## 深入浅出 - Android系统移植与平台开发（三）－ 编译并运行Android4.0模拟器

1.    编译Android模拟器

在Ubuntu下，我们可以在源码里编译出自己的模拟器及SDK等编译工具，当然这个和在windows里下载的看起来没有什么区别。

编译[**Android**](http://lib.csdn.net/base/15)模拟器的步骤和编译Android系统很相似：

Ø  source build/envsetup.sh

Ø  lunch sdk-eng

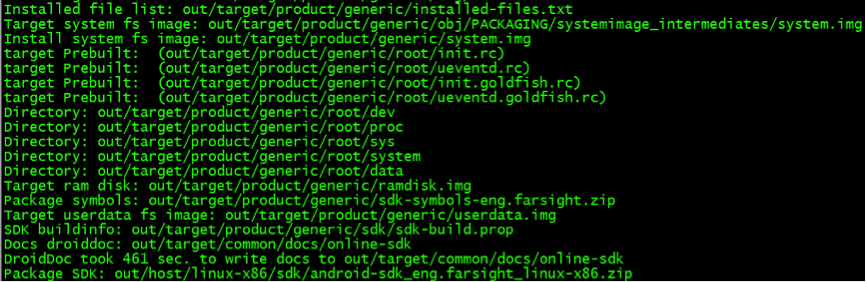
Ø  make sdk

在每次执行make的时候，编译系统会自动的验证上次编译项和这次编译项，如果两次不一样，会删除所有的目标代码，其实现在./build/core/main.mk

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. 71 # This allows us to force a clean build - includedafter the config.make
2. 72 # environment setup is done, but before we generateany dependencies.  This
3. 73 # file does the rm -rf inline so the deps which areall done below will
4. 74 # be generated correctly
5. 75 include $(BUILD\_SYSTEM)/cleanbuild.mk

我们将第75行注释掉，这样就不会每次都自动删除上次编译的中间代码重新编译了。 编译完之后效果如下图所示：



sdk安装在了下面的目录里：

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. WORKING\_DIRECTORY/out/host/linux-x86/sdk

在该目录下有一个压缩包，里面就是我们从eclipse里更新下载指定Target的sdk包。

如果你做过app开发都会知道，我们只有下载了不同版本的sdk之后才能创建不同版本的模拟器。当我们编译完sdk之后就可以创建并运行Android4.0的模拟器了。

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. $ cd WORKING\_DIRECTORY/out/host/linux-x86/sdk/android-sdk\_eng.linux\_linux-x86/tools

首先，查看当前编译出来的sdk的id号，用于创建一个模拟器：

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. $ ./android list targets
2. Available Android targets:
3. ----------
4. id: 1 or "android-14"
5. Name:Android 4.0
6. Type:Platform
7. API level:14
8. Revision: 2
9. Skins:WXGA800, WQVGA400, WXGA720, HVGA, WVGA854, WQVGA432, WVGA800 (default), QVGA,WSVGA
10. ABIs : no ABIs.

创建一个模拟器

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. $ ./android create avd -n mydroid -t 1

注：-n 表示指定模拟器的名字， -t表示指定创建模拟的版本id

启动模拟器

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. $ ./emulator -avd mydroid &

由于模拟器要启动android系统，它要使用几个映像文件，分别是：kernel-qemu , system.img, userdata.img, ramdisk.img

Ø  kernel-qemu：内核映像文件，它是linux操作系统针对qemu模拟器编译的内核文件，里面安装有qemu的仿真硬件驱动，如果将来在开发板或手机上移植android，那么，第一步就是要先make出一个支持开发板或手机的kernel来。

Ø  system.img：是android系统的文件系统，里面包含了android系统的应用程序（apk），系统用到的各种库（jar, so, a）和资源，配置文件(etc目录下)，系统命令(bin,usr/bin, xbin），该映像文件是由out/target/product/PRODUCT/system目录打包生成的，我们可以对这个目录里的东西进行定制化，比如，你要想让android系统默认为安装一个应用程序，那么可以将要安装的apk文件拷贝到out/target/product/PRODUCT/system/app目录下

Ø  userdata.img：用户数据映像，里面包含有程序安装信息，好比如是windows的C:/Program Files/目录

Ø  ramdisk.img：内存磁盘映像，这个文件对于没有做过Linux移植的人来说，不好理解，linux内核启动起来，要挂载一个文件系统作为自己的根文件系统，通常文件系统是存放在块设备上的，也就是硬盘或flash上的，里面含有Linux内核启动过程中依赖的一些程序和配置文件，前面所述的userdata.img和system.img映像，在linux系统启动起来后挂载到其data，system目录下。

我们可以通过下面的脚本run\_emulator.sh，来指定Android模拟器来加载指定的映像文件：

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. #!/bin/bash
2. PWD\_PATH=`pwd`
3. IMG\_PATH=$PWD\_PATH/out/target/product/generic
4. export PATH=$PWD\_PATH/out/host/linux-x86/sdk/android-sdk\_eng.linux\_linux-x86/tools:$PATH
5. export ANDROID\_PRODUCT\_OUT=$PWD\_PATH/out/target/product/generic
6. export ANDROID\_SWT=$PWD\_PATH/out/host/linux-x86/framework
7. emulator -kernel $PWD\_PATH/kernel/goldfish/arch/arm/boot/zImage

注：

ANDROID\_PRODUCT\_OUT：环境变量必须指定，emulator就是从该环境变量下查找所有的映像文件的

-kernel：指定内核映像

当然也可以分别指定每个映像文件，通过下面的命令：

**[plain]** [view plain](http://blog.csdn.net/mr_raptor/article/details/7958212) [copy](http://blog.csdn.net/mr_raptor/article/details/7958212)

1. $ emulator -kernel $PWD\_PATH/kernel/goldfish/arch/arm/boot/zImage \
2. -image $IMG\_PATH/system.img \
3. -data $IMG\_PATH/userdata.img \
4. -ramdisk $IMG\_PATH/ramdisk.img

注：

-system指定system.img

-data指定userdata.img

-ramdisk指定ramdisk.img

**如果将来要运行adb shell连接模拟器的话，我们都要使用root权限，否则会出现：error：device not found的错误。**

通过启动起来的Android系统->System Settings->About phone，可知使用的是我们自己编译的内核启动的。

