1、实验要求

- 学学习java和android环境搭建（主要是后者）
- 学习并掌握adb命令
- 学习并掌握安卓自动化测试（monkey测试）
- 学习使用aapt获取报名

2、实验过程

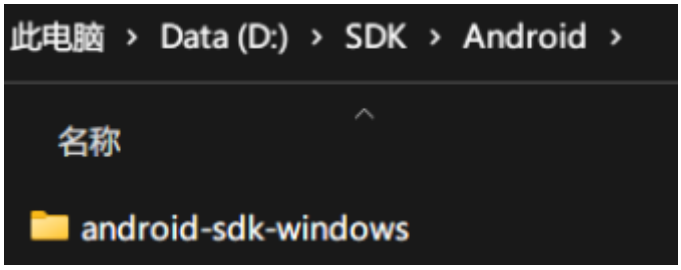
2.1 环境搭建

2.1.1 配置Java环境

已经配置过，略

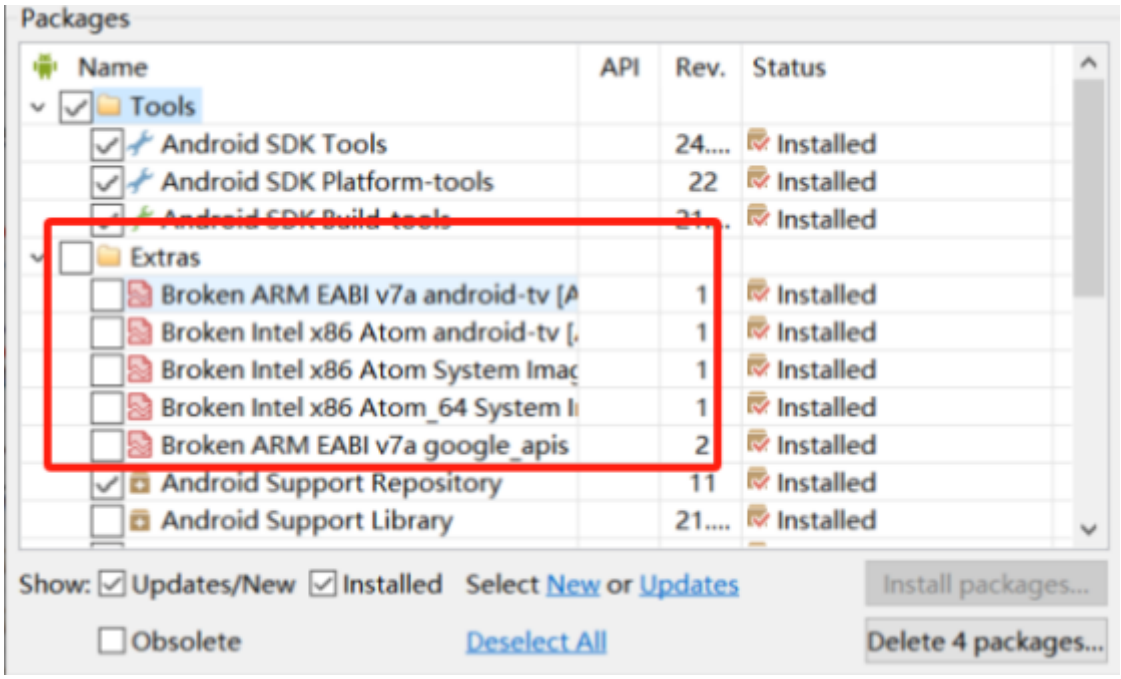
2.1.2 配置Android sdk环境

（1）依据实验要求解压android-sdk-windows文件，解压位置如下图所示：

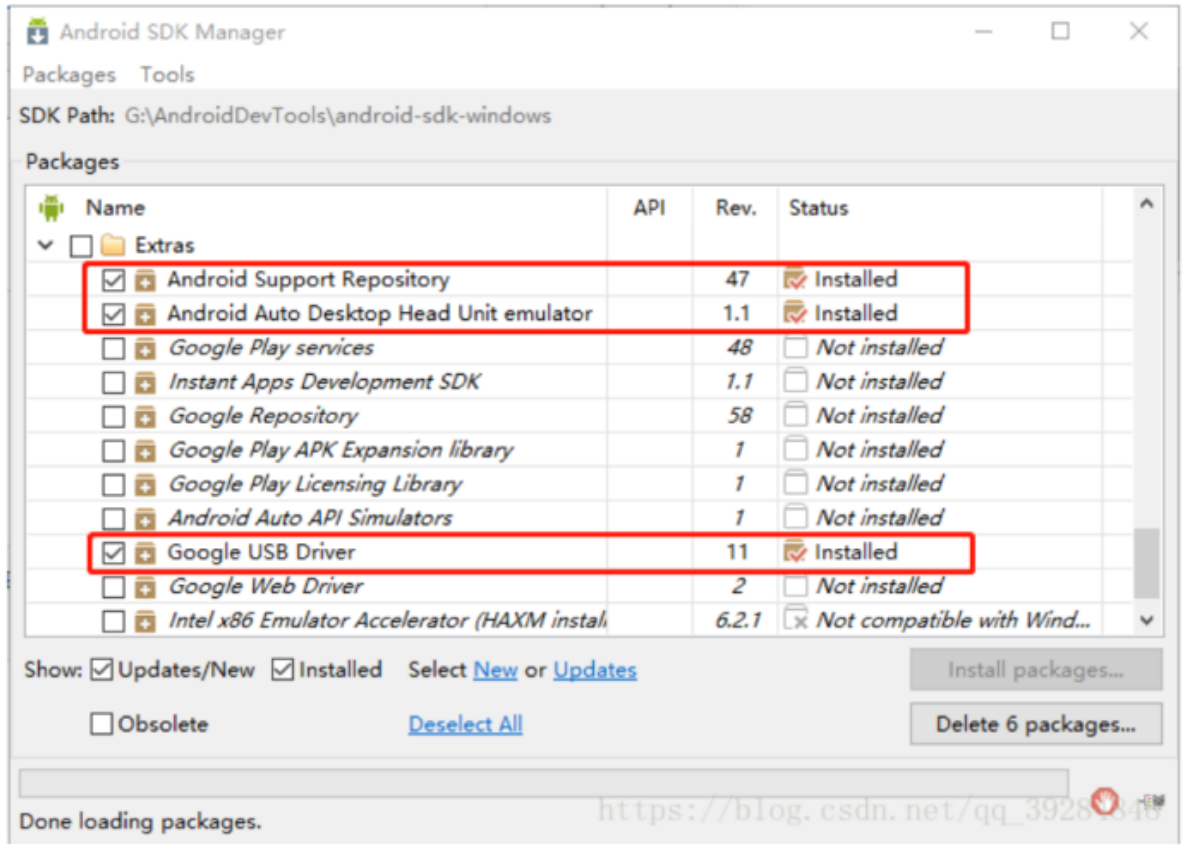


（2）双击“SDK Manager.exe”，启动SDK Manager，对照教程的说明进，发现extras包下面少了一些选项，如图所示：

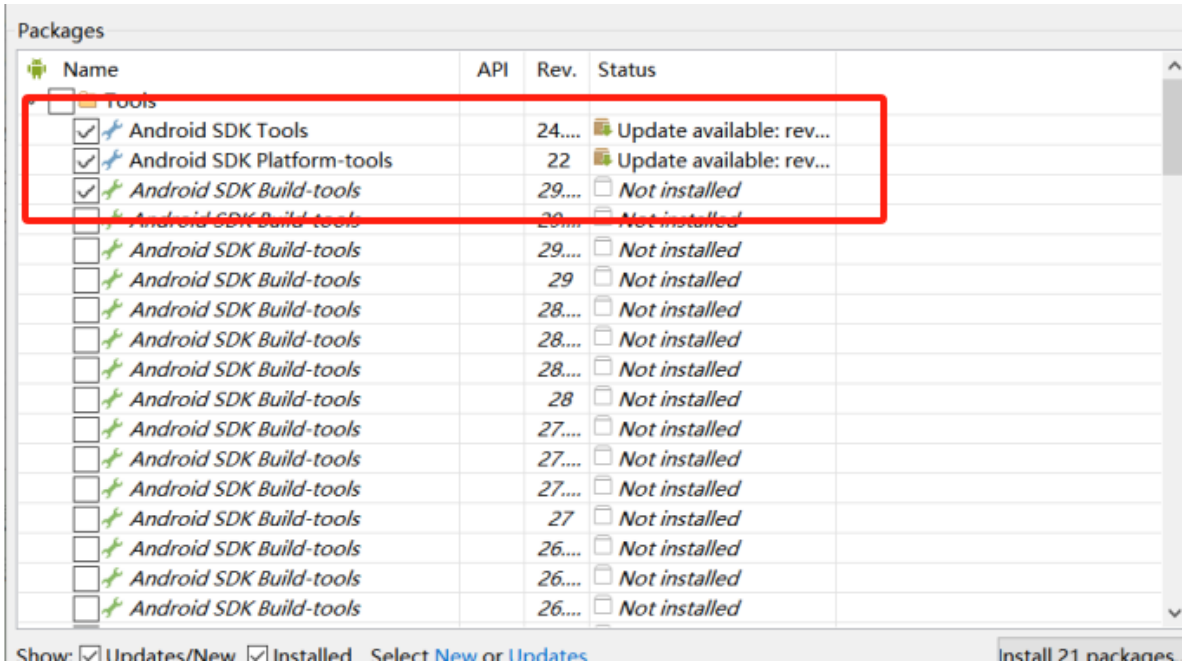
我的SDK Manager：

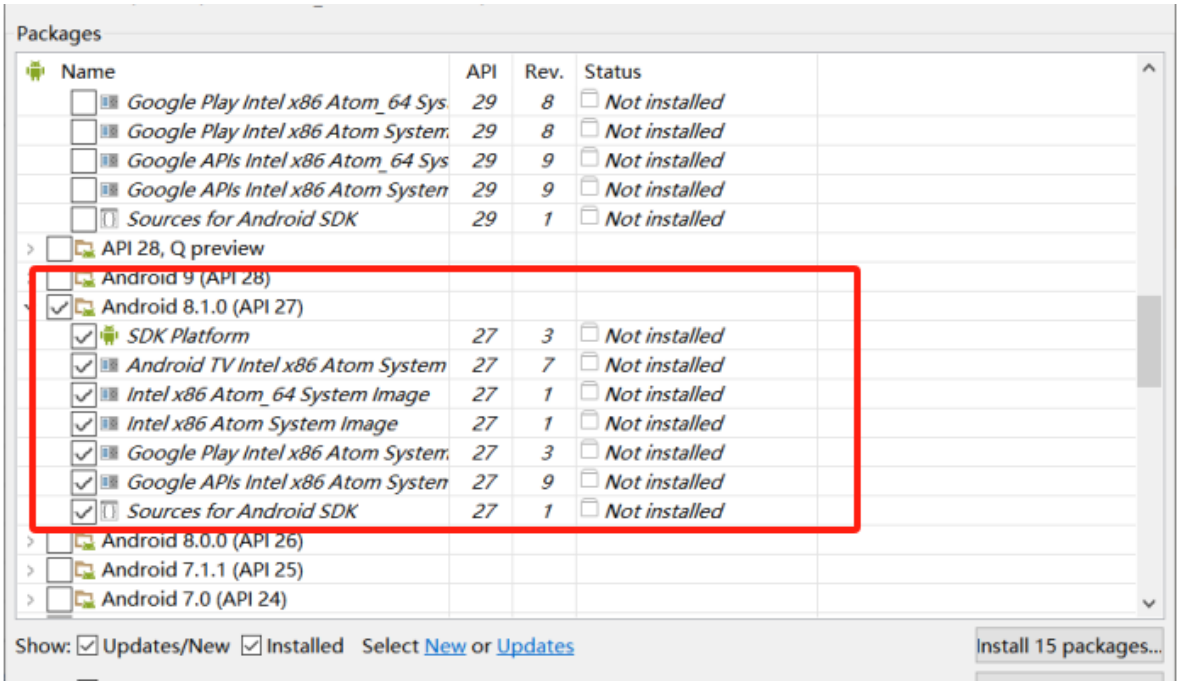


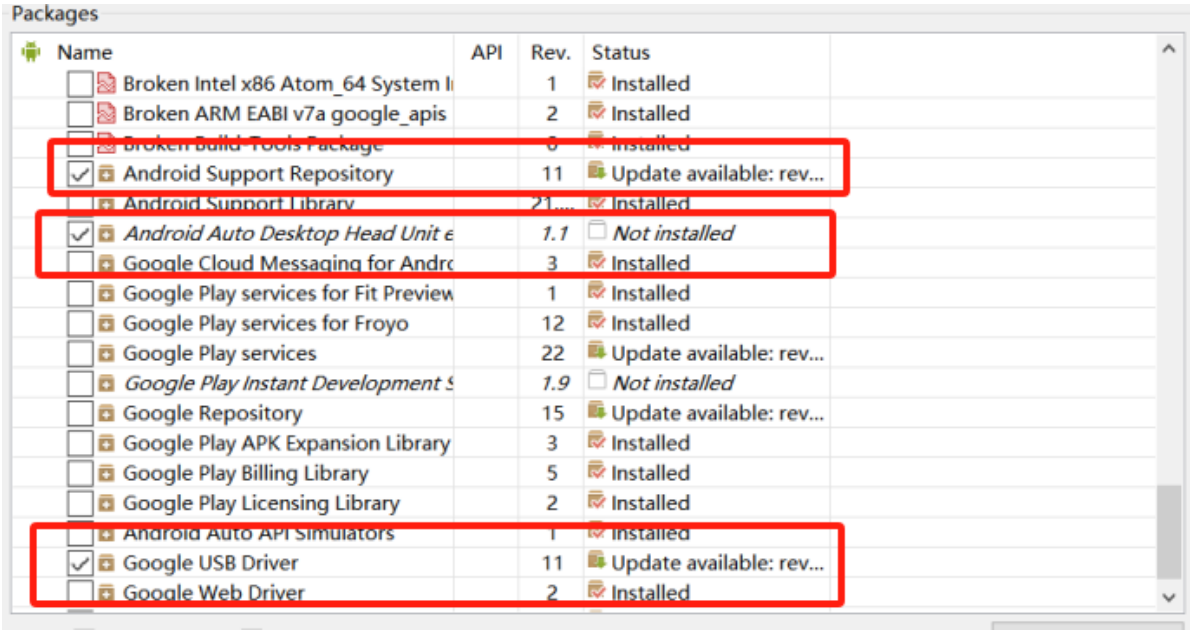
教程的SDK Manager：



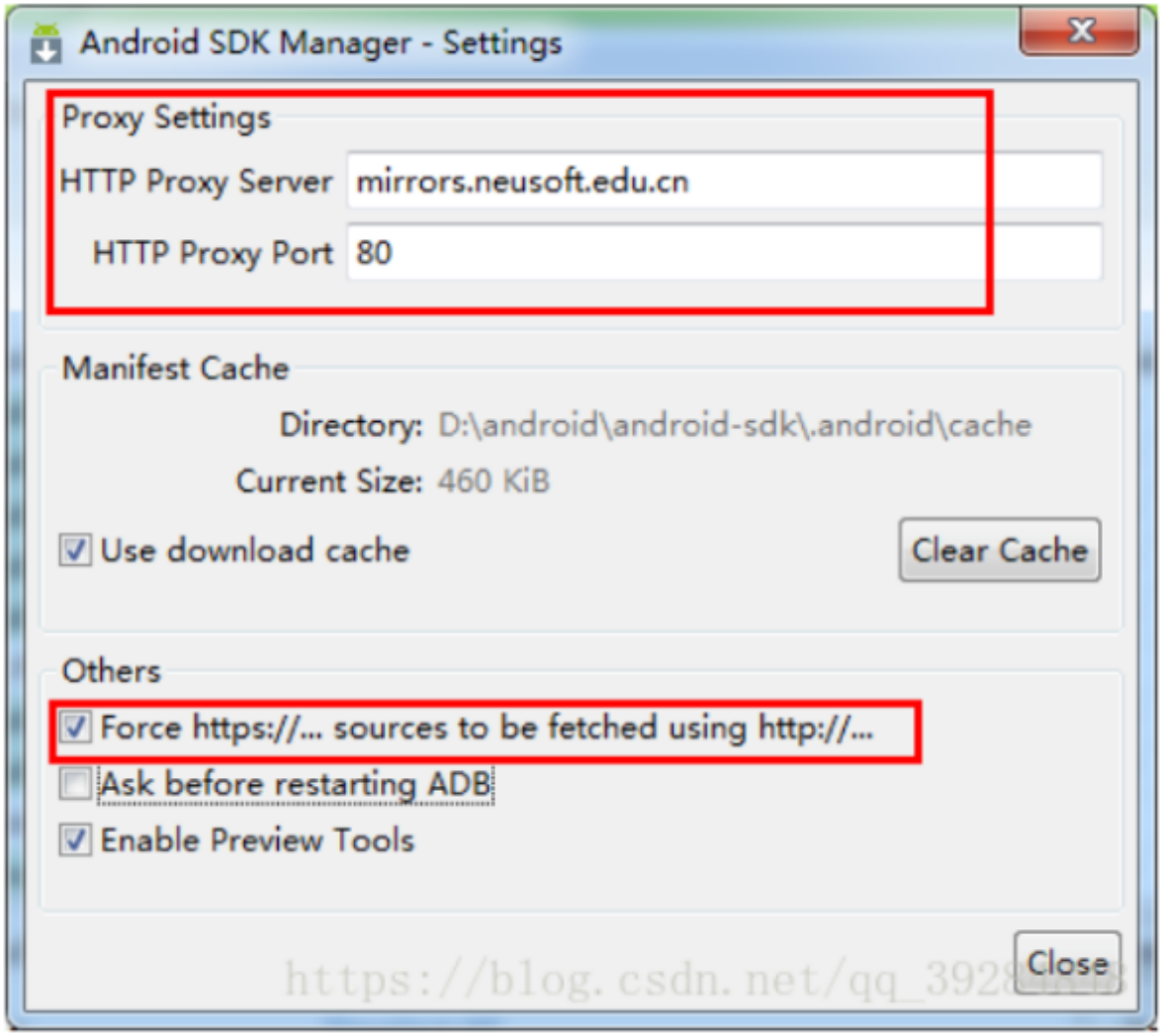
（3）为了确保后续实验的顺利，安装选项应该尽量和实验说明保持一致。我查阅网上大量资料，发现可能是国内镜像网站的问题，于是尝试挂vpn下载，这次的SDK Manager中选项全了，按照教程说明进行安装：







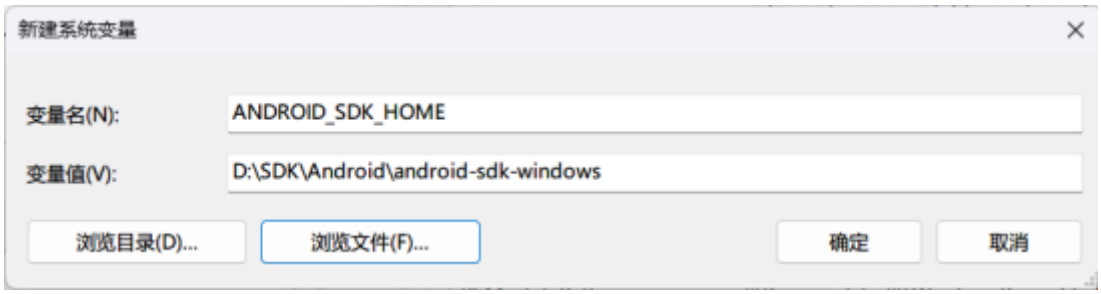
（4）然后在弹出的对话框中，填写HTTP Proxy Server为mirrors.neusoft.edu.cn（镜像服务器的地址，注意前面不要加http），然后填写HTTP Proxy Port为80 （端口号）。最后在勾选下面的『Forcehttps://... sources to be fetched using http://...』复选框，如下图所示：



（5）漫长的下载安装过程，安装成功。

2.1.3 配置Android SDK环境变量

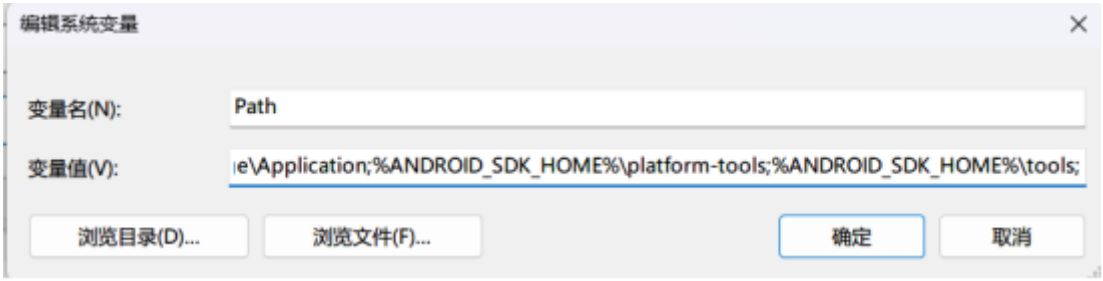
（1）新建一个系统环境变量，变量名为ANDROID_SDK_HOME，变量值为SDK安装路径，这里我的安装路径为D: \SDK\Android\android-sdk-windows，如图所示：



（2）然后就是在系统的Path变量后，追加：

- %ANDROID_SDK_HOME%\platform-tools
- %ANDROID_SDK_HOME%\tools

  如图所示：



（3）在终端输入命令：android -h和adb，测试tools和platform-tools配置是否成功，如图所示：

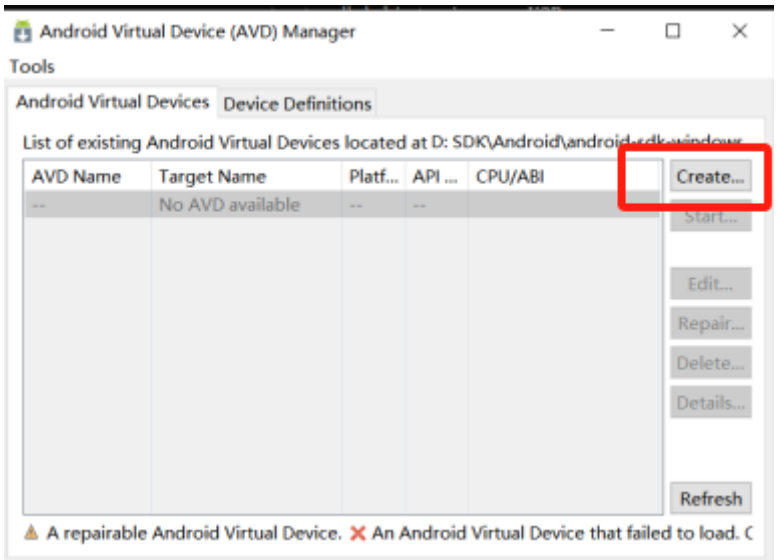
```
PS C:\Users\Meteorite> android -h

       Usage:
       android [global options] action [action options]
       Global options:
  -s --silent      : Silent mode, shows errors only.
  -v --verbose     : Verbose mode, shows errors, warnings and all messages.
     --clear-cache: Clear the SDK Manager repository manifest cache.
  -h --help        : Help on a specific command.

                                                                  Valid
                                                                  actions
                                                                  are
                                                                  composed
                                                                  of a verb
                                                                  and an
                                                                  optional
                                                                  direct
                                                                  object:
-    sdk              : Displays the SDK Manager window.
-    avd              : Displays the AVD Manager window.
-   list              : Lists existing targets or virtual devices.
-   list avd          : Lists existing Android Virtual Devices.
```

到此为止，Android SDK环境配置成功，可以使用命令行工具进行测试

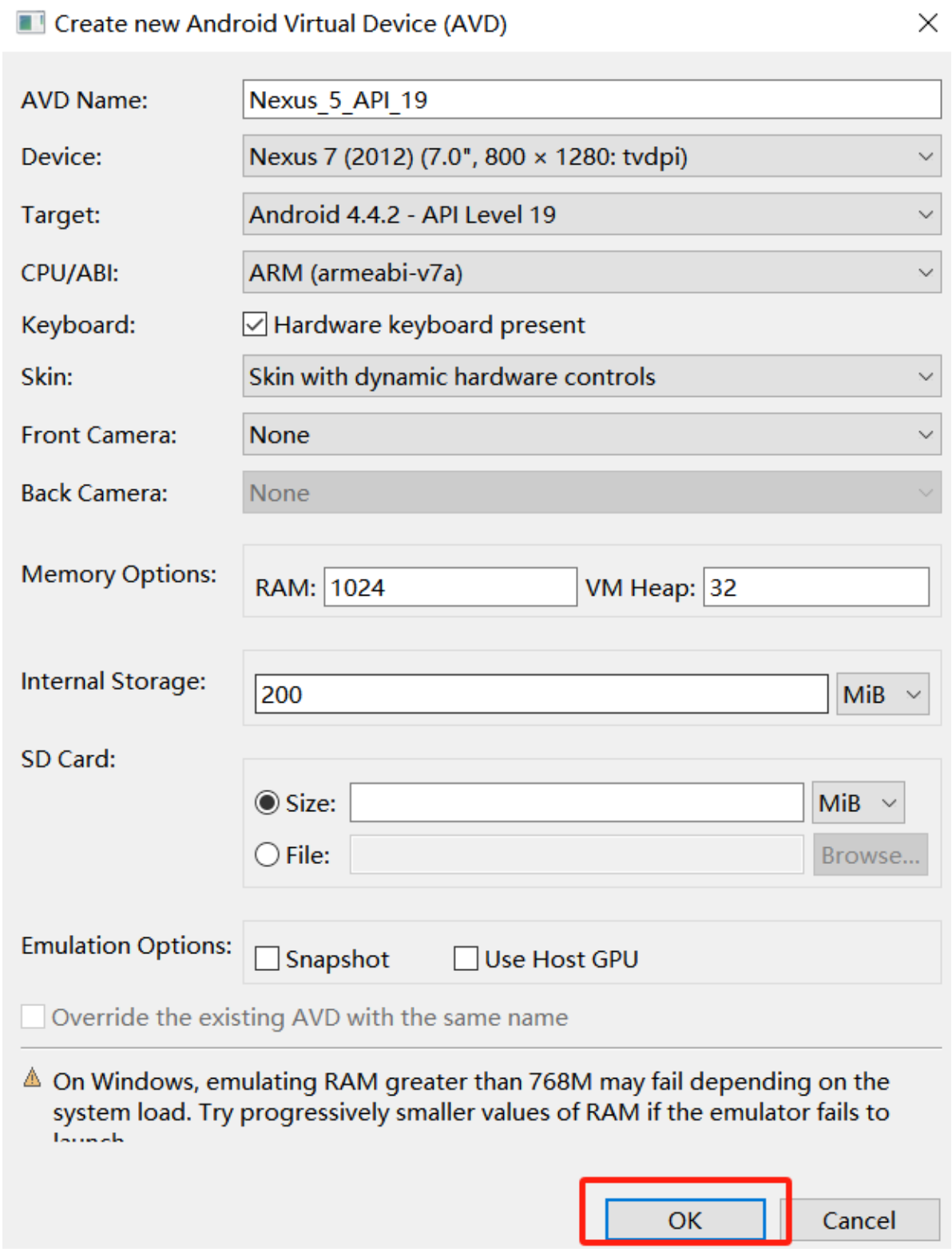
2.2 安卓自动化测试

2.2.1 创建并启动Android虚拟机

（1）在终端输入命令：android avd，弹出如下界面，点击Create按钮：



（2）配置如下格式的模拟器，点击ok，如图所示：



（3）由于模拟器在我的电脑上运行速度太慢了，所以换用真机的方式。按照教程上更推荐的Android真机方式，将征用的舍友的安卓手机连接到电脑上，然后开启开发者模式，打开USB调试功能，如图所示：

（4）然后在cmd输入：adb devices，确定设备是否连接上，此时应出现该手机的udid号，则SDK安装配置成功，如下图所示：

```
PS C:\Users\Meteorite> adb devices
List of devices attached
a613c407        device
```

2.2.2 安装app到模拟器或者手机

> 命令格式：adb install -r 路径+包名.apk　# -r参数表示重新安装apk，可不使用

在实验材料\apk文件夹下，从中选择任意一个apk作为实验对象，按照格式输入命令，看到命令行中出现Success提示，手机里出现了对应的app：

```
PS C:\Users\Meteorite> adb install -r D:\Users\Documents\test-apks\com.addi_44.apk
3588 KB/s (21379082 bytes in 5.818s)
Success
```

2.2.3 aapt 命令查看apk包名、主activity、版本等信息

（1）根据教程找到aapt的位置，我的appt在D: \SDK\Android\android-sdk-windows，进入这个文件夹（因为没有将aapt加入到环境变量中，所以通过这种方式）

```
PS D:\SDK\Android\android-sdk-windows\build-tools\33.0.0> .
```

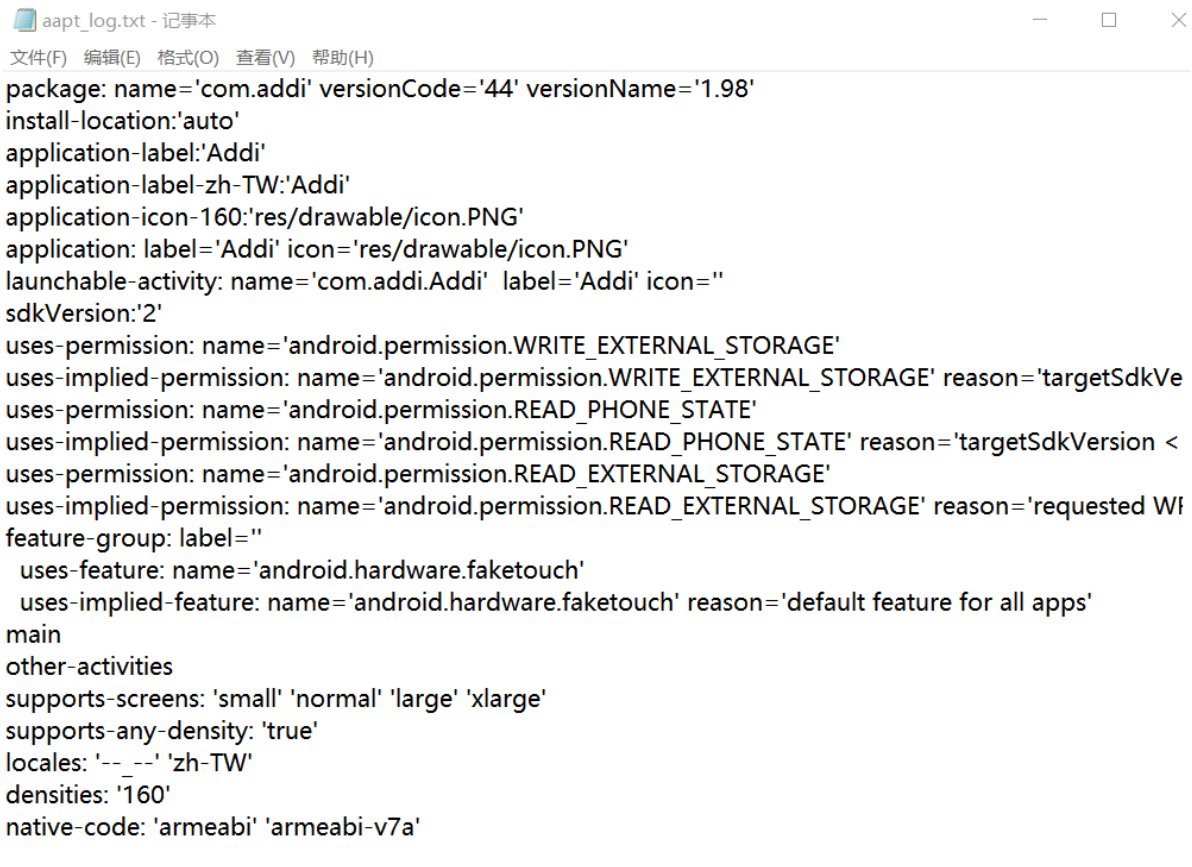
（2）使用aapt 命令查看apk包名、主activity、版本等信息

> 命令格式：aapt[空格]dump[空格]badging[空格]APK文件

```
PS D:\SDK\Android\android-sdk-windows\build-tools\33.0.0> .\aapt dump badging D:\Users\Documents\test-apks\com.addi_44.a
pk
package: name='com.addi' versionCode='44' versionName='1.98'
install-location:'auto'
application-label:'Addi'
application-label-zh-TW:'Addi'
application-icon-160:'res/drawable/icon.PNG'
application: label='Addi' icon='res/drawable/icon.PNG'
launchable-activity: name='com.addi.Addi'  label='Addi' icon=''
sdkVersion:'2'
uses-permission: name='android.permission.WRITE_EXTERNAL_STORAGE'
uses-implied-permission: name='android.permission.WRITE_EXTERNAL_STORAGE' reason='targetSdkVersion < 4'
uses-permission: name='android.permission.READ_PHONE_STATE'
uses-implied-permission: name='android.permission.READ_PHONE_STATE' reason='targetSdkVersion < 4'
uses-permission: name='android.permission.READ_EXTERNAL_STORAGE'
uses-implied-permission: name='android.permission.READ_EXTERNAL_STORAGE' reason='requested WRITE_EXTERNAL_STORAGE'
feature-group: label=''
  uses-feature: name='android.hardware.faketouch'
  uses-implied-feature: name='android.hardware.faketouch' reason='default feature for all apps'
main
other-activities
supports-screens: 'small' 'normal' 'large' 'xlarge'
supports-any-density: 'true'
locales: '--_--' 'zh-TW'
densities: '160'
native-code: 'armeabi' 'armeabi-v7a'
```

（3）保存运行日志，将信息保存为txt文件：

```
PS D:\SDK\Android\android-sdk-windows\build-tools\33.0.0> .\aapt dump badging D:\Users\Documents\test-apks\com.addi_44.a
pk > D:\aapt_log.txt
```



2.2.4 采用monkey测试待测app

（1）查阅资料，学习Monkey常用命令：

-p <允许的包名列表>

- 用此参数指定一个或多个包。指定包之后，monkey将只允许系统启动指定的app。如果丌指定包， monkey将允许系统启动设备中的所有app。
- 指定一个包：adb shell monkey -p com.shjt.map 100
- 指定多个包：adb shell monkey -p fishjoy.control.menu –p com.shjt.map 100

-v

- 用于指定反馈信息级别（信息级别就是日志的详细程度），总共分3个级别，分别对应的参数如下表所示：
- Level 0 : adb shell monkey -p com.shjt.map -v 100　　// 缺省值，仅提供启动提示、测试完成和最终结果等少量信息
- Level 1 : adb shell monkey -p com.shjt.map -v -v 100　// 提供较为详细的日志，包括每个发送到Activity的事件信息
- Level 2 : adb shell monkey -p com.shjt.map -v -v -v 100　// 最详细的日志，包括了测试中选中/未选中的Activity信息

-s（随机数种子）

- 用于指定伪随机数生成器的seed值，如果seed相同，则两次Monkey测试所产生的事件序列也相同的。
- 示例：monkey测试1：adb shell monkey -p com.shjt.map –s 10 100

  monkey测试2：adb shell monkey -p com.shjt.map –s 10 100

--throttle <毫秒>

- 用于指定用户操作（即事件）间的时延，单位是毫秒；如果丌指定这个参数，monkey会尽可能快的 生成和发送消息。
- 示例：adb shell monkey -p com.shjt.map --throttle 3000 100

（2）采用不同的monkey命令测试待测app

```
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 5 -v -v -v 2000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi -v -v -v 2000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --pct-touch 30 -v -v -v 2000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --pct-touch 30 -v -v -v 2000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --pct-touch 30 -v -v -v 2000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 2 --pct-touch 20 -v -v -v 2000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 2 --pct-touch 20 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 2 --pct-touch 20 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1  -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi -s 12312 --throttle 1  -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1 --pct-monition 20 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1 --pct-trackball 20 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1 --pct-nav 25 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1 --pct-majornav 20 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1 --pct-syskeys 20 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1 --pct-appswtich 20 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite> adb shell monkey -p com.addi --throttle 1 --pct-anyevent 30 -v -v -v 10000 > D:\test_log.txt
PS C:\Users\Meteorite>
```

（3）在日志里找到一些IOException，如下图：

```
  :Sending Trackball (ACTION_MOVE): 0:(2.0,0.0)
  :Sending Trackball (ACTION_MOVE): 0:(-3.0,0.0)
  :Sending Trackball (ACTION_MOVE): 0:(-3.0,4.0)
  :Sending Trackball (ACTION_MOVE): 0:(4.0,-4.0)
  :Sending Trackball (ACTION_MOVE): 0:(4.0,-5.0)
  :Sending Trackball (ACTION_MOVE): 0:(-3.0,2.0)
  :Sending Trackball (ACTION_MOVE): 0:(4.0,2.0)
  :Sending Trackball (ACTION_MOVE): 0:(4.0,1.0)
  :Sending Trackball (ACTION_MOVE): 0:(-4.0,-4.0)
  :Sending Trackball (ACTION_MOVE): 0:(0.0,1.0)
  :Sending Flip keyboardOpen=true
  Got IOException performing flipjava.io.IOException: write failed: EINVAL (Invalid argument)
     // Injection Failed
```

（4）将测试日志以txt格式存储到指定文件夹下：

test_log.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
  bash arg: -p
  bash arg: com.addi
  bash arg: --throttle
  bash arg: 1
  bash arg: -v
  bash arg: -v
  bash arg: -v
  bash arg: 10000
args: [-p, com.addi, --throttle, 1, -v, -v, -v, 10000]
 arg: "-p"
 arg: "com.addi"
 arg: "--throttle"
 arg: "1"
 arg: "-v"
 arg: "-v"
 arg: "-v"
 arg: "10000"
data="com.addi"
arg="--throttle" mCurArgData="null" mNextArg=3 argwas="--throttle" nextarg="1"
data="1"
:Monkey: seed=1667061761489 count=10000
:AllowPackage: com.addi
:IncludeCategory: android.intent.category.LAUNCHER
:IncludeCategory: android.intent.category.MONKEY
// Selecting main activities from category android.intent.category.LAUNCHER
//   - NOT USING main activity com.android.calendar.homepage.AllInOneActivity (from package com.an
//   - NOT USING main activity com.android.camera.Camera (from package com.android.camera)
//   - NOT USING main activity com.android.contacts.activities.PeopleActivity (from package com.androi
//   - NOT USING main activity com.android.contacts.activities.TwelveKeyDialer (from package com.andr
//   - NOT USING main activity com.android.deskclock.DeskClockTabActivity (from package com.androic
//   - NOT USING main activity com.android.fileexplorer.FileExplorerTabActivity (from package com.andı
```

2.2.5 卸载所装的app

使用adb uninstall com.addi卸载所装app，如图所示：

```
PS C:\Users\Meteorite> adb uninstall com.addi
Success
```

命令行中显示卸载成功，并看到手机上刚才所安装的app已被卸载。

3、实验分析

3.1 问题1：文件路径名太长导致解压失败

本次实验刚开始就碰到了困难，我刚开始在D盘已经创建好的文件夹下解压extras文件时，解压软件一直报错说路径太长，后来换了个路径解压之后在复制到原来的路径下，解决了这个问题。

3.2 问题2：sdk manager程序的extras包中缺少一些选项

在解压之后打开sdk manager程序发现extras包下面少了一些选项，后来又试着从镜像网站里下载了最新的版本，发现还是缺项，因为担心环境配置不正确会导致后续实验无法进行，没有贸然安装。在这里卡了好久，最后看到群里有同学说挂vpn下载，就试了一下，虽然还是缺几项，但是不影响安装了。

4、实验总结

这是第一次做移动测试的实验，刚开始做确实很懵，不知道如何下手，之后仔细看实验指导书，完成了本次实验。

本次实验主要的难点在Android SDK环境的搭建，环境搭建中有些问题牵扯的范围很广，出现问题时很难发现问题的真正原因和解决方法。不过好在我这次实验中遇到的问题不是疑难杂症，有不少同学也遇到了类似的，也从侧面说明实验文档需要更新了，通过和同学交流讨论，最终成功完成实验。

此外，我刚开始尝试用模拟器进行试验，但是它真的太慢了。后来像舍友借了一部安卓手机，尝试实验手册推荐的真机，速度快了很多，也方便一些。

在环境配置完成之后，剩下的命令操作简单很多。在测试过程中，我学习到了如何使用adb的常用命令、如何使用aapt查看包名以及版本等相关信息，还有monkey测试的指令格式，以及monkey自动化测试在用户指令的要求下生成一些相关的操作。