### 1、请解释事件代理 (event delegation)

事件委托技术能让你避免对特定的每个节点添加事件监听器；相反，事件监听器是被添加到它们的父元素上。事件监听器会分析从子元素冒泡上来的事件，找到是哪个子元素的事件。

**优点：**

性能得到了优化（需要创建的以及驻留在内存中的事件处理器少了）

动态添加的元素也能绑定事件了

### 2、请解释 JavaScript 中 this 是如何工作的

this 永远指向函数运行时所在的对象，而不是函数被创建时所在的对象。

函数调用es3和非严格es5为全局对象，严格es5为undefined

方法调用this指向调用该方法的对象（调用上下文）

构造函数时，this指向新创建的对象

call() apply() 调用方法时，this指向调用方法的对象，而不是该方法拥有者对象。

### 3、请解释原型继承 (prototypal inheritance) 的原理？

原型继承的基础是原型链查找。

原型链查找基本概念：  
每一个函数 F 都有一个原型对象（prototype）F.prototype  
每一个函数都可以通过 new 关键字化身成为一个类构造函数，new F 会产生一个对象 O  
在调用对象的某个属性或者方法，比如 http://O.xxx 的时候，会首先查找对象自身是否有这个方法或者属性，如果没找到就会去对象的构造函数的原型对象中查找（注意有两个定语），也就是查找 O 的构造函数 F 的原型对象 http://F.prototype.xxx  
F.prototype 也是一个对象，查找 http://F.prototype.xxx 的时候会重复第 3 步的过程

**4、请解释为什么接下来这段代码不是 IIFE (立即调用的函数表达式)：function foo(){ }();要做哪些改动使它变成 IIFE?**

　这里只是声明一个叫foo的function，直接用()执行这样是不成功的，想要变成IIFE就要把声明变成表达式，就可以立即执行了，可以这样(function foo(){})()或者(function foo(){}())，这就是用括号把定义强转成表达式，当然还有其他方法，关键就是声明不可以执行，表达式才可以执行。

**5、描述以下变量的区别：null，undefined 或 undeclared？ 该如何检测它们？**

undefined:未定义，在变量没有赋值的时候的值即为undefined。“缺少值”，就是此处应该有一个值，但是还没有定义。  
underclared:即为被污染的命名，访问没有被声明的变量，会抛出异常，终止执行。尝试访问一个undeclared的变量，浏览器会报错，JS执行会中断。  
null:是一个空的对象引用。“没有对象”，即该处不应该有值

区别:  
undefined和null在if语句中，都会被自动转为false，相等运算符甚至直接报告两者相等。typeof undefined会返回undefined ，而typeof null 总返回 object(typeof有六种可能：“number”、“string”、“boolean”、“object”、“function”、“undefined”)

false == undefined;//false

false == null;//false

null == undefined;//true

该如何检测它们？

var obj;

obj ===undefined; //检测undfined 方法一

typeof obj === ‘undefined’;//检测undefined方法2

obj = null;

obj === null;//来检测null

typeof null;//‘object’

### 6、什么是闭包 (closure)，如何使用它，为什么要使用它？

定义：闭包就是可以读取到其他函数内部变量的函数。  
闭包的用途：

可以读取函数内部的变量。(外界无法访问函数的内部的私有方法和变量，只能通过提供的接口访问)

让变量的值始终保持在内存中。

可以避免污染全局变量，实现私有方法或者变量等

注意：

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。

闭包会在父函数外部，改变父函数内部变量的值。所以，如果你把父函数当作对象（object）使用，把闭包当作它的公用方法（Public Method），把内部变量当作它的私有属性（private value），这时一定要小心，不要随便改变父函数内部变量的值。

### 7、请举出一个匿名函数的典型用例？

匿名函数可以用作回调函数执行，可以防止全局变量污染。  
在 JS 框架中常使用匿名函数来避免全局变量的污染。  
$.(“input”).each(function(e){this.val(‘OK’)});  
(function(){})();  
$(document).ready(function(){ });  
$(function() {})

### 8、你是如何组织自己的代码？是使用模块模式，还是使用经典继承的方法？ 请指出 JavaScript 宿主对象 (host objects) 和原生对象 (native objects) 的区别？

原生对象:独立于宿主环境的 ECMAScript 实现提供的对象。为array obj regexp date function等可以new实例化的对象。

内置对象:为gload Math 等，开发者不必明确实例化内置对象，它已被实例化了。类似于isNaN()、parseInt()和parseFloat()方法等，看起来都是函数，而实际上，它们都是Global对象的方法。具体可以参考 JavaScript 全局对象

宿主对象:即由 ECMAScript 实现的宿主环境（操作系统和浏览器）提供的对象。所有的BOM和DOM对象都是宿主对象。因为其对于不同的“宿主”环境所展示的内容不同（这就是兼容性和特性检测的缘由）。ECMAScript官方未定义的对象都属于宿主对象。

### 9、请指出以下代码的区别：function Person(){}、var person = Person()、var person = new Person()？

第一个为函数声明，第二个将函数person()返回值赋值给person,第三个通过Person()的构造器创建了一个对象让person变量引用该对象；

### 10、.call 和 .apply 的区别是什么？

call和apply都是调用一个对象的一个方法，以另一个对象替换当前对象。它们都属于Function.prototype的一个方法，所以每个function实例都有call和apply属性。这两个方法可以用来代替另一个对象调用一个方法，可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。

区别:  
两者传递的参数不同，虽然函数第一个参数都是要传入给当前对象的对象，但是，apply的第二个参数是一个参数数组，将多个参数组合成为一个数组传入；而call第二个参数则是直接的参数列表。

### 11、请解释 Function.prototype.bind？

Function.prototype.bind()其实就是函数绑定。函数的接收者取决于他是如何被调用，可以通过调用.bind()给函数绑定作用域上下文（this的值），即函数的接收者。

var foo = { x: 3}

var bar = function(){console.log( this.x);}

bar(); // undefinedvar

boundFunc = bar.bind(foo);//隐式看作是在foo作用域里调用bar方法

boundFunc(); // 3

.bind()创建了一个函数，当这个函数在被调用的时候，它的 this 关键词会被设置成被传入的值（这里指调用bind()时传入的参数）也就是我们传入想要的上下文。 简单的用法： 关于 Function.prototype.bind() 内部，这里有个非常简单的例子：

Function.prototype.bind = function (scope) {

var fn = this;

return function () {

return fn.apply(scope);//使用call效果一样

};

}

### 12、在什么时候你会使用 document.write()？

document.write()方法可以用在两个方面：

页面载入过程中用实时脚本创建页面内容，该方法需要一个字符串参数，它是写到窗口或框架中的HTML内容。

以及用延时脚本创建本窗口或新窗口的内容。该方法需要一个字符串参数，它是写到窗口或框架中的HTML内容。

记住，在载入页面后，浏览器输出流自动关闭。在此之后，任何一个对当前页面进行操作的document.write()方法将打开—个新的输出流，它将清除当前页面内容(包括源文档的任何变量或值)。因此，假如希望用脚本生成的HTML替换当前页面，就必须把HTML内容连接起来赋给一个变量，使用一个document.write()方法完成写操作。不必清除文档并打开一个新数据流，一个document.write()调用就可完成所有的操作。

关于document.write()方法还有一点要说明的是它的相关方法document.close()。脚本向窗口(不管是本窗口或其他窗口)写完内容后，必须关闭输出流。在延时脚本的最后一个document.write()方法后面，必须确保含有document.close()方法，不这样做就不能显示图像和表单。并且，任何后面调用的document.write()方法只会把内容追加到页面后，而不会清除现有内容来写入新值。为了演示document.write()方法，我们提供了同一个应用程序的两个版本。

大多数生成的广告代码依旧使用 document.write()，虽然这种用法会让人很不爽。

### 13、请指出浏览器特性检测，特性推断和浏览器 UA 字符串嗅探的区别？

检测浏览器的特殊名称和版本（用户代理检测）即浏览器UA字符串嗅探。浏览器嗅探技术可以快捷的将代码进行分支，以便针对不同的浏览器应用不同的指令；针对特定浏览器的特定版本，超出范围之外都是不可靠的请解释事件代理 (event delegation)

事件委托技术能让你避免对特定的每个节点添加事件监听器；相反，事件监听器是被添加到它们的父元素上。事件监听器会分析从子元素冒泡上来的事件，找到是哪个子元素的事件。

优点：

性能得到了优化（需要创建的以及驻留在内存中的事件处理器少了）

动态添加的元素也能绑定事件了

### 14、请解释 JavaScript 中 this 是如何工作的

this 永远指向函数运行时所在的对象，而不是函数被创建时所在的对象。

函数调用es3和非严格es5为全局对象，严格es5为undefined

方法调用this指向调用该方法的对象（调用上下文）

构造函数时，this指向新创建的对象

call() apply() 调用方法时，this指向调用方法的对象，而不是该方法拥有者对象

### 15、请解释原型继承 (prototypal inheritance) 的原理？

原型继承的基础是原型链查找。

原型链查找基本概念：  
每一个函数 F 都有一个原型对象（prototype）F.prototype  
每一个函数都可以通过 new 关键字化身成为一个类构造函数，new F 会产生一个对象 O  
在调用对象的某个属性或者方法，比如 http://O.xxx 的时候，会首先查找对象自身是否有这个方法或者属性，如果没找到就会去对象的构造函数的原型对象中查找（注意有两个定语），也就是查找 O 的构造函数 F 的原型对象 http://F.prototype.xxx  
F.prototype 也是一个对象，查找 http://F.prototype.xxx 的时候会重复第 3 步的过程

### 16、请解释为什么接下来这段代码不是 IIFE (立即调用的函数表达式)：function foo(){ }();要做哪些改动使它变成 IIFE?

　这里只是声明一个叫foo的function，直接用()执行这样是不成功的，想要变成IIFE就要把声明变成表达式，就可以立即执行了，可以这样(function foo(){})()或者(function foo(){}())，这就是用括号把定义强转成表达式，当然还有其他方法，关键就是声明不可以执行，表达式才可以执行。

### 17、描述以下变量的区别：null，undefined 或 undeclared？ 该如何检测它们？

undefined:未定义，在变量没有赋值的时候的值即为undefined。“缺少值”，就是此处应该有一个值，但是还没有定义。  
underclared:即为被污染的命名，访问没有被声明的变量，会抛出异常，终止执行。尝试访问一个undeclared的变量，浏览器会报错，JS执行会中断。  
null:是一个空的对象引用。“没有对象”，即该处不应该有值

区别:  
undefined和null在if语句中，都会被自动转为false，相等运算符甚至直接报告两者相等。typeof undefined会返回undefined ，而typeof null 总返回 object(typeof有六种可能：“number”、“string”、“boolean”、“object”、“function”、“undefined”)

false == undefined;//false

false == null;//false

null == undefined;//true

该如何检测它们？

var obj;

obj ===undefined; //检测undfined 方法一

typeof obj === ‘undefined’;//检测undefined方法2

obj = null;

obj === null;//来检测null

typeof null;//‘object’

### 18、什么是闭包 (closure)，如何使用它，为什么要使用它？

定义：闭包就是可以读取到其他函数内部变量的函数。  
闭包的用途：

可以读取函数内部的变量。(外界无法访问函数的内部的私有方法和变量，只能通过提供的接口访问)

让变量的值始终保持在内存中。

可以避免污染全局变量，实现私有方法或者变量等

注意：

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。

闭包会在父函数外部，改变父函数内部变量的值。所以，如果你把父函数当作对象（object）使用，把闭包当作它的公用方法（Public Method），把内部变量当作它的私有属性（private value），这时一定要小心，不要随便改变父函数内部变量的值。

### 19、请举出一个匿名函数的典型用例？

匿名函数可以用作回调函数执行，可以防止全局变量污染。  
在 JS 框架中常使用匿名函数来避免全局变量的污染。  
$.(“input”).each(function(e){this.val(‘OK’)});  
(function(){})();  
$(document).ready(function(){ });  
$(function() {})

### 20、你是如何组织自己的代码？是使用模块模式，还是使用经典继承的方法？ 请指出 JavaScript 宿主对象 (host objects) 和原生对象 (native objects) 的区别？

原生对象:独立于宿主环境的 ECMAScript 实现提供的对象。为array obj regexp date function等可以new实例化的对象。

内置对象:为gload Math 等，开发者不必明确实例化内置对象，它已被实例化了。类似于isNaN()、parseInt()和parseFloat()方法等，看起来都是函数，而实际上，它们都是Global对象的方法。具体可以参考 JavaScript 全局对象

宿主对象:即由 ECMAScript 实现的宿主环境（操作系统和浏览器）提供的对象。所有的BOM和DOM对象都是宿主对象。因为其对于不同的“宿主”环境所展示的内容不同（这就是兼容性和特性检测的缘由）。ECMAScript官方未定义的对象都属于宿主对象。

### 21、请指出以下代码的区别：function Person(){}、var person = Person()、var person = new Person()？

第一个为函数声明，第二个将函数person()返回值赋值给person,第三个通过Person()的构造器创建了一个对象让person变量引用该对象；

### 22、.call 和 .apply 的区别是什么？

call和apply都是调用一个对象的一个方法，以另一个对象替换当前对象。它们都属于Function.prototype的一个方法，所以每个function实例都有call和apply属性。这两个方法可以用来代替另一个对象调用一个方法，可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。

区别:  
两者传递的参数不同，虽然函数第一个参数都是要传入给当前对象的对象，但是，apply的第二个参数是一个参数数组，将多个参数组合成为一个数组传入；而call第二个参数则是直接的参数列表。

### 23、请解释 Function.prototype.bind？

Function.prototype.bind()其实就是函数绑定。函数的接收者取决于他是如何被调用，可以通过调用.bind()给函数绑定作用域上下文（this的值），即函数的接收者。

var foo = { x: 3}

var bar = function(){console.log( this.x);}

bar(); // undefinedvar

boundFunc = bar.bind(foo);//隐式看作是在foo作用域里调用bar方法

boundFunc(); // 3

.bind()创建了一个函数，当这个函数在被调用的时候，它的 this 关键词会被设置成被传入的值（这里指调用bind()时传入的参数）也就是我们传入想要的上下文。 简单的用法： 关于 Function.prototype.bind() 内部，这里有个非常简单的例子：

Function.prototype.bind = function (scope) {

var fn = this;

return function () {

return fn.apply(scope);//使用call效果一样

};

}

### 24、在什么时候你会使用 document.write()？

document.write()方法可以用在两个方面：

页面载入过程中用实时脚本创建页面内容，该方法需要一个字符串参数，它是写到窗口或框架中的HTML内容。

以及用延时脚本创建本窗口或新窗口的内容。该方法需要一个字符串参数，它是写到窗口或框架中的HTML内容。

记住，在载入页面后，浏览器输出流自动关闭。在此之后，任何一个对当前页面进行操作的document.write()方法将打开—个新的输出流，它将清除当前页面内容(包括源文档的任何变量或值)。因此，假如希望用脚本生成的HTML替换当前页面，就必须把HTML内容连接起来赋给一个变量，使用一个document.write()方法完成写操作。不必清除文档并打开一个新数据流，一个document.write()调用就可完成所有的操作。

关于document.write()方法还有一点要说明的是它的相关方法document.close()。脚本向窗口(不管是本窗口或其他窗口)写完内容后，必须关闭输出流。在延时脚本的最后一个document.write()方法后面，必须确保含有document.close()方法，不这样做就不能显示图像和表单。并且，任何后面调用的document.write()方法只会把内容追加到页面后，而不会清除现有内容来写入新值。为了演示document.write()方法，我们提供了同一个应用程序的两个版本。

大多数生成的广告代码依旧使用 document.write()，虽然这种用法会让人很不爽。

### 25、请指出浏览器特性检测，特性推断和浏览器 UA 字符串嗅探的区别？

检测浏览器的特殊名称和版本（用户代理检测）即浏览器UA字符串嗅探。浏览器嗅探技术可以快捷的将代码进行分支，以便针对不同的浏览器应用不同的指令；针对特定浏览器的特定版本，超出范围之外都是不可靠的

### 26、请尽可能详尽的解释 Ajax 的工作原理？ 使用 Ajax 都有哪些优劣？

优势：可以刷新局部页面，而不用整体页面都刷新  
缺点：用户禁用javascript的情况

### 27、请解释 JSONP 的工作原理，以及它为什么不是真正的 Ajax。

工作原理:JSONP动态创建script标签，回调函数。Ajax是页面无刷新请求数据操作，动态添加一个&ltscript>标签，而script标签的src属性是没有跨域的限制的。这样说来，这种跨域方式其实与ajax XmlHttpRequest协议无关了。  
当GET请求从被调用页面返回时，可以返回一段JavaScript代码，这段代码会自动调用主页面中的一个callback函数。

优点:不受同源策略的影响，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；并且在请求完毕后可以通过调用callback的方式回传结果 　　  
缺点: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。

### 28、请解释变量声明提升 (hoisting)

在JavaScript代码运行之前其实是有一个编译阶段的。编译之后才是从上到下，一行一行解释执行。变量提升就发生在编译阶段，它把变量和函数的声明提升至作用域的顶端。（编译阶段的工作之一就是将变量与其作用域进行关联）。

变量提升需要注意两点：

提升的部分只是变量声明，赋值语句和可执行的代码逻辑还保持在原地不动

提升只是将变量声明提升到变量所在的变量范围的顶端，并不是提升到全局范围

函数声明：

变量声明和函数声明都会得到变量提升，但函数声明会最先得到提升，然后是变量声明（函数是一等公民）

对于函数声明来说，如果定义了相同的函数变量声明，后定义的声明会覆盖掉先前的声明

### 29、请描述事件冒泡机制 (event bubbling)

从目标元素开始，往顶层元素传播。途中如果有节点绑定了相应的事件处理函数，这些函数都会被依次触发。如果想阻止事件起泡，可以使用e.stopPropagation()（Firefox）或者e.cancelBubble=true（IE）来组织事件的冒泡传播

### 30、“attribute” 和 “property” 的区别是什么？

DOM元素的attribute和property两者是不同的东西。attribute翻译为“特性”，property翻译为“属性”。

attribute是一个特性节点，每个DOM元素都有一个对应的attributes属性来存放所有的attribute节点，attributes是一个类数组的容器，说得准确点就是NameNodeMap，不继承于Array.prototype，不能直接调用Array的方法。attributes的每个数字索引以名值对(name=”value”)的形式存放了一个attribute节点。

property就是一个属性，如果把DOM元素看成是一个普通的Object对象，那么property就是一个以名值对(name=”value”)的形式存放在Object中的属性。要添加和删除property和普通的对象类似。

很多attribute节点还有一个相对应的property属性，比如上面的div元素的id和class既是attribute，也有对应的property，不管使用哪种方法都可以访问和修改。

总之，attribute节点都是在HTML代码中可见的，而property只是一个普通的名值对属性

### 31、为什么扩展 JavaScript 内置对象不是好的做法？

因为你不知道哪一天浏览器或javascript本身就会实现这个方法，而且和你扩展的实现有不一致的表现。到时候你的javascript代码可能已经在无数个页面中执行了数年，而浏览器的实现导致所有使用扩展原型的代码都崩溃了。

需要给Array原型添加一个distinct的方法，最好检查是否存在同名的方法，避免自定义方法覆盖原生方法：

Arrray.prototype.distinct = Arrray.prototype.distinct || function(){/…../}

### 32、请指出 document load 和 document DOMContentLoaded 两个事件的区别。

ready 表示文档的 DOM 已经加载完成（不包含图片、视频等资源）；load 表示整个网页加载完成。可以看出，ready 事件发生在 load 事件之前。

**33、== 和 === 有什么不同？**

如果两边的操作数具有一致的类型且拥有相同的值时，=== 返回 true，!== 返回 false。

### 34、请解释 JavaScript 的同源策略 (same-origin policy)。

同源策略限制了一个源（origin）中加载文本或脚本与来自其它源（origin）中资源的交互方式。

同源策略出于安全，不允许源 A 的脚本读取（read）源 B 的资源的内容，但却允许执行（execute）源 B 的资源。这个概念也有些拗口。简单说，有一个页面调用了 Google CDN 提供的 jQuery，以及其它 CDN 上的 Bootstrap JS、CSS 代码，虽然它们与我的博客不同源，但我可以用它们来操作这个页面，并应用样式，这是执行的概念。

### 35、如何实现下列代码：[1,2,3,4,5].duplicator(); // [1,2,3,4,5,1,2,3,4,5]

将此方法添加至 Array.prototype 实现，代码如下：

Array.prototype.duplicator = function(){

　　var l = this.length,i;

　　for(i=0;i<l;i++){

　　　this.push(this[i])

}

}

**36、请解释事件代理 (event delegation)**

事件委托技术能让你避免对特定的每个节点添加事件监听器；相反，事件监听器是被添加到它们的父元素上。事件监听器会分析从子元素冒泡上来的事件，找到是哪个子元素的事件。

优点：

性能得到了优化（需要创建的以及驻留在内存中的事件处理器少了）

动态添加的元素也能绑定事件了

**37、请解释 JavaScript 中 this 是如何工作的**

this 永远指向函数运行时所在的对象，而不是函数被创建时所在的对象。

函数调用es3和非严格es5为全局对象，严格es5为undefined

方法调用this指向调用该方法的对象（调用上下文）

构造函数时，this指向新创建的对象

call() apply() 调用方法时，this指向调用方法的对象，而不是该方法拥有者对象

**38、请解释原型继承 (prototypal inheritance) 的原理？**

原型继承的基础是原型链查找。

原型链查找基本概念：  
每一个函数 F 都有一个原型对象（prototype）F.prototype  
每一个函数都可以通过 new 关键字化身成为一个类构造函数，new F 会产生一个对象 O  
在调用对象的某个属性或者方法，比如 http://O.xxx 的时候，会首先查找对象自身是否有这个方法或者属性，如果没找到就会去对象的构造函数的原型对象中查找（注意有两个定语），也就是查找 O 的构造函数 F 的原型对象 http://F.prototype.xxx  
F.prototype 也是一个对象，查找 http://F.prototype.xxx 的时候会重复第 3 步的过程

**39、请解释为什么接下来这段代码不是 IIFE (立即调用的函数表达式)：function foo(){ }();要做哪些改动使它变成 IIFE?**

　这里只是声明一个叫foo的function，直接用()执行这样是不成功的，想要变成IIFE就要把声明变成表达式，就可以立即执行了，可以这样(function foo(){})()或者(function foo(){}())，这就是用括号把定义强转成表达式，当然还有其他方法，关键就是声明不可以执行，表达式才可以执行。

**40、描述以下变量的区别：null，undefined 或 undeclared？ 该如何检测它们？**

undefined:未定义，在变量没有赋值的时候的值即为undefined。“缺少值”，就是此处应该有一个值，但是还没有定义。  
underclared:即为被污染的命名，访问没有被声明的变量，会抛出异常，终止执行。尝试访问一个undeclared的变量，浏览器会报错，JS执行会中断。  
null:是一个空的对象引用。“没有对象”，即该处不应该有值

区别:  
undefined和null在if语句中，都会被自动转为false，相等运算符甚至直接报告两者相等。typeof undefined会返回undefined ，而typeof null 总返回 object(typeof有六种可能：“number”、“string”、“boolean”、“object”、“function”、“undefined”)

false == undefined;//false

false == null;//false

null == undefined;//true

该如何检测它们？

var obj;

obj ===undefined; //检测undfined 方法一

typeof obj === ‘undefined’;//检测undefined方法2

obj = null;

obj === null;//来检测null

typeof null;//‘object’

**41、什么是闭包 (closure)，如何使用它，为什么要使用它？**

定义：闭包就是可以读取到其他函数内部变量的函数。  
闭包的用途：

可以读取函数内部的变量。(外界无法访问函数的内部的私有方法和变量，只能通过提供的接口访问)

让变量的值始终保持在内存中。

可以避免污染全局变量，实现私有方法或者变量等

注意：

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。

闭包会在父函数外部，改变父函数内部变量的值。所以，如果你把父函数当作对象（object）使用，把闭包当作它的公用方法（Public Method），把内部变量当作它的私有属性（private value），这时一定要小心，不要随便改变父函数内部变量的值。

42、写出5 种以上ie6 bug 的解决方法，哪些你认为是解决起来最麻烦的？

float情况下有双边距的bug，使用display: inline解决

宽高为奇数时有bug，使用偶数

min-height设置不了，加!important

z-index问题，给父亲设置position：relative

设置高度小与10px左右的时候，实际高度高于设置高度，因为有默认行高，把行高也设置

44、什么是Web workers？为什么我们需要他？

一个运行在后台的JavaScript，有助于异步执行JavaScript，提高页面性能

46、写出至少三个CSS3中新添加的样式属性的名字及参数。

阴影，rgba，圆角

47、给一个元素添加CSS样式有哪几种方法说说他们的优先级。

引用css，内联样式，head里style标签内样式

49、为了让网页更快的加载，你会如何处理你的css文件和js文件，以及图片文件，页面性能优化你还知道哪些方法。

50、你经常遇到的浏览器兼容性有哪些？通常是怎么处理的。

51、HTML中div与span区别；

52、HTML5 存储类型有哪些，以及与他们的区别；

53、css实现垂直水平居中（口语描述）。

54、浏览器内核分别是什么；

55、列举5种以上表单元素中input的type类型；

56、alt和title分别表示什么含义以及具体应用体现；

57、css中position的属性值都有哪些？并描述其含义及具体解释；

58、对html语义化理解，以及语义化有哪些优点。

59、经常使用的页面[开发工具](http://msd.misuland.com/pd/3127746505234977086)级测试工具。

60、经常使用什么脚本库，开发或使用什么应用或组件；

61、使用css如何让一个宽度为200px的div水平居中，（要求兼容ie，可提供多种方法）；

62、简要画出盒模型，并描述关键要素的含义；

63、页面构造中你遇到过什么样的兼容问题，如何解决

64、Doctype作用？标准模式与兼容模式有什么区别；

65、html5有哪些新特性，移除了哪些元素；

66、介绍一下标准css盒子模型；

67、什么事css预处理器；

68、css3中你最常用的有哪些？说明用法。

69、document.write、innerHTML和innerText 的区别？

70、请说出至少三种减少页面加载时间的方法。（加载时间是指感知的时间或者实际加载 的时间）

71、请描述你曾使用过的css预处理器的优缺点

72、实现 class为test的div 在屏幕宽为400一下的宽度为200，屏幕宽为400~800的情况下宽 度为350；

73、实现当屏幕宽度大雨700,小于800是时引用外部样式style-7-9.css

74、HTML中input的6个新属性

75、列一个手机端可以实现手势滑动的插件或类库

76、对WEB标准以及W3C的理解与认识

77、xhtml和html有什么区别

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

79、行内元素有哪些?块级元素有哪些?CSS的盒模型?

80、CSS引入的方式有哪些? link和@import的区别是?

81、CSS选择符有哪些?哪些属性可以继承?优先级算法如何计算?内联和important哪个优先 级高?

82、标签选择符 类选择符 id选择符

83、标签上title与alt属性的区别是什么?

84、描述css reset的作用和用途

85、解释css sprites，如何使用。

86、浏览器标准模式和怪异模式之间的区别是什么?

87、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：

88、清除浮动的几种方式，各自的优缺点

89、CSS3有哪些新特性？

90、CSS中的class和id有什么区别。

91、请说一下移动端常见的适配不同屏幕大小的方法。

92、一个高宽未知的图片如何在一个比他大的容器内水平居中。

93、Table标签的作用是什么。

94、定义链接四种状态的伪类的正确书写顺序是？

95、你知道的css选择器有哪些。

96、遇到疑难问题时，你通常时如何解决的？

97、Inline，inline-block和block的区别是。

98、页面导入样式时，使用link和@import有什么区别。

99、CSS超过宽度的文字显示点点，必须要设置的属性。

100、设置文字阴影属性。设置表格边框合并属性。

101、文本强制换行。

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

103、超链接访问过后hover样式后会出现什么问题？如何解决。

104、前端页面有那三部分构成，其作用是什么。

105、写出至少三个css3中新添加的样式属性的名字及参数。

106、给一个元素添加css样式有哪几种方法说说他们的优先级。

107、多人项目中，你如何规划css文件样式命名。

108、为了让网页更快的加载，你会如何处理你的css文件和js文件，以及图片文件，页面性 能优化你还知道哪些方法。

109、你经常遇到的浏览器兼容性有哪些？通常是怎么处理的。

110、HTML中div与span区别；

111、HTML5 存储类型有哪些，以及与他们的区别；

112、css实现垂直水平居中。

113、浏览器内核分别是什么；

114、列举5种以上表单元素中input的type类型；

115、alt和title分别表示什么含义以及具体应用体现；

116、css中position的属性值都有哪些？并描述其含义及具体解释；

117、对html语义化理解，以及语义化有哪些优点。

118、经常使用的页面开发工具级测试工具。

119、经常使用什么脚本库，开发或使用什么应用或组件；

120、使用css如何让一个宽度为200px的div水平居中，（要求兼容ie，可提供多种方法）；

121、简要画出盒模型，并描述关键要素的含义；

122、页面构造中你遇到过什么样的兼容问题，如何解决

123、Doctype作用？标准模式与兼容模式有什么区别；

124、html5有哪些新特性，移除了哪些元素；

125、介绍一下标准css盒子模型；

126、什么事css预处理器；

127、css3有哪些新特性；

128、html5 中的应用缓存是什么？

129、本地存储和cookies’之间的区别是什么

130、简单介绍boostrap栅格系统

131、你如何对网站的文件和资源进行优化？

132、ie 和标准下有哪些兼容性的写法

133、get和post区别？

134、Doctype的作用是什么？

135、浏览器标准模式和怪异模式的区别是什么。

136、解释一下浮动和它的工作原理。

137、列举不同的清除浮动的方法，并指出他们各自适用的场景。