javascript

1、闭包  
闭包就是能够读取其他函数内部变量的函数  
闭包是指有权访问另一个函数作用域中变量的函数，创建闭包的最常见的方式就是在一个函数内创建另一个函数，通过另一个函数访问这个函数的局部变量,利用闭包可以突破作用链域  
  
闭包的特性：  
函数内再嵌套函数  
内部函数可以引用外层的参数和变量  
参数和变量不会被垃圾回收机制回收  
  
说说你对闭包的理解  
使用闭包主要是为了设计私有的方法和变量。闭包的优点是可以避免全局变量的污染，缺点是闭包会常驻内存，会增大内存使用量，使用不当很容易造成内存泄露。在js中，函数即闭包，只有函数才会产生作用域的概念  
  
闭包 的最大用处有两个，一个是可以读取函数内部的变量，另一个就是让这些变量始终保持在内存中  
闭包的另一个用处，是封装对象的私有属性和私有方法  
  
好处：能够实现封装和缓存等；  
坏处：就是消耗内存、不正当使用会造成内存溢出的问题  
  
使用闭包的注意点  
由于闭包会使得函数中的变量都被保存在内存中，内存消耗很大，所以不能滥用闭包，否则会造成网页的性能问题，在IE中可能导致内存泄露  
解决方法是，在退出函数之前，将不使用的局部变量全部删除  
  
  
2、说说你对作用域链的理解  
作用域链的作用是保证执行环境里有权访问的变量和函数是有序的，作用域链的变量只能向上访问，变量访问到window对象即被终止，作用域链向下访问变量是不被允许的  
简单的说，作用域就是变量与函数的可访问范围，即作用域控制着变量与函数的可见性和生命周期  
  
3、JavaScript原型，原型链 ? 有什么特点？  
每个对象都会在其内部初始化一个属性，就是prototype(原型)，当我们访问一个对象的属性时  
如果这个对象内部不存在这个属性，那么他就会去prototype里找这个属性，这个prototype又会有自己的prototype，于是就这样一直找下去，也就是我们平时所说的原型链的概念  
关系：instance.constructor.prototype = instance.\_\_proto\_\_  
特点：  
JavaScript对象是通过引用来传递的，我们创建的每个新对象实体中并没有一份属于自己的原型副本。当我们修改原型时，与之相关的对象也会继承这一改变  
当我们需要一个属性的时，Javascript引擎会先看当前对象中是否有这个属性， 如果没有的  
就会查找他的Prototype对象是否有这个属性，如此递推下去，一直检索到 Object 内建对象  
  
  
4、请解释什么是事件代理  
事件代理（Event Delegation），又称之为事件委托。是 JavaScript 中常用绑定事件的常用技巧。顾名思义，“事件代理”即是把原本需要绑定的事件委托给父元素，让父元素担当事件监听的职务。事件代理的原理是DOM元素的事件冒泡。使用事件代理的好处是可以提高性能  
可以大量节省内存占用，减少事件注册，比如在table上代理所有td的click事件就非常棒  
可以实现当新增子对象时无需再次对其绑定  
  
5、Javascript如何实现继承？  
构造继承  
原型继承  
实例继承  
拷贝继承  
  
原型prototype机制或apply和call方法去实现较简单，建议使用构造函数与原型混合方式  
  
<**script**>  
 *//例子* **function** *Parent*(){  
 **this**.**name** = **'wang'**;  
 }  
 **function** *Child*(){  
 **this**.**age** = 28;  
 }  
 *Child*.**prototype** = **new** *Parent*();*//继承了Parent，通过原型* **var *demo*** = **new** *Child*();  
 alert(***demo***.**age**);  
 alert(***demo***.**name**);*//得到被继承的属性*</**script**>  
  
  
6、谈谈This对象的理解  
this总是指向函数的直接调用者（而非间接调用者）;如果有new关键字，this指向new出来的那个对象;在事件中，this指向触发这个事件的对象，特殊的是，IE中的attachEvent中的this总是指向全局对象Window  
  
7、事件模型  
W3C中定义事件的发生经历三个阶段：捕获阶段（capturing）、目标阶段（targetin）、冒泡阶段（bubbling）  
冒泡型事件：当你使用事件冒泡时，子级元素先触发，父级元素后触发  
捕获型事件：当你使用事件捕获时，父级元素先触发，子级元素后触发  
DOM事件流：同时支持两种事件模型：捕获型事件和冒泡型事件  
阻止冒泡：在W3c中，使用stopPropagation（）方法；在IE下设置cancelBubble = true  
阻止捕获：阻止事件的默认行为，例如click - a后的跳转。在W3c中，使用preventDefault（）方法，在IE下设置window.event.returnValue = false。  
  
8、new操作符具体干了什么呢?  
创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型  
属性和方法被加入到 this 引用的对象中  
新创建的对象由 this 所引用，并且最后隐式的返回 this  
  
9、Ajax原理  
Ajax的原理简单来说是在用户和服务器之间加了—个中间层(AJAX引擎)，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。使用户操作与服务器响应异步化。这其中最关键的一步就是从服务器获得请求数据;Ajax的过程只涉及JavaScript、XMLHttpRequest和DOM。XMLHttpRequest是ajax的核心机制  
*//例子1. 创建连接* **var *xhr*** = **null**;  
 ***xhr*** = **new** XMLHttpRequest()  
 *// 2. 连接服务器* ***xhr***.open(**'get'**, url, **true**)  
 *// 3. 发送请求* ***xhr***.send(**null**);  
 *// 4. 接受请求* ***xhr***.**onreadystatechange** = **function**(){  
 **if**(***xhr***.**readyState** == 4){  
 **if**(***xhr***.**status** == 200){  
 success(***xhr***.**responseText**);  
 } **else** { *// fail* fail && fail(***xhr***.**status**);  
 }  
 }  
 }  
ajax 有那些优缺点?  
优点：  
通过异步模式，提升了用户体验.  
优化了浏览器和服务器之间的传输，减少不必要的数据往返，减少了带宽占用.  
Ajax在客户端运行，承担了一部分本来由服务器承担的工作，减少了大用户量下的服务器负载。  
Ajax可以实现动态不刷新（局部刷新）  
缺点：  
安全问题 AJAX暴露了与服务器交互的细节。  
对搜索引擎的支持比较弱。  
不容易调试。  
  
  
10、如何解决跨域问题?  
jsonp、 iframe、window.name、window.postMessage、服务器上设置代理页面  
  
  
11、模块化开发怎么做？  
立即执行函数,不暴露私有成员  
<**script**>  
 **var *module1*** = (**function**(){  
 **var** \_count = 0;  
 **var** *m1* = **function**(){  
 *//...* };  
 **var** *m2* = **function**(){  
 *//...* };  
 **return** {  
 m1 : *m1*,  
 m2 : *m2* };  
 })();  
</**script**>  
  
  
12、异步加载JS的方式有哪些？  
defer，只支持IE  
async：创建script，插入到DOM中，加载完毕后callBack  
  
13、那些操作会造成内存泄漏？  
内存泄漏指任何对象在您不再拥有或需要它之后仍然存在  
setTimeout 的第一个参数使用字符串而非函数的话，会引发内存泄漏  
闭包使用不当  
  
  
14、XML和JSON的区别？  
数据体积方面  
JSON相对于XML来讲，数据的体积小，传递的速度更快些。  
数据交互方面  
JSON与JavaScript的交互更加方便，更容易解析处理，更好的数据交互  
数据描述方面  
JSON对数据的描述性比XML较差  
传输速度方面  
JSON的速度要远远快于XML  
  
  
15、谈谈你对webpack的看法  
WebPack 是一个模块打包工具，你可以使用WebPack管理你的模块依赖，并编绎输出模块们所需的静态文件。  
它能够很好地管理、打包Web开发中所用到的HTML、Javascript、CSS以及各种静态文件（图片、字体等），让开发过程更加高效。  
对于不同类型的资源，webpack有对应的模块加载器。webpack模块打包器会分析模块间的依赖关系，最后 生成了优化且合并后的静态资源  
  
  
  
16、说说你对AMD和Commonjs的理解  
CommonJS是服务器端模块的规范，Node.js采用了这个规范。CommonJS规范加载模块是同步的，也就是说，只有加载完成，才能执行后面的操作。AMD规范则是非同步加载模块，允许指定回调函数  
AMD推荐的风格通过返回一个对象做为模块对象，CommonJS的风格通过对module.exports或exports的属性赋值来达到暴露模块对象的目的  
  
  
  
17、常见web安全及防护原理  
sql注入原理  
就是通过把SQL命令插入到Web表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的SQL命令  
总的来说有以下几点  
永远不要信任用户的输入，要对用户的输入进行校验，可以通过正则表达式，或限制长度，对单引号和双"-"进行转换等  
永远不要使用动态拼装SQL，可以使用参数化的SQL或者直接使用存储过程进行数据查询存取  
永远不要使用管理员权限的数据库连接，为每个应用使用单独的权限有限的数据库连接  
不要把机密信息明文存放，请加密或者hash掉密码和敏感的信息  
  
XSS原理及防范  
Xss(cross-site scripting)攻击指的是攻击者往Web页面里插入恶意html标签或者javascript代码。比如：攻击者在论坛中放一个看似安全的链接，  
骗取用户点击后，窃取cookie中的用户私密信息；或者攻击者在论坛中加一个恶意表单，当用户提交表单的时候，却把信息传送到攻击者的服务器中，而不是用户原本以为的信任站点  
  
XSS防范方法  
首先代码里对用户输入的地方和变量都需要仔细检查长度和对”<”,”>”,”;”,”’”等字符做过滤；  
其次任何内容写到页面之前都必须加以encode，避免不小心把html tag 弄出来。这一个层面做好，至少可以堵住超过一半的XSS 攻击  
  
XSS与CSRF有什么区别吗？  
XSS是获取信息，不需要提前知道其他用户页面的代码和数据包。CSRF是代替用户完成指定的动作，需要知道其他用户页面的代码和数据包。要完成一次CSRF攻击，受害者必须依次完成两个步骤  
登录受信任网站A，并在本地生成Cookie;在不登出A的情况下，访问危险网站B  
  
CSRF的防御  
服务端的CSRF方式方法很多样，但总的思想都是一致的，就是在客户端页面增加伪随机数  
通过验证码的方法  
  
  
18、用过哪些设计模式？  
工厂模式：  
工厂模式解决了重复实例化的问题，但还有一个问题,那就是识别问题，因为根本无法  
主要好处就是可以消除对象间的耦合，通过使用工程方法而不是new关键字  
构造函数模式  
使用构造函数的方法，即解决了重复实例化的问题，又解决了对象识别的问题，该模式与工厂模式的不同之处在于  
直接将属性和方法赋值给 this对象;  
  
  
19、为什么要有同源限制？  
同源策略指的是：协议，域名，端口相同，同源策略是一种安全协议  
举例说明：比如一个黑客程序，他利用Iframe把真正的银行登录页面嵌到他的页面上，当你使用真实的用户名，密码登录时，他的页面就可以通过Javascript读取到你的表单中input中的内容，这样用户名，密码就轻松到手了。  
  
  
20、offsetWidth/offsetHeight,clientWidth/clientHeight与scrollWidth/scrollHeight的区别  
offsetWidth/offsetHeight返回值包含content + padding + border，效果与e.getBoundingClientRect()相同  
clientWidth/clientHeight返回值只包含content + padding，如果有滚动条，也不包含滚动条  
scrollWidth/scrollHeight返回值包含content + padding + 溢出内容的尺寸  
  
21、javascript有哪些方法定义对象  
对象字面量： var obj = {};  
构造函数： var obj = new Object();  
Object.create(): var obj = Object.create(Object.prototype);  
  
22、常见兼容性问题？  
png24位的图片在iE6浏览器上出现背景，解决方案是做成PNG8  
浏览器默认的margin和padding不同。解决方案是加一个全局的\*{margin:0;padding:0;}来统一,，但是全局效率很低，一般是如下这样解决：  
body,ul,li,ol,dl,dt,dd,form,input,h1,h2,h3,h4,h5,h6,p{  
margin:0;padding:0;}  
IE下,event对象有x,y属性,但是没有pageX,pageY属性  
Firefox下,event对象有pageX,pageY属性,但是没有x,y属性.  
  
  
22、说说你对promise的了解  
依照 Promise/A+ 的定义，Promise 有四种状态：  
pending: 初始状态, 非 fulfilled 或 rejected.  
fulfilled: 成功的操作.  
rejected: 失败的操作.  
settled: Promise已被fulfilled或rejected，且不是pending  
另外， fulfilled与 rejected一起合称 settled  
Promise 对象用来进行延迟(deferred) 和异步(asynchronous) 计算  
  
构造一个 Promise，最基本的用法如下：  
 *//例子* **var** promise = **new** Promise(**function**(resolve, reject) {  
 **if** (...) { *// succeed* resolve(result);  
 } **else** { *// fails* reject(Error(errMessage));  
 }  
 });

Promise 实例拥有 then 方法（具有 then 方法的对象，通常被称为thenable）。它的使用方法如下：  
promise.then(onFulfilled, onRejected)  
接收两个函数作为参数，一个在 fulfilled 的时候被调用，一个在rejected的时候被调用，接收参数就是 future，onFulfilled 对应resolve, onRejected对应 reject  
  
  
23、你觉得jQuery源码有哪些写的好的地方  
jquery源码封装在一个匿名函数的自执行环境中，有助于防止变量的全局污染，然后通过传入window对象参数，可以使window对象作为局部变量使用，好处是当jquery中访问window对象的时候，就不用将作用域链退回到顶层作用域了，从而可以更快的访问window对象。同样，传入undefined参数，可以缩短查找undefined时的作用域链  
jquery将一些原型属性和方法封装在了jquery.prototype中，为了缩短名称，又赋值给了jquery.fn，这是很形象的写法  
有一些数组或对象的方法经常能使用到，jQuery将其保存为局部变量以提高访问速度  
jquery实现的链式调用可以节约代码，所返回的都是同一个对象，可以提高代码效率  
  
  
24、vue、react、angular  
Vue.js  
一个用于创建 web 交互界面的库，是一个精简的 MVVM。它通过双向数据绑定把 View 层和 Model 层连接了起来。实际的 DOM 封装和输出格式都被抽象为了Directives 和 Filters  
  
AngularJS  
是一个比较完善的前端MVVM框架，包含模板，数据双向绑定，路由，模块化，服务，依赖注入等所有功能，模板功能强大丰富，自带了丰富的 Angular指令  
  
react  
React 仅仅是 VIEW 层是facebook公司。推出的一个用于构建UI的一个库，能够实现服务器端的渲染。用了virtual dom，所以性能很好。  
  
  
25、Node的应用场景  
特点：  
1、它是一个Javascript运行环境  
2、依赖于Chrome V8引擎进行代码解释  
3、事件驱动  
4、非阻塞I/O  
5、单进程，单线程  
优点：  
高并发（最重要的优点）  
缺点：  
1、只支持单核CPU，不能充分利用CPU  
2、可靠性低，一旦代码某个环节崩溃，整个系统都崩溃  
  
  
26、谈谈你对AMD、CMD的理解  
CommonJS是服务器端模块的规范，Node.js采用了这个规范。CommonJS规范加载模块是同步的，也就是说，只有加载完成，才能执行后面的操作。AMD规范则是非同步加载模块，允许指定回调函数  
AMD推荐的风格通过返回一个对象做为模块对象，CommonJS的风格通过对module.exports或exports的属性赋值来达到暴露模块对象的目的  
  
  
27、那些操作会造成内存泄漏？  
内存泄漏指任何对象在您不再拥有或需要它之后仍然存在  
setTimeout 的第一个参数使用字符串而非函数的话，会引发内存泄漏  
闭包、控制台日志、循环（在两个对象彼此引用且彼此保留时，就会产生一个循环）  
  
  
28、web开发中会话跟踪的方法有哪些  
cookie，session，url重写，隐藏input，ip地址  
  
29、介绍js的基本数据类型  
Undefined、Null、Boolean、Number、String  
  
30、介绍js有哪些内置对象？  
Object 是 JavaScript 中所有对象的父对象  
数据封装类对象：Object、Array、Boolean、Number 和 String  
其他对象：Function、Arguments、Math、Date、RegExp、Error  
  
31、说几条写JavaScript的基本规范？  
不要在同一行声明多个变量  
请使用===/!==来比较true/false或者数值  
使用对象字面量替代new Array这种形式  
不要使用全局函数  
Switch语句必须带有default分支  
If语句必须使用大括号  
for-in循环中的变量 应该使用var关键字明确限定作用域，从而避免作用域污  
  
  
32、JavaScript有几种类型的值？  
栈：原始数据类型（Undefined，Null，Boolean，Number、String）  
堆：引用数据类型（对象、数组和函数）  
两种类型的区别是：存储位置不同；  
原始数据类型直接存储在栈(stack)中的简单数据段，占据空间小、大小固定，属于被频繁使用数据，所以放入栈中存储；  
引用数据类型存储在堆(heap)中的对象,占据空间大、大小不固定,如果存储在栈中，将会影响程序运行的性能；引用数据类型在栈中存储了指针，该指针指向堆中该实体的起始地址。当解释器寻找引用值时，会首先检索其  
在栈中的地址，取得地址后从堆中获得实体  
  
  
  
33、javascript创建对象的几种方式？  
javascript创建对象简单的说,无非就是使用内置对象或各种自定义对象，当然还可以用JSON；但写法有很多种，也能混合使用  
  
对象字面量的方式：person={firstname:"Mark",lastname:"Yun",age:25,eyecolor:"black"};  
  
用function来模拟无参的构造函数  
 **function** *Person*(){}  
 **var *person***=**new** *Person*();*//定义一个function，如果使用new"实例化",该function可以看作是一个Class* ***person***.**name**=**"Mark"**;  
 ***person***.**age**=**"25"**;  
 ***person***.work=**function**(){  
 alert(***person***.**name**+**" hello..."**);  
 }  
 ***person***.work();

用function来模拟参构造函数来实现（用this关键字定义构造的上下文属性）  
  
 **function** *Pet*(name,age,hobby){  
 **this**.**name**=name;*//this作用域：当前对象* **this**.**age**=age;  
 **this**.**hobby**=hobby;  
 **this**.eat=**function**(){  
 alert(**"我叫"**+**this**.**name**+**",我喜欢"**+**this**.**hobby**+**",是个程序员"**);  
 }  
 }  
 **var *maidou*** =**new** *Pet*(**"麦兜"**,25,**"coding"**);*//实例化、创建对象* ***maidou***.eat();*//调用eat方法*  
  
用工厂方式来创建（内置对象）  
<**script**>  
 **var *wcDog*** =**new** Object();  
 ***wcDog***.**name**=**"旺财"**;  
 ***wcDog***.**age**=3;  
 ***wcDog***.work=**function**(){  
 alert(**"我是"**+***wcDog***.**name**+**",汪汪汪......"**);  
 }  
 ***wcDog***.work();  
</**script**>  
  
用原型方式来创建  
<**script**>  
 **function** *Dog*(){  
 }  
 *Dog*.**prototype**.**name**=**"旺财"**;  
 *Dog*.**prototype**.eat=**function**(){  
 alert(**this**.**name**+**"是个吃货"**);  
 }  
 **var *wangcai*** =**new** *Dog*();  
 ***wangcai***.eat();  
</**script**>

用混合方式来创建  
<**script**>  
 **function** *Car*(name,price){  
 **this**.**name**=name;  
 **this**.**price**=price;  
 }  
 *Car*.**prototype**.sell=**function**(){  
 alert(**"我是"**+**this**.**name**+**"，我现在卖"**+**this**.**price**+**"万元"**);  
 }  
 **var *camry*** =**new** *Car*(**"凯美瑞"**,27);  
 ***camry***.sell();  
</**script**>  
  
34、eval是做什么的？  
它的功能是把对应的字符串解析成JS代码并运行  
应该避免使用eval，不安全，非常耗性能（2次，一次解析成js语句，一次执行）  
由JSON字符串转换为JSON对象的时候可以用eval，var obj =eval('('+ str +')')  
  
35、null，undefined 的区别？  
undefined 表示不存在这个值。  
undefined :是一个表示”无”的原始值或者说表示”缺少值”，就是此处应该有一个值，但是还没有定义。当尝试读取时会返回 undefined  
例如变量被声明了，但没有赋值时，就等于undefined  
null 表示一个对象被定义了，值为“空值”  
null : 是一个对象(空对象, 没有任何属性和方法)  
例如作为函数的参数，表示该函数的参数不是对象；  
在验证null时，一定要使用　=== ，因为 ==无法分别null 和　undefined  
  
36、[“1”, “2”, “3”].map(parseInt) 答案是多少？  
[1, NaN, NaN]因为 parseInt 需要两个参数 (val, radix)，其中radix 表示解析时用的基数。  
map传了 3个(element, index, array)，对应的 radix 不合法导致解析失败。  
  
37、javascript 代码中的”use strict”;是什么意思 ? 使用它区别是什么？  
use strict是一种ECMAscript 5 添加的（严格）运行模式,这种模式使得 Javascript 在更严格的条件下运行,使JS编码更加规范化的模式,消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为  
  
  
  
38、JSON 的了解？\*\*  
JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式  
它是基于JavaScript的一个子集。数据格式简单, 易于读写, 占用带宽小  
JSON字符串转换为JSON对象:  
<**script**>  
 **var *obj*** =eval(**'('**+ str +**')'**);  
 **var *obj*** = str.parseJSON();  
 **var *obj*** = **JSON**.parse(str);  
</**script**>  
JSON对象转换为JSON字符串：  
<**script**>  
 **var *last***=***obj***.toJSONString();  
 **var *last***=**JSON**.stringify(***obj***);  
</**script**>  
  
39、js延迟加载的方式有哪些？  
defer和async、动态创建DOM方式（用得最多）、按需异步载入js  
  
40、同步和异步的区别?  
同步：浏览器访问服务器请求，用户看得到页面刷新，重新发请求,等请求完，页面刷新，新内容出现，用户看到新内容,进行下一步操作  
异步：浏览器访问服务器请求，用户正常操作，浏览器后端进行请求。等请求完，页面不刷新，新内容也会出现，用户看到新内容  
  
41、渐进增强和优雅降级  
渐进增强 ：针对低版本浏览器进行构建页面，保证最基本的功能，然后再针对高级浏览器进行效果、交互等改进和追加功能达到更好的用户体验。  
优雅降级 ：一开始就构建完整的功能，然后再针对低版本浏览器进行兼容  
  
42、defer和async  
defer并行加载js文件，会按照页面上script标签的顺序执行  
async并行加载js文件，下载完成立即执行，不会按照页面上script标签的顺序执行  
  
43、说说严格模式的限制  
变量必须声明后再使用  
函数的参数不能有同名属性，否则报错  
不能使用with语句  
禁止this指向全局对象  
  
  
44、attribute和property的区别是什么？  
attribute是dom元素在文档中作为html标签拥有的属性；  
property就是dom元素在js中作为对象拥有的属性。  
对于html的标准属性来说，attribute和property是同步的，是会自动更新的  
但是对于自定义的属性来说，他们是不同步的  
  
45、谈谈你对ES6的理解  
新增模板字符串（为JavaScript提供了简单的字符串插值功能）  
箭头函数  
for-of（用来遍历数据—例如数组中的值。）  
arguments对象可被不定参数和默认参数完美代替。  
ES6将promise对象纳入规范，提供了原生的Promise对象。  
增加了let和const命令，用来声明变量。  
增加了块级作用域。  
let命令实际上就增加了块级作用域。  
还有就是引入module模块的概念  
  
  
46、ECMAScript6 怎么写class么，为什么会出现class这种东西?  
这个语法糖可以让有OOP基础的人更快上手js，至少是一个官方的实现了  
但对熟悉js的人来说，这个东西没啥大影响；一个Object.creat()搞定继承，比class简洁清晰的多  
  
47、什么是面向对象编程及面向过程编程，它们的异同和优缺点  
面向过程就是分析出解决问题所需要的步骤，然后用函数把这些步骤一步一步实现，使用的时候一个一个依次调用就可以了  
面向对象是把构成问题事务分解成各个对象，建立对象的目的不是为了完成一个步骤，而是为了描叙某个事物在整个解决问题的步骤中的行为  
面向对象是以功能来划分问题，而不是步骤  
  
48、面向对象编程思想  
基本思想是使用对象，类，继承，封装等基本概念来进行程序设计  
优点  
易维护  
采用面向对象思想设计的结构，可读性高，由于继承的存在，即使改变需求，那么维护也只是在局部模块，所以维护起来是非常方便和较低成本的  
易扩展  
开发工作的重用性、继承性高，降低重复工作量。  
缩短了开发周期  
  
  
49、对web标准、可用性、可访问性的理解  
可用性（Usability）：产品是否容易上手，用户能否完成任务，效率如何，以及这过程中用户的主观感受可好，是从用户的角度来看产品的质量。可用性好意味着产品质量高，是企业的核心竞争力  
可访问性（Accessibility）：Web内容对于残障用户的可阅读和可理解性  
可维护性（Maintainability）：一般包含两个层次，一是当系统出现问题时，快速定位并解决问题的成本，成本低则可维护性好。二是代码是否容易被人理解，是否容易修改和增强功能。