构造函数。 缺点：方法都在构造函数中定义，函数的复用就无从谈起。在超类型的原型中定义的方 法，对子类而言也是不可见的，结果所有的类型都只能使用构造函数模式。

> 2. 组合继承。也叫伪经典继承。指的是将原型链和借用构造函数的技术组合到一起， 从而发挥二者之长。 思路：使用原型链实现对原型属性属性和方法的继承，通过借用构造函数来实现实例属 性的继承。 优点：既通过在原型上定义方法实现了函数复用，又能保证每一个实例都有它自己的数 组。 组合继承避免了原型链和借用构造函数的缺陷，融合了他们的优点，成为 JavaScript 中常用的继承模式。

> 3. 原型链继承。 思路：借助原型可以基于已有的对象创建对象，同时还不必因此创建自定义类型。 在object()函数内部，先创建一个临时的构造函数，然后将传入的对象作为这个构造函 数的原型，最后返回了这个临时类型的一个新实例。

> 4. 寄生式继承。 思路：创建一个仅用于封装继承过程的函数，该函数在内部以某种方式来增强对象，最 后再像真的是它做了所有的工作一样返回对象。缺点：使用寄生式继承来为对象添加函数，会由于不能做到函数复用而降低效率，这一 点和构造函数模式类似。

> 5. ）寄生组合式继承。是JavaScript 最常用的继承模式。 思路：通过借用构造函数来继承属性，通过原型链的混成形式来继承方法。 本质上，就是使用寄生式继承来继承超类型的原型，然后再将结果指定给子类型的原型。 开发人员普遍认为寄生组合式继承时引用类型最理想的继承范式。 extend（）方法才用了这样的方式。

30.DOM节点的增删改查？

答：> 1. 创建节点、追加节点

createElement（标签名）创建一个元素节点（具体的一个元素）。

createTextNode（节点文本内容）创建一个文本节点。

createDocumentFragment() //创建一个 DOM 片段。

appendChild（节点）追加一个节点。

>2. 插入节点

appendChild（节点）也是一种插入节点的方式，还可以添加已经存在的元素，会将其元素从原来的位置移到新的位置。

insertBefore（a,b）是参照节点，意思是 a节点会插入 b节点的前面。

> 3. 删除、移除节点

removeChild(节点) 删除一个节点，用于移除删除一个参数（节点）。其返回的被移除的节点，被移除的节点仍在文档中，只是文档中已没有其位置了。

>4. 复制节点

cloneNode() 方法，用于复制节点， 接受一个布尔值参数， true 表示深复制（复制节点及其所有子节点）， false 表示浅复制（复制节点本身，不复制子节点）。

> 5. 替换节点

replaceChild(插入的节点，被替换的节点) ，用于替换节点，接受两个参数，第一参数是要插入的节点，第二个是要被替换的节点。返回的是被替换的节点。

> 6. 查找节点

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

31.JS中的typeof关键字能返回哪些数据类型？

答：typeof一般判断基本数据类型，是一个操作符而不是函数，圆括号可有可无。

Typeof返回值有：string,number,Boolean,undefined,object,function.

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

基本数据类型中数字、字符串、布尔类型返回其对象类型undefined返回undefined。

九大内置构造函数及其他所有函数返回function，其他所有复杂类型对象和null返回object。

32.什么是短路表达式？

答：短路表达式只是y一种简写形式，也就是用&&和||来赋值或者执行函数的形式。

例如：var foo=foo1 || foo2;意思是如果foo1是真的，那么就把foo1的值赋给foo，否则就把foo2的值赋给foo。

Foo &&foo()当foo存在的时候，我们就执行foo函数，如果这个时候foo不是一个函数，就会报错，所以这个只是一种简写形式而已。

33.控制台中使用哪些部分调试？

答：只要是用console.来进行调试> 1. console.log 用于输出普通信息

> 2. console.info 用于输出提示性信息

> 3. console.error用于输出错误信息

> 4. console.warn用于输出警示信息

> 5. console.debug用于输出调试信息

34. 说说事件委托机制？这样做由什么好处？

答： 事件委托，就是某个事件本来该自己干的，但是自己不干，交给别人来干。就叫事件委 托。打个比方：一个button 对象，本来自己需要监控自身的点击事件，但是自己不来监控这个点击事件，让自己的父节点来监控自己的点击事件。

>优点： 提高性能：列如，当有很多 li同时需要注册事件的时候，如果使用传统方法来注册 事件的话，需要给每一个 li 注册事件。然而如果使用委托事件的话，就只需要将事件委托给 该一个元素即可。这样就能提高性能。

35. bind,call和apply的区别？

答：它们的共同之处：都“可以用来代替另一个对象调用一个方法，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。”

不同之处：  Apply：最多只能有两个参数——新this 对象和一个数组 argArray。如果给该方法传递多 个参数，则把参数都写进这个数组里面，当然，即使只有一个参数，也要写进数组里面。如 果 argArray 不是一个有效的数组或者不是 arguments 对象，那么将导致一个 TypeError。如果没有提供 argArray 和 thisObj 任何一个参数，那么 Global 对象将被 用作 thisObj， 并且无法被传递任何参数。

Call：则是直接的参数列表，主要用在js 对象各方法互相调用的时候，使当前 this 实例指 针保持一致，或在特殊情况下需要改变this指针。如果没有提供 thisObj 参数，那么 Global 对象被用作 thisObj。

Bind:ES5中新增方法，ie9以后才支持。新建一个方法，bind中的第一个参数可以改变函数中的this指向，如果传的是null，那么this的指向不会有任何改变。Bind是函数的方法，只有函数才可以调用，返回值是一个新的函数，返回的函数体跟bind的调用函数是一致的，可以有第二个第三个参数，具体多少不确定，会把第一个参数之后所有参数优先传递给函数去执行。

36.在JS的计时器运行原理是怎样的，为什么可以触发计时效果？计时器是多线程吗？

答：> 1. javascript引擎只有一个线程，强迫异步事件排队等待被执行。

> 2. setTimeout 和setInterval 本质上不同的地方是他们如何执行异步代码的。

> 3. 如果一个定时器正在执行的时候被阻塞了，那么它将会被推迟到下一个可能的执行点，这既是使得延迟时间有可能会超过声明定时器时设置的值。

> 4. Interval 如果有足够的时间来执行（大于制定的延迟），那么它将会无延迟的一个紧 接着一个执行。

> 5. 原理： 计时器通过设定一定的时间段（毫秒）来异步的执行一段代码。因为 Javascript 是一 个单线程语言，计时器提供了一种绕过这种语言限制来执行代码的能力。

> 6. 总结： 计时器是单线程的， 需要等待上一个执行完， 如果上一个没有执行完， 下一个需要 延迟执行， 直到上一个执行完。

37. 面向对象和类的区别？

答：简单的说类是对象的模板。

>在js中没有类，所以在js中所谓的类就是构造函数，对象就是由构造函数创建出来的实例对象，面向对象就是使用面向对象的方式处理问题，面向对象是对面向过程进行封装。

>面向对象三大特性：抽象性，需要通过核心数据和特定环境才能描述对象的具体意义。

封装性，封装就是将数据和功能组合到一起，在js中对象就是键值对的集合，对象将属性和方法封装起来，方法将过程封装起来。

继承性，将别人的属性和方法成为自己的，传统继承基于模板类，Js中继承基于构造函数。

38 .JavaScript中的垃圾回收机制？

答：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。

方式：1.标记清除：大部分浏览器以此方式进行垃圾回收，当变量进入执行环境（函数中声明变量）的时候，垃圾回收器将其标记为“进入环境”，当变量离开环境的时候（函数执行结束）将其标记为“离开环境”，再离开环境之后还有的变量则是需要被删除的变量。标记方式不定，可以是某个特殊位的反转或维护一个列表等。

2.引用计数：这种方式常常会引起内存泄漏，低版本的ie使用这种方式。机制就是跟踪一个值的引用次数，当声明一个变量并将一个引用类型赋值给该变量时该值引用次数加1，当这个变量指向其他一个时该值的引用次数便减一。当该值引用次数为0时就会被回收。

39. 列出3条以上FF和IE的脚本兼容问题。

答：> 1. window.event：表示当前的事件对象，IE有这个对象，FF没有，FF通过给事件处理函数传递事件对象。

>2. 获取事件源： IE用srcElement获取事件源，而FF用target获取事件源

>3.添加，去除事件： IE：element.attachEvent(“onclick”,function)       element.detachEvent(“onclick”, function)

FF：element.addEventListener(“click”,function, true) element.removeEventListener(“click”, function, true)

>4.获取标签的自定义属性：IE：div1.value或div1[‘value’]

FF:可用div1.getAtrribute(“value”)

40. 谈谈你对jQuery的理解？

答：> JQuery 是继 prototype 之后又一个优秀的 Javascript 库。它是轻量级的js 库 ，它兼容 CSS3，还兼容各种浏览器（IE 6.0+，FF1.5+，Safari 2.0+，Opera 9.0+），jQuery2.0及后续版本将不再支持 IE6/7/8 浏览器。jQuery 使用户能更方便地处理 HTML（标准通用标记语言下的一个应用）、events、实现动画效果，并且方便地为网站提供 AJAX 交互。

> jQuery还有一个比较大的优势是，它的文档说明很全，而且各种应用也说得很详细，同时还有许多成熟的插件可供选择。jQuery 能够使用户的 html页面保持代码和 html 内容 分离，也就是说，不用再在 html里面插入一堆 js 来调用命令了，只需要定义 id即可。

> jQuery 是一个兼容多浏览器的 javascript 库，核心理念是write less，do more(写得更少，做得更多)。 jQuery是免费、开源的，使用 MIT 许可协议。jQuery 的语法设计可以使开发更加便捷，例如操作文档对象、选择 DOM 元素、制作动画效果、事件处理、使用 Ajax 以及其他功能。除此以外，jQuery 提供 API让开发者编写插件。其模块化的使用方式使开发者可以很轻松的开发出功能强大的静态或动态网页。

41. 属性查找规则？

答：1.先在对象obj上查找，找到则直接返回该属性，结束查找。

2.在对象的原型obj\_proto\_上查找，找到则直接返回该属性，结束查找。

3.在原型的原型（原型链）上查找，找到则直接返回该属性，结束查找。

4.重复第三部，知道原型指向null，结束查找，返回undefined。

42.Javascript中callee和caller的作用？

答：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。

43.原型和原型链的关系？

答：在JavaScript中，一共有两种类型的值，原始值和对象值，每个对象都有一个内部属性[prototype]，我们通常称之为原型。

原型的值可以是一个对象，也可以是null，如果它的值是一个对象，则这个对象也一定有自己的原型，这样就形成了一条线性的链，我们称之为原型链（比如我们新建一个数组，数组的方法就是从数组的原型上继承而来的）。

44.原型和原型对象？

答：1.所有函数的\_proto\_都是指向function 的prototype。

2.构造函数new出来的对象\_proto\_指向构造函数的prototype。

3.非构造函数实例化出的对象或者对象的prototype的\_proto\_指向object的prototype.

4.object的prototype指向null。

45. 了解递归，了解斐波那契数列？

答：简单的定义：当函数直接或间接调用自己时，则发生了递归。但递归并不直观，也不符合我们的思维习惯。因为递归的过程就是出入桟的过程。所以，每个递归程序都可以把它改写为非递归版本，只需要桟，通过入栈和出栈两个操作就可模拟递归的过程。

代码：int fibo(int n) {

Int f2=0,f0=2,f1=1;

For(var int i=0;i<n-1;++i){

F2=f0+f1;

F0=f1;

F1=f2;

}

Return f2;

}

46.innerHTML和innerText的区别？

答：inner HTML获取内容的时候，如果内容中有标签，会把标签页获取到，原封不动把内容获取到，

InnerText：获取内容的时候，会把标签过滤掉，会把前后的换行和空白都去掉。

47.DOM获取元素的集中方式？

答：document.getElementById:获取id元素

document.getElementByTagName:获取标签元素

document.getElementByClassName:根据标签的class属性获取元素

document.getElementByName:根据标签的name属性获取元素

document.querySelector(‘a’):只会返回第一个元素

document.querySelectorAll:返回所有匹配到的元素

48.事件的基本使用？

答：方法一：btn.onclick=function(){console.log(1)};

方法二：Btn.addEventListener(‘click’,function(){console.log(11)}

49. 非表单元素的属性以及表单元素的属性？

答：非表单元素：href,title,id,src,className,background

表单元素属性：value 用于大部分元素的内容获取（option除外）

Type 可以获取input标签的类型（输入框复选框等）

Disable 禁用属性，只有一个属性就是disable

Checked 复选框选中属性

Selected 下拉菜单选中属性

50. Java捕获异常处理的常用方法有哪些？

答：在Java中，异常情况分为Exception（异常）和Error（错误）两大类，Java异常通常是指程序运行过程中出现的非正常情况，如用户输入错误，除零为零，需要处理的文件不存在、数组下标越界等，对于异常情况的出现，可以采用异常处理，一保证程序可以正常执行。

Java中定义两种类型的异常和错误：

1.JVM(Java虚拟机)异常：由JVM抛出的异常或错误，例如：NullPointerException类，ArrayIndexOutOfBoundsException类，ClassCastException类。

2.程序级异常：由程序或者api程序抛出的异常，例如：IllegalArgumentException类。

捕获异常的方法：使用try和catch关键字可以捕获异常，try/catch代码块中的代码称为保护代码，可以放在异常可能发生的地方，try{程序代码}catch（ExceptionName el）{catch块}catch语句包含要捕获异常类型的声明。当报错代码块中发生一个异常时，try后面的catch快就会被检查。如果发生的异常包含在catch快中，异常会被传递到该catch块，这和传递一个参数到方法时一样的。也可以进行多重捕获，具体用法实在try代码块后面跟随多个catch代码块。

Catch语句往往是和finally配合使用，finally关键字用来创建在try代码块后面执行的代码块，无论是否发生异常，finally代码块中的代码总会被执行，因此，在finally代码块中，可以运行清理类型等首尾善后性值的语句。

声明自定义异常：在Java中可以自定义异常，在自定义异常类时需注意以下几点。

1. 所有异常都必须是throwable的子类；
2. 自定义检查性异常类，则需要继承exception类；
3. 自定义运行时异常类，则需要继承runtime exception类；

自定义的异常类和其他任何类都一样，包含由变量和方法。起作用机制均是抛出异常和捕捉异常，一个方法能捕捉异常，一定是Java代码在某处所抛出的异常，异常总是先被抛出，后被捕捉的。

51.jQuery中attr()和prop()区别？

答：对于html元素本身就带有的固有属性，再处理时，使用prop方法。

具有true和false两个属性的属性，如checked,selected或者disabled使用prop().

对于html元素我们自己定义的DOM属性，再处理时，使用attr方法。

Prop（）函数的结果：

1.如果有相应的属性，返回指定属性值。

2.如果没有相应的属性，返回值是空字符串。

Attr()函数的结果：

1.如果有相应的属性，返回指定属性值。

2.如果没有相应的属性，返回值是undefined。

总结：再获取某些值为false或者true的属性时，attr不能正常获取，只有prop才能正常获取。

52.jquery中扩展方法？

答：jQuery.extend：Query本身的扩展方法 ，是扩展jQuery的这个类。

jQuery.fn.extent(Object) jQuery 所选对象扩展方法

53.简述在jQuery中.eq()和.get()的异同？

答：相同：get()取得其中一个匹配的元素，数字序号表示取得第几个匹配的元素。

Eq():获取第n个元素，下标都是从0开始，用法基本相同。

不同：eq返回的是一个jQuery对象，就可以继续调用其他方法。

Get返回的是一个html对象数组，不能调用jQuery的其他方法。

54.双伪元素特点：before，after？

答：是一个行内元素，如果想设置宽高则需要转成display：block；或者position，float；

只有双标签才可以使用双伪元素，必须添加content，哪怕不设置内容也要。

在旧版本中是伪类，新版本中是伪元素，自动识别。

Ie6，ie7，ie8不支持此版本。

Js不可以操作。

H5C3部分

1.H5新增特性有哪些？

答：语意化标签（header，nav,main,footer），新增表单属性(email,tel,url,number,search,range,color,date,time)，本地存储特性(sessionStorage,localstorage)，网页多媒体(embed,flash,audio,video)，二维三维，特效（过渡，动画）。

2. HTML5兼容性解决方案？

答：1.将HTML5元素定义为块元素：html5定义了8个新的HTML语义元素，所有这些元素都是块级元素，为了能让旧版本的浏览器正确显示这些元素，可以设置CSS的display属性值为block;

2.为HTML添加新元素：document.createElement(“myElement”)是为ie浏览器添加新的元素。

3.通过HTML5的底层框架来屏蔽不同浏览器的兼容性问题，所有的Html5应用在框架之上来开发。

3.cookie、sessionStorage和localStorage的区别？

答：cookie是网站为了标示用户身份而存储在用户本地终端（client Side）上的数据（通常经过加密）。

Cookie数据始终在同源的http请求中携带（即使不需要），记住在浏览器和服务器间来回传递。

sessionStorage和localStorage不会自动把数据发送给服务器，仅在本地保存。

存储大小：

Cookie数据大小不能超过4K;

sessinStorage数据存储在本地，数据大小5mb左右，以键值对的形式存储。

LocalStorage数据存储在硬盘上，大小为20mb左右。

2有期时间：

localStorage存储永久数据，浏览器关闭后数据不会丢失，除非主动删除数据；

sessionStorage数据在当前浏览器窗口关闭后自动给删除；

cookie设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭。

4.HTML5的离线储存怎么使用，工作原理能不能解释以下？

答:在用户没有与因特网链接时，可以正常访问站点或应用，在用户与因特网链接时，更新用户机器上的缓存文件。

原理：html5的离线存储时基于一个新建的.appcache文件的缓存机制（不是存储技术），通过这个文件上的解析清单离线存储资源，这些资源就会像cookie一样被存储下来，之后当网络在处于离线状态下时，浏览器会通过离线存储的数据进行页面展示。

如何使用：1.页面头部像下面一样加入一个manifest的属性；

2.在cache.manifest文件的编写离线存储的资源；

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

5.如何实现多个浏览器内多个标签页之间的通信？

答：WebSocket、SharedWorker；

Cookie也可以实现，需要设置共享参数。

通过url页面跳转带入的参数。

Localstorage另一个浏览上下文被添加、修改或删除时，它都会触发一个事件，我们通过监听事件，控制它的值来进行页面信息通信。

6.CSS3有哪些新特性？

答：新增各种css选择器，圆角，多列布局，阴影和反射，文字特效，文字渲染，线性渐变，旋转，缩放，定位，动画。

7.伪类选择器？

答：first-child:查找父元素的第一个子元素,

last-child：查找父元素的最后一个子元素,

nth-child(even)：所有的偶元素,

nth-child(odd)：所有的奇数,

nth-of-type(n)：指定类型，其他类型过滤掉,

empty：选中没有任何子节点的元素，注意空格也算子元素,

target：结合描点进行使用，处于当前描点的元素才会被选中。

8.盒模型？

答：页面就是由一个个盒模型堆砌起来的，每个HTML元素都可以叫做盒模型，盒模型由外而内包括：边距（margin）、边框（border）、填充（padding）、内容（content）。它在页面中所占的实际宽度是margin + border + paddint + content 的宽度相加。

IE的盒模型是指：border+padding+content的值；而w3c标准盒模型内容大小是指content的大小。

9.h5和h4的区别/html和xml的区别?

答：语义化：可以优化搜索引擎，便于百度、谷歌搜索时的排名。

新的标签/属性：本地存储/canvas/vudio等

Document不需要加一串代码：

HTML:超文本标记语言

XML：可扩展标记语言

1.语法上

HTML：语法上要求不是很严格

XML：与之相反，严格区分大小写

2.标记不同

HTML：使用固有的标记，而xml没有固有的标记。

Html标签时预定义的，XML标签时免费的、自定义的、可扩展的。

3.作用不同

Html是用来显示数据的，xml是用来描述数据、存放数据的，所以可以作为持久化的介质。Html将数据和显示结合在一起，在页面中把这数据显示出来，xml则将数据和显示分开，xml被设计用来描述数据，其焦点是数据的内容。Html被设计用来显示数据，其焦点是数据的外观。

10.flex布局的注意事项？

答：1.flex布局以后，子元素的float、clear和vertical-align属性将失效。

2.flex内部的子元素（除了position：absolute或fixed）无论是display：block或者display：inline，都成为了伸缩项目。

3.Android浏览器4.4之前对于flex布局支持不好。

4.flex-shrink属性用于设置或检索弹性盒的收缩比率，默认为1。不允许为负值，flex-shrink值为0时表示不收缩，保持自身。

5.pc兼容性相对于较差，ie要10，甚至11以上才有很好的兼容。

11.CSS3动画和jQuery动画的区别？

答：1.css3中的过渡和animation动画都是基于css实现机制的，属于css范畴之内，并没有设计导任何语言操作，效率率高，与jQuery中的animation（）函数，但兼容性很差。

2.jQuery中的animation（）函数可以简单的理解为css样式的逐帧动画，是css样式不同状态的快速的效果，效率略低于css3动画执行效率，但是兼容性好。

3．jQuery中的animation（）函数中的参数：1.键值对-🡪对象，2.时间，3函数🡪回调函数，此方法不能改变背景颜色，需要一个jQuery插件。

4.css3中的animation函数必填项：animation-name指定动画的名称，animation-duration设置动画总耗时。创建动画@keyframes moveTest{-iteration-count:设置动画的播放次数，默认为1，可以是定具体的数值，也可以指定infinite无限次；}.

12.背景图片contain和cover的区别？

答：contain：按比例调整图片大小，使用图片宽高自适应整个元素的背景区域，使图片以最大程度全部包含在容器内。

图片小于容器：有可能造成容器空白区域，将图片放大。

图片大于容器：有可能造成容器空白区域，将图片缩小。

Cover：与contain刚好相反，背景图片会按比例缩放自适应整个背景区域，如果背景区域包含不了，图片内容会溢出。

图片大于容器：等比例缩小，会填满整个背景区域，有可能造成图片的某些区域不可见。

图片小于容器：等比例放大，填满整个背景区域，图片有可能造成某个方向上内容溢出。

移动web

1.要你出一套适应不同分辨率，不同终端的前端实现方案你有什么思路？

答：流式布局：使用非固定像素来定义网页内容，也就是百分比布局，通过盒子的宽度设置成百分比来根据屏幕的宽度来进行伸缩，不受固定像素的限制，内容向两侧填充。这样的布局方式，就是移动web开发使用的常用布局方式，这样的布局可以适配移动端不同的分辨率设备。

响应式开发：简而言之，就是一个网站能够兼容多个终端，越来越多的设计师也采用了这种设计。CSS3中的Media Query(媒体查询)。通过查询screen的宽度来指定某个宽度区间的网页布局。

> 超小屏幕（移动设备） 768px以下

> 小屏设备 768px-992px

> 中等屏幕 992px-1200px

> 宽屏设备 1200px 以上，由于响应式开发显得繁琐，一般使用第三种响应式框架来完成，比如bootstrap来完成一部分工作，当然也可以自己写响应式。

2.px em rem取用选择依据？

答：1.px像素，绝对单位，像素px是相对于显示器屏幕分辨率而言的，是一个虚拟长度单位，是计算机系统的数字化图像单位，如果px要换算成物理长度，需要指定精度DPI。

2.em是相对长度单位，相对于当前对象文本的字体尺寸，如果当前文本的字体尺寸未被人为设置，则相对于浏览器的默认字体尺寸，它会继承父级元素的字体大小，因此并不是一个固定的值。

3.rem是CSS3新增的一个相对单位，使用rem为元素设定字体大小时，任然是相对大小，但相对于的只是HTML根元素。

4.区别：IE无法调整那些使用px作为单位字体大小，而em和rem可以缩放，rem相对的只是HTML根元素，这个单位可谓集相对大小和绝对大小的优点于一声，通过它既可以做到只修改根元素就成比例地调整所有字体大小，又可以避免字体大小逐层复合的连锁反应。目前，除了IE8及更早版本外，所有浏览器均已支持rem。

3.Zepto和jQuery的区别？

答：Zepto相对jQuery更加轻量，主要用在移动端，jQuery也有对应的jQuery Mobile移动端框架。

4.移动端touch事件判断滑屏手势的方向？

答：当开始一个touchstart事件的时候，获取此手指的横坐标startX和纵坐标startY;当触发touchmove事件时，在获取此手指的横坐标moveEndX和纵坐标moveEndY;最后，通过这两次获取的坐标差值来判断手指在屏幕上的滑动方向。

用touchmove的最后坐标减去touchstart的起始坐标，X的结果如果正数，则说明手指是从左往右划动，X的结果如果负数，则说明手指是从右往左划动；Y的结果如果正数，则说明手指是从上往下划动，Y的结果如果负数，则说明手指是从下往上划动。

> startX = e.touches[0].pageX;

> startY = e.touches[0].pageY;

> moveEndX = e.changedTouches[0].pageX;

> moveEndY = e.changedTouches[0].pageY;

> X = moveEndX - startX;

> Y = moveEndY - startY;

> if ( X > 0 ) {alert(‘向右’);}

> else if ( X < 0 ) {alert(‘向左’);}

> else if ( Y > 0) {alert(‘向下’);}

> else if ( Y < 0 ) { alert(‘向上’);}

5.移动端对图片优化有哪些方式，怎么实现？

答：1.去掉无意义的修饰

2.不用图片

3.使用矢量图替代位图

4.使用恰当的图片格式

5.使用data url

6.按照HTTP协议设置合理的缓存

7.使用支持SPDY的服务器

8.资源的lazyload或postpone（lazyload:延迟到其他资源下载完成后再加载；postpone：延迟到元素可见在加载。

6．Rem布局中的尺寸是怎样计算的，实际举例说明一下？

答：其实rem布局的本质是等比缩放，一般是基于宽度，试想一下如果UE图能够等比缩放，假设我们将屏幕宽度平均分成100份，每一份的宽度用x表示，x=屏幕宽度/100，如果将X作为单位，x前面的数值就代表屏幕宽度的百分比。

7.移动端常见兼容性问题？

答：1.设置文字行高为字体行高，解决文字上下边留白问题。

2.给动态元素添加事件，需要使用事件委托（绑定到document），解绑也需要用委托的方式。苹果机点击事件不能触发，需要用touch系列事件。

3.Img标签src属性无值（php渲染过的），在苹果机上显示无图片，在安卓机上显示图片裂开，可加alt属性及值。

4.同一标签多次绑定同一个四件（页面复杂情况容易粗线这种情况，尽量避免这种情况），可以减少bug的出现，利玉维护页面。

5.在rem自适应页面使用精灵图，会容易出现图片缺角的问题（约1-2像素）。解决办法（使装精灵图的盒子变大，让图片居中显示）。

6.当祖父元素用overflow属性时，父元素采用transform属性会影响子元素定位position：absolute，导致子元素超出隐藏，建议用其他属性替换transform属性。

7.click事件在ISO系统上有时失效，给绑定click事件的元素加上cursor：pointer解决。

8.placeholder垂直居中问题：在iOS和android中显示不同，解决方法时：在保证input输入文本垂直居中的条件下，给placehoder设置padding-top.

8.列嵌套注意的事项？

答：如果在外层没有在包括container，那么嵌套列的宽度就是参照当前所在的栅格，如果在外层添加了container，那么参照就是就是核心样式文件所设置的容器宽度。

9.less样式

答：主要用于嵌套：实现选择器的继承，可以减少代码量，同时使用代码结构更清晰。

Less中，@important引入其他less样式。

混入：可以将一个定义好的样式引入到另一个样式中，类似于函数的调用。、

10.bootstrap经典布局是什么，有什么特点？

答：特点：1跨设备，跨浏览器：可以兼容所有现代浏览器，包括ie7、8；

2.响应布局：部队那支持pc端的各种分辨率，还支持移动端pad，手机登屏幕的响应切换显示

3.提供的全面的组件：bootrasp提供了实用性很强的组件，包括：导航，标签，工具条，按钮等开发者使用

4.内置jQuery插件：bootstrap提供了很多实用性的jQuery插件，这些插件方便开发者实现web中各种常规特效。

5.支持html5css3z:html5语义化标签和css3属性，都可以很好的支持。

6.支持less动态样式。

**PHP.Ajax部分**

1.表单验证传输的什么数据？明文还是暗文==加密？如何加密？是每一次传输数据，都是加密之后才传输么？

答：GET是从服务器上请求数据，POST是发送数据到发送数据到服务器。事实上，GET方法是把数据参数队列（query string）加到一个URL上，值和表单是一一对应的，比如说，name=John，在队列里，值和表单用一个&符号分开，空格用+号替换，特殊的符号转换成十六进制的代码。因为这一队列在URL里边，这样队列的参数就能看得到，可以被记录下来，或更改。通常GET方法还限制字符的大小（大概是256字节）。事实上POST方法可以没有事件限制的传递数据到服务器，用户在浏览器端是看不到这一过程的，所以POST方法比较适合用于发送一个保密（比如信用卡号）或者比较大量的数据到服务器。

区别：POST是允许传输大量数据的方法，而GET方法会将传输的数据附在数据附在网址后面，然后一起送达服务器，因此传送的数据量就会受到限制，但是执行效率却比POST方法好。

总结：>get方式的安全性较post方式要差些，包含机密信息的话，建议用post数据提交方式；

>在做数据查询时，建议用get方式，而在做数据添加、修改或删除时，建议用Post方式；

>所以，表达如果时向服务器传输数据（如账号密码等）都是加密数据（post）,如果只是单单想要从服务器获得数据或者传输的数据并不重要，可以直接使用明文方式传输（get）

2.如何实现跨域？

答：JSONP(JSONN with Padding 填充式JSON或参数式JSON)

在js中，我们虽然不能直接用XMLHttpRequest请求不同域上的数据，但是在页面在页面上引入不同域上的js脚本文件却是可以的，jsonp正是利用这个特性来实现的。

JSONP由两部分组成：回调函数和数据，回调函数式是当响应到来时应该在页面中调用的函数，而数据就是传入回调函数中的JSON数据。

优点：它的兼容性更好，在更加古老的浏览器中都可以运行，不需要XMLHttpRequest或ActiveX的支持；能够直接访问响应文本，支持在浏览器与服务器之间双向通信。

缺点：JSONP是从其他域中加载代码执行，如果其他域不安全，很可能会在响应中夹带一些恶意代码，而此时除了完全放弃JSONP调用之外，没有办法追究。因此在使用不是你自己运维的web服务时，一定得保证他安全可靠。它只支持GET请求而不支持POST等其他类型的HTTP请求，它只支持跨域HTTP请求这种情况，不能解决不同域的两个页面之间如何进行JavaScript调用的问题。

3.PHP inset和empty的区别？

答：empty函数用途：检测变量是否为空，

判断：如果var时非空或非零的值，则empty（）返回false。换句话说，””、0、“0”、null、false、array（）、var$var；以及没有任何属性的对象都被认为时空的，如果var为空，则返回true。注意：empty（）只检测变量，检测任何非变量的东西都将导致解析错误。换句话说，后边的语句将不会起作用

Unset删除变量，is\_string判断是否是字符串，如若是则为true。

Isset函数 用途：简述变量是否设置

判断：检测变量是否设置，并且不是null，如果已经使用unset（）释放了一个变量之后，它将不再是isset（）。若使用isset（）测试一个被设置成null的变量，将返回false，同时要注意的是一个null字节（“\0”）并不等于PHP的null常熟。

4．Agax是什么？

答：ajax并不算是一种新的技术，全称是asychroonous javascript and xml，可以说是已有技术的组合，主要用来实现客户端域服务器端的异步通信效果，实现页面的局部刷新，早期的浏览器并不能原声支持Ajax，可以使用影藏帧（iframa）方式变相实现异步效果，后来的浏览器提供了对Ajax的原声支持。

使用Ajax原生方式发送请求主要通过xmlhttprequest（标准浏览器）、activexobject(ie浏览器)对象实现异步通信效果。

5.页面编码和被请求的资源编码不一样如何处理？

答：对于Ajax请求传递的参数，如果是get请求方式，参数如果传递中文，在有写浏览器会乱码，不同的浏览器对参数编码的处理饭是钢hi不同，所以对于get请求的参数需要使用encodeUPIComponet函数对参数进行编码处理，后台开发语言都有相应的解码api，对于post请求不需要进行编码。

6.简述Ajax的过程？

答：1.创建XMLHttpReuest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，并制定该HTTP请求的方法，URL及验证信息。

1. 设置响应HTTP请求状态变化的函数。
2. 发送HTTP请求。
3. 获取异步调用返回的数据。
4. 使用JavaScript和DOM实现局部刷新。

7.请解释一下JavaScript的同源策略？

答：同源策略在什么情况下不会起作用呢？当页面使用多个<iframa>元素或者打开其他浏览器窗口的时候，这一策略就会起作用。

同源策略的含义：脚本智能读取和所属文档来源相同的窗口和文档的属性。这里就涉及到了一个浏览器如何判断两者是否同源以及如何判断脚本来源的问题。

注意一点：脚本本省的来源并不作为判断是否同源的依据，而是将脚本所属文档的来源作为判断依据。

8.get和post的区别？

答：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，有的浏览器是8000个字符。

Post：一般用于修改服务器上的资源，对发送的信息没有限制。

在以下情况下，使用post请求：

1. 无法使用缓存文件（更新服务器上的文件或数据库）
2. 向服务器发送大量数据（post没有数据量限制）
3. 发送包含未知字符的用户输入时，post比get更稳定页更可靠。

9.解释jsonp的原理？

答：jsonp并不是一种数据格式，而json是一种数据格式 ，jsonp是用来解决跨域获取数据的一种解决方案，具体是用过动态创建script标签，然后通过标签的src属性获取js文件中的js脚本，该脚本的内容是一个函数调用，参数就是服务器返回的数据，为了处理这些返回的数据，需要事先在页面定义好回调函数，本质上使用的并不是Ajax技术。

10.Ajax请求时如何解释json数据？

答：使用eval()或者json.parse()鉴于安全性考虑，推荐使用json.parse()更靠谱，对数据的安全性更好。

11.HTTP状态吗都有哪些？

答：100 正在初始化（一般是看不到的）

101 正在切换协议（websocket浏览器提供的）

200或者以2揩油的两位数 都是代表响应主体的内容已经成功返回了

202 表示接受

301 永久重定向/永久转移

302 临时重定向/临时转移（议案用来做服务器负载均衡）

304 本次获取的内容是读取缓存中的数据，会每次去服务器校验

400 参数出现错误（客户端传递给服务器端的参数出现错误）

401 禁止访问，没有权限

404 客户端访问的地址不存在

500 未知的服务器错误

503 服务器超负荷（假设一台服务器只能承受10000人，当第10001人访问的时候，如果服务器没有做负载均衡，那么这个人的网络状态码就是503）

12.浏览器渲染页面的过程？

答：浏览器会把HTML、SVG、XHTML三种格式的文件会产生一个DOM树，css解析css会产生css规则树JavaScript会通过DOM api来操作DOM树和css规则树。

13.简述描述下同步和异步的区别，并举例实际应用中哪些是同步，哪些是异步？

答：同步是阻塞的，浏览器向服务器发送请求，服务器比较忙，浏览器一直等着（页面白屏），直到服务器返回数据，页面才可以正常显示。

异步是非阻塞的，浏览器向服务器请求数据，服务器比较忙，浏览器可以干自己原来的事情（显示页面），服务器返回数据的时候通知浏览器一声，浏览器把返回的数据再渲染到页面，局部更新。

14.简述Ajax的工作原理？

答：Ajax的原理简单来说通过XMLHTTPREQUEST对象来向服务器发异步请求，从服务器获得数据，然后用JavaScript来操作dom而更新页面，这其中最关键的一步就是从服务器获得请求数据。

XMLHTTPREQUEST是Ajax的核心机制，他是在ie5中首先引入的，是一种支持异步请求的技术，简单的来说，也就是JavaScript可以及时向服务器提出请求和处理响应，而步阻塞用户。达到无刷新的效果。

15.PHP中的session？

答：优点：session也是一种会话技术。

Session数据存在服务器，相对于比较安全。

服务器将session数据保存在服务器上，而将session的用户id存储在客户端电脑上。

Session存储的数据类型，除了资源外的数据类型都可以。

Session文件没有大小限制。

16.什么是json？

答：JavaScript对象表示法，是存储和交换文本信息的语法，轻量级的文本数据交换格式。

Json数据和普通的js对象的区别:json对象没有变量，json形式的数据结尾没有分号，json数据中的键必须用双引号包括。

17.jsonp的原理？

答：本质：动态创建script标签，然后通过它src属性发送跨域请求，然后服务器响应的数据格式为（函数调用（foo（实参））），所以在发送请求之前必须先声明一个函数，并且函数的名字与参数中传递的名字要一致，这里声明的函数是由服务器响应的内容（实际就是一段js代码—函数调用）来调用。

核心：动态添加<script>标签来调用服务器提供的js脚本。

18.跨域的几种方式？

答：1.src属性/href属性跨域。

2.jsonp🡪以上两种方法适用于get请求。

3．后台跨域设置：header（“access-control-allow-origin”，“\*”）；

4.nginx反向代理，设置跨域的方法和上面的一样。

5.浏览器跨域（适用于公司内部人员使用）。

19. Iframe的作用？

答：用法：iframe是用来在网页插入第三方页面，早期的页面使用iframe主要用于导航栏这种很多页面都相同的部分，这样可以在切换页面的时候避免重复下载。

优点：便于修改，模块分离，像一些信息管理系统会用到。但现在基本上不推荐使用，除非特殊需要，一般不推荐使用。

缺点：1.iframe的创建比一般的DOM元素慢乐1-2个数量级。

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

3.iframe对于SEO不友好，替代方案一般就是动态语言的incude机制的Ajax动态填充内容等。

**Nodejs**

1. nodejs的两种运行方式？

答：1》在终端中输入node文件名.js

2>REPL 在终端中输入node回车进入REPL环境开始输入js代码。

1. Node.js中JavaScript的组成？

答：ECMAScript 核心 + 全局成员 + 核心 API 模块

全局成员：console、setInterval、setTimeout

核心 API 模块：就是Node平台单独提供的一些API，这些API是Node平台所独有的

3.ECMAScript 6常用语法之let和const

答：let主要特性：

不存在变量提升问题，只有定义之后才能使用此变量

有{}作用域

Const主要特性：

不存在变量提升问题，只有定义之后才能使用此变量

Const定义的常量，无法在重新赋值

当定义常量的时候，必须定义且初始化，否则报语法错误

Const定义的常量，也有块级作用域

4.JavaScript的单线程和异步

答：JavaScript的解析和执行一直是单线程的，但是宿主环境（浏览器或node）是多线程的。

异步任务是由宿主环境开启子线程完成，并通过时间驱动、回调函数、队列，把完成的任务，交给主线程执行。

JavaScript解析引擎，一直在做一个工作，就是从任务队列里提取任务，放到主线程里执行。

5.模块作用域和全局作用域

答：全局作用域用global来访问，类似于浏览器中的window；

每个JavaScript文件，都是一个单独模块，每个模块都有自己独立的作用域，因此：模块中的成员，默认无法被其他模块访问。

Module属性是commonjs规范中定义的，它是一个对象，表示当前这个具体的js模块

Require（引用模块）每一个实现了commonjs规范的模块2，必须定义一个require（）函数，使用这个require函数，就能够很方便的导入其他模块中的成员，共自己使用。

Exports（暴露模块成员）每一个模块中，如果要把自己的私有成员，暴露给别人使用，那么，必须实现一个exports对向象，通过exports对象，可以方便的把模块内私有的成员，暴露给外界使用。

6.module.exports和exports的关系？

答：module.exports和exports默认引用了同一个空对象

Module.exports和exports作用一致，都可以向外暴露成员

Module.exports可以被赋值，而exports不可以

一个模块作用域中，向外暴露私有成员时，永远以module.exports为准

7.什么时包？

答：英文名叫做package，包是在模块基础上更深一步的抽象

包的目的：方便分布和推广基于commonjs规范实现的应用程序或类库

包可以看作是模块、代码和其他资源组合起来形成的独立作用域

8.规范包结构

答：1.包都要以一个单独的目录而存在；

2.package.json必须在包的顶层目录下；

3.package.json文件必须符合json格式，并且必须包含name：报的名字，version：包的版本号，main：表示包的入口文件这三个属性；

4.二进制文件应该在bin目录下；

5.JavaScript代码应该在lib目录下；

6.文档应该在doc目录下；

7.单元测试应该在test目录下；

8.node.js对包要求并没有那么严格，只要顶层目录下有package.json，并符合基本规范即可；

**Vue**

1. Node中MVC思想和前端中MVVM思想的区别？

答：MVVM是前端视图层的分层开发思想，主要把每个页面分成了M、V、和VM。其中，VM是MVVM思想的核心，因为VM是M和V之间的调度者。

M：这里的M保存的是每个页面中单独的数据；

VM：它是一个调度者，分割了M和V，每当V层想要获取后保存数据的时候，都要由VM做中间的处理；

V：就是每个页面中的HTML结构，视图层；

MVC是后端的分层概念，M数据模型与数据库访问CRUD；V前端视图，C控制器业务逻辑处理；

前端页面中使用MVVM的思想，主要是为了让我们开发更加方便，因为MVVM提供了数据源的双向绑定。

1. Vue是什么？

答：.js（读音 /vju?/，类似于 view） 是一套构建用户界面的渐进式框架，与其他重量级框架不同的是，Vue 采用自底向上增量开发的设计。Vue 的核心库只关注视

图层，它不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与单文件组件和 Vue 生态系统支持的库结合使用时，Vue 也完全能够为复杂的单页应用程序提供驱动。

1. Vue中的一些基本指令？

答：1.v-cloak：能够解决插值表达式闪烁的问题；

2.v-text就相当于在标签上将值已经绑定成功了，在显示出来，如果绑定不成功，也不显示这一块，也就是与{{}}的区别，{{}}在加载不出来时显示{{}},如果内容为标签时，则会把标签当成字符串渲染出来，而不会解析标签，所以使用v-html代替，但是v-html容易导致黑客攻击，所以只有在确保页面安全性的情况下，才能使用

3. v-html：都会替换标签内的内容，将数据类型解析成纯文本，相当于innerHTML，不会把标签渲染出来；

4. -bind绑定标签上的属性，最常用的是class，后面可以跟数组，对象，三元运算符直接绑定style:对象形式绑定，每次绑定时，在data中注册，或者在style中声明对象，注意font-size这类带-的，需要写成fontSize这样的写法；

5. v-model绑定数据源，实现了双项数据绑定，修饰符.lazy，当鼠标的焦点离开文本框才开始改变，.number，如果事按着12334sdg这样输入，就是12334,如果按着sfgh123则.number失效，按着原样输出，.tirm就是去空格。

6.v-once：v-once关联的实例，只会渲染一次。之后的重新渲染，实例极其所有的子节点将被视为静态内容跳过，这可以用于优化更新性能；

7.v-if：v-if可以实现条件渲染，Vue会根据表达式的值的真假条件来渲染元素。

8.v-else：v-else是搭配v-if使用的，它必须紧跟在v-if或者v-else-if后面，否则不起作用。

9.v-elseif： v-else-if充当v-if的else-if块，可以链式的使用多次。可以更加方便的实现switch语句。

10.v-on：v-on主要用来监听dom事件，以便执行一些代码块。表达式可以是一个方法名。  
简写为：【 **@** 】

11.v-for：根据循环遍历数组进行渲染。

4.事件修饰符？

答：1> .stop 阻止事件冒泡

2> .prevent 阻止事件默认行为

3> .capture 添加事件侦听器时实现捕获触发事件的机制

4> .self 实现只有点击当前元素的时候，才会触发事件处理函数

5> .once 只触发一次事件处理函数

注意：.self只会阻止自己身上冒泡行为的触发，并不会真正阻止冒泡的行为；

.capture事件有三个阶段，但是默认都是执行的是冒泡，设置事件执行捕获阶段，希望谁先执行就给谁。

1. 双向绑定？

答：v-model指令：可以实现表单元素和model中数据的双向数据绑定，智能运用在表单元素中；

Vue中双向数据绑定的原理就是object .defineProperty，参数1是对象，参数2是对象中需要定义的属性，参数3是该属性的一些配置。

5.v-if和v-show的区别？

答：v-if：每次都会重新删除或者创建元素，有较高的切换性能消耗，添加的元素会从页面移除/添加，如果和v-for使用，每一次循环都会判断一次，成立元素会在dom中结构渲染，值是false是删除这个元素.

v-show：每次不会重新进行dom的删除和创建操作，是指切换了元素的display：none属性，有较高的初始渲染消耗，控制的元素会在页面显示/隐藏。

一般如果是异步数据进行渲染的时候用v-if，操作大量的dom的时候用v-show。

**6.vuex设置全局的，有没有什么安全问题？**

答：有安全问题，用户这样可以在控制台中获取到Vuex中的数据(vue devTools插件可以直接查看vuex的数据)，那么在控制台就可以通过this.$vm.$store对象来修改store里的值，获取一些权限。

7. **vue中用什么发送异步请求**

答：基于Promise语法的axios.js或者vue-resourse.js

8. **ajax和axios的区别以及axios的好处**

答：JQuery整个项目太大，单纯使用ajax却要引入整个JQuery非常的不合理，基于原生XHR开发，XHR本身架构不清晰。ajax处理回调麻烦。

Axios只是一个单独的js文件，文件小，模块化开发思想

Axios基于Promise语法，提供了一些并发请求的接口，可以同时处理大量异步操作

Axios自动转换JSON数据

Axios在客户端支持防止CSRF/XSRF

9. **按需加载 ？**

答：就是模块化开的思想，以前我们要使用JQ中的AJAX,需要引入整个JQ文件，按需加载就是把特定的功能封装成一个模块，在哪个模块中需要使用就导入哪个模块

语法： import ... from ‘路径’

10. **如何创建自定义指令**

答：全局： Vue.directive(‘指令名称’, { 钩子函数(el, binding){}

局部： 组件中 directives{ ‘指令名称’ :{ 钩子函数(el, binding){} } }

11. **Vue中的jsonp怎么实现**

答：有点复杂，可以参考这个网址https://segmentfault.com/a/1190000008036838

1.生成script标签

2.注册callback函数

3.返回一个Promise

4.移除script标签

5.移除callback函数

如果不会自己动手写，就说用vue-resourse或者axios(有jsonp请求,需要手动安装一下，引用就行,npm i jsonp --save-dev 然后在页面中引用import jsonp from 'jsonp'，使用方式也是非常简单的)

12.生命周期和钩子函数？

答：定义：从VUE实例创建、运行到销毁期间，总是伴随这各种各样的事件，这些事件统称为生命周期。

钩子函数：beforeCreate:实例已经创建，还没有初始化好data和methods属性；

Created:初始化好了data和methods属性，还未开始编译模板（数据请求，放在此阶段较好）

beforeMount:此时已经完成了模板的编译，但是还没有挂载到页面中。

mounted:此时，已经将编译好的模板，挂载到了页面指定的容器中显示。

beforeUpdate: 视图层的数据改变前触发，即data数据变动，视图层还未改变时。

updated:事件执行的时候，页面和data数据已经保持同步了，都是最新的，界面已经被重新渲染好了。

beforeDestroy: 页面销毁前触发。

destroyed: 页面销毁后触发。

13. 父子传值?

答：父传子：是在子组件标签内通过v-bind属性绑定父组件内的值，在子组件中，通过props属性获取子组件的值,props的值是一个数组，里面的字符串元素就是绑定的属性名，在子组件内可以直接通过插值表达式获得父组件传过来的值

子传父：子组件内通过触发一个事件，通过函数内this.$emit(‘传给父组件内的事件名’,值)的方法,在父组件的子组件标签内设置@传过来的事件名=“父组件内的方法”,父组件的方法函数中，形参会接收从子组件传过来的参数

兄弟组件：通过新建一个中间的Vue实例newVue对象，要传值的组件内注册一个事件，通过 newVue.$emit(‘传递的事件名’,值)方法，然后再需要接收兄弟组件值得组件里，在mounted()钩子函数内设置newVue.$on(‘传递过来的事件名’, data=>this.xx=data),获取值

14. **路由传值**

答：①通过路由配置中的name属性，在组件中{{$route.name}}

②通过路由带参数进行传值

两个组件 A和B,A组件通过query把orderId传递给B组件（触发事件可以是点击事件、钩子函数等）

this.$router.push({ path: '/conponentsB', query: { orderId: 123 } }) // 跳转到B

在B组件中获取A组件传递过来的参数

this.$route.query.orderId

③通过<router-link>标签中的to传参

<router-link :to="{name:'hi1',params:{username:xxx}}">跳转到B页面</router-link> //A页面

{path:'/xxxxx',name:'hi1',component:组件名} //路由配置

$route.params.uesrname //B页面中接收A页面传过来的值

④利用url传递

{

    path:'/params/:newsId/:newsTitle',

     component:Params

} //路由配置 我们需要传递参数是新闻ID（newsId）和新闻标题（newsTitle）.所以我们在路由配置文件里制定了这两个值

在跳转后的页面使用$route.params.newsId 和 $route.params.newsTitle来获取通过url传递过来的值

15. **vue项目最后打包是什么指令？Vue打包完成的dist文件夹都有是么？**

答：npm run build 这个是在webpack中配置好的，你也可以自己定义，package.json中可以改名字

dist文件夹是默认的，可以自己修改路径和名称，在项目config文件下的index.js的build：{ } 这个属性里

16. **vue代码如何调试**

答：①通过vue-devtools这个插件

②移动端调试代码：

和电脑一样在同一个局域网下，输入地址就能看到效果了，iphone可以用safari来检查元素；android可以用chrome远程调试。

17. **什么是单页面应用程序？双页面应用程序？**

答：Vue就是典型的单页面应用程序，系统只加载一次资源，即CSS，JS只加载一次，在全局中加载后，其他组件都可以使用。后面的操作都是通过路由，异步请求来实现的

优点：用户体验好，内容改变不需要重新加载整个页面，

减轻服务器压力

一套代码，适用于PC，移动端

缺点：不利于SEO

初始加载慢

导航不可以用，要通过路由实现

双（多）页面应用程序就是我们常见的PC端的页面

18. **路由怎么切换**

答：①导入路由 import Router from 'vue-router'

②使用/注册：Vue.use(Router)

③配置路由

④把路由注入到根实例中(main.js里)

⑤在APP.vue组件中定义<router-link to=”要跳转的路径”>A页面</router-link>

<router-view></router-view> //对应的组件内容渲染到router-view中

19. Vue.js核心思想？

答：Vue.js是一个提供MVVM数据双向绑定的库，专注于UI层面，核心思想是：数据驱动、组件系统。

Vue.js数据观测原理在技术实现上，利用的是ES5Object.defineProperty和存储器属性: getter和setter（所以只兼容IE9及以上版本），可称为基于依赖收集的观测机制。核心是VM，即ViewModel，保证数据和视图的一致性。

20. template参数选项的有无对生命周期的影响？

答： 1. 如果Vue实例对象中有template参数选项，则将其作为模板编译成render函数

2. 如果没有template参数选项，则将外部的HTML作为模板编译（template），也就是说，template参数选项的优先级要比外部的HTML高

3. 如果1,2条件都不具备，则报错

21. 简述一些vue项目中文件构成？

答：build文件夹：主要就是webpack的配置；

Config文件夹：主要的就是index.js 这个文件进行配置代理服务器

Src文件夹：

“assets”共用的样式和图片

“components”业务代码存放

“router”路由

“APP.vue”vue 文件入口界面

“main.js”对应App.vue创建的实例，也是入口文件，对应webpack.base.config.js里的入口配置

Static文件夹：静态资源

Pack.json:scripts 里面设置命令，例如设置了dev用于调试则我们开发时输入的是npm run dev ；例如设置了build 则是输入 npm run build 用于打包;另一部分是这里可以看到我们需要的依赖包,在dependencies和devDependencies中，分别对应全局下载和局部下载的依赖包

22.

**其他问题**

1. 请谈谈你对性能优化的认识？

答：a 减少http请求次数

80%的响应事件花在下载网页内容（images、stylesheets、javascripts、script、flash等），减少请求次数是在缩短响应事件的关键，可以通过简化页面设计来减少请求次数，但页面内容较多可以采用以下技巧。

b 减少DNS查询次数（DNS域名系统）

DNS查询也消耗响应时间，如果我们的页面内容来自各个不同的domain（比如嵌入乐开放广告，引用了外部图片或脚本），那么客户端首次解析这些domain也需要消耗一定时间。DNS查询结果缓存在本地系统和浏览器中一段时间，所以DNS查询一般是对首次访问响应速度有所影响。

C 缓存Ajax

Ajax可以帮助我们异步的下载网页内容，但是有些网页内容即使是异步的，用户还是在等待它的返回结果，所以我们还是要注意精良应用延迟加载提高Ajax的响应速度。

D 延迟加载

这里讨论延迟加载需要我们知道我们的网页最初加载需要的最小内容集是什么。剩下的内容就可以堆到延迟加载的集合中。JavaScript是典型的可以延迟加载内容。一个比较激进的做法是开发网页时先确保网页在没有JavaScript的时候也可以基本工作，然后通过延迟加载脚本完成一些高级的功能。

1. 谈谈你对预加载的理解？

答：web预加载指的是在网页全加载完成之前，在页面优先显示一些主要内容，以提高用户体验。对于一个比较庞大的网站，如果没有使用预加载技术，用户界面就会长时间显示一片空白，直到资源加载完成，页面才会显示内容，用户体验很差。通过js预先从服务器加载图片资源（动态创建image，设置src属性），只要浏览器把图片下载到本地，就会被缓存，再次请求相当的src时就会优先寻找浏览器缓存，提高访问速度。

1. CDN是啥？

答：CDN的全称：ContentDeliveryNetwork，即内容分发网络，加速的意思，那么网站CDN服务就是网站加速服务，CDN加速原理：CDN加速将网站的内容缓存在网络边缘（离用户接入网络最近的地方），然后再用户访问网站内容的时候，通过调度系统将用户的请求路由或者引导到离用户接入网络最近或者访问效果最佳的缓存服务器上，有该缓存服务器为用户提高内容服务；相对于直接用户访问源站，这种方式缩短了用户和内容之间的网络距离，从而达到加速的效果。

CDN的特点：1.本地加速提高了站点（尤其含有大量图片和静态页面站点）的访问速度，并大大提高以上性质站点的稳定性。

2.镜像服务 消除了不同运营商之间互联的瓶颈造成的影响，实现了跨运营商的网络速度，保证不同网络中的用户都能得到良好的访问质量。

3.远程加速 远程访问用户根据DNS负载均衡技术，智能自动选择Cache服务器，选择最快的cache服务器，加快远程访问的速度。

4.带宽优化 自动生成服务器的远程Mirror（镜像）cache服务器，远程用户访问时从cache服务器上读取数据，减少远程访问的带宽、分担网络流量、减轻原站点web服务器负载等功能。

5.集群抗攻击 广泛分布的CDN节点加上节点之间的智能冗余机制，可以有效的预防黑客入侵以及降低各种DOS攻击对网站的影响，同时保证较好的服务质量。

4.ES5新增方法？

答：1.forEach（）

ForEach方法中的function回调支持3个参数，第一个时遍历的数组内容；第二个时对应的数组索引，第三个是数组本身。也就是：arr.forEach(value,index,arr){}。ForEach（）不仅可以接受一个回调函数，还可以接受一个可选的上下文参数（用来改变回调函数this的指向），不会遍历为空的数组。

代码：[1,2,3,4].forEach(function(i){alert(i)})

//等价于

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

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

alert(i)

}

2.map方法（同foreach方法类似）用法：[].map();

原数组被“映射”成对应新数组，再实际的使用，我们可以使用map方法方便的获取对象数组里面的特定的属性值，map的回调函数必须有返回值。

代码：var data = [1, 2, 3, 4];

var arrayOfSquares = data.map(function (item) {

return item \* item;

});

alert(arrayOfSquares); // 1, 4, 9, 16

3.filter过滤，刷选，filer过滤掉数组不需要的元素，返回过滤后的新数组，用法同map相同；

Array.filter(callback,[thisObject]);filter的回调函数返回boolean为true的值。

代码：var arr=[1,3,4,5,0];

var arr2=arr.filter(function(item){

return item;//为true则返回回去 })

//arr2=[1,3,4,5];

var arr2=arr.filter(function(item){

return item=="3"; })

//arr2=[3];

4.indexof用法：同字符串的indexof一样，arr.indexOf(str,index);str:为索引的内容，index为从第几页开始找（包括这一项）；兼容ie6-ie8；

代码：var arr=[1,2,3,4];

arr.indexOf(2,"x")//此时的“x”不是整数，被忽略。

arr.indexOf(4,2)//从索引是2的地方开始找，返回3，如果找不到就返回-1

5.lastindexof,同indexof一样

Arr.lastIndexOf(searchElement,Formindex);

从数组的末端开始找，Formindex默认为arr.length-1，一次递减。兼容ie6-ie8；

6.reduce递归

Array.reduce(callback[,initialValue])

Callback函数接受4个参数：之前值，当前值，索引值以及数组本身。InitialValue参数可选，表示初始值。若指定，则当作最初使用的previous值；如果缺省，则使用数组的第一个元素作为previous初始值，同时current往后排一位，相比有initialValue值少一次迭代。

因为initialvalue不存在，因此一开始的previous值等于数组的第一个元素。

从而current值再第一次调用的时候就是2

最后两个参数为索引值index以及数组本身array

代码：var sum = [1, 2, 3, 4].reduce(function (previous, current, index, array) {

return previous + current;

});

console.log(sum); // 10