**JavaScript语言编程规范**

**东华软件©2010**

## 1. 命名规则

### 1.1. 类名

规则：

一个或多个单词构成，每个单词的首字母大写，其余字母小写，单词之间直接相连，没有其他符号。

说明：

例子：

正确的：TestClass

错误的：Test\_Class、testClass、test\_class等

### 1.2. 公共方法

规则：

一个或多个单词构成，第一个单词的所有字母均小写，其余每个单词的首字母大写，其余字母小写，单词之间直接相连，没有其他符号。

说明：

例子：

正确的：testMethod(), getHtml(), getXml()

错误的：TestMethod()、test\_Method()、test\_method()、getHTML()、getXML()等

### 1.3. 全局变量

规则：

以“g\_”开头，而后由一个或多个单词构成，第一个单词的所有字母均小写，其余每个单词的首字母大写，其余字母小写，单词之间直接相连，没有其他符号。这里，第一个单词前，可使用一个缩写短语表示全局变量的类型，如果使用了类型缩写短语，那么其余所有单词的首字母均大写，其余字母小写。这里的变量类型缩写包括：

n 表示某种节点（如XML节点、HTML节点等），如nText等

i 表示整数的列举形式，如nValue等

lst list类型

s string类型

d Date类型

f 有小数点的数字

b 布尔类型

说明：

这里的全局变量指，为全局所有js文件共享的变量，不包括仅被本js文件使用的内部全局变量。例如，一个变量虽然在语法上是全局变量，但是仅在声明它的js文件内部作为全局变量使用，那么其命名规则遵循私有变量的命名规则，而不是本规则。

上述代表变量类型的缩写词，如果变量的类型已经可以通过变量名明确，则缩写词可以省略，如：fileName，不需要写为sFileName，除非FileName确实不是字符串类型，fileCount不需要写为iFileCount，除非FileCount确实不是整数类型。

例子：

正确的：g\_fileName、g\_lstMenuItems等

错误的：testVar、TestVar、test\_var等

### 1.4. 常量

规则：

一个或多个单词构成，每个单词的所有字母均大写，单词之间用“\_”相连。类型缩写规则同1.3.。

说明：

例子：

正确的：TEST\_CONST

错误的：test\_const、TESTCONST等

### 1.5. 私有方法

规则：

以“\_”开头，一个或多个单词构成，第一个单词的所有字母均小写，其余每个单词的首字母大写，其余字母小写，单词之间直接相连，没有其他符号。

说明：

这里的私有方法指，仅供类内部调用的方法或者声明方法的js文件的内部调用的方法。

例子：

正确的：\_testMethod()

错误的：\_TestMethod()、test\_Method()、test\_method()等

### 1.6. 私有变量

规则：

以“\_”开头，一个或多个单词构成，第一个单词的所有字母均小写，其余每个单词的首字母大写，其余字母小写，单词之间直接相连，没有其他符号。类型缩写规则同1.3.。

说明：

这里的私有变量指，仅供类内部使用的变量或者声明变量的js文件内部使用的变量。

例子：

正确的：\_testVar

错误的：\_TestVar、test\_Var、test\_var等

### 1.7. 方法的参数

规则：

一个或多个单词构成，第一个单词的所有字母均小写，其余每个单词的首字母大写，其余字母小写，单词之间直接相连，没有其他符号。类型缩写规则同1.3.。

说明：

例子：

正确的：testVar

错误的：TestVar、test\_Var、test\_var等

### 1.8. 局部变量

同1.7.。

### 1.9. 方法命名

规则：

方法的第一个单词必须为动词，如“set”、“get”等。

对象的方法，如果以对象本身作为宾语，则可以省略，如：

应写为：object.getName()，而不是object.getObjectName()等。

### 1.10. 单词

规则：

所有的单词必须使用英语单词，除非没有相应的英语单词，才允许使用汉语拼音或首拼。

### 1.11. 缩写

规则：

全局变量、类的公用函数、公共属性、常量一律使用单词的完整形式，而不使用首拼。

局部变量、私有变量等如需要使用缩写，首先参照如下的单词缩写表：

完全拼写 缩写

connection conn

database db

destination dst

document doc

event evt

image img

initialize init

list lst

number num （注意不要缩写为no）

option opt

source src

table tbl

此外，可以采用如下的命名规则：

去除除首字母外的所有元音字母，取前4个英文字母，如attribute缩写为attr。

### 1.12. 类名和构造函数名

规则：

类名和构造函数名应体现类的继承规则，如：

EventHandler

UIEventHandler

MouseEventHandler

### 1.13. 其他命名规则

规则：

获取/设置类私有属性的方法，必须以get/set开头。

布尔变量名或返回布尔变量的方法命名应使用直接的方法，并以诸如“is”、“has”、“found”、“can”开头。

对于集合类型，必须使用复数形式作为变量名称，如lstItems等。

表示数量的变量，结尾一律为“Count”，不要使用“Number”或者“Num”。

简单循环变量名可使用i、j、k命名，但是如果循环变量还有一些复杂的含义，则需要使用若干单词组合成有意义的变量名。

表示相反含义的变量名必须成对使用反义词，如：get/set、begin/end、start/stop、add/remove、insert/delete、create/destroy等。

应避免使用表示反意的布尔变量名，如：isNotError、isNotFound等，这通常会让人的思想多转一个弯。

## 2. 排版

### 2.1. 缩进

每层缩进一律为四个空格大小，并且必须使用“TAB”而不是空格实现缩进。

### 2.2. 括号

使用如下的括号规则（也就是右花括号始终单独占一行）：

function someMethod() {

statement1;

if (condition1) {

...

}

else if (condition2) {

...

}

else {

...

}

while (condition3) {

if (condition4) { //if、for、while等语句，即使其内部只有一条语句，也必须用花括号括起来

continue;

}

if (condition5) {

break;

}

}

do {

...

} while (condition6);

switch (condition 7) {

case A:

s1;

...

break;

case B:

s2;

...

break;

case C:

s3;

...

break;

default:

...

}

try {

sTry;

...

}

catch (exception) {

sCatch;

...

}

finally {

sFinally;

...

}

} //someMethod

注意上述方法中，即使if语句后只有一行语句，也需要用花括号将其括起。

### 2.3. 折行

规则：

如果算术表达式较长的（超过100列），可以折行书写，每行开头为运算符，如：

var someVar = expression1

+ expression2

+ expression3;

如果函数的参数列表较长的，也可折行书写，如：

var someVar = someFunction(

arg1,

arg2,

arg3);

### 2.4. 空行

规则：

空行每次最多保留一行。

函数声明和函数的第一句语句间应有一个空行，如：

function someMethod() {

statment;

}

循环的进入语句和循环体的第一条语句之间应有一个空行，如：

while (condition) {

statement;

}

逻辑上紧密的语句应放在一起形成一个“语句块”，语句块之间用空行分隔开，如：

statment1;

statment2;

statment3;

statment4;

statment5;

statment6;

### 2.5. 空格

用例子说明：

a = a + 1; //每个算符之间要留空格

if (a == b) { //if和之后的(间要留空格

...

}

while (condition1) { //while和之后的(间要留空格

...

}

for (var i = 0; i < length; i++) { //注意for语句后的空格

...

}

c = a\*b + d; //高优先级的算符两边不留空格，低优先级的算符两边留空格，使得运算优先级明确

a = (d? b: c); //三元算子的操作符写法

## 3. 注释

### 3.1. 注释的首要规则

注释的首要规则是，先写注释，后写代码。

编写超过10行的代码前，必须先用块注释的方式（即/\*...\*/的形式），将需要完成的工作步骤描述清楚，然后，对工作步骤描述进行检查（检查内容见“JS代码审查表”），检查通过后才能够开始编写具体的JS语句。

编写工作步骤描述时应注意重在描述做什么工作，而不是怎样做工作，即不要将工作步骤描述得过细。

上述的代码行数计算，不包括空行、函数定义语句（即“function”语句）、仅包括单独花括号的行。

### 3.2. 注释的位置

块注释必须位于其所描述的代码之前。行注释（即//的形式）位于代码行之右。