1.

cocos2dx--两个场景切换各函数调用顺序

场景A切换到场景B，有切换特效

调用顺序如下：（AAABABABA）

A：构造函数

A：onEnter

A：onEnterTransitionDidFinish

B：构造函数

A：onExitTransitionDidStart

B：onEnter

A：onExit

B：onEnterTransitionDidFinish

A：析构函数

场景A切换到场景B，无切换特效

调用顺序如下：（AAABAABBA）

A：构造函数

A：onEnter

A：onEnterTransitionDidFinish

B：构造函数

A：onExitTransitionDidStart

A：onExit

B：onEnter

B：onEnterTransitionDidFinish

A：析构函数

场景A中pushScene B场景，有切换特效（AAABABAB）

调用顺序如下：

A：构造函数

A：onEnter

A：onEnterTransitionDidFinish

B：构造函数

A：onExitTransitionDidStart

B：onEnter

A：onExit

B：onEnterTransitionDidFinish

场景A中pushScene B场景，无切换特效（AAABAABB）

调用顺序如下：

A：构造函数

A：onEnter

A：onEnterTransitionDidFinish

B：构造函数

A：onExitTransitionDidStart

A：onExit

B：onEnter

B：onEnterTransitionDidFinish

场景B中popScene

调用顺序如下：

B：onExitTransitionDidStart

B：onExit

A：onEnter

A：onEnterTransitionDidFinish

2.

具体看如下分析和代码：

启动异步处理工作（下面两句代码需要在主线程里执行）：

UpdateTask task = new UpdateTask();

task.execute(“liming”,“zhangxx”); //这里输入的参数会传给doInBackground()方法

停止异步处理工作：

if (task != null && task.getStatus() == AsyncTask.Status.RUNNING) {

task.cancel(true);//如果Task还在运行，则先取消它

task = null;

}

java代码

public class UpdateTask extends AsyncTask<String, Integer, String> {

protected void onPreExecute() {

progressBar.setMax(100);

}

protected String doInBackground(String... params) {

for (int i = 0; i < 100; i++) {

this.publishProgress(i + 1);

try {

Thread.sleep(100);

} catch (InterruptedException e) {

e.printStackTrace();

}

if (isCancelled())

return null;

}

return "itcast";

}

protected void onPostExecute(String result) {

Toast.makeText(getApplicationContext(), "结果 ：" + result, 1).show();

}

protected void onProgressUpdate(Integer... values) {

progressBar.setProgress(values[0]);

}

}

AsyncTask类的三个泛型参数：

第一个指定了doInBackground() 方法输入参数的类型；

第二个指定了onProgressUpdate()方法输入参数的类型；

第三个指定了onPostExecute()方法输入参数的类型和doInBackground() 方法返回值的类型

这三个泛型参数的类型是根据你自己的业务需求来指定的。

onPreExecute()： 该方法在UI线程运行，当AsyncTask的execute()方法执行后，onPreExecute()会首先执行。可以在该方法中做一些准备工作，如初始化进度条的最大值。

doInBackground(Params...)： 将在onPreExecute()方法执行后马上执行，该方法运行在子线程中，负责执行耗时操作。在执行耗时操作的过程中你可以不断地调用publishProgress()方法，导致onProgressUpdate()不断地被调用。

onProgressUpdate(Progress...)，调用publishProgress()方法，就会导致该方法被执行，该方法运行在UI线程，例如在该方法中你可以更新进度条的显示。

onPostExecute(Result)：doInBackground()方法执行后的返回结果会传给该方法，该方法运行在UI线程，在该方法中你可以显示处理结果。

3.

C++调用Java

#if (CC\_TARGET\_PLATFORM == CC\_PLATFORM\_ANDROID)

JniMethodInfo minfo;//定义Jni函数信息结构体

CCLOG("iTag=====%d----PRO\_ID[iTag]--=%d",iTag,PRO\_ID[iTag]);

//getStaticMethodInfo 次函数返回一个bool值表示是否找到此函数

bool isHave = JniHelper::getStaticMethodInfo(minfo,

"cn/easymobi/leidian/Leidian",

"doJifei", "(ILjava/lang/String;)V");

//第一个参数是；第二个参数是在安卓方面的目录；第三个是安卓中的方法名字；第四个参数是根据不同的返回值类型来写

jstring rtstr = minfo.env->NewStringUTF(sdata);

if (!isHave) {

} else {

//调用此函数

minfo.env->CallStaticVoidMethod(minfo.classID,

minfo.methodID, PRO\_ID[iTag],rtstr);

//后两个是传的参数

CCDirector::sharedDirector()->pause();

}

minfo.env->DeleteLocalRef(rtstr);

minfo.env->DeleteLocalRef(minfo.classID);

#endif

Java回调C++

extern "C"

//纯c的方法 在安卓方面直接声明方法名字如：private native static void jnicallback(int iCallback); 直接用不用实现

{

void Java\_cn\_easymobi\_leidian\_Leidian\_jnicallback(JNIEnv\* env, jobject firedragonpzy,jint x)

{

iApp->iAndroidCallBak = x;

UserDefault::saveBoolCf(KEY\_BUY\_SUCCESD, true);

CCDirector::sharedDirector()->resume();

}

}

#endif

有关C++调用java 第四个参数如何传

创建一个.java文件（与你要调用的java方法的返回值类型一致）

然后在终端进行编译这个.java文件（跳的这个文件的目录，在终端输入javac 名字.java）

然后在终端输入javap -s 名字 文件

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_UNKNOWN

// mac

#if defined(CC\_TARGET\_OS\_MAC)

#undef CC\_TARGET\_PLATFORM

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_MAC

#endif

// iphone

#if defined(CC\_TARGET\_OS\_IPHONE)

#undef CC\_TARGET\_PLATFORM

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_IOS

#endif

// android

#if defined(ANDROID)

#undef CC\_TARGET\_PLATFORM

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_ANDROID

#endif

// WinRT (Windows Store App)

#if defined(WINRT) && defined(\_WINRT)

#undef CC\_TARGET\_PLATFORM

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_WINRT

#endif

// WP8 (Windows Phone 8 App)

#if defined(WP8) && defined(\_WP8)

#undef CC\_TARGET\_PLATFORM

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_WP8

#endif

// win32

#if defined(WIN32) && defined(\_WINDOWS)

#undef CC\_TARGET\_PLATFORM

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_WIN32

#endif

// linux

#if defined(LINUX)

#undef CC\_TARGET\_PLATFORM

#define CC\_TARGET\_PLATFORM CC\_PLATFORM\_LINUX

#endif