20 元表和元方法

元表可以修改一个值在面对一个未知操作时的行为。

当Lua语言试图将两个表相加时，它会先检查两者之一是否有元表且该元表中是否有\_\_add字段。

使用函数setmetatable来设置或修改任意表的元表。

在Lua语言中，只能为表设置元表，如果要为其他类型的值设置元表，则必须通过C代码或调试库完成。

字符串标准库为所有字符串都设置了同一个元表，而其他类型在默认情况下都没有元表。

20.1 算术运算相关的元方法

除了加法和乘法外，还有减法，除法，floor除法，负数，取模，幂运算。

类似地，为操作也有元方法：按位与，按位或，按位异或，按位取反，向左移位和向右移位。

还可以用\_\_concat字段定义连接运算符的行为。

20.2 关系运算相关的元方法

元表还允许我们指定关系运算符的含义，其中的元方法包括等于(\_\_eq)，小于(\_\_lt)和小于等于(\_\_le)。

其他三个关系运算符没有单独的元方法，Lua会将a~=b转换为not(a==b)，a<b转换为b>a，a>=b转换为b<=a。

20.3 库定义相关的元方法

函数print总是调用tostring来进行格式化输出。不过，当对值进行格式化时，函数tostring会首先检查值是否有一个元方法\_\_tostring。

如果在元表中设置\_\_metatable字段，那么getmetatable会返回这个字段的值，而setmetatable则会引发一个错误。

从Lua5.2开始，函数pairs也有对应了元方法，因此我们可以修改表被遍历的方式和为非表的对象增加遍历的行为。

20.4.1 表相关的元方法

当访问一个表中不存在的字段时，这些访问会引发解释器查找一个名为\_\_index的元方法。

元方法\_\_index不一定是一个函数，还可以是一个表。

将一个表用作\_\_index元方法为实现单继承提供了一种简单快捷的方法。

如果我们希望在访问一个表时不调用\_\_index元方法，那么可以使用函数rawget。

20.4.2 \_\_newindex元方法

当对一个表中不存在的索引赋值时，解释器就会查找\_\_newindex元方法。

如果\_\_newindex是一个表，解释器就在此表中执行赋值，而不是在原始的表中进行赋值。

还有一个原始函数允许我们绕过元方法:调用rawset(t,k,v)来等价于t[k]=v,但不涉及任何元方法。

20.4.3 具有默认值的表

20.4.4 跟踪对表的访问

20.4.5 只读的表