1. 注释：通常记录所有的函数，它们的参数和返回值，或是任何不寻常的技术和方法。
2. 其它命名形式：

常量定义：全部单词大写；var PI = 3.14；

全局变量：全部单词大写；var MAX\_WIDTH = 800；

使用一个下划线前缀来表示一个私有属性或方法。

1. 分隔单词: 对于构造函数，可以使用大驼峰式命名法，如MyConstructor()。对于函数和方法名称，你可以使用小驼峰式命名法，像是myFunction()。要是变量不是函数呢？开发者通常使用小驼峰式命名法，但还有另外一种做法就是所有单词小写以下划线连接：例如，first\_name。

4.使用花括号，并始终把在与之前的语句放在同一行：

function func() {  
 return {  
 name : "Batman"  
 };  
}

1. parseInt( string , radix ) radix:2~36，超过范围输出NAN

当参数 radix 的值为 0，或没有设置该参数时，parseInt() 会根据 string 来判断数字的基数。

如果 string 以 "0x" 开头，parseInt() 会把 string 的其余部分解析为十六进制的整数；

如果省略该参数或其值为 0，则数字将以 10 为基础来解析。如果它以 “0x” 或 “0X” 开头，将以 16 为基数。

1. 避免Eval(string) : 可计算某个字符串，并执行其中的JavaScript代码。

如果你现在的代码中使用了eval()，记住该咒语“eval()是魔鬼”。此方法接受任意的字符串，并当作JavaScript代码来处理。当有 问题的代码是事先知道的（不是运行时确定的），没有理由使用eval()。如果代码是在运行时动态生成，有一个更好的方式不使用eval而达到同样的目 标。例如，用方括号表示法来访问动态属性会更好更简单：

// 反面示例  
var property = "name";  
alert(eval("obj." + property));  
// 更好的  
var property = "name";  
alert(obj[property]);

使用eval()也带来了安全隐患，因为被执行的代码（例如从网络来）可能已被篡改。这是个很常见的反面教材，当处理Ajax请求得到的JSON 相应的时候。在这些情况下，最好使用JavaScript内置方法来解析JSON相应，以确保安全和有效。若浏览器不支持JSON.parse()，你可 以使用来自JSON.org的库。

同样重要的是要记住，给setInterval(), setTimeout()和Function()构造函数传递字符串，大部分情况下，与使用eval()是类似的，因此要避免。在幕后，JavaScript仍需要评估和执行你给程序传递的字符串：

// 反面示例  
setTimeout("myFunc()", 1000);  
setTimeout("myFunc(1, 2, 3)", 1000);  
// 更好的  
setTimeout(myFunc, 1000);  
setTimeout(function () {  
 myFunc(1, 2, 3);  
}, 1000);

使用新的Function()构造就类似于eval()，应小心接近，eval()可以访问和修改它外部作用域中的变量，这是 Function做不来的（注意到使用Function和new Function是相同的）

1. JavaScript的变量在比较的时候会隐式类型转换。这就是为什么一些诸如：false == 0 或 “” == 0 返回的结果是true。为避免引起混乱的隐含类型转换，在你比较值和表达式类型的时候始终使用===和!==操作符。
2. 从技术上将，你可以使用for-in循环数组（因为JavaScript中数组也是对象），但这是不推荐的。因为如果数组对象已被自定义的功能增强，就可能发生逻辑错误。另外，在for-in中，属性列表的顺序（序列）是不能保证的。所以最好数组使用正常的for循环，对象使用for-in循环。

// 对象  
var man = {  
 hands: 2,  
 legs: 2,  
 heads: 1  
};

for (var i in man) {  
 if (man.hasOwnProperty(i)) { // 过滤  
 console.log(i, ":", man[i]);  
 }  
}

1. HTMLCollections指的是DOM方法返回的对象，例如

document.getElementsByName()  
document.getElementsByClassName()  
document.getElementsByTagName()

document.images: 页面上所有的图片元素  
document.links : 所有a标签元素  
document.forms : 所有表单  
document.forms[0].elements : 页面上第一个表单中的所有域

1. 在所有浏览器下，循环获取内容时缓存HTMLCollections的长度或者数组的长度是更快的。
2. var置顶解析
3. 在浏览器中，全局对象可以通过window属性在代码的任何位置访问（除非你做了些比较出格的事情，像是声明了一个名为window的局部变量）。但是在其他环境下，这个方便的属性可能被叫做其他什么东西（甚至在程序中不可用）。如果你需要在没有硬编码的window标识符下访问全局对象，你可以在任何层级的函数作用域中做如下操作：

var global = (function () {  
 return this;  
}());

1. 隐式全局变量和明确定义的全局变量间有些小的差异，就是通过delete操作符让变量未定义的能力。

通过var创建的全局变量（任何函数之外的程序中创建）是不能被删除的。

无var创建的隐式全局变量（无视是否在函数中创建）是能被删除的。(通过delete)

14.function foo() {  
 var a = b = 0;  
}

由于从左到右的赋值，所以b为全局变量。每个JavaScript环境有一个全局对象，当你在任意的函数外面使用this的时候可以访问到，也就是说此时window===this