# Docker Commands

Dockers documentation is very good and has plenty of examples. You can find a detailed description for each command, e.g docker run

Your docker cli also has help built-in.

```
# Docker built-in help
$ docker --help

# You can get help for a specific command with
$ docker run --help
```

## Docker on windows

If you use docker in git bash, you may need to run it with `winpty` if it says `the input device is not a TTY.`

```
# So use winpty before docker (git bash on windows only)
winpty docker run -d -P nginx:alpine
```

## Docker exit status

When your docker command fails with a exit status != 0, it can originate from e.g the docker daemon, a wrongful command inside the container or the command inside the container.

If the docker daemon is not stared

```
$ docker run nginx:alpine; echo $?

docker: Cannot connect to the Docker daemon at
unix:///var/run/docker.sock. Is the docker daemon running?.
See 'docker run --help'.
125
```

If the command is not found within the container

```
$ docker run nginx:alpine do_something; echo $?
/docker-entrypoint.sh: exec: line 38: do_something: not found
127
```

If the command ends with a exit status inside the container

```
$ docker run alpine /bin/sh -c 'exit 3'; echo $?
3
```

## Docker container

### Docker container run

Usage: docker container run [OPTIONS] IMAGE [COMMAND] [ARG...]

Run a command in a new container

| Options | Description |
|---|---|
| -d, --detach | Run container in background and print container ID |
| -i, --interactive | Keep STDIN open even if not attached |
| --name string | Assign a name to the container |
| -p, --publish | Publish a container's port(s) to the host |
| -P, --publish-all | Publish all exposed ports to random ports |
| --rm | Automatically remove the container when it exits |
| -t, --tty | Allocate a pseudo-TTY |

Read more about nginx at Docker Hub nginx)

```
# Run a container from Docker Hub detached and with random ports
$ docker container run -d -P nginx:alpine

# Run a container from Docker Hub detached and with the specific port
published
$ docker container run -d -p 2222:80 nginx:alpine

# You test that nginx page is accessible with
$ curl localhost:2222

# Run a container and remove it at exit
$ docker container run --rm IMAGE

# Run a container with input and tty, remove the container when it exits
$ docker container run --rm -it alpine sh
```

### Docker container list

Usage: docker container ls [OPTIONS]

List containers

| Options | Description |
|---|---|
| -a, --all | Show all containers (default shows just running) |
| -l, --latest | Show the latest created container |
| -q, --quiet | Only display container IDs |

```
# List running containers
$ docker container ls

# List all containers (including stopped)
$ docker container ls --all

# List latest container id (including stopped)
$ docker container ls -lq
```

### Docker container stop

Usage: docker container stop [OPTIONS] CONTAINER [CONTAINER...]

Stop one or more running containers

```
# Stop a running docker container
$ docker stop CONTAINER

# Stop all running containers (linux/mac)
$ docker stop $(docker ps -q)
```

### Docker container start

Usage: docker container start [OPTIONS] CONTAINER [CONTAINER...]

Start one or more stopped containers

```
# Start a stopped container
$ docker container start CONTAINER
```

### Docker container exec

Usage: docker container exec [OPTIONS] CONTAINER COMMAND [ARG...]

Run a command in a running container

| Options | Description |
|---|---|
| -i, --interactive | Keep STDIN open even if not attached |
| -t, --tty | Allocate a pseudo-TTY |

```
# Open a shell on a running container
$ docker exec -it CONTAINER sh

# List a folder on a running container
$ docker exec -it CONTAINER ls /usr/bin
```

## Docker build

Usage: docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile

| Options | Description |
|---|---|
| -f, --file string | Name of the Dockerfile (Default is 'PATH/Dockerfile') |
| --no-cache | Do not use cache when building the image |
| --pull | Always attempt to pull a newer version of the image |
| -t, --tag list | Name and optionally a tag in the 'name:tag' format |

```
# Build a image used locally
$ docker built -t NAME .

# Build a image that can be pushed to remote registry
$ docker build -t registry.gitlab.com/username/project-name .
```

## Docker push

Usage: docker push [OPTIONS] NAME[:TAG]

Push an image or a repository to a registry

```
# Before pushing to a remote registry you must first log in (once).

# Enter your username when prompted and then your password or personal
access token
$ docker login registry.gitlab.com

# Push build image to gitlab container registry
$ docker push registry.gitlab.com/username/project-name
```

## Docker Cleanup

```
# Basic cleanup
$ docker system prune

# To remove all unused images add -a
$ docker system prune -a

# To remove everything unused
$ docker system prune -a --volumes
WARNING! This will remove:
  - all stopped containers
  - all networks not used by at least one container
  - all volumes not used by at least one container
  - all images without at least one container associated to them
  - all build cache
```