引子是一篇文章，原文地址：

http://www.jb51.net/article/100661.htm

讲述了几种面对异步的返回值的应对方法。

看完之后，感觉promise对象和**async最好用。但是目前es7好像支持还比较一般，所以先用比较原始的promise对象。**

今天研究一个小问题： 怎么拿到JavaScript异步函数的返回值？

**1.错误尝试**

当年未入行时，我的最初尝试：

[?](http://www.jb51.net/article/100661.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | <script>  function getSomething() {   var r = 0;   setTimeout(function() {   r = 2;   }, 10);   return r;  }    function compute() {   var x = getSomething();   alert(x \* 2);  }  compute();  </script> |

**2.回调函数**

弹出的不是4，而是0，后来知道这是异步的问题，

要用回调技术来做：

[?](http://www.jb51.net/article/100661.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | <script>  function getSomething(cb) {   var r = 0;   setTimeout(function() {   r = 2;   cb(r);   }, 10);  }    function compute(x) {   alert(x \* 2);  }  getSomething(compute);  </script> |

**3.promise**

回调函数真是个好东西，然后一直这么写代码写了很久。遇到异步就传函数！！后来我知道有promise这一个东西，专门解决由于回调函数引起的问题，又学会了promise：

[?](http://www.jb51.net/article/100661.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <script>  function getSomething() {   var r = 0;   return new Promise(function(resolve) {   setTimeout(function() {    r = 2;    resolve(r);   }, 10);   });  }    function compute(x) {   alert(x \* 2);  }  getSomething().then(compute);  </script> |

promise仍然没有放弃回调，只是回调的位置发生了改变。

**4.generator**

再后来我又学会了generator，知道其有中断函数执行的能力，又做了新的尝试：

[?](http://www.jb51.net/article/100661.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <script>  function getSomething() {   var r = 0;   setTimeout(function() {   r = 2;   it.next(r);   }, 10);  }    function \*compute(it) {   var x = yield getSomething();   alert(x \* 2);  }  var it = compute();  it.next();  </script> |

同步的写法，能实现异步的逻辑，感觉高大上了很多。

**5.promise + generator**

后来又听说promise加generator，才是异步的完美方式，赶紧用高射炮打蚊子（这个例子，还不足以说出二者在一起用的好处）：

[?](http://www.jb51.net/article/100661.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | <script>  function getSomething() {   var r = 0;   return new Promise(function(resolve) {   setTimeout(function() {    r = 2;    resolve(r);   }, 10);   });  }    function \*compute() {   var x = yield getSomething();   alert(x \* 2);  }  var it = compute();  it.next().value.then(function(value) {   it.next(value);  });  </script> |

**6.async**

心想这算是够屌的吧，后来又听说es7给出了终极方案:async。

作为爱学习的少年，心想自己不能被落下：

[?](http://www.jb51.net/article/100661.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | <script>  function getSomething() {   var r = 0;   return new Promise(function(resolve) {   setTimeout(function() {    r = 2;    resolve(r);   }, 10);   });  }    async function compute() {   var x = await getSomething();   alert(x \* 2);  }  compute();  </script> |

到这里终于长出了一口气。

——————————————————————————————————分割线————————————————————————————————

下面开始说promise的具体用法