ADB Shell Commands

The Android Debug Bridge (adb) provides a Unix shell that you can use to run a variety of commands on an emulator or connected device. The command binaries are stored in the file system of the emulator or device, at `/system/bin/...`

In this document

Issuing Shell Commands

You can use the `shell` command to issue commands, with or without entering the adb remote shell on the emulator/device. To issue a single command without entering a remote shell, use the `shell` command like this:

```
adb [-d|-e|-s <serialNumber>] shell <shell_command>
```

Or enter a remote shell on an emulator/device like this:

```
adb [-d|-e|-s <serialNumber>] shell
```

When you are ready to exit the remote shell, press CTRL+D or type `exit` .

Using activity manager (am)

Within an adb shell, you can issue commands with the activity manager (`am`) tool

shell, the syntax is:

```
am <command>
```

Console

Download

Android Studio

You can also issue an activity manager command directly from ac hout entering a remote shell. For example:

Workflow

```
adb shell am start -a android.intent.action.VIEW
```

Tools Help

Build System

Table 2. Available activity manager commands

Peformance Tools

Command	Description
`start [options] <INTENT>`	Start an `Activity` specified by `<INTENT>`. See the Specification for <INTENT> arguments. Options are: • `-D` : Enab' 'bugging. • `-W` : Wait for launch to complete. • `--start-profiler <FILE>` : Start profil d send results to `<FILE>`. • `-P <FILE>` : Like `--start-profiler`, but profiling stops when the app goes idle. • `-R` : Repeat the activity launch `<COUNT>` times. Prior to each repeat, the top activity will be finished. • `-S` : Force stop the target app before starting the activity.

Testing Tools

Support Library

Data Binding Library

Revisions

NDK

Eclipse with ADT

	• `--user <USER_ID> | current` : ify which user run as; if not specified, then run as the curren .
`startservice [options] <INTENT>`	Start the Ser specified by `<INTENT>` . See the Speci on for <INTENT> arguments. Options are: • `--user < _ID> | current` : Specify which user to run as; if n ecified, then run as the current user.
`force-stop <PACKAGE>`	Force stop ev ng associated with `<PACKAGE>` (the app's package name).
`kill [options] <PACKAGE>`	Kill all proces ssociated with `<PACKAGE>` (the app's package name). This command kills only processes tha safe to kill and that will not impact the user experience. Options are: • `--user <USER_ID> | all | current` : Specify user whose processes to kill; all users if not specified.
`kill-all`	Kill all background processes.
`broadcast [options] <INTENT>`	Issue a broadcast intent. See the Specification for <INTENT>

	• `[--user <USER_ID> | all | current]` cify which user to send to; if pecified then send to all users
`instrument [options] <COMPONENT>`	Start monitori th an `Instrumentation` instance. Typically the t `<COMPONENT>` is the form `<TEST_PACK /<RUNNER_CLASS>` . Options are: • `-r` : Print raw results (otherwise decode `<REPORT_ STREAMRESULT>`). Use with `perf true]` to generate raw output for performance measurements. • `-e <NAME ALUE>` : Set argument `<NAME>` to `<VALUE>` . For test runners a common form is `-e <te nner_flag> <value>[ lue>...]` . • `-p <FILE>` : Write profiling data to `<FILE>` . • `-w` : Wait for instrumentation to finish before returning. Required for test runners. • `--no-window-animation` : Turn off window animations while running. • `--user <USER_ID> |`

	user if not specified.
`profile start <PROCESS> <FILE>`	Start profiler on `<PROCESS>`, writing results to `<FILE>`.
`profile stop <PROCESS>`	Stop profiler on `<PROCESS>`.
`dumpheap [options] <PROCESS> <FILE>`	Dump the heap of `<PROCESS>`, write to `<FILE>`. Options are: • `--user [<USER_ID>|current]`: When supplying process name, specify user of process to dump; uses current user if not specified. • `-n`: Dump native heap instead of managed heap.
`set-debug-app [options] <PACKAGE>`	Set application `<PACKAGE>` to debug. Options are: • `-w`: Wait for debugger when application starts. • `--persistent`: Retain this value.
`clear-debug-app`	Clear the package previous set for debugging with `set-debug-app`.
`monitor [options]`	Start monitoring for crashes or ANRs. Options are: • `--gdb`: Start gdbserv on the given port at crash/ANR.
`screen-compat [on|off] <PACKAGE>`	Control screen compatibility mode of `<PACKAGE>`.

	testing your app across different screen sizes by mimicking a small screen resolu sing a device with a large screer vice versa. Example: `am display e 1280x800`
`display-density <dpi>`	Override emulator/device display density. This and is helpful for testing your app across different screen densit high-density screen enviro t using a low density scree vice versa. Example: `am display density 480`
`to-uri <INTENT>`	Print the given intent specification as a URI. See the Specification for <INTENT> arguments.
`to-intent-uri <INTENT>`	Print the given intent specification as an `intent:` URI. See the Specification for <INTENT> arguments.

Specification for <INTENT> arguments

Using package manager (pm)

Within an adb shell, you can issue commands with the package manager (`pm`) tool to perform actions and queries on application packages installed on the device. While in a shell, the syntax is:

```
pm <command>
```

```
adb shell pm uninstall com.example.MyApp
```


Table 3. Available package manager commands.

Command	Description
`list packages [options] <FILTER>`	P all packages, optionally only th whose package name c ns the text in `<FILTER>`. O s: • `-f`: See their associated file. • `-u`: Filter to only show abled packages. : Filter to only show abled packages. • `-s`: Filter to only show system ckages. • `-3`: Filter to only show third party packages. : See the installer for the packages. • `-u`: Also include uninstalled packages. • `--user <USER_ID>`: The user space to query.
`list permission-groups`	Prints all known permission groups.
`list permissions [options] <GROUP>`	Prints all known permissions, optionally only those in `<GROUP>`. Options:

		• `-s` : Short summ • `-u` : Only list dangerous missions. : List only the permissions users will see.
`list instrumentation`	Tools Help Build System Peformance Tools Testing Tools	L test packages. Options: • `-f` : List the APK file for the t package. `ARGET_PACKAGE>` : List test packages for only this app.
`list features`	Support Library	F all features of the system.
`list libraries`	Data Binding Library	Prints all the libraries supported by the current device.
`list users`	Revisions	F all users on the system.
`path <PACKAGE>`	NDK	Print the path to the APK of the given `<PACKAGE>` .
`install [options] <PATH>`	Eclipse with ADT	I s a package (specified by `<PATH>`) to the system. Options: • `-l` : Install the package with forward lock. • `-r` : Reinstall an exisiting app, keeping its data. • `-t` : Allow test APKs to be installed. • `-i`

Console

	name. : Install packa shared mass storage (such as card). : Install package on the internal system memory. : Allow version code downgrade.
`uninstall [options] <PACKAGE>`	Removes a package from the system. Options: : Keep the data and cache directories around after package removal.
`clear <PACKAGE>`	Deletes all data associated with a package.
`enable <PACKAGE_OR_COMPONENT>`	Enable the given package or component (written as "package/class").
`disable <PACKAGE_OR_COMPONENT>`	Disable the given package or component (written as "package/class").
`disable-user [options] <PACKAGE_OR_COMPONENT>`	Options: • `--user <USER_ID>` : The user to disable.
`grant <PACKAGE_PERMISSION>`	Grant permissions to applications. Only optional permissions the application has declared can be granted.

	permissions the application has declared can be revo
`set-install-location <LOCATION>`	Changes the default install l n. Location values: • `0` : Auto—Let system decide best location. : Internal—install on internal device storage. : External—install on ernal media. **Note:** This is only intended for bugging; using this can use applications to break d other undesireable behavior.
`get-install-location`	R s the current install location. Return values: • `0 [auto]` : Lets system decide the best location • `1 [internal]` : Installs on internal device storage • `2 [external]` : Installs on external media
`set-permission-enforced <PERMISSION> [true|false]`	Specifies whether the given permission should be enforced.
`trim-caches <DESIRED_FREE_SPACE>`	Trim cache files to reach the given free space.
`create-user <USER_NAME>`	Create a new user with the given `<USER_NAME>` , printing the new

	<USER_IDENTIFIER> , deleting a a associated w
get-max-users	Prints the maximum number of u supported by the device.

Taking a device screenshot

The `screencap` command is a shell utility for taking a screenshot of a device display. While in a shell, the syntax is:

```
screencap <filename>
```

To use the `screencap` from the command line, type the followin

```
$ adb shell screencap /sdcard/screen.png
```

Here's an example screenshot session, using the adb shell to capture the screenshot and the `pull` command to download the file from th ice:

```
$ adb shell
shell@ $ screencap /sdcard/screen.png
shell@ $ exit
$ adb pull /sdcard/screen.png
```

Recording a device screen

The `screenrecord` command is a shell utility for recording the display of devices running Android 4.4 (API level 19) and higher. The utility records screen activity to an MPEG-4 file.

> **Note:** Audio is not recorded with the video file.

A developer can use this file to create promotional or training videos. While in a shell, the syntax is:

To use `screenrecord` from the command line, type the following:

Console

```
$ adb shell screenrecord /sdcard/demo.mp4
```

Stop the screen recording by pressing Ctrl-C, otherwise the recording stops automatically at three minutes or the time limit set by `--time-li`.

To begin recording your device screen, run the `screenrecord` c and to record the video. Then, run the `pull` command to download the from the device to the host computer. Here's an example recording sessior

```
$ adb shell
shell@ $ screenrecord --verbose /sdcard/demo.mp4
(press Ctrl-C to stop)
shell@ $ exit
$ adb pull /sdcard/demo.mp4
```

The `screenrecord` utility can record at any supported resolution and bit rate you request, while retaining the aspect ratio of the device display. The utility records at the native display resolution and orientation by default, with a ma n length of three minutes.

There are some known limitations of the `screenrecord` utility that you should be aware of when using it:

- Some devices may not be able to record at their native display resolution. If you encounter problems with screen recording, try using a lower screen resolution.
- Rotation of the screen during recording is not supported. If the screen does rotate during recording, some of the screen is cut off in the recording.

Table 4. `screenrecord` options

Options	Description
`--help`	Displays command syntax and options
`--size <WIDTHxHEIGHT>`	Sets the video size: `1280x720`. The default value is the device's native display resolution (if supported), 1280x720 if not. For best

Download
Android Studio
Workflow
Tools Help
Build System
Peformance Tools
Testing Tools
Support Library
Data Binding Library
Revisions
NDK
Eclipse with ADT

`--bit-rate <RATE>`	Sets the video bit rate for the video, in megabits per second. The default value is 4Mbps. You car ease the bit rat video quality, but doing so resul... ...arger movie files. The following example sets the rec bit rate to 6Mbps: `screenrecord --bit-rate 0000 /sdcard/demo.mp4`
`--time-limit <TIME>`	Sets the maximum recording ti... seconds. The default and maximum value is 180 (3 minut
`--rotate`	Rotates the output 90 degrees. feature is experimental.
`--verbose`	Displays log information on the mand-line screen. If you do not set this option, the utility do t display any information while running.

Other shell commands

For a list of all the available shell programs, use the following command:

```
adb shell ls /system/bin
```

Help is available for most of the commands.

Table 5 lists some of the more common adb shell commands.

Table 5. Some other adb shell commands

Shell Command	Description	Comments
`dumpsys`	Dumps system data to the screen.	The Dalvik Debug Monitor Server (DDMS) tool offers integrated debug environment
`dumpstate`	Dumps state to a file.	
`logcat [option]... [filter-spec]...`	Enables system and app logging	

`dmesg`	Prints kernel debugging messages to the screen.	use.
`start`	Starts (restarts) an emulator/device instance.	
`stop`	Stops execution of an emulator/device instance.	

Console

Download
Android Studio
Workflow
Tools Help
Build System
Peformance Tools
Testing Tools
Support Library
Data Binding Library
Revisions
NDK
Eclipse with ADT