NAME

docker-network-ls - List networks

SYNOPSIS

docker network ls [OPTIONS]

DESCRIPTION

Lists all the networks the Engine **daemon** knows about. This includes the networks that span across multiple hosts in a cluster, for example:

```
$ docker network ls
NETWORK ID          NAME                DRIVER          SCOPE
7fca4eb8c647        bridge              bridge          local
9f904ee27bf5        none                null            local
cf03ee007fb4        host                host            local
78b03ee04fc4        multi-host          overlay         swarm
```

Use the **--no-trunc** option to display the full network id:

```
$ docker network ls --no-trunc
NETWORK ID                                                         NAME
18a2866682b85619a026c81b98a5e375bd33e1b0936a26cc497c283d27bae9b3   none
c288470c46f6c8949c5f7e5099b5b7947b07eabe8d9a27d79a9cbf111adcbf47   host
7b369448dccbf865d397c8d2be0cda7cf7edc6b0945f77d2529912ae917a0185   bridge
95e74588f40db048e86320c6526440c504650a1ff3e9f7d60a497c4d2163e5bd   foo
63d1ff1f77b07ca51070a8c227e962238358bd310bde1529cf62e6c307ade161   dev
```

Filtering

The filtering flag (**-f** or **--filter**) format is a **key=value** pair. If there is more than one filter, then pass multiple flags (e.g. **--filter "foo=bar" --filter "bif=baz"**). Multiple filter flags are combined as an**OR** filter . For example, **-f type=custom -f type=builtin** returns both **custom** and **builtin** networks.

The currently supported filters are:

- driver
- id (network's id)
- label (**label=<key>** or **label=<key>=<value>**)
- name (network's name)
- scope (**swarm|global|local**)
- type (custom|builtin)

Driver

The **driver** filter matches networks based on their driver.

The following example matches networks with the **bridge** driver:

```
$ docker network ls --filter driver=bridge
NETWORK ID          NAME                DRIVER
```

```
db9db329f835        test1               bridge
f6e212da9dfd        test2               bridge
```

ID

The **id** filter matches on all or part of a network's ID.

The following filter matches all networks with an ID containing the **63d1ff1f77b0...** string.

```
$ docker network ls --filter id=63d1ff1f77b07ca51070a8c227e962238358bd310bde1529cf62
NETWORK ID          NAME                DRIVER
63d1ff1f77b0        dev                 bridge
```

You can also filter for a substring in an ID as this shows:

```
$ docker network ls --filter id=95e74588f40d
NETWORK ID          NAME                DRIVER
95e74588f40d        foo                 bridge

$ docker network ls --filter id=95e
NETWORK ID          NAME                DRIVER
95e74588f40d        foo                 bridge
```

Label

The **label** filter matches networks based on the presence of a **label** alone or a **label** and a value.

The following filter matches networks with the **usage** label regardless of its value.

```
$ docker network ls -f "label=usage"
NETWORK ID          NAME                DRIVER
db9db329f835        test1               bridge
f6e212da9dfd        test2               bridge
```

The following filter matches networks with the **usage** label with the **prod** value.

```
$ docker network ls -f "label=usage=prod"
NETWORK ID          NAME                DRIVER
f6e212da9dfd        test2               bridge
```

Name

The **name** filter matches on all or part of a network's name.

The following filter matches all networks with a name containing the **foobar** string.

```
$ docker network ls --filter name=foobar
NETWORK ID          NAME                DRIVER
06e7eef0a170        foobar              bridge
```

You can also filter for a substring in a name as this shows:

```
$ docker network ls --filter name=foo
NETWORK ID          NAME                DRIVER
```

```
95e74588f40d        foo                 bridge
06e7eef0a170        foobar              bridge
```

Scope

The **scope** filter matches networks based on their scope.

The following example matches networks with the **swarm** scope:

```
$ docker network ls --filter scope=swarm
NETWORK ID          NAME                DRIVER              SCOPE
xbtm0v4f1lfh        ingress             overlay             swarm
ic6r88twuu92        swarmnet            overlay             swarm
```

The following example matches networks with the **local** scope:

```
$ docker network ls --filter scope=local
NETWORK ID          NAME                DRIVER              SCOPE
e85227439ac7        bridge              bridge              local
0ca0e19443ed        host                host                local
ca13cc149a36        localnet            bridge              local
f9e115d2de35        none                null                local
```

Type

The **type** filter supports two values; **builtin** displays predefined networks (**bridge**, **none**, **host**), whereas **custom** displays user defined networks.

The following filter matches all user defined networks:

```
$ docker network ls --filter type=custom
NETWORK ID          NAME                DRIVER
95e74588f40d        foo                 bridge
63d1ff1f77b0        dev                 bridge
```

By having this flag it allows for batch cleanup. For example, use this filter to delete all user defined networks:

```
$ docker network rm ‘docker network ls --filter type=custom -q‘
```

A warning will be issued when trying to remove a network that has containers attached.

Format

Format uses a Go template to print the output. The following variables are supported:

- .ID - Network ID
- .Name - Network name
- .Driver - Network driver
- .Scope - Network scope (local, global)
- .IPv6 - Whether IPv6 is enabled on the network or not

- .Internal - Whether the network is internal or not
- .Labels - All labels assigned to the network
- .Label - Value of a specific label for this network. For example **{{.Label "project.version"}}**

OPTIONS

-f, **--filter**= Provide filter values (e.g. "driver=bridge")

--format="" Format output using a custom template: 'table': Print output in table format with column headers (default) 'table TEMPLATE': Print output in table format using the given Go 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

--no-trunc[=false] Do not truncate the output

-q, **--quiet**[=false] Only display network IDs

SEE ALSO

docker-network(1)