JavascriptCoding Standard javascript代码规范

**规范理念**

Any violation to this guide is allowed if it enhances readability.

所有的代码都要变成可供他人容易阅读的。

--引用自Dojo Javascript 语法规

# 规范详解

## 命名规范

1. 必须使用 Tab 键进行代码缩进，以节约代码大小(建议设置编辑器的tab为4个空格的宽度)
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. 其他建议风格，非必要

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

1. 所有语句结束后，必须使用 ; 号结束
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，并且每个单词间加下划线
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，merge 时注意，必须把常量与枚举定义的文件放在文件列表的第一位
8. 变量内的简写单词不能全大写XmlDocument strHtml
9. "on"只能用作事件的命名
10. 函数开头必须是有意义的动词或动词短语
11. 私有类的变量属性成员 建议 使用混合式命名，并前面下下划线
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. 保留字以及特有的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值，亦可使用 “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” 作为局部变量或参数，表示为Object
6. “e” 作为局部变量或参数，表示为Element
7. “evt” 作为局部变量或参数，表示为event
8. “err” 作为局部变量或参数，表示为error
9. 重复变量建议使用 "i", "j", "k" （依次类推）等名称的变量(全世界公认)
10. 能缩写的单词尽量缩写
11. 避免产生令人误解的布尔值 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：
2. while(!isDone){
3. doSomething();
4. isDone = moreToDo();

}

1. if 语句应该看起来像这样：
2. if(someCondition){
3. statements;
4. } else if(someOtherCondition){
5. statements;
6. } else {
7. statements;

}

1. for 语句应该看起来像这样：
2. for(initialization; condition; update){
3. statements;

}

1. while 语句应该看起来像这样：
2. while(!isDone){
3. doSomething();
4. isDone = moreToDo();

}

1. do ... while 语句应该看起来像这样：
2. do{
3. statements;

}while(condition);

1. switch 语句应该看起来像这样：
2. switch(condition){
3. case ABC:
4. statements;
5. //  fallthrough
6. case DEF:
7. statements;
8. break;
9. default:
10. statements;
11. break;

}

1. try ... catch 语句应该看起来像这样：
2. try{
3. statements;
4. }catch(ex){
5. statements;
6. }finally{
7. statements;

}

1. 单行的 if - else，while 或者 for 语句也必须加入括号：
2. if(condition){ statement;}
3. while(condition){ statement;}

for(intialization; condition; update){ statement;}

## 注释规范

1. 一些你不打算给其他人使用的函数，建议添加 @ignore 让文档输出时可以忽略这段注释
2. 一些相关的功能相关的函数，建议加上@see Function 来对上下文做索引
3. 对于一些函数不建议或则需要注意的使用方法，必须加上 @deprecated作为提醒
4. 每个js文件的文件头都必须包含 @fileoverview @author, 建议加上@version
5. 每个函数都必须使用JsDoc 来注释他的用意
6. 每个带参数的函数必须包含 @param
7. 每个有返回值的函数必须包含 @return
8. 构造函数必须加上 @constructor
9. 继承函数建议加上 @base 表示其继承于哪个类
10. 常用全局变量建议使用 JsDoc 的注释方式
11. 一般的变量及局部变量才用 // 方式进行注释，建议在需要做注释的语句的上一行
12. 其他详情请参考 JsDoc 注释方法

## 其他

1. String 优化

循环体内的字符串累加使用join方式。 例如:

        var r = [];

        for (var i=0;i<100;i++){

                r.push("hello");

        }

        var k = r.join("");

1. Switch 建议采用hash-table

switch 可以才用 Object代替 例如:

        var a = {

                "1":doAction1,

                "2":doAction2,

        }

        function doAction1(){

        }

        function doAction2(){

        }

        a[1]();

1. 不建议使用eval

不推荐使用eval来执行脚本。除非用来解释json数据。

1. 注意 IE 的内存泄露问题