
第一章 Android 开发环境

1.1 下载 android 源码

安装相关工具

```
sudo apt-get install curl gnupg flex bison gperf libstdc++-dev libstdc++0-dev  
libwxgtk2.6-dev build-essential zip libncurses5-dev zlib1g-dev
```

如果需要检测内存泄露、堆栈破坏、数组溢出，可安装 **valgrind**

```
sudo apt-get install valgrind
```

创建目录

创建并切换到 **android** 工作目录

```
mkdir ~/android  
cd ~/android
```

创建 **bin** 目录、分支目录、主线目录

```
mkdir ~/android/bin  
mkdir ~/android/froyo  
mkdir ~/android/master
```

安装配置 repo

下载 **repo**，设置可执行属性：

```
cd bin  
curl http://android.git.kernel.org/repo >repo  
chmod a+x repo
```

修改 **repo** 脚本，使用 **http** 协议替代 **git** 协议：

```
gedit repo  
REPO_URL='git://android.git.kernel.org/tools/repo.git'
```

修改为：

```
REPO_URL='http://android.git.kernel.org/tools/repo.git'
```

创建分支源代码库：

```
cd ../froyo  
../bin/repo init -u http://android.git.kernel.org/platform/manifest.git -b  
froyo
```

创建 **master** 主线源代码库，则无须 **-b** 选项：

```
cd ../master  
../bin/repo init -u http://android.git.kernel.org/platform/manifest.git
```

初始化过程中，提示配置 git 帐户，回车默认即可。

```
Your Name [xxx]: ✓  
Your Email [xxx@xxx.(none)]: ✓  
Your identity is: xxx xxx@xxx.(none)  
Is this correct [y/n]? y✓
```

但前目录下应该有隐藏目录.repo，修改配置文件，使用 http 协议替代 git 协议

```
gedit froyo/.repo/manifests/default.xml
```

将

```
fetch="git://android.git.kernel.org/"
```

改为

```
fetch="http://android.git.kernel.org/"
```

下载源代码，大约耗时 8 小时 39 分

```
../bin/repo sync
```

1.2 搭建开发环境

安装配置 Java

Froyo 标准分支需要 Java5，因为新版本的 ubuntu 中没有 java5 安装文件，需要修改 apt 源，使用 ubuntu9.10 源安装的 java5

```
sudo cp /etc/apt/sources.list /etc/apt/sources.list_backup  
sudo gedit /etc/apt/sources.list
```

添加如下源的配置

```
deb http://run.hit.edu.cn/ubuntu/ jaunty main restricted universe multiverse  
deb-src http://run.hit.edu.cn/ubuntu/ jaunty main restricted universe multiverse  
deb http://run.hit.edu.cn/ubuntu/ jaunty-updates main restricted universe  
multiverse  
deb-src http://run.hit.edu.cn/ubuntu/ jaunty-updates main restricted universe  
multiverse  
deb http://run.hit.edu.cn/ubuntu/ jaunty-backports main restricted universe  
multiverse  
deb-src http://run.hit.edu.cn/ubuntu/ jaunty-backports main restricted universe  
multiverse  
deb http://run.hit.edu.cn/ubuntu/ jaunty-security main restricted universe  
multiverse  
deb-src http://run.hit.edu.cn/ubuntu/ jaunty-security main restricted universe  
multiverse
```

安装 Java5

```
sudo apt-get update  
sudo apt-get install sun-java5-jdk  
sudo cp /etc/apt/sources.list_backup /etc/apt/sources.list
```

最新的主线代码需要使用 Java6

```
sudo apt-get update  
sudo apt-get install sun-java6-jdk
```

如果同时安装了 Jvav5 和 Java6，可以配置指定要使用的 Java 版本

```
sudo update-alternatives --config java (slected java1.6)  
sudo update-alternatives --config javac
```

```
sudo update-alternatives --config jar
```

如果编译 doc 出现问题，可以设置 Java6 使用 Java5 的文档组件

```
cd /etc/alternatives
sudo mv javadoc.1.gz javadoc.1_bak.gz
sudo ln -s /usr/lib/jvm/java-1.5.0-sun/man/man1/javadoc.1.gz javadoc.1.gz
sudo mv javadoc javadoc_bak
sudo ln -s /usr/lib/jvm/java-1.5.0-sun/bin/javadoc javadoc
```

安装配置 Eclipse3.6.0

可以在 Eclipse 官方网站上下载安装包，也可以使用服务器上已经下载的安装包：

```
smb://ia-sz/product/android/eclipse-SDK-3.6-linux-get.tar.gz
```

解压到主目录的 eclipse 目录下：

```
tar zxvf eclipse-SDK-3.6-linux-get.tar.gz -C ~/
```

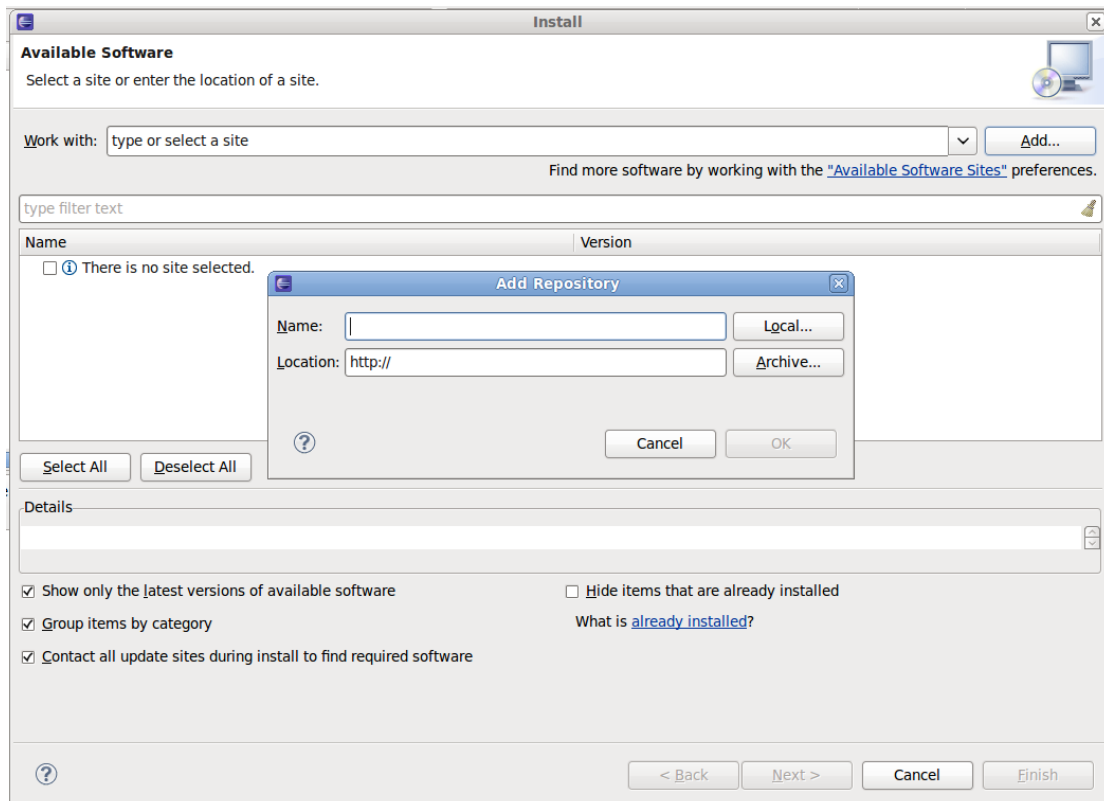
把解压后的路径添加到环境变量中：

```
export PATH=$PATH:$~/eclipse
```

安装配置 ADT-0.9.9

ADT 是 eclipse 的插件，用于把 Android 开发工具整合在 eclipse 的 UI 界面中。可以从 android 官方网站下载，也可以使用服务器上已经下载的安装包：

```
smb://ia-sz/product/android/ADT-0.9.9.zip
```



启动 eclipse，选择菜单 Help-->Install New Software，
点击[Add...]，弹出 Add Repository 对话框
在 name 文本框中输入：

```
Android Plugin
```

点击 location 文本框后面的按钮[Archive...]，选择 ADT-0.9.9.zip，点击[OK]

在 Install 对话框中选中 Developer Tools 复选框，点击[Next >]

选择接受开源协议后点击[Finish]

安装完成后可能需要重启 eclipse

安装 Android SDK

SDK 是 Android 的开发库，主要包含开发应用程序的 Java API 库，文档和一些工具。从官网上下载的 SDK：

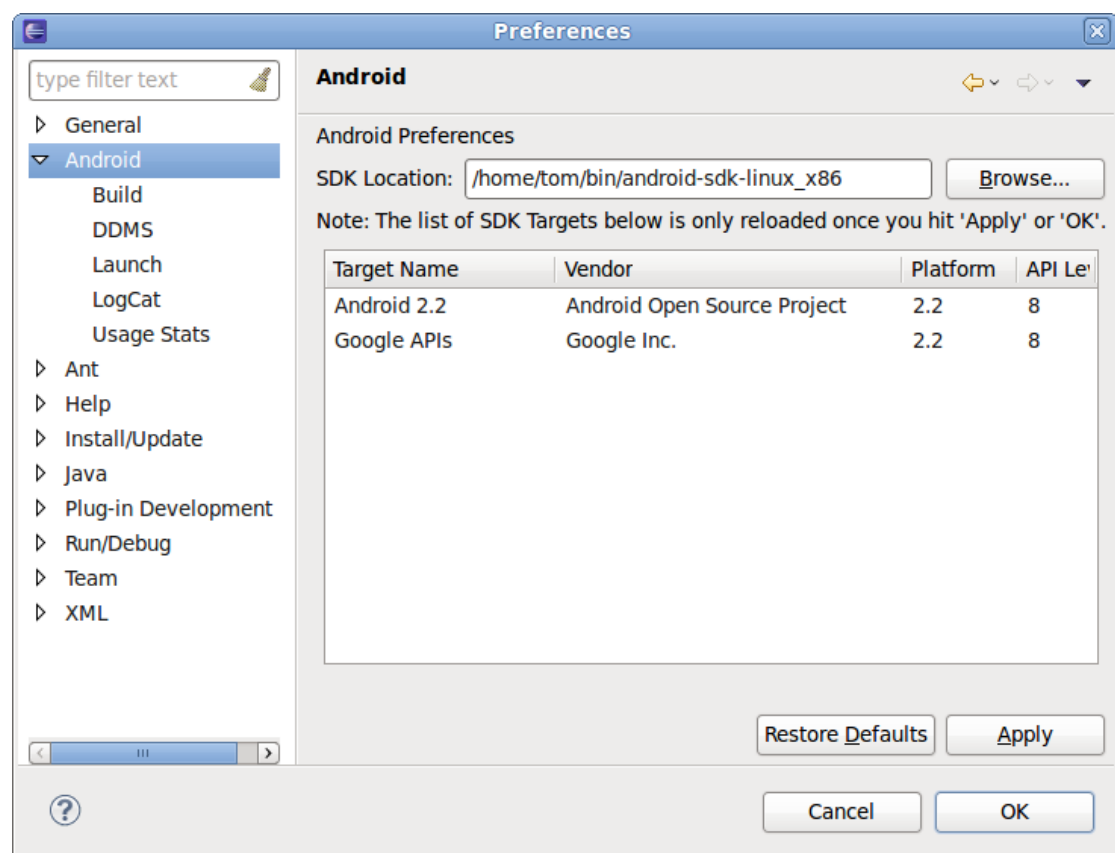
```
http://developer.android.com/sdk/index.html
```

解压 SDK 到~/android-sdk-linux_x86 目录

```
tar zxvf android-sdk_r07-linux-x86.tgz -C ~/
```

添加 tools 路径到 PATH 环境变量中：

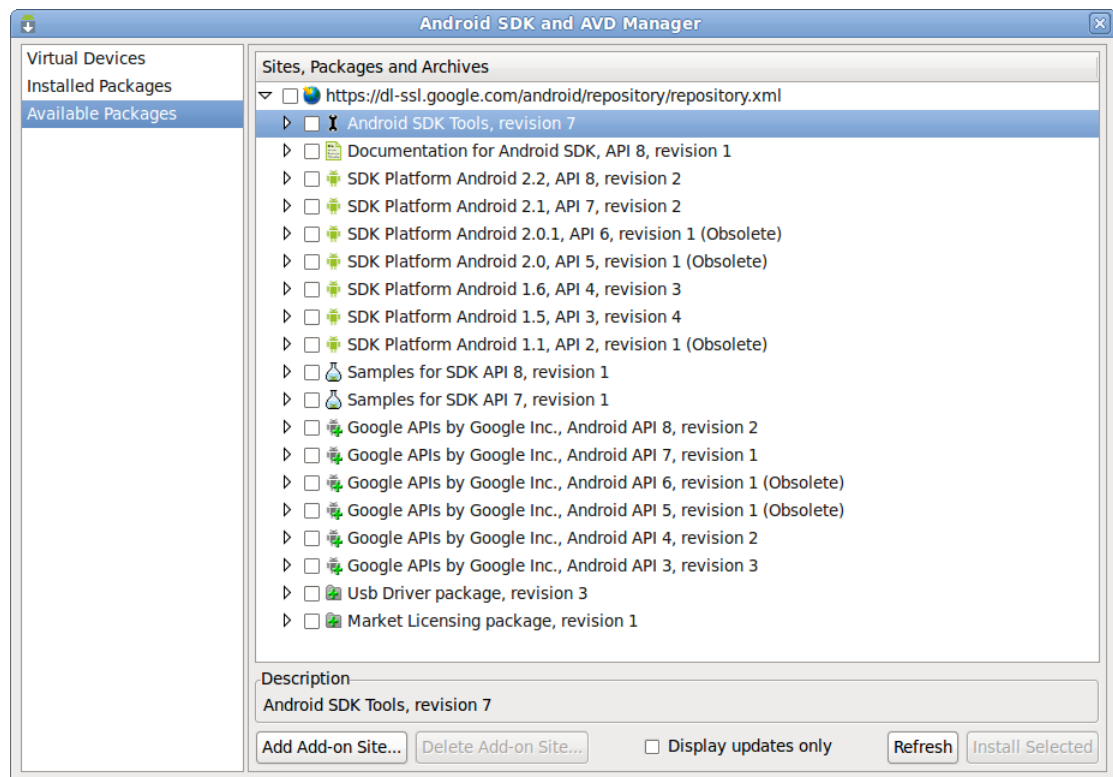
```
export PATH=$PATH:~/android-sdk-linux_x86/tools
```



启动 eclipse，选择菜单 Window->Preferences 打开 Preferences 对话框

选择级联菜单 Android，在右边 SDK Location 文本框中输入：android sdk 的路径，点击“OK”。

选择 eclipse 菜单 Window->Android SDK and AVD Manager 打开 Android SDK and AVD Manager 对话框，选择 Available Packages 项，展开右边的级联菜单，选中所有“API 8”及 SDK Tools 点击[Install Selected]安装。



安装配置 NDK

NDK 是 Android 提供的用于开发本地库的开发包，用于开发 c/c++程序库。从官方网站或服务器下载 android-ndk-r4b-linux-x86.zip

```
smb://ia-sz/product/android
http://developer.android.com/sdk/ndk/index.html
```

解压，把解压后的路径添加到环境变量中

```
tar zxvf android-ndk-r4b-linux-x86.zip -C ~/
export PATH=$PATH:~/android-ndk-r4b
```

安装 CDT

CDT 是 eclipse 插件，用于在 eclipse 环境开发 c/c++程序。从服务器上下载安装包

```
smb://ia-sz/product/android/cdt-master-7.0.0-I201006141710.zip
```

解压后，将 features 与 plugins 目录下的文件拷贝到 eclipse 的对应目录下。

1.3 编译 Android

设置交叉编译环境

切换到 android 源代码目录执行环境设置脚本

```
cd ~/android/master
. build/envsetup.sh
```

这个脚本会提供一组有用的命令，用 **help** 可以查看命令列表

设置 **build** 环境

```
Choosecombo
```

选择目标设备

```
Build for the simulator or the device?
1. Device
2. Simulator
Which would you like? [1]
```

选择 **debug** 级别

```
Build type choices are:
1. release
2. debug
Which would you like? [1]
```

选择平台

```
Which product would you like? [generic]
You can also type the name of a product if you know it.
```

选择目标

```
Variant choices are:
1. user
2. userdebug
3. eng
Which would you like? [eng]
```

编译 Android

完全编译，耗时 1 小时 25 分

```
$ make
```

编译当前目录下的模块，耗时 1 小时 31 分


```
mm
```

编译指定目录下的模块

```
mmm 模块的根目录
```

清除上次编译输出

```
make clean
```

单独编译模块生成 文件

```
make snod
```

编译 SDK，耗时大约 40 分钟

```
make sdk
```

1.4 工具

adb（官网下载的 sdk）

export PATH=\$PATH:<sdk 目录>/tools

- 查看设备

```
adb devices
```

- 安装软件,将指定的 apk 文件安装到设备上

```
adb install <apk 文件路径>
```

- 卸载软件, -k 参数,为卸载软件但是保留配置和缓存文件

```
adb uninstall -k <软件名>
```

- 执行一条 shell 命令

```
adb shell [command]
```

- 进入设备或模拟器的 shell

```
adb shell
```

- 端口转发

```
adb forward tcp:端口号 tcp:端口号
```

- 从电脑上发送文件到设备

```
adb push <本地路径> <远程路径>
```

- 从设备上下载文件到电脑

```
adb pull <远程路径> <本地路径>
```

- 查看 bug 报告

```
adb bugreport
```

- 使/system 目录可写

```
adb remount
```

- 获取设备的序列号

```
adb get-serialno
```

- 显示应用程序的调试信息

```
adb logcat
```

android

- 列出模拟器类型

```
android -h 或 android -help
```

- 启动模拟器管理器

```
android
```

- 列出模拟器类型

```
android list targets
```

-
- 建立模拟器, id 为模拟器类型的,YouAvdName 为新建模拟器的名字

```
android create avd -t id -n YouAvdName
```

- 列出已经建立的模拟器

```
android list avd
```

- 改变模拟器的皮肤

```
android create avd -t id -n YouAvdName --skin QVGA
```

- 删除模拟器

```
android delete avd --name YouAvdName
```

1.5 运行

启动模拟器

```
cd out/target/product/generic
```

配置运行模拟器

在 eclipse 中使用 adnroid 创建模拟器:

```
Window-->android sdk and avd manager--->new  
Name:android_avd  
Target:Android2.2 -API Level 8
```

直接用 android 创建模拟器:

```
android list targets  
android create avd -t id -n You_Avd_Name
```

启动模拟器

根据模拟器名字启动

```
emulator @YouAvdName  
emulator -avd YouAvdName
```

使用指定 img 文件启动模拟器

```
emulator -system system.img -data userdata.img -ramdisk ramdisk.img
```

1.6 调试

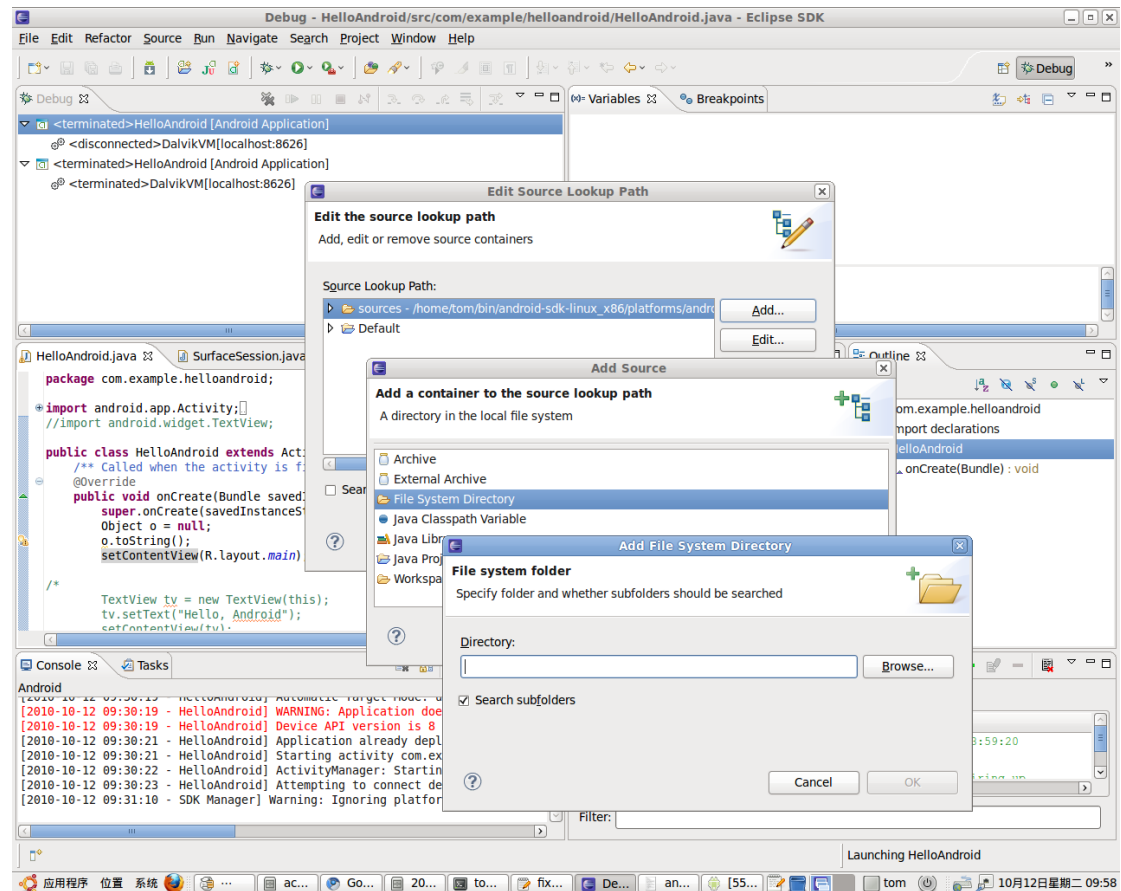
配置调试 Application 环境

为了让 eclipse 找到 Java API 的定义, 执行下面的脚本把 Java 源文件放到 SDK 路径下, 脚本根据 Java 包建立目录, 将源文件复制到 android-sdk-linux_x86/platforms/andoid-2.2/sources

目录中（此处请自行更改为绝对路径）：

```
./fix_android_sdk.py ~/android/master ~/android-sdk-linux_x86
```

在 eclipse 配置调试环境，使其能找到源代码：



启动 eclipse-->new-->Android project, 选上” create project from existing source”,然后 browse to NDK_ROOT/samples/hello-jin/---->finish.

在 eclipse 切换到 Debug 模式，在 debug 标签页中右键点击

选择弹出菜单项 edit source lookup

在 Edit Source Lookup Path 对话框中点击[Add..]

选择 File System Directory 源

在 Directory 文本框中填入~/android-sdk-linux_x86/platforms/android-2.2/sources（绝对路径）

此后 debug 的时候可以在断点处看到 API 调用栈，并且可以看到 Java 库的实现代码

创建和调试 Library（JNI）

创建 android project

创建目录 src，存放 java code

创建 jni 目录，存放.c 和.mk file

在 AndroidManifest.xml 的 Application 中加上 android:debuggable="true"

编译项目

```
ndk-build
```

开启虚拟机后，安装

```
adb install ../bin/project.apk
```

调试

```
ndk-gdb
```