

storagectl



- [Description](#)
 - [Commands](#)
 - [Options](#)
 - [Use as a mount helper](#)
 - [Examples](#)
 - [Exit status](#)
 - [See Also](#)
-

systemd

storagectl

1

storagectl

mount.storage

Enumerate and mount storage volumes provided by storage providers

storagectl

OPTIONS

COMMAND

NAME

mount

-t

storage

PROVIDER

:

VOLUME

DIRECTORY

mount

-t

storage.

FSTYPE

PROVIDER

:

VOLUME

DIRECTORY

Description

`storagectl` may be used to inspect storage providers and the storage volumes they expose. A storage provider is a service implementing the `io.systemd.StorageProvider` [Varlink](#) interface, registered as an AF_UNIX socket below the well-known socket directory `/run/systemd/io.systemd.StorageProvider/` (in system mode) or `$XDG_RUNTIME_DIR` `/systemd/io.systemd.StorageProvider/` (in user mode). The two storage providers shipped with systemd are `systemd-storage-block@.service(8)`, which exposes the system's block devices, and `systemd-storage-fs@.service(8)`, which exposes regular files and directories from a backing file system.

The tool also provides a `mount(8)` helper for the file system type `storage`, which permits mounting storage volumes to arbitrary places. See "Use as a mount helper" below for details.

Commands

The following commands are understood:

`volumes` [`<GLOB>`]

List storage volumes provided by all storage providers running on the system (or, with `--user`, in the user runtime). The optional `<GLOB>` argument is a shell-style pattern (see `fnmatch(3)`) that filters the result by volume name. The output is a table containing the providing service, the volume name, its type (`blk`, `reg` or `dir`), whether it is read-only, and — if known — its size and the number of bytes used.

This is the default command if none is specified.

`templates` [`<GLOB>`]

List volume templates supported by the running storage providers. Templates encapsulate a configuration to use when creating volumes on-the-fly, when they are acquired. Template support is an optional feature for providers, and only applies to providers that allow creation of volumes on-the-fly. See the respective provider documentation for details, for example `systemd-storage-fs@.service(8)`. The optional `<GLOB>` argument filters by template name. Storage providers that do not implement template-based volume creation (such as the block-device provider) do not contribute to this output.

`providers`

List the storage providers known to the system. This is determined by scanning the well-known socket directory for `AF_UNIX` sockets that look like `io.systemd.StorageProvider` endpoints. For each provider it is also reported whether the socket can currently be connected to.

Options

The following options are understood:

`--system`

Operate on system-wide storage providers. Sockets are looked for in `/run/systemd/io.systemd.StorageProvider/`. This is the default.

`--user`

Operate on per-user storage providers. Sockets are looked for in `$XDG_RUNTIME_DIR/systemd/io.systemd.StorageProvider/`.

Use as a mount helper

The tool provides the `/sbin/mount.storage` alias, implementing the `mount(8)` "external helper" interface, allowing storage volumes to be mounted with the regular `mount` command. The volume to mount is encoded as the source of the mount, in the form `PROVIDER:VOLUME`, where `<PROVIDER>` is the name of a storage provider (as listed by `storagectl providers`) and `<VOLUME>` is the volume

name. Two file system type spellings are recognized:

`storage`

Acquires a directory volume and bind-mounts its directory tree onto the target.

`storage.FSTYPE`

Acquires a regular file or block device volume and mounts it as a file system of type `<FSTYPE>` (for example `storage.ext4`, `storage.btrfs`, ...).

The standard `-o` mount options are forwarded to `mount`. In addition, the following `storage.-` prefixed options are interpreted by `mount.storage` itself and stripped from the forwarded list:

`storage.create=<MODE>`

Takes one of `any` (open if it exists, otherwise create — the default), `open` (fail if the volume does not yet exist) or `new` (fail if the volume already exists).

`storage.template=<NAME>`

The template to use when creating a new volume, if it is missing and the provider supports on-the-fly creation of volumes.

`storage.create-size=<BYTES>`

When creating a new volume on-the-fly, the size in bytes to allocate. Accepts the usual `K/M/G/T` suffixes (base 1024). Required when creating a regular file volume.

Examples

Enumerate available storage providers, volumes and templates

```
$ storagectl providers
$ storagectl volumes
$ storagectl volumes '*foo*'
$ storagectl templates
```

Mount a directory volume from the file system provider

```
# mount -t storage fs:myvol /mnt/myvol
```

If the volume `myvol` does not yet exist, it will be created using the default `subvolume` template.

Create and mount an ext4 file system from a regular file.

```
# mount -t storage.ext4 fs:scratch /mnt/scratch -o loop
```

Mount a block device volume read-only

```
# mount -t storage.ext4 -o ro block:/dev/disk/by-id/usb-foo /mnt/foo
```

Exit status

On success, 0 is returned, a non-zero failure code otherwise.

See Also

`systemd(1)`, `systemd-storage-block@.service(8)`, `systemd-storage-fs@.service(8)`, `varlinkctl(1)`,
`mount(8)`

ⓘ ⓘ ⓘ #1

ⓘ ⓘ ⓘ 2026-10-31 14:58:46 UTC

ⓘ ⓘ ⓘ 2026-10-31 14:58:46 UTC