[**Android 常用 adb 命令总结**](https://www.cnblogs.com/bravesnail/articles/5850335.html)

针对移动端 Android 的测试， adb 命令是很重要的一个点，必须将常用的 adb 命令熟记于心， 将会为 Android 测试带来很大的方便，其中很多命令将会用于自动化测试的脚本当中。

**Android Debug Bridge**

adb 其实就是 Android Debug Bridge, Android 调试桥的缩写，adb 是一个 *C/S* 架构的命令行工具，主要由 3 部分组成：

* 运行在 PC 端的 Client : 可以通过它对 Android 应用进行安装、卸载及调试

Eclipse 中的 ADT、SDK Tools 目录下的 DDMS、Monitor 等工具，都是同样地用到了 adb 的功能来与 Android 设备进行交互。  
  
PC 端的手机助手，诸如 360 手机助手、豌豆荚、应用宝等，其除了安装第三方应用方便，其他的功能，基本上都可以通过 adb 命令去完成，这里建议测试人员尽量不要在电脑上安装这类手机助手，因为其自带的 adb 程序可能会与 Android SDK 下的 adb 程序产生冲突，5037 端口被占用，导致使用 adb 命令时无法连接到设备

* 运行在 PC 端的 Service : 其管理客户端到 Android 设备上 adb 后台进程的连接

adb 服务启动后，Windows 可以在任务管理器中找到 adb.exe 这个进程

* 运行在 Android 设备上的 adb 后台进程

执行 adb shell ps | grep adbd ，可以找到该后台进程，windows 请使用 findstr 替代 grep

[xuxu:~]$ adb shell ps | grep adbd

root 23227 1 6672 832 ffffffff 00019bb4 S /sbin/adbd

这里注意一个地方，就是 adb 使用的端口号，5037，有必要记一下  
  
接下来我将 adb 命令分为三部分进行介绍，adb 命令、adb shell 命令、linux 命令

**adb 命令**

在开发或者测试的过程中，我们可以通过 adb 来管理多台设备，其一般的格式为：

adb [-e | -d | -s <设备序列号>] <子命令>

在配好环境变量的前提下，在命令窗口当中输入 adb help 或者直接输入 adb ，将会列出所有的选项说明及子命令。

这里介绍一些里面常用的命令：

* adb devices , 获取设备列表及设备状态
* [xuxu:~]$ adb devices
* List of devices attached

44c826a0 device

* adb get-state , 获取设备的状态
* [xuxu:~]$ adb get-state

device

设备的状态有 3 钟，device , offline , unknown  
  
device：设备正常连接  
  
offline：连接出现异常，设备无响应  
  
unknown：没有连接设备

* adb kill-server , adb start-server , 结束 adb 服务， 启动 adb 服务，通常两个命令一起用

一般在连接出现异常，使用 adb devices 未正常列出设备， 设备状态异常时使用 kill-server，然后运行 start-server 进行重启服务

* adb logcat , 打印 Android 的系统日志，这个可以单独拿出来讲
* adb bugreport , 打印dumpsys、dumpstate、logcat的输出，也是用于分析错误

输出比较多，建议重定向到一个文件中

adb bugreport > d:\bugreport.log

* adb install , 安装应用，覆盖安装是使用 -r 选项

windows 下如果需要安装含有中文名的 apk ，需要对 adb 进行修改，百度可以找到做出修改的adb , 支持中文命令的 apk，请自行搜索

* adb uninstall , 卸载应用，后面跟的参数是应用的包名，请区别于 apk 文件名

'-k' means keep the data and cache directories , -k 选项，卸载时保存数据和缓存目录

* adb pull , 将 Android 设备上的文件或者文件夹复制到本地  
    
  例如复制 Sdcard 下的 pull.txt 文件到 D 盘：

adb pull sdcard/pull.txt d:\

如果需要重命名为 rename.txt：

adb pull sdcard/pull.txt d:\rename.txt

注意权限，复制系统权限的目录下的文件，需要 root ，并且一般的 Android 机 root 之后并不能使用命令去复制，而需要在手机上使用类似于 RE 的文件浏览器，先对系统的文件系统进行挂载为可读写后，才能在手机上复制移动系统文件，这里推荐使用小米手机的开发版本，IUNI 也是不错滴~~

* adb push , 推送本地文件至 Android 设备  
    
  例如推送 D 盘下的 push.txt 至 Sdcard：

adb push d:\push.txt sdcard/

sdcard 后面的斜杠不能少，否则会出现下面的错误：

[xuxu:~]$ adb push push.txt sdcard

failed to copy 'push.txt' to 'sdcard': Is a directory

权限问题同 pull 命令

* adb root , adb remount, 只针对类似小米开发版的手机有用，可以直接已这两个命令获取 root 权限，并挂载系统文件系统为可读写状态
* adb reboot , 重启 Android 设备

bootloader , 重启设备，进入 fastboot 模式，同 adb reboot-bootloader 命令  
  
recovery , 重启设备，进入 recovery 模式，经常刷机的同学比较熟悉这个模式

* adb forward , 将 宿主机上的某个端口重定向到设备的某个端口

adb forward tcp:1314 tcp :8888

执行该命令后所有发往宿主机 1314 端口的消息、数据都会转发到 Android 设备的 8888 端口上，因此可以通过远程的方式控制 Android 设备。

* adb connect 远程连接 Android 设备

手机、PC处于相同的网络下，手机 root ，安装应用 adbWireless ，启动应用后点击界面中间的按钮：

[](http://i2.buimg.com/73b185a561f2c622.jpg)

接着运行 adb connect 192.168.1.102 , 即可通过无线的方式连接手机，缺点是速度比较慢

**adb shell 命令**

有人问过我，为什么会知道这么多的命令，答案就是我比较爱折腾，这里大家先要了解我为什么要区分 adb 命令和 adb shell 命令 。  
简单点讲，adb 命令是 adb 这个程序自带的一些命令，而 adb shell 则是调用的 Android 系统中的命令，这些 Android 特有的命令都放在了 Android 设备的 system/bin 目录下，例如我再命令行中敲这样一个命令：

[xuxu:~]$ adb shell hehe

/system/bin/sh: hehe: not found

很明显，在 bin 目录下并不存在这个命令。  
自己爱折腾，想看看有哪些命令，也不想去找文档，于是就启动模拟器，将整个 system/bin 目录复制了出来，然后一个一个的去试。。囧~~  
  
打开这些文件就可以发现，里面有些命令其实是一个 shell 脚本，例如打开 monkey 文件：

# Script to start "monkey" on the device, which has a very rudimentary

# shell.

#

base=/system

export CLASSPATH=$base/framework/monkey.jar

trap "" HUP

exec app\_process $base/bin com.android.commands.monkey.Monkey $\*

再比如打开 am：

#!/system/bin/sh

#

# Script to start "am" on the device, which has a very rudimentary

# shell.

#

base=/system

export CLASSPATH=$base/framework/am.jar

exec app\_process $base/bin com.android.commands.am.Am "$@"

还有 SDK sources/android-20/com/android/commands 目录下：

[xuxu:...oid-20/com/android/commands]$ pwd

/Users/xuxu/utils/android/android-sdk-macosx/sources/android-20/com/android/commands

[xuxu:...oid-20/com/android/commands]$ ll

total 0

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 am

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 bmgr

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 bu

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 content

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 ime

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 input

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 media

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 pm

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 requestsync

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 settings

drwxr-xr-x 7 xuxu staff 238B 4 2 10:57 svc

drwxr-xr-x 6 xuxu staff 204B 4 2 10:57 uiautomator

drwxr-xr-x 3 xuxu staff 102B 4 2 10:57 wm

有没有熟悉的命令？ am 、pm、uiautomator ...

下面介绍一些常用的 adb shell 命令 （其中pm、am 命令比较庞大，使用四级标题）

**pm**

Package Manager , 可以用获取到一些安装在 Android 设备上得应用信息  
  
pm 的源码 [Pm.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.2_r1/com/android/commands/pm/Pm.java#Pm) , 直接运行 adb shell pm 可以获取到该命令的帮助信息

* pm list package 列出安装在设备上的应用

不带任何选项：列出所有的应用的包名（不知道怎么找应用的包名的同学看这里）

adb shell pm list package

-s：列出系统应用

adb shell pm list package -s

-3：列出第三方应用

adb shell pm list package -3

-f：列出应用包名及对应的apk名及存放位置

adb shell pm list package -f

-i：列出应用包名及其安装来源，结果显示例子：  
  
package:com.zhihu.android installer=com.xiaomi.market

adb shell pm list package -i

命令最后增加 FILTER：过滤关键字，可以很方便地查找自己想要的应用

参数组合使用，例如，查找三方应用中知乎的包名、apk存放位置、安装来源：

[xuxu:~]$ adb shell pm list package -f -3 -i zhihu

package:/data/app/com.zhihu.android-1.apk=com.zhihu.android installer=com.xiaomi.market

* pm path 列出对应包名的 .apk 位置
* [xuxu:~]$ adb shell pm path com.tencent.mobileqq

package:/data/app/com.tencent.mobileqq-1.apk

* pm list instrumentation , 列出含有单元测试 case 的应用，后面可跟参数 -f （与 pm list package 中一样），以及 [TARGET-PACKAGE]
* pm dump , 后跟包名，列出指定应用的 dump 信息，里面有各种信息，自行查看

adb shell pm dump com.tencent.mobileqq

Packages:

Package [com.tencent.mobileqq] (4397f810):

userId=10091 gids=[3003, 3002, 3001, 1028, 1015]

pkg=Package{43851660 com.tencent.mobileqq}

codePath=/data/app/com.tencent.mobileqq-1.apk

resourcePath=/data/app/com.tencent.mobileqq-1.apk

nativeLibraryPath=/data/app-lib/com.tencent.mobileqq-1

versionCode=242 targetSdk=9

versionName=5.6.0

applicationInfo=ApplicationInfo{43842cc8 com.tencent.mobileqq}

flags=[ HAS\_CODE ALLOW\_CLEAR\_USER\_DATA ]

dataDir=/data/data/com.tencent.mobileqq

supportsScreens=[small, medium, large, xlarge, resizeable, anyDensity]

usesOptionalLibraries:

com.google.android.media.effects

com.motorola.hardware.frontcamera

timeStamp=2015-05-13 14:04:24

firstInstallTime=2015-04-03 20:50:07

lastUpdateTime=2015-05-13 14:05:02

installerPackageName=com.xiaomi.market

signatures=PackageSignatures{4397f8d8 [43980488]}

permissionsFixed=true haveGids=true installStatus=1

pkgFlags=[ HAS\_CODE ALLOW\_CLEAR\_USER\_DATA ]

User 0: installed=true blocked=false stopped=false notLaunched=false enabled=0

grantedPermissions:

android.permission.CHANGE\_WIFI\_MULTICAST\_STATE

com.tencent.qav.permission.broadcast

com.tencent.photos.permission.DATA

com.tencent.wifisdk.permission.disconnect

* pm install , 安装应用

目标 apk 存放于 PC 端，请用 adb install 安装  
  
目标 apk 存放于 Android 设备上，请用 pm install 安装

* pm uninstall , 卸载应用，同 adb uninstall , 后面跟的参数都是应用的包名
* pm clear , 清除应用数据
* pm set-install-location , pm get-install-location , 设置应用安装位置，获取应用安装位置

[0/auto]：默认为自动  
  
[1/internal]：默认为安装在手机内部  
  
[2/external]：默认安装在外部存储

**am**

又是一个庞大的命令。。。  
  
am 源码 [Am.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.4_r1/com/android/commands/am/Am.java#Am)

* am start , 启动一个 Activity，已启动系统相机应用为例

启动相机

[xuxu:~]$ adb shell am start -n com.android.camera/.Camera

Starting: Intent { cmp=com.android.camera/.Camera }

先停止目标应用，再启动

[xuxu:~]$ adb shell am start -S com.android.camera/.Camera

Stopping: com.android.camera

Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.android.camera/.Camera }

等待应用完成启动

[xuxu:~]$ adb shell am start -W com.android.camera/.Camera

Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.android.camera/.Camera }

Status: ok

Activity: com.android.camera/.Camera

ThisTime: 500

TotalTime: 500

Complete

启动默认浏览器打开一个网页

[xuxu:~]$ adb shell am start -a android.intent.action.VIEW -d http://testerhome.com

Starting: Intent { act=android.intent.action.VIEW dat=http://testerhome.com }

启动拨号器拨打 10086

[xuxu:~]$ adb shell am start -a android.intent.action.CALL -d tel:10086

Starting: Intent { act=android.intent.action.CALL dat=tel:xxxxx }

* am instrument , 启动一个 instrumentation , 单元测试或者 Robotium 会用到
* am monitor , 监控 crash 与 ANR
* [xuxu:~]$ adb shell am monitor
* Monitoring activity manager... available commands:
* (q)uit: finish monitoring

\*\* Activity starting: com.android.camera

* am force-stop , 后跟包名，结束应用
* am startservice , 启动一个服务
* am broadcast , 发送一个广播

还有很多的选项，自己多多发掘~~

**input**

这个命令可以向 Android 设备发送按键事件，其源码 [Input.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.4_r1/com/android/commands/input/Input.java#Input)

* input text , 发送文本内容，不能发送中文

adb shell input text test123456

前提先将键盘设置为英文键盘

* input keyevent , 发送按键事件，[KeyEvent.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.4_r1/android/view/KeyEvent.java#KeyEvent)

adb shell input keyevent KEYCODE\_HOME

模拟按下 Home 键 ，源码里面有定义：  
  
public static final int KEYCODE\_HOME = 3;  
  
因此可以将命令中的 KEYCODE\_HOME 替换为 3

* input tap , 对屏幕发送一个触摸事件

adb shell input tap 500 500

点击屏幕上坐标为 500 500 的位置

* input swipe , 滑动事件

adb shell input swipe 900 500 100 500

从右往左滑动屏幕  
  
如果版本不低于 4.4 , 可以模拟长按事件

adb shell input swipe 500 500 501 501 2000

其实就是在小的距离内，在较长的持续时间内进行滑动，最后表现出来的结果就是长按动作

到这里会发现，MonkeyRunner 能做到的事情，通过 adb 命令都可以做得到，如果进行封装，会比 MR 做得更好。

**screencap**

截图命令

adb shell screencap -p /sdcard/screen.png

截屏，保存至 sdcard 目录

**screenrecord**

4.4 新增的录制命令

adb shell screenrecord sdcard/record.mp4

执行命令后操作手机，ctrl + c 结束录制，录制结果保存至 sdcard

**uiautomator**

执行 UI automation tests ， 获取当前界面的控件信息

runtest：executes UI automation tests [RunTestCommand.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.4_r1/com/android/commands/uiautomator/RunTestCommand.java#RunTestCommand)  
  
dump：获取控件信息，[DumpCommand.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.4_r1/com/android/commands/uiautomator/DumpCommand.java#DumpCommand)

[xuxu:~]$ adb shell uiautomator dump

UI hierchary dumped to: /storage/emulated/legacy/window\_dump.xml

不加 [file] 选项时，默认存放在 sdcard 下

**ime**

输入法，[Ime.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.4_r1/com/android/commands/ime/Ime.java#Ime)

[xuxu:~]$ adb shell ime list -s

com.google.android.inputmethod.pinyin/.PinyinIME

com.baidu.input\_mi/.ImeService

列出设备上的输入法

[xuxu:~]$ adb shell ime set com.baidu.input\_mi/.ImeService

Input method com.baidu.input\_mi/.ImeService selected

选择输入法

**wm**

[Wm.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/4.4.4_r1/com/android/commands/wm/Wm.java#Wm)

[xuxu:~]$ adb shell wm size

Physical size: 1080x1920

获取设备分辨率

**monkey**

请参考 [Android Monkey 的用法](http://xuxu1988.com/2015/05/14/2015-05-02-Monkey/)

**settings**

[Settings.java](http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android-apps/4.4.4_r1/com/android/settings/Settings.java#Settings)，请参考 [探究下 Android4.2 中新增的 settings 命令](http://testerhome.com/topics/1993)

**dumpsys**

请参考 [android 中 dumpsys 命令使用](http://testerhome.com/topics/1462)

**log**

这个命令很有意思，可以在 logcat 里面打印你设定的信息，具体用途自己思考！

adb shell log -p d -t xuxu "test adb shell log"

-p：优先级，-t：tag，标签，后面加上 message

[xuxu:~]$ adb logcat -v time -s xuxu

--------- beginning of /dev/log/system

--------- beginning of /dev/log/main

05-15 13:57:10.286 D/xuxu (12646): test adb shell log

**getprop**

查看 Android 设备的参数信息，只运行 adb shell getprop，结果以 key : value 键值对的形式显示，如要获取某个 key 的值：

adb shell getprop ro.build.version.sdk

获取设备的 sdk 版本

**linux 命令**

操作你的 Android 设备，常用到的命令，只列出，不详解！

cat、cd、chmod、cp、date、df、du、grep、kill、ln、ls、lsof、netstat、ping、ps、rm、rmdir、top、touch、重定向符号 ">" ">>"、管道 "|"

有些可能需要使用 busybox ，另外建议 windows 下 安装一个 Cygwin , 没用过的请百度百科 [Cygwin](http://baike.baidu.com/link?url=NatnFuanrdGKgYFGyMvMT_T86-r5ETnVJd8HH6G3jPSj46QosKYAfUO3RafthymAiubSFAafN558p_lMqpJOua)

**END**

补充一个引号的用途：  
场景1、在 PC 端执行 monkey 命令，将信息保存至 D 盘 monkey.log，会这么写：

adb shell monkey -p com.android.settings 5000 > d:\monkey.log

场景2、在 PC 端执行 monkey 命令，将信息保存至手机的 Sdcard，可能会这么写：

adb shell monkey -p com.android.settings 5000 > sdcard/monkey.log

这里肯定会报错，因为最终是写向了 PC 端当前目录的 sdcard 目录下，而非写向手机的 Sdcard

这里需要用上引号：

adb shell "monkey -p com.android.settings 5000 > sdcard/monkey.log"

对这些命令都熟悉之后，那么接下来就是综合对编程语言的应用，思考如何用语言去处理这些命令，使得这些命令更加的方便于测试工作。

所以个人[github](https://github.com/gb112211)上的几个工具，核心都是 adb 命令，关键的地方在于怎么用自己所学的语言去处理这些命令。

貌似内容有点长。。

[二维码](https://testerhome.com/topics/2565)

转载文章时务必注明原作者及原始链接，并注明「发表于 TesterHome 」，并不得对作品进行修改。