



每当执行对应的函数，自动创建对应的执行环境，并将执行环境推入环境栈中，除了执行环境，同时也会自动创建对应的作用域链和活动对象，当函数运行完毕，环境栈被推出，作用域链和活动对象同时也被销毁，返回到上一层函数。

但如果内部函数被return出去，以后还需要用到内部变量对象，那么作用域链会被销毁，活动对象却依旧保存不被销毁，这就是闭包的原理，当设置为null，闭包才会回收。

虽然相关概念很多也很全，但如何才能让一个即使不清楚的人也能够马上理解其中大致原理呢？

我把它的整个原理比作翻书，当我打开一本书a，一页页从头开始翻阅，就好像js从上至下一行行开始执行，因为js是单线程，的确非常恰当。当我翻到某一个地方，上面写着需要引入书b，这就好像我执行某个函数，这时候，我就把书b拿过来，嵌入书本a,然后我开始一页页翻阅书b，（这等价于将函数的执行环境推入环境栈中）。当书b翻完，我就把它从书a当中移开，继续翻阅a剩下的内容。当然书本b可以引入其它书。整个js就是类似这种机制有条不紊的执行着。