JavaScript 变量提升 JavaScript 使用误区

JavaScript 严格模式(use strict)

JavaScript 严格模式（strict mode）即在严格的条件下运行。

使用 "use strict" 指令

"use strict" 指令在 JavaScript 1.8.5 (ECMAScript5) 中新增。

它不是一条语句，但是是一个字面量表达式，在 JavaScript 旧版本中会被忽略。

"use strict" 的目的是指定代码在严格条件下执行。

严格模式下你不能使用未声明的变量。

支持严格模式的浏览器:
Internet Explorer 10 +、 Firefox 4+ Chrome 13+、 Safari 5.1+、 Opera 12+。

严格模式声明

严格模式通过在脚本或函数的头部添加 "use strict"; 表达式来声明。

实例中我们可以在浏览器按下 **F12 (或点击"工具>更多工具>开发者工具")** 开启调试模式，查看报错信息。

Gif 图演示如下：

实例

```
"use strict";
x = 3.14;       // 报错 (x 未定义)
```

尝试一下 »

实例

```
"use strict";
myFunction();

function myFunction() {
    y = 3.14;   // 报错 (y 未定义)
}
```

尝试一下 »

在函数内部声明是局部作用域 (只在函数内使用严格模式):

实例

```
x = 3.14;       // 不报错
myFunction();

function myFunction() {
   "use strict";
    y = 3.14;   // 报错 (y 未定义)
}
```

尝试一下 »

为什么使用严格模式:

消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为;

- 消除代码运行的一些不安全之处，保证代码运行的安全；
- 提高编译器效率，增加运行速度；
- 为未来新版本的Javascript做好铺垫。

"严格模式"体现了Javascript更合理、更安全、更严谨的发展方向，包括IE 10在内的主流浏览器，都已经支持它，许多大项目已经开始全面拥抱它。

另一方面，同样的代码，在"严格模式"中，可能会有不一样的运行结果；一些在"正常模式"下可以运行的语句，在"严格模式"下将不能运行。掌握这些内容，有助于更细致深入地理解Javascript，让你变成一个更好的程序员。

严格模式的限制

不允许使用未声明的变量：

```
"use strict";
x = 3.14;                // 报错 (x 未定义)
```

尝试一下 »

对象也是一个变量。

```
"use strict";
x = {p1:10, p2:20};      // 报错 (x 未定义)
```

尝试一下 »

不允许删除变量或对象。

```
"use strict";
var x = 3.14;
delete x;                // 报错
```

尝试一下 »

不允许删除函数。

```
"use strict";
function x(p1, p2) {};
delete x;                // 报错
```

尝试一下 »

不允许变量重名:

```
"use strict";
function x(p1, p1) {};   // 报错
```

尝试一下 »

不允许使用八进制:

```
"use strict";
var x = 010;             // 报错
```

尝试一下 »

不允许使用转义字符:

```
"use strict";
var x = \010;            // 报错
```

尝试一下 »

不允许对只读属性赋值:

```
"use strict";
var obj = {};
Object.defineProperty(obj, "x", {value:0, writable:false});

obj.x = 3.14;            // 报错
```

尝试一下 »

不允许对一个使用getter方法读取的属性进行赋值

```
"use strict";
var obj = {get x() {return 0} };

obj.x = 3.14;            // 报错
```

尝试一下 »

不允许删除一个不允许删除的属性：

```
"use strict";
delete Object.prototype; // 报错
```

尝试一下 »

变量名不能使用 "eval" 字符串:

```
"use strict";
var eval = 3.14;         // 报错
```

尝试一下 »

变量名不能使用 "arguments" 字符串:

```
"use strict";
var arguments = 3.14;    // 报错
```

尝试一下 »

不允许使用以下这种语句:

```
"use strict";
with (Math){x = cos(2)}; // 报错
```

尝试一下 »

由于一些安全原因，在作用域 eval() 创建的变量不能被调用：

```
"use strict";
eval ("var x = 2");
alert (x);               // 报错
```

尝试一下 »

禁止this关键字指向全局对象。

```
function f(){
    return !this;
}
// 返回false，因为"this"指向全局对象，"!this"就是false

function f(){
    "use strict";
    return !this;
}
// 返回true，因为严格模式下，this的值为undefined，所以"!this"为true。
```

因此，使用构造函数时，如果忘了加new，this不再指向全局对象，而是报错。

```
function f(){
    "use strict";
    this.a = 1;
};
f();// 报错，this未定义
```

保留关键字

为了向将来Javascript的新版本过渡，严格模式新增了一些保留关键字：

- implements
- interface
- let
- package
- private
- protected
- public
- static
- yield

```
"use strict";
var public = 1500;      // 报错
```

尝试一下 »

"use strict" 指令只允许出现在脚本或函数的开头。

← JavaScript 变量提升 JavaScript 使用误区 →

点我分享笔记