1. (function(){ }()); 与(function(){ })();

在运行方面并没有什么不一样。

从结果而言，两者的AST是一模一样的，最终结果都是一次**函数调用**。

从作用上看，两者的作用都是创建一个函数并调用之。

从语法的定义上看，这两者几乎是一模一样的（唯一的区别是函数表达式可以省略函数名称）。

学习于 <http://www.zhihu.com/question/20292224>

实例检测：

Test.js中代码:

(function(){

var n=0;

$(".click").click(function() {

n++;

alert(n);

});

})();

(function(){

var n=0;

$(".click2").click(function() {

n+=10;

alert(n);

});

})();

对应html代码:

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml" >

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>测试案例用</title>

<script type="text/javascript" src="js/jquery-1.11.0.js"></script>

<!-- <script type="text/javascript" src="js/test.js"></script> -->

</head>

<body>

<center class="wrap">

<input class="click" name="click" type="button" value="click here!"/>

<br/>

<input class="click2" name="click2" type="button" value="click here!"/>

<div class="output"></div>

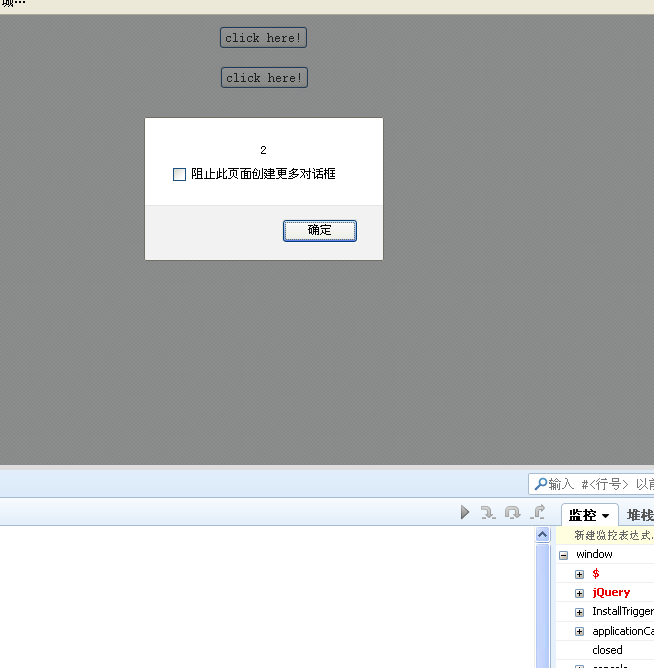
</center>

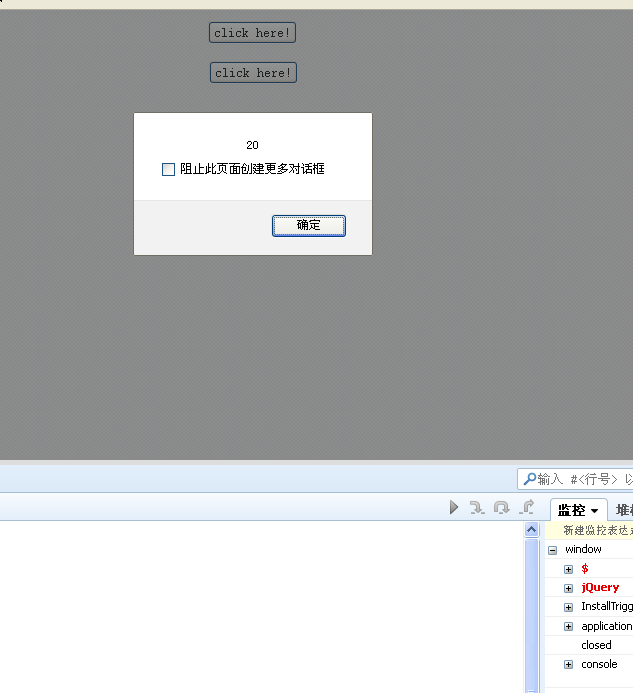
<script type="text/javascript" src="js/test.js"></script>

</body>

</html>

FF运行结果：



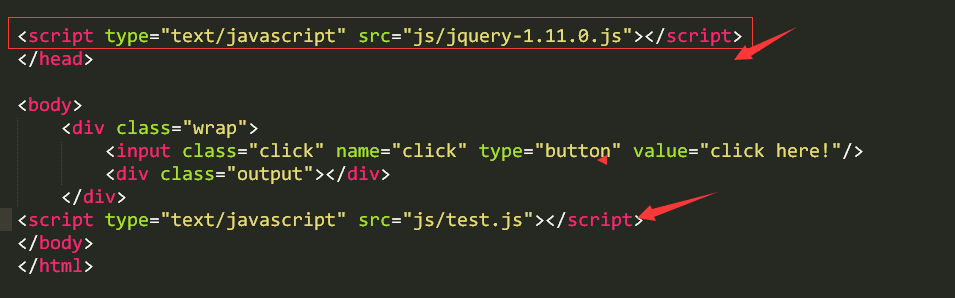


由结果可知两个闭包里面的变量n是相互分离的，在自己的闭包作用域中，当页面关闭的时候才会被销毁。同时，全局变量并没有被污染，没有全局n变量。

当测试代码形式为(function(){ }());和(function(){ }()); 或者(function(){ }());和(function(){ })();时，结果不变。

测试代码见test1

1. 带有选择器的js执行需要在DOM创建之后



如上图，当将test.js放在head里面时，由于body中内容尚未加载，因此在DOM树中找不到input.click，故而test.js中的代码并未运行。(在ff中设置断点可以很清楚的看到)

如果将$(“.click”)改为$(window)时可以运行的，因为window是html子节点，当html标签创建的时候window节点也就同时创建了。

当将test.js放在body最下面或者对其设置延迟加载(defer,setTimeout())或者使用$(document).ready();包装的时候，运行时body中的DOM已经完成，因此代码会被顺利执行。

测试代码见test1

3. 传入函数的参数不确定的情况

直接定义函数时使用() 不定参数

在函数内部用arguments判断参数个数并引用。当然也可以直接用传入的参数。见test2

//测试参数

function add() {

return c + b;

}

var a = 1,

b = 2,

c = 3;

console.log(add(a, b));

4.异步永远比同步慢

//测试

(function () {

var i = 0;

setInterval(function(){

if(i<10){

i++;

}

},10);

return console.log(i);

}());

输出0

5.newWindow中代码为js页面之间的通信四种解决方式