ScutSDK使用说明文档

版本：V 1.0

# 前言：

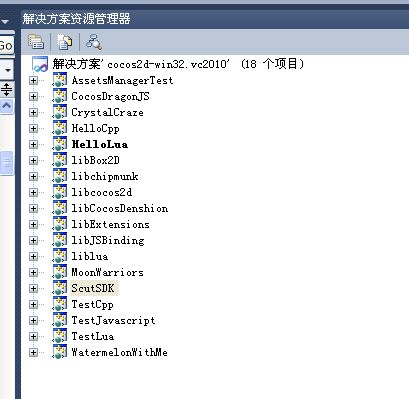
ScutSDK是一个开源的项目，开发的宗旨是为了能让网游开发和单机一样简单，以便于降低有志于游戏开发的小伙伴们进入网络游戏开发的门槛。ScutSDK彻底打通了Scut开源游戏服务器引擎与Cocos2dX开源项目间的通信，从而和Cocos2dX一起实现整套的网络游戏解决方案。

## 1.1 使用准备

（熟悉Cocos2dx的小伙伴们可以跳过此部分）

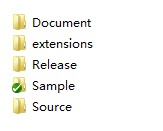
下载Cocos2dx的最新版本，下载地址：<http://www.cocos2d-x.org/download>

打开Cocos2dx，先编译运行HelloLua项目。

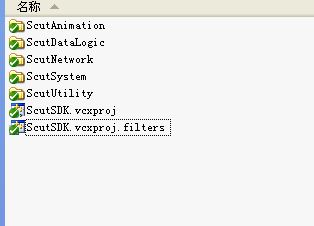


## 1.2 文件目录介绍

获取ScutSDK可以从github进行更新（<https://github.com/ScutGame/Scut>），也可以从官网下载（<http://www.scutgame.com/download>），目录结构如下：



1. Source目录为ScutSDK的源码。想近距离接触ScutSDK的小伙伴们就需要对它有所了解。本文档后面会对Source做详细的接口说明介绍和编译方法介绍。



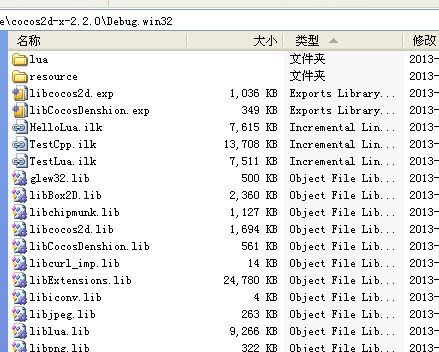
1. Sample目录为DEMO需要的脚本和资源。DEMO是小伙伴们了解ScutSDK使用方法最直接的方式了。



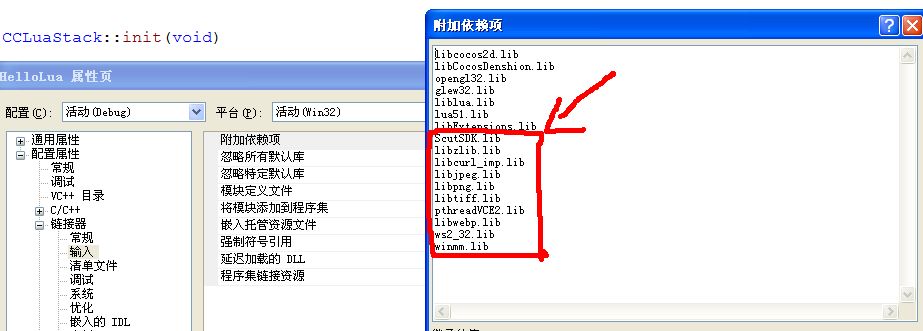
1. release目录为ScutSDK的静态库和所需要包含的头文件。

# ScutSDK之使用篇（Win32平台）

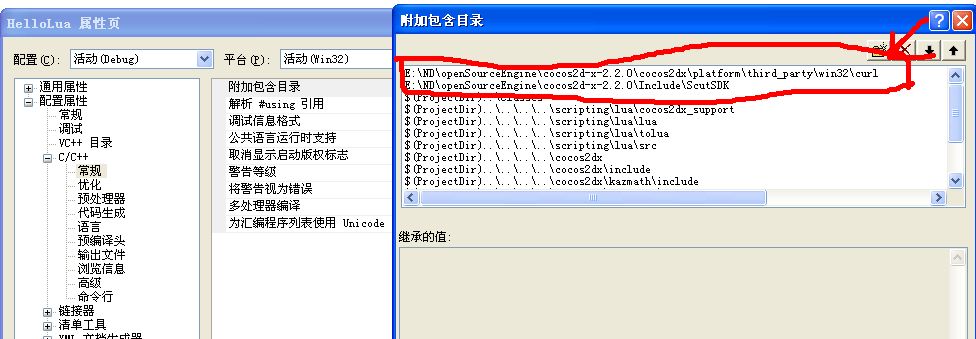
运行过Cocos2dx的helloLua 例子后会在cocos2dx的工作目录下生成一个Debug.win32目录。这个就是工作目录，将release目录下的所有lib文件和dll文件拷贝至此目录。



复制 samplecode目录下的lua 和resource 到Debug.win32下，并将新增的lib库添加到项目工程中。



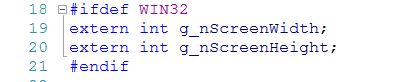
指定ScutSDK的头文件包含目录。



注意：因为ScutSDK有使用到cocos2dx的curl静态库，所以这边需要指定依赖的cocos2dx curl头文件目录，并在cocos2dx的入口添加相关的ScutSDK相关初始化函数。

1. 修改AppDelegate.cpp 新增如下代码

|  |
| --- |
| #ifdef WIN32  extern int g\_nScreenWidth;  extern int g\_nScreenHeight;  #endif |



|  |
| --- |
| CCEGLView::sharedOpenGLView()->setDesignResolutionSize(g\_nScreenWidth, g\_nScreenHeight, kResolutionNoBorder); |

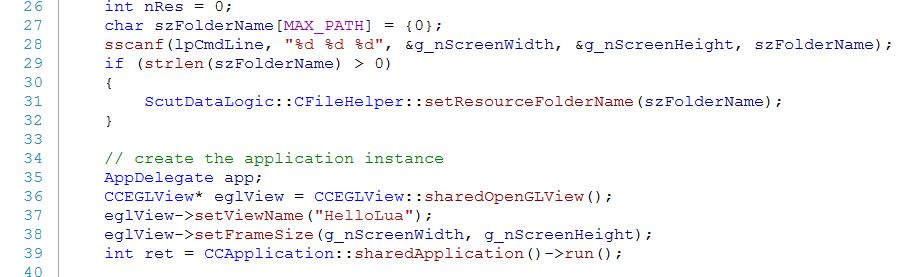


1. 修改main.cpp 新增如下代码

|  |
| --- |
| int g\_n\_ScreenWidth = 960;  int n\_ScreenHeight = 640 |

D:\桌面\截图10.jpg

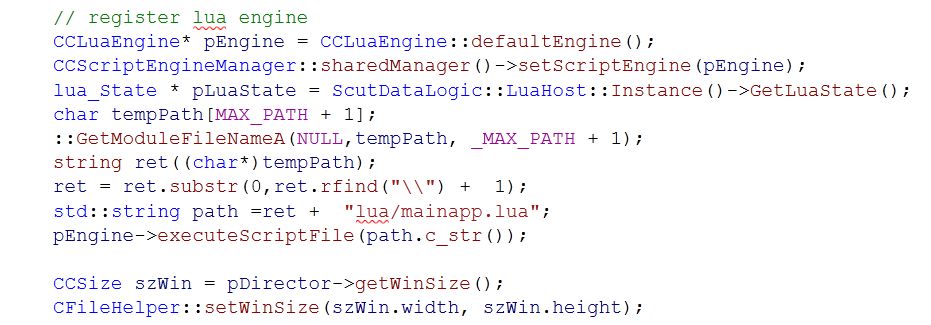
|  |
| --- |
| int nRes =0;  char szFolderName[MAX\_PATH] = {0};  sscanf(lpCmdLine, “%d %d %d”, &g\_nScreenWidth, &g\_nScreenHeight, szFolderName);  if(strlen(szFolderName) > 0)  {  ScutDataLogic::CFileHelper::setResourceFolderName(szFolderName);  }  AppDelegate app;  CCEGLView\* eglView = CCEGLView::sharedOpenGLView();  eglView->setViewName(“HelloLua”);  eglView->setFrameSize(g\_nScreenWidth, g\_nScreenHeight);  int ret = CCApplication::sharedApplication->run(); |



这几处的代码是将屏幕的分辨率改为默认960 \* 640，同时支持命令行输入分辨率大小。

1. 修改AppDelegate.cpp

|  |
| --- |
| CCLuaEngine\* pEngine = CCLuaEngine::defaultEngine();  CCScriptEngineManager::sharedManager()->setScriptEngine(pEngine);  Lua\_State\* pLuaState = ScutDataLogic::LuaHost::Instance()->GetLuaState();  Char tempPath[MAX\_PATH + 1];  String ret((char\*)tempPath);  Ret = ret.substr(0,ret.rfind(“\\”) + 1);  Std::string path = ret + “lua/mainapp.lua”;  pEngine->executeScriptFile(path.c\_str());  CCSize szWin = pDirector->getWinSize();  CFileHelper::setWinSize(szWin.width, szWin.height); |



此处修改是为了加载samplecode的脚本。

d.在 AppDelegate.cpp 和 main.cpp都要包含ScutSDK.h头文件。

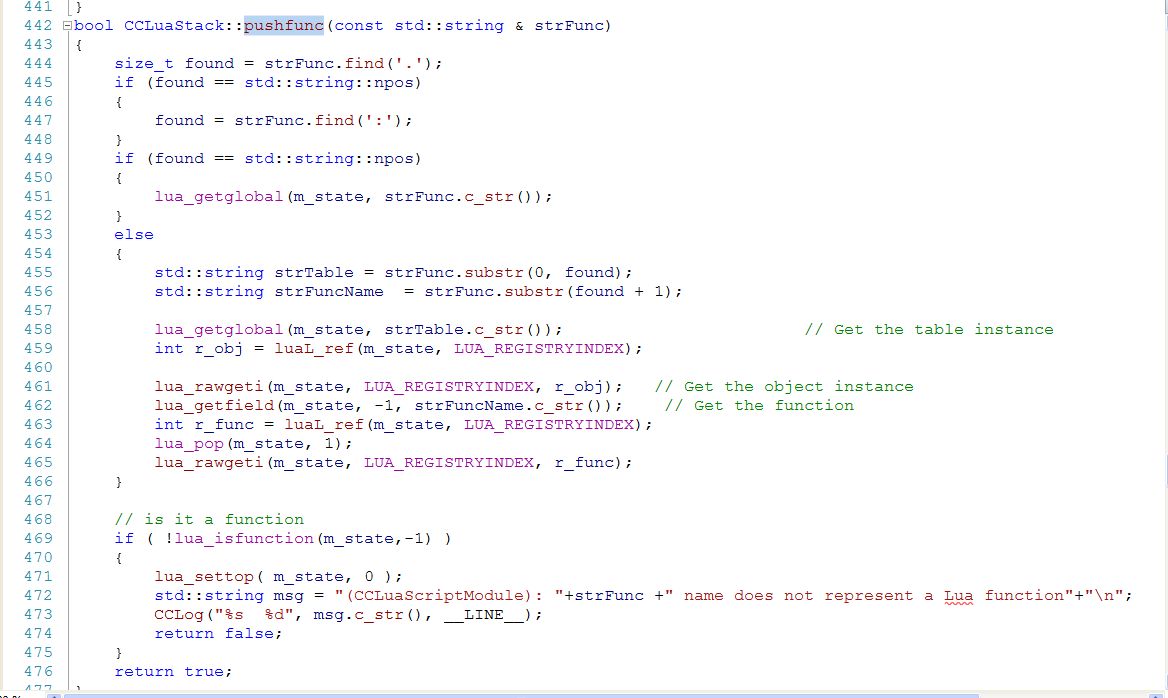
e.在CCLuaStack.h 中添加如下代码

|  |
| --- |
| Virtual bool pushfunc(const std::string &strFunc);  Virtual void executeLogEvent(std::string &func, std::string &errlog); |

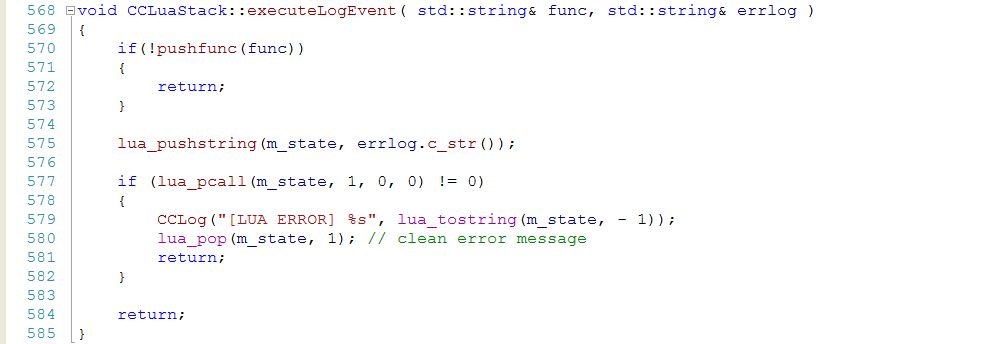
D:\桌面\截图13.jpg

f.在CCLuaStack.cpp 中添加如下代码

|  |
| --- |
| bool CCLuaStack::pushfunc(const std::string & strFunc)  {  size\_t found = strFunc.find(‘.’);  if(found == std::string::npos)  {  lua\_getglobal(m\_state, strFunc.c\_str());  }  else  {  Std::string strTable = strFunc.substr(0,found);  Std::string strFuncName = strFunc.substr(found + i);  lua\_getglobal(m\_state, strTable.c\_str());  int r\_obj = luaL\_ref(m\_state, LUA\_REGISTRYINDEX);  lua\_rawgeti(m\_state, LUA\_REGISTRYINDEX, r\_obj);  lua\_getfield(m\_state, -1, strFuncName.c\_str());  int r\_func = luaL\_ref(m\_state, LUA\_REGISTRYINDEX);  lua\_pop(m\_state, 1);  lua\_rawgeti(m\_state, LUA\_REGISTRYINDEX, r\_func);  }  if( !lua\_isfunction(m\_state, -1))  {  lua\_settop(m\_state, -1);  std::string msg = “{CCLuaScriptModule}:” + strFunc +  “name dose not represent a Lua function” + “\n”;  CCLog(“%s %d”, msg.c\_str(), \_\_LINE\_\_);  }  return true;  } |



|  |
| --- |
| void CCLuaStack::executeLogEvent(std::string &func, std::string &errlog)  {  if(!pushfunc(func))  {  Return;  }  lua\_pushstring(m\_state, errlog.c\_str());  if(lua\_pcall(m\_state, 1, 0,0) != 0)  {  CCLog(“[LUA ERROR] %s”, lua\_tostring(m\_state, -1));  lua\_pop(m\_state, 1);  return;  }  return ;  } |



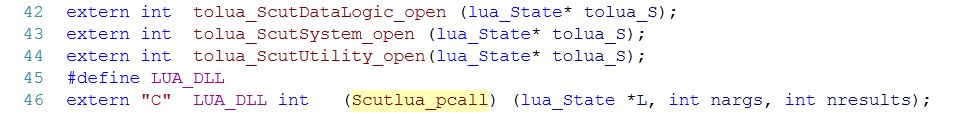
在文件的末尾处添加

|  |
| --- |
| #include “CCLuaEngine.h”  int nderror\_handler(lua\_State\* L)  {  lua\_Debug debug\_info;  int level = 0;  std::string err;  char tmp[10];  if(lua\_gettop(L) > 0 && lua\_isstring(L, -1))  {  err += lua\_tostring(L, -1);  }  while()(lua\_getstack(L, level++, &debug\_info))  {  lua\_getinfo(L, “1”, &debug\_info);  lua\_getinfo(L, “n”, &debug\_info);  lua\_getinfo(L, “s”, &debug\_info);  err += ‘\n’;  err += ‘[’;  err += debug\_info.what;  err += ‘][’;  err += ‘[]’;  if(debug\_info.name) err += debug\_info.name;  err += ‘]@[’;  err += debug\_info.source;  err += ‘]’;  }  std::string& err\_handler = cocos2d::CCDirector::sharedDirector()->GetErrorHandler();  if(err\_handler.size() > 0)  {  cocos2d::CCLuaEngine::defaultEngine()->getLuaStack()->executeLogEvent(err\_handler, err);  }  return 1;  }  }  extern "C" LUA\_DLL int Scutlua\_pcall(lua\_State \*L, int nargs, int nresults)  {  lua\_pushcfunction(L, nderror\_handler);  lua\_insert(L, -2-nargs);  return lua\_pcall(L, nargs, nresults, -2-nargs);  } |

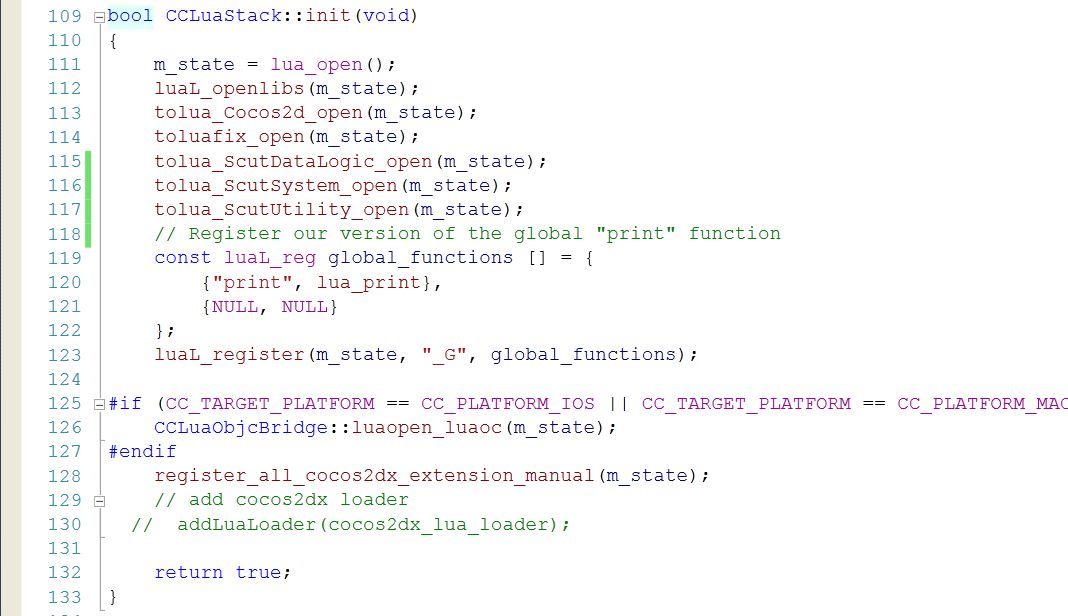
这些是打印LUA LOG所使用的方法。

g. 在CCLuaStack.cpp 中添加如下代码

|  |
| --- |
| extern int tolua\_ScutDataLogic\_open(lua\_State\* toua\_S);  extern int tolua\_ScutSystem\_open(lua\_State\* tolua\_S);  extern int tolua\_ScutUtility\_open(lua\_State\* tolua\_S);  #define LUA\_DLL  extern “C” LUA\_DLL int (Scutlua\_pcall) (lua\_State \*L, int nargs, int nresoults); |



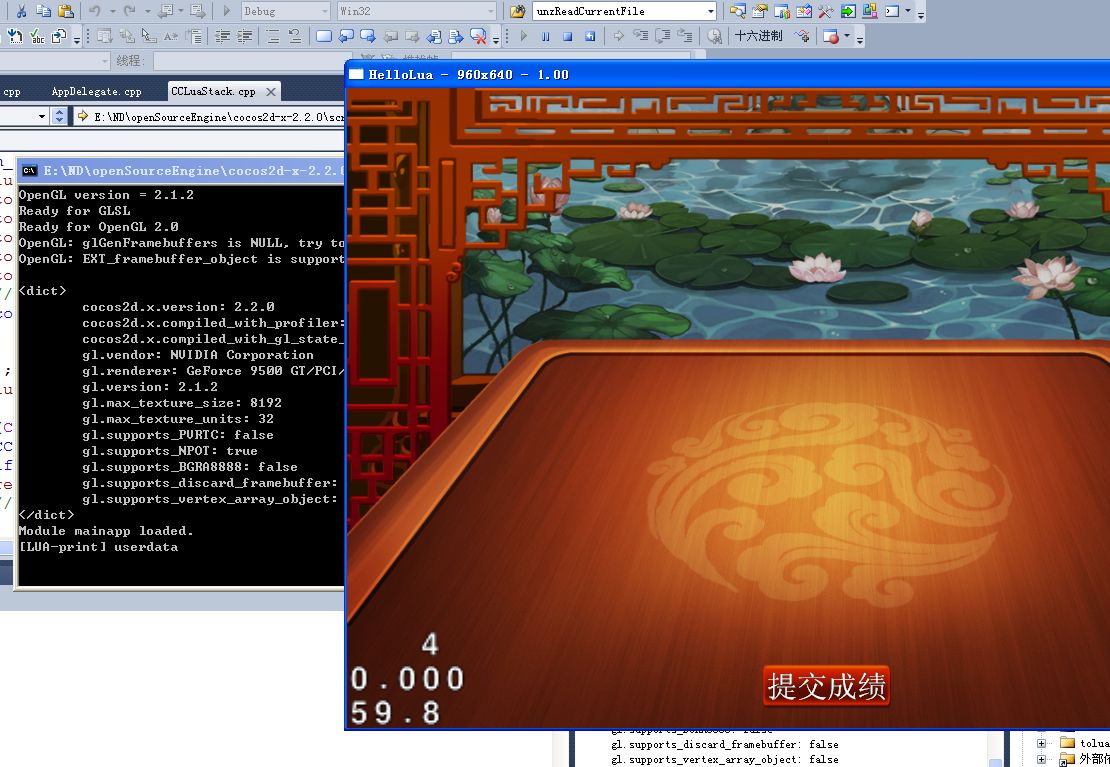
|  |
| --- |
| bool CCLuaStack::init(void)  {  m\_state = lua\_open();  luaL\_openlibs(m\_state);  tolua\_Cocos2d\_open(m\_state);  toluafix\_open(m\_State);  tolua\_ScutDataLogic\_open(m\_state);  tolua\_ScutSystem\_open(m\_state);  tolua\_ScutUtility\_open(m\_state);  const luaL\_reg global\_functions [] = {  {“print”, lua\_print};  {NULL, NULL}  };  luaL\_register(m\_state, “\_G”, global\_functions);  #if (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_IOS || CC\_TARGET\_PLATFORM == CC\_PLATFORM\_MAC)  CCLuaObjcBridge::luaopen\_luaoc(m\_state);  #endif  Return true;  } |



这些方法是为了lua能够调用ScutSDK的C++方法。

h. 拷贝 extensions 文件夹下 CCDirector.cpp 和 CCDirector.h 两个文件，替换cocos2d-x-2.2.0\cocos2dx目录下的同名文件。ScutSDK在CCDirector中添加了一些和网络相关的监控方法。接下来运行cocos2d-x-2.2.0\tools\tolua++目录下build.bat。这样是为了将刚才替换的CCDirector 中的方法导出成LUA接口。

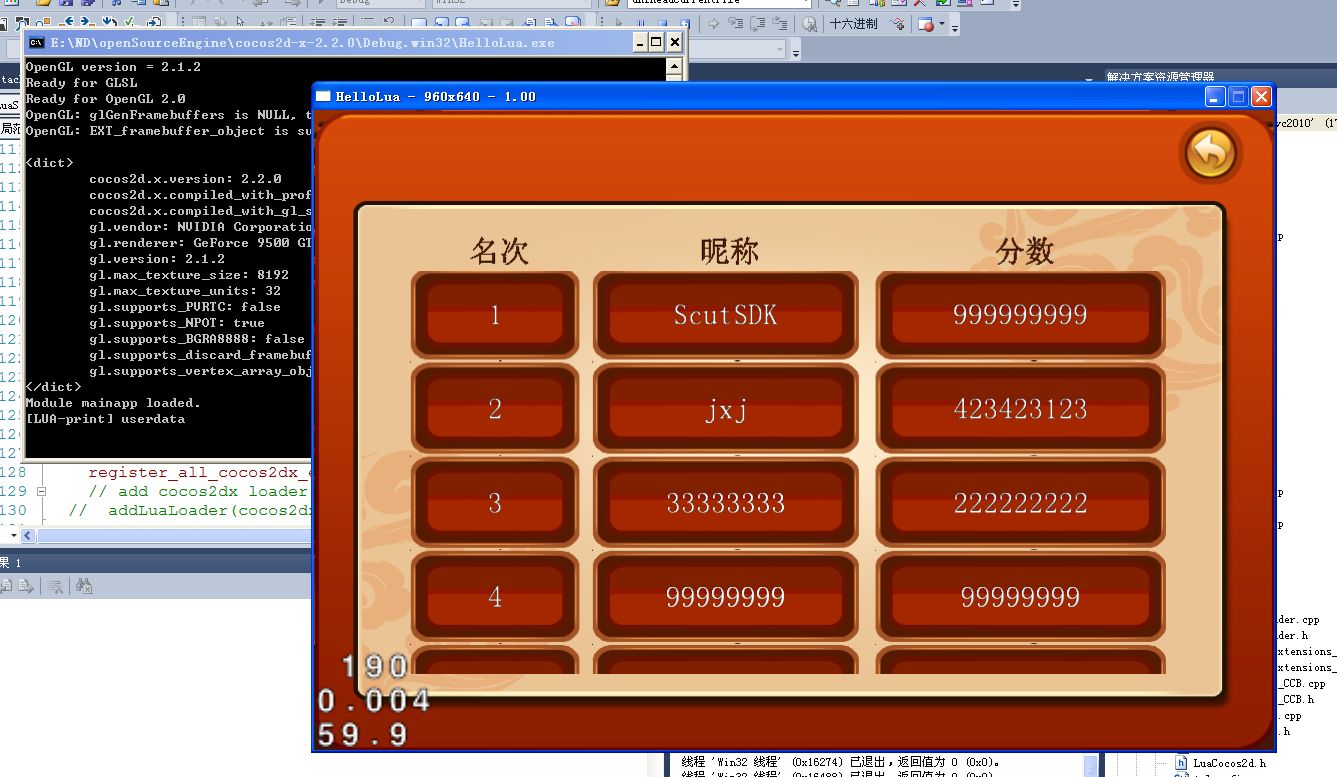
最后，可以运行hellolua了。



试试网络请求，提交一个成绩。



再请求一下排行版的信息。



# ScutSDK之编译篇

# ScutSDK 之LUA LIB 讲解篇