Web 前端与移动开发面试宝典

Web 前端与移动开发面试宝典

一、HTML+CSS 部分（ ）

1、怎么让一个不定宽高的 DIV，垂直水平居中？

答：1）**使用 CSS 方法：**

父盒子设置：display：table-cell； text-align：center；vertical-align：middle；

Div 设置： display：inline-block；vertical-align：middle；

2）**使用 CSS3 transform：**

父盒子设置：display：relative

Div 设置： transform： translate(-50%，-50%)；position： absolute；top： 50%；left： 50%；

2、position 几个属性的作用？

答：position 的常见四个属性值： relative，absolute，fixed，static。一般都要配合"left"、"top"、"right" 以及 "bottom" 属性使用。

1）Static：默认位置，设置为 static 的元素，它始终会处于页面流给予的位置（static 元素会忽略任何 top、bottom、left 或 right 声明）。一般不常用。

2）Relative：位置被设置为 relative 的元素，可将其移至相对于其正常位置的地方，意思就是如果设置了 relative 值，那么，它偏移的 top，right，bottom，left 的值都以它原来的位置为基准偏移，而不管其他元素会怎么样。注意 relative 移动后的元素在原来的位置仍占据空间。

3）Absolute：位置设置为 absolute 的元素，可定位于相对于包含它的元素的指定坐标。意思就是如果它的父容器设置了 position 属性，并且 position 的属性值为 absolute 或者 relative，那么就会依据父容器进行偏移。如

果其父容器没有设置 position 属性，那么偏移是以 body 为依据。注意设置 absolute 属性的元素在标准流中不占位置。

4）Fixed：位置被设置为 fixed 的元素，可定位于相对于浏览器窗口的指定坐标。不论窗口滚动与否，元素都会留在那个位置。它始终是以 body 为依据的。 注意设置 fixed 属性的元素在标准流中不占位置。

3、px，em，rem 的区别？

答： 1）px 像素（Pixel）。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长度单位，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，相对于当前对象内文本的字体尺寸。如当前对行内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（root em，根 em），使用 rem 为元素设定字体大小时，仍然是相对大小，但相对的只是 HTML 根元素。

4）区别：IE 无法调整那些使用 px 作为单位的字体大小，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身，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。

4、什么是 BFC？

答： 1）定义：

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。

3）哪些元素会生成 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

5、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 是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。

6、box-sizing、transition、translate 分别是什么？

答：Box-sizing： 用来指定盒模型的大小的计算方式。主要分为 boreder-box（从边框固定盒子大小）、content-box（从内容固定盒子大小）两种计算方式。

transition： 当前元素只要有“属性”发生变化时，可以平滑的进行过渡。通过 transtion-propety 设置过渡属性；transtion-duration 设置过渡时间；trantion-timing-function 设置过渡速度；trantion-delay 设置过渡延时

translate：通过移动改变元素的位置；有 x、y、z 三个属性

7、选择器优先级是怎样的？

答：! important>行内样式>id 选择器>类选择器>标签选择器>通配符>继承

权重算法：

（0，0，0，0）==》第一个 0 对应的是 important 的个数，第二个 0 对应的是 id 选择器的个数，第三个 0 对应的类选择器的个数，第四个 0 对应的是标签选择器的个数，就是当前选择器的权重。

比较：

先从第一个 0 开始比较，如果第一个 0 大，那么说明这个选择器的权重高，如果第一个相同，比较第二个，依次类推

8、CSS3 选择器有哪些？

答：属性选择器、伪类选择器、伪元素选择器。

9、Iframe 的作用？

答：用法：

Iframe 是用来在网页中插入第三方页面，早期的页面使用 iframe 主要是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 元素慢了 1-2 个数量级

（2）iframe 标签会阻塞页面的加载，如果页面的 onload 事件不能及时触发，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 机制和 ajax 动态填充内容等.

10、有一个导航栏在 chrome 里面样式完好？在 IE 里文字都聚到一起了，是哪个兼容性问题？

答：用了 display：flex 属性，在 ie10 以下都是无效的。

11、CSS3 新特性有哪些？

答：1. 颜色：新增 RGBA，HSLA 模式
2. 文字阴影（text-shadow、）
3.边框： 圆角（border-radius）边框阴影： box-shadow
4. 盒子模型：box-sizing
5.背景：background-size 设置背景图片的尺寸 background-origin 设置背景图片的原点
background-clip 设置背景图片的裁切区域，以”，”分隔可以设置多背景，用于自适应布局
6.渐变：linear-gradient、radial-gradient
7. 过渡：transition，可实现动画
8. 自定义动画
9. 在 CSS3 中唯一引入的伪元素是 ::selection.
10. 媒体查询，多栏布局
11. border-image
12.2D 转换：transform：translate(x，y) rotate(x，y) skew(x，y) scale(x，y)
13. 3D 转换

12、xhtml 和 html 有什么区别？

答：HTML 是一种基本的 WEB 网页设计语言，XHTML 是一个基于 XML 的置标语言
最主要的不同：
XHTML 元素必须被正确地嵌套。
XHTML 元素必须被关闭。
标签名必须用小写字母。

XHTML 文档必须拥有根元素。

13、CSS 引入的方式有哪些? link 和@import 的区别是?

答：内联 内嵌 外链 导入

区别 ：同时加载

前者无兼容性，后者 CSS2.1 以下浏览器不支持

Link 支持使用 javascript 改变样式，后者不可

14、标签上 title 与 alt 属性的区别是什么?

答：Alt 当图片不显示时，用文字代表。

Title 为该属性提供信息。

15、描述 css reset 的作用和用途？

答：Reset 重置浏览器的 css 默认属性 ，览器的品种不同，样式不同，然后重置，让他们统一。

16、解释 css sprites，如何使用？

答：Css 精灵图，把一堆小的图片整合到一张大的图片（png）上，减轻服务器对图片的请求数量。

17、清除浮动的几种方式？

答：1，父级 div 定义 height

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.

2，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 .

3，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 转有属性)可解决 ie6，ie7 浮动问题 ，推荐使用，建议定义公共类，以减少 CSS 代码。

4，父级 div 定义 overflow：hidden

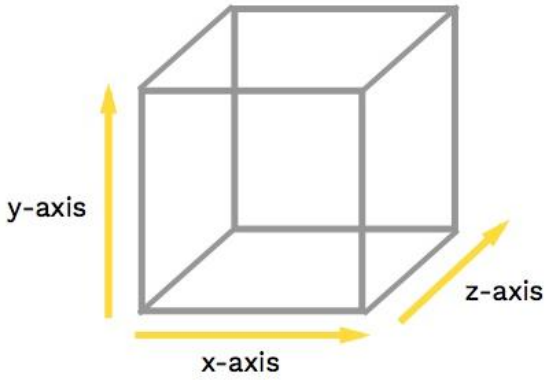
超出盒子部分会被隐藏，不推荐使用.

5. 双伪元素法：

```
    .clearfix: before, .clearfix: after {
              content:  "";
              display:  block;
              clear:  both;
          }
          .clearfix {
              zoom:  1;
          }
```

18、z-index 说说 z-index 的工作原理及适用范围?

答：原理：



z-index 这个属性控制着元素在 z 轴上的表现形式。

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（Stacking Level）

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它所在的文档树。默认情况下，文档中后来声明的元素具有更高的堆叠顺序。当父元素的堆叠顺序被设置的时候，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：

1. 网页两侧浮动窗口(播放器，置顶按钮，浮动广告，功能按钮等)
2. 导航栏浮动置顶。
3. 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)

19、能否简述一下如何使一套设计方案，适应不同的分辨率，有哪些方法可以实现？

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：

那么 Ethan Marcotte 在 2010 年 5 月份提出的一个概念，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（媒介查询），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些，一般使用第三方响应式框架来完成，比如 bootstrap 来完成一部分工作，当然也可以自己写响应式。

阐述下移动 web 和响应式的区别：

开发方式	移动web开发+PC开发	响应式开发
应用场景	一般在已经有PC端的网站，开发移动站的时候，只需单独开发移动端。	针对新建站的一些网站，现在要求适配移动端，所以就一套页面兼容各种终端，灵活。
开发	针对性强，开发效率高。	兼容各种终端，效率低，
适配	只适配 移动设备，pad上体验相对较差。	可以适配各种终端
效率	代码简洁，加载快。	代码相对复杂，加载慢。

20、你能描述一下渐进增强和优雅降级之间的不同吗？

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。

举个例子：

```
a{
    display:  block;
    width:  200px;
    height:  100px;
    background: aquamarine;
    /*我就是要用这个新 css 属性*/
     transition:  all 1s ease 0s;
   /*可是发现了一些低版本浏览器不支持怎么吧*/
    /*往下兼容*/
    -webkit-transition: all 1s ease 0s;
    -moz-transition: all 1s ease 0s;
    -o-transition:  all 1s ease 0s;
   /*那么通常这样考虑的和这样的侧重点出发的 css 就是优雅降级*/
}
a: hover{
    height:  200px;
}

/  *那如果我们的产品要求我们要重低版本的浏览器兼容开始*/
  a{
 /*优先考虑低版本的*/
```

```
    -webkit-transition: all 1s ease 0s;
    -moz-transition: all 1s ease 0s;
    -o-transition:  all 1s ease 0s;
    /*高版本的就肯定是渐进渐强*/
   transition:  all 1s ease 0s;
}
```

“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。

21、改变元素的外边距用什么属性？改变元素的内填充用什么属性？

答：改变元素的外边距用 margin，改变元素的内填充用 padding。

22、在新窗口打开链接的方法是？

答：target：_blank。

23、合理的页面布局中常听过结构与表现分离，那么结构是什么？表现是什么？

答：结构是 html，表现是 css。

24、简述对 Web 语义化的理解？

答：就是让浏览器更好的读懂你写的代码，在进行 HTML 结构、表现、行为设计时，尽量使用语义化的标签，使程序代码简介明了，易于进行 Web 操作和网站 SEO，方便团队协作的一种标准，以图实现一种“无障碍”的 Web 开发。

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

答：DOCTYPE 是一种标准通用标记语言的文档类型声明，它的目的是要告诉标准通用标记语言解析器，它应该使用什么样的文档类型定义来解析文档。只有确定了一个正确的文档类型，超文本标记语言或可扩展超文本标记语言中的标签和层叠样式表才能生效，甚至对 javascript 脚本都会有所影响。

26、CSS 都有哪些选择器？CSS 选择器的优先级是怎么样定义的？

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

！important >行内样式> ID > 类 > 标签 | 伪类 | 属性选择 > 伪对象 > 继承 > 通配符。

27、display： none；与 visibility： hidden 的区别是什么？

答：display：none； 使用该属性后，HTML 元素（对象）的宽度、高度等各种属性值都将“丢失”；

visibility：hidden； 使用该属性后，HTML 元素（对象）仅仅是在视觉上看不见（完全透明），而它所占据的空间位置仍然存在，也即是说它仍具有高度、宽度等属性值。

28、请用 CSS 定义 p 标签 要求实现以下效果 :字体颜色在 IE6 下为黑色(#000000) ;IE7 下为红色(#ff0000)；而其他浏览器下为绿色(#00ff00)

```
答：p{
    color: green;
    *color: blue;
    _color: black;
}
```

二、JS 部分

1、AJAX 请求数据步骤是什么？传输的数据是用的暗文还是明文？

答：

```
   var xhr;
xhr = new XMLHttpRequest();  //创建一个异步对象
xhr.open("Get", "test.ashx", true); //Get 方式括号中的三个参数分别为：1.发送请求的方式 2.样请求的页面 3.是否异步
//xhr.open("post", "test.ashx", true);
//xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded"); Post 方式发送数据

//这个回调函数主要用来检测服务器是否把数据返回给异步对象
xhr.setRequestHeader("If-Modified-Since", "0"); //设置浏览器不使用缓存
xhr.onreadystatechange = function () {
if (xhr.readystate == 4) {
//readyState 属性指出了 XMLHttpRequest 对象在发送/接收数据过程中所处的几个状态。XMLHttpRequest 对象会经历 5 种不同的状态。
```

```
//0：未初始化。对象已经创建，但还未初始化，即还没调用 open 方法；
//1：已打开。对象已经创建并初始化，但还未调用 send 方法；
//2：已发送。已经调用 send 方法，但该对象正在等待状态码和头的返回；
//3：正在接收。已经接收了部分数据，但还不能使用该对象的属性和方法，因为状态和响应头不完整；
//4：已加载。所有数据接收完毕
 if(xhr.status==200){ //检测服务器返回的响应报文的状态码是否为 200
    alert(xhr.responseText); //服务器返回的 Response 数据
    //解析服务器返回的 jason 格式的数据
    var s=xhr.responseText;
    var json=eval("("+s+")");
    alert(jason.data);
    }
    };
};
 xhr.send(null); //异步对象发送请求
//xhr.send("txtName=roger&txtPwd=123");  以 post 方式发送数据

    ajax 中 get 和 post 方式请求数据都是明文的。
```

2、怎么实现跨域问题？

答：

URL	说明
http://www.a.com/b.js	同一域名下
http://www.a.com:8000/a.js	同一域名，不同端口
https://www.a.com/b.js	同一域名，不同协议
http://script.a.com/b.js	主域相同，子域不同
http://a.com/b.js	同一域名，不同二级域名
http://www.a.com/b.js	不同域名

对于端口和协议的不同，只能通过后台来解决。我们要解决的是域名不同的问题。

1. 下面是用 php 进行的设置，“*”号表示允许任何域向我们的服务端提交请求：header{"Access-Control-Allow-Origin： *"}

2.JSONP(JSON 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时，但是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。

```
<script type="text/javascript">
    function dosomething(jsondata){
        //处理获得的json数据
    }
</script>
<script src="http://example.com/data.php?callback=dosomething"></script>
```

首先第一个 script 便签定义了一个处理数据的函数；

然后第二个 script 标签载入一个 js 文件，http：//example.com/data.php 是数据所在地址，但是因为是当做 js 来引入的，所以 http：//example.com/data.php 返回的必须是一个能执行的 js 文件；

最后 js 文件载入成功后会执行我们在 url 参数中指定的函数，并且会把我们需要的 json 数据作为参数传入。所以 php 应该是这样的：

```
<?php
$callback = $_GET['callback'];//得到回调函数名
$data = array('a','b','c');//要返回的数据
echo $callback.'('.json_encode($data).')';//输出
?>
```

JSONP 的优缺点

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，一定得保证它安全可靠。

它只支持 GET 请求而不支持 POST 等其它类型的 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、谈谈 js 作用域和闭包？

答：简单的说，作用域是针对变量的，比如我们创建一个函数 a1，函数里面又包了一个子函数 a2。此时就存在三个作用域：

全局作用域 - a1 作用域 - a2 作用域；即全局作用域包含了 a1 的作用域，a2 的作用域包含了 a1 的作用域。

当 a1 在查找变量的时候会先从自身的作用域区查找，找不到再到上一级 a2 的作用域查找，如果还没找到就到全局作用域区查找，这样就形成了一个作用域链。

理解闭包首先要理解，js 垃圾回收机制，也就是当一个函数被执行完后，其作用域会被收回，如果形成了闭

包，执行完后其作用域就不会被收回。

如果某个函数被他的父函数之外的一个变量引用，就会形成闭包。

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。

4、什么是原型链？

答：Javascript 是面向对象的，每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，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，直到查到 Object.prototype._proto_为 nul，这样也就形成了原型链。

5、实现继承的方法有什么？

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。

```
function SuperType(){
    this.colors = ["red", "blue", "green"];
}

function SubType(){
    //继承了 SuperType
    SuperType.call(this);
}

var instance1 = new SubType();
instance1.colors.push("black");
alert(instance1.colors);    //"red,blue,green,black"

var instance2 = new SubType();
alert(instance2.colors);    //"red,blue,green"
```

（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。

```
Function object(o){
Function  F(){};
F.prototype=o;
Return new F();
}
```

（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。

```
Function createAonter(original){
Var clone=object(original); //通过调用函数创建一个新对象
Clone.sayHi=function(){   //以某种方式来增强这个对象
Alert(“hi”);
}
Return clone;  //返回这个对象
}
```

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二降低效率，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。

6、什么是事件冒泡/捕获？

答：

事件冒泡：子元素事件的触发会影响父元素事件；

开关事件冒泡：

A，开启事件冒泡：element.addEventListener(eventName，handler，false)；

B，关闭事件冒泡：假设传统方式事件的返回值为 e，就可以通过 e.stopPropagation()来关闭事件冒泡；

事件捕获：父元素的事件会影响子元素的事件；

开启事件捕获：element.addEventListener(eventName，handler，true)

7、请说说事件委托机制？这样做有什么好处？

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。

好处：

A，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。

B，新添加的元素还会有之前的事件；

8、请列举字符串操作的方法？

charCodeAt 方法返回一个整数，代表指定位置字符的 Unicode 编码；

charAt 方法返回指定索引位置处的字符。如果超出有效范围的索引值返回空字符串；

slice 方法返回字符串的片段；

substring 方法返回位于 String 对象中指定位置的子字符串。

substr 方法返回一个从指定位置开始的指定长度的子字符串。

indexOf 方法返回 String 对象内第一次出现子字符串位置。如果没有找到子字符串，则返回-1；

lastIndexOf 方法返回 String 对象中字符串最后出现的位置。如果没有匹配到子字符串，则返回-1；

search 方法返回与正则表达式查找内容匹配的第一个字符串的位置。

concat 方法返回字符串值，该值包含了两个或多个提供的字符串的连接；

split 将一个字符串分割为子字符串，然后将结果作为字符串数组返回；

9、请说出==和===的区别？

答：== 判断内容是否相等 不比较类型

```
console.log(1 == "1");  true
```

=== 判断内容相等 且类型也相等

```
console.log(1 ==="1");  false
```

10、如何判断一个数组是数组？

```
1.[] instanceof Array   true/false
2. arr.constructor == Array   true/false
3. ES5：Array.isArray()   true/false
4. Object.prototype.toString.call([])==="[object Array]"true/false
```

```
    5.目前完美方法
function isArray(value){
if (typeof Array.isArray === "function") {
return Array.isArray(value);
    }else{
return Object.prototype.toString.call(value) === "[object Array]";
    }
}
```

11、怎么理解 jQuery？

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 在 2006 年 1 月由美国人 John Resig 在纽约的 barcamp 发布，吸引了来自世界各地的众多 JavaScript 高手加入，由 Dave Methvin 率领团队进行开发。如今，jQuery 已经成为最流行的 javascript 库，在世界前 10000 个访问最多的网站中，有超过 55%在使用 jQuery。

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。**

jQuery，顾名思义，也就是 JavaScript 和查询（Query），即是辅助 JavaScript 开发的库。

12、Jquery .on 这个方法怎么看？

```
    答:
    jQuery.on()方法是官方推荐的绑定事件的一个方法。
    $(selector).on(event, childSelector, data, function, map)
    多个事件绑定同一个函数
    $(document).ready(function(){
      $("p").on("mouseover mouseout", function(){
        $("p").toggleClass("intro");
      });
    });
    多个事件绑定不同函数
    $(document).ready(function(){
```

```
  $("p").on({
    mouseover: function(){$("body").css("background-color", "lightgray"); },
    mouseout: function(){$("body").css("background-color", "lightblue"); },
    click: function(){$("body").css("background-color", "yellow"); }
  });
});
```

绑定自定义事件

```
$(document).ready(function(){
  $("p").on("myOwnEvent", function(event, showName){
    $(this).text(showName + "! What a beautiful name!").show();
  });
  $("button").click(function(){
    $("p").trigger("myOwnEvent", ["Anja"]);
  });
});
```

传递数据到函数

```
function handlerName(event)
{
  alert(event.data.msg);
}

$(document).ready(function(){
  $("p").on("click", {msg: "You just clicked me!"}, handlerName)
});
```

适用于未创建的元素

```
$(document).ready(function(){
  $("div").on("click", "p", function(){
    $(this).slideToggle();
  });
  $("button").click(function(){
    $("<p>This is a new paragraph.</p>").insertAfter("button");
  });
})
```

13、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吗？

答：概述：

GET 是从服务器上请求数据，POST 是发送数据到服务器。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 方法可以没有时间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的（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 方法会将所要传输的数据附在网址后面，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：

1、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；

2、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：

表达如果是向服务器传输数据(如帐号密码等)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)

14、面向对象和类的区别？

答：简单的说类是对象的模版。

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。

面向对象有三大特性

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

对象的概念，面向对象编程的程序实际就是多个对象的集合，我们可以把所有的事物都抽象成对象，在程序设计中可以看作：对象=属性+方法。属性就是对象的数据，而方法就是对象的行为。

类的概念，类是对象的模版，而对象是类的实例化。举个例子，汽车设计图可以看作是类，而具体的汽车就是对象。再比如有一个类是表示人，然后可以通过人这个模版来实例化出张三、李四。。。

15、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。

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。

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， 知道上一个执行完

16、如何查找构造函数和原型中的属性？

答：构造函数.prototype 查看构造函数的原型属性

实例对象.__proto__ 查看实例对象的构造函数的原型

实例对象.__proto__.constructor 查看实例对象的构造函数

17、js 中一共有几种数据类型？

答：Undefined、Null、Boolean、Number 和 String。

还有一种复杂的数据类型 Object，Object 本质是一组无序的名值对组成的。

18、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。”

它们的不同之处：

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。

更简单地说，apply 和 call 功能一样，只是传入的参数列表形式不同：如 func.call(func1，var1，var2，var3)

对应的 apply 写法为：func.apply(func1，[var1，var2，var3])。

19、说说你对 this 的理解？

答：this 是一个关键字，它代表函数运行时，自动生成的一个内部对象，只能在函数内部使用。

1.作为纯粹的函数调用 this 指向全局对象

2.作为对象的方法调用 this 指向调用对象

3.作为构造函数被调用 this 指向新的对象（new 会改变 this 的指向）

4.apply 调用 this 指向 apply 方法的第一个参数

20、谈谈你对递归的认识？

答：递归：程序调用自身的编程技巧称为递归，自己调用自己。

```
function factorial(num) {
    return ( num <= 1 ) ? 1 :  num * factorial(num - 1);
}
    console.log(factorial(8));
```

21、js 的异步加载有哪几种方法？

答 :方案一 :<script>标签的 async="async"属性。HTML5 中新增的属性 ,Chrome、FF、IE9&IE9+均支持(IE6~8 不支持)。此外，这种方法不能保证脚本按顺序执行。

方案二：<script>标签的 defer="defer"属性。兼容所有浏览器。此外，这种方法可以确保所有设置 defer 属性的脚本按顺序执行。

方案三：AJAX　eval（使用 AJAX 得到脚本内容，然后通过 eval_r(xmlhttp.responseText)来运行脚本)。兼容所有浏览器。

方案四：iframe 方式（这里可以参照：iframe 异步加载技术及性能中关于 Meboo 的部分)。兼容所有浏览器。

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

答：　对象：Window document location screen history navigator

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

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

答： document.getElementById 根据元素 id 查找元素

document.getElementByName 根据元素 name 查找元素

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

24、原生 JS 的 window.onload 与 Jquery 的$(document).ready(function () {})，$(function () {})有什么不同？

答：

1.执行时间

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

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

2.编写个数不同

window.onload 不能同时编写多个，如果有多个 window.onload 方法，只会执行一个

$(document).ready()可以同时编写多个，并且都可以得到执行

3.简化写法

window.onload 没有简化写法

$(document).ready(function(){})可以简写成$(function(){})；

25、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； 返回的是 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；

26、简述 for in 循环的特点及使用场景？

答：for...in 语句用于对数组或者对象的属性进行循环操作。

```
for ... in 循环中的代码每执行一次，就会对数组的元素或者对象的属性进行一次操作。
for (变量 in 对象){
    在此执行代码
```

```
}
```

“变量”用来指定变量，指定的变量可以是数组元素，也可以是对象的属性。

注意：for in 循环不会按照属性的下标来排列输出。

27、Javascript 内置的常用对象有哪些？并列举该对象常用的方法？

答：

Arguments 函数参数集合

arguments[ ] 函数参数的数组

Arguments 一个函数的参数和其他属性

Arguments.callee 当前正在运行的函数

Arguments.length 传递给函数的参数的个数

Array 数组

length 属性：动态获取数组长度

join()：将一个数组转成字符串。返回一个字符串。

reverse()：将数组中各元素颠倒顺序

delete 运算符：只能删除数组元素的值，而所占空间还在，总长度没变(arr.length)。

shift()：删除数组中第一个元素，返回删除的那个值，并将长度减 1。

pop()：删除数组中最后一个元素，返回删除的那个值，并将长度减 1。

unshift()：往数组前面添加一个或多个数组元素，长度要改变。arrObj.unshift(“a” , “b，“c”)

push()：往数组结尾添加一个或多个数组元素，长度要改变。arrObj.push(“a” , “b” , “c”)

concat( ) 连接数组

slice( ) 返回数组的一部分

sort( ) 对数组元素进行排序

splice( ) 插入、删除或替换数组的元素

toLocaleString( ) 把数组转换成局部字符串

toString( ) 将数组转换成一个字符串

Boolean 布尔对象

Boolean.toString( ) 将布尔值转换成字符串

Boolean.valueOf( ) Boolean 对象的布尔值

Date 日期时间

创建 Date 对象的方法

（1）创建当前(现在)日期对象的实例，不带任何参数

var today = new Date()；

（2）创建指定时间戳的日期对象实例，参数是时间戳。

时间戳：是指某一个时间距离 1970 年 1 月 1 日 0 时 0 分 0 秒，过去了多少毫秒值(1 秒=1000 毫秒)。

var timer = new Date(10000)； //时间是 1970 年 1 月 1 日 0 时 0 分 10 秒

（3）指定一个字符串的日期时间信息，参数是一个日期时间字符串

var timer = new Date(“2015/5/25 10：00：00”)；

（4）指定多个数值参数

var timer = new Date(2015+100，4，25，10，20，0)； //顺序为：年、月、日、时、分、秒，年、月、日是必须的。

方法：

Date.getDate( ) 返回一个月中的某一天

Date.getDay( ) 返回一周中的某一天

Date.getFullYear( ) 返回 Date 对象的年份字段

Date.getHours( ) 返回 Date 对象的小时字段

Date.getMilliseconds( ) 返回 Date 对象的毫秒字段

Date.getMinutes( ) 返回 Date 对象的分钟字段

Date.getMonth( ) 返回 Date 对象的月份字段

Date.getSeconds( ) 返回 Date 对象的秒字段

Date.getTime( ) 返回 Date 对象的毫秒表示

Date.getTimezoneOffset( ) 判断与 GMT 的时间差

Date.getUTCDate( ) 返回该天是一个月的哪一天(世界时)

Date.getUTCDay( ) 返回该天是星期几(世界时)

Date.getUTCFullYear( ) 返回年份(世界时)

Date.getUTCHours( ) 返回 Date 对象的小时字段(世界时)

Date.getUTCMilliseconds( ) 返回 Date 对象的毫秒字段(世界时)

Date.getUTCMinutes( ) 返回 Date 对象的分钟字段(世界时)

Date.getUTCMonth( ) 返回 Date 对象的月份(世界时)

Date.getUTCSeconds( ) 返回 Date 对象的秒字段(世界时)

Date.getYear( ) 返回 Date 对象的年份字段(世界时)

Date.parse( ) 解析日期/时间字符串

Date.setDate( ) 设置一个月的某一天

Date.setFullYear( ) 设置年份，也可以设置月份和天

Date.setHours( ) 设置 Date 对象的小时字段、分钟字段、秒字段和毫秒字段

Date.setMilliseconds( ) 设置 Date 对象的毫秒字段

Date.setMinutes( ) 设置 Date 对象的分钟字段和秒字段

Date.setMonth( ) 设置 Date 对象的月份字段和天字段

Date.setSeconds( ) 设置 Date 对象的秒字段和毫秒字段

Date.setTime( ) 以毫秒设置 Date 对象

Date.setUTCDate( ) 设置一个月中的某一天(世界时)

Date.setUTCFullYear( ) 设置年份、月份和天(世界时)

Date.setUTCHours( ) 设置 Date 对象的小时字段、分钟字段、秒字段和毫秒字段(世界时)

Date.setUTCMilliseconds( ) 设置 Date 对象的毫秒字段(世界时)

Date.setUTCMinutes( ) 设置 Date 对象的分钟字段和秒字段(世界时)

Date.setUTCMonth( ) 设置 Date 对象的月份字段和天数字段(世界时)

Date.setUTCSeconds( ) 设置 Date 对象的秒字段和毫秒字段(世界时)

Date.setYear( ) 设置 Date 对象的年份字段

Date.toDateString( ) 返回 Date 对象日期部分作为字符串

Date.toGMTString( ) 将 Date 转换为世界时字符串

Date.toLocaleDateString( ) 回 Date 对象的日期部分作为本地已格式化的字符串

Date.toLocaleString( ) 将 Date 转换为本地已格式化的字符串

Date.toLocaleTimeString( ) 返回 Date 对象的时间部分作为本地已格式化的字符串

Date.toString( ) 将 Date 转换为字符串

Date.toTimeString( ) 返回 Date 对象日期部分作为字符串

Date.toUTCString( ) 将 Date 转换为字符串(世界时)

Date.UTC( ) 将 Date 规范转换成毫秒数

Date.valueOf( ) 将 Date 转换成毫秒表示

Error 异常对象

Error.message 可以读取的错误消息

Error.name 错误的类型

Error.toString( ) 把 Error 对象转换成字符串

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

RangeError 在数字超出合法范围时抛出

ReferenceError 在读取不存在的变量时抛出

TypeError 当一个值的类型错误时，抛出该异常

URIError 由 URl 的编码和解码方法抛出

Function 函数构造器

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

Math 数学对象

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。

Number 数值对象

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( ) 将一个数字转换成字符串

Number.valueOf( ) 返回原始数值

Object 基础对象

Object 含有所有 JavaScript 对象的特性的超类

Object.constructor 对象的构造函数

Object.hasOwnProperty( ) 检查属性是否被继承

Object.isPrototypeOf( ) 一个对象是否是另一个对象的原型

Object.propertyIsEnumerable( ) 是否可以通过 for/in 循环看到属性

Object.toLocaleString( ) 返回对象的本地字符串表示

Object.toString( ) 定义一个对象的字符串表示

Object.valueOf( ) 指定对象的原始值

RegExp 正则表达式对象

RegExp.exec( ) 通用的匹配模式

RegExp.global 正则表达式是否全局匹配

RegExp.ignoreCase 正则表达式是否区分大小写

RegExp.lastIndex 下次匹配的起始位置

RegExp.source 正则表达式的文本

RegExp.test( ) 检测一个字符串是否匹配某个模式

RegExp.toString( ) 把正则表达式转换成字符串

String 字符串对象

length：获取字符串的长度。如：var len = strObj.length

toLowerCase()：将字符串中的字母转成全小写。如：strObj.toLowerCase()

toUpperCase()：将字符串中的字母转成全大写。如：strObj.toUpperCase()

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。

区别：'"

"abcdefgh" .substring(2，3) = "c"

"abcdefgh" .substr(2，3) = "cde"

'"

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( ) 从字符编码创建一个字符串

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

28、split() join()的区别？

答：split() 方法通过把字符串分割成子字符串来把一个 String 对象分割成一个字符串数组。

语法

```
str.split([separator][, limit])
```

参数 separator 指定用来分割字符串的字符（串）。separator 可以是一个字符串或正则表达式。 如果忽略 separator，则返回整个字符串的数组形式。如果 separator 是一个空字符串，则 str 将会把原字符串中每个字符的数组形式返回。

参数 limit 是一个整数，限定返回的分割片段数量。split 方法仍然分割每一个匹配的 separator，但是返回的数组只会截取最多 limit 个元素。

例子：

```
"2: 3: 4: 5".split(": ") //将返回["2", "3", "4", "5"]
"|a|b|c".split("|") //将返回["", "a", "b", "c"]
"hello".split("")   //可返回 ["h", "e", "l", "l", "o"]
"hello".split("", 3)   //可返回 ["h", "e", "l"]
```

join() 方法将数组中的所有元素连接成一个字符串。

语法

```
str = arr.join([separator = ', '])
```

参数 separator 可选，用于指定连接每个数组元素的分隔符。分隔符会被转成字符串类型；如果省略的话，默认为一个逗号。如果 seprator 是一个空字符串，那么数组中的所有元素将被直接连接。

例子：

```
var a = ['Wind', 'Rain', 'Fire'];
var myVar1 = a.join();      // myVar1 的值变为"Wind，Rain，Fire"
var myVar2 = a.join(', ');   // myVar2 的值变为"Wind， Rain， Fire"
var myVar3 = a.join(' + '); // myVar3 的值变为"Wind + Rain + Fire"
var myVar4 = a.join('');     // myVar4 的值变为"WindRainFire"
```

29、例举 3 中强制类型转换和 2 中隐式类型转换？

答：强制

转化成字符串 toString() String()

转换成数字 Number()、 parseInt()、 parseFloat()

转换成布尔类型 Boolean()

隐式

拼接字符串 例子 var str = "" + 18

- / % ===

30、Html5 重要的新的表单元素有哪些(至少举例 5 个)？

答：1 输入类型

email 输入 email 格式

tel 手机号码

url 只能输入 url 格式

number 只能输入数字

search 搜索框

range 范围

color 拾色器

time 时间

date 日期 不是绝对的

datetime 时间日期

month 月份

week 星期

部分类型是针对移动设备生效的，且具有一定的兼容性，在实际应用当中可选择性的使用。

2 表单元素（标签）

<datalist> 下拉选项，使用中文时要注意

<keygen> 生成加密字符串

<output> 不可当做数据提交？

<meter> 表示度量器，不建议用作进度条

3 表单属性

placeholder 占位符

autofocus 获取焦点

multiple 文件上传多选或多个邮箱地址

autocomplete 自动完成，用于 form 元素，也可用于部分 input，默认值 on

form 指定表单项属于哪个 form，处理复杂表单时会需要

novalidate 关闭验证，可用于<form>标签，(只适应用 form)

required 验证条件，必填项

pattern 正则表达式 自定义验证规则

表单重写没有提及，自行验证，共包含

formaction、formenctype、formtarget、formmethod、formnovalidate

应用于提交按钮上，如：<input type="submit" formaction="xxx.php">

4 表单事件

oninput 用户输入内容时触发，可用于移动端输入字数统计

oninvalid 验证不通过时触发

31、HTML5 中的本地存储概念是什么？生命周期有多长？

答：随着互联网的快速发展，基于网页的应用越来越普遍，同时也变的越来越复杂，为了满足各种各样的需求，会经常性在本地存储大量的数据，传统方式我们以 document.cookie 来进行存储的，但是由于其存储大小只有 4k 左右，并且解析也相当的复杂，给开发带来诸多不便，HTML5 规范则提出解决方案。HTML5 storage 提供了一种方式让网站能够把信息存储到你本地的计算机上，并再以后需要的时候进行获取。这个概念和 cookie 相似，区别是它是为了更大容量存储设计的。

1 特性

1、设置、读取方便

2、容量较大，sessionStorage 约 5M、localStorage 约 20M

3、只能存储字符串，可以将对象 JSON.stringify() 编码后存储

2 window.sessionStorage

1、生命周期为关闭浏览器窗口

2、在同一个窗口下数据可以共享

3 window.localStorage

1、永久生效，除非手动删除

2、可以多窗口共享

4 相关的一些方法详解

setItem(key， value) 设置存储内容

getItem(key) 读取存储内容

removeItem(key) 删除键值为 key 的存储内容

clear() 清空所有存储内容

key(n) 以索引值来获取存储内容

32、Post 与 get 的区别，什么时候用 post，什么时候用 get？

答：区别：

get 存储内容小，不能超过 2kb，有限；文件上传只能用 post。

get 不安全，显示在地址栏。

get 效率高，因为 post 请求需要加密和解密的过程，get 不需要。

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；在提交一些不紧要信息时，使用 get，效率高。

33、Javascript 面向对象中继承实现？

答：继承：就是自己没有， 别人有. 拿过来为自己所用， 并成为自己的东西

传统继承基于模板，js 继承基于对象

1.原型链

2.借用构造函数

3.组合继承

4.原型式继承

5.寄生式继承

6.寄生组合式继承

34、’data-’属性的作用是什么？如何获取’data-’属性的值？

答：data-属性是 H5 中的，用来自定义属性，通过 dataset 属性获取。

不是所有的浏览器都支持，不支持用 getAttribute 获取。

35、JavaScript 的事件流模型都有什么，以及怎么阻止他们？

答：1、原始事件模型

普通的事件绑定，比如事件赋值，按钮上绑定事件等

2、DOM2 事件模型

addEventListener("eventType"，"handler"，"true!false")；

removeEventListner("eventType"，"handler"，"true!false")；

气泡模型（与 ie 有点区别），冒泡和捕获

3、IE 模型

气泡模型

attachEvent("eventType"，"handler")

detachEvent("eventType"，"handler")

与 dom2 不同的是 eventType 有 on 前缀

36、选其中一个正则问题作答(写出正则即可)

a) 写一个控制在 30 天到 180 天的正则表达式

```
/^([3-9]|1[1-8])\d$/
```

b) 写一个能够验证 座机号码 的正则表达式。如：010-12345678

```
/^\d{3}_\d{8}$/
```

三、其他问题（相关的优化问题）

1、请谈谈你对性能优化的认识？

答：网页内容

减少 http 请求次数

80%的响应时间花在下载网页内容(images， stylesheets， javascripts， scripts， flash 等)。减少请求次数是缩短响应时间的关键！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。

减少 DNS 查询次数

DNS 查询也消耗响应时间，如果我们的网页内容来自各个不同的 domain (比如嵌入了开放广告，引用了外部图片或脚本)，那么客户端首次解析这些 domain 也需要消耗一定的时间。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。下面是我清空本地 dns 后访问博客园主页 dns 的查询请求。

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，例如 ajax 的返回是用户联系人的下拉列表。所以我们还是要注意尽量应用以下规则提高 ajax 的响应速度。

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。剩下的内容就可以推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来完成一些高级的功能。

2、如何避免 XSS？

答：禁止危险脚本

IE8 是第一款内置了 XSS 脚本拦截保护的浏览器。谷歌的 Chrome 也会紧随其后推出类似功能。这两款浏览器都会首先查看来自某个 Web 服务器的脚本是否是恶意的——如果是，就拦截它。在今年 4 月的黑帽欧洲 2010 大会上，研究专家 David Lindsay 和 Eduardo Vela Nava 却演示了一种可以破除这种拦截的办法，不过谷歌已经修复了 Chrome 中的这个漏洞。微软则在今年 1 月（补丁 MS10-002）和 3 月（MS10-018）也已解决了大部分问题，并计划在 6 月修复第 3 个漏洞，所以在你读到这篇文章的时候，破除 XSS 脚本拦截的问题可能已经完全解决了。

Firefox 的用户则可以利用免费的 NoScrpit 附加组件有选择地拦截脚本。比如说，你可以放行一段 Flash 视频，而同时拦截该网站上的其他脚本组件。IE 和 Chrome 在拦截可疑脚本方面没有这么细的粒度——它们是要么全拦截，要么全不拦截。

NoScrpit 也有一个问题，那就是大多数用户并不喜欢放行个别脚本的做法，因为这样会带来不便。不过拦截和放行今后可能会成为你的第二天性。你还可以对某个特定网站上的所有脚本进行认证，无论是为了一次性访问还是今后的所有访问，这样的认证如今在 IE 8 和 chrome 中也可以做了，使得防范 XSS 攻击实现更加可能。

3、平时如何管理项目？

答：所谓项目，简单地说，就是在既定的资源和要求的约束下，为实现某种目的而相互联系的一次性工作任务。一般来说，项目具有如下的基本特征：

1）明确的目标其结果只可能是一种期望的产品，也可能是一种所希望得到的服务。

2）独特的性质每一个项目都是唯一的。

3）资源成本的约束性每一项目都需要运用各种资源来实施，而资源是有限的。

4）项目实施的一次性项目不能重复。

5)项目的不确定性在项目的具体实施中 ,外部和内部因素总是会发生一些变化 ,因此项目也会出现不确定性。

4、请谈谈项目的迭代周期？

答：软件项目开发，一般都会采用增量、迭代、（或者叫进化、演化、演进）的软件开发模型，众多的软件开发模型大多是以经典的瀑布模型为基础进行改进、变形，改进原则是：增加客户在整个项目周期中的参与度，降低软件开发过程中的风险，增强软件项目的后期可维护性。

不同的软件开发模型，迭代周期长短也不相同，有的是一个月，有的是两周，我们一般都是根据实际情况确定，一个周期完成，将项目成果（可运行的软件）提交给用户（或进行内部评审），通过后就进入下一个迭代开发周期。

5、工作中用过什么构建工具？

答：用过 gulp。

第一步：安装 node 和 npm，搭建 node 环境。

第二步：安装 gulp

第三步：新建 Gulpfile 文件，运行 gulp

安装依赖，提醒下，如果以上命令提示权限错误，需要添加 sudo 再次尝试。

Gruntfile 维护起来那么困难，有几个原因：

配置和运行分离

程序员都知道，变量的声明和使用挨在一起，最方便理解和修改。但 Gruntfile 里，配置 Task 和调用它们的地方离得很远，极大地增加了心智负担

每个插件做的事太多

每个 Task 的结果必须写到磁盘文件，另一个 Task 再读，损害性能倒是小事，更麻烦的是让整个过程变复杂了。就像一个个小作坊，来料加工又返回给客户，这中间的沟通成本、出错机会都大大增加。配置项过多做事多了，配置项自然也多。至少输入和输出的位置得配吧。每个插件的配置规则还不尽相同。用每个插件，都得去学习一番。

6、谈谈你对模块化的理解？

答：

- 模块化就是为了减少系统耦合度，提高高内聚，减少资源循环依赖，增强系统框架设计。
- 让开发者便于维护，同时也让逻辑相同的部分可复用
- 模块化开发：针对 js、css，以功能或业务为单元组织代码。js 方面解决独立作用域、依赖管理、api 暴露、按需加载与执行、安全合并等问题，css 方面解决依赖管理、组件内部样式管理等问题。

任何事物都有一个过程，那么模块化的过程通俗点讲就是：

模块化的过程就是：

- 1、拆分

将整个系统按功能，格式，加载顺序，继承关系分割为一个一个单独的部分.

注意：拆分的粒度问题，可复用问题，效率问题.如何这些问题处理的不好，就有可能出现不想要的后果。

将功能或特征相似的部分组合在一起，组成一个资源块.

将每个资源块按找需求，功能场景以及目录约束放到固定的地方以供调用.

模块的历程

模块化的发展也是从草根一步一步走过来的。从最开始到现在成熟方案：

1. namespace

2. sass，less
3. AMD&CMD
4. html 模版
5. grunt，gulp，webpack
6. FIS，YUI，KISSY

7、平时都用什么第三方插件？

答：pullpage，zepto，underscore，JqueryUI，JqueryMobile，Echart，ueditor，animate.js 等。

8、请描述一下 cookie，sessionStorage 和 localStorage 的区别？

答：cookies 兼容所有的浏览器，Html5 提供的 storage 存储方式。

① Document.cookie
② Window.localstorage
③ Window.sessionstorage

cookie 数据始终在同源的 http 请求中携带（即使不需要），即 cookie 在浏览器和服务器间来回传递。而 sessionStorage 和 localStorage 不会自动把数据发给服务器，仅在本地保存。

存储大小限制也不同，cookie 数据不能超过 4k，同时因为每次 http 请求都会携带 cookie，所以 cookie 只适合保存很小的数据，如会话标识。sessionStorage 和 localStorage 虽然也有存储大小的限制，但比 cookie 大得多，可以达到 5M 或更大。

数据有效期不同，sessionStorage：仅在当前浏览器窗口关闭前有效，自然也就不可能持久保持；localStorage：始终有效，窗口或浏览器关闭也一直保存，因此用作持久数据；cookie 只在设置的 cookie 过期时间之前一直有效，即使窗口或浏览器关闭。

作用域不同，sessionStorage 不在不同的浏览器窗口中共享，即使是同一个页面；localStorage 在所有同源窗口中都是共享的；cookie 也是在所有同源窗口中都是共享的。

9、如何使用缓存？

答：可以基于 http 的头信息控制缓存

ajax 请求对早期的 IE 浏览器默认就是缓存的，可以通过时间戳防止缓存。

10、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

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。

例如，可以通过 js 预先从服务加载图片资源(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。

11、缓存和预加载的区别是什么？

答：缓存就是把请求过的数据缓存起来，下次请求的时候直接使用缓存内容，提高响应速度

预加载指的是提前把需要的内容加载完成，访问的时候可以明天提高响应效率，比如图片的预加载（可以提前加载一定数量的图片，当用户访问图片的时候一般只看前几张，由于是预加载好的，所以速度比较快）。

12、图片如何压缩？

答：可以使用一些在线的图片压缩工具

优先用 png 而不是 gif

压缩 png

去掉 jpg 的 metadata

压缩 gif 动画

尝试使用 png8

避免使用 AlphaImageLoader

压缩动态生成的图像

使 favicon 更小 可缓存

使用 CSS Sprites

13、压缩文件有哪些方法？

答：使用 Grunt、Sass、ant 压缩

14、如何区分静态页面和动态页面？

答：要区分这两个，最简单的方法就是看后缀了，动态网页网址中有两个标志性的符号“?”和“&”（有的可能没有&），这个问号和&就是用来带参数的。现在几乎爱所有的网页都是动态网页。

15、字符串拼接和模板引擎，项目中会如何操作？模板引擎减少 http 请求，字符串不可变?模板引擎会不会利于 SEO 优化？

答：简单的数据渲染，拼接字符串即可，稍微复杂的业务逻辑使用前端模板引擎，过于复杂的页面基本上使用后台渲染的方式；模板引擎会影响 SEO 优化，为了解决这个问题，需要关注 SEO 的页面最好采用后台渲染的方式。

16、前台兼容性问题有哪些？

答：主要是常用浏览的（前端）API 差异，渲染差异，等等。

17、你如何对网站的文件和资源进行优化?期待的解决方案包括？

答：文件合并
文件最小化/文件压缩
使用 CDN 托管
缓存的使用

18、内存泄漏怎么理解？

答：IE6 时代有 bug，闭包会造成内存泄漏，这个现在已经无须考虑了。
其次，闭包本身不会造成内存泄漏，但闭包过多很容易导致内存泄漏。
这句话很矛盾，技术上讲，闭包是不会造成内存泄漏的，浏览器的 bug 除外。但是，闭包会造成对象引用的生命周期脱离当前函数的上下文，因此，如果不仔细考虑闭包函数的生命周期，的确有可能出现意料之外的内存泄漏，当然，从严格意义上讲，这是程序员自己的 bug，而不是闭包的错。

19、微格式到底是做啥用？

答：是开放的数据格式，面向的是普通用户，任何用户可以透过简单的程序读取微格式内容。而不是像 Flickr、Amazon、Google 等提供特定的面向技术人员的 API（一般基于 XML-PRC、REST，相对复杂）。RSS 具有微格式的部分优点，但限制还是比较多的，比如有限的元数据（标题、描述、URL 等），不能更好地描述语义，不太容易与已存在的工具结合等。用微格式可以来聚合外部 Blog，Flickr，YouTube，MapQuest，甚至 MySpace 里的内容。

微格式实际就是为现有的(X)HTML 元素添加元数据和其他属性，增强语义。

20、懒加载是用滚轮判断高度好还是用插件？

答：使用插件比较好，插件考虑的问题比较全面，仅仅通过滚轮高度判断很容易导致一些副作用（比如一次性请求多次）。

21、如何缓存整个页面，在没有网络的时候可以来回的跳转？

答：使用 HTML5，通过创建 cache manifest 文件，可以轻松地创建 web 应用的离线版本。

如需启用应用程序缓存，请在文档的 <html> 标签中包含 manifest 属性。

每个指定了 manifest 的页面在用户对其访问时都会被缓存。如果未指定 manifest 属性，则页面不会被缓存（除非在 manifest 文件中直接指定了该页面）。

manifest 文件的建议的文件扩展名是：".appcache"。

manifest 文件需要配置正确的 MIME-type，即 "text/cache-manifest"。必须在 web 服务器上进行配置。

manifest 文件可分为三个部分：

CACHE MANIFEST - 在此标题下列出的文件将在首次下载后进行缓存

NETWORK - 在此标题下列出的文件需要与服务器的连接，且不会被缓存

FALLBACK - 在此标题下列出的文件规定当页面无法访问时的回退页面（比如 404 页面）

实例 - 完整的 Manifest 文件

```
CACHE MANIFEST
# 2012-02-21 v1.0.0
/theme.css
/logo.gif
/main.js

NETWORK:
login.asp

FALLBACK:
/html5/ /404.html
```

22、CDN 是啥？

答：**CDN 的全称**：是 Content Delivery Network，即内容分发网络，加速的意思，那么网站 CND 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），然后在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供内容服务；相对于直接访问源站，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 提高了企业站点(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加速，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于机制，可以有效地预防黑客入侵以及降低各种 D.D.o.S 攻击对网站的影响，同时保证较好的服务质量。

23、浏览器一次可以从一个域名下做多少资源？

答：浏览器的并发请求数目限制是针对同一域名的，同一时间针对同一域名下的请求有一定数量限制，不同浏览器这个限制的数目不一样，超过限制数目的请求会被阻塞；

目前的话，所有浏览器的并发数目一般限制在 10 以内。

24、什么是垃圾回收机制（GC）？

答：早期的计算机语言，比如 **C 和 C++，需要开发者手动的来跟踪内存**，这种机制的优点是内存分配和释放的效率很高。但是它也有着它的**缺点，程序员很容易不小心忘记释放内存，从而造成内存的泄露**。

新的编程语言，比如 JAVA， C#， javascript， 都提供了所谓“垃圾回收的机制”，运行时自身会运行相应的垃圾回收机制。**程序员只需要申请内存**，而不需要关注内存的释放。垃圾回收器(GC)会在适当的时候将已经终止生命周期的变量的内存给释放掉。**GC 的优点就在于它大大简化了应用层开发的复杂度，降低了内存泄露的风险。**

25、image 和 canvas 在处理图片的时候有什么区别？

答：image 是通过对象的形式描述图片的；

canvas 通过专门的 API 将图片绘制在画布上。

26、简述移动开发的注意点，如何做好不同手机的适配，你以前的项目是怎么做的?

答：1、单独做移动端项目，采用百分比布局

2、采用响应式的方式做适配

27、响应式布局的时候，轮播图使用两张不同的图片去适配大屏幕和超小屏幕，还是一张图片进行压缩适配不同终端，说明原因？

答：最好使用两张不同大小的图片去适配大屏幕和超小屏幕，这样可以针对不同设备的屏幕大小，来加载响应的图片，减少超小屏幕设备的网络流量消耗，加快响应速度，同时防止图片在大屏幕下分辨率不够导致失真的问题。

28、http 和 tcp 有什么区别？

答：TPC/IP 协议是传输层协议，主要解决数据如何在网络中传输，是一种“经过三次握手”的可靠的传输方式；

HTTP 协议即超文本传送协议(Hypertext Transfer Protocol)，是应用层协议，是 Web 联网的基础，也是手机联网常用的协议之一，**HTTP 协议是建立在 TCP 协议之上的一种应用。**

29、向 git 中添加一个文件并 commit，然后 push 到 remote server，请写出相关命令？

答：

```
$ git add README.md
$ git commit -m "add README.md"
$ git push origin master
```

30、请把以下 HTML 文档翻译成 MarkDown 格式？

```
    <h3>Header</h3>
    <p>Hello world!<a href="https: //www.google.com">Google</a></p>
    <ol>
        <li>Number One</li>
        <li>Number Two</li>
    </ol>
答: ### Header
Hello world![Google](https: //www.google.com)
1.  Number One
```

```
2. Number Two
```

31、你做的页面在哪些流览器测试过?这些浏览器的内核分别是什么?

答：Ie(Ie 内核) 火狐（Gecko） 谷歌（webkit） opear(Presto)

32、写出几种 IE6 BUG 的解决方法？

答：1. 双边距 BUG float 引起的 使用 display：inline；

2. 3 像素问题 使用 float 引起的 使用 dislpay：inline -3px

3. 超链接 hover 点击后失效 使用正确的书写顺序 link visited hover active

4. Ie z-index 问题 给父级添加 position：relative

5. Png 透明 使用 js 代码 改

6. Min-height 最小高度 ！Important 解决’

7. select 在 ie6 下遮盖 使用 iframe 嵌套

8. 为什么没有办法定义 1px 左右的宽度容器（IE6 默认的行高造成的，使用 over：hidden，zoom：0.08 line-height：1px）

33、图片优化

项目中图片处理相关的优化，项目中用到的优化方案，图片大小达到多少的时候选择处理？

答：1、首先了解在 web 开发中常见的图片有那些格式。

JPG 通常使用的背景图片，照片图片，商品图片等等。这一类型的图片都属于大尺寸图片或较大尺寸图片一般使用的是这种格式。

PNG 这种格式的又分为两种 一种 PNG-8，一种 PNG-24。

PNG-8 格式不支持半透明，也是 IE6 兼容的图片存储方式。

PNG-24 图片质量要求较高的半透明或全透明背景，保存成 PNG-24 更合适（为了兼容 IE 可以试用 js 插件 pngfix）一般是背景图标中使用的多。

GIF 这种格式显而易见的是在需要 gif 动画的时候使用了。

2．优化方案

- 样式代替图片

例如：半透明、圆角、阴影、高光、渐变等。这些效果主流的浏览器都能够完美支持，而对于那些低端浏览

器，我们并不会完全抛弃他们，“渐进增强”则是一个很好的解决方案。

- 精灵图

CSS Sprites，将同类型的图标或按钮等背景图合到一张大图中，减少页面请求。

- 字体图标

Icon Font，将图标做成字体文件。优点是图标支持多个尺寸，兼容所有浏览器，减少页面请求等。美中不足的是只支持纯色的 icon。SVG，对于绝大多数图案、图标等，矢量图更小，且可缩放而无需生成多套图。现在主流浏览器都支持 SVG 了，所以可放心使用！

- Base64

```
将图片转化为 base64 编码格式，资源内嵌于 CSS 或 HTML 中，不必单独请求。
    Base64 格式
    data: [][; charset=][; base64],
    Base64 在 CSS 中的使用
    .demoImg{ background-image:  url("data: image/jpg; base64, /9j/4QMZRXhpZgAA
SUkqAAgAAAA   L....");  }
    Base64 在 HTML 中的使用
    <img width="40" height="30"   src="data: image/jpg; base64, /9j/4QMZRXhpZg
AASUkqAAgAAAAL...." />
```

- 图片响应式

通常图片加载都是可以通过 lazy 加载的形式来的，那么可以在加载的时候来判断屏幕的尺寸来达到加载大图还是小图的目的来达到优化。

34、你知道有哪些方法可以提高网站的性能？

答：我们从两个方面来讲：

（1）资源加载

CSS 顶部， JS 底部

CSS JS 文件压缩

尽量使用图片使用精灵图，字体图标

图片加载可通过懒加载的方式。

总之就是减少资源体积减少资源请求次数。

（2）代码性能

Css：

使用 CSS 缩写，减少代码量；

减少查询层级：如.header .logo 要好过.header .top .logo；

减少查询范围：如.header>li 要好过.header li；

避免 TAG 标签与 CLASS 或 ID 并存：如 a.top、button#submit；

删除重复的 CSS；

....

Html：

减少 DOM 节点：加速页面渲染；

正确的闭合标签：如避免使用 ，浏览器会多一个将它解析成<div\></div\>的过程；

减少页面重绘。比如 给图片加上正确的宽高值：这可以减少页面重绘，

.....

Js：

尽量少用全局变量；

使用事件代理绑定事件，如将事件绑定在 body 上进行代理；

避免频繁操作 DOM 节点；

减少对象查找，如 a.b.c.d 这种查找方式非常耗性能，尽可能把它定义在变量里；

35、设计模式有哪些？列举你在前端开发工作中自己应用到或者了解到其他框架所用到的设计模式？

答：单例、工厂、观察者、适配器、代理模式。

36、请描述你熟悉的语言的垃圾回收(GC)机制，他们对循环引用是如何处理的？如何查找内存泄漏(MemoryLeak)?

答 :JavaScript 的垃圾回收机制主要是根据数据是否还存在引用 ,没有引用的数据空间可能在某个时间被回收；在 java 中垃圾回收机制采用对象遍历来解决循环引用；windows 的任务管理器就可以查看到内存泄露。

四、Angular、主流框架和服务器相关问题

1、ng-app 是什么？

答：ng-app 指令用于告诉 AngularJS 应用当前这个元素是根元素。

所有 AngularJS 应用都必须要要一个根元素。

HTML 文档中只允许有一个 ng-app 指令，如果有多个 ng-app 指令，则只有第一个会被使用。

2、说说 MVC 和 MVVM 分别是什么？

答：MVC 全名是 Controller 模型(model) - 视图(view) - 控制器(controller)的缩写，
MVVM 是 Model-View-ViewModel 的简写。

3、-g 是什么？

答：-g 是-global 的简称，全局的意思。

4、自定义指令的类型（E，A，C，M）？

答：元素（E）、属性（A）、类（C）、注释（M）。

5、$scope 和自定义指令里的 scope 有啥区别？

答：$scope 对象在 AngularJS 中充当数据模型的作用，也就是一般 MVC 框架中 Model 得角色.但又不完全与通常意义上的数据模型一样，因为 $scope 并不处理和操作数据，它只是建立了视图和 HTML 之间的桥梁，让视图和 Controller 之间可以友好的通讯.

自定义指令里的 scope 表示指令的作用域，它有三个可选值：true、false、对象{}

6、Ionic 中的路由？

答：Ionic 也是基于 Angular 的，使用的是 ui-router，
ui-router 的核心理念是将子视图集合抽象为一个状态机，导航意味着 状态的切换，Ionic 之所以没有使用 Angular 官方的 ngRoute，是回 ngRoute 缺少一些高级的特性，比如视图命名，视图嵌套。

7、filter？

答：过滤器。

8、ng-bind？

答：ng-bind 指令告诉 AngularJS 使用给定的变量或表达式的值来替换 HTML 元素的内容。
如果给定的变量或表达式修改了，指定替换的 HTML 元素也会修改。

9、说一说 link？

答：link 中可以拿到 scope 和 controller，可以与 scope 进行数据绑定，与其他指令进行通信。

10、为什么 angular 不推荐使用 dom 操作？

答：Angular 倡导以测试驱动开发，在的 service 或者 controller 中出现了 DOM 操作，那么也就意味着的测试是无法通过的

使用 Angular 的其中一个好处是啥，那就是双向数据绑定，这样就能专注于处理业务逻辑，无需关系一堆堆的 DOM 操作。如果在 Angular 的代码中还到处充斥着各种 DOM 操作，那为什么不直接使用 jquery 去开发呢。

11、看过 Angular 的源码吗，它是怎么实现双向数据绑定的？

答：angular 对常用的 dom 事件，xhr 事件等做了封装， 在里面触发进入 angular 的 digest 流程。
在 digest 流程里面， 会从 rootscope 开始遍历， 检查所有的 watcher。

12、ui-router 和 ng-router 区别？

答：AngularJS 的 ng-route 模块为控制器和视图提供了[Deep-Linking]URL
ui-router 的核心理念是将子视图集合抽象为一个状态机，导航意味着 状态的切换。

13、什么是指令？

答：指令是指示计算机执行某种操作的命令，它由一串二进制数码组成。一条指令通常由两个部分组成：操作码+地址码。

14、service 服务三种方式是什么？

答：angularjs 中可通过三种（$provider，$factory，$service）方式自定义服务。

15、gulp 任务都是怎么定义，怎么执行的？

答：通过 gulp.task 方法定义任务，在项目中新建 gulpfile.js 文件，书写代码，如：

```
var gulp = require('gulp')
gulp.tasks(
'script' // 任务名
```

```
, function(){
  // 在这里写任务需要执行的代码
});
```
在命令输入`gulp 任务`，可以执行所在目录 gulpfile.js 文件中的任务。

16、Bootstrap 中最多可以分多少列？lg、md、sm、xs 这几个屏幕宽度的界限是多少？

答：12 列

.col-xs- 超小屏幕手机 (<768px)

.col-sm- 小屏幕平板 (≥768px)

.col-md- 中等屏幕桌面显示器 (≥992px)

.col-lg- 大屏幕大桌面显示器 (≥1200px)

17、angular 中方法 apply 和 digest 区别？

答：当数据出现没有经过 angular 但是发生改变的情况下，需要调用 apply。Apply 的范围比较广，只执行一次，但是 digest 针对某一元素执行多次。

18、前端路，什么是“前端路由”?什么时候适合使用“前端路由”？ “前端路由”有哪些优点和缺点?

答：路由 Router

前端的路由都是通过 hash 来实现的，hash 能兼容低版本的浏览器。

Web 服务并不会解析 hash，也就是说 # 后的内容 Web 服务都会自动忽略，但是 JavaScript 是可以通过 window.location.hash 读取到的，读取到路径加以解析之后就可以响应不同路径的逻辑处理。

优点可以控制 业务逻辑 做无页面刷新 体验更好

缺点页面不刷的话无法释放内存，如果过多的操作会造成页面体验不好。

19、ng-show/hide 和 ng-if 的区别是什么？

答：Show/hide 是显示隐藏，if 是是否存在某一部分。

20、react 虚拟 DOM 运行机制是什么？

答：在 React 中，render 执行的结果得到的并不是真正的 DOM 节点，结果仅仅是轻量级的 JavaScript 对象，我们称之为 virtual DOM。虚拟 dom。

21、react 中 prop 和 state 的区别？

答：需要理解的是，props 是一个父组件传递给子组件的数据流，这个数据流可以一直传递到子孙组件。而 state 代表的是一个组件内部自身的状态（可以是父组件、子孙组件）。

22、redux 的原理？

答：Redux 把一个应用程序中，所有应用模块之间需要共享访问的数据，都应该放在 State 对象中。这个应用模块可能是指 React Components ,也可能是你自己访问 AJAX API 的代理模块 ,具体是什么并没有一定的限制。State 以 “树形” 的方式保存应用程序的不同部分的数据。这些数据可能来自于网络调用、本地数据库查询、甚至包括当前某个 UI 组件的临时执行状态（只要是需要被不同模块访问）。

23、node 常用模块？

答：http fs path url Buffer process

24、了解 npm，spm，nodejs 吗，请简要描述？

答：NPM 便于 JavaScript 开发者共享和重用代码，它可以很容易地更新你的代码；再分享。是全球最大的开源库生态系统。

SPM 是淘宝社区电商业务（xTao）为外部合作伙伴（外站）提供的一套跟踪引导成交效果数据的解决方案。

Node.js 是一个基于 Chrome V8 引擎的 JavaScript 运行环境。Node.js 使用了一个事件驱动、非阻塞式 I/O 的模型，使其轻量又高效。它使我们能够在本地运行 javascript。

25、请列举在内网的两台服务器中拷贝文件的方法？用 Shell 脚本解答数据库？

答：scp -P 1234 -r test_folder user@192.168.1.20：~

主要就是 scp 命令的使用

26、请描述你所熟悉的 Web 服务器框架(如 Django)作为一个成熟的 Web 框架，需要提供哪些重要的功能模块？

答：提供了网站开发的常用模块：处理用户请求、操作数据库、模板渲染、配置文件管理等。

27、服务器 Node.js 和浏览器 js 的区别是什么？Node.js 把 js 从客户端迁移了到服务端、主要做了哪些工作？为什么说 Node.js 适合做高并发的互联网应用？

答：Node 采用一系列“非阻塞”库来支持事件循环的方式。本质上就是为文件系统、数据库之类的资源提供接口。Node.js 使用事件驱动，非阻塞 I/O 模型而得以轻量和高效，非常适合在分布式设备上运行数据密集型的实时应用。

五、网络相关问题

1、请解释下列术语：UrlEncode，Utf8，JSON，UTC，MD5？

答：Urlencode：将字符串以 url 形式编码（在编程语言中通常都有实现该功能的内置函数或者 API）。

Utf8：是一种针对 Unicode 的可变长度字符编码，主要用于在网页上显示各国语言字符。

2、请解释 GET/POST 的区别，以及请求参数放到 url 里和放到 body 里面的区别？

答：Post 与 Get 区别：

GET 请求，请求的数据会附加在 URL 之后，以?分割 URL 和传输数据，多个参数用&连接。URL 的编码格式采用的是 ASCII 编码，而不是 uniclde，即是说所有的非 ASCII 字符都要编码之后再传输。

POST 请求：POST 请求会把请求的数据放置在 HTTP 请求包的包体中。上面的 item=bandsaw 就是实际的传输数据。

因此，GET 请求的数据会暴露在地址栏中，而 POST 请求则不会。

传输数据的大小

在 HTTP 规范中，没有对 URL 的长度和传输的数据大小进行限制。但是在实际开发过程中，对于 GET，特定的浏览器和服务器对 URL 的长度有限制。因此，在使用 GET 请求时，传输数据会受到 URL 长度的限制。

对于 POST，由于不是 URL 传值，理论上是不会受限制的，但是实际上各个服务器会规定对 POST 提交数据大小进行限制，Apache、IIS 都有各自的配置。

安全性

Get 是 Form 的默认方法，安全性相对比较低。

请求参数放到 url 里和放到 body 里面的区别

首先，参数的存放位置我们无法直接指定，而是不同的请求方法参数传递的方式不同。

常用的 HTTP 请求主要为 GET 请求和 POST 请求两种，GET 请求的参数会通过以跟随在 URL 后边以键值对的方式进行传递（例：key1=a&key2=b&key3...）；而 POST 请求的参数会通过 HEADER 进行传递。考虑到安全性

的问题，可以确定两者都不安全，原因是 HTTP 请求可以被轻易抓包和截获，其中的请求参数值自然会很容易被获取。

3、请列举出常用的 Http Header，Cookie 是怎么实现的？

答：Content-Length，请求、响应体的数据字节大小

Accept-Encoding，请求头，可接受的文本压缩算法，如： gzip， deflate

Accept-Language，请求头，支持语言，客户端浏览器的设置，如：zh-cn，zh；q=0.8，en-us；q=0.5，en；q=0.3

User-Agent，请求头，浏览器信息，

Cookie，请求头，服务器或客户端在上次设置的 COOKIE，包括作用域名(.360buy.com)，过期时间，键与值。

Referer，从一个连接打开一个新页面，新页面的请求一般会加此信息，标名是从哪里跳过来的，所有的页面的打开历史链就可被挖掘出来，有利于分析用户行为与 CPS 分成

Cookie 在浏览器本地会有一个文件存储数据，通信的时候通过请求头和响应头传递数据

4、请解释下列返回码的含义：200，302，400，403，500，502

答：200：请求成功

302：请求的资源临时从不同的 URI 响应请求。(资源临时重定向)

400：错误请求（请求的参数错误或者服务器不理解请求的语法）

402：10.4.3 402 Payment Required This code is reserved for future use.

该状态码是为了将来可能的需求而预留的

500：服务器端错误

502：网关或代理无效/无响应，网络错误

5、长连接和短连接的区别

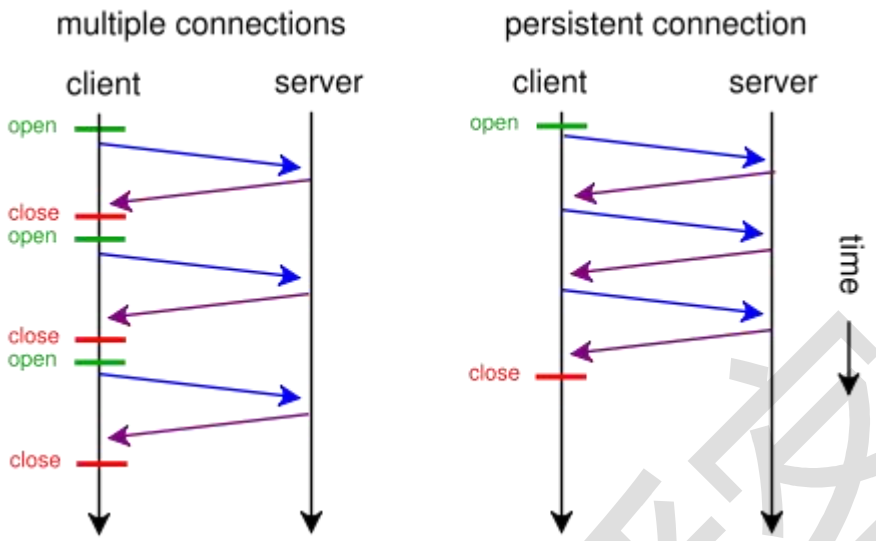
HTTP 协议目前常用的有哪几个？KEEPALIVE 从哪个版本开始出现的？

到现在 http 出现了 1.0 和 1.1 版本

Keep-Alive 是从 1.1 默认就支持了。

1、什么是 Keep-Alive 模式？

我们知道 HTTP 协议采用“请求-应答”模式，当使用普通模式，即非 KeepAlive 模式时，每个请求/应答客户和服务器都要新建一个连接，完成之后立即断开连接（HTTP 协议为无连接的协议）；当使用 Keep-Alive 模式（又称持久连接、连接重用）时，Keep-Alive 功能使客户端到服务器端的连接持续有效，当出现对服务器的后继请求时，Keep-Alive 功能避免了建立或者重新建立连接。



http 1.0 中默认是关闭的，需要在 http 头加入"Connection： Keep-Alive"，才能启用 Keep-Alive；http 1.1 中默认启用 Keep-Alive，如果加入"Connection： close "，才关闭。目前大部分浏览器都是用 http1.1 协议，也就是说默认都会发起 Keep-Alive 的连接请求了，所以是否能完成一个完整的 Keep-Alive 连接就看服务器设置情况。

2、启用 Keep-Alive 的优点

从上面的分析来看，启用 Keep-Alive 模式肯定更高效，性能更高。因为避免了建立/释放连接的开销。

6、从服务器考虑提高网站性能？

答：业界常用的优化 WEB 页面加载速度的方法（可以分别从页面元素展现，请求连接，css，js，服务器等方面介绍）？

对于服务器方面前端能做的工作：

使用 CDN 加速，使用户从离自己最近的服务器下载文件；

减少 Cookie 的大小，使用无 cookie 的域，客户端请求静态文件的时候，减少 Cookie 的反复传输对主域名的影响；

为文件头指定 Expires，使内容具有缓存性；

前端优化：DNS 预解析提升页面速度

<link rel="dns-prefetch" href="http：//hm.baidu.com" />

<link rel="dns-prefetch" href="http：//eiv.baidu.com" />

服务器端能做的工作：

负载均衡，分布式存储，提升服务器性能等等。

7、什么是 Daemon 进程？

答：Daemon()程序是一直运行的服务端程序，又称为守护进程。通常在系统后台运行，没有控制终端，不与前台

交互，Daemon 程序一般作为系统服务使用。Daemon 是长时间运行的进程，通常在系统启动后就运行，在系统关闭时才结束。一般说 Daemon 程序在后台运行，是因为它没有控制终端，无法和前台的用户交互。Daemon 程序一般都作为服务程序使用，等待客户端程序与它通信。我们也把运行的 Daemon 程序称作守护进程。

8、优化一个以 I/O 为瓶颈的程序，以下哪些方法效果比较显著，Why?

a) 增加 CPU 数目
b) 提高 CPU 主频
c) 增大内存的容量
d) 采用多线程
e) 采用异步 I/O 和多路(Multlplex)I/O
f) 对每次 I/O 进行 Bath 访问(多次 I/O 合并一次完成)

答：c、e、d、f 提升的效果会比较显著

c 通过将数据预读取到内存中（建立内存池）的方式，提高访问时候的效率，有效减少磁盘 IO 读写次数。

9、什么是内存对象的序列化(Serialiization)？为什么要序列化？请描述你熟悉的网络传输序列化(Serialiization)框架或格式(Server)？

答：把对象转换为字节序列的过程称为对象的序列化
序列化主要用于网络传输数据及将数据保存在硬盘上
常见的序列化以后的格式有：XML　Jason　，但它们都是字符串

六、项目相关问题

1、请谈下团购倒计时如何实现？

答：团购倒计时页面端的效果比较好实现，主要是样式和时间的操作，重要的考虑时间要和服务器端同步，其实这个效果也可以基于服务器端推送技术来实现。

2、轮播图有哪几种？如何实现？

答：纯 CSS 可以实现轮播图；JS 实现轮播图。

3、如何实现数组去重？

```
答:
A. 最简单的可以直接利用 ES5 的 indexOf 方法。
function dupRemove(arr) {
       var tmp = []; //一个新的临时数组
    for (var i = 0; i < arr.length; i++) {
          // 判断 tmp 数组中是否存在 arr 中第 i 元素，如果不存在则添加到 tmp 数据组。
    if (tmp.indexOf(arr[i]) == -1) {
    tmp.push(arr[i]);
          }
       }
    return tmp;
    }
B. 还有一种比较有意思的写法
function dupRemove (arr) {
    var tmp = [];
    for (var i = 0, l = arr.length; i < l; i++) {
    for (var j = i + 1; j < l; j++) {
    if (arr[i] === arr[j]) {
                j = ++i;
             }
          }
tmp.push(arr[i]);
       }
return r;
    }
```

4、写一个方法获取 url 中？号后面的参数，并将参数对象化？

```
答: function getSearch (url) {
var reg_url = /^[^\?]+\?([\w\W]+)$/,
    reg_params = /([^&=]+)=([\w\W]*?)(&|$|#)/g,
    arr_url = reg_url.exec(url),
    ret = {};
if (arr_url && arr_url[1]) {
var str_ params = arr_url[1], result;
while ((result = reg_ params.exec(str_ params)) != null) {
```

```
    ret[result[1]] = result[2];
      }
     }
    return ret;
    }
```

七、程序题

1、var a=[]； a[0]=0；a[1]=1；a[4]=4；请问 a.length 的值是多少？a[3]的输出结果是什么？

5 undefined

2、var a=[5，6]； var b=a； b[0]="hello"； alert(a[0])； 请问值是多少？

"hello"

3、typeof(null)，typeof(undefined)，typeof(NaN)，typeof(NaN==NaN)，说出上面代码执行结果？

object
undefined
number
boolean

4、function doSomething(){

```
    for(var i = 0;  4 > i;  i++) {
        var k = 100;
        aMrg +=', ' + (k + i);
        }
        }
    var k = 1, aMrg = k;
    doSomething();
    aMrg +=k;
    log(aMrg);

    1, 100, 101, 102, 1031
```

5、请写出下面输出的值

Console.log(undefined || 1)；//值___1__

Console.log(null || NaN)；//值__NaN___

Console.log(0 && 1)；//值__0___

Console.log(0 && 1 || 0)；//值__0___

6、看下列代码，<p>标签内的文字是什么颜色的？ 红色

```
<style>
.classA{color: blue};
.classB{color: red};
</style>
<body>
<p class=”classB classA”>123</p>
</body>
```

7、var a = [5，6]； var b = a； b[0] = “hello”； alert(a[0])； 值是多少？

"hello"

8、你面前有一座高塔，这座高塔有 N(N > 100)个台阶，你每次只能往前迈 1 个或者 2 个台阶，请写出程序计算总共有多少种走法？

```
这个案例满足斐波那契定律 1，1，2，3，5，8，13，21， 34， 55， 89， 144
var n1 = 1;
var n2 = 1;
var n3 = n1 + n2;
for (var i = 3; i <= n; i++) {
n3 = n1 + n2;
n1 = n2; //往后推一项
n2 = n3; //往后推一项
}
console.log(n3);
```

9、请阅读下面的 CSS 代码

```
#left {
color: white !important;
}
```

```
#container #left {
color: red;
}
#left {
color: green !important;
}
.container #left {
color: blue;
}
```

则在如下 html 中

```
<div class="container" id="container">
<span id="left">left</span>
</div>
```

#left 最终 color 属性值为？ 绿色

10、下面这段代码想要循环延时输出结果 0 1 2 3 4，请问输出结果是否正确，如果不正确说明为什么，并修改循环内的代码使其输出正确的结果。

```
for (var i = 0; i < 5; ++i) {
setTimeout (function () {
console.log(i + '');
}, 100*i);
}
```

不正确，先执行 FOR 循环。for 循环完成后，在去执行 setTimeout。但是这个时候 I 已经是 5 了，所以输入了 5 次 5

```
for (var i = 0; i <5; ++i) {
var a = 0;
setTimeout (function () {
console.log(a++);
}, 100*i);
}
```

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

```
<body>
<script type="text/javascript">
Function showImg (oSel) {

};
</script>
<img id="pic" src="img1.jpg" width="200" height="200">
<br />
```

```
    <select id="sel" onchange="showImg(this)">
    <option value="img1">城市生活</option>
    <option value="img2">都市早报</option>
    <option value="img3">青山绿水</option>
    </select>
    </body>

    var pic=document.getElementById('pic')
    function showImg (oSel) {
    pic.src=oSel.options[oSel.selectedIndex].value
    console.log(pic.src);
};
```

答案说明：当 select 发生改变的时候调用 showImg 函数，实参为 this（select 对象本身），可以通过 select 对象的属性来为 pic 的 src 赋值实现图片切换

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

```
    <html>
    <head>
    <meat http-equiv="Content-Type" content="text/html;  charset=utf-8">
    </head>
    <body>
    <script type="text/javascript">
    function foo() {

    };
    </script>
    <form name="form1" onsubmit="retuen foo()">
    <input type="radio" name = "radioGroup">
    <input type="radio" name = "radioGroup">
    <input type="radio" name = "radioGroup">
    <input type="radio" name = "radioGroup">
    </form>
    </body>
    </html>

    var a=document.getElementsByTagName('input')
    function foo() {
    for(var i=0; i<a.length; i++){
    if(a[i].checked){
    alert(i+1)
        }
```

```
    }
}
```

题出的有问题，onsubmit 只有在提交的时候才会触发这里面没有 submit 按钮，在提交事件触发的时候遍历哪个 input 表单是选中状态然后 alert 粗来

13、计算下面程序运行结果

```
var msg = 'hello';
function great(name, attr) {
name = 'david';
var greating = msg + name + '!';
var msg = '您好';
For (var i = 0 ; i < 10; i++) {
var next = msg + '您的 id 是' + i*2 + i;
}
console.log(arguments[0]);
console.log(arguments[1]);
console.log(greating);
console.log(next);
}
geat('Tom');
```

答案：david //参数 1

hellow world 01.html：20 undefined //参数 2 未传入为未定义

hellow world 01.html：21 undefineddavid! //name 虽然是参数但是参数重新赋值为 david 了 msg 因为变量声明提升所以值为 undefined

您好您的 id 是 189//因为 number+string=string 所以 for 循环最后一次声明 next=****18+9

14、下面这段 JS 输出什么，并简述为什么？

```
function Foo() {
var i = 0;
return function () {
console.log(i++);
}
}
var f1 = Foo(),
f2 = Foo();
f1();
f1();
f2();
```

```
    console.log(i);
    0   //f1=Foo() 相当于 f1 赋值为函数 Foo()的返回值 f1=function（）{
    console.log(i++)
    }
    1   //因为 f1=了一个 function 所以有了作用域，f2 和 f1 不同，不在一个内存中
    0
    报错  //i 为 Foo 内部的变量全局不可访问，全局中没有 i 变量所以会报错
```

15、请写出下面输出的值

```
(1)var num = 1;
   var fun = function () {
   console.log(num); //值___undefined___
   var num = 2;
   console.log(num); //值___2___
   }
   fun();
(2) var num = 1;
   function fun () {
   console.log(num); //值___1____
   num = 2;
   console.log(num); //值___2____
   }
   fun();
```

16、写出以下程序执行的结果

```
1)   var a = 10;
     a.pro = 10;
     console.log(a.pro + a);
   NAN number 对象不可以定义私有属性 namber+非数字和字符的值就等于 NaN
2)    var s = 'hello';
     s.pro = 'world';
     console.log(s.pro + s);
     //undefinedhello s 位字符串，字符串不可以自定义属性，所以 s.pro 为 undefined 字符串做加运算会强制拼接位字符串
3)    console.log(typeof fn);
     function fn() {};
     var fn;
   //function 函数提升优先于变量提升
(4)   var f = true;
     If(f === true) {
   var a = 10;
        }
```

```
  function fn() {
var b = 20;
c = 30;
  }
  fn();
  console.log(a);
//10
```

17、请看如下的代码，写出结果

```
var a = 5, b = 3;
function test() {
alert(b++);
var a = 4;
alert(--a);
alert(this.a);
}
```

1)tese()，三次 alert()的值依次是什么？335　435　535

2)new test()，三次 alert()的值依次是什么？ 33undefined 43undefined 53undefined //this 更改了指向原来是指向 window 用了 new 关键字后指向 test test 木有 a 属性所以为 undefined

18、p 最后显示什么颜色。怎么让 p 的颜色变成黑色，并简要说明 css 选择器优先级关系

```
<style>
#classA{color: yellow};
p.classB(color: red);
</style>
<body>
<p id="classA" class="classB">123</p>
</body>
//p#classA{color: black}
```

19、关于正则表达式声明 6 位数字的邮编，一下代码正确的是(C)

A.var reg = /\d6/；

B.var reg = \d{6}\；

C.var reg = /\d{6}/；

D.var reg = new RegExp（"\d{6}"）；

20、关于 JavaScript 里 xml 处理，一下说明正确的(A)

D.xml 是种可扩展标记语言，格式更规范，是作为未来 html 的替代　//貌似 XML 是被替代的

E.Xml 一般用于传输和存储数据，是对 html 的补充，两者的目的不同

F.在 JavaScript 里解析和处理 xml 数据时，因为浏览器的不同，其做法也不同

G.在 IE 浏览器里处理 xml，首先需要创建 ActiveXObject 对象

21、请选择对 javascript 理解有误的(B)

H.javascript 是网景公司开发的一种基于事件和驱动网页脚本语言

I.JScript 是 javascript 的简称　//微软自己的浏览器才支持

J.FireFox 和 IE 存在大量兼容性问题的主要原因在于他们对 javascript 的支持不同

K.AJAX 技术一定要使用 javascript 技术

22、在 Jquery 中下面哪一个是用来追加到指定元素的末尾(B)

A.inserAfter()
B.Append()
C.appendTo()
D.After()

23、在 javascript 中定义变量 var a=”35”，var b = “7”运算 a % b 的结果为(C)

A.357
B.57
C.0
D.5

24、下面哪个属于 javascript 的字符型　C

A.False

B.你好

C.“123”
D.Null

25、下面哪个属于 javascript 的布尔值　(C)

A.1.2
B.”true”

C.false
D.null

26、请选择结果为真的表达式(C)

A.null instanceof Object
B.Null === undefined；
C.null == undefined
D.NaN == NaN

27、下列运算方式不属于逻辑运算的是(D)

A.!a
B.a&&b
C.a||b
D.a>b

28、声明一个对象，给它加上 name 属性和 show 方法显示其 name 值，以下代码中正确的是(D)

E.var obj = [name ： “zhangsan” ，show ： function(){alert(name)；}]；
F.Var obj = {name ： “zhangsan”，show ： “alert(this.name)” }；
G.Var obj = {name ： “zhangsan”，show ： function () {alert(name)；}}；
H.Var obj = {name ： “zhangsan”，show ： function () {alert(this.name)；}}

29、以下过于 Array 数组对象的说法不正确的是(C)

I.对数组里数据的排序可以用 sort 函数，如果排序效果非预期，可以给 sort 函数加一个排序函数的参数
J.reverse 用于对数组数据的倒序排列
K.向数组的最后位置加一个新元素，可以用 pop 方法 //push 吧
L.unshift 方法用于向数组删除一个元素

30、要将页面的状态显示”已经选中该文本”，下列 JavaScript 语句正确的是(A)

M.window.status = “已经选中该文本”

N.Document.status = “已经选中该文本”

O.Window.screen = “已经选中该文本”

P.Document.screen = “已经选中该文本”

31、点击页面的按钮，使之打开一个新窗口，加载一个页面，以下 JavaScript 代码中可执行的是(D)

Q.<input type=”button” value=”new” onclick=”open(‘new.html’，’_blank’)”>

R.<input type=”button” value=”new” onclick=”window.location=’new.html’；”>

S.<input type=”button” value=”new” onclick=”location.assign(‘new.html’)；”>

T.<form target=”_blank” action=”new.html”>

<input type=”submit” value=”new”>

</form>

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

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

if(form1.elements[i].type ==”text”)

form1.elements[i].value = “”；

}

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

if(forms[0].elements[i].type == “text”)

form[0].elements[i].value = “”；

}

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

form.elements[i].value = “”；

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

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

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

document.forms[i].elements[j].value = “”；

}

}

33、在表单(form1)中有一个文本框元素(fname)，用于输入电话号码，格式如：010-82668155，要求前 3 位是 010，紧接一个”-”，后面是 8 位数字。要求在提交表单时，根据上述条件验证该文本框中输入内容的有效性，下列语句中(A)能正确实现以上功能

A.var str = form1.fname.value；If(str.substr(0，4)!=”010-”||str.substr(4).length!=8||isNaN(parseFloat(str.substr(4))))
Alert(“无效的电话号码！”)；

B.var str = form1.fname.value；If(str.substr(0，4)!=”010-”&&str.substr(4).length!=8&&isNaN(parseFloat(str.substr(4))))
Alert(“无效的电话号码！”)；

C.var str = form1.fname.value；If(str.substr(0，3)!=”010-”||str.substr(3).length!=8||isNaN(parseFloat(str.substr(3))))
alert(“无效的电话号码！”)；

D.var str = form1.fname.value；If(str.substr(0，4)!=”010-”&&str.substr(4).length!=8&&isNaN(parseFloat(str.substr(4))))
alert(“无效的电话号码！”)；

34、关于正则表达式声明 6 位数字的邮编，一下代码正确的是(C)

A.var reg = /\d6/；

B.var reg = \d{6}\；

C.var reg = /\d{6}/；

D.var reg = new RegExp (“\d{6}”)；

35、下面关于 cookie 的说明正确的是(D)

A.Cookie 设置的过期时间为 3600s 是指 60 分钟过期

B.Cookie 设置的过期时间为 3600s 是指只要在间隔 60 分钟内有动作时就不过期

C.Cookie 保存在服务器端

D.Cookie 保存在用户本地

36、使用 js 代码实现，将下面段落中含有的链接替换成可直接点击打开的链接

<p id=” text”>这个段落里有链接
比如：http：//www.abc.comm/和 https：//www.github.com/都是链接。
可是他们显示在网页中是，链接不可点，还得复制粘贴到地址栏打开，好麻烦
</p>

37、写一个方法获取 url? 后面的参数，并将参数对象化。

```
答: function parseQueryString(url){
        var params = {};
        var arr = url.split("?");
        if (arr.length <= 1)
           return params;
        arr = arr[1].split("&");
        for(var i=0, l=arr.length; i<l; i++){
           var a = arr[i].split("=");
           params[a[0]] = a[1];
        }
        return params;
    }
    var url = "http: //witmax.cn/index.php?key0=0&key1=1&key2=2";
    var ps = parseQueryString(url);
    console.log(ps["key1"]);
```

38、Node.js 中，一段访问 redis 的代码如下

```
var redis = require('redis');
var client = redis.createClient();
client.set('key', 'value', function (err, data) {
if(err) {
console.error(err.message);
process.exit(1; )
}
cilent.get('key', function (err, data) {
if(err) {
console.error(err.message);
return;
}
console.log(data);
process.exit(0);
})
});
```

请用 Promise 的异步调用方式重写

请用 ES6 yield 的异步调用方式重写

在经历了多个异步回调之后，如果拿到完整的堆信息(stack trace)?

39、用你认为合适的数据库产品，请设计数据结构，并完成一下方法(Server)；

1.当出现一次网页浏览的时候，请实现函数 click(url， ip)

2.请实现查询函数 pv(url)以及 uv(url)

40、补充按钮事件的函数，确认用户是否退出当前页面，确认之后关闭窗口

```
<html>
<head>
<script type=”text/javasccript”>
Function closeWin() {

}
</script>
</head>
<body>
<input type=”button” value=”关闭窗口” onclick=”closeWin()” />
</body>
</html>
```

41、请用 JavaScript 实现，控制一个文本框只能输入正整数，如输入不符合条件则文本全部字体标红，要求写出完整的文本框 HTML 代码和 JavaScript 逻辑代码？

```
答: <!DOCTYPE html>
<htmllang="en">
<head>
<meta charset="utf-8">
<title>只能输入正整数</title>
</head>
<body>
<input id="txt" type="text">
<script>
var txt= document.getElementById('txt');
var color = window.getComputedStyle(txt, '').color
txt.addEventListener('keyup', function () {
var reg = new RegExp("^[0-9]*$");;
console.log(reg.test(this.value));
if(reg.test(this.value)){
this.style.color=color;
}else{
```

```
this.style.color='red';
      }
    });
</script>
</body>
</html>
```

42、请对以下代码进行优化

```
var wrap = document.getElementById("wrap");
for(var i = 0;  i < 10;  i++) {
var li = document.createElement("li");
var text = document.createTextNode("hello" + i);
li.appendCChild(text);
wrap.appendChild(li);
}
```

43、请看下面的 HTML，写出您的 CSS 使左边元素宽度为 200px 保持不变，右边元素随浏览器大小自适应

左侧盒子左浮动，给固定宽度，右侧盒子宽度 100%；

```
<div class="body">
    <div class="left"></div>
    <div class="right"></div>
</div>
.left{
    float:  left;
    width: 200px;
    height: 300px;
}
.right{
    height: 300px;
}
```

八、非技术问题

1、请概述一下你上家公司中项目的具体情况(工作所使用的技术，业务流程，周期)？

答：我在上家公司做的的网站 pc 端页面的开发与维护。

主要技术：利用 div+css 布局，对用户注册页面开发，利用 ajax 技术与后台进行数据交互，使用 Validation 插件进行表单验证；使用 bootstrap/zepto 框架开发移动端页面，解决移动端设备的适配问题；侧边菜单栏的开发；利用 css3 完成轮播图动画。

业务流程：根据需求分析，进行详细的总体设计，产生各栏目文件夹的结构图，根据美工的表现需要，设计静态网页和其它动态页面界面框架，程序员进行代码开发，做一些必要的测试，由项目组共同联调测试，发现 bug，完善一些具体的细节。

开发周期：3 个月

2、常用调试和优化工具？

Firebug+YSlow+其它 Firefox 扩展

浏览器自带工具，IE Developer Toolbar，Opera Dragonfly

Fiddler、HTTP Analyzer、HttpWatch、Web Developer、Web Accessibility Toolbar

3、什么叫代码部署？如何部署？

答：代码部署就是把开发好的网站代码放到应用服务器上对外提供服务、部署方式根据编程语言的不同而不同，但是大体的流程是一致的，生产环境主要是通过命令加配置文件的形式进行部署。

4、新技术通过哪些渠道了解和学习？

优秀的博客，github，http：//www.daqianduan.com

5、对于前端这个岗位，兴趣的比例占多少？

70%

6、前端到底工作内容是什么？和 UI 有什么区别？

Web 前端：主要讲 UI 提供的设计图，编码成静态 html，实现所有特效；并负责所有交互的对接，对 js 要求较高
UI：主要对移动端和网页的设计

7、你当时进公司时是以什么身份进的，实习生吗？

当时是以软件工程师身份，进公司之后直接上岗开发

8、工作中如果出现空档期的时候，你们都在做些什么？

答：空档期的话大家就都学习，学一些新的技术，也可以跟着 js 大神学习嘛，不过空档期的情况也很少，一般就几天。

9、平常在公司有做网页制作吗？

答：如果实在忙的时候，我也帮制作师制作一些网页。

10、忙的时候，会帮网页制作做到什么程度，百分之多少？

答：一般忙的时候，我会先完成我自己的工作，像一些 JS 的特效和交互之后，再去帮网页制作，大概是 10%。

11、你在你做过的哪个项目调试中，遇到了哪些比较深刻的部分，说一说。你发现到解决这个问题用了多久？

答：通过你描述的问题难度，和你发现到解决问题的时间，看你 js 程度。

12、身为一位 web 前端工程师，你肯定知道现在最流行的前端技术有哪些吧？请例举 3 例？

答：浏览器兼容性，hack 技术，Node.js，Angular.js，Vue，react 等。

13、现有 2 个空水壶，容积分别为 5 升和 6 升，如何利用这两水壶取出 3 升水，假设水无限？

答： 假设有 A，B 两只壶，A 壶的容积为 5 升，B 壶的容积为 6 升，

第一步：将 B 壶装满水，倒入 A 壶中，此时 A 壶满，B 壶还剩一升

第二步：将 A 壶水到掉，将 B 壶水倒入，此时 A 壶为一升，B 壶空

第三步：将 B 壶装满，倒入 A 壶中，此时 A 壶满，B 壶还剩两升

第四步：重复第二步的操作，此时 A 壶剩两升，B 壶空

第五步：重复第三步的操作，此时 A 壶满，B 壶还剩三升

14、小明有 100 元去买汽水，汽水三元一瓶，正好小店有个促销活动，就是一个空瓶可以换 1 元钱，假设小明足够能喝，问他最多可以喝多少瓶汽水，还剩多少钱或空瓶？

答：49 瓶汽水，还剩 1 元