ScutSDK使用说明文档

版本：V 1.0

# 前言：

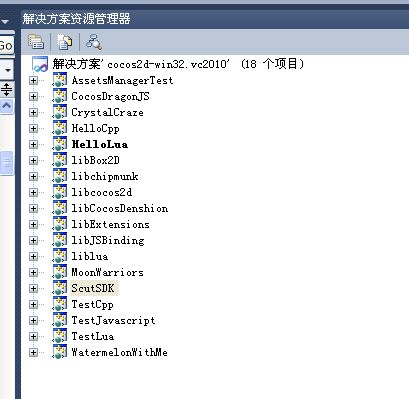
ScutSDK是一个开源的项目，开发的宗旨是为了能让网游开发和单机一样简单，以便于降低有志于游戏开发的小伙伴们进入网络游戏开发的门槛。ScutSDK彻底打通了Scut开源游戏服务器引擎与Cocos2dX开源项目间的通信，从而和Cocos2dX一起实现整套的网络游戏解决方案。

## 1.1 使用准备

（熟悉Cocos2dx的小伙伴们可以跳过此部分）

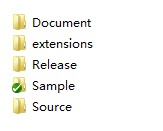
下载Cocos2dx的最新版本，下载地址：<http://www.cocos2d-x.org/download>

打开Cocos2dx，先编译运行HelloLua项目。

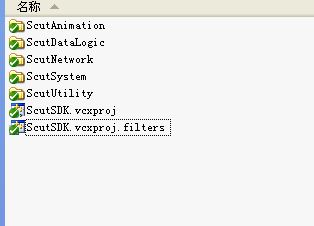


## 1.2 文件目录介绍

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。



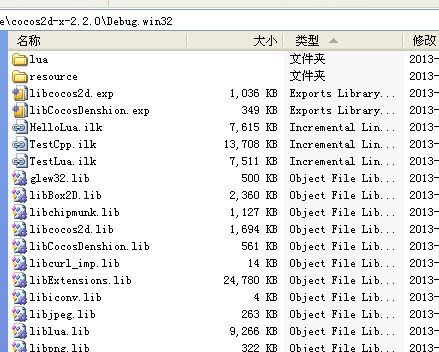
1. Sample目录为DEMO需要的脚本和资源。DEMO是小伙伴们了解ScutSDK使用方法最直接的方式了。



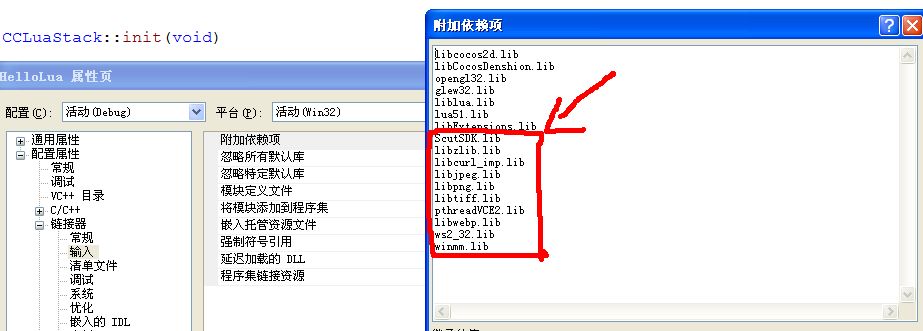
1. release目录为ScutSDK的静态库和所需要包含的头文件。

# ScutSDK之使用篇（Win32平台）

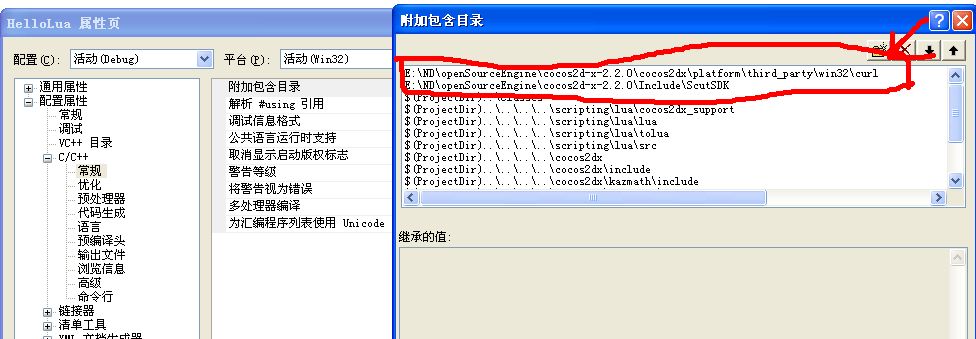
运行过Cocos2dx的helloLua 例子后会在cocos2dx的工作目录下生成一个Debug.win32目录。这个就是工作目录，将release目录下的所有lib文件和dll文件拷贝至此目录。



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。

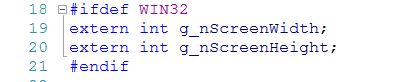


指定ScutSDK的头文件包含目录。



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。

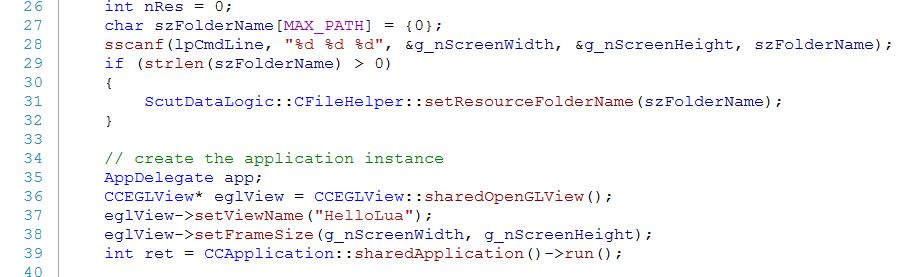
a.修改AppDelegate.cpp 新增如下代码





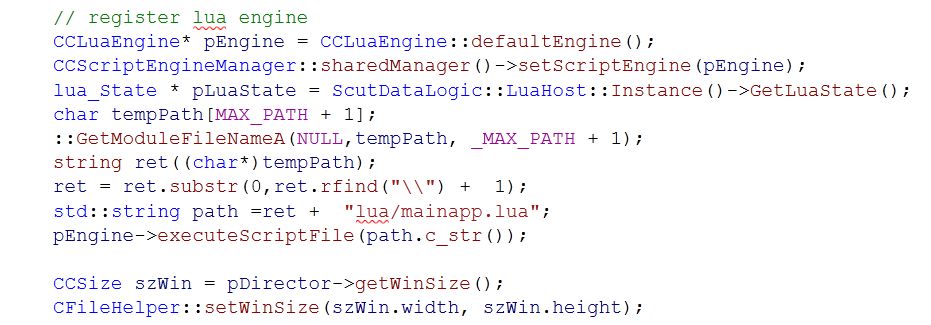
b.修改main.cpp 新增如下代码

D:\桌面\截图10.jpg



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。

c.修改AppDelegate.cpp



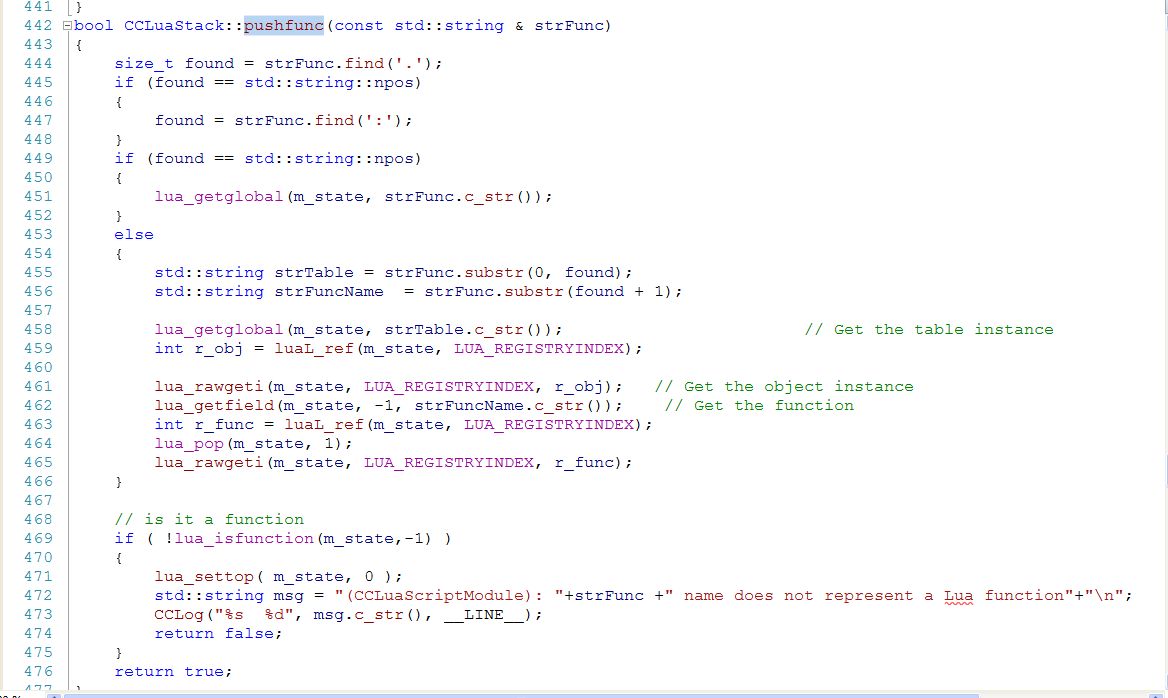
此处修改是为了加载samplecode的脚本。

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

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)

{

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 + 1);

lua\_getglobal(m\_state, strTable.c\_str()); // Get the table instance

int r\_obj = luaL\_ref(m\_state, LUA\_REGISTRYINDEX);

lua\_rawgeti(m\_state, LUA\_REGISTRYINDEX, r\_obj); // Get the object instance

lua\_getfield(m\_state, -1, strFuncName.c\_str()); // Get the function

int r\_func = luaL\_ref(m\_state, LUA\_REGISTRYINDEX);

lua\_pop(m\_state, 1);

lua\_rawgeti(m\_state, LUA\_REGISTRYINDEX, r\_func);

}

// is it a function

if ( !lua\_isfunction(m\_state,-1) )

{

lua\_settop( m\_state, 0 );

std::string msg = "(CCLuaScriptModule): "+strFunc +" name does not represent a Lua function"+"\n";

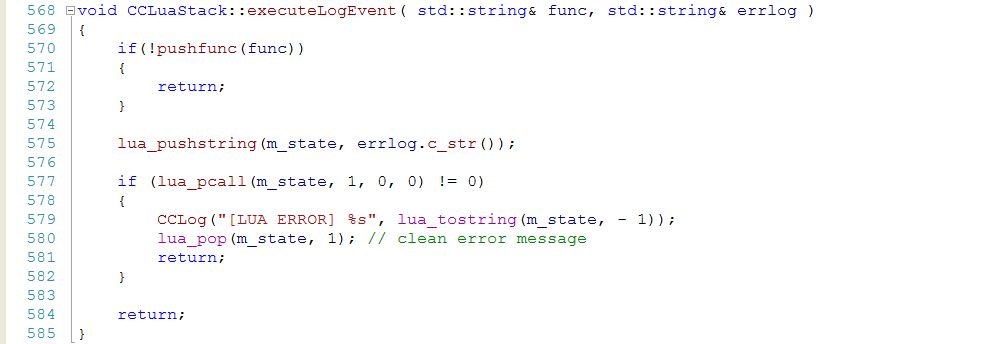
CCLog("%s %d", msg.c\_str(), \_\_LINE\_\_);

return false;

}

return true;

}



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, "l", &debug\_info);

lua\_getinfo(L, "n", &debug\_info);

lua\_getinfo(L, "S", &debug\_info);

err += '\n';

err += '[';

err += debug\_info.what;

err += "][";

err += debug\_info.namewhat;

err += "][";

//err += \_itoa(debug\_info.currentline, tmp, 10);

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);

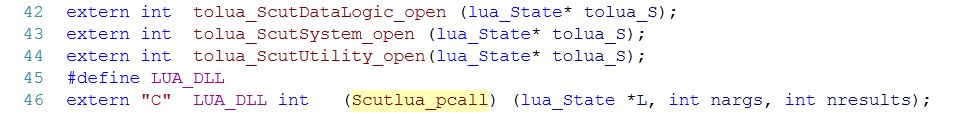
lua\_insert(L, -2-nargs);

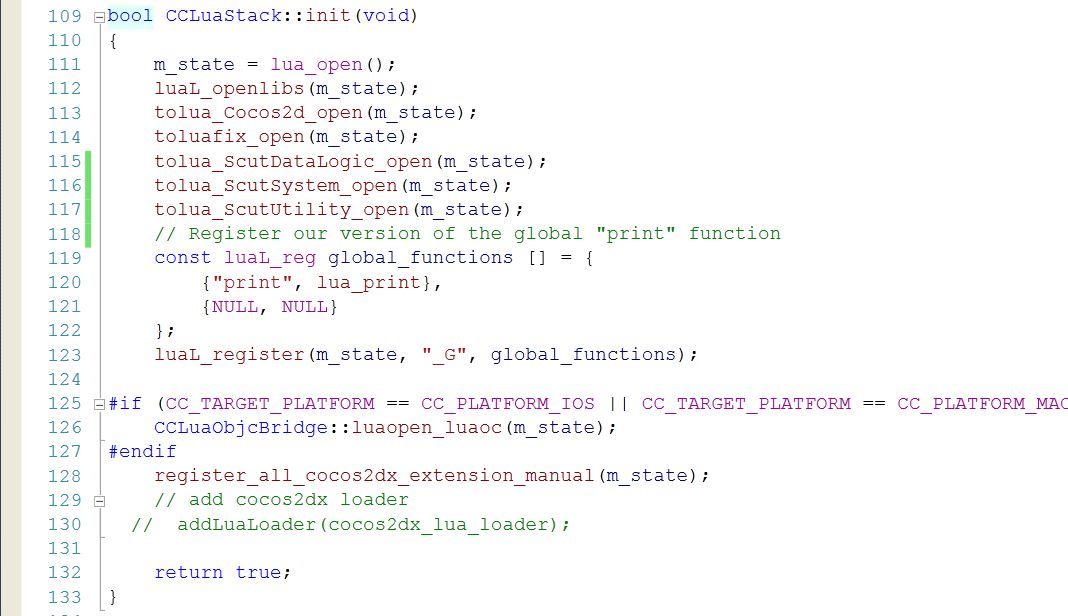
return lua\_pcall(L, nargs, nresults, -2-nargs);

}

这些是打印LUA LOG所使用的方法。

g. 在CCLuaStack.cpp 中添加如下代码





这些方法是为了lua能够调用ScutSDK的C++方法。

# ScutSDK之编译篇