# nvme-cli NVMe drive management

**START HERE**

`nvme list`
Show all NVMe drives

`nvme id ctrl /dev/nvme0`
Controller identity

`nvme id ns /dev/nvme0n1`
Namespace details

`nvme log smart /dev/nvme0`
Health and wear

`nvme log error /dev/nvme0`
Recent error entries

`nvme device-self-test /dev/nvme0 -s 1`
Start a short self-test

`nvme log self-test /dev/nvme0`
Self-test results

`nvme list-subsys`
Subsystems and paths

## CONCEPTS

Use the character device (/dev/nvme0) for controller commands and the block device (/dev/nvme0n1) for namespace ones. The digits are not linked across the two.

**Subsystem** nvme-subsys0
**Controller** /dev/nvme0
**Namespace** /dev/nvme0n1

nvme-cli 3.x groups commands (nvme id ctrl, nvme log smart, nvme ns create). The old dashed names still work as deprecated aliases that warn on stderr.

format, sanitize and ns delete destroy data and cannot be undone. Check the device path against nvme list first, twice.

## DISCOVER

`nvme list`
List NVMe devices with size and model

`nvme list -o json`
List devices as JSON

`nvme list-subsys`
Show subsystems, controllers and paths

`nvme id ctrl /dev/nvme0`
Show controller model, serial, firmware

`nvme id ns /dev/nvme0n1`
Show namespace size and LBA formats

`nvme id ns /dev/nvme0 -n <nsid>`
Show a namespace via the controller

`nvme id ctrl /dev/nvme0 -V`
Include vendor-specific region

`nvme list-ctrl /dev/nvme0`
List controller IDs in the subsystem

`nvme --version`
Show the nvme-cli version

## HEALTH AND LOGS

`nvme log smart /dev/nvme0`
Show SMART health, wear, temperature

`nvme log smart /dev/nvme0 -o json`
SMART log as JSON for scripts

`nvme log error /dev/nvme0`
Show the error log

## HEALTH AND LOGS (cont.)

`nvme log error /dev/nvme0 -e <n> -V`
Last n entries, valid ones only

`nvme log fw /dev/nvme0`
Show firmware slots and active revision

`nvme log sanitize /dev/nvme0`
Show sanitize status and progress

`nvme get-feature /dev/nvme0 -f 0x04`
Read the temperature threshold feature

`nvme log smart /dev/nvme0 -b > smart.bin`
Dump the raw SMART log

**Key SMART fields**

| Field | Meaning |
|---|---|
| `critical_warning` | Non-zero means act now |
| `percentage_used` | Wear estimate, may pass 100 |
| `available_spare` | Spare blocks left (%) |
| `media_errors` | Data integrity errors |
| `unsafe_shutdowns` | Power loss without flush |
| `data_units_written` | Units of 512,000 bytes |

## SELF-TEST

`nvme device-self-test /dev/nvme0 -s 1`
Start a short self-test

`nvme device-self-test /dev/nvme0 -s 2`
Start an extended self-test

`nvme device-self-test /dev/nvme0 -s 1 -w`
Short test, wait for completion

`nvme log self-test /dev/nvme0`
Show self-test results

`nvme device-self-test /dev/nvme0 -s 0xf`
Abort a running self-test

`nvme device-self-test /dev/nvme0`
Show whether a test is running

## NAMESPACES

`nvme id ns-list /dev/nvme0`
List active namespace IDs

`nvme ns create /dev/nvme0 --nsze=<blocks> --ncap=<blocks> --flbas=0`
Create a namespace
*Sizes are in logical blocks. --nsze-si takes sizes like 6.14T instead.*

## NAMESPACES (cont.)

`nvme ns attach /dev/nvme0 -n <nsid> -c <ctrl-id>`
Attach a namespace to a controller

`nvme ns detach /dev/nvme0 -n <nsid> -c <ctrl-id>`
Detach a namespace from a controller

`nvme ns delete /dev/nvme0 -n <nsid>`
DESTROYS the namespace and its data

`nvme ns create /dev/nvme0 --nsze-si=<size> --ncap-si=<size> --block-size=4096`
Create by size with 4K blocks

## FORMAT AND ERASE

`nvme format /dev/nvme0n1 --lbaf=<n>`
ERASES: reformat with LBA format n
*Run nvme id ns first to see which LBA formats the drive offers.*

`nvme format /dev/nvme0n1 --block-size=4096`
ERASES: reformat to 4K blocks

`nvme format /dev/nvme0n1 --ses=1`
ERASES: format plus user data erase

`nvme format /dev/nvme0n1 --ses=2`
ERASES: format plus cryptographic erase

`nvme sanitize /dev/nvme0 -a start-block-erase`
ERASES: block erase the whole drive
*Runs in the background. Watch it with nvme log sanitize.*

`nvme sanitize /dev/nvme0 -a start-crypto-erase`
ERASES: crypto erase the whole drive

`nvme sanitize /dev/nvme0 -a start-overwrite -n <passes>`
ERASES: overwrite the whole drive

`nvme sanitize /dev/nvme0 -a exit-failure`
Leave sanitize failure mode

## FIRMWARE

`nvme fw download /dev/nvme0 --fw=<file>`
Upload a firmware image

`nvme fw commit /dev/nvme0 -s <slot> -a 1`
Store image in slot, activate at reset

## FIRMWARE (cont.)

`echo 1 > /sys/class/nvme/nvme0/device/reset`
Reset controller to run new firmware

`nvme fw commit /dev/nvme0 -s <slot> -a 3`
Activate without a reset, if supported

`nvme fw download /dev/nvme0 --fw=<file> --xfer=0x20000`
Download in 128 KiB chunks

`nvme fw download /dev/nvme0 --fw=<file> --stream`
Download without mapping the whole file

## OPTIONS AND HELP

`nvme <command> -o json`
Machine-readable output

`nvme <command> -v`
More detail (repeat for more)

`nvme <command> --dry-run`
Print the command without sending it

`nvme <command> --timeout=<ms>`
Set the command timeout

`nvme help`
List all subcommands

`man nvme-<command>`
Read the manual for a subcommand

## RECIPES

**Health check of every drive**

```
for d in /dev/nvme[0-9]*; do
  [ -c "$d" ] || continue
  echo "== $d"; nvme log smart
"$d" | grep -E 'critical|
percentage|media'
done
```

**Extract temperature with jq**

```
nvme log smart /dev/nvme0 -o
json | jq '.temperature'
# check the field name with:
nvme log smart /dev/nvme0 -o
json
```

**Wipe a drive and wait**

```
# confirm the target first
nvme list
nvme sanitize /dev/nvme0 -a
start-block-erase
watch -n5 nvme log sanitize /
dev/nvme0
```

**Update firmware**

```
nvme log fw /dev/nvme0
nvme fw download /dev/nvme0 --
fw=<file>
nvme fw commit /dev/nvme0 -s
<slot> -a 1
echo 1 > /sys/class/nvme/nvme0/
device/reset
nvme id ctrl /dev/nvme0 | grep
-w fr
```