# JS命名规范

1. 必须使用Tab键进行代码的缩进，以节约代码大小，建议设置编辑器的tab为4个空格宽度
2. 接口风格

|  |  |  |
| --- | --- | --- |
| **结构** | **规则** | **例如** |
| 类 | 驼峰式 | ModuleClass |
| 公有方法 | 混合式 | getPosition() |
| 公有变量 | 混合式 | frameStyle |
| 常量 | 大写式 | DEFAULT\_FRAME\_LAYOUT |

1. 其它建议风格，非必要

|  |  |
| --- | --- |
| **结构** | **规则** |
| 公有方法和属性 | 混合，例子：mixedCase |
| 私有方法和属性 | 混合，例子：mixedCase |
| 方法（method）参数 | 混合，例子：mixedCase |
| 本地（local）变量 | 混合，例子：mixedCase |

1. 所有语句结束后，必须使用 “ ; ” 结束
2. 所有变量必须是有意义的英文，严厉禁止拼音
3. 变量允许使用公认英文缩写
4. 类命名必须是驼峰式
5. 常量必须所有单词大写，并且每个单词间用下划线分割
6. 枚举类型时，枚举的命名必须有意义，枚举与枚举成员必须以驼峰式
7. 常量和枚举必须在最前端定义，混合时注意，必须把常量和枚举定义的文件放在文件列表的第一位
8. 变量内简写单词如果在开头则小写：xmlDocument，如果不在开头则全部大写：loadXML
9. “ on ” 只用作事件的命名
10. 函数开头必须是有意义的动词或者动词短语
11. 私有类的变量属性成员建议使用混合式命名，并前面使用下划线，例如：\_mixedCase
12. 临时的全局变量放到一个全局的哈希表里，方便变量回收
13. 所有的全局变量必须初始化，尽量少用全局变量
14. 大括号前面不能有换行符
15. 保留字以及特有dom属性不能作为变量名

# 特殊命名约定

1. 前面加 “ is ” 的变量名应该为布尔类型，亦可用can，has，should
2. 前面加 “ str ” 的变量名应该为字符串
3. 前面加 ” arr ” 的变量名应该为数组
4. 前面加 “ num ” 或 “ count ” 的变量名应该为数字
5. “ o ” 作为局部变量或者参数，表示为Object
6. “ e ” 作为局部变量或者参数，表示为Element
7. “ evt ” 作为局部变量或者参数，表示为event
8. “ err ” 作为局部变量或者参数，表示为error
9. 重复的变量建议使用 “ i ” , “ j ” , “ k ” （以此类推）等命名的变量
10. 能缩写的单词尽量缩写
11. 避免产生令人误解的的布尔值 isNotNumber , isNan
12. 处理错误的变量，必须在后面跟着 ” Error ”
13. 初始化用的函数必须使用 ” init ” 开头，如果一个页面只有初始化可以单独使用init()
14. 尽量做有意义的代码折行，不要让一行代码过长。（HTML字符串除外）
15. 操作符建议使用空格隔开
16. 函数调用和方法避免使用空白
17. 逗号（，）建议使用空白隔开
18. 不允许频繁使用previousSibling和nextSibling

# 词法结构

1. 普通代码段应该看起来如下：  
   while(!isDone){  
    doSomething();  
    isDone = moreToDo();  
   }
2. 变量定义如下：  
   var a = null,  
    b = 1,  
    c = 0;
3. 函数定义方法如下：  
   var funcA = function(){  
    var a = 0;  
    …  
   }
4. If语句应该看起来像这样：  
   if(someCondition){  
    statements;  
   } else if(someOtherCondition){  
    statements;  
   } else {  
    statements;  
   }
5. for 语句应该看起来像这样：  
   for(initialization;condition;update){  
    statements;  
   }
6. while语句看起来像这样 :  
   while(!isDone){  
    doSomething();  
    isDone = moreToDo();  
   }
7. do … while 语句看起来像这样：  
   do{  
    statements;  
   } while(condition);
8. switch语句应该看起来像这样：  
   switch(condition){  
    case “A”:  
    statements; //注释  
    case “B”:  
    statements; //注释  
    break;  
    default:  
    statements; //注释  
    break;  
   }
9. try … catch 语句看起来像这样：  
   try {  
    statements;  
   } catch(ex) {  
    statements  
   } finally {  
    statements;  
   }
10. 单行的if … else , while 或者for 也必须加入括号  
    if(condition){  
     statements;  
    }  
    while(condition){  
     statements;  
    }  
    for(initialization;condition;update){  
     statements;  
    }

# 注释规范

|  |  |
| --- | --- |
| @param  @argument | 指定参数名和说明来描述一个函数参数 |
| @returns | 描述函数的返回值 |
| @author | 指示代码的作者 |
| @deprecated | 指示一个函数已经废弃，而且在将来的代码版本中将彻底删除。要避免使用这段代码 |
| @see | 创建一个HTML链接，指向指定类的描述 |
| @version | 指定发布版本 |
| @requires | 创建一个HTML链接，指向这个类所需的指定类 |
| @throws @exception | 描述函数可能抛出的异常的类型 |
| {@link} | 创建一个HTML链接，指向指定的类。这与@see很类似，但{@link}能嵌在注释文本中 |
| @fileoverview | 这是一个特殊的标记。如果在文件的第一个文档块中使用这个标记，则指定该文档块的余下部分将用来提供这个文件的概述 |
| @class | 提供类的有关信息，用在构造函数的文档中 |
| @constructor | 明确一个函数是某个类的构造函数 |
| @type | 指定函数的返回类型 |
| @extends | 指示一个类派生了另一个类。JSDoc通常自己就可以检测出这种信息，不过，在某些情况下则必须使用这个标记 |
| @inner @private | 指示一个类或函数是私有的。私有类和函数不会出现在HTML文档中，除非运行JSDoc时提供了--private命令行选项 |
| @final | 指示一个值是常量值。要记住JavaScript无法真正保证一个值是常量 |
| @ignore | JSDoc忽略有这个标记的函数 |
| @constant | 常数 |
| @constructs | 构造 |
| @default | 默认值 |
| @description | 说明 |
| @example | 范例 |
| @field | 变量（非功能） |
| @event | 事件 |
| @scope ||@lends | 作用域 |
| @public | 说明内在变量是公开的 |
| @property | 属性 |
| @namespace | 命名空间 |

# 其它

1. String优化  
   循环体内的字符串累加使用join方式，例如：  
   var s = [];  
   for(var I = 0;I < 100;i++){  
    s.push(“hello”);  
   }  
   var k = s.join(“”);
2. Switch 建议采用hash-table  
   switch可以用Object替代，例如：  
   var a = {  
    “1”:doAction1,  
    ”2”: doAction2  
   };  
   function doAction1(){  
    …  
   }  
   function doAction2(){  
    …  
   }  
   a[1]();
3. 不建议使用eval  
   不推荐使用eval来执行脚本，除非用来解释已经确定的安全的json数据。
4. 不要使用Function构造器
5. 不要给setTimeout或者setInterval传递字符串参数
6. 注意IE的内存泄露
7. JavaScript代码独立  
   JavaScript不应该被包含在HTML文件中，除非这是段特定只属于此部分的代码，在HTML中的JavaScript代码会明显增加文件大小，而且也不能对其进行缓存和压缩
8. []和{}  
   使用[]代替new Array()使用{}代替 new Object()
9. === 和 !==操作符  
   始终使用===和!==会更好，==和!=操作符会做类型强制转换。特别是不要使用==来和假值做比较
10. 使用var和不使用var的区别  
    var a = 100; 显式声明  
    a = 100; 隐式声明  
    在函数中使用var关键字进行显式申明的变量是做为局部变量，而没有用var关键字，使用直接赋值方式声明的是全局变量。
11. use strict 严格模式  
    <http://www.ruanyifeng.com/blog/2013/01/javascript_strict_mode.html>

# JavaScript面向对象编程的思想

查看内容地址：<http://coolshell.cn/articles/6441.html>

# JavaScript闭包

<http://my.oschina.net/u/2349331/blog/470142>

<http://www.cnblogs.com/frankfang/archive/2011/08/03/2125663.html>

# JavaScript内存泄露相关问题

<http://www.cnblogs.com/carekee/articles/1733847.html>