NAME

udisksctl - The udisks command line tool

SYNOPSIS

udisksctl status

udisksctl info {--object-path *OBJECT* | --block-device *DEVICE* | --drive *DRIVE*}

udisksctl mount {--object-path *OBJECT* | --block-device *DEVICE*} [--filesystem-type *TYPE*]
[--options *OPTIONS*...] [--no-user-interaction]

udisksctl unmount {--object-path *OBJECT* | --block-device *DEVICE*} [--force] [--no-user-interaction]

udisksctl unlock {--object-path *OBJECT* | --block-device *DEVICE*} [--no-user-interaction]
[--key-file *PATH*] [--read-only]

udisksctl lock {--object-path *OBJECT* | --block-device *DEVICE*} [--no-user-interaction]

udisksctl loop-setup --file *PATH* [--read-only] [--offset *OFFSET*] [--size *SIZE*] [--no-user-interaction]

udisksctl loop-delete {--object-path *OBJECT* | --block-device *DEVICE*} [--no-user-interaction]

udisksctl power-off {--object-path *OBJECT* | --block-device *DEVICE*} [--no-user-interaction]

udisksctl smart-simulate --file *PATH* {--object-path *OBJECT* | --block-device *DEVICE*}
[--no-user-interaction]

udisksctl monitor

udisksctl dump

udisksctl help

DESCRIPTION

udisksctl is a command-line program used to interact with the **udisksd**(8) daemon process.

COMMANDS

status

Shows high-level information about disk drives and block devices.

info

Shows detailed information about *OBJECT*, *DEVICE* or *DRIVE*.

mount

Mounts a device. The device will be mounted in a subdirectory in the /run/media hierarchy - upon successful completion, the mount point will be printed to standard output.

-t, **--filesystem-type**

Filesystem type to use. If not specified, autodetected filesystem type will be used.

-o, **--options**

The device will be mounted with a safe set of default options. You can influence the options passed to the **mount**(8) command using this option. Note that only safe options are allowed - requests with inherently unsafe options such as *suid* or *dev* that would allow the caller to gain additional privileges, are rejected.

unmount

Unmounts a device. This only works if the device is mounted. The option **--force** can be used to request that the device is unmounted even if active references exists.

-f, **--force**

Lazy unmount. Detach the filesystem from the file hierarchy now, and clean up all references to this filesystem as soon as it is not busy anymore.

unlock

Unlocks an encrypted device. The passphrase will be requested from the controlling terminal and upon successful completion, the cleartext device will be printed to standard output.

--key-file=*PATH*

Read passphrase from the given file.

lock

Locks a device. This only works if the device is a cleartext device backed by a cryptotext device.

loop-setup

Sets up a loop device backed by *FILE*.

-f, **--file=***FILE*

File to set up a loop device for.

-r, **--read-only**
Set up a read-only loop device.

-o, **--offset**=*OFFSET*
The data start is moved *OFFSET* bytes into the specified file.

-s, **--size**=*SIZE*
The data end is set to no more than *SIZE* bytes after the data start.

loop-delete
Tears down a loop device.

power-off
Arranges for the drive to be safely removed and powered off. On the OS side this includes ensuring that no process is using the drive, then requesting that in-flight buffers and caches are committed to stable storage. The exact steps for powering off the drive depends on the drive itself and the interconnect used. For drives connected through USB, the effect is that the USB device will be deconfigured followed by disabling the upstream hub port it is connected to.

Note that as some physical devices contain multiple drives (for example 4-in-1 flash card reader USB devices) powering off one drive may affect other drives. As such there are not a lot of guarantees associated with performing this action. Usually the effect is that the drive disappears as if it was unplugged.

smart-simulate
Sets SMART data from the libatasmart blob given by *FILE* - see /usr/share/doc/libatasmart-devel-VERSION/ for blobs shipped with libatasmart. This is a debugging feature used to check that applications act correctly when a disk is failing.

-f, **--file**=*FILE*
File with the libatasmart blob.

monitor
Monitors the daemon for events.

dump
Prints the current state of the daemon.

help
Prints help and exit.

DEVICE SPECIFICATION

For commands that require a device as an argument following options can be used to specify it.

-b, **--block-device=***DEVICE*
Specify a device by its device file path. For example */dev/sda*.

-p, **--object-path=***OBJECT*
Specify a device by the UDisks internal object path without the /org/freedesktop/UDisks2 prefix. For example *block_devices/sda* for the /dev/sda disk.

-d, **--drive=***DRIVE*
Specify a drive by name, for example *VirtIO_Disk*. This can be currently used only together with the **info** command.

COMMON OPTIONS

The option **--no-user-interaction** can be used to request that no interaction (such as the user being presented with an authentication dialog) must occur when checking with **polkit**(8) whether the caller is authorized to perform the requested action.

AUDIENCE

This program does not assume that the caller is the super user - it is intended to be used by unprivileged users and authorizations are checked by the udisks daemon using **polkit**(8). Additionally, this program is not intended to be used by scripts or other programs - options/commands may change in incompatible ways in the future even in maintenance releases. See the "API STABILITY" section of **udisks**(8) for more information.

BASH COMPLETION

udisksctl ships with a bash completion script to complete commands, objects, block devices and some options.

AUTHOR

This man page was originally written for UDisks2 by David Zeuthen <zeuthen@gmail.com> with a lot of help from many others.

BUGS

Please send bug reports to either the distribution bug tracker or the upstream bug tracker at **https://github.com/storaged-project/udisks/issues**.

SEE ALSO

udisks(8), **udisksd**(8), **umount.udisks2**(8), **polkit**(8)