GENYMOTION

GMTool Guide

Version 2.12

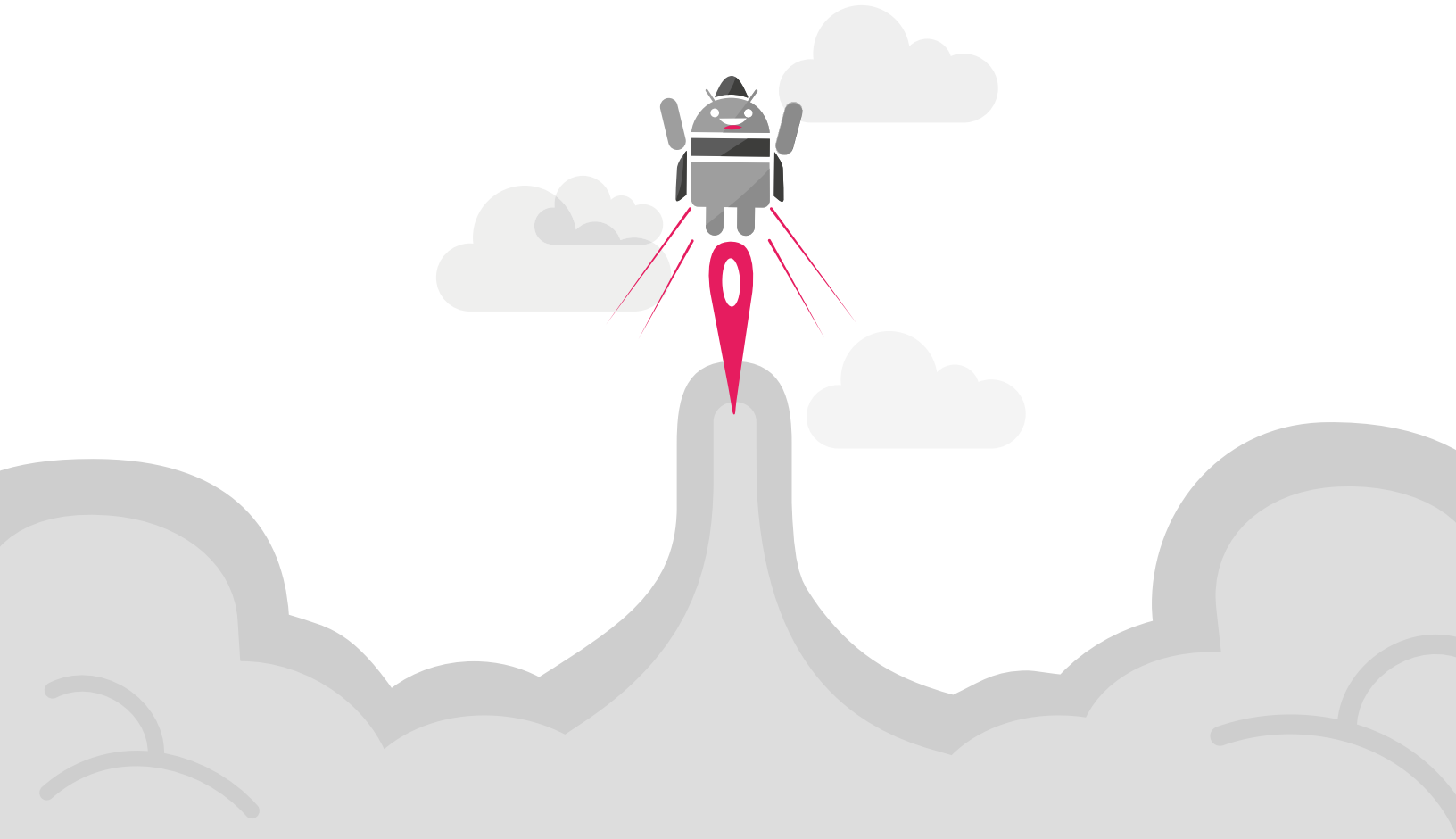

Table of contents

Overview

GMTool is a command line tool allowing you to use every command of Genymotion and virtual devices, in order to automate series of actions. It can be started by running `gmtool` from a command prompt.

This guide lists and explains all commands available in GMTool, as well as error messages that can be returned.

In this guide, the following instructional icons are used:

 ***Notes, tips or additional information.***

 ***Situations that could cause performance issues or data losses.***

General commands

The following commands return information related to the use of GMTool:

- `version`: returns Genymotion version and revision number.
- `help`: displays the help for the entire tool, for a command group or for a specific command.
  Example
  This command `gmtool admin create help` or `gmtool help admin create` displays all commands and options available for the `admin create` command, i.e.:

```
admin create <TEMPLATE_NAME> <VIRTUAL_DEVICE_NAME>
[--density=<SCREEN_DENSITY>] [--width=<SCREEN_WIDTH>]
[--height=<SCREEN_HEIGHT>] [--virtualkeyboard=<on|off>]
[--navbar=<on|off>] [--nbcpu=<NUMBER_OF_CPUs>]
[--ram=<RAM_IN_MB>] [--network-mode=<nat|bridge>]
[--bridged-if=<BRIDGED_INTERFACE>] [--sysprop=<PROPERTY:VALUE>]
[--sysprop=<PROPERTY:VALUE>]...
```

Command groups

Some commands are contained within a group, which allows you to interact on a given scope. Command groups are:

- License;
- Config;
- Admin;
- Device.

The following options can be used in any command group:

- `--verbose`: when added to a command, gives a complete description of the command to help diagnose errors.
- `-t|--timeout=<TIMEOUT_IN_SECONDS>`: specifies network timeout in seconds for every command that uses network connections.
- `--cloud`: use virtual devices from the Genymotion Cloud. You need access to the Genymotion Cloud platform, however some features may not be available on Genymotion Cloud yet. They will exit with a "This action is not supported" error message.

In GMTool, the `=` boolean operator is optional; the argument can be defined with or without entering it.

License

The `license` command group allows you to perform actions related to the Genymotion license. Commands available within this group are:

- `info`: returns the license type, number of activated workstations and expiration date.
- `register <LICENSE_KEY>`: registers a license key or renews Genymotion activation.
- `count`: returns the number of activated workstations with the registered license key.
- `validity`: returns the number of days of validity remaining for the registered license key.

Example

This command `gmtool license register 0123456789` registers the license key "0123456789" and authenticates the user with the username and password given previously with the `config` command.

Config

The `config` command group allows you to define Genymotion settings. Commands available within this group are:

- `reset`: restores default settings.
- `clearcache`: removes temporary and downloaded files.
- `signout`: signs out the user and removes the license.
- `print`: returns values of the specified configuration options.
- `OPTION=<ARGUMENT>`: sets the option with an argument. Options and their possible values are:
  - `statistics=<on|off>`: allows or prevents Genymotion to collect usage statistics.
  - `username=<USERNAME>`: sets the user to authenticate.
  - `password=<PASSWORD>`: sets the password to associate with the username.
  - `store_credentials=<on|off>`: remembers credentials or not.
  - `license_server=<on|off>`: activates or deactivates the use of a license server.
  - `license_server_address=<SERVER_ADDRESS>`: sets the license server address.
  - `proxy=<on|off>`: activates or deactivates the use of a proxy.
  - `proxy_address=<PROXY_ADDRESS>`: sets the proxy address.
  - `proxy_port=<PROXY_PORT>`: sets the proxy port.
  - `proxy_authentication=<on|off>`: activates or deactivates proxy authentication.
  - `proxy_username=<PROXY_USERNAME>`: sets the username to be used for proxy authentication.
  - `proxy_password=<PROXY_PASSWORD>`: sets the password to be used for proxy authentication.
  - `virtual_device_path=<VIRTUAL_DEVICE_PATH>`: sets the path to the virtual device directory.
  - `sdk_path=<SDK_PATH>`: sets the path to the custom Android SDK directory.
  - `use_custom_sdk=<on|off>`: activates or deactivates the use of the custom Android SDK.
  - `screen_capture_path=<SCREEN_CAPTURE_PATH>`: sets the path to the screen capture directory.

Admin

The `admin` command group allows you to administrate virtual devices.

This feature is only available with Indie and Business licenses.

Commands available within this group are:

- `templates`: lists all online and offline available virtual device templates and their basic properties (name, screen size and Android version). Options available for this command are:
  - `--full`: displays all properties of the virtual device templates (name, UUID, description, Android version, API level, Genymotion version, screen width, height, density and DPI, number of CPUs, RAM, internal storage, telephony, Android navigation bar visibility, virtual keyboard).
  - `-f|--force-refresh`: forces a refresh of the template list from the server. By default, the template list is stored locally to avoid too many server requests.
- `create <TEMPLATE_NAME> | <TEMPLATE_UUID> <VIRTUAL_DEVICE_NAME>`: creates a virtual device from the specified template.

  

  If some shared virtual devices have the same name as other templates, you must specify the template UUID instead of the template name. You can retrieve the UUID using `gmtool admin templates --full`

  The default template configuration can be overridden by specifying optional arguments. Options available for this command are:

  - `--density=<SCREEN_DENSITY>`: sets the screen density of the virtual device. The value must be the density in pixels per inch or the name of the corresponding bucket, as shown in the table below.

Density values

Density bucket name	Pixels per inch
ldpi	120
mdpi	160
tvdpi	213

Density bucket name	Pixels per inch
hdpi	240
xhdpi	320
420dpi	420
xxhdpi	480
560dpi	560
xxxhdpi	640

- `--width=<SCREEN_WIDTH>`: sets the screen width of the virtual device.
- `--height=<SCREEN_HEIGHT>`: sets the screen height of the virtual device.
- `--virtualkeyboard=<on|off>`: activates or deactivates the virtual keyboard.
- `--navbar=<on|off>`: displays or hides the Android navigation bar in the virtual device.
- `--nbcpu=<NUMBER_OF_CPUs>`: sets the number of processors used by the virtual device.
- `--ram=<RAM_IN_MB>`: sets the memory space allocated to the virtual device in MB.
- `--network-mode=<nat|bridge>`: sets the host network interface mode for the virtual device.
- `--bridged-if=<BRIDGED_INTERFACE>`: when network mode is bridge, sets the bridged interface for the virtual device.
- `--sysprop=<PROPERTY:VALUE>`: sets one build system property of the virtual device. Available properties are: MODEL, PRODUCT, MANUFACTURER, BOARD, BRAND, DEVICE, DISPLAY, SERIAL, TYPE, FINGERPRINT, TAGS. You can set multiple system properties. Values of each property are detailed in *developer.android.com*.

- `edit <VIRTUAL_DEVICE_NAME>`: edits the specified virtual device with optional arguments. Options available for this command are:
  - `--density=<SCREEN_DENSITY>`: sets the screen density of the virtual device. For more information about density values, please refer to table *Density values*.
  - `--width=<SCREEN_WIDTH>`: sets the screen width of the virtual device.

  - `--height=<SCREEN_HEIGHT>`: sets the screen height of the virtual device.
  - `--virtualkeyboard=<on|off>`: activates or deactivates the virtual keyboard.
  - `--navbar=<on|off>`: displays or hides the Android navigation bar in the virtual device.
  - `--nbcpu=<NUMBER_OF_CPUs>`: sets the number of processors used by the virtual device.
  - `--ram=<RAM_IN_MB>`: sets the memory space allocated to the virtual device in MB.
  - `--network-mode=<nat|bridge>`: sets the host network interface mode for the virtual device.
  - `--bridged-if=<BRIDGED_INTERFACE>`: when network mode is bridge, sets the bridged interface for the virtual device.
  - `--sysprop=<PROPERTY:VALUE>`: sets one build system property of the virtual device. Available properties are: MODEL, PRODUCT, MANUFACTURER, BOARD, BRAND, DEVICE, DISPLAY, SERIAL, TYPE, FINGERPRINT, TAGS. You can set multiple system properties. Values of each property are detailed in *developer.android.com*.
- `delete <VIRTUAL_DEVICE_NAME>`: deletes the specified virtual device.
- `clone <ORIGINAL_VIRTUAL_DEVICE_NAME> <NEW_VIRTUAL_DEVICE_NAME>`: duplicates the specified original virtual device to a new virtual device.
- `list`: lists all virtual devices. Options available for this command are:
  - `--running`: lists running virtual devices.
  - `--off`: lists turned off virtual devices.
- `details [<VIRTUAL_DEVICE_NAME_1>] [<VIRTUAL_DEVICE_NAME_2>]`: returns the properties of one or more specified virtual devices. If no virtual device is specified, returns the properties of all virtual devices.
- `start <VIRTUAL_DEVICE_NAME>`: starts the specified virtual device. This command is not available with `--cloud`.
- `stop <VIRTUAL_DEVICE_NAME>`: stops the specified virtual device. This command is not available with `--cloud`.
- `stopall`: stops all virtual devices.
- `factoryreset <VIRTUAL_DEVICE_NAME>`: restores the specified virtual device to factory state.
- `logzip`: generates an archive of all Genymotion log files. Option `-n|--name <VIRTUAL_DEVICE_NAME> [LOG_ARCHIVE_PATH]` generates a log archive of the specified virtual device, at the defined location.

  If an archive file already exists, it will be overwritten.

- `startdisposable <TEMPLATE_NAME> | <TEMPLATE_UUID> <VIRTUAL_DEVICE_NAME>`: creates and starts a virtual device from the specified template. The default template

configuration can be overridden by specifying optional arguments (see the list of optional arguments available in the `create` command). A disposable device is a device that is automatically created when you start it, and automatically deleted when you stop it. To set the connection on a specific port, you can define option `--adb-serial-port <PORT>` with a port comprised between 1024 and 65535.
This command is only available with `--cloud`.

> ***If some shared virtual devices have the same name as other templates, you must specify the template UUID instead of the template name. You can retrieve the UUID using `gmtool admin templates --full`***

- `stopdisposable <VIRTUAL_DEVICE_NAME>`: stops and deletes the specified virtual device started with `startdisposable`. This command is only available with `--cloud`.

Device

The `device` command group allows you to directly interact with a virtual device.

> ***This feature is only available with Indie and Business licenses.***

Options available within this group are:

- `-n|--name <VIRTUAL_DEVICE_NAME>`: interacts with the specified virtual device.

  > ***If not specified, interacts with the running virtual device. If more than one virtual device is running, the use of options `-n` or `-all` is mandatory.***

- `--all`: interacts with all running virtual devices.
- `--start`: starts the virtual device specified by `-n` if not already started.

Commands available in the `device` command group are:

- `logcatdump <DESTINATION_FILE_PATH>`: copies the logcat output onto the specified destination file.

  > ***If a logcat dump file already exists, it will be overwritten.***

- `logcatclear`: empties the logcat content of a virtual device.
- `push <SOURCE_PATH> [DESTINATION_DIRECTORY_PATH]`: sends a file or directory from the host computer to the virtual device.

  > ***If no destination directory is specified, the file or directory is stored by default in `/sdcard/Downloads`.***

- `pull <SOURCE_PATH> <DESTINATION_DIRECTORY_PATH>`: copies a file or directory from the virtual device to the host computer.

  

  If combined with option `--all`, destination directories are created for each running virtual device.

  

  If a destination file already exists, it will be overwritten.

- `install <APK_FILE_PATH>`: installs an application using its APK file on the virtual device.
- `flash <ARCHIVE_PATH>`: installs the archive content into the specified virtual device.
- `adbconnect`: connects the specified virtual device using the ADB tool.
  To set the connection on a specific port, you can define option `--adb-serial-port <PORT>` with a port comprised between 1024 and 65535. This option is only available with `--cloud`.
- `adbdisconnect`: disconnects the specified virtual device from the ADB tool.

Error messages

The table below explains error codes returned by GMTool.

Code	Message
1	The command does not exist.
2	A wrong parameter value has been entered.
3	The command has failed.
4	The virtualization engine does not respond.
5	The specified virtual device could not be found.
6	Unable to sign in.
7	Unable to register the license key.
8	Unable to activate the license.
9	The license has not been activated.
10	The license key is invalid.
11	The command has missing arguments.
12	Unable to stop the virtual device.
13	Unable to start the virtual device.
14	This command can only run with Indie and Business licenses.

Autocompletion

To be more productive with GMTool, you can install a completion script for your shell. We provide completion scripts for Bash and Zsh shells. In this chapter, we explain how to install those scripts and we refer to the directory containing the Genymotion installer as `<GENYMOTION_DIR>`, the directory containing the GMTool binary as `<GMTOOL_DIR>` and the directory containing the completion script as `<COMPLETION_DIR>`.

- **On Linux:**
  - `<GENYMOTION_DIR>` is where you installed Genymotion
  - `<GMTOOL_DIR>` is the same as `<GENYMOTION_DIR>`
  - `<COMPLETION_DIR>` is `<GENYMOTION_DIR>/completion`
- **On macOS:**
  - `<GENYMOTION_DIR>` is `/Applications/Genymotion.app`
  - `<GMTOOL_DIR>` is `<GENYMOTION_DIR>/Contents/MacOS`
  - `<COMPLETION_DIR>` is `<GENYMOTION_DIR>/Contents/Resources/completion`

Requirements

To take advantage of shell completion, the gmtool binary and the vboxmanage binary (provided by VirtualBox) must be in your PATH.

- For gmtool, make sure you add `<GMTOOL_DIR>` to `$PATH`.
- For vboxmanage, add your VirtualBox installation directory to `$PATH`.

Installing the Bash completion script

To install the Bash completion script, add this line at the end of your `~/.bash_profile` file:

```
. <COMPLETION_DIR>/bash/gmtool.bash
```

Completion works with Bash 3.2 or later, but we recommend using at least version 4.0, especially if you work with file names containing spaces.

Installing the Zsh completion script

To install the Zsh completion script, open your `~/.zshrc` and add this line before the call to compinit:

```
fpath=(<COMPLETION_DIR>/zsh $fpath)
```