docker create

Estimated reading time: 11 minutes

Description

Create a new container

Usage

```
docker create [OPTIONS] IMAGE [COMMAND] [ARG...]
```

Options

Name, shorthand	Default	Description
`--add-host`		Add a custom host-to-IP mapping (host:ip)
`--attach , -a`		Attach to STDIN, STDOUT or STDERR
`--blkio-weight`		Block IO (relative weight), between 10 and 1000, or 0 to disable (default 0)
`--blkio-weight-device`		Block IO weight (relative device weight)
`--cap-add`		Add Linux capabilities
`--cap-drop`		Drop Linux capabilities
`--cgroup-parent`		Optional parent cgroup for the container
`--cidfile`		Write the container ID to the file
`--cpu-count`		CPU count (Windows only)
`--cpu-percent`		CPU percent (Windows only)
`--cpu-period`		Limit CPU CFS (Completely Fair Scheduler) period
`--cpu-quota`		Limit CPU CFS (Completely Fair Scheduler) quota
`--cpu-rt-period`		API 1.25+ (https://docs.docker.com/engine/api/v1.25/) Limit CPU real-time period in microseconds
`--cpu-rt-runtime`		API 1.25+ (https://docs.docker.com/engine/api/v1.25/) Limit CPU real-time runtime in microseconds
`--cpu-shares , -c`		CPU shares (relative weight)
`--cpus`		API 1.25+ (https://docs.docker.com/engine/api/v1.25/) Number of CPUs
`--cpuset-cpus`		CPUs in which to allow execution (0-3, 0,1)
`--cpuset-mems`		MEMs in which to allow execution (0-3, 0,1)

Name, shorthand	Default	Description
`--device`		Add a host device to the container
`--device-cgroup-rule`		Add a rule to the cgroup allowed devices list
`--device-read-bps`		Limit read rate (bytes per second) from a device
`--device-read-iops`		Limit read rate (IO per second) from a device
`--device-write-bps`		Limit write rate (bytes per second) to a device
`--device-write-iops`		Limit write rate (IO per second) to a device
`--disable-content-trust`	`true`	Skip image verification
`--dns`		Set custom DNS servers
`--dns-opt`		Set DNS options
`--dns-option`		Set DNS options
`--dns-search`		Set custom DNS search domains
`--entrypoint`		Overwrite the default ENTRYPOINT of the image
`--env` , `-e`		Set environment variables
`--env-file`		Read in a file of environment variables
`--expose`		Expose a port or a range of ports
`--group-add`		Add additional groups to join
`--health-cmd`		Command to run to check health
`--health-interval`		Time between running the check (ms|s|m|h) (default 0s)
`--health-retries`		Consecutive failures needed to report unhealthy
`--health-start-period`		API 1.29+ (https://docs.docker.com/engine/api/v1.29/) Start period for the container to initialize before starting health-retries countdown (ms|s|m|h) (default 0s)
`--health-timeout`		Maximum time to allow one check to run (ms|s|m|h) (default 0s)
`--help`		Print usage
`--hostname` , `-h`		Container host name
`--init`		API 1.25+ (https://docs.docker.com/engine/api/v1.25/) Run an init inside the container that forwards signals and reaps processes
`--interactive` , `-i`		Keep STDIN open even if not attached
`--io-maxbandwidth`		Maximum IO bandwidth limit for the system drive (Windows only)
`--io-maxiops`		Maximum IOps limit for the system drive (Windows only)
`--ip`		IPv4 address (e.g., 172.30.100.104)
`--ip6`		IPv6 address (e.g., 2001:db8::33)
`--ipc`		IPC mode to use

Name, shorthand	Default	Description
`--isolation`		Container isolation technology
`--kernel-memory`		Kernel memory limit
`--label` , `-l`		Set meta data on a container
`--label-file`		Read in a line delimited file of labels
`--link`		Add link to another container
`--link-local-ip`		Container IPv4/IPv6 link-local addresses
`--log-driver`		Logging driver for the container
`--log-opt`		Log driver options
`--mac-address`		Container MAC address (e.g., 92:d0:c6:0a:29:33)
`--memory` , `-m`		Memory limit
`--memory-reservation`		Memory soft limit
`--memory-swap`		Swap limit equal to memory plus swap: '-1' to enable unlimited swap
`--memory-swappiness`	`-1`	Tune container memory swappiness (0 to 100)
`--mount`		Attach a filesystem mount to the container
`--name`		Assign a name to the container
`--net`		Connect a container to a network
`--net-alias`		Add network-scoped alias for the container
`--network`		Connect a container to a network
`--network-alias`		Add network-scoped alias for the container
`--no-healthcheck`		Disable any container-specified HEALTHCHECK
`--oom-kill-disable`		Disable OOM Killer
`--oom-score-adj`		Tune host's OOM preferences (-1000 to 1000)
`--pid`		PID namespace to use
`--pids-limit`		Tune container pids limit (set -1 for unlimited)
`--platform`		experimental (daemon) (https://docs.docker.com/engine/reference/commandline/dockerd/#daemon-configuration-file) API 1.32+ (https://docs.docker.com/engine/api/v1.32/) Set platform if server is multi-platform capable
`--privileged`		Give extended privileges to this container
`--publish` , `-p`		Publish a container's port(s) to the host
`--publish-all` , `-P`		Publish all exposed ports to random ports
`--read-only`		Mount the container's root filesystem as read only
`--restart`	`no`	Restart policy to apply when a container exits

Name, shorthand	Default	Description
`--rm`		Automatically remove the container when it exits
`--runtime`		Runtime to use for this container
`--security-opt`		Security Options
`--shm-size`		Size of /dev/shm
`--stop-signal`	`SIGTERM`	Signal to stop a container
`--stop-timeout`		API 1.25+ (https://docs.docker.com/engine/api/v1.25/) Timeout (in seconds) to stop a container
`--storage-opt`		Storage driver options for the container
`--sysctl`		Sysctl options
`--tmpfs`		Mount a tmpfs directory
`--tty` , `-t`		Allocate a pseudo-TTY
`--ulimit`		Ulimit options
`--user` , `-u`		Username or UID (format: <name|uid>[:<group|gid>])
`--userns`		User namespace to use
`--uts`		UTS namespace to use
`--volume` , `-v`		Bind mount a volume
`--volume-driver`		Optional volume driver for the container
`--volumes-from`		Mount volumes from the specified container(s)
`--workdir` , `-w`		Working directory inside the container

Parent command

Command	Description
docker (https://docs.docker.com/engine/reference/commandline/docker)	The base command for the Docker CLI.

Extended description

The `docker create` command creates a writeable container layer over the specified image and prepares it for running the specified command. The container ID is then printed to `STDOUT` . This is similar to `docker run -d` except the container is never started. You can then use the `docker start <container_id>` command to start the container at any point.

This is useful when you want to set up a container configuration ahead of time so that it is ready to start when you need it. The initial status of the new container is `created` .

Please see the run command (https://docs.docker.com/engine/reference/commandline/run/) section and the Docker run reference (https://docs.docker.com/engine/reference/run/) for more details.

Examples

Create and start a container

```
$ docker create -t -i fedora bash

6d8af538ec541dd581ebc2a24153a28329acb5268abe5ef868c1f1a261221752

$ docker start -a -i 6d8af538ec5

bash-4.2#
```

Initialize volumes

As of v1.4.0 container volumes are initialized during the `docker create` phase (i.e., `docker run` too). For example, this allows you to `create` the `data` volume container, and then use it from another container:

```
$ docker create -v /data --name data ubuntu

240633dfbb98128fa77473d3d9018f6123b99c454b3251427ae190a7d951ad57

$ docker run --rm --volumes-from data ubuntu ls -la /data

total 8
drwxr-xr-x  2 root root 4096 Dec  5 04:10 .
drwxr-xr-x 48 root root 4096 Dec  5 04:11 ..
```

Similarly, `create` a host directory bind mounted volume container, which can then be used from the subsequent container:

```
$ docker create -v /home/docker:/docker --name docker ubuntu

9aa88c08f319cd1e4515c3c46b0de7cc9aa75e878357b1e96f91e2c773029f03

$ docker run --rm --volumes-from docker ubuntu ls -la /docker

total 20
drwxr-sr-x  5 1000 staff  180 Dec  5 04:00 .
drwxr-xr-x 48 root root  4096 Dec  5 04:13 ..
-rw-rw-r--  1 1000 staff 3833 Dec  5 04:01 .ash_history
-rw-r--r--  1 1000 staff  446 Nov 28 11:51 .ashrc
-rw-r--r--  1 1000 staff   25 Dec  5 04:00 .gitconfig
drwxr-sr-x  3 1000 staff   60 Dec  1 03:28 .local
-rw-r--r--  1 1000 staff  920 Nov 28 11:51 .profile
drwx--S---  2 1000 staff  460 Dec  5 00:51 .ssh
drwxr-xr-x 32 1000 staff 1140 Dec  5 04:01 docker
```

Set storage driver options per container.

```
$ docker create -it --storage-opt size=120G fedora /bin/bash
```

This (size) will allow to set the container rootfs size to 120G at creation time. This option is only available for the `devicemapper`, `btrfs`, `overlay2`, `windowsfilter` and `zfs` graph drivers. For the `devicemapper`, `btrfs`, `windowsfilter` and `zfs` graph drivers, user cannot pass a size less than the Default BaseFS Size. For the `overlay2` storage driver, the size option is only available if the backing fs is `xfs` and mounted with the `pquota` mount option. Under these conditions, user can pass any size less than the backing fs size.

Specify isolation technology for container (--isolation)

This option is useful in situations where you are running Docker containers on Windows. The `--isolation=<value>` option sets a container's isolation technology. On Linux, the only supported is the `default` option which uses Linux namespaces. On Microsoft Windows, you can specify these values:

Value	Description
`default`	Use the value specified by the Docker daemon's `--exec-opt` . If the `daemon` does not specify an isolation technology, Microsoft Windows uses `process` as its default value if the
daemon is running on Windows server, or `hyperv` if running on Windows client.	
`process`	Namespace isolation only.
`hyperv`	Hyper-V hypervisor partition-based isolation.

Specifying the `--isolation` flag without a value is the same as setting `--isolation="default"` .

Dealing with dynamically created devices (--device-cgroup-rule)

Devices available to a container are assigned at creation time. The assigned devices will both be added to the cgroup.allow file and created into the container once it is run. This poses a problem when a new device needs to be added to running container.

One of the solution is to add a more permissive rule to a container allowing it access to a wider range of devices. For example, supposing our container needs access to a character device with major `42` and any number of minor number (added as new devices appear), the following rule would be added:

```
docker create --device-cgroup-rule='c 42:* rmw' -name my-container my-image
```

Then, a user could ask `udev` to execute a script that would `docker exec my-container mknod newDevX c 42 <minor>` the required device when it is added.

NOTE: initially present devices still need to be explicitly added to the create/run command