如何避免js中ASI带来的问题

如何避免js中Automatic Semicolon Insertion（自动分号插入机制，ASI）带来的问题

背景：

自动分号插入 (automatic semicolon insertion, ASI）在 JavaScript 程序的语法分析 (parsing) 阶段起作用。
根据ECMAScript规范，有些分号在源码中是可以被省略的，这些被省略的分号在解析阶段会自动插入到符号流 (token stream) 中，这种机制称为 ASI。（不过实际上分号并没有真的被插入，这只是个便于解释的形象说法。）
如何避免：
而ASI机制所造成的问题是，如果我们没有不清楚地了解上面的ASI规则，我们写出来的代码就可能无法解析成我们想要的样子，导致出错，因此，我们要避免这样的情况出现，下面总结了一些我们应该注意的情况，在写代码时要注意。

下列情况，不能换行

1.continue, break, return, throw，yield这些关键字后的表达式应该和关键字放在同一行（因为asi机制会在这些token后加上；）

```
function a() {

   return

   {};

}

a()   // => undefined
```

会被解析成

```
function a() {

   return;

   {};

}

a()  // => undefined 因为return后面会被加上；导致直接return
```

而我们希望的结果其实是这样的

```
function a() {

    return {};

}

a()  // => {} (empty object)
```

2.后缀运算符 ++ 或 -- 和它的操作数应该出现在同一行

（因为asi机制会在这些运算符之前加上；）例如：

```
a

++

b
```

会被解析为

```
a;

++b;
```

总结：在上面这些生产式中不要换行，换行就自动插入分号。

下列情况，要注意在前面加上分号

1.遇到[、(、/、+、-这五个符号在他们前面显式地加上防御性分号 (defensive semicolon) 来保护其含义不被改变
例如：

```
var a = [1, [2, 3]]

[3, 2, 1].map(function(num) {

return num * num;

})
```

由于 parser 总是优先将换行符前后的符号流当作一条语句解析，parser 实际上先解析了下面的语句：

```
var a = [1, [2, 3]][3, 2, 1].map(function(num) {

return num * num;

})
```

2.for 循环头部的分号不能省略

```
var a = ['a', 'b', 'c', 'd']

var msg = ''

for (var i = 0, len = a.length

 i < len

 i++) {

 msg += a[i] + ' '

 }

 console.log(msg)
```

接抛出了语法错误 Uncaught SyntaxError: Unexpected Identifier
因为在for语句头部解析器不会进行分号插入，于是就会被解析成下面这样
(var i = 0, len = a.length \n i < len \n i++)
3.作为空语句存在的分号不能省略
例如：

```
function infiniteLoop() {

  while('empty')

  }
```

此段代码parser 会抛出语法错误 Uncaught SyntaxError: Unexpected token } 。

这是因为循环体中作为空语句而存在的 ; 不能省略。

ASI机制：

ECMAScript 标准定义的 ASI 包括 三条规则 和 两条例外。

三条规则是描述何时自动插入分号：

1. 解析器从左往右解析代码（读入 token），当碰到一个不能构成合法语句的 token 时，它会在以下几种情况中在该 token 之前插入分号，此时这个不合群的 token 被称为 offending token：
    - 如果这个 token 跟上一个 token 之间有至少一个换行。
    - 如果这个 token 是 }。
    - 如果 前一个 token 是)，它会试图把前面的 token 理解成 do...while 语句并插入分号。
2. 当解析到文件末尾发现语法还是有问题，就会在文件末尾插入分号。
3. 当解析时碰到 restricted production 的语法（比如 return），并且在 restricted production 规定的 [no LineTerminator here] 的地方发现换行，那么换行的地方就会被插入分号。

两条例外表示，就算符合上述规则，如果分号会被解析成下面的样子，它也不能被自动插入：

1. 分号不能被解析成空语句。
2. 分号不能被解析成 for 语句头部的两个分号之一。

相关资料：

ECMAScript: ASI

http://www.ecma-international.org/ecma-262/7.0/index.html#sec-rules-of-automatic-semicolon-insertion

JavaScript ASI 机制详解

https://segmentfault.com/a/1190000004548664