22 环境

我们可以认为Lua把所有的全局变量保存在一个称为全局环境的普通表\_G中。

22.1 具有动态名称的全局变量

22.2 全局变量的声明

编程：禁止创建全局变量，并声明一个创建全局变量的方法

如果要测试一个变量是否存在，并不能简单地将它与nil比较，而应该使用rawget

22.3 非全局变量

22.4 使用\_ENV

\_ENV永远指向的是当前的环境，而假设在可见且无人改变过其值的前提下，\_G通常指向的是全局变量。

22.5 环境和模块

22.6 \_ENV和load

5.1之前, 全局变量存储在\_G这个table中，a = 1 相当于：\_G['a'] = 1

但在5.2之后，引入了\_ENV叫做环境，与\_G全局变量表产生了一些混淆，需要从原理上做一个理解。

在5.2中， 操作a = 1相当于\_ENV['a'] = 1

其次要格外注意\_ENV不是全局变量，而是一个upvalue(非局部变量)。

其次，\_ENV['\_G']指向了\_ENV自身，这一目的是为了兼容5.1之前的版本，

\_G['a'] = 2 ，在5.2中，这相当于\_ENV['\_G']['a']，为了避免5.1之前的老代码在5.2中运行错误，所以5.2设置了\_ENV['\_G']=\_ENV来兼容这个问题。

在5.1中，我们可以为一段代码块（或者函数）设置环境，使用函数setfuncs，这样会导致那一段代码/函数访问全局变量的时候使用了setfuncs指定的table，而不是全局的\_G。

在5.2中，setfuncs遭到了废弃，因为引入了\_ENV。 通过在函数定义前覆盖\_ENV变量即可为函数定义设置一个全新的环境