ScutSDK使用文档

**ScutSDK使用说明**

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [ ] 正式发布  [ ] 正在修改 | 文件标识： |  |
| 当前版本： | V1.0 |
| 作 者： | 陈雄 |
| 完成日期： |  |

**目 录**

[ScutSDK使用文档 1](#_Toc375311801)

[1. 什么是ScutSDK 3](#_Toc375311802)

[2. Lua API讲解 3](#_Toc375311803)

[2.1 使用准备 3](#_Toc375311804)

[2.2 类与接口说明 3](#_Toc375311805)

[2.3 HTTP请求使用举例（排行榜） 4](#_Toc375311806)

## 1. 什么是ScutSDK

Scut开源游戏服务器引擎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是和Scut开源游戏服务器引擎，简化客户端开发的配套SDK，她彻底打通了Scut开源游戏服务器引擎与客户端引擎（如Cocos2d-x/Quick-x/Unity3D）项目间的通信，进而实现整套的网络游戏解决方案。

## 2. Lua API讲解

### 2.1 使用准备

使用Lua API必须要掌握Lua语言编程。同时需要熟悉ScutSDK的Lua使用接口。

### 2.2 类与接口说明

#### 2.2.1 ScutScene接口

1.new：构建ScutScene对象。

2.registerCallback：注册网络请求回调函数。

3.registerNetErrorFunc：注册网络错误回调函数。

4. registerNetCommonDataFunc：注册通用解析函数。

5.registerNetDecodeEnd：注册解码结束回调函数。

6. execCallback：执行网络回调函数。

#### 2.2.2 FrameManager接口

1.new：构建FrameManager对象

2.init：初始化FrameManager对象

3.update：每帧回调

#### 2.2.3 NetHelper接口

1．初始化接口（在lib/nethelper.lua中已经初始化）：

|  |
| --- |
| ZyReader = ScutDataLogic.CNetReader:getInstance() |

ZyReader为获取C++中的CNetReader对象实例的Lua封装。该对象提供了解析Scut二进制流必须的方法，为ScutSDK核心组成部分。

2．recordBegin()：结构体中，开始解析二进制流

3．recordEnd()：结构体闭合，结束解析二进制流

4．getBYTE()：解析unsigned char（byte，1字节）方法。

5．getWORD()：解析unsigned short（WORD，2字节）方法。

6．getDWORD()：解析unsigned int（DWORD，4字节）方法。

7．getFloat()：解析float方法。

8．getDouble()：解析double方法。

9．readString()：解析String方法。

10．readInt64()：解析Int64方法。

11．getInt()：解析Int32方法。

12．readErrorMsg()：解析错误消息。该错误消息需要与服务端约定。当网络请求发生异常，或者业务逻辑出现错误时候，打印的消息。

13．getResult()：获取当前网络请求状态，服务端返回错误码，该错误码需要与客户端约定。

14. getActionID ()：获取当前请求的ActionID。

15．ZyExecRequest（pScutScene, lpData, bShowLoading,type）

参数pScutScene 为请求接口发起的scene对象，此对象为CCScene类型。

参数lpData为用户附加数据，该数据将被直接传回。

参数bShowLoading，是否显示loading。

参数type请求的类型。传nil为socket请求，非nil为http请求。

### 2.3 HTTP请求使用举例（排行榜）

在LUA程序的入口模块，（我们的实例为mainapp.lua）将ScutSDK必须的两个lua文件引入进来。如下图所示：



实例化一个FrameManager对象，并初始化。FrameManager对象初始化时，将会注册一个每帧回调的函数。

|  |
| --- |
| g\_frame\_mgr = FrameManager:new()  g\_frame\_mgr:init() |

在全局实现一个回调函数，当网络请求返回时，该函数将会调用，并把数据派发给绑定的scene回调函数。

|  |
| --- |
| function OnHandleData(pScene, nTag, nNetRet, pData)  pScene = tolua.cast(pScene, "CCScene")  *g\_scenes[pScene]:execCallback(nTag, nNetRet, pData)*  end |

接下来我们要设置一下请求的URL地址的头，之后发出的请求会基于此头部自动构建请求的完整URL。

|  |
| --- |
| ScutDataLogic.CNetWriter:setUrl("http://ph.scutgame.com/service.aspx") |

好了，完成这两件事情之后我们创建一个场景，来实验一下网络请求。

|  |
| --- |
| function init()  if mScene then  return  end  local scene = ScutScene:new()  mScene = scene.root  scene:registerCallback(netCallback)  CCDirector:sharedDirector():pushScene(mScene)  pWinSize = mScene:getContentSize()    mLayer = CCLayer:create()  mLayer:setAnchorPoint(CCPoint(0,0))  mLayer:setPosition(CCPoint(0,0))  mScene:addChild(mLayer, 0)    mRankLayer = CCLayer:create();  mRankLayer:setAnchorPoint(PT(0.5, 0.5));  mRankLayer:setPosition(PT(pWinSize.width/2, pWinSize.height/2));  CCDirector:sharedDirector():pushScene(mScene)  local bgSprite=CCSprite:create(P("beijing.jpg"))  bgSprite:setScaleX(pWinSize.width/bgSprite:getContentSize().width)  bgSprite:setScaleY(pWinSize.height/bgSprite:getContentSize().height)  bgSprite:setAnchorPoint(CCPoint(0.5,0.5))  bgSprite:setPosition(CCPoint(pWinSize.width/2,pWinSize.height/2));  mScene:addChild(bgSprite);  ScutDataLogic.CNetWriter:setUrl("http://ph.scutgame.com/service.aspx")  local button = ZyButton:new("icon\_1011.png");  button:addto(mScene,0);  button:setPosition(PT(pWinSize.width - button:getContentSize().width - SX(10), SY(10)));  button:registerScriptTapHandler(showRank)    local button2 = ZyButton:new("button/button\_1011.png", "button/button\_1012.png",nil,commonString.IDS\_SUBMIT)  button2:setPosition(PT(pWinSize.width/2 - button2:getContentSize().width/2 ,SY(10)));  button2:addto(mScene,0)  button2:registerScriptTapHandler(submit);  end |

该段代码中，我们创建了一个scutScene对象，并绑定了一个网络回调函数scene:registerCallback(netCallback)。



让我们点击一下获取排行版按钮。该操作代码如下：

|  |
| --- |
| function showRank()  if isCanGetRank == false then  return  end  ScutDataLogic.CNetWriter:getInstance():writeString("ActionId",1001)  ScutDataLogic.CNetWriter:getInstance():writeString("PageIndex",1 )  ScutDataLogic.CNetWriter:getInstance():writeString("PageSize",30)  ZyExecRequest(mScene, nil,false,1)  end |

此段代码为请求排行版数据，将参数填写好后，ZyExecRequest将会组建请求的URL。请求发送后，我们将会在刚才注册的netCallback回调中收到数据。

|  |
| --- |
| function netCallback(pZyScene, lpExternalData)  local actionID = ZyReader:getActionID()  local lpExternalData = lpExternalData or 0  local userData = ZyRequestParam:getParamData(lpExternalData)  if actionID==1001 then  local table = \_1001Callback(pZyScene, lpExternalData);  if table then  bgLayer= UIHelper.createUIBg(nil,nil,ccc3(255,255,255),nil,true)  mScene:addChild(bgLayer)  local closeBtn=ZyButton:new(Image.image\_exit, Image.image\_exit);  closeBtn:setPosition(PT(bgLayer:getContentSize().width-closeBtn:getContentSize().width - SX(15),bgLayer:getContentSize().height-closeBtn:getContentSize().height - SY(5)));  closeBtn:registerScriptTapHandler(closeBtnActon);  closeBtn:addto(bgLayer,99);  showLayout()  update(table.RecordTabel)  end  elseif actionID == 1000 then  \_1000Callback(pZyScene, lpExternalData);  end  end |

|  |
| --- |
| function \_1001Callback(pZyScene, lpExternalData)  local DataTabel=nil  if ZyReader:getResult() == 0 then  DataTabel={}  DataTabel.PageCount= ZyReader:getInt()  local RecordNums\_1=ZyReader:getInt()  local RecordTabel\_1={}  if RecordNums\_1~=0 then  for k=1,RecordNums\_1 do  local mRecordTabel\_1={}  ZyReader:recordBegin()  mRecordTabel\_1.UserName= ZyReader:readString()  mRecordTabel\_1.Score= ZyReader:getInt()  ZyReader:recordEnd()  table.insert(RecordTabel\_1,mRecordTabel\_1)  end  end  DataTabel.RecordTabel = RecordTabel\_1;  else  local box = ZyMessageBoxEx:new()  box:doPrompt(pZyScene, nil, ZyReader:readErrorMsg(),commonString.IDS\_OK)  end  return DataTabel  end |

function\_1001Callback(pZyScene, lpExternalData) 解析函数将会由服务端的协议生成器自动生成。我们收到数据后，就可以用此数据更新我们的排行版界面了。

