**[Javascript中的循环变量声明，到底应该放在哪儿？](http://www.cnblogs.com/front-end-ralph/p/4874726.html)**

相信很多Javascript开发者都在声明循环变量时犹豫过var i到底应该放在哪里：放在不同的位置会对程序的运行产生怎样的影响？哪一种方式符合Javascript的语言规范？哪一种方式和ecma标准未来的发展方向匹配？本文将对四种常见的声明循环变量的书写方式进行简单的分析和比较。

**习惯1：不声明直接使用**

function loop(arr) {

for (i = 0; i < arr.length; i++) {

// do something

}

}

非常危险的使用习惯，一般情况下循环变量将成为window对象上的一个属性被全局使用，极有可能影响程序的正常逻辑实现，想想都蛋疼，大家都懂的，就不在这里赘述了。   
需要着重提一下的是，在strict模式下，未声明变量而直接赋值的使用方式会直接抛出异常，早就该这么做啦！引用一下ecma-262标准附录C中的一段话：  
*"Assignment to an undeclared identifier or otherwise unresolvable reference does not create a property in the global object. When a simple assignment occurs within strict mode code, its LeftHandSide must not evaluate to an unresolvable Reference. If it does a ReferenceError exception is thrown (6.2.3.2)."*  
换言之，如果再使用未经声明的变量的话，ReferenceError异常会被抛出。

**习惯2：放在for循环初始语句块中并反复声明**

[复制代码](javascript:void(0);)

function loop(arr) {

for (var i = 0; i < arr.length; i++ ){

// do someting

}

console.log(i);

for (var i = 0; i < arr.length; i++ ){

// do something else

}

}

[复制代码](javascript:void(0);)

这种方式看似最安全规范，很多从C和Java转到前端开发的同学都偏爱这样的写法，事实上，这也许是由于对Javascript中一个重要概念有所误解造成的——变量作用域。不同于C和Java，Javascript并不具备真正的块级作用域，也就是说，在第一个循环结束之后，console.log(i)并不会打印undefined或者抛出ReferenceError异常，而是会正常打印出arr.length。  
当然，这样的写法虽然除了美观以外意义不大，但是长久以来兼容性良好且没有违反任何规范——ecma标准中并没有禁止在某一个作用域内对于同一变量的重复声明。不仅如此，其实这里还有一个另外好消息，在ECMAScript 6中，一个新的，为支持真正的块级作用域而生的关键字出现了——let。这里放一个传送门，有兴趣的同学可以自行了解：  
<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/let>

**习惯3：在函数顶部和其他变量一起集中定义**

[复制代码](javascript:void(0);)

function loop(arr) {

var var1;

var var2;

var i;

for (i = 0; i < arr.length; i++) {

// do something

}

}

[复制代码](javascript:void(0);)

这种c89-like式的变量定义方式在Javascript中几乎无可挑剔，既不会造成Javascript支持块级作用域的误解，又不会污染全局scope，还不违反任何标准和规范，主要缺点就是循环变量的声明和循环体可能会隔开比较远。在不借助更多代码的前提下，除了等待各大主流浏览器厂商实现ECMAScript 6中的let关键字以外，这个问题似乎找不到更好的解决方案。

**习惯4：将循环代码封装到IIFE中**

[复制代码](javascript:void(0);)

function loop(arr) {

(function () {

for (var i = 0; i < arr.length; i++) {

// do something

}

})();

}

[复制代码](javascript:void(0);)

最后一种习惯是前端程序员们熟悉的IIFE（Immediately-Invoked Function Expression），即立即执行函数。此种方法的主要缺点是书写相对麻烦，且有多余的性能损耗（很小），但在兼容性、对各标准规范的遵循上表现良好。如果不嫌麻烦，开发者可以采取这种方式。

以上就是对Javascript中四种常见循环变量定义书写习惯的简单介绍和分析，各有利弊，读者可以结合自己的需求择优使用。应该说，在ECMAScript 6之前并没有一种定义循环变量的完美解决方案。好在ECMAScript标准委员会也及时发现了这个问题，让我们一起期待let关键字吧。