# JavaScript规范

## 1.注释

/\*\*

\* [测试脚本]

\* @Author zhanggongze

\* @DateTime 2017-12-12

\*/

## 2.代码模块化

一个页面一个js

## 3. 首选函数式风格

函数式编程让你可以简化代码并缩减维护成本，因为它容易复用，又适当地解耦和更少的依赖。

init()

/\*\*

\* [init ...]

\* @Author zhanggongze

\* @DateTime 2017-12-06

\* @param

\* @return

\*/

function init(){

//...

getDate();

}

/\*\*

\* [getDate ...]

\* @Author zhanggongze

\* @DateTime 2017-12-06

\* @param

\* @return

\*/

function getDate(){

//...

isNullData();

}

/\*\*

\* [isNullData...]

\* @Author zhanggongze

\* @DateTime 2017-12-06

\* @param

\* @return

\*/

function isNullData(){

//...

outExcles();

}

/\*\*

\* [outExcles ...]

\* @Author zhanggongze

\* @DateTime 2017-12-06

\* @param

\* @retur

\*/

function outExcles(){

//...

}

## 4.全局命名空间污染与 IIFE

无论何时，想要创建一个新的封闭的定义域，那就用 IIFE。它不仅避免了干扰，也使得内存在执行完后立即释放。

所有脚本文件建议都从 IIFE 开始。

## 5.变量声明

总是使用 var 来声明变量。如不指定 var，变量将被隐式地声明为全局变量，这将对变量难以控制。

## 6.提升声明

将要用到的变量提前申明可避免些诡异的问题

## 7.明智地使用真假判断

当我们在一个 if 条件语句中使用变量或表达式时，会做真假判断。if(a == true) 是不同于 if(a)的。后者的判断比较特殊，我们称其为真假判断。这种判断会通过特殊的操作将其转换为 true 或 false，下列表达式统统返回 false：false, 0, undefined, null, NaN, ''（空字符串）.

这种真假判断在我们只求结果而不关心过程的情况下，非常的有帮助。

## 8.变量赋值时的逻辑操作

逻辑操作符 || 和 && 也可被用来返回布尔值。如果操作对象为非布尔对象，那每个表达式将会被自左向右地做真假判断。基于此操作，最终总有一个表达式被返回回来。这在变量赋值时，是可以用来简化你的代码的。

## 9.分号

总是使用分号，因为隐式的代码嵌套会引发难以察觉的问题。当然我们更要从根本上来杜绝这些问题,给语句末加上分号。

## 10.语句块内的函数声明

切勿在语句块内声明函数，在 ECMAScript 5 的严格模式下，这是不合法的。函数声明应该在定义域的顶层。但在语句块内可将函数申明转化为函数表达式赋值给变量。

## 11.三元条件判断（if 的快捷方法）

用三元操作符分配或返回语句。在比较简单的情况下使用，避免在复杂的情况下使用。没人愿意用 10 行三元操作符把自己的脑子绕晕。

## 12.字符串

统一使用单引号(‘)，不使用双引号(“)。这在创建 HTML 字符串非常有好处

## 13.代码检查

对于比较宽松自由的编程语言来说，严格遵循编码规范和格式化风格指南就显得极为重要。--对于javascript建议使用ESLint

## 14.语法意义相互独立的代码将用空行分隔

if (a > b) {

value = a; //行间不用空行间隔

}

var variableName = value; //与上一代码行使用空行间隔