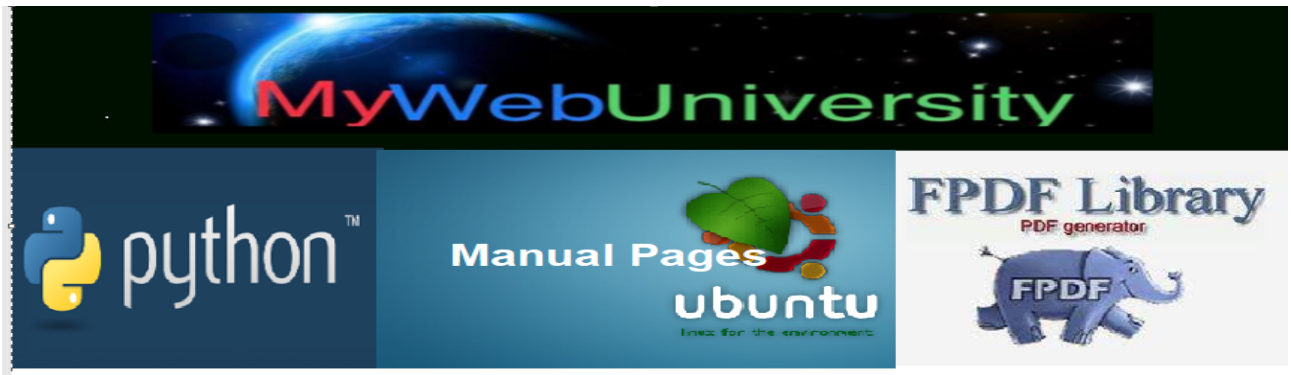

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'fwupdmgr.1'

$ man fwupdmgr.1

fwupdmgr(1) fwupdmgr man page fwupdmgr(1)

NAME

fwupdmgr - firmware update manager client utility

SYNOPSIS

fwupdmgr [CMD]

DESCRIPTION

fwupdmgr is a command line fwupd client intended to be used interactively. The terminal output between versions of fwupd is not guaranteed to be stable, but if you plan on parsing the results then adding --json might be just what you need.

There are also graphical tools to firmware available for various desktop environments. These applications may be more useful to many users compared to using the command line.

? GNOME Software: <https://wiki.gnome.org/Apps/Software>

? GNOME Firmware: <https://gitlab.gnome.org/World/gnome-firmware>

? KDE Discover: <https://userbase.kde.org/Discover>

? Canonical Firmware Updater: <https://github.com/canonical/firmware-updater>

? System76 Firmware Manager: <https://github.com/pop-os/firmware-manager>

On most systems fwupd is configured to download metadata from the Linux Vendor Firmware Ser? vice <https://fwupd.org/> and more information about the LVFS is available here: <https://lvfs.readthedocs.io/>

Most users who want to just update all devices to the latest versions can do fwupdmgr re? fresh and then fwupdmgr update. At this point the system will asking for confirmation, up? date some devices, and may then reboot to deploy updates that require a restart.

OPTIONS

The fwupdmgr command takes various options depending on the action. The actions are split into rough behavior groups as follows:

Interacting With Discovered Devices

The following actions can be used to view details about the enumerated devices on the sys? tem:

get-devices: Show all devices, with optional filtering applied using --filter.

get-updates: Show all the available updates for a specific device.

get-releases: Show all the installable versions (older, the same, or newer) for a specific device.

get-details: View details about a specific .cab archive, even if the target device has not been found on this system.

search: Find updates present in the metadata, regardless if the device has been enumerated on the system. Searching will match firmware releases by:

? The full GUID of a device, e.g. eb68dbae-3aef-5077-92ae-9016d1f0c856

? The AppStream ID of a device, e.g. work.frame.Desktop.RyzenAIMax300.BIOS.firmware

? The name (full or subset) of a firmware update, e.g. Desktop Ryzen AI MAX 300

? The name (full or subset) of a firmware vendor, e.g. Framework

? The protocol used by the firmware, e.g. org.uefi.capsule

? Any issue reported as solved by the update, e.g. CVE-2022-21894

? Any cabinet checksum included with the update, in SHA-1 or SHA-256 format, e.g. b34950fc65dabc0cb50e4c5f081829e40bf92e13

Deploying Firmware

The following actions can be used to install firmware onto devices:

update: Update each device in turn to the newest available version.

install: Install any available release to a specific device.

reinstall: Re-install the same version to a specific device if available.

downgrade: Install an older release to a specific release if available.

switch-branch: Switch between different firmware ?branches? for a specific device. For in? stance, there may be a ?community supported? branch and a ?non-free upstream vendor? branch for the exact same hardware, both available on the LVFS.

local-install: Install a local firmware archive. The regular install will also fall back to this if the first argument is a local file that exists.

activate: Set the installed firmware version as running, which may mean the device becomes

offline or unresponsive for a few moments.

sync: Install all releases with a matching BKC tag if a ?Best Known Configuration? has been set for the local machine.

Approved Firmware

The following actions can be used to allow specific firmware releases from being installed:

The following actions can be used to control the allowlist of specific firmwares:

set-approved-firmware: Sets the list of approved firmware for remotes. Once the allow-list has been set to a non-empty value only firmware matching these checksums will be marked as installable. It is still possible to install firmware ?manually? using fwupdmgr install -- this setting only affects firmware updates referenced in enabled remotes with the Approval? Required=true setting.

get-approved-firmware: Gets the list of approved firmware, returning an empty list if there is no allow-list in place.

The checksums used for allowing are the cabinet archive checksums in SHA-1 or SHA-256 for? mat.

Emulation

Device emulation allows recording the device behavior so that we can replay the device re? sponses and test writing firmware without that actual hardware plugged in.

The following actions can be used when emulating devices:

emulation-tag: Adds devices to watch for future emulation.

emulation-save: Save captured device emulation data to a JSON file. Only data from devices that have previously been tagged for emulation will be returned.

emulation-load: Load device emulation data from a JSON file.

emulation-untag: Removes devices to watch for future emulation.

When tagging devices, either the device ID or GUID can be used to identify the device.

The following actions can be used to run automated device tests:

device-emulate: Emulate a device using a JSON manifest, which will operate on emulated de? vices only.

device-test: Test a device using a JSON manifest which downloads and verifies cabinet archives and deploys them on actual physical devices.

Remotes

The following actions can be used to control the remotes (the online firmware metadata store) configured in the daemon:

refresh: Download the latest online metadata from configured and enabled remotes.

get-remotes: Get the list of configured remotes, which may or may not be enabled.

enable-remote: Enable a pre-configured remote so that it can be refreshed and used.

disable-remote: Disable a remote, but do not delete or remote stored metadata.

modify-remote: Edit a remote, for instance turning on properties such as AutomaticReports.

clean-remote: Cleans a remote, deleting metadata where required.

Historical Data

The following actions can be used to upload, export or clear historical data:

report-export: Export firmware history as an offline file for manual upload to a service such as the LVFS.

report-history: Share firmware history with the remote owner, typically used to indicate the success ratio for a specific update. In some cases, sharing the history will return results to webpages describing the failure in more details, some with workarounds.

get-history: Show the firmware update history as stored by fwupd.

get-results: Show the result of the last firmware update for a specific device.

clear-results: Clears the results from the last update if possible.

report-devices: Upload the list of updatable devices to a remote server so that they can put pressure on the vendor to support Linux users.

Platform Security

The following actions can be used to view or fix platform security issues:

security: Gets the list of host security attributes which are used to evaluate the security level of the machine.

security-fix: Fix a specific host security attribute failure.

security-undo: Undo the host security attribute fix, which may be required if security-fix caused a regression.

The following actions can be used to list or set firmware BIOS settings:

get-bios-settings: Retrieve BIOS firmware settings and the allowable values.

set-bios-setting: Sets one or more BIOS firmware settings.

verify-update: Update the stored checksums that are used by the verify action.

Others

The following actions may be for from non-interactive scripts or use in CI:

check-reboot-needed: Check if any devices are pending a reboot to complete update.

device-wait: Wait for a device to appear in the daemon device list.

download: Download a file using the same mechanisms that firmware and metadata are used.

modify-config: Modifies a daemon configuration value such as IgnorePower. See man fwupd.conf for the full list of variables.

reset-config: Resets a daemon configuration section back to the default values.

quit: Asks the daemon to quit after it has finished any firmware update in progress.

inhibit: Inhibit the system to prevent accidental manual or automatic upgrades.

uninhibit: Uninhibit the system to allow upgrades.

unlock: Unlocks the device for firmware access, which may be required for some platform de? vices.

get-plugins: Show all plugins registered with the daemon.

hwids: Return all the hardware IDs for the machine. These GUIDs are sometimes called CHIDs when using Microsoft Windows, and the values returned by fwupd should also match those from ComputerHardwareIds.exe.

EXIT STATUS

Commands that successfully execute will return ?0?, with generic failure as ?1?.

There are also several other exit codes used: A return code of ?2? is used for commands that have no actions but were successfully executed, and ?3? is used when a resource was not found.

BUGS

See GitHub Issues: <https://github.com/fwupd/fwupd/issues>

SEE ALSO

<fwupdtool(1)> <fwupd.conf(5)>