Lua metatable

Each value in Lua can have a table. A metatable is an ordinary Lua table that defines the behavior of the original value under certain actions. You can set a specific field as a key value in the value of the table, the value of the metatable as the corresponding operation. For example, when a numeric value is used as an operand of addition, Lua checks whether the "\_\_add" field in its meta table has a function. If so, Lua calls it to perform addition. We call the key in the table as an event (event), which is called the value element method (metamethod). The event in the preceding example is "add", and the meta method is a function of the addition. You can query any value of the table by function getmetatable.

It can be replaced by the function setmetatable table of the table. You cannot change other types of meta tables from Lua (except for the debug library); you must use C API to do so.

1. teacher = {}
2. teacher.mt = {}
3. teacher.mt.\_\_index = {yes = 1, no = 2,ok = 3,truth = 4}
4. function teacher.new (ttt)
5. return setmetatable(ttt,teacher.mt)
6. end
7. student = {no = 0, ok = 99}
8. teacher.new(student)
9. print(student.yes)
10. print(student.ok)
11. print(student.no)

Output:

1. 1
2. 99
3. 0
4. [Finished in 0.0s]

setmetatable (table, metatable)

if there is the key value in metastable, then the  method setmetatable will fail. As the example, set no = 2 failed .

1. a = {}
2. a.asd = "ddd"
3. function a:new(b)
4. b = b or {}
5. return setmetatable(b, {
6. \_\_index = self
7. })
8. end
9. function a.fuck()
10. print("hehe,just a joke")
11. end
12. c = a:new()
13. c.fuck()
14. c.fuck = "jjjj"
15. print(c.fuck)
16. a.fuck()

Output：

1. hehe,just a joke
2. jjjj
3. hehe,just a joke