ADB 命令大全

常用命令用于速查，都是经常需要用到的，没有接触过 adb 命令从第二节开始看，对每个命令有详细解释。

1. 常用命令：

```
adb devices #查看连接设备

adb -s cf27456f shell # 指定连接设备使用命令

adb install test.apk # 安装应用

adb install -r demo.apk #安装apk 到sd 卡：

adb uninstall cn.com.test.mobile #卸载应用，需要指定包

adb uninstall -k cn.com.test.mobile #卸载app 但保留数据和缓存文件

adb shell pm list packages #列出手机装的所有app 的包名

adb shell pm list packages -3 #列出除了系统应用的第三方应用包名

adb shell pm clear cn.com.test.mobile #清除应用数据与缓存

adb shell am start -ncn.com.test.mobile/.ui.SplashActivity #启动应用

adb shell dumpsys package #包信息Package Information

adb shell dumpsys meminfo #内存使用情况Memory Usage

adb shell am force-stop cn.com.test.mobile #强制停止应用

adb logcat #查看日志

adb logcat -c #清除Log 缓存

adb reboot #重启

adb get-serialno #获取序列号

adb shell getprop ro.build.version.release #查看Android 系统版本

adb shell top -s 10 #查看占用内存前10 的app

adb push <local> <remote> #从本地复制文件到设备

adb pull <remote> <local> #从设备复制文件到本地

adb bugreport #查看bug 报告
```

```
adb help #查看ADB 帮助
```

2.1 连接设备

adb [-d|-e|-s <serialNumber>] <command>

连接指定设备

参数：

-d 指定当前唯一通过 USB 连接的 Android 设备为命令目标

-e 指定当前唯一运行的模拟器为命令目标

-s <serialNumber> 指定相应 serialNumber 号的设备 / 模拟器为命令目标

command 为所需对设备执行的命令

示例：

```
$adb devices
List of devices attached
cf263b7f device
emulator-5554 offline
192.168.1.6:5555 device
$adb -s cf263b7f #连接cf264b8f 设备
```

adb devices 查看已连接的设备信息，上面已经连接 3 台设备。

2.2 查看信息

2.2.1 查看版本设备

adb version 查看 adb 版本信息

adb devices 查看 adb 连接设备

示例：

```
$adb devices
List of devices attached
1226959f device
3426422f offline
```

注意：offline 表示设备未连接成功或无响应，device 设备已连接

adb shell getprop ro.product.model 查看设备型号

adb get-serialno 获取设备序列号

adb bugreport 查看 bug 报告

adb logcat 查看日志

adb shell wm size 查看屏幕分辨率

adb shell wm density 查看屏幕密度

2.2.2 查看应用信息

adb shell pm list packages 列出手机装的所有 app 的包名

adb shell pm list packages -s 列出系统应用的所有包名

adb shell pm list packages -3 列出除了系统应用的第三方应用包名

adb shell pm list packages | find "test" win 列出手机装带有的 test 的包

adb shell pm list packages | grep ‘test’ linux 列出手机装带有的 test 的包

adb shell cat /sys/class/net/wlan0/address 获取 MAC 地址, 根据系统版本参数可能不同

adb shell getprop ro.build.version.release 查看 Android 系统版本

adb shell dumpsys activity services [<packagename>] 查看正在运行的 Services

<packagename> 参数不是必须的，指定 < packagename> 表示查看与某个包名相关的 Services，不指定表示查看所有 Services。

<packagename> 不一定要给出完整的包名，比如运行 adb shell dumpsys activity services org.zhihu，那么包名 org.zhihu.demo1、org.zhihu.demo2 和 org.zhihu 等相关的 Services 都会列出来。

2.3 app 安装和卸载

2.3.1 app 安装：

adb install <apkfile> 参数 apkfile 为. apk 文件名称

adb install –r test.apk 保留数据和缓存文件，重新安装 apk

adb install –s test.apk 安装 apk 到 sd 卡

2.3.2 app 卸载

adb uninstall <package> 参数 package 为软件包名称

示例：

```
$adb uninstall cn.com.test.mobile

#  卸载app 但保留数据和缓存文件
$adb uninstall -k ccn.com.test.mobile
```

2.4 启动停止服务

adb start–server

启动 adb 服务，基本不会用到，因为只要设备连接正确，会自动启动 adb server

adb kill–server

停止 adb server

adb –P <port> start–server

指定 adb server 的网络端口 port （默认为 5037）启动服务

2.5 与应用交互

adb shell pm clear <packagename>

清除应用数据与缓存

示例：

```
$adb shell pm clear cn.com.test.mobile
```

adb shell am force-stop <packagename>

强制停止应用

示例：强制停止微信

```
$adb shell am force-stop com.tencent.mm
```

adb shell am <command>

command 命令详解

> command 用途
> start [options] <INTENT> 启动 <INTENT> 指定的 Activity
> startservice [options] <INTENT> 启动 <INTENT> 指定的 Service
> broadcast [options] <INTENT> 发送 <INTENT> 指定的广播
> force-stop <packagename> 停止 <packagename> 相关的进程

<INTENT> 参数很灵活，和写 Android 程序时代码里的 Intent 相对应, 用于决定 intent 对象的选项如下：

> -a <ACTION> 指定 action，如 android.intent.action.VIEW
> -c <CATEGORY> 指定 category，如 android.intent.category.APP_CONTACTS
> -n <COMPONENT> 指定完整 component 名，用于明确指定启动哪个 Activity，如 com.example.app/.ExampleActivity

2.5.1 启动 Activity

adb shell am start [options] <INTENT>

示例：

```
#指定完整 component 名，用于明确指定启动哪个Activity
$adb shell am start -n <COMPONENT>
如： #表示调起微信主界面
$adb shell am start -n com.tencent.mm/.ui.LauncherUI
```

2.5.2 启动 Service

adb shell am startservice [options] <INTENT>

示例： 表示调起微信的某 Service。

```
$adb shell am startservice -n com.tencent.mm/.plugin.accountsync.model.
```

2.5.3 发送广播

可以向所有组件广播，也可以只向指定组件广播。

adb shell am broadcast [options] <INTENT>

示例：

```
#向所有组件广播 BOOT_COMPLETED (开机广播)
$adb shell am broadcast -a android.intent.action.BOOT_COMPLETED
#如：只向 org.mazhuang.boottimemeasure/.BootCompletedReceiver 广播 BOOT_
$adb shell am broadcast -a android.intent.action.BOOT_COMPLETED -n org.
```

系统预定义的广播：

action	触发时机
android.net.conn.CONNECTIVITY_CHANGE	网络连接发生变化
android.intent.action.SCREEN_ON	屏幕点亮
android.intent.action.SCREEN_OFF	屏幕熄灭
android.intent.action.BATTERY_LOW	电量低，会弹出电量低提示框
android.intent.action.BATTERY_OKAY	电量恢复了
android.intent.action.BOOT_COMPLETED	设备启动完毕
android.intent.action.DEVICE_STORAGE_LOW	存储空间过低
android.intent.action.DEVICE_STORAGE_OK	存储空间恢复
android.intent.action.PACKAGE_ADDED	安装了新的应用
android.net.wifi.STATE_CHANGE	WiFi 连接状态发生变化
android.net.wifi.WIFI_STATE_CHANGED	WiFi 状态变为启用/关闭/正在启动/正在关闭/未知
android.intent.action.BATTERY_CHANGED	电池电量发生变化
android.intent.action.INPUT_METHOD_CHANGED	系统输入法发生变化
android.intent.action.ACTION_POWER_CONNECTED	外部电源连接
android.intent.action.ACTION_POWER_DISCONNECTED	外部电源断开连接
android.intent.action.DREAMING_STARTED	系统开始休眠
android.intent.action.DREAMING_STOPPED	系统停止休眠
android.intent.action.WALLPAPER_CHANGED	壁纸发生变化
android.intent.action.HEADSET_PLUG	插入耳机
android.intent.action.MEDIA_UNMOUNTED	卸载外部介质
android.intent.action.MEDIA_MOUNTED	挂载外部介质
android.os.action.POWER_SAVE_MODE_CHANGED	省电模式开启

2.5.4 强制停止应用

adb shell am force-stop <packagename>

示例：

```
# 查询出包名
$adb shell pm list packages
.....
# 强制停止微信
$adb shell am force-stop com.tencent.mm
```

2.6 文件管理

2.6.1 复制设备里的文件到电脑

adb pull <设备里的文件路径> [电脑上的目录]

示例：

```
$adb pull /sdcard/abc.mp4  ~/tmp/
```

> *小技巧：设备上的文件路径可能需要 root 权限才能访问，如果你的设备已经 root 过，可以先使用 adb shell 和 su 命令在 adb shell 里获取 root 权限后，先 cp /path/on/device /sdcard/filename 将文件复制到 sdcard，然后 adb pull /sdcard/filename /path/on/pc。*

2.6.2 复制电脑里的文件到设备

adb push <电脑上的文件路径> < 设备里的目录 >

示例：

```
$adb push  e:/ss.au3  /data/local/tmp/
```

> *小技巧：设备上的文件路径普通权限可能无法直接写入，如果你的设备已经 root 过，可以先 adb push /path/on/pc /sdcard/filename，然后 adb shell 和 su 在 adb shell 里获取 root 权限后，cp /sdcard/filename /path/on/device*

2.7 使用 ADB 命令模拟按键 / 输入

adb shell input keyevent <keycode>

keycode 位操作参数，不同的 keycode 能实现不同的功能

完整的功能见：
https://developer.android.com/reference/android/view/KeyEvent.html

keycode 能实现不同的功能，对应的编码如下：

keycode	含义
3	HOME 键
4	返回键
5	打开拨号应用
6	挂断电话
24	增加音量
25	降低音量
26	电源键
27	拍照（需要在相机应用里）
64	打开浏览器
82	菜单键
85	播放/暂停
86	停止播放
87	播放下一首
88	播放上一首
122	移动光标到行首或列表顶部
123	移动光标到行末或列表底部
126	恢复播放
127	暂停播放
164	静音
176	打开系统设置
187	切换应用
207	打开联系人
208	打开日历
209	打开音乐
210	打开计算器
220	降低屏幕亮度

示例：

```
$adb shell input keyevent 24 #增加音量
$adb shell input keyevent 25 #降低音量
$adb shell input keyevent 164 #静音
$adb shell input keyevent 85  #播放/暂停
$adb shell input keyevent 86 #停止播放
$adb shell input keyevent 87 #播放下一首
$adb shell input keyevent 88 #播放上一首
$adb shell input keyevent 126 #恢复播放
$adb shell input keyevent 127 #暂停播放
$adb shell input keyevent 224 #点亮屏幕
$adb shell input keyevent 223 #熄灭屏幕
$adb shell input swipe 300 1000 300 500  #滑动解锁，向上滑动手势解锁
#参数 300 1000 300 500 分别表示起始点x坐标 起始点y坐标 结束点x坐标 结束点y坐
$adb shell input text hello #焦点处于某文本框时输入文本
```

2.8 查看日志

2.8.1 Android 日志

[adb] logcat [<option>] ... [<filter-spec>] ...

按级别过滤日志

按某级别过滤日志则会将该级别及以上的日志输出，Android 日志的优先级如下：

> V —— Verbose（最低，输出得最多）
> D —— Debug
> I —— Info
> W —— Warning
> E —— Error
> F —— Fatal
> S —— Silent（最高，啥也不输出）

示例：输出 W 之上的日志，W,E,F,S

```
$adb logcat *:W
```

按 tag 和级别过滤日志

<filter-spec> 可以由多个 <tag>[:priority] 组成

示例：输出 tag ActivityManager 的 I 以上级别日志，输出 tag MyApp 的 D 以上级别日志，及其它 tag 的 S 级别日志（即屏蔽其它 tag 日志）。

```
$adb logcat ActivityManager:I MyApp:D *:S
```

日志格式

adb logcat –v <format>

指定日志输出格式

示例：

```
$adb logcat -v <format>  指定日志输出格式
$adb logcat -v brief   #默认格式,<priority>/<tag>(<pid>): <message>
$adb logcat -v process #<priority>(<pid>) <message>
$adb logcat -v tag  #<priority>/<tag>: <message>
$adb logcat -v raw  #<message>
$adb logcat -v time  #<datetime> <priority>/<tag>(<pid>): <message>
$adb logcat -v threadtime #<datetime> <pid> <tid> <priority> <tag>: <me
$adb logcat -v long  #[ <datetime> <pid>:<tid> <priority>/<tag>:] <mess
$adb logcat -v long ActivityManager:I *:S  #指定格式可与上面的过滤同时使用
```

清空日志

```
$adb logcat -c
```

2.8.2 内核日志

```
$adb shell dmesg
```

查看内核日志

2.9 查看设备信息

2.9.1 型号

```
$adb shell getprop ro.product.model
```

2.9.2 电池状况

```
$adb shell dumpsys battery
```

2.9.3 屏幕分辨率

```
$adb shell wm size
```

2.9.4 屏幕密度

```
$adb shell wm density
```

2.9.5 显示屏参数

```
$adb shell dumpsys window displays
```

2.9.6 android_id

```
$adb shell settings get secure android_id
```

2.9.7 IMEI

```
$adb shell dumpsys iphonesubinfo
#而在 Android 5.0 及以上版本里这个命令输出为空，得通过其它方式获取了（需要 rc
adb shell
su
$service call iphonesubinfo 1
```

2.9.8 Android 系统版本

```
$adb shell getprop ro.build.version.release
```

2.9.9 IP 地址

```
$adb shell ifconfig | find “Mask”
$adb shell ifconfig wlan0 #设备连着 WiFi，可以使用如下命令来查看局域网 IP
$adb shell netcfg # 上面两个无结果可以用这个，查看网络连接状态
```

2.9.10 Mac 地址

```
$adb shell cat /sys/class/net/wlan0/address # 设备不同可能地址不同
```

2.9.11 CPU 信息

```
$adb shell cat /proc/cpuinfo
```

2.9.12 内存信息

```
$adb shell cat /proc/meminfo
```

2.9.13 更多硬件与系统属性

```
$adb shell cat /system/build.prop
```

adb shell getprop <属性名>

也可以

```
$adb shell cat /system/build.prop | grep ro.product.cpu.abi
$adb shell cat /system/build.prop | find “ro.product.cpu.abi”
```

属性名	含义
ro.build.version.sdk	SDK 版本
ro.build.version.release	Android 系统版本
ro.build.version.security_patch	Android 安全补丁程序级别
ro.product.model	型号
ro.product.brand	品牌
ro.product.name	设备名
ro.product.board	处理器型号
ro.product.cpu.abilist	CPU 支持的 abi 列表[节注一]
persist.sys.isUsbOtgEnabled	是否支持 OTG
dalvik.vm.heapsize	每个应用程序的内存上限
ro.sf.lcd_density	屏幕密度

2.10 修改设置

修改设置之后，运行恢复命令仍然不太正常，可以运行 adb reboot 重启设备或手动重启。

修改设置的原理主要是通过 settings 命令修改 /data/data/com.android.providers.settings/databases/settings.db 里存放的设置值。

2.10.1 分辨率

adb shell wm size 480x1024 #将分辨率修改为 480px * 1024px

adb shell wm size reset #恢复原分辨率

2.10.2 屏幕密度

adb shell wm density 160 #屏幕密度修改为 160dpi

adb shell wm density reset #恢复原屏幕密度

2.10.3 显示区域

adb shell wm overscan 0,0,0,100

四个数字分别表示距离左、上、右、下边缘的留白像素，以上命令表示将屏幕底部 100px 留白

adb shell wm overscan reset #恢复显示区域

2.10.4 关闭 USB 调试模式

adb shell settings put global adb_enabled 0

使用命令无法恢复调试模式，只能通过手动

2.10.5 状态栏和导航栏的显示隐藏

adb shell settings put global policy_control <key-values>

<key-values> 可由如下几种键及其对应的值组成，格式为 <key1>=<value1>:<key2>=<value2>

key 键对应的值

key	含义
immersive.full	同时隐藏
immersive.status	隐藏状态栏
immersive.navigation	隐藏导航栏
immersive.preconfirms	?

这些键对应的值可则如下值用逗号组合

value	含义
apps	所有应用
*	所有界面
packagename	指定应用
-packagename	排除指定应用

示例：

```
# 设置在所有界面下都同时隐藏状态栏和导航栏
$adb shell settings put global policy_control immersive.full=*

# 设置在包名为 com.package1 和 com.package2 的应用里隐藏状态栏，在除了包名为
$adb shell settings put global policy_control immersive.status=com.packa
```

2.11 实用功能

2.11.1 屏幕截图

adb exec-out screencap -p > img.png # 老版本无 exec-out 命令，只适合于新版的截图

adb shell screencap -p /sdcard/img.png # 老版本截图先保存在设备端

adb pull /sdcard/img.png # 通过 pull 拷贝到本地

screencap 参数：

参数 含义

-p 指定保存文件为 png 格式

-d display-id 指定截图的显示屏编号（有多显示屏的情况下）

2.11.2 录制屏幕

默认录制时间和最长录制时间都是 180s

adb shell screenrecord /sdcard/filename.mp4

screenrecord 参数：

参数 含义

--size WIDTHxHEIGHT 视频的尺寸，比如 1280x720，默认是屏幕分辨率。

--bit-rate RATE 视频的比特率，默认是 4Mbps。

--time-limit TIME 录制时长，单位秒。

--verbose 输出更多信息。

2.11.3 重新挂载 system 分区为可写

注：需要 root 权限

/system 分区默认挂载为只读，但有些操作比如给 Android 系统添加命令、删除自带应用等需要对 /system 进行写操作，所以需要重新挂载它为可读写。

步骤：

1. 进入 shell 并切换到 root 用户权限。

```
$adb shell
$su
```

2. 查看当前分区挂载情况。

```
debugfs /sys/kernel/debug debugfs rw,relatime 0 0
none /acct cgroup rw,relatime,cpuacct 0 0
none /sys/fs/cgroup tmpfs rw,seclabel,relatime,mode=750,gid=1000 0 0
tmpfs /mnt/obb tmpfs rw,seclabel,relatime,mode=755,gid=1000 0 0
none /dev/cpuctl cgroup rw,relatime,cpu 0 0
/dev/block/mtdblock0 /system ext4 ro,seclabel,relatime,data=ordered 0 0
.......
```

找到其中我们关注的带 /system 的那一行：

3. 重新挂载。

```
$mount -o remount,rw -t yaffs2 /dev/block/mtdblock0 /system
```

2.11.4 查看连接过的 WiFi 密码

注：需要 root 权限

```
$adb shell
$su
$cat /data/misc/wifi/*.conf
```

2.11.5 设置系统日期和时间

注：需要 root 权限

```
$adb shell
$su
$date -s 20190531.131600  #将系统日期和时间更改为 2019 年 05 月 31 日 13 点
```

2.11.6 重启手机

```
$adb reboot
```

2.11.7 检测设备是否已 root

```
$adb shell
$su
```

此时命令行提示符是 $ 则表示没有 root 权限，是 # 则表示已 root。

2.11.8 使用 Monkey 进行压力测试

Monkey 可以生成伪随机用户事件来模拟单击、触摸、手势等操作，可以对正在开发中的程序进行随机压力测试。

```
# 向 <packagename> 指定的应用程序发送 500 个伪随机事件
$adb shell monkey -p <packagename> -v 500
```

monkey 官方文档

https://developer.android.com/studio/test/monkey.html

2.11.9 开启 / 关闭 WiFi

注：需要 root 权限

开启 WiFi：

```
$adb root
$adb shell svc wifi enable
```

关闭 WiFi：

```
$adb root
$adb shell svc wifi disable
```

2.12 刷机相关命令

注：不要随便操作，没实验过无法保证成功

2.12.1 重启到 Recovery 模式

```
$adb reboot recovery
```

2.12.2 从 Recovery 重启到 Android

```
$adb reboot
```

2.12.3 重启到 Fastboot 模式

```
$adb reboot bootloader
```

2.12.4 通过 sideload 更新系统

如果我们下载了 Android 设备对应的系统更新包到电脑上，那么也可以通过 adb 来完成更新。

以 Recovery 模式下更新为例：

1. 重启到 Recovery 模式。

```
$adb reboot recovery
```

2. 在设备的 Recovery 界面上操作进入 Apply update–Apply from ADB。

注：不同的 Recovery 菜单可能与此有差异，有的是一级菜单就有 Apply update from ADB。

3. 通过 adb 上传和更新系统。

```
$adb sideload <path-to-update.zip>
```

2.13 更多 adb shell 命令

2.13.1 查看进程

adb shell ps

2.13.2 查看实时资源占用情况

adb shell top

top 命令参数如下：

使用方法: top [–m max_procs] [–n iterations] [–d delay] [–s sort_column] [–t] [–h]

> –m num 最多显示多少个进程
> –n num 刷新多少次后退出
> –d num 刷新时间间隔（单位秒，默认值 5）
> –s col 按某列排序（可用 col 值：cpu, vss, rss, thr）
> –t 显示线程信息
> –h 显示帮助文档

2.13.3 其它

命令	功能
cat	显示文件内容
cd	切换目录
chmod	改变文件的存取模式/访问权限
df	查看磁盘空间使用情况
grep	过滤输出
kill	杀死指定 PID 的进程
ls	列举目录内容
mount	挂载目录的查看和管理
mv	移动或重命名文件
ps	查看正在运行的进程
rm	删除文件
top	查看进程的资源占用情况

全文完