Docker Cheatsheet

Stefan Hornburg (Racke)

Contents

Images

Building images

```
$ docker build -t myimage:0.001 .
```

Use a different Dockerfile:

```
$ docker build -t myimage:0.001 -f Dockerfile.myimage .
```

Skipping cache:

```
$ docker build --no-cache -t myimage:0.001 .
```

Prune images

Delete all images without an associated container:

```
$ docker image prune
```

Inspect images

In order to inspect the images, you may run them as container with a shell command - usually *bash* but *sh* for Alpine based ones:

```
$ docker run -it myimage:0.001 bash
```

Containers

Run from image

```
$ docker run --name sympa -d -i sympa:6.2.48
```

Explanation:

-d detached

-i image tag

–**name** name of the container (optional)

You can instruct Docker to automatically remove the container when it is terminated by adding the `--rm` flag:

```
$ docker run --rm --name sympa -d -i sympa:6.2.48
```

Please note that this also removes the logs for the container.
Environment variables can be passed to the container with the `--env` flag:

```
$ docker run --env SYMPA_DB_USER=sympa --env SYMPA_DB_PASSWORD=sympa --name sympa -d -i sympa:6.2.48
```

Remove and prune

Remove container

```
docker rm sympa
```

Please note that the corresponding image isn't affected.

Prune containers

Remove all stopped containers:

```
$ docker container prune
```

Remove all stopped containers older than a day:

```
$ docker container prune --filter "until=24h"
```

Shell on a running container

```
$ docker exec -it 593d9b3219c6 bash
```

You can also pass environment variables to the command executed in the Docker container:

```
$ docker exec -e SPACEWALK_API_USER=admin -e SPACEWALK_API_PASS=nevairbe -it spacewalk /bin/setup-sp
```

Note: the flags for the environment preceed the flag for the container.

Copy files from/to container

From:

```
$ docker cp wp-site:/var/www/html/wp-config.php .
```

To:

```
$ docker cp  ./setup-spacecmd.sh spacewalk2:/bin
```

Logs

To see the log messages for the container *nginx*:

```
$ docker logs nginx
```

However, this shows only the messages without any further information.
You can add the timestamp with the `--timestamps` commandline option:

```
$ docker logs --timestamps nginx
```

Show the latest 30 entries:

```
$ docker logs --tail 30 nginx
```

Follow new entries:

```
$ docker logs --tail 0 nginx --follow
```

Container Information

IP address

To get the IP address of the container *perldancer*:

```
$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' perldancer
```

Determine whether a container is running properly

This checks whether the container with the name *sympa* is active **and** running, e.g. not restarting because of a problem with the command it executes on startup.

```
$ docker ps -q --filter=name='/sympa$' --filter=status=running
```

Startup problems

- Missing environment variables, e.g. `MYSQL_ROOT_PASSWORD` for *mysql* container

Compose

Run containers and detach:

```
$ docker-compose up -d
```

Proxy

Configure proxy in `/etc/docker/daemon.json`:

```
{
  "proxies": {
    "http-proxy": "http://10.11.12.13:3128/",
    "https-proxy": "http://10.11.12.13:3128/",
    "no-proxy": ".example.org"
  }
}
```

Linuxia Wiki

Stefan Hornburg (Racke)
Docker Cheatsheet

wiki.linuxia.de