刚学的时候见到闭包很头疼，用的也很少，现在好好写一写。

先看一句话，从其他大神博客抄来的，具体哪位原创现在已经分不太清楚了。。。

===========================================================

只要存在调用内部函数的可能，js就会保存被引用的函数和变量，js在运行的时候会跟踪这个函数内部的所有变量，直到所有的变量都废弃的时候，js的垃圾收集器才会释放相应的内存空间。

稍微理解一下就是，父级函数定义的变量在子函数的作用域链中，子函数没有被销毁，它作用域链中的所有变量和函数就都会被保留。

看完这句话我们再来理解闭包，引用作用域链的那个例子，



这里很容易犯错，当我们依次点击按钮1，2，3时会输出什么？

答案是 button4 button4 button4

为啥呢？ 页面加载完之后，我们就会一次给三个按钮添加监听器，

当我们点击按钮的时候，会触发监听的点击事件，这时候调用预先设定好的回调函数

这个函数会输出 button+i 问题就出现在i上面，当我们执行完添加监听事件之后，这时候i就已经是4了，当执行回调函数的时候，function(){}本身没有i这个变量，根据作用域链，会往上层函数找，此时i为4，找到了，直接返回，所以点哪个按钮都会输出button4

这里我们这时候在来看,因为点击时间随时都可能发生，所以这个函数会被一直保留，它的父级函数由于在其作用域链里面，所以也会一直保存，i也会一直保持为4的状态。

再次更新。

闭包简单的说就是函数套函数，里面的函数可以引用外面函数的参数和变量。

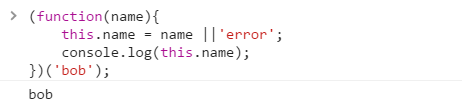
也是为了读取其他函数内部的变量。

闭包一般有两种形式，

函数作为返回值，或者函数作为参数传递。

函数作为返回值的时候，因为返回的是函数，它可以自己创建一个独立的作用域，而我们此时都引用了父级函数的变量，那么，这个变量就不能被销毁了，会一直在内存中，这也就是为什么上面的例子i会一直是4.

匿名自执行函数，闭包的必备知识，也给大家简答介绍一下



后面的括号是传递给里面函数的参数。

这种函数，匿名，并且会立即执行，因为外部无法引用他内部的变量，所以它执行完之后会很快被释放，这种机制不影响全局变量。

2.1 进行赋值操作的方式

var a=function(){

　　console.log('test');

}();

　　2.2 给函数体加小括号的方式

(function aaa() {

console.log('test');

})();

　　2.3 在函数体前加逻辑运算符的方式 eg：

!function() {

console.log('test');

}();

　　2.4 在函数体前加数学运算符的方式 eg：

+function () {

console.log('test');

}();

闭包的用处

第一种，自执行函数，不污染全局变量，代码后面补。

第二种，将将要多次引用的函数放到内存中，提高性能，

第三种，封装性，可以封装私有变量和私有方法。

第四种，建立一个类，这个不太懂