Lua实现类创建对象：

local Account={balance = 0}

function Account:new(o)

o=o or {}

setmetatable(o,self)

self.\_\_index=self

return o

end

实现私密性

function newAccount(initialBalance)

local self={

balance=initialBalance,

haha =10

}--私有变量

--私有方法

local withdraw=function(v)

self.balance=self.balance+v

end

local deposit=function(v)

self.balance=self.balance+v

end

return {

deposit=deposit--放入接口变成共有方法

}

End

弱引用表，如果一个数据只被弱引用了 则会被回收

a={}

--回收key

b={\_\_mode="k"}

--字符串，数字，布尔 不会从弱引用的table中删除

setmetatable(a,b)

a[1]=10

--数字不会被回收

collectgarbage()

print(a[1])--10

local key = {}--key被引用着 所以不会被回收

a[key]=10

collectgarbage()

print(a[key])--10

a[2]={}

collectgarbage()

print(a[2])--table...

--回收value

b.\_\_mode="v"

collectgarbage()

print(a[2])--nil

b.\_\_mode="kv"

string库

local str = "A122df"

print(str:upper())--A122DF

print(string.upper(str))--A122DF

print(str:lower())--a122df

print(string.lower(str))--a122df

print(string.rep(str,3))--a122dfa122dfa122df

print(string.char(97))--a

print(string.char(97,98,99))--abc

print(string.byte("abc"))--97

print(string.byte("abc",2))--98

print(string.byte("abc",-2))--99

print(string.byte("abc",2,-1))--98 99

print(string.format("abc%.4f",3.1415926))--3.1416

print(string.sub(str,5,6))--df

print(string.sub(str,5,-1))--df

print(string.sub(str,5))--df

print(string.find("abchellode","hello"))--4 8

print(string.match("abchellode","hello"))--"hello"

print(string.gsub("abchellodhelloe","hello","w"))--"abcwdwe" 2

--把句子中的单词收集到word中

local word = {}

local xx = "im a grate man"

for w in string.gmatch(xx,"%a+") do

word[#word+1]=w

end

io库

local file = io.open("my.txt","w")--打开，或者创建并打开文件

io.input("in.txt")--打开指定文件，设为当前的输入文件

io.output("out.txt")--打开指定文件，设为当前的输出文件

io.write("haha","txt")--写入到当前输出文件（out.txt)

io.write("haha","txt")--写入到当前输出文件（out.txt)

file:write("heihei")--写入到my.txt

--local r = io.read('\*number')--读取in.txt的一个数字

--local r = io.read('\*line')--读取in.txt的第一行

--local r = io.read('\*line')--读取in.txt的第二行

--local r = io.read('\*all')--读取全部（已经读取了两行，会读取剩下的）

--local r = io.read('\*number')

--逐行读取 保存到ltab中

local ltab = {}

for line in io.lines() do

ltab[#ltab]=line

end

io.input():close()

file:close()

系统库

--下面两个方法相反

os.time({year=1970,month=1,day=1,hour=9})--3600

local timeTab = os.date("\*t",3600)

--wday 1表示星期天 yday表示一年中的第几天

--打印今天的星期与月份 --%y两位数年份 %Y完整年份 等

print(os.date("today is %A in %B"))

print(os.getenv("LUA\_PATH"))—打印环境变量

lua C API

#include <stdio.h>

#include "lua.h"

#include "lauxlib.h"

static coid stackDump(lua\_State\* L)

{

/\*打印\*/

}

int main(void){

lua\_State \*L luaL\_newstate();

lua\_pushboolean(L,1);

lua\_pushnumber(L,10);

lua\_pushnil(L);

lua\_pushstring(L,"hello");

stackDump(L); /\*true 10 nil 'hello'\*/

lua pushvalue(L,-4)

stackDump(L); /\*true 10 nil 'hello' true\*/

lua\_replace(L,3);

stackDump(L); /\*true 10 true 'hello'\*/

lua\_settop(L,6);

stackDump(L); /\*true 10 true 'hello' nil nil\*/

lua\_remove(L,-3);

stackDump(L); /\*true 10 true nil nil\*/

lua\_settop(L,-5);

stackDump(L); /\*true\*/

lua\_close(L);

return 0;

}