

Genesis-3D 工程配置说明

目录

生成 Genesis-3D 工程	1
用引擎调试游戏.....	7
附:	9

生成 Genesis-3D 工程

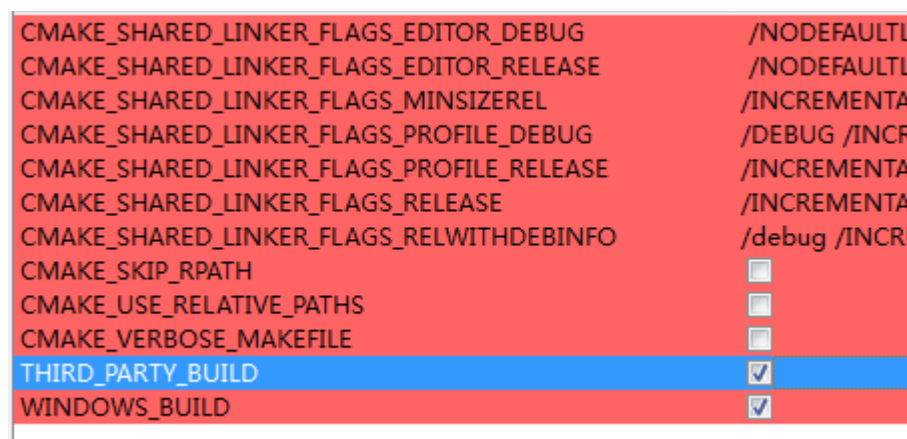
Genesis-3D 引擎的开源工程是用 `cmake` 进行管理的。所以在生成工程之前需要预先下载工程配置工具 `cmake`。推荐 `cmake` 版本在 2.8.6 及以上。

工程里用到了很多开源的第三方库。没有修改过的第三方库引用，不附带到 Genesis-3D 发布的开源版本里。开发者，需自行下载这些第三方库，或在 Genesis-3D 官网上下载整理过的第三方库，并整合到工程中。

Genesis-3D 的第三方库的引用方式有两种，一种是引用的源码，一种是直接引用的 `lib` 库。

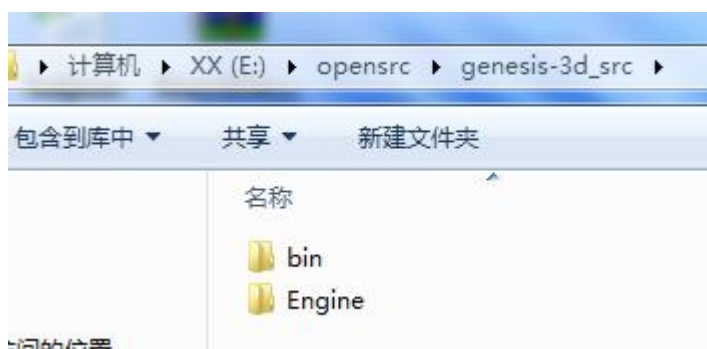
用 `cmake` 生成工程时，默认不把引用到的、以源码形式引用的、未经修改的第三方库添加到工程里。把这些源码添加到工程的指定位置后(不要急喔，后面我们会说到怎么添加)，如果想要把这些源码工程添加到 Genesis-3D 的工程里，在用 `cmake` 生成工程的时候，需要显式的把 `THIRD_PARTY_BUILD` 的值设为 `TRUE`。

如图（以 windows 平台为例）：

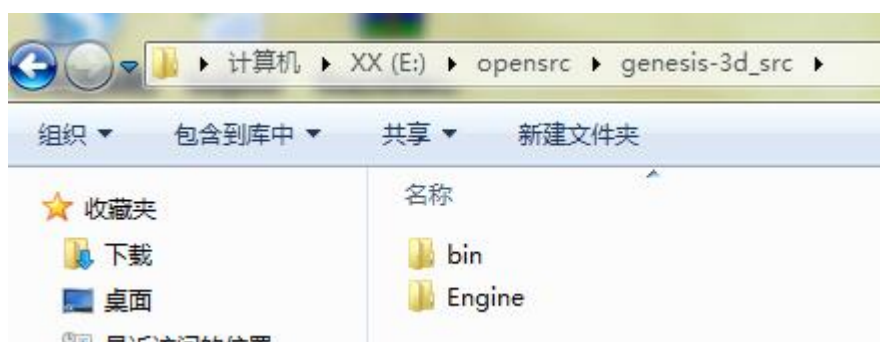


我们也整理了一版可用的第三方库集，并打包给开发者下载。我们现在来讲讲怎么把官方整理的第三方库合到 Genesis-3D 的开源工程去吧。。

如果你解压了引擎的源码包，你会看到这样的目录结构：



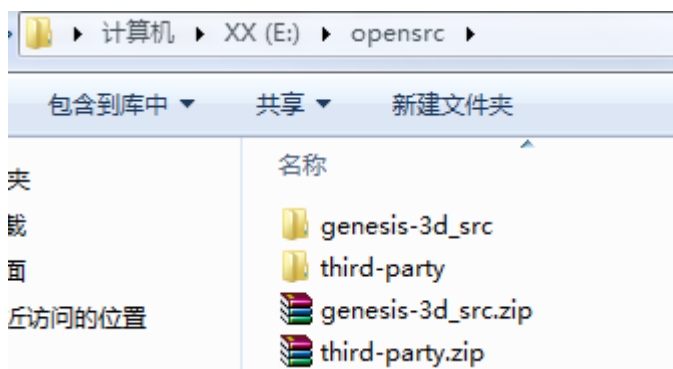
如果你解压了官方发布的第三方库资源包，你会看到这样的目录：



是的没错，这两个包里都分别有自己的 `bin` 和 `Engine` 目录，你要做的就只要把解压出来的第三方库目录下的 `Engine`, `bin` 与源码下的 `Engine`, `bin` 合在一起。就是一个完整的 `Genesis-3D` 工程了。

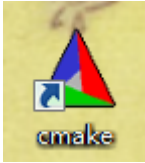
伸手党福利：如何用 `cmake` 生成 `Genesis-3D` 的 `vs2010` 工程。

下载引擎源码包和第三方库集并解压，

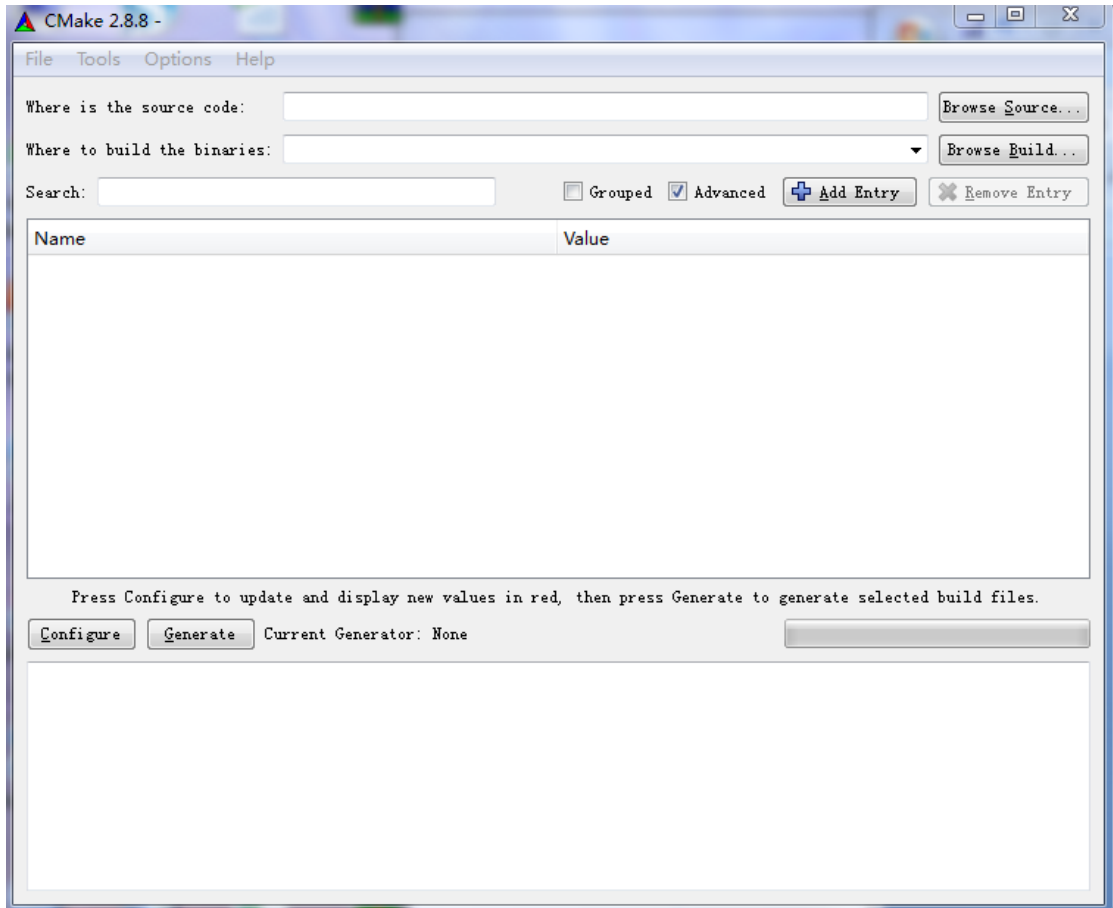


把第三方库目录中的 `engine` 和 `bin` 分别合到源码目录中的 `engine` 和 `bin`。

双击 `cmake` 图标



打开 cmake 界面。

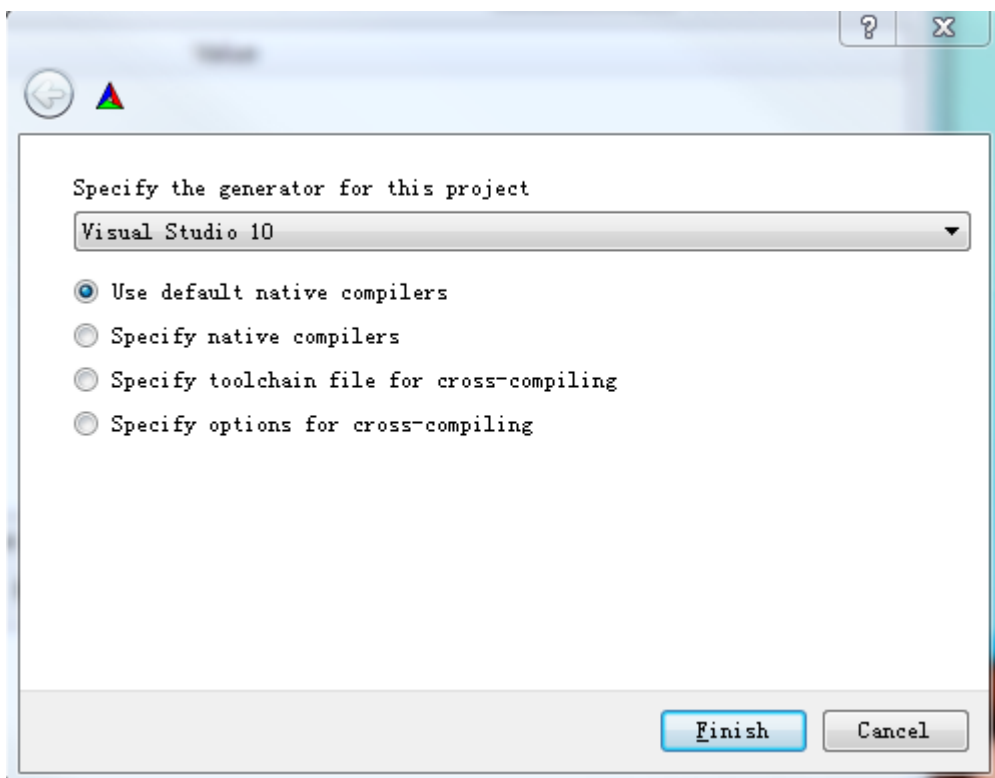


输入源码路径和编译路径：

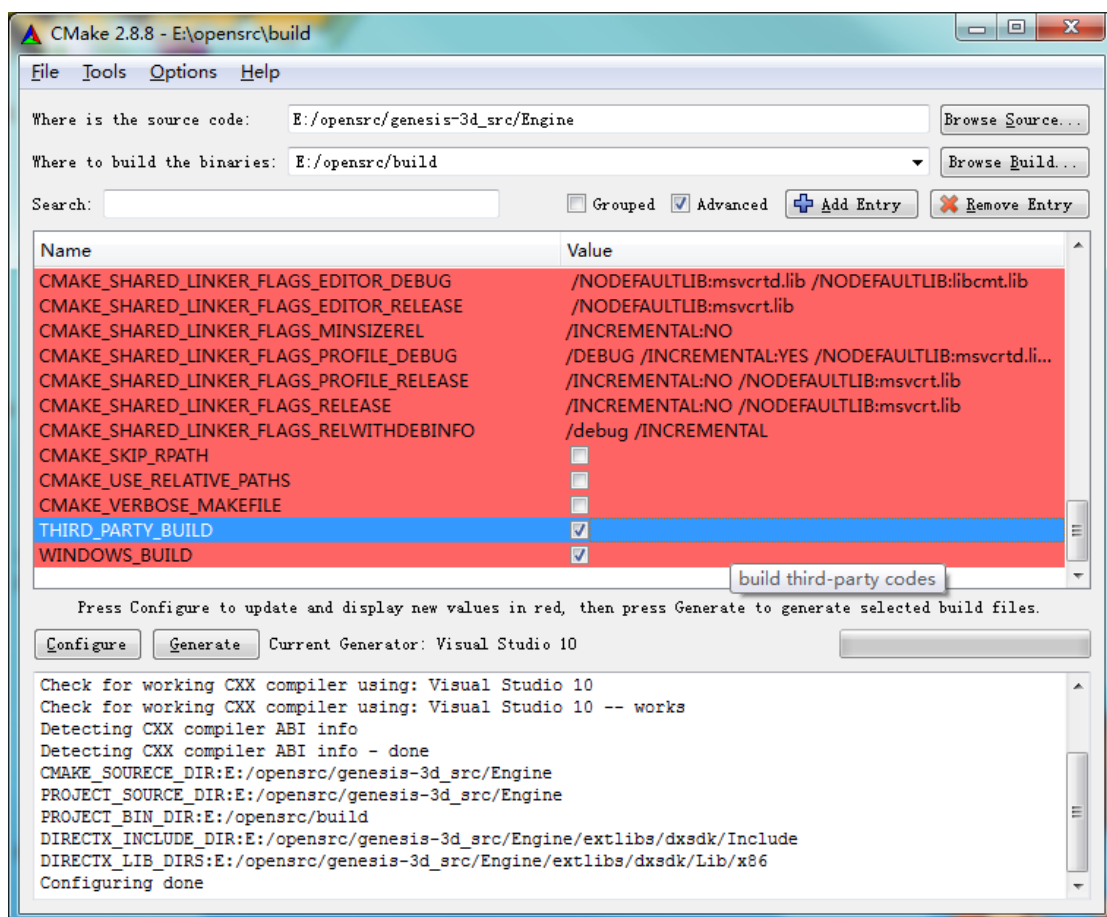


其中，编译路径，是我们自己建的一个空文件夹，这空文件夹的位置没有要求爱建在哪里都可以。

然后点击 **Configure** 按钮。

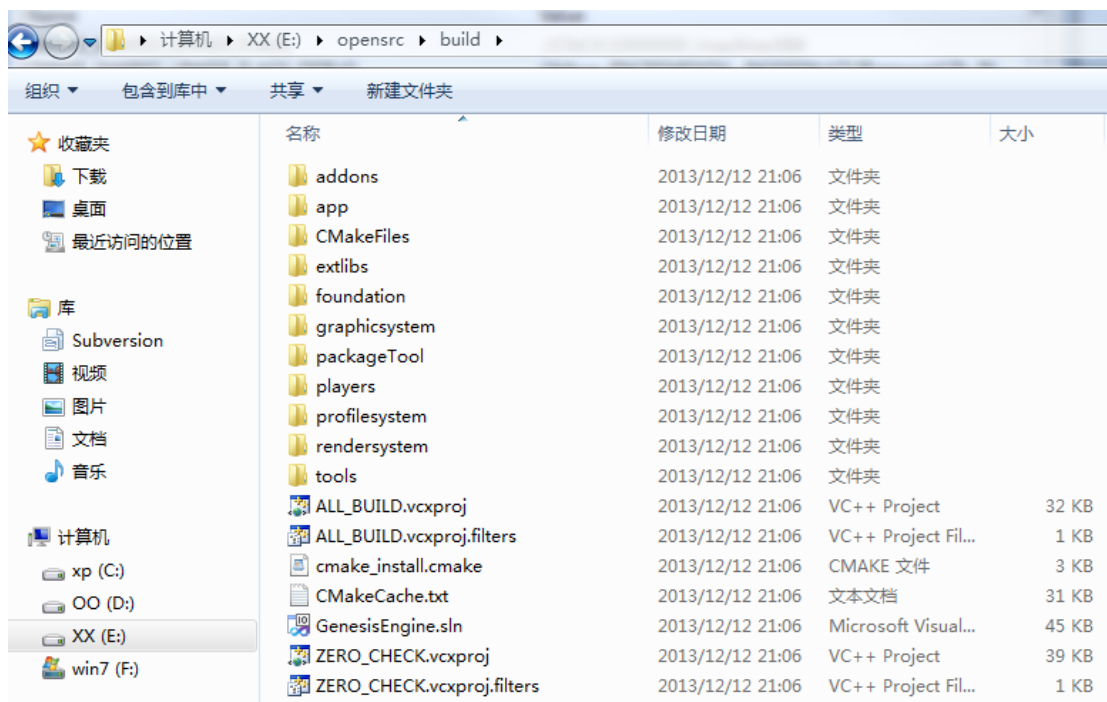


选择 Visual Studio 10， 点击 Finish.
把 *THIRD_PARTY_BUILD* 勾上。



再点击一次 **Configure** 按钮，这时你会看到红色的背景变成白色了。

点 **Generate** 按钮，这样就可以生成 Genesis-3D 的 vs2010 工程了。
打开你的 **build** 文件夹，是不是可以看到熟悉的 vs2010 工程了呢？



双击 GenesisEngine.sln，开始你的新旅程吧。

用引擎调试游戏

以 debug 版的引擎调试游戏为例：

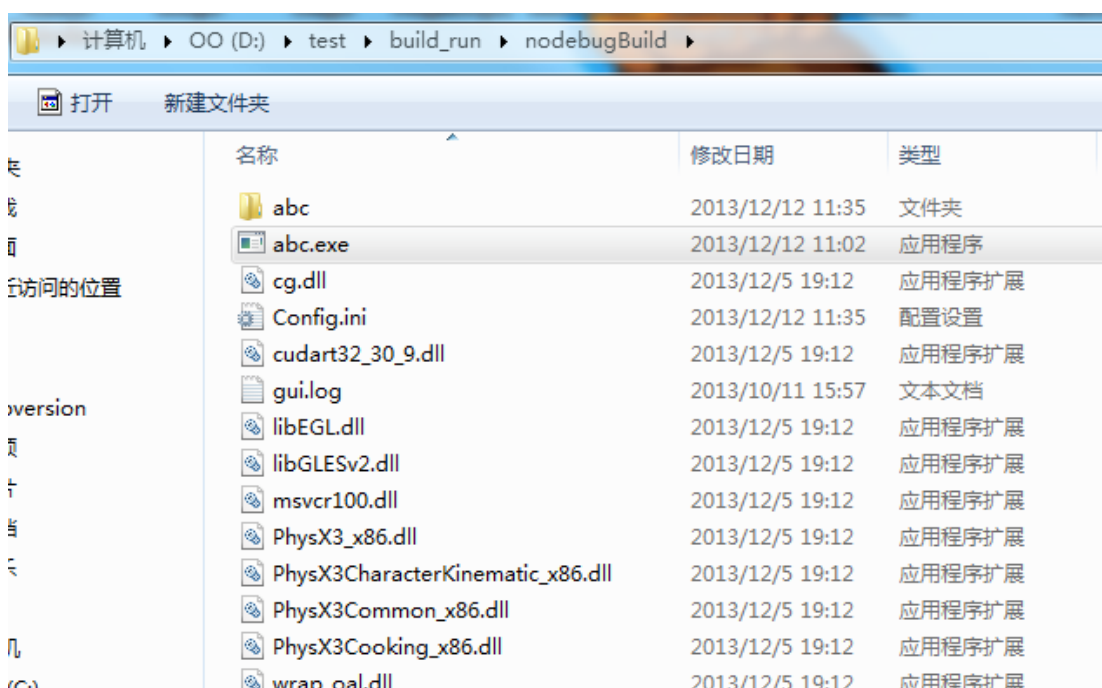
打开 GenesisEngine.sln 编译出你的 exe.

你就可以在源码目录下的“bin\win32\Debug”找到“Genesis.exe”。

言归正传。

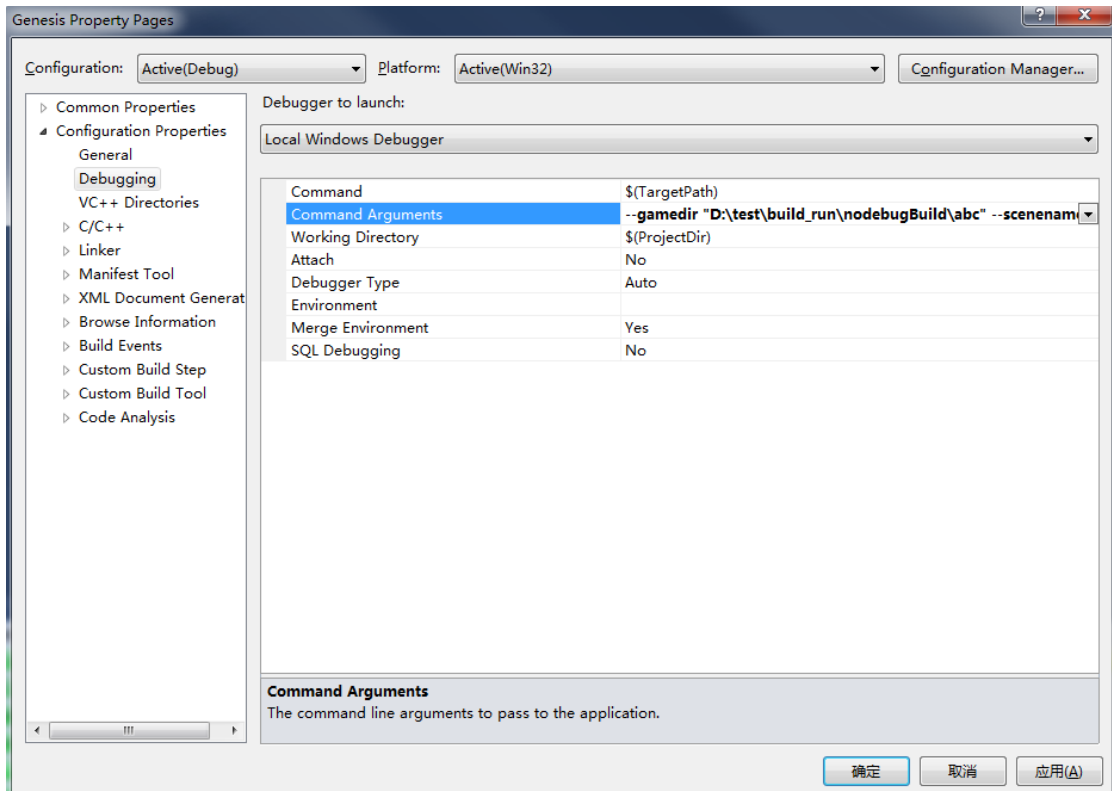
假设我用 Genesis-3D 的编辑器发布了一个游戏，我怎么用引擎源码去调试这个游戏呢？

如图



这是我发布出来的一个游戏。

点击 GenesisEngine solution 下的工程“Genesis”右键选择属性（Property）。

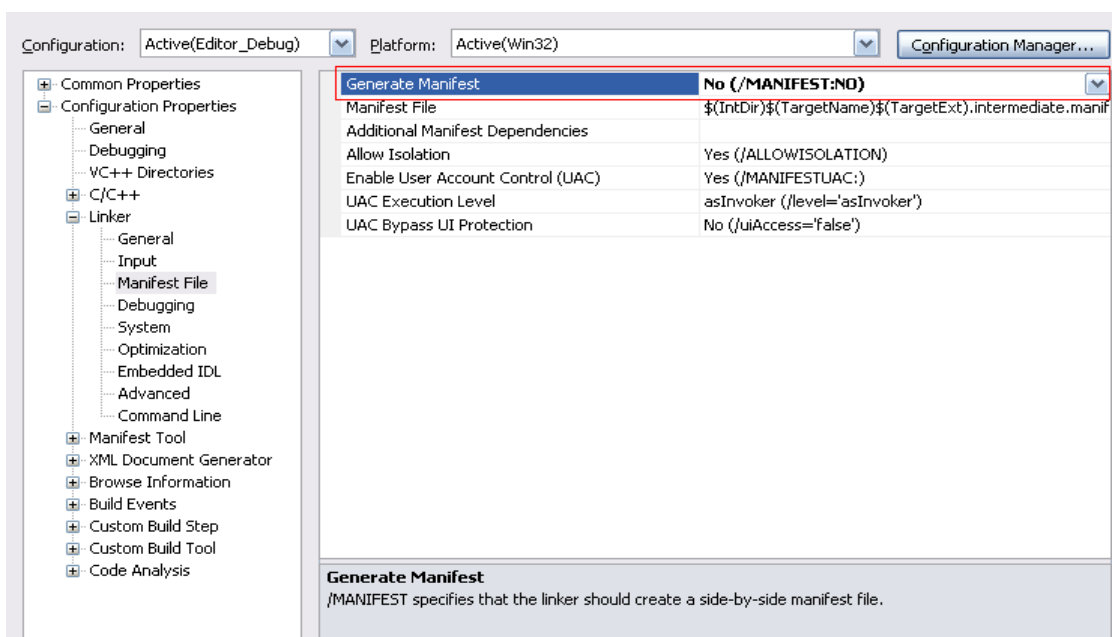


在 Command Arguments 里填入

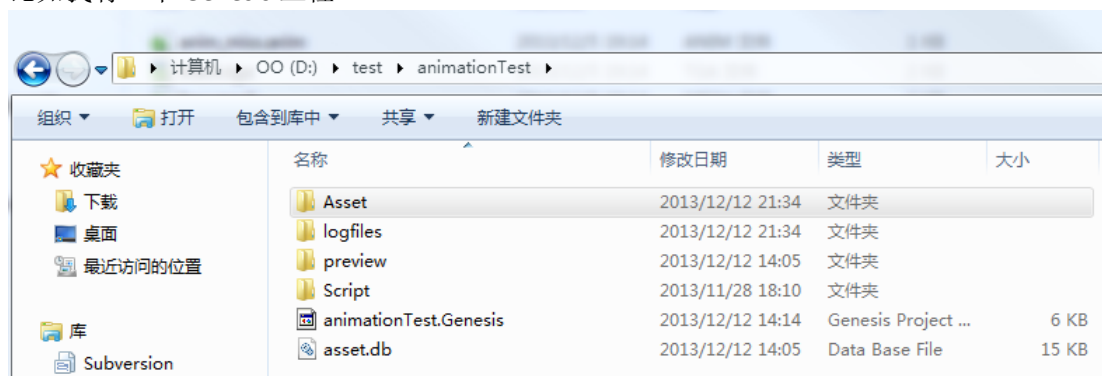
```
--gamedir "D:\test\build_run\nodebugBuild\abc" --scenename "asset:animationTest.scene"
```

gamedir 就是我们发布出来的资源路径，asset:animationTest.scene 就是我们创建好的一个场景，然后，就启动调试吧。

提示：如果启动失败，提示参数有误，按照下图修改：



什么？你嫌每次调试都要先发布很麻烦？
好吧，你也可以直接 Genesis-3D 编辑器创建的 Genesis 工程哦。
比如我有一个 Genesis 工程



我只需把 Command Arguments 里的命令行参数改为：

```
--gamedir "D:\test\animationTest" --scenename "asset:animationTest.scene" --sysdir  
"F:\Program Files (x86)\GenesisEditor\DefaultAssets\System" --shddir "F:\Program Files  
(x86)\GenesisEditor\DefaultAssets\System"
```

好吧，这个参数确实列表太长了。

发布出来的 demo 目录里是有 sysdir 和 shddir 的，但 Genesis 工程里没有这两个目录，
所以要在 Command Arguments 里指定这两个目录。

sysdir 和 shddir 可以在编辑器安装目录里找到的。

然后，开始调试吧。

附：

第三方库源码引用的目录结构：

```
"Engine\extlibs\boostWraper",  
"Engine\extlibs\FreeImage",  
"Engine\extlibs\freetype",  
"Engine\extlibs\mpg123",  
"Engine\extlibs\NDK",  
"Engine\extlibs\Ogg",  
"Engine\extlibs\OPCODE",  
"Engine\extlibs\OPCODEA",  
"Engine\extlibs\OpenGLS2.0",  
"Engine\extlibs\physX3",
```

"Engine\extlibs\tinyxml",
"Engine\extlibs\Vorbisfile",

第三方库的目录结构:

"bin\lib\fileservice\debug\libcurl.lib",
"bin\lib\fileservice\debug\libprotobuf.lib",
"bin\lib\fileservice\debug\libsqlite.lib",
"bin\lib\fileservice\debug\libzmq.lib",
"bin\lib\fileservice\release\libcurl.lib",
"bin\lib\fileservice\release\libprotobuf.lib",
"bin\lib\fileservice\release\libsqlite.lib",
"bin\lib\fileservice\release\libzmq.lib",

"bin\lib\CG",
"bin\lib\FreeImage",
"bin\lib\GlsOptimizer",
"bin\lib\hls2glsfork",
"bin\lib\hlslang",
"bin\lib\mojoshader",
"bin\lib\Opcode",
"bin\lib\OpenGLES2.0",
"bin\lib\PhysX3",
"Engine\extlibs\dxsdk",

"bin\win32\Debug\cg.dll",
"bin\win32\Debug\cudart32_30_9.dll",
"bin\win32\Debug\libEGL.dll",
"bin\win32\Debug\libGLESv2.dll",
"bin\win32\Debug\wrap_oal.dll",
"bin\win32\Debug\PhysX3CharacterKinematicCHECKED_x86.dll",
"bin\win32\Debug\PhysX3CHECKED_x86.dll",
"bin\win32\Debug\PhysX3CommonCHECKED_x86.dll",
"bin\win32\Debug\PhysX3CookingCHECKED_x86.dll",

"bin\win32\Release\cg.dll",
"bin\win32\Release\cudart32_30_9.dll",
"bin\win32\Release\libEGL.dll",
"bin\win32\Release\libGLESv2.dll",
"bin\win32\Release\wrap_oal.dll",
"bin\win32\Release\PhysX3_x86.dll",
"bin\win32\Release\PhysX3CharacterKinematic_x86.dll",
"bin\win32\Release\PhysX3Common_x86.dll",

"bin\win32\Release\PhysX3Cooking_x86.dll",

"bin\win32\Profile_Debug\cg.dll",

"bin\win32\Profile_Debug\cuda32_30_9.dll",

"bin\win32\Profile_Debug\libEGL.dll",

"bin\win32\Profile_Debug\libGLSv2.dll",

"bin\win32\Profile_Debug\wrap_oal.dll",

"bin\win32\Profile_Debug\PhysX3CharacterKinematicCHECKED_x86.dll",

"bin\win32\Profile_Debug\PhysX3CHECKED_x86.dll",

"bin\win32\Profile_Debug\PhysX3CommonCHECKED_x86.dll",

"bin\win32\Profile_Debug\PhysX3CookingCHECKED_x86.dll",

"bin\win32\Profile_Release\cg.dll",

"bin\win32\Profile_Release\cuda32_30_9.dll",

"bin\win32\Profile_Release\libEGL.dll",

"bin\win32\Profile_Release\libGLSv2.dll",

"bin\win32\Profile_Release\wrap_oal.dll",

"bin\win32\Profile_Release\PhysX3_x86.dll",

"bin\win32\Profile_Release\PhysX3CharacterKinematic_x86.dll",

"bin\win32\Profile_Release\PhysX3Common_x86.dll",

"bin\win32\Profile_Release\PhysX3Cooking_x86.dll",