



Linux Ubuntu 26.04 Manual Pages on command 'docker-system-events.1'

\$ man docker-system-events.1

DOCKER(1)

Docker User Manuals

DOCKER(1)

NAME

docker-system-events - Get real time events from the server

SYNOPSIS

docker system events [OPTIONS]

DESCRIPTION

Get event information from the Docker daemon. Information can include historical information and real-time information.

Docker containers will report the following events:

attach, commit, copy, create, destroy, detach, die, exec_create, exec_detach, exec_start, export, kill, oom, pause, rename, resize, restart, start, stop, top, unpause, update

Docker images report the following events:

delete, import, load, pull, push, save, tag, untag

Docker volumes report the following events:

create, mount, unmount, destroy

Docker networks report the following events:

create, connect, disconnect, destroy

OPTIONS

The --since and --until parameters can be Unix timestamps, date formatted timestamps, or Go duration strings supported by `ParseDuration` ?<https://pkg.go.dev/time#ParseDuration>? (e.g.

10m, 1h30m) computed relative to the client machine's time. If you do not provide the

--since option, the command returns only new and/or live events. Supported formats for date

formatted time stamps include RFC3339Nano, RFC3339, 2006-01-02T15:04:05,

2006-01-02T15:04:05.999999999, 2006-01-02T07:00, and 2006-01-02. The local timezone on the client will be used if you do not provide either a Z or a +-00:00 timezone offset at the end of the timestamp. When providing Unix timestamps enter seconds[.nanoseconds], where seconds is the number of seconds that have elapsed since January 1, 1970 (midnight UTC/GMT), not counting leap seconds (aka Unix epoch or Unix time), and the optional .nanoseconds field is a fraction of a second no more than nine digits long.

EXAMPLES

Listening for Docker events

After running docker events a container 786d698004576 is started and stopped (The container name has been shortened in the output below):

```
# docker events
```

```
2015-01-28T20:21:31.000000000-08:00 59211849bc10: (from whenry/testimage:latest) start
2015-01-28T20:21:31.000000000-08:00 59211849bc10: (from whenry/testimage:latest) die
2015-01-28T20:21:32.000000000-08:00 59211849bc10: (from whenry/testimage:latest) stop
```

Listening for events since a given date

Again the output container IDs have been shortened for the purposes of this document:

```
# docker events --since '2015-01-28'
```

```
2015-01-28T20:25:38.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) create
2015-01-28T20:25:38.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) start
2015-01-28T20:25:39.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) create
2015-01-28T20:25:39.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) start
2015-01-28T20:25:40.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) die
2015-01-28T20:25:42.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) stop
2015-01-28T20:25:45.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) start
2015-01-28T20:25:45.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) die
2015-01-28T20:25:46.000000000-08:00 c21f6c22ba27: (from whenry/testimage:latest) stop
```

The following example outputs all events that were generated in the last 3 minutes, relative to the current time on the client machine:

```
# docker events --since '3m'
```

```
2015-05-12T11:51:30.999999999Z07:00 4386fb97867d: (from ubuntu:24.04) die
2015-05-12T15:52:12.999999999Z07:00 4386fb97867d: (from ubuntu:24.04) stop
2015-05-12T15:53:45.999999999Z07:00 7805c1d35632: (from redis:7.2) die
2015-05-12T15:54:03.999999999Z07:00 7805c1d35632: (from redis:7.2) stop
```

If you do not provide the `--since` option, the command returns only new and/or live events.

Format

If a format (`--format`) is specified, the given template will be executed instead of the default

format. Go's text/template package describes all the details of the format.

```
# docker events --filter 'type=container' --format 'Type={{.Type}} Status={{.Status}} ID={{.ID}}'
```

```
Type=container Status=create ID=2ee349dac409e97974ce8d01b70d250b85e0ba8189299c126a87812311951e26
```

```
Type=container Status=attach ID=2ee349dac409e97974ce8d01b70d250b85e0ba8189299c126a87812311951e26
```

```
Type=container Status=start ID=2ee349dac409e97974ce8d01b70d250b85e0ba8189299c126a87812311951e26
```

```
Type=container Status=resize ID=2ee349dac409e97974ce8d01b70d250b85e0ba8189299c126a87812311951e26
```

```
Type=container Status=die ID=2ee349dac409e97974ce8d01b70d250b85e0ba8189299c126a87812311951e26
```

```
Type=container Status=destroy ID=2ee349dac409e97974ce8d01b70d250b85e0ba8189299c126a87812311951e26
```

If a format is set to `{{json .}}`, the events are streamed as valid JSON Lines. For information

about JSON Lines, refer to <https://jsonlines.org>.

```
# docker events --format '{{json .}}'
```

```
{"status":"create","id":"196016a57679bf42424484918746a9474cd905dd993c4d0f4..
```

```
{"status":"attach","id":"196016a57679bf42424484918746a9474cd905dd993c4d0f4..
```

```
{"Type":"network","Action":"connect","Actor":{"ID":"1b50a5bf755f6021dfa78e..
```

```
{"status":"start","id":"196016a57679bf42424484918746a9474cd905dd993c4d0f42..
```

```
{"status":"resize","id":"196016a57679bf42424484918746a9474cd905dd993c4d0f4..
```

Filters

```
$ docker events --filter 'event=stop'
```

```
2014-05-10T17:42:14.999999999Z07:00 container stop 4386fb97867d (image=ubuntu:24.04)
```

```
2014-09-03T17:42:14.999999999Z07:00 container stop 7805c1d35632 (image=redis:7.2)
```

```
$ docker events --filter 'image=ubuntu:24.04'
```

```
2014-05-10T17:42:14.999999999Z07:00 container start 4386fb97867d (image=ubuntu:24.04)
```

```
2014-05-10T17:42:14.999999999Z07:00 container die 4386fb97867d (image=ubuntu:24.04)
```

```
2014-05-10T17:42:14.999999999Z07:00 container stop 4386fb97867d (image=ubuntu:24.04)
```

```
$ docker events --filter 'container=7805c1d35632'
```

```
2014-05-10T17:42:14.999999999Z07:00 container die 7805c1d35632 (image=redis:7.2)
```

```
2014-09-03T15:49:29.999999999Z07:00 container stop 7805c1d35632 (image=redis:7.2)
```

```
$ docker events --filter 'container=7805c1d35632' --filter 'container=4386fb97867d'
```

```
2014-09-03T15:49:29.999999999Z07:00 container die 4386fb97867d (image=ubuntu:24.04)
```

```
2014-05-10T17:42:14.999999999Z07:00 container stop 4386fb97867d (image=ubuntu:24.04)
```

```

2014-05-10T17:42:14.999999999Z07:00 container die 7805c1d35632 (image=redis:7.2)
2014-09-03T15:49:29.999999999Z07:00 container stop 7805c1d35632 (image=redis:7.2)
$ docker events --filter 'container=7805c1d35632' --filter 'event=stop'
2014-09-03T15:49:29.999999999Z07:00 container stop 7805c1d35632 (image=redis:7.2)
$ docker events --filter 'type=volume'
2015-12-23T21:05:28.136212689Z volume create test-event-volume-local (driver=local)
2015-12-23T21:05:28.383462717Z volume mount test-event-volume-local (read/write=true,
container=562fe10671e9273da25eed36cdce26159085ac7ee6707105fd534866340a5025, destination=/foo, driver=local,
propagation=rprivate)
2015-12-23T21:05:28.650314265Z volume unmount test-event-volume-local
(container=562fe10671e9273da25eed36cdce26159085ac7ee6707105fd534866340a5025, driver=local)
2015-12-23T21:05:28.716218405Z volume destroy test-event-volume-local (driver=local)
$ docker events --filter 'type=network'
2015-12-23T21:38:24.705709133Z network create
8b111217944ba0ba844a65b13efcd57dc494932ee2527577758f939315ba2c5b (name=test-event-network-local,
type=bridge)
2015-12-23T21:38:25.119625123Z network connect
8b111217944ba0ba844a65b13efcd57dc494932ee2527577758f939315ba2c5b (name=test-event-network-local,
container=b4be644031a3d90b400f88ab3d4bdf4dc23adb250e696b6328b85441abe2c54e, type=bridge)
$ docker events --filter 'type=plugin' (experimental)
2016-07-25T17:30:14.825557616Z plugin pull
ec7b87f2ce84330fe076e666f17dfc049d2d7ae0b8190763de94e1f2d105993f (name=tiborvass/sample-volume-plugin:latest)
2016-07-25T17:30:14.888127370Z plugin enable
ec7b87f2ce84330fe076e666f17dfc049d2d7ae0b8190763de94e1f2d105993f (name=tiborvass/sample-volume-plugin:latest)

```

OPTIONS

```

-f, --filter=      Filter output based on conditions provided
--format=""        Format output using a custom template: 'json':      Print in JSON
format 'TEMPLATE': Print output using the given Go template. Refer to
https://docs.docker.com/go/formatting/ for more information about formatting output with
templates
--since=""         Show all events created since timestamp
--until=""         Stream events until this timestamp

```

docker-system(1)

Docker Community

Jan 2026

DOCKER(1)