# ADB: Android Debug Bridge

## and other Android command line tools

How ADB Works - from developer.android.com; includes all available commands

How ADB Works - from Google Git; adb overview

How ADB Works - from dummies.com / do not take it personally!

Killing the adb server - why

Always starting with adb devices - why

Starting an emulator from command line

Connecting a physical device via TCP/IP

Command redirection

Install, update and uninstall an application

How to find a package name

Collecting application logs

Recording video from terminal/command prompt

Taking a screenshot from terminal/command prompt

Pull and push files from and to Android device

Changing runtime permissions from terminal/command prompt

Using dumpsys tool to diagnose the device (when needed)

### Killing the adb server - why

“In some cases, you might need to terminate the adb server process and then restart it to resolve the problem (e.g., if adb does not respond to a command).

To stop the adb server, use

```
adb kill-server
```

You can then restart the server by issuing any other adb command.”

### Always starting with adb devices - why

To make sure the device you are going to manipulate with is actually connected, always start with

**adb devices**

And then issue your next adb command

### Starting an emulator from command line

emulator -list-avds - returns the list of created emulators on your machine (their

names) emulator -avd <emulator_name> - start the emulator

### Connecting a physical device via TCP/IP

**Note:** the very first link How ADB Works has the steps as well

Steps:

1. Your phone and computer are **ON THE SAME WiFi**

2. Connect your Android phone via USB

3. adb devices

4. adb tcpip <port_number_for_server>

6. Disconnect device from USB

6. adb connect 192.168.4.198:5559 - <phone_ip>:<port_number_for_server>

7. adb disconnect - disconnects every physical device connected this way

**How to find ip from command line** (alternatively, search in the phone settings)

adb -d shell ip addr show wlan0

### Command redirection

adb -d <command> - sends a command to **the only** connected physical device **(CONNECTED via USB)**

adb -e <command> - sends a command **to the only** connected emulator (**CONNECTED via TCP/IP**)

***Note:*** *Once a physical device is connected via TCP/IP, use -e or -s <serial_number> command redirection option as -d sends command to devices connected via USB*

if more than one device or more than one emulator connected, use

adb -s <serial> <command>

For ex.,

adb -s **emulator-5554** install .apk

## Install, update and uninstall an application

adb install Downloads/<file_name>.apk - use **your** path to .apk `adb devices |`

grep device | grep -v devices | cut -f 1 | xargs -I {} adb -s {} install **Reinstal**

**(updated) application**

adb install **-r** Downloads/<file_name>.apk - add -r before **your** path to .apk

**Unistall**

adb uninstall com.adjoy.standalone.test2 - use package name

## How to find a package name

adb shell pm list packages - returns the list of packages installed on the device adb

shell pm list packages -f <app_name> - returns the package for the specific app

## Collecting application logs

Logcat is a command-line tool for debugging Android applications

adb logcat - command to start logging

**useful options:**

adb logcat -c clears all the info that might be in buffer from the previous sessions

**adb logcat | grep 'adjoy'** - filter the log for a particular application

***Note:*** *On Windows machine, please use* **find** *instead of* **grep - for ex., find “adjoy”**

***Note:*** *If you need to grep more than one word, please do the following:* **adb logcat |**

**grep -E "(adjoy|dabbl)"**

adb logcat > file_name.txt - writes the log a text file (or -f <file_name>), for example,

**adb logcat | grep 'adjoy' > zip_code_crash.txt**

***Note:*** *On Windows machine, please use* **find** *instead of* **grep - for ex., find “adjoy”**

adb logcat `tag:priority` - filtering by priority; for example,

adb logcat *:W

**Note:** *Tags are defined by an app developer, use * in a tag place* **Note:** *if you*

*use* ***zsh,*** *you need to use single quotes around the expression* ***'*:W'*** More about

filtering from the official website

**adb logcat '*:W' | grep 'adjoy' > zip_code_crash.txt**

***Note:*** *On Windows machine, please use* **find** *instead of* **grep - for ex., find “adjoy”**

## Recording video from terminal/command prompt

adb shell screenrecord /sdcard/ErrorMsgRegistrationScreen.mp4 give your

files a meaningful name. You may use a bug # as a file name, too.

Default recording time is 180 seconds (3 minutes). You may, however, change the that by adding following the arguments

adb shell screenrecord --time-limit <TIME> /sdcard/ErrorMsgRegistrationScreen.mp4,

instead of <TIME> placeholder, insert the needed time in seconds: --time-limit 120 will produce a 2-minute video.

**Since video is saved to sdcard, we need to "pull" it from the device**

adb pull /sdcard/ErrorMsgRegistrationScreen.mp4 - pulls to current working directory
adb pull /sdcard/ErrorMsgRegistrationScreen.mp4 **/Users/tanya/Desktop - pulls to a specified destination**

If no distention directory specified, the file will be stored at your current working directory (to check - pwd on Mac, cd on Windows)

**To remove a file from your device, run**

adb shell rm /sdcard/ErrorMsgRegistrationScreen.mp4

Recording the video in Android Studio -

https://developer.android.com/studio/debug/am-video.html?hl=en

## Taking a screenshot from **terminal/command prompt**