Skip to content

- Home
- API
  - CUDA
  - DirectX
  - Mantle
  - OpenGL
  - OpenGL ES
  - WebGL
- GPU
- KlayGE
  - DXBC2GLSL
  - glloader
  - KFL
  - kfont
  - KlayMark
  - MeshMLLib
- Mobile
- News
- Physics
- Resource
  - Model
  - Tool
- Tech article
  - Trick
- Uncategorized
- Website

KlayGE游戏引擎

To arm your engine with cutting-edge technology

- About
  - Features
  - FAQ
  - Wishlist

用CMake管理Android工程

Apr 25
KlayGE, Mobile, Tech article

转载请注明出处为KlayGE游戏引擎，本文的永久链接为http://www.klayge.org/?p=2851

2年前KlayGE的工程文件就已经在王锐的帮助下切换到CMake。上个版本中，第三方库也已经都改用CMake。但是，对于Android版本来说，原先仍然使用的是NDK的那套东西。所以在维护上带来了不小的开销。随着代码量越来越大，如果能把Android也纳入到CMake工程中，开发起来就能更方便。所以前一阵子做了些探索，最终解决了这个问题。

原有的编译方法

我一直都没有使用Eclipse，而是纯用命令行的方式进行编译。需要：

1. 准备好AndroidManifest.xml、Android.mk、 Application.mk等工程文件
2. android update project -p . -s –target=android-10，生成build.xml等

3. ndk-build，编译出.so
4. 把所需资源编译和拷贝到assets目录
5. ant debug，打包成apk
6. adb install bin/xxx-debug.apk，把apk安装到Android

一般把2-6放在一个脚本里，直接执行那个脚本就能完成这一系列动作。这么做就需要手工对每个工程写AndroidManifest.xml、Android.mk、Application.mk等，费时费力。而且其中大部分与CMake里的内容重复。

另外，使用NDK的工程还有一系列的限制，比如lib的目录结构、module的位置等。都是技术上没必要的人为限制。

新的编译方法

android-cmake

现在的NDK提供了make-standalone-toolchain.sh，本质上就能把编译工具链当作一个独立的gcc编译系统使用。这给连到CMake提供了可能。CMake本身是可以通过命令行参数来指定使用非默认的编译器。不过如果那么做的话，就需要指定相当多的参数，会非常的繁复。好在有了android-cmake这个神器。android-cmake的目标正是让CMake能与Android开发结合起来。其中最关键的就是提供了一个叫android.toolchain.cmake的文件。它可以简单地通过cmake -DCMAKE_TOOLCHAIN_FILE=path/to/android.toolchain.cmake这样的形式，一次设定所有CMake所需的路径、编译器名、链接器名等信息。其他参数都和原先使用CMake的时候无区别。

需要注意的是，googlecode上的android-cmake已经不更新了，作者搬到了github上，但原网站并没有注明。这个配置文件也被用到了OpenCV中。所以从OpenCV的source tree里也能下载到。

交叉编译

android-cmake的使用只是迈出了第一步，要完全切换到CMake，还需要一些额外工作。首先遇到的问题是，以前KlayGE的python编译脚本和CMakeLists.txt都只考虑了同平台编译，没有考虑交叉编译的情况。而编译到Android很显然是个交叉编译的过程。所以py和CMakeLists里都需要做相应的修改，把host platform和target platform分开。并针对不同情况做独立判断。原先gcc的分支就已经是完备的，所以只需要加为数不多的几处判断即可。经过这些修改，.a和.so可以顺利生成，也就是替换了前面的第三步。这下彻底不需要ndk-build、Android.mk和Application.mk了。把工程文件统一到了同样的CMakeLists.txt里。NDK带来的限制也都不存在了。

这里有个坑在于whole-archive。原先使用Android.mk的时候可以直接指定一些库是通过whole-archive的方式进行链接，而其他库是普通的方式。对于循环以来的情况，需要用whole-archive来解决。在CMake里，这里就只有一个TARGET_LINK_LIBRARIES了，无法区分。插了一些网站后，发现了一个解决方法，用-Wl,–whole-archive a b c -Wl,–no-whole-archive d e f这样的方式，就能指定哪些库用whole-archive，哪些不用。

更进一步

对于AndroidManifest.xml和res/values/strings.xml这样的文件，虽然每个工程都不同，但内容大同小异。所以只要写个.in，然后同CMake的CONFIGURE_FILE生成工程所需的文件即可。对于资源的编译和拷贝，可以把资源的文件写到一个独立分组CONTENT_FILES里。在PRE_BUILD的时候扫一遍CONTENT_FILES，把需要编译的编译了，都拷贝到assets里。所以，步骤1，3，4都被CMake取代了。对于其他几个步骤，也可以用ADD_CUSTOM_COMMAND，加到PRE_BUILD和POST_BUILD。这么一来，整个编译过程都放到了CMake中，不再需要直接接触Android的NDK和SDK。

总结

至此，CMake已经可以完全取代原先NDK的方法。只需要管理CMakeLists.txt，就能兼顾Windows、Linux、Android三个平台的编译。而且整个过程都可以使用公开的工具完成，不需要对CMake做额外修改，很方便。

接下去我会探讨一下如何把WinRT平台也纳入CMake中进行管理。这次就没那么容易了，需要对CMake打补丁才可以。

你可能感兴趣的:

- **用CMake管理WinRT工程**
- **不同平台上的HDR Post Process**
- **用Android NDK r6编译boost 1.47**
- **KlayGE 4.4中渲染的改进（五）：OpenGL 4.4和OpenGLES 3**
- **BC7的快速压缩（一）：BC7的结构**

Direct3D 12 API预览 **用CMake管理WinRT工程**

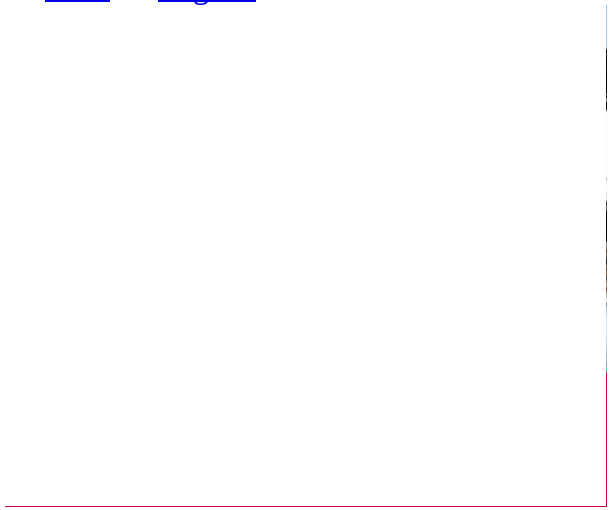
cmake

Comments

- 用CMake管理WinRT工程 - KlayGE游戏引擎
  April 26th, 2014 at 5:42 PM

[…] 上一篇我提到了如何用CMake管理Android工程。KlayGE除了Windows、Linux和Android之外，还支持WinRT平台。是的，WinRT虽然和Windows桌面很接近，但在一些细节上存在一些差异，使得它们应该被考虑为两个不同的平台。就好像Android和Linux应该考虑为不同平台一样。 […]

RSS feed for this post (comments)
KlayGE游戏引擎

- 中文 English
- 
- 
- If you liked KlayGE, please consider a donation. Even $1 helps.

- # Blogroll
  - OpenGPU Forum (开源图形处理器论坛)
  - SALVIA软件渲染器
  - Xbox笔记

- **Tags**

3rd party AA Adreno AMD Android anti-alias BC7 boost BRDF C++11 cmake CUDA D3D11 D3D12 Deferred driver engineering extension git GLSL GPGPU hash hdr hg HLSL Intel iOS Kinect Mali NDK NVIDIA PBR plan PVR release Rendering SDK shader shadow SIGGRAPH SRAA Stereo TBDR Tegra Tessellation

- **Categories**

  - API
    - CUDA
    - DirectX
    - Mantle
    - OpenGL
    - OpenGL ES
    - WebGL
  - GPU
  - KlayGE
    - DXBC2GLSL
    - glloader
    - KFL
    - kfont
    - KlayMark
    - MeshMLLib
  - Mobile
  - News
  - Physics
  - Resource
    - Model
    - Tool
  - Tech article
    - Trick
  - Uncategorized
  - Website

- **Sponsors**

- **In-site Search**

Search