**代 码 编 写 规 范**



课 程 名 称 软件工程基础

项 目 名 称 模拟三国

小 组 名 称 明德1-518

指 导 老 师 杨枨



|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 代码版本号 | 更新内容 | 更新日期 |
| 1 | V0.0.1 | 详细的代码编写规范 | 2019.5.28 |
|  |  |  |  |

注:红色为当前版本

[1.嵌入规则 4](#_Toc9944855)

[2.对齐缩进与换行 4](#_Toc9944856)

[a) 缩进 4](#_Toc9944857)

[b) 换行 4](#_Toc9944858)

[1).字符串过长截断 5](#_Toc9944859)

[2).三元运算符过长 5](#_Toc9944860)

[3).过长的逻辑条件组合 6](#_Toc9944861)

[4).过长的JSON和数组 6](#_Toc9944862)

[5).return语句 7](#_Toc9944863)

[3.命名 7](#_Toc9944864)

[a).命名法说明 7](#_Toc9944865)

[b).变量名：必须使用camel命名法 8](#_Toc9944866)

[c).参数名：必须使用camel命名法 8](#_Toc9944867)

[d).函数名：必须使用camel命名法 8](#_Toc9944868)

[e).方法/属性：必须使用camel命名法 8](#_Toc9944869)

[f).私有（保护）成员：必须以下划线\_开头 8](#_Toc9944870)

[g).命名空间：必须使用camel命名法 8](#_Toc9944871)

[h).语义：命名同时还需要关注语义 8](#_Toc9944872)

[变量名应当使用名词； 8](#_Toc9944873)

[4.注释 8](#_Toc9944874)

[5.声明 9](#_Toc9944875)

[6.参考资料 10](#_Toc9944876)

# 1.嵌入规则

Javascript程序应该尽量放在.js的文件中，需要调用的时候在页面中以<script src="filename.js">的形式包含进来。Javascript代码若不是该页面专用的，则应尽量避免在页面中直接编写Javascript代码。

# 2.对齐缩进与换行

## a) 缩进

在同一系统中应采用同一种缩进标准，本文提倡缩进大小为4个空格。各编译器对Tab键所代替的空白大小定义不同。建议在设置开发环境时，将编辑器里的Tab快捷键重新设置成4个空格。多数编译器提供了此功能。否则建议按4次空格来进行缩进。

## b) 换行

在以下位置必须换行：

每个独立语句结束后；

if、else、catch、finally、while等关键字前；

运算符处换行时，运算符必须在新行的行首。

对于因为单行长度超过限制时产生的换行，参考行长度中的策略进行分隔。

## 1).字符串过长截断

每行代码应小于80个字符。若代码较长应尽量换行，换行应选择在操作符和标点符号之后，最好是在分号“;”或逗号“,”之后。下一行代码相对上一行缩进4个空格。这样可以有效防止复制粘贴引起的代码缺失等错误并增强可读性。

按一定长度截断字符串，并使用+运算符进行连接。分隔字符串尽量按语义进行，如不要在一个完整的名词中间断开。特别的，对于HTML片段的拼接，通过缩进，保持和HTML相同的结构：

也可使用数组来进行拼接，相对+运算更容易调整缩进：

## 2).三元运算符过长

三元运算符由3部分组成，因此其换行应当根据每个部分的长度不同，形成3种不同的情况：

不得出现以下情况：

## 3).过长的逻辑条件组合

当因为较复杂的逻辑条件组合导致80个字符无法满足需求时，应当将每个条件独立一行，逻辑运算符放置在行首进行分隔，或将部分逻辑按逻辑组合进行分隔。最终将右括号)与左大括号{放在独立一行，保证与if内语句块能容易视觉辨识。如：

## 4).过长的JSON和数组

如果对象属性较多导致每个属性一行占用空间过大，可以按语义或逻辑进行分组的组织，如：

通过5个一组的分组，将每一行控制在合理的范围内，并且按逻辑进行了切分。 对于项目较多的数组，也可以采用相同的方法，如：

## 5).return语句

return如果用表达式的执行作为返回值，请把表达式和 return 放在同一行中，以免换行符被误解析为语句的结束而引起返回错误。return 关键字后若没有返回表达式，则返回 undefined。构造器的默认返回值为 this。

# 3.命名

命名的方法通常有以下几类：

## a).命名法说明

1).camel命名法，形如thisIsAnApple （小驼峰命名法）

2).pascal命名法，形如ThisIsAnApple

3).下划线命名法，形如this\_is\_an\_apple ·

4).中划线命名法，形如this-is-an-apple

根据不同类型的内容，必须严格采用如下的命名法(本项目主要用camel命名法)：

## b).变量名：必须使用camel命名法

## c).参数名：必须使用camel命名法

## d).函数名：必须使用camel命名法

## e).方法/属性：必须使用camel命名法

## f).私有（保护）成员：必须以下划线\_开头

## g).命名空间：必须使用camel命名法

## h).语义：命名同时还需要关注语义

## 变量名应当使用名词；

boolean类型的应当使用is、has等起头，表示其类型；·

函数名应当用动宾短语；

类名应当用名词。

# 4.注释

注释要尽量简单，清晰明了。着重注释的意思，对不太直观的部分进行注解：

（当然这种直接定义一堆全局变量的做法不推荐）

此外，JavaScript 的注释有两种"//" 和"/\* .... \*/"，建议"//"用作代码行注释，"/\* .... \*/"形式用作对整个代码段的注销，或较正式的声明中，如函数参数、功能、文件功能等的描述中：

另：复制粘贴应注意注释是否与代码对应。

# 5.声明

**1).变量的声明**

尽管 JavaScript 语言并不要求在变量使用前先对变量进行声明。但我们还是应该养成这个好习惯。这样可以比较容易的检测出那些未经声明的变量，避免其变为隐藏的全局变量，造成隐患。

在函数的开始应先用 var 关键字声明函数中要使用的局部变量，注释变量的功能及代表的含义，且应以字母顺序排序。每个变量单独占一行，以便添加注释。这是因为 JavaScript 中只有函数的 {} 表明作用域，用 var 关键字声明的局部变量只在函数内有效，而未经 var 声明的变量则被视为全局变量。示例：

用 var 声明过的变量 valueA 和没有声明的变量 valueB 是有区别的。特别需要注意的是，在函数内部用 var 声明的变量为局部变量，这样可以有效地避免因局部变量和全局变量同名而产生的错误。

**2).函数的声明**

函数也应在调用前进行声明，内部函数应在 var 声明内部变量的语句之后声明，可以清晰地表明内部变量和内部函数的作用域。

此外，函数名紧接左括号'('之间，而右括号')'和后面的'{'之间要有个空格，以清楚地显示函数名以其参数部分，和函数体的开始。若函数为匿名 / 无名函数，则 function 关键字和左括号'('之间要留空格，否则可能误认为该函数的函数名为 function。

**内部函数声明示例：**

从上例的输出可以看出，inF() 函数仅在 outF() 函数的内部生效，局部变量 innerA 对内部函数的作用域生效。这样的编码方式使得变量和函数的作用域变得清晰。

# 6.参考资料

博客园段盼盼JS代码编写规范

<https://www.cnblogs.com/paul-3/p/5992627.html>