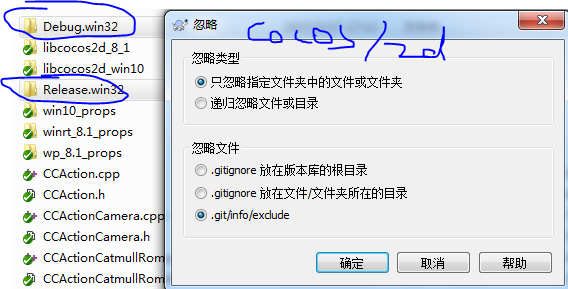
项目制作之随笔

cocos 下载 <https://github.com/fusijie/Cocos-Resource>



目录

[1、Lua的Debug调试输出 2](#_Toc513369147)

[2、日志颜色区别输出 2](#_Toc513369148)

[3、调试的debug控制台 4](#_Toc513369149)

[4、cocos2d-lua 3.5以后的全局变量声明 4](#_Toc513369150)

[5、CocosStudio导入plist资源 5](#_Toc513369151)

[6、cocos2d-Lua继承关系 6](#_Toc513369152)

[7、项目加载进来后会提示缺少cocos2dx包 7](#_Toc513369153)

[8、Microsoft Visual Studio正忙解决办法 8](#_Toc513369154)

[9、打包Apk配置开发环境 8](#_Toc513369155)

[10、批量修改文件名称是rename命令 11](#_Toc513369156)

[11、工程中classpath中指向的包路径错误 12](#_Toc513369157)

[12、修改proj.android/jni文件夹下的Android.mk文件 12](#_Toc513369158)

[13、cocos2dx解决iconv库的移植问题 13](#_Toc513369159)

[14、cocos2dx解决iconv库的移植问题 14](#_Toc513369160)

[15、Ant自动编译打包 16](#_Toc513369161)

[16、cocos2d compile 使用ant编译打包报错 16](#_Toc513369162)

[17、android黑屏 17](#_Toc513369163)

[18、使用ScriptHandlerMgr注册自定义时间 18](#_Toc513369164)

[19、使用cocos2d-x和libvlc实现一个视频播放器 19](#_Toc513369165)

[20、cocos2d-x 3 lua回调自定义函数的方法 20](#_Toc513369166)

[21、在android中添加一个应用层 23](#_Toc513369167)

[22、android目录下的res目录说明 24](#_Toc513369168)

[23、R cannot be resolved to a variable问题的解决方法 25](#_Toc513369169)

[24、lua的require和module 25](#_Toc513369170)

[25、visual studio 开启debug模式 26](#_Toc513369171)

[26、背景图的一种适配方式 27](#_Toc513369172)

[27、透明度变化 27](#_Toc513369173)

[28、Cocos 图片变灰 倒计时进度条 27](#_Toc513369174)

[29、ListView|ScrollView|TableView|PageView的区别 27](#_Toc513369175)

[30、cocos2dx 自带的openURL方法 28](#_Toc513369176)

[31、解决scrollview上的menu拖动问题以及menu item在可视区外仍能触发的问题 29](#_Toc513369177)

[32、tableView的reloadData刷新数据后怎么样停留在当前位置 31](#_Toc513369178)

[33、Lua实现计算 UTF8 字符串的长度，每一个中文算一个字符 32](#_Toc513369179)

[34、-1会因不同编译环境而不同，在Win32上转换为无符号255，而安卓为0 32](#_Toc513369180)

[35、error LNK2001: 无法解析的外部符号 "\_\_declspec(dllimport) 32](#_Toc513369181)

[36、使用cocos compile -p android时总是报错误 BUILD FAILED \ant\build.xml:892 34](#_Toc513369182)

[37、贝赛尔曲线 35](#_Toc513369183)

[38、Cocos2d-x Lua引擎制作的游戏代码加密 35](#_Toc513369184)

## 1、Lua的Debug调试输出





使用xpcall可以解决这个问题, 比pcall多了一个参数。使用debug.traceback可以将traceback的信息存储到msg变量。

xpcall 接受两个参数：调用函数、错误处理函数。当错误发生时，Lua会在栈释放以前调用错误处理函数，因此可以使用debug库收集错误相关信息。

常用的debug处理函数：debug.debug和debug.traceback，前者给出Lua的提示符，你可以自己动手察看错误发生时的情况；后者通过traceback创建更多的错误信息，也是控制台解释器用来构建错误信息的函数。

你可以在任何时候调用debug.traceback获取当前运行的traceback信息。

cocos.init文件根据配置信息初始化Cocos2d-lua框架（require加载lua模块时必须使用”.”来代替路径中的”/”符号）

根据项目需求，可以在cocos.init文件中注释掉不使用的模块，比如physics3d、spine、cocosbuilder等。

## 2、日志颜色区别输出

引入LoggerHelper.h 文件，并放于Win-Class中，在AppDelegate.cpp中引用. 之后，便可在lua文件中引用logger\_info("logger\_info")、logger\_debug("logger\_debug")、logger\_warning("logger\_warning")、logger\_error("logger\_error")、logger\_fatal("logger\_fatal")等带有颜色的log输出。

由于只在Win平台下引入了颜色日志，故在Lua中需要对log进行封装，使得在不同平台调用不同方法。





## 3、调试的debug控制台



## 4、cocos2d-lua 3.5以后的全局变量声明



Cocos2d-x 3.5的lua项目生成后，变成了MVC模式，并且，加入了一个全局变量的检测功能。也就是说，你不小心用了全局变量，他会提示你出错！

所以，C2d加入了一个全局命名空间的检测，通过setmetatable的方式。该函数在src/[**cocos**](http://lib.csdn.net/base/cocos)/framework/init.lua



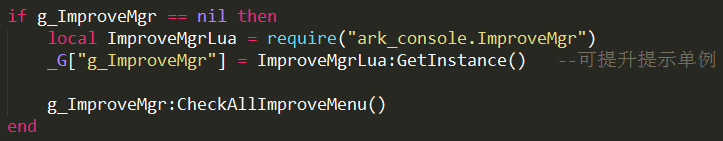
那么如果我一定要用到全局变量才行呢？这个问题有若干种解决办法

1、最简单粗暴的就是直接注释掉着一块。

2、将CC\_DISABLE\_GLOBAL设置为False

3、在mian.lua的  require "cocos.init"之前 require 所需的库和全局的变量

但是如果是开发者自己需要使用全局变量。而不是其他库需要用到全局变量的话！请使用使用官方推荐的方法cc.exports.\*  来充当全局变量！

或者使用\_G[]，比如：

## 5、CocosStudio导入plist资源

在Cocos Studio编辑器中，目前支持的plist文件有两种，一种以TexturePacker为代表导出的图像集合，另一种粒子（Particle）文件。我们的Cocos Studio也支持将碎图生成为plist图片。plist图像文件都是两个文件（.plist/.png）同时存在的，并且文件名称相同，.plist负责图像文件信息存储，.png则存储整合后的图像。plist粒子文件则可能由一个或两个文件组成（.png是非必须存在的，当图片过小时图像将被base64转换成字符串存在.plist文件中）。

另一种图形文件PSD，它是Adobe公司的图形设计软件Photoshop的专用格式。同一个文件中也包含多个图层信息，也是美术使用最重要的格式。

CocoStudio为了方便开发者，对这些格式都提供了支持。但用户如果想顺利使用这些功能需要注意一些地方：

1：UI、动作、场景都支持plist图像和PSD图像。Plist粒子文件只有场景编辑器和动作编辑器可以使用。

2：Plist文件导入时必须将对应的.PNG图像一同导入到资源区。如果是直接放到项目的Resource文件夹内，可以通过刷新资源区更新数据。

3：Plist图像和PSD图像导入到资源区时都是以一个文件夹的形式显示（分别为“PL”和“PS”图标），文件夹内包含所有的子图。CocoStudio不提供对子图的编辑功能，你也不可以单独删除一个子图。

4：PSD图像的所有图层必须要做栅格化处理，未经栅格化的图像将不能被识别。

5：导出项目功能，不管你选择导出大图模式或导出小图模式，Plist图像将不会被再次编辑，而是直接导出。（比如选择导出小图模式，plist图像还是会直接导出的。）

6：导出项目时PSD则将所有使用的图层当做小图处理。如果导出大图将把需要的图层整合进plist图像中。

7：Cocos Studio支持将Plist、PSD图形导出为碎图。（步骤：在资源区选中Plist图像，右键菜单选择“导出为碎图”选项。）

## 6、cocos2d-Lua继承关系



如果子类GameLayer重新定义了派生类，则执行结果如左。如果子类GameLayer没有重定义派生类，执行结果如右。注意，如果直接调用.new()或:new()方法则会直接调用ctor()而不再调用create()。



## 7、项目加载进来后会提示缺少cocos2dx包

表现为创建的Android项目提示org.cocos2dx.lib.Cocos2dxActivity找不到问题

解决方法（最直接方便）：

复制 \*\*\*\cocos2dx\platform\android\java\src\ (\*\*\*为cocos2d-x更目录) 到你android项目中的src文件夹中。

方法一、那我们将设置一下cocos2d-x的引用即可

右键项目->build path->link source->Browse->选择地址

F:\Android\cocos2d\cocos2d-2.1rc0-x-2.1.2\cocos2dx\platform\android\Java\src

Folder name 填写为一个不与src冲突的名称，如cocos2dx-src 即可，然后确定

方法二、也可以Eclipse可以导入libcocos2dx

1.右键Import项目，选择libcocos2dx所在cocos2d引擎下路径，“\cocos2d\cocos\platform\Android\Java”

2.右键->"Properties"->"Android"->"Library"->"Add"会自动显示刚才添加的lib，添加即可。

另，编译时总是提示couldn't find import for class com.enhance.gameservice.

IGameTuningService；似乎是找不到IGameTuningService类，而实际上我们都知道com.enhance.gameservice.aidl确实是存在的。此时需要在Eclipse中打开相应aidl文件，重新保存即可。

另，import com.loopj.android.XXXX报错

android-async-http开源框架可以是我们轻松的获取网络数据或者向服务器发送数据，使用起来也很简单，下面做简单介绍，具体详细使用看官网：https://github.com/loopj/android-async-http

1.新建项目，去官网下载zip包，解压，打开releases文件，把里面最新的jar包，考入项目工程libs目录下，引入包。

2.通过1，就可以使用了，很简单，下面是自己写的demo，用它提供的各种不同方法完成从服务器获取一个json数据。

另，关于AsyncHttpClient的cz.msebera.android.XXXX

从这个网站上下载那个JAR包就可导进去，报错的地方就解决了

http://mvnrepository.com/artifact/cz.msebera.android/httpclient/4.3.6

或者使用**import** org.apache.http.Header;

**import** org.apache.http.message.BasicHeader;

代替

**import** cz.msebera.android.httpclient.Header;

**import** cz.msebera.android.httpclient.message.BasicHeader;

HttpClient是Apache Jakarta Common下的子项目，用来提供高效的、最新的、功能丰富的支持HTTP协议的客户端编程工具包，并且它支持HTTP协议最新的版本和建议。

## 8、Microsoft Visual Studio正忙解决办法

关闭vs进程重启vs还是无法解决。

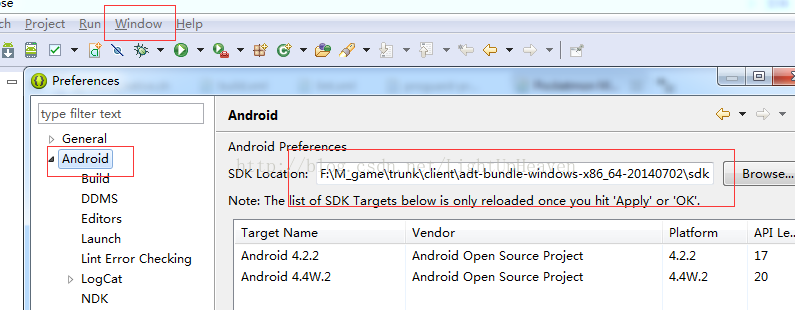


如何解决：进入本地项目文件夹。这个文件夹下有个.suo文件，删除该文件，用任务管理器杀掉vs的进程，重新打开vs即可。

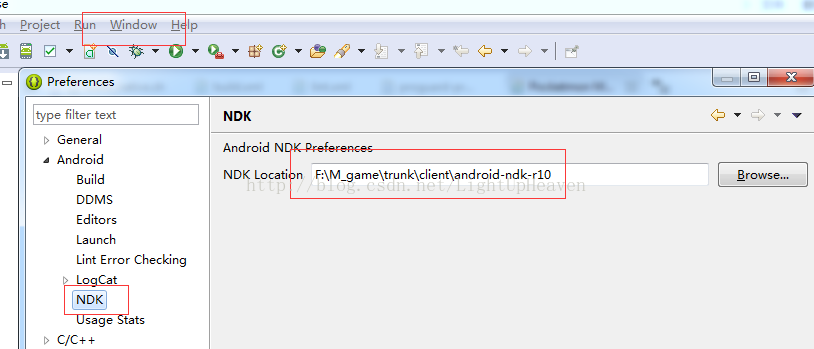
## 9、打包Apk配置开发环境

https://my.oschina.net/Jacedy/blog/350226

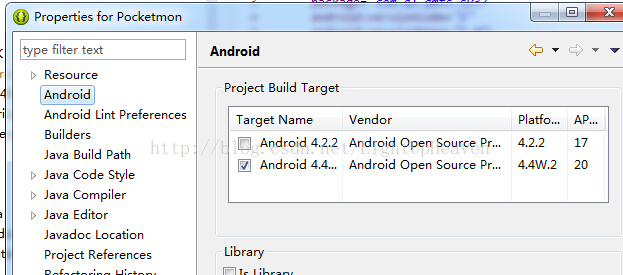
<http://blog.csdn.net/lightupheaven/article/details/42267145>



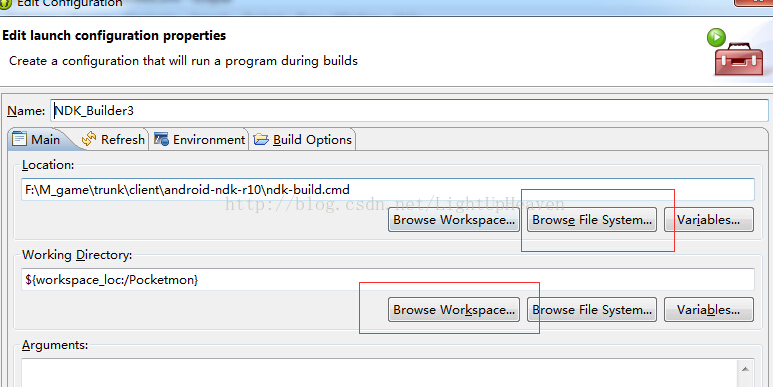
为什么要下载NDK，下载它干嘛？这个NDK就是Java和C++通过这个东西可以有联系，不再是毫不相关的两个玩意。

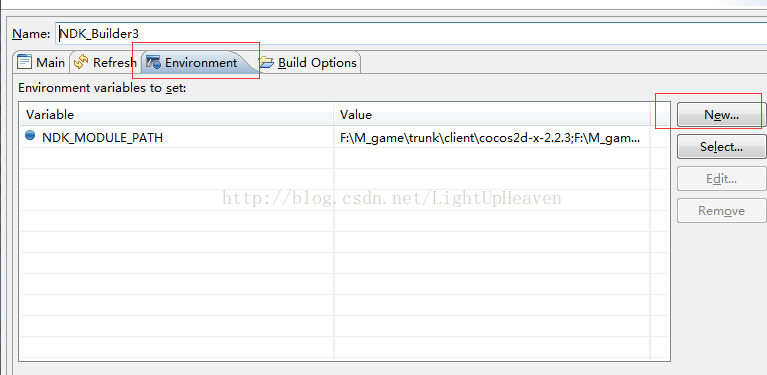


SDK的配置决定了你的工程最终会生成哪个版本的Android API 代码。



新建NDKBUilder。名字随便取





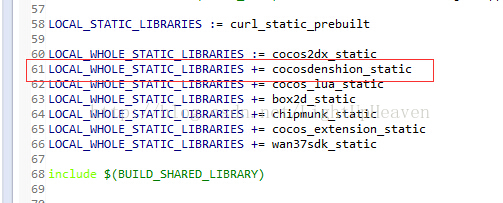
注意，其上的NDK Location和NDK\_MODULE\_PATH一定要配置自己的本地路径。比如：

Location：D:\Android\android-ndk-r10e\ndk-build.cmd， NDK\_MODULE\_PATH = D:\NewFlowerGame\new\_flower\_client\trunk\frameworks\cocos2d-x\cocos;D:\NewFlowerGame\new\_flower\_client\trunk\frameworks\cocos2d-x;D:\NewFlowerGame\new\_flower\_client\trunk\frameworks\cocos2d-x\external

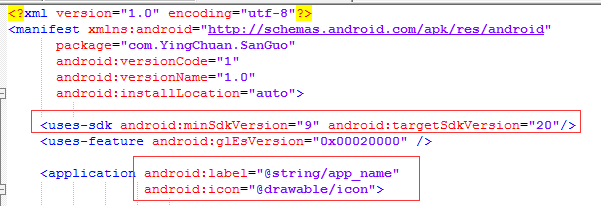
至于为什么这里要这么设置，而不是上边一级或者下面一级目录呢？其实 这跟Android .mk文件有关，其实，这些目录都是作为 NDK的根目录存在的，就是你可以以这个目录为准，就好像你以C盘为准一样，在Android.mk这个文件里你可以加载静态库，可以加载Cpp源文件，所以如果你要使用这两者，并且路径很长，你就可以在 NDK\_MODULE\_PATH里加入路径。然后再Android.mk文件里，根据你加入的这些任意路径为基础，填写相对路径就能找到对应的比如



这些都是有Android.mk文件存在的文件夹，加载的是静态库



AndroidManifest.xml中设置你的应用最低和目标Android版本，设置了最低的在低于这个版本的Android机上可能就运行部了，而在高于你设置的android:targetSdkVersion和android:minSdkVersion的机器上，你的应用都可以运行。



icon和label一个设置你的游戏的手机桌面图标，一个是名字。这里 Application和Activity都设置了，而实际上Activity的的更快执行。这两处设置一个就可以了。

最后 记得把你的游戏资源 放到eclipse工程的assert文件夹。

（注意：在打包apk之前需要修改proj.android/jni文件夹下的Android.mk文件 添加自己新增的CPP文件，这里的新增指的是非cocos2dx生成的项目文件（自己新增的文件）。一定要把自己新增的文件或文件夹包含进apk。）



然后就可以使用 cocos compile -p android –-ap 20方式编译APK文件了（--ap 20是指定android平台）。当然这里要提醒一点3.2版本用的是 android-20，如果E:\Android\sdk\platforms目录下没有android-20 请创建一个空的。注意修改项目android目录下和cocos android目录下的project.properties，将android-13修改为target=android-20。

如果需要把项目在真机上弄成竖屏，只需在proj.android文件夹下的AndroidManifest.xml中改一改， android:screenOrientation="landscape"（横屏），android:screenOrientation ="portrait"（竖屏），它默认为横屏。

## 10、批量修改文件名称是rename命令

用法是：ren(空格)驱动盘:路径\旧文件名（空格）新文件名 比如将D:\迅雷下载\海贼王文件夹下的.net为后缀名的文件改成以.mp4为后缀名的文件 示例： ren D:\迅雷下载\海贼王\\*.net \*.mp4

## 11、工程中classpath中指向的包路径错误

当import一个新的Android项目时，项目名称上出现一个红色的感叹号，程序不能运行。

原因：工程中classpath中指向的包路径错误。

解决方法：在项目名称上点右键选择Properties，然后选择Java Bulid Path中的libraries选项，其中出现红色x号的路径为错误路径，删除之后，通过Add External Jars添加正确路径，点击确定之后，在Package Explorer中刷新一下。

## 12、修改proj.android/jni文件夹下的Android.mk文件

Android.mk文件来向NDK编译系统描述你的源代码

Application.mk文件向NDK编译系统描述你的应用程序需要的原生代码。

Application.mk用来描述你的工程下的native模块, 一般放在$PROJECT/jni/Application.mk, 其中$PROJECT表示你的工程目录，这样就可以被ndk-build脚本文件找到。(注：在这种方式下，进入jni目录，即$PROJECT/jni/,然后执行ndk-build，就可以直接编译jni生成.so文件了)。

还有另外一种方法就是将Application.mk文件放在$NDK/app/<myapp>/Application.mk，其中$NDK为NDK的安装目录，<myapp>为你的应用程序名。在这种方式下，进入$NDK安装目录，然后输入make APP=<name>，即可编译你的JNI代码.此种方法是ndk-r4之前的方法，虽然出于兼容的原因目录还支持，但是不建议使用此种方法，因为第一种方法简单，且方便。

APP\_PROJECT\_PATH（可选）绝对路径.

此变量值必须是你工程根目录的绝对路径。这用于指定JNI生成的.so文件安装路径或拷贝路径。注：此变量对于第一种方法是可选了，但对于第二种方法却是必须的。

APP\_MODULES（可选） 模块名

此变量是可选的，如果没有在此文件中定义，则默认由Android.mk文件决定。如果在Application.mk文件中定义此变量，那么它必须是一串由空格相隔的模块名列表，并且与Android.mk文件中的LOCAL\_MODULE变量定义保持一致.需要注意地是，NDK会自动计算模块所需要的依赖文件。注：此变量在NDK R4版本之前是在Application.mk中是必须的,所有模块必须显式地列出。

APP\_OPTIM（可选）优化选项

此变量是可选的，其值可以为'release'或'debug'.此变量用来修改优先等级.默认情况下为release.在release模式下，将编译生成被优化了的二进制的机器码，而debug模块用来生成便于调试的未被优化的二进制机器码。需要注意地是，如果你的程序在manifest文件中的节点下<application>已经设置了android:debuggable属性为"true",则Apllication.mk文件此变量的默认值则为'debug'而不是'release',可以在Application.mk文件中将此变量值设置为'release'而覆盖其值。虽然在两种模块下都可以调试，但是release模式下提供更少的调试信息，比如：一些变量被优化从而不能被检查，代码重组，及跟踪。

APP\_CFLAGS编译选项

可以在Application.mk文件中修改此编译选项参数，此参数可以覆盖Android.mk文件中的相同定义，从而不必去Android.mk文件中修改.

需要注意地是，这个选项里的路径必须是以顶层NDK目录为相对路径，如存在以下两个文件：sources/foo/Android.mk 和 sources/bar/Android.mk

如果在foo/Android.mk文件中想要添加bar目录，那么得这样便用：

APP\_CFLAGS += -Isources/bar

如果这样使用:

APP\_CFLAGS += -I$(LOCAL\_PATH)/../bar

将会变成'-I$NDK\_ROOT/../bar'，从而出现并不是你想要的结果。在android-ndk-1.5\_r1时，此选项中只在C代码中被支持，而C＋＋不支持，不过现在已经被纠正了。

APP\_CXXFLAGS是APP\_CPPFLAGS的别名，将来此变量将会被抛弃。

APP\_CPPFLAGS

C++代码的编译选项。在android-ndk-1.5\_r1版本中，此变量只适用于C++，但是现在可以同时适用于C和C＋＋。

APP\_BUILD\_SCRIPT

在默认情况下，NDK会在jni目录下查找Android.mk文件并使用它，如果你想修改它，那么在此变量中你可以指定一个你自己的脚本来执行，路径还是以工程顶层目录为相对路径。

APP\_ABI

在默认情况下，NDK会使用'armeabi' ABI 来生成二进制机器码，这是基于ARMv5TE的浮点运算CPU，这可以通过使用此变量来选项不同的ABI(Application Binary Interface)。例如:

支持基于armv7 FPU指令集的设备: APP\_ABI := armeabi-v7a

支持IA-32指令集： APP\_ABI := x86

同时支持三种： APP\_ABI := armeabi armeabi-v7a x86

从NDK-r7版本后，同时支持三种还可以这样写： APP\_ABI := all

APP\_STL是C++标准库

在默认情况下，NDK通过Androoid自带的最小化的C++运行库(system/lib/libstdc++.so)来提供标准C++头文件.然而，NDK提供了可供选择的C++实现，你可以通过此变量来选择使用哪个或链接到你的程序。

APP\_STL := stlport\_static --> static STLport library

APP\_STL := stlport\_shared --> shared STLport library

APP\_STL := system --> default C++ runtime library

## 13、cocos2dx解决iconv库的移植问题

cocos2d-x显示中文乱码最常用的解决方式主要有xml文件读取、使用iconv库。由于在使用iconv库出现了比较多的问题，这里作个总结，给后来人少走一些弯路。

网上有一种方法要自己下载iconv库，我觉得那种方法挺麻烦的，而且我自己跟着步骤试了下，没有成功，后面放弃了，最后又找了一种方法，也就是下面要介绍的用NDK自带的iconv库。我用的是android-ndk-r10e，可以发现有个iconv.h文件，可以cocos2d-x端代码里引用的路径。这个根据自己ndk存放的路径做调整。



然后修改proj.android/jni目录下的Android.mk文件。主要加了

LOCAL\_STATIC\_LIBRARIES += android\_support 和 $(call import-module, android/support)



做了以上这些步骤，正常情况下是可以通过了。

注意以下几点：

1、android编译碰到如下错误invalid conversion from 'const char\*\*' to 'char\*\*'

这是由于我们定义GBKToUTF8这个函数时，参数使用const char\* strChar,然后传入iconv函数时ndk的iconv.h定义的是size\_t iconv(iconv\_t, char\*\*, size\_t\*, char\*\*, size\_t\*); 所以只需要把const去掉就ok了。或者使用const\_cast<>强制去掉常量属性。



2、网上有介绍说ndk r9的iconv.h代码有问题，需要修改如下，大家注意一下就行，r10是没有这个问题。



## 14、cocos2dx解决iconv库的移植问题

http://www.zaojiahua.com/iconv.html

我们在cocos2dx中使用了第三方库的时候，移植到android平台的过程中是会报错的问题，典型的例子就是我在上几篇博客中使用了编码转换的库iconv，在我移植到android平台测试的时候就出现了错误。

多国语言要用到开源字符转换 iconv。注意设置安卓版本的路径和win32版本的路径win32版本是rc1版本里面有的自带iocnv.h头文件和lib。但是Android版本需要自己下载。



在windows上用iconv.dll，在IOS上系统有提供这个库，导入就可以，在android上要自己去下一个。官网 http://ftp.gnu.org/pub/gnu/libiconv/

下载完毕，解压放到E:\MyCocos\cocos2dx-workspace\hello\cocos2d目录下(这里是我自己的目录),改名为iconv,和cocos，extensions，external等同目录。

参考（http://www.2cto.com/kf/201404/295851.html）

在iconv目录下增加Android.mk文件，写入

接下来，在工程\proj.android\jni工程的Android.mk里修改，加入这个lib和include目录（下面的路径要根据项目实际情况修改）



此上两处定义在iconv库的根目录下的Android.mk文件里面定义：



LOCAL\_MODULE定义了库的名称，如果缺少LOCAL\_MODULE\_FILENAME定义，则生成的库会在名前自动添加lib并生成.so动态库或.a静态库（include $(BUILD\_STATIC\_LIBRARY)决定生成静态库）。

LOCAL\_MODULE\_FILENAME则用于显式定义库文件名。

$(call import-module,libiconv)中， libiconv为库在cocos2d-x中的路径文件夹名。

网上很多教程都说到这里，但实际上，如果这个时候进行编译，问题还一大堆。

其一就是iconv.h不存在，在ivonv/include/有 iconv.h.in, iconv.h.build.in，就是没看到iconv.h把iconv.h.in改名成iconv.h，发现编译很多错误主要错误是类型不匹配，还有很多@符号。因此需要进行修改。主要改法是不存在的改成存在的，@符号去掉

除了修改iconv.h文件，还需要改其他2个文件 \iconv\lib\iconv.c和 \iconv\libcharset\lib\localcharset.c

交叉编译时还有一个问题config.h找不到。原因是库里面所有的config.h文件名都是config.h.in，在iocnv下面搜索config.h将3个文件的.in去掉交叉编译成功。（最好不要删除要来的.h.in文件）

## 15、Ant自动编译打包

http://www.cnblogs.com/tt\_mc/p/3891546.html

Eclipse用起来虽然方便，但是编译打包android项目还是比较慢，尤其将应用打包发布到各个渠道时，用Eclipse手动打包各种渠道包就有点不切实际了，这时候我们用到Ant帮我们自动编译打包了。

ant的安装比较简单，下载ant压缩包 http://ant.apache.org (最新的为1.9.7版本)，下载之后将其解压到某个目录(本人解压到E:\Program Files\apache-ant-1.9.7) ，然后配置环境变量(新建ANT\_HOME环境变量，值为ant所在的目录，然后将%ANT\_HOME%/bin添加到path中)。打开命令行工具，输入 ant -version ,如果出现如下结果，说明ant 安装成功。

为Android项目生成Ant配置build.xml

　　打开命令行工具，切换路径到项目所在的目录；输入命令

　　android update project --name <project\_name> --target <target\_ID>--path <path\_to\_your\_project>

项目根目录下多了build.xml，以及local.properties两个文件。

　　其中local.properties写明了我们的android SDK的目录(其实是环境变量ANDROID\_HOME的值，所以如果环境变量中没有这个的，请增加)。

　　build.xml则是ant构建的最重要脚本，打开一看，发现里面其实大部分都是写注释，有用的没几行，这是因为生成的这个build.xml引用了android SDK自带的构建脚本，具体目录是{sdk目录}/tools/ant/build.xml 。

　　这样，项目就支持ant编译打包了

## 16、cocos2d compile 使用ant编译打包报错



查资料发现，发现：cocos compile 的脚本里会自动把“frameworks\js-bindings\cocos2d-x

\cocos\platform\android\java\bin\classes.jar”（2dx的java代码）添加到classes.dex中。

而当用eclipse打开android项目时，默认是没有引入2dx的java类的，此时如果想正常运行android项目，需要引入2dx的jar包，或者引入2dx的java源码。

如 果引入java的源码，当再次执行“cocos compile”时，2dx的java源码生成的class文件，会跟默认引入的“frameworks\js-bindings\cocos2d-x \cocos\platform\android\java\bin\

classes.jar”发生冲突，会报 “com.android.dx.util.DexException: Multiple dex files define ……”的错误；

如 果不引入java源码，而是直接在“Build path”中引入了jar包（位于引擎的：frameworks\js-bindings\cocos2d-x\cocos\platform \android\java\bin\下的classes.jar），此时android的项目不报错了，但是直接使用eclipse生成apk包时会发 现安装到手机上无法运行，这是因为项目需要的2dx文件没有被打入到apk中。此时，可以将对应的jar包放到android项目的libs目录里 （frameworks\runtime-src\proj.android\libs\），这样生成的apk包就可以正常运行了。

但是，如果这样做了，再次执行“cocos compile”时，发现还会报如上的错误，这是因为脚本里会默认将libs目录里的文件添加到classes.dex中。

综上所述：出现如上错误的原因是在使用“cocos compile”脚本时，2dx文件被重复添加了。

解决方法注释掉project.properties文件中的#android.library.reference.1=../cocos2d/cocos

/platform/android/java就可以啦！！！



注意，eclipse编译时，需要将工程项目的res和src文件夹拷贝到android/assets中。

gen中的包和类是自动生成的，里面的R.java里面的内容是所有控件的ID号，当布局有问题的时候，clean一下，gen里面的内容会消失，就是丢失R包的现象搜索，当然，有时候别的原因也会出现这一现象。

Android除了提供/res目录存放资源文件外，在/assets目录也可以存放资源文件，而且/assets目录下的资源文件不会在R.java自动生成ID，所以读取/assets目录下的文件必须指定文件的路径。android工程中assets中文件在打包生成apk的时候不会被编译，以文件原有的方式来保存，可以通过AssetManager来操作这些文件。

bin文件里面有一个apk文件，可以直接用来安装android项目。

res是存放图片之类的资源的，里面的文件夹分别表示方的内容：values里面是放xml文件的，一般是String.xml，color.xml等属性内容；drawable是放图片文件的；layout是放布局文件的。

## 17、android黑屏

安装到android手机里,点击HOME键，然后再回到游戏中，发生黑屏。

有可能是 android:lanuchMode 问题

在多Activity开发中，有可能是自己应用间的activity 跳转，或者夹带其他应用的可复用activity。可能会希望跳转到原来某个activity实例，而非产生多个重复的activity。我们可借助 activity 四种启动模式来实现不同的需求：

standard 默认模式 --------- 来了intent，每次都创建新的实例。

singleTop -------- 来了intent, 每次都创建新的实例，仅一个例外：当栈顶的activity 恰恰就是该 activity的实例（即需要创建的实例)时，不再创建新实例。这解决了栈顶复用问题，想一想，你按两次back键，退出的都是同一个activity，这感觉肯定不爽。

singleTask ---------- 来了intent后，检查栈中是否存在该activity的实例，如果存在就把intent发送给它，否则就创建一个新的该activity的实例，放入 一个新的task栈的栈底。肯定位于一个task的栈底，而且栈中只能有它一个该activity实例，但允许其他activity加入该栈。解决了在一 个task中共享一个activity。

singleInstance ----------- 肯定位于一个task的栈底, 并且是该栈唯一的activity。解决了多个task共享一个activity。

解决方法：找到AndroidManifest.xml文件， 打开，添加了lanuchMode

<activity 中间部分略 android:launchMode="singleInstance">



这一次，打包，安装到手机中， 再次点击 “打开”，按HOME键返回，没有再出现黑屏的问题。

用来解决返回Android游戏加载资源时黑屏的问题。需要修改两个文件：

1) cocos2dx/cocos/platform/CCPlatformMacros.h将Line88修改为 #define CC\_ENABLE\_CACHE\_TEXTURE\_DATA 0（原来为1， 有人建议不要改，理由是默认从后台切换到前台是会重建纹理 的）



2)cocos2dx/cocos/platform/android/java/src/org/cocos2dx/lib/Cocos2dxGLSurfaceView.java注释掉Line201。（文件中已经注释掉了）



基本原理就是在activity进入后台后，不清除texture缓存，这样在activity回到前台后，不需要重新load资源。在游戏上试了一把，修改之前切回游戏需要4秒左右时间，修改后不用1秒，可谓瞬间完成。

## 18、使用ScriptHandlerMgr注册自定义时间



注意在\src\cocos\cocos2d\Cocos2dConstants中定义cc.Handler.EVENT\_CUSTOM\_COMMON



## 19、使用cocos2d-x和libvlc实现一个视频播放器

开发工具使用Visual Studio 2013，测试运行平台为Windows（网上教程采用cocos2d-x版本2.0.3，VLC版本2.0.5。http://www.cnblogs.com/evan-cai/archive/2013/01/25/2876803.html）（或cocos3.x +vlc, http://blog.csdn.net/llong568/article/details/43484731）本地测试采用cocos2d-x3.14 + vlc 2.0.5

直接下载vlc播放器的安装版本然后在sdk目录中把vlc文件夹拷贝到（C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\include）目录下，库文件和动态链接库文件则放在本项目的Debug.win32和Release.win32目录下。（本测试为了方便tolua导出，直接将vlc文件夹放到工程Class下）。

开发VLC所需的文件的位置：

1. 动态链接库\*.dll：安装目录下的libvlc.dll，libvlccore.dll以及plugins目录下的所有文件。PS：VLC支持非常多的plugin。因此plugins目录的体积确实是非常大的。

2. 静态链接库\*.lib：安装目录/sdk/lib

3. 头文件\*.h：安装目录/sdk/include

新建一个VC工程后，把上述三类文件分别拷贝至新工程目录下，并且配置它们的路径之后，就可以使用libVLC进行开发了。

VLC SDK 获取:到 VLC 官方网站http://www.videolan.org/ 下载最新的VLC安装包；安装完成后， 到安装目录下有一个“sdk”文件夹，里面包含相应的 “include” 和“lib”文件夹，配置到VC工程就可以了。或者从http://download.videolan.org/pub/videolan/vlc/下载libvlc，包含相应的头文件，lib文件，和相关的二进制文件。

运行环境： 经安装目录下的所有文件和编译好的exe放到一起就可以正常运行了，否则会出现一些指针为空的错误。

接下来进入正题：创建自定义文件MoviePlayer.h和MoviePlayer.cpp。参照http://www.cnblogs.com/evan-cai/archive/2013/01/25/2876803.html

release版本程序调用libvlc接口出现异常 ，可以用dumpbin 和lib命令生成的libvlc.lib文件重新构建release程序。

http://blog.csdn.net/raymond\_lu\_rl/article/details/7515900

https://wiki.videolan.org/GenerateLibFromDll

dumpbin /exports "C:\Program Files\VideoLAN\VLC\libvlc.dll" > "C:\Program Files\VideoLAN

\VLC\libvlc.def"

lib /def:"C:\Program Files\VideoLAN\VLC\libvlc.def" /out:"C:\Program Files\VideoLAN\VLC

\libvlc.lib" /machine:x86

最简单的基于libVLC的视频播放器

http://blog.csdn.net/leixiaohua1020/article/details/42363079

编译时可能会遇到error LNK2019: 无法解析的外部符号 \_libvlc\_new 之类的错误。遇到这种问题一般都是由于缺少相应的库文件。右击项目，选择“属性”--“链接器”--“输入”--“附加依赖项”。根据错误中的提示在下图右边的红色框内添加相应的lib文件。



有时候会报vlc PLUGIN 没有找到或者 libvlc\_new()返回NULL，需要配置系统变量

VLC\_PLUGIN\_PATH = D:\GitSanGuo\MySanGuo\sanguo\frameworks\runtime-src\Classes\vlc\bin\

plugins， 并添加到path中。

## 20、cocos2d-x 3 lua回调自定义函数的方法

c++自定义类注册到lua中使用tolua就可以了，然后注册一个lua函数让c++调用。

方法一：在Cocos-2dx 3.x中，Lua回调使用了ScriptHandlerMgr新的机制，通过创建C++对象，HandlerType和lua函数(handlerID)一一对应的映射关系，应用ScriptHandlerMgr进行统一管理。

其实，ScriptHandlerMgr这个类其实就是帮助我们去和lua交互的一个管理器了，所以我们可以看到的是tolua\_Cocos2d\_ScriptHandlerMgr\_registerScriptHandler00 当中的这个是个方法用来在lua中注册的registerScriptHandler。

在Lua中我们注册事件回调



注意，要定义cc.Handler.EVENT\_CUSTOM\_COMMON在frameworks\cocos2d-x\cocos

\scripting\lua-bindings\script下面的Cocos2dConstant.lua中



在frameworks\cocos2d-x\cocos\scripting\lua-bindings\manual\cocos2dScriptHandlerMgr.h这个类里面有个枚举HandlerType注册一下。



在frameworks\cocos2d-x\cocos\scripting\lua-bindings\manual\CCScriptSupport.h中的enum ScriptEventType中定义



在frameworks\cocos2d-x\cocos\scripting\lua-bindings\manual\CCLuaEngine中声明和定义



并在sendEvent中添加







最后，在自定义的C++类中添加Lua的回调



在lua中应用如下



方法二：把Lua函数转换为整形Id值才能够提供给C++调用。





打开lua\_WinMoviePlayer\_auto.cpp文件，找到



改为



此时Lua函数就被转换为int值能够提供给C++调用了

或者改为



其中，Lua\_Function是在CCLuaValue.h定义(typedef int LUA\_FUNCTION)；其实就是int类型故需要#include "scripting/lua-bindings/manual/CCLuaValue.h"和#include "scripting/lua-bindings/

manual/cocos2d/LuaScriptHandlerMgr.h"





## 21、在android中添加一个应用层

下面在android中添加一个公司Logo层SplashScreen。接在在AndroidManifest.xml中添加修改如下：



SplashScreen.java文件就是一个Activity的派生类。



同时在AppActivity.java中添加：



## 22、android目录下的res目录说明

首先说明一下android目录下的res目录。

Android手机屏幕大小不一，有480x320, 640x360, 800x480……怎样才能让Application自动适应不同的屏幕呢？其实很简单，只需要在res目录下创建不同的layout文件夹，比如：layout-640x360、layout-800x480……所有的layout文件在编译之后都会写入R.Java里，而系统会根据屏幕的大小自己选择合适的layout进行使用。

在中Menifest中添加子元素android:anyDensity="true"时，应用程序安装在不同密度的终端上时，程序会分别加载xxhdpi、xhdpi、hdpi、mdpi、ldpi文件夹中的资源。相反，如果设为false，即使在文件夹下拥有相同资源，应用不会自动地去相应文件夹下寻找资源：

1) 如果drawable-hdpi、drawable-mdpi、drawable-ldpi三个文件夹中有同一张图片资源的不同密度表示，那么系统会去加载drawable\_mdpi文件夹中的资源；

2) 如果drawable-hpdi中有高密度图片，其它两个文件夹中没有对应图片资源，那么系统会去加载drawable-hdpi中的资源，其他同理；

3) 如果drawable-hdpi,drawable-mdpi中有图片资源，drawable-ldpi中没有，系统会加载drawable-mdpi中的资源，其他同理，使用最接近的密度级别。

drawable-hdpi里面存放高分辨率的图片,如WVGA (480x800),FWVGA (480x854)。drawable-mdpi里面存放中等分辨率的图片,如HVGA (320x480)。drawable-ldpi里面存放低分辨率的图片,如QVGA (240x320)。xhdpi：至少960\*720。xxhdpi：1280×720。

## 23、R cannot be resolved to a variable问题的解决方法

1. 检查Android 的SDK是否丢失需要重新下载，检查build path，把需要导入的JAR包确认都导入成功。

2. 确保class没有import Android.R，注意是不能有Android.R，可以有其他R文件，比如你自己的包。

3. 错误class引用的layout的xml文件没有错误,经常是xml里面有错误,导致R文件无法正常生成。很多时候都是这个地方出错,可以尝试在layout 的xml文件手写添加一个控件，看id能否在R.Java中自动生成，如果不能，那很大可能就是这个layout 的xml文件有问题，查看格式是否使用正确，或者包含什么非法字符串，或者调用到了不正确的字符串，等等，可以使用排除法，挨个去掉控件，直到发现error message消失或者id能在R.java中自动生成。记得以前经常因为android这个单词拼错而出现R文件无法正常生成而总是R cannot be resolved to a variable。

4. 检查AndroidManifest.xml文件里边的package，layout配置文件，strings.xml等的字符串全部书写正确。

5.删掉gen文件夹，使R.java重新自动生成一次，如果不能生成，继续检查layout的xml文件是否有如上不易发觉的问题。

6.Clean project ，重新build，或者重新import project。

7.重启eclipse。

8.重启电脑，以防Android 虚拟机的问题。

## 24、lua的require和module

require会搜索目录加载文件;require会判断是否文件已经加载避免重复加载同一文件。

Lua保留一张所有已经加载的文件的列表（使用table保存）。如果一个加载的文件在表中存在, 则require简单的返回；表中保留加载的文件的虚名，而不是实文件名。所以如果你使用不同的虚文件名require同一个文件两次，将会加载两次该文件。比如require "foo"和require "foo.lua"，路径为"?;?.lua"将会加载foo.lua两次。我们也可以通过全局变量\_LOADED访问文件名列表，这样我们就可以判断文件是否被加载过；同样我们也可以使用一点小技巧让require加载一个文件两次。比如，require "foo"之后\_LOADED["foo"]将不为nil，我们可以将其赋值为nil，require "foo.lua"将会再次加载该文件。

require(modelname)。require(在lua中它是ll\_require函数)的查找顺序如下：

a. 首先在package.loaded查找modelname,如果该模块已经存在，就直接返回它的值;

b. 在package.preload查找modelname, 如果preload存在，那么就把它作为loader，调用loader(L);

c. 根据package.path的模式查找lua库modelname，这个库是通过module函数定义的，对于顶层的lua库，文件名和库名是一样的而且不需要调用显式地在lua文件中调用module函数(在ll\_require函数中可以看到处理方式)，也就是说lua会根据lua文件直接完成一个loader的初始化过程;

d. 根据package.cpath查找c库，这个库是符合lua的一些规范的(export具有一定特征的函数接口)，lua先已动态的方式加载该c库，然后在库中查找并调用相应名字的接口，例如:luaopen\_hello\_world;

e. 以第一个"."为分割，将模块名划分为:(main, sub)的形式，根据package.cpath查找main，如果存在，就加载该库并查询相应的接口:luaopen\_main\_sub，例如：先查找hello库，并查询luaopen\_hello\_world接口

f. 得到loder后，用modname作为唯一的参数调用该loader函数。当然参数是通过lua的栈传递的，所以loader的原型必须符合lua的规范:int LUA\_FUNC(lua\_State \*L)

ll\_require会将这个loader的返回值赋给package.loaded[modelname],如果loader不返回值同时package.loaded[modelname]不存在时, ll\_require就会把package.loaded[modelname]设为true。最后ll\_reuqire把package.loaded[modelname]返回给调用者。

module的处理流程module(name, cb1, cb2, ...)

　　a. 如果package.loaded[name]是一个table，那么就把这个table作为一个mod

　　b. 如果全局变量name是一个table，就把这个全局变量作为一个mod

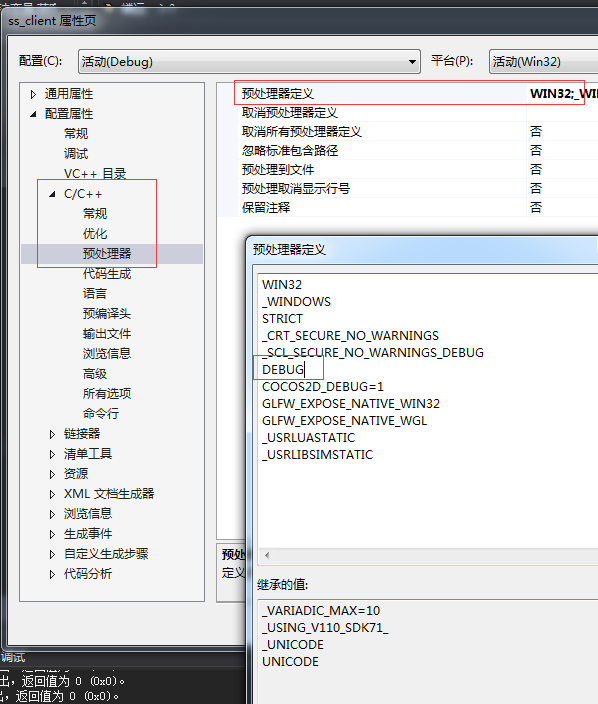
　　c. 创建table:t = {[name]=package.loaded[name], ["\_NAME"]=name, ["\_M"]=t, ["\_PACKAGE"]=\*name\*(删除了最后的".XXXX"部分)}. 如果name是一个以点分割的串，那么得到的mod类似这个样子：hello.world -- {["hello"]={["world"]={XXXXXXX}}}

　　d. 依次调用cbs：cb1(mod), cb2(mod),...

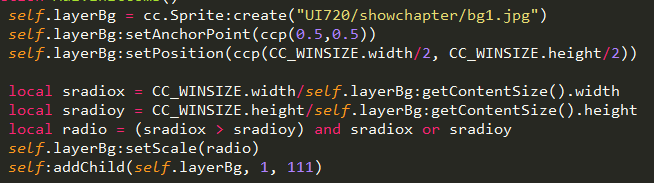
　　e. 将当前模块的环境设置为mod,同时把package.loaded[name] = mod

## 25、visual studio 开启debug模式

为了显示代码中的一些隐藏的问题，可以用visual studio 开启debug模式。



## 26、背景图的一种适配方式



## 27、透明度变化

要想使用cocos2d的透明度函数cc.FadeOut:create(0.3)、cc.FadeIn:create(0.1)、cc.FadeTo:create(0.5, 255)，必需现在父节点中开启

self.ccbiNode:setCascadeOpacityEnabled(true)

## 28、Cocos 图片变灰 倒计时进度条

auto shaderkey = GLProgram::SHADER\_NAME\_POSITION\_GRAYSCALE;

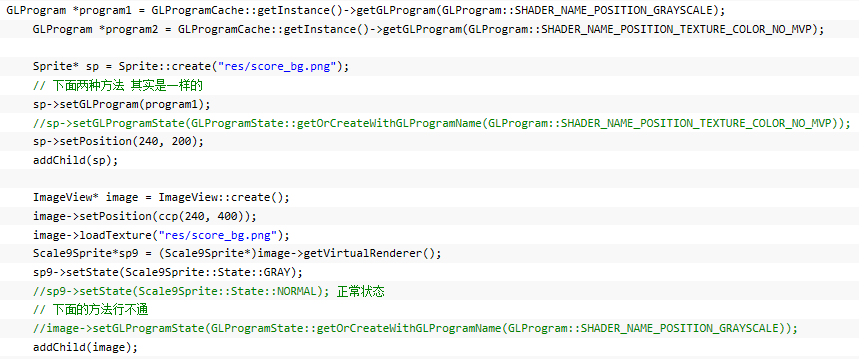
if ((Sprite\*)sp){

((Sprite\*)sp)->setGLProgram(GLProgramCache::getInstance()->getGLProgram(shaderkey));

GLProgram \*program2 = GLProgramCache::getInstance()->getGLProgram(GLProgram::

SHADER\_NAME\_POSITION\_TEXTURE\_COLOR\_NO\_MVP);

}

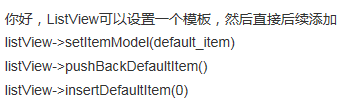


## 29、ListView|ScrollView|TableView|PageView的区别

ScrollView的控件常用的设置属性有：设置scrollView的显示窗口大小（Size）、设置scrollView的容器层大小（ContentSize）、设置BounceBack也就是当scrollView拖到边缘时有一种弹性的效果等。注意，容器层的大小（ContentSize）不应该设置成与显示窗口（Size）一样，而应该比它大，这样才能拖动scrollView。

ListView这货说到底就是tableView，但个人感觉比tableView好用一点。在ListView中不一定要设置容器层的大小，因为当你添加控件到ListView中，如果控件所处的坐标已经超出了ListView的显示窗口范围，那么ListView会自动调整容器层的大小。当出现效率问题的时候建议使用listView，他可以使用setIIemModel 达到使用一个模版，不用重复加载模版，可以提升很高的效率。

listView是继承自scrollview的。scrollview滚动是多个方向的，listview是单个方向的。而且listview可以设置模板，使用时替换数据即可。ListView 不能用来做大量的列表item， 当item数量超过100，加载的时候就会出现无法忍受的延迟了。



PageView其实也就是ScrollView的特别版。scrollView拖动多少界面就挪动多少，PageView拖一点界面就直接翻页了。你必须在PageView控件最初始的状态下才能改变它的Size，如果你已经添加了好几个页面，那么将无法改变它的Size。

tableview可以显示多列元素，并且可以通过滑动来即时加载元素。ListView中的每个单元称为item，TableView的每个单元称为cell，如果分别创建20个单元，但每次只能显示5个单元。当ListView创建的时候，会直接创建20个item；而TableView只会创建5个cell，然后在队列中反复利用。

用TableView的时候，每次滑动，并不会滑动固定的距离，就是说滑动的距离难以固定，而且往往伴随着偏移量，比如手往下一划，当你手离开时候，它还会继续往下划一点（就像有惯性一样）。而PageView就不同了，如果你往下滑动，比如从编号1滑动到编号3的一半，它会自己判断最后定位到滑到3还是4。就像你翻书一样，不可能翻一半，要么不翻动，要么翻过去。

注意，TableView的numberOfCellsInTableView返回的个数和TableView创建的cell数量通常是不一样的，这是因为cocos2dx设计上为了节省资源，创建的cell数量 = tabview的高度 / 单个cell的高度 + 1。所以在触摸和选中等逻辑处理的时候，一定不能使用cell来标识。因为同一个cell物理对象，可能会映射N个逻辑对象。通常的做法是在tableCellAtIndex中把当前cell对应的逻辑对象存起来，这样在tableCellTouched就可以直接找到物理cell对应的逻辑对象来处理了。

## 30、cocos2dx 自带的openURL方法

接SDK需要添加一个渠道的更多游戏的网站链接，百度了下基本都说的是自己在android 和ios OS层增加方法给上层调用，其实在cocos v3版本后已经自带了名为openURL方法，并且把它绑定到Application上了。

http://blog.csdn.net/dcliu2012/article/details/45313653

cc.Application:getInstance():openURL("http://play.cn")

## 31、[解决scrollview上的menu拖动问题以及menu item在可视区外仍能触发的问题](http://blog.csdn.net/n5/article/details/38266855)

在使用cocos2d-x的scroll view或者tableview的时候，会遇到两个问题：

1）scroll view上放menu时，如果拖动menu scroll view不会被拖动，如果scroll view上全是按钮就几乎没地方可以拖动了。

这是由于默认情况下menu如果处理了事件则会吞掉，这样scroll view就收不到事件了，自然无法拖动。

我们只需要设置一个变量m\_bSwallowsTouches来控制是否吞事件，当然为了兼容，m\_bSwallowsTouches默认为true然后我们只要加一个方法 void setSwallowsTouches(bool isSwallowsTouches); 来控制这个menu是否吞事件。当menu被放到一个scroll view里面时，只要调用setSwallowsTouches(false);则拖动按钮时scroll view就会被拖动，并且这个不影响按钮本身的事件处理。

但是事情并没有完，可以拖动后引发了一个新问题：当拖动按钮带着scroll view拖动之后，松开手，按钮被触发了，这样感觉不太爽。其实这个也好解决的，只要判断菜单的世界坐标变了即可。

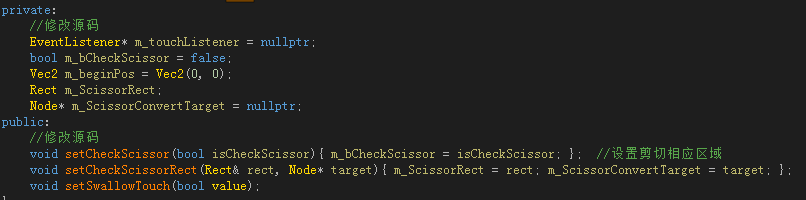
然后在touch end时再获取一下menu的世界坐标，比较一下是否有变化，如果有变化则说明菜单随着scroll view移动了，因此可以取消菜单项的激活。

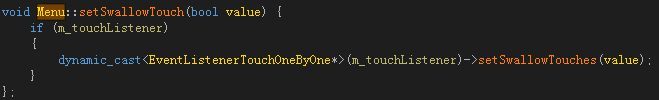
2）当menu item滚动出scrollview的可视区域时，仍然能被触发。

这个问题的原因是menu的事件处理并不知道scroll view的存在，也就更不会知道scroll view绘制时使用了opengl的scissor测试在屏幕上剪切出一个区域，使得只能绘制在该区域中。

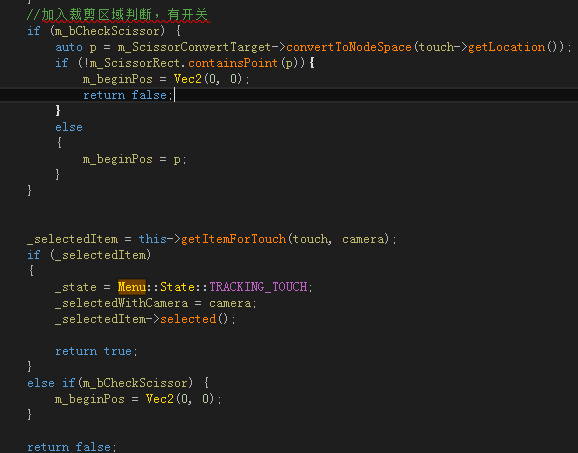
因此要解决这个问题，我们只要当menu在scroll view中时，检测touch点是否发生再scissor box中，如果在scrissor box之外则返回false。

为了效率优化（使用opengl的api去取状态也是要尽量避免的），我们要加一个开关，只有menu在scroll view中时才打开这个开关。

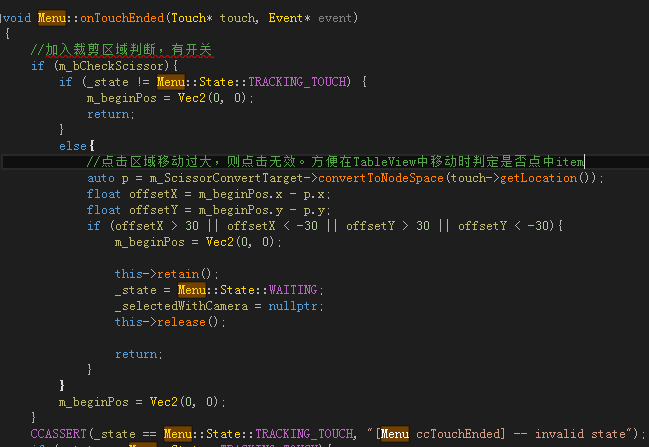




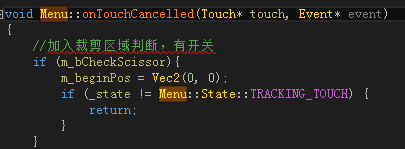
在bool Menu::onTouchBegan(Touch\* touch, Event\* event) 中处理



在void Menu::onTouchEnded(Touch\* touch, Event\* event) 中处理

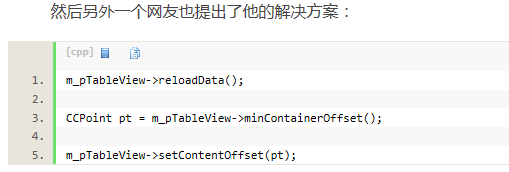


注意在相应的onTouchCancelled()、onTouchMoved()中处理



## 32、tableView的reloadData刷新数据后怎么样停留在当前位置

TableView reloadData实现的下拉刷新，但是要怎么才能停留在当前位置而不是自动滚到顶部呢？

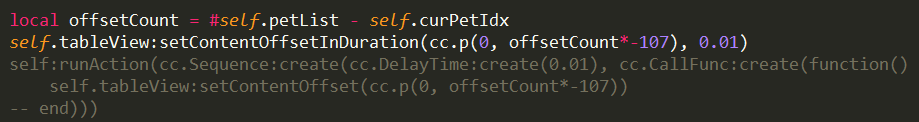


另一位网友给出了另一种方法：知道listView是继承scrollView的，创建出tableView之后，可以通过getInnerContainerSize().height得到tableView的高度，getInnerContainerPosition()得到tableView的位置，，剩下的就是根据新的数据计算出偏移量就可以了。

经测试，第二张单纯偏移量不行，第一种的偏移是拉到底部的便宜。因此综合两者，得出一种经自身验证的方法。即（其中107为cell高度）：







## 33、Lua实现计算 UTF8 字符串的长度，每一个中文算一个字符

function string\_utf8len(input)

local len = string.len(input)

local left = len

local cnt = 0

local arr = {0, 0xc0, 0xe0, 0xf0, 0xf8, 0xfc}

while left ~= 0 do

local tmp = string.byte(input, -left)

local i = #arr

while arr[i] do

if tmp >= arr[i] then

left = left - i

break

end

i = i - 1

end

cnt = cnt + 1

end

return cnt

end

## 34、-1会因不同编译环境而不同，在Win32上转换为无符号255，而安卓为0

## 35、error LNK2001: 无法解析的外部符号 "\_\_declspec(dllimport)

error LNK2001:无法解析的外部符号真的是一个非常头疼的问题；因为问题各种各样，我把这个LNK2001的解决方法贴出来供大家分享，可能不适用于所有的LNK2001，希望对大家有帮助。

OperationScaleSceneOverall2D.obj : error LNK2001: 无法解析的外部符号 "\_\_declspec(dllimport) public: static class DataObject \* \_\_cdecl DataObject::GetObjectW(unsigned int)" (\_\_imp\_?GetObjectW@DataObject@@SAPEAV1@I@Z)

我这个LNK2001报的错误是同一个项目下的另一个工程.obj有错误，这里显示的是GetObjectW的问题，而事实上我另一个工程生成的.obj文件里是GetObjectA。注意到GetObjectW和GetObjectA只有最后一个字符不同，一个是W，另一个是A。W代表宽字符，A代表ASCII码。说明问题出在字符设置上。查看一下，果然，两个工程的生成字符集不同

两个工程生成字符集一个是“未设置”另一个是“使用 Unicode 字符集”，引发LNK2001的错误。同一个项目下的不同工程字符集应该是设为一样的。有相同问题的朋友，希望能对你有帮助。

初学者在学习VC＋＋的过程中，遇到的LNK2001错误的错误消息主要为： unresolved external symbol “symbol”（不确定的外部“符号”）。如果连接程序不能在所有的库和目标文件内找到所引用的函数、变量或标签，将产生此错误消息。一般来说，发生错误的原因有两个：一是所引用 的函数、变量不存在、拼写不正确或者使用错误；其次可能使用了不同版本 的连接库。以下是可能产生LNK2001错误的原因：

　　一．由于编码错误导致的LNK2001。

　　1．不相匹配的程序代码或模块定义（.DEF）文件能导致LNK2001。例如, 如果在C++源文件内声明了一变量“var1”，却试图在另一文件内以变量 “VAR1”访问该变量, 将发生该错误。

　　2．如果使用的内联函数是在.CPP文件内定义的，而不是在头文件内定义将导致LNK2001错误。

　　3．调用函数时如果所用的参数类型同函数声明时的类型不符将会产生LNK2001。

　　4．试图从基类的构造函数或析构函数中调用虚拟函数时将会导致LNK2001。

5．要注意函数和变量的可公用性，只有全局变量、函数是可公用的。静态函数和静态变量具有相同的使用范围限制。当试图从文件外部访问任何没有在该文件内声明的静 态变量时将导致编译错误或LNK2001。函数内声明的变量（局部变量） 只能在该函数的范围内使用。C++的全局常量只有静态连接性能。这不同于C，如果试图在C++的多个文件内使用全局变量也会产生LNK2001错误。一种解决的方法是需要时在头文件中加入该常量的初始化代码，并在.CPP文件中包含该头文件；另一种 方法是使用时给该变量赋以常数。

二．由于编译和链接的设置而造成的LNK2001

　　1．如果编译时使用的是/NOD（/NODEFAULTLIB）选项，程序所需要的运行库和MFC库在连接时由编译器写入目标文件模块，但除非在文件中明确包含这些库名，否 则这些库不会被链接进工程文件。在这种情况下使用/NOD将导 致错误LNK2001。

　　2．如果没有为wWinMainCRTStartup设定程序入口，在使用Unicode和MFC时将得到“unresolved external on \_WinMain@16”的LNK2001错误信息。

　　3．使用/MD选项编译时,既然所有的运行库都被保留在动态链接库之内，源文件中对“func”的引用，在目标文件里即对“\_\_imp\_\_func”的引用。如果试图使用静态LIBC.LIB 或LIBCMT.LIB进行连接，将在\_\_imp\_\_func上发生LNK2001；如果不使用/MD选项编译，在使用MSVCxx.LIB连接时也会发生LNK2001。

　　4．使用/ML选项编译时，如用LIBCMT.LIB链接会在\_errno上发生LNK2001。

　　5．当编译调试版的应用程序时，如果采用发行版模态库进行连接也会产生LNK2001；同样，使用调试版模态库连接发行版应用程序时也会产生相同的问题。

　　6．不同版本的库和编译器的混合使用也能产生问题，因为新版的库里可能包含早先的版本没有的符号和说明。

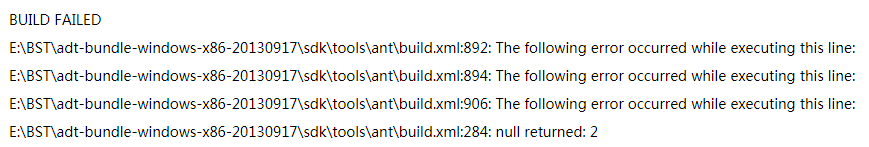
　　7．在不同的模块使用内联和非内联的编译选项能够导致LNK2001。如果创建C＋＋库时打开了函数内联（/Ob1或/Ob2），但是在描述该函数的相应头文件里却关闭了函 数内联（没有inline关键字），这时将得到该错误信息。为避免该问题的发生，应该在相应的头文件中用inline关键字标志内联函数。

　　8．不正确的/SUBSYSTEM或/ENTRY设置也能导致LNK2001。

## 36、使用cocos compile -p android时总是报错误 BUILD FAILED \ant\build.xml:892

错误如下。。。但是使用eclipse生成apk包时不收影响。。





查资料发现，发现：

cocos compile 的脚本里会自动把“frameworks\js-bindings\cocos2d-x\cocos\platform\android

\java\bin\classes.jar”（2dx的java代码）添加到classes.dex中。

而当用eclipse打开android项目时，默认是没有引入2dx的java类的，此时如果想正常运行android项目，需要引入2dx的jar包，或者引入2dx的java源码。

如果引入java的源码，当再次执行“cocos compile”时，2dx的java源码生成的class文件，会跟默认引入的“frameworks\js-bindings\cocos2d-x \cocos\platform\android\java\bin

\classes.jar”发生冲突，会报 “com.android.dx.util.DexException: Multiple dex files define ……”的错误；

如果不引入java源码，而是直接在“Build path”中引入了jar包（位于引擎的：frameworks\js-bindings\cocos2d-x\cocos\platform \android\java\bin\下的classes.jar），此时android的项目不报错了，但是直接使用eclipse生成apk包时会发 现安装到手机上无法运行，这是因为项目需要的2dx文件没有被打入到apk中。此时，可以将对应的jar包放到android项目的libs目录里 （frameworks\runtime-src\proj.android\libs\），这样生成的apk包就可以正常运行了。

但是，如果这样做了，再次执行“cocos compile”时，发现还会报如上的错误，这是因为脚本里会默认将libs目录里的文件添加到classes.dex中。

综上所述：

出现如上错误的原因是在使用“cocos compile”脚本时，2dx文件被重复添加了。

不过，如果像我这样操作，使用“cocos compile”编译脚本（主要是因为该脚本会将js文件和资源拷贝的android项目里），使用eclipse将android项目打包成apk，可以忽略如上问题的；感觉很不爽。。。。

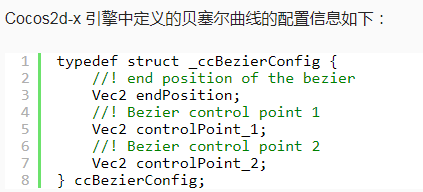
-------------解决方法

注释掉project.properties文件中的#android.library.reference.1=../cocos2d/cocos/platform

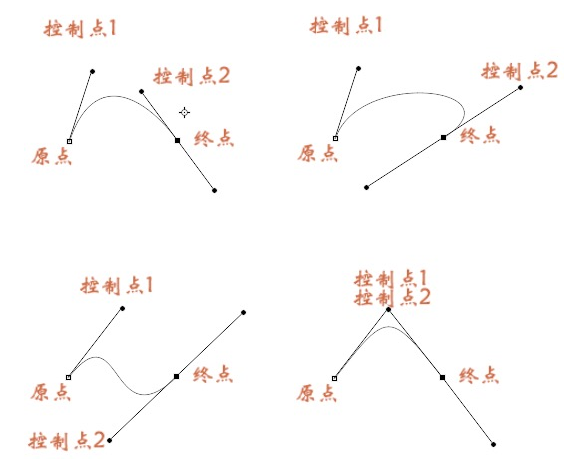
/android/java就可以啦！！！

## 37、贝赛尔曲线

贝赛尔曲线是应用于二维图形应用程序的数学曲线，它是依据四个位置任意的点坐标绘制出的一条光滑曲线。曲线定义的四个点分别是：起始点、终止点以及两个相互分离的中间点。滑动两个中间点，贝塞尔曲线的形状就会发生变化。



结构体中三个点分别是：曲线的终点，控制点1和控制点2。曲线的起点这里没有配置进去，这是因为引擎中 Bezier 的起始点我们认为就是它当前的坐标原点。引擎中 BezierTo / BezierBy 类是根据贝赛尔曲线原理封装的两个动作，它们的区别在于 BezierTo 是设置终点到指定坐标位置，而 BezierBy 则是设置终点到相对的坐标位置。



## 38、Cocos2d-x Lua引擎制作的游戏代码加密

在工程目录终端键入：cocos luacompile -h，可以获得这个命令的帮助：

<https://blog.csdn.net/hqc17/article/details/54565581>

https://www.cnblogs.com/ebchange/p/8671421.html



我们主要用到-s,-d,-e,-k,-b以及--disable-compile这几个参数：

-s就是你需要加密的lua脚本所在的目录。

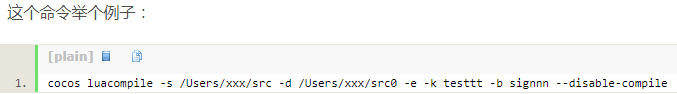
-d就是逆需要将加密后的lua脚本放置的目录。

-e是开启XXTEA加密功能，如果不用这个参数，后面的也就不用了，这样子做是编译为字节码。

-k是在-e开启的情况下使用，XXTEA加密算法的key字段，这个自己定，不要太长，但越复杂越好。

-b是在-e开启的情况下使用，XXTEA加密算法的sign字段，这个自己定，不要太长，但越复杂越好。

--disable-compile是关闭字节码编译。为什么要加这个？因为现有的cocos引擎使用的是luajit来编译字节码，它仅支持32位，不支持64位的机子（截止至2017.01数据），而苹果官方要求上传的app必须兼容arm64的机子，因此只能选择XXTEA加密，这种加密方式也是很不错的，算法小巧，在没有key和sign的情况下不易被破解，兼容32和64位，是目前用cocos开发唯一可取的加密方式。

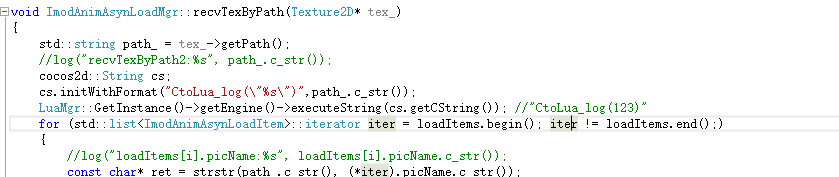


这条demo就是对src的所有lua脚本做key=testtt,sign=signnn的XXTEA加密，并关闭字节码编译。请注意，如果项目中除了lua脚本外，还有其他一些文件，比如plist,json,xml这些，是不会被编译的，实际跑的时候记得拷贝过去，保持源文件和目标文件具有一致性，否则会有问题的。

还需要在AppDelegate.cpp文件的 applicationDidFinishLaunching方法中加入setXXTEAKeyAndSign。这点它那个luacompile 的help就没提了，这是一个坑。

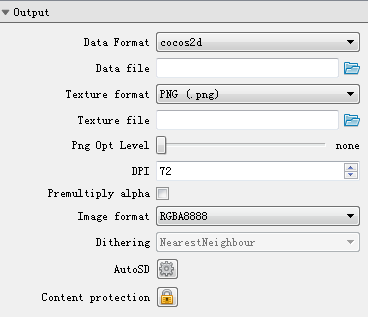


## 39、在C++中调用Lua文件中函数



## 40、TexturePacker教程

TexturePacker是一款快速打包图片的工具，功能强大，非常适合打包游戏素材。单击Edit，如果你的精灵图片都是存放在文件夹里，选择Add Folder，如果是分散的，或只想选特定的几个，可以点击Add Sprites逐个选择添加。左边栏由4个部分的设置组成：output输出设置，和输出的相关参数有关；Geometry图形设置，主要和打包后的形态和大小有关；Layout布局设置，主要就是精灵打包的分布状态；Advance是其他设置，一般用不到。



output设置，自上而下一共有11个可选：

DataFormat是数据输出的格式，也就是说你的游戏基于的引擎选择，有很多引擎框架供选择，包括常见的cocos和unity，当然也可以自己定义输出的格式。

DataFile是基于DataFormat的设置，设置好了某种引擎框架，就可以选择对应的输出文件类型，以及设置路径。比如我选的cocos2d，那么就会输出plist文件，如果选择unity，那么就会输出txt文件，具体文件命名和路径根据实际情况来定。

TextureFormat是打包出来的图像格式，一般做游戏都是用png的图像。

TextureFile是打包的纹理图路径，根据实际情况来定。

PngOptLevel是基于TextureFormat的设置，选择png后会出现这个优化选项，优化可以压缩png的大小，但会降低打包效率。我们做游戏本身资源图片不是很大情况下，就不做优化。

DPI设置，即Dots Per Inch每英寸打印的点数，值越大，图片就越清晰，通常制作场景地图之类的都用300dpi，这里用默认的72是可以的，根据个人情况可以自行调整。

PremultiplyAlpha是预处理alpha通道（透明度）。要搞清楚这个问题，先得理解Alpha通道的工作原理。最常见的像素表示格式是RGBA8888即 (r, g, b, a)，每个通道8位，0-255。例如红色60%透明度就是 (255, 0, 0, 153)，为了表示方便alpha通道一般记成正规化后的0-1的浮点数，也就是 (255, 0, 0, 0.6)。而 Premultiplied Alpha 则是把RGB通道乘以透明度也就是 (r \* a, g \* a, b \* a, a)，50%透明红色就变成了(153, 0, 0, 0.6)。透明通道在渲染的时候通过 Alpha Blending 产生作用，如果一个透明度为 as 的颜色 Cs 渲染到颜色 Cd上，混合后的颜色通过C o =α s C s +(1−α s )C d计算，以60%透明的红色渲染到白色背景为例：

C o =(255,0,0)⋅0.6+(255,255,255)⋅(1−0.6)=(255,102,102)

也就是说，从视觉上，(255, 0, 0, 0.6)渲染到白色背景上 和 (255, 102, 102) 是同一个颜色，如果颜色以 Premultiplied Alpha 形式存储，也就是Cs已经乘以透明度了，所以混合公式变成了：C o =C s  ′ +(1−α s )C d

Premultiplied Alpha 后的像素格式变得不直观，因为在画图的时候都是先从调色板中选出一个RGB颜色，再单独设置透明度，如果RGB乘以透明度就搞不清楚原色是什么了。从前面的 Alpha Blending 公式可以看出，Premultiplied Alpha 之后，混合的时候可以少一次乘法，这可以提高一些效率，但这并不是最主要的原因。最主要的原因是：**没有 Premultiplied Alpha 的纹理无法进行 Texture Filtering（除非使用最近邻插值）**。所以 Premultiplied Alpha 最重要的意义是使得带透明度图片纹理可以正常的进行线性插值。这样旋转、缩放或者非整数的纹理坐标才能正常显示，否则在透明像素边缘附近产生奇怪的颜色。

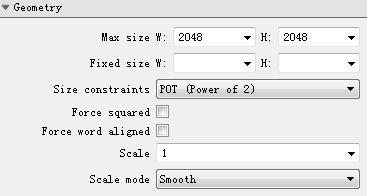
我们使用的PNG图片纹理，一般是不会 Premultiplied Alpha 的。游戏引擎在载入PNG纹理后会手动处理，然后再glTexImage2D传给GPU，比如 Cocos2D-x 中的 CCImage::premultipliedAlpha。而GPU专用的纹理格式，比如 PVR、ETC 一般在生成纹理都是默认 Premultiplied Alpha 的，这些格式一般是GPU硬解码，引擎用CPU处理会很慢。总之 glTexImage2D 传给 GPU 的纹理数据最好都是 Multiplied Alpha 的，要么在生成纹理时由纹理工具 Pre-multiplied，要么载入纹理后由游戏引擎或UI框架 Post-multiplied。

ImageFormat是图像的格式，其实就是设置有没有存在alpha通道，以及RGBA四个通道的顺序和位深度之类，默认RGBA8888，输出为32位深度的图像。如果不想要透明度，就RGB888，输出为24位深度的图像。

Dithering是抖动显示设置，在设置低位深度的图像使用，比如RGBA4444，可用色彩是有限的，不如8888的多，那么可以通过具体选择的抖动算法，来增加更多的色彩，提高图像质量。高深度的图像默认不可选。

AutoSD可以通过增加list来设置不同的输出scale等，应用较少，默认空。

ContentProtection是对输出的内容加密，影响不大，通常也是不需要的。



Geometry图形设置设置，自上而下有7个选择：

MaxSize是设置打包目标的大小，要注意的是下面的Force squared不能勾选才能自己设置。

FixedSize是修正大小参数，通常默认空，如果改变，则MaxSize失效。

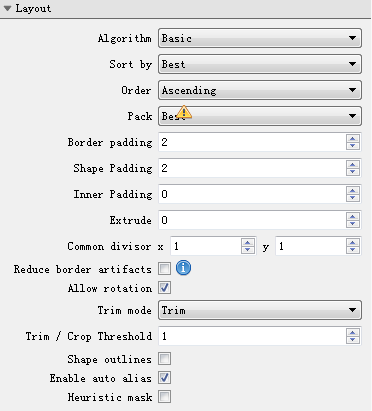
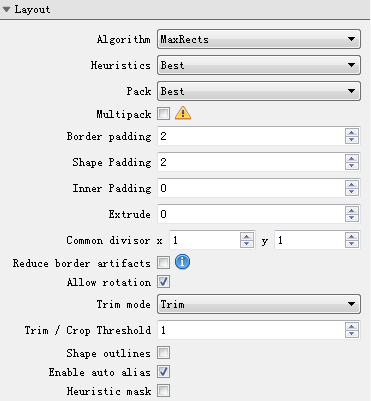
SizeConstraints是和精灵密切相关的，初始默认是POT，POT是2为底的指数值，将精灵强制转换为2,4,8,16……的大小，这种情况用于一些设定为2为底的指数大小的精灵。而通常使用的话会影响打包效果，因此大部分均按照AnySize设置，也就是TP会自己计算精灵图像的最小值，从来参与排列。

ForceSquared是限定打包的结果是正方形，宽和高一样，影响MaxSize的设置。

ForceWordAligned是字对齐设置，是一个优化的参数，默认不勾选。

Scale是缩放比例，可以设置打包后的缩放比例，0-1是缩小，大于1是放大，默认1。

ScaleMode是选择缩放的模式，默认smooth，该设置影响精灵的缩放以及前面Output的AutoSD设置，有不同的算法可选择。



Layout布局设置，自上而下有14-16个选择，我们经过合并后列出12种：

Algorithm是布局的算法，默认MaxRects，使用的空间最小，但常会造成不美观的现象，且算法有限，不方便使用。因此通常选择Basic，可以根据自己喜好调整精灵位置。

Heuristics基于Algorithm的设置，选择MaxRects才有Heuristics，试一下启发式算法，调整布局排列的。如果选Basic则是，可以按照精灵的名字大小等按升序或降序排列，想要的排列顺序可以通过修改名称来实现。

Border/Shape/InnerPadding是加空隙，第一个padding是边界，意思是精灵与精灵之间，精灵与打包之间的空隙，说白了，就是在四周加上一圈透明的，但不改变精灵大小，具体按照希望的像素大小输入即可，默认2，可以根据自己喜好添加，只是美观作用。同样，第二个padding是精灵与精灵之间的空隙，默认2，不与第一个padding叠加，即如果border设置的是2，shape设置的是0，则精灵之间是没有空隙的，与边界有2的空隙。第三个是也是增加一圈透明的，但这圈透明的是计算在精灵大小内的，默认0。这些值影响到Geometry设置的Maxsize大小，如果计算的值卡的很死，那么久应该将这些padding置于0。

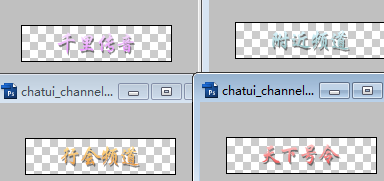
Extrude是将图片精灵边界的像素重复向外扩张，默认0，一般用不到。

Commondivisor是用来扩张精灵大小用的，扩张的部分是透明的，扩展到的大小是可以被这个公约数所除，通常用不到，默认x=1,y=1。

ReduceBorderArtifacts也就是alpha混合处理技术，只在premutiplied alpha关闭情况下有用，默认关闭。

AllowRotation是在软件调整精灵位置的时候，允许软件自动旋转精灵，默认勾选。做游戏通常是不希望旋转的，因为我们有自己的一套旋转资源，再被旋转的话就会出错。这个功能并不适用全部的引擎框架，结合实际情况选择。

TrimMode是修剪模式，选择trim就是自动裁剪精灵图片内部边缘的透明部分，但是保留原精灵大小，这样做可以提高游戏的渲染效率。但有时候设置Trim之后，应用中图片会变形，此时就要关闭Trim。

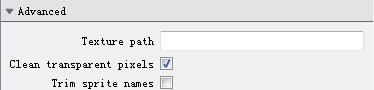


Trim/CropThreshold是裁剪的透明度阈值，若alpha通道的值低于设定的门槛，则会被裁剪。这个设置非常实用，有些精灵需要一定的透明度，则可以自由调节。

ShapeOutline是给精灵划出边界，便于查看，通常用于测试。

EnableAutoAlias是自动使用别名，这是一个优化功能，默认勾选。

Neuristic几乎没怎么用过，意思是对精灵图片的四角像素抽样，并且移除特定的背景色，默认不勾选。



Advaced的功能这里不介绍，就是两个，一个是加路径的，一个是切除后缀名的。

最后，Publish导出资源文件（.plist和png）

41、