TMF – Technologie- und Methodenplattform
für die vernetzte medizinische Forschung e.V.

Dockerbank

Container-basiertes Deployment von biomedizinischen IT-Lösungen

Praktische Übungen 1: Installation der Docker-Plattform und Nutzung vorhandener Container

Sebastian Stäubert[1], Benjamin Baum[2]

[1] *Institut für Medizinische Informatik, Statistik und Epidemiologie, Universität Leipzig*

[2] *Abteilung Medizinische Informatik, Universitätsmedizin Göttingen*

Praktische Übungen 1
a) Installation der Docker-Plattform

Inhalt

- Installation unter Windows
- Installation unter Linux
- Verwendung der Docker TMF VM

Installation unter Windows[1]

a) Docker for Windows (Native Installation)

Voraussetzungen: Windows 10 Enterprise/Pro/Edu 64Bit, Hyper-V!

- Anleitung unter https://docs.docker.com/docker-for-windows/
- Docker steht anschließend nativ auf der Kommandozeile zur Verfügung

b) Docker Toolbox

Voraussetzungen: Windows 7 64Bit oder neuer, Virtualisierung aktiviert

- Anleitung unter https://docs.docker.com/toolbox/toolbox_install_windows/ bzw. 2016-03-02-Docker-Kurzanleitung.pdf[2]
  - Docker Client
  - Docker Toolbox management tool and ISO
  - Oracle VM VirtualBox
  - Git MSYS-git UNIX tools
- Docker läuft anschließend in einer VM
- Docker Toolbox terminal

*https://docs.docker.com/engine/installation/windows/

Installation unter Linux*

Voraussetzung: 64Bit, Kernel 3.10, Ubuntu ab v12

Paketinfos aktualisieren:

```
$ sudo apt-get update
```

1. `$ sudo apt-get install apt-transport-https ca-certificates`
2. GPG key hinzufügen:
3. `$ sudo apt-key adv`
4. `--keyserver hkp://p80.pool.sks-keyservers.net:80`
5. `--recv-keys 58118E89F3A912897C070ADBF76221572C52609D`
6. Paketquelle hinzufügen:
7. `$ echo „deb https://apt.dockerproject.org/repo ubuntu-xenial main“ > /etc/apt/sources.list.d/docker.list`
8. Paketinfos aktualisieren und docker installieren:
9. `$ sudo apt-get update`
10. `$ sudo apt-get install docker-engine`
11. Docker Daemon starten:
12. `$sudo service docker start`

* https://docs.docker.com/engine/installation/linux/ubuntulinux/
https://docs.docker.com/engine/installation/linux/debian/

Docker TMF VM

Docker TMF

- Voraussetzungen:
  - Oracle VirtualBox
  - Docker TMF VM
  - PgAdmin3 (Optional) https://www.pgadmin.org/
  - Putty (Optional) http://www.putty.org/
  - Webbrowser

TMF Docker VM

- Login: **docker/root**
- Password: **docker**
- Root werden: **sudo -s**
- SSH (putty): **localhost:5022**

- Vorbereitete Übungen unter: **/docker/excercise**
- Lösungen unter: **/docker/solution**
- Cheatsheet **/docker/dockercheatsheet.txt**

Praktische Übungen 1
b) Nutzung vorhandener Container

Inhalt

- Docker-Hub & Suche nach Images: docker search
- Herunterladen von Images: docker pull
- Ausführen von Containern: docker run
- Informationen zu Containern:
  docker info / ps / images / inspect
  docker logs / top / history
- Container starten & stoppen: docker start / stop / restart
- Interaktion mit Containern: docker exec / (attach) / cp
- Container & Images löschen: docker rm / rmi
- docker save / import / commit / login / push

Docker-Hub* (I)

→ https://hub.docker.com/

= Cloud-basierte Registry für die Verteilung von Docker Images

Die wichtigsten Funktionen sind:

Image Repositories: Auffinden, Verwalten, sowie Hoch- und Herunterladen von Images

Automated Builds: Automatische Erzeugung von Images nach Änderungen in einem verknüpften Sourcecode Repository (GitHub, Bitbucket)

Webhooks: Ermöglicht Aktionen nach einem erfolgreichen Automated Buils

Organizations: Gruppen zur Verwaltung des Zugriffs

* https://hub.docker.com/
https://docs.docker.com/docker-hub/

Docker-Hub* (II)

Suche nach Images:

a) Über https://hub.docker.com/
   - Auch ohne Anmeldung möglich
   - „official“ Images bevorzugen

b) Über die Console:
   - $ `docker search ubuntu`

```
root@hummel:~# docker search ubuntu
NAME                      DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
ubuntu                    Ubuntu is a Debian-based Linux operating s...   4656      [OK]
ubuntu-upstart            Upstart is an event-based replacement for ...   66        [OK]
rastasheep/ubuntu-sshd    Dockerized SSH service, built on top of of...   39                   [OK]
```

Informationen zu Images:

z.B. der TMF Docker Images: https://hub.docker.com/r/tmfev/

* https://hub.docker.com/
https://docs.docker.com/docker-hub/

Docker Kommando Referenz

→ https://docs.docker.com/engine/reference/commandline/

Darüber hinaus gibt es:

- Howtos (Docker Website, „Internet“, usw.)
- Cheatsheets (→ google Suche)
- Zeitschriftenbeiträge (Admin Magazin, c‘t, uvm.)
- Beispiele (Docker Website)
- uvm. (→ Wiki-Seiten?)

Docker Images herunterladen

Syntax:

```
docker pull IMAGE:TAG
```

Bsp.:

```
docker pull hello-world:latest
```

Aufgabe:

Suchen und laden Sie ein Image der minimal-Shell „busybox“.

* https://docs.docker.com/engine/reference/commandline/pull/

Docker Container ausführen (I)

Syntax:

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

→ Container erzeugen

```
docker run [OPTIONS] IMAGE [COMMAND] [ARG…]
```

→ Container erzeugen und starten

OPTIONS:

- Namen vergeben: `--name <name>`
- Start im Hintergrund: `-d`
- Interaktiv starten: `-it IMAGE /bin/bash`
- Volumes einbinden: `-v <path-host>:<path-container>`
- `--volumes-from <container>`
- Workdir setzen: `-w <path-container>`
- Ports mappen: `-p <ip-host>:<port-host>:<port-container>`
- Variablen übergeben: `-e <VarName>:<VarValue>`
- Container verknüpfen: `--link`
- Temp. Container: `--rm`
- uvm.

* https://docs.docker.com/engine/reference/commandline/run/
https://docs.docker.com/engine/reference/commandline/create/

Docker Container ausführen – docker run* (II)

Bsp.:

```
docker run --name test-nginx -v /var/www -p 80:80 nginx
→ http://localhost:5080/
```

Aufgabe:
Erzeugen Sie einen Container auf Basis des Images *hello-world* und benennen Sie den Container mit hello-test.

Führen Sie das obige Beispiel aus und lassen Sie sich die durch den Container bereit gestellte Webseite anzeigen.

* https://docs.docker.com/engine/reference/commandline/run/

Docker Container Status und Infos

Syntax:

```
docker info
docker images [-a]
docker ps [-a]
docker inspect <container-ID|container-Name>
```

Bsp.:

```
docker inspect hello-test
```

Aufgabe:

Finden Sie heraus, wann der Container hello-test erzeugt wurde.

* https://docs.docker.com/engine/reference/commandline/info/
  https://docs.docker.com/engine/reference/commandline/ps/
  https://docs.docker.com/engine/reference/commandline/inspect/

Docker Container Status und Infos

Syntax:

```
docker top <container-ID|container-Name>
docker logs -f <container-ID|container-Name>
docker history IMAGE
```

Bsp.:

```
docker logs hello-test
```

Aufgabe:

Lassen Sie sich die Log-Ausgabe eines laufenden Containers dauerhaft anzeigen.

* https://docs.docker.com/engine/reference/commandline/top/
  https://docs.docker.com/engine/reference/commandline/logs/
  https://docs.docker.com/engine/reference/commandline/history/

Docker Container starten, stoppen

Syntax:

```
docker start <container-ID|container-Name>
docker stop <container-ID|container-Name>
docker restart <container-ID|container-Name>
```

Bsp.:

```
docker stop hello-test
```

Aufgabe:

Stoppen und Starten Sie einen vorhandenen Container.

* https://docs.docker.com/engine/reference/commandline/start/
  https://docs.docker.com/engine/reference/commandline/restart/
  https://docs.docker.com/engine/reference/commandline/stop/

Commando in einem Docker Container ausführen

Syntax:

```
docker cp [container-name]<src> [container-name]<dest>
docker exec [-it] <container-ID|container-Name> Command
```

Bsp.:

```
docker cp /var/log/dmesg hello-test:/root/
```

Aufgabe:

Kopieren Sie eine Datei in einen laufenden Container, loggen Sie sich anschließend in den Container ein und lassen Sie sich die Datei anzeigen.

* https://docs.docker.com/engine/reference/commandline/cp/
https://docs.docker.com/engine/reference/commandline/exec/

Docker Container und Images löschen

Syntax:

```
docker rm [-f | -v] <container-ID|container-Name>
docker rmi [-f] IMAGE
```

Bsp.:

```
docker rm -f hello-test
```

Aufgabe:

Löschen Sie den von Ihnen erzeugten nginx Container.

Löschen Sie das hello-world Image.

* https://docs.docker.com/engine/reference/commandline/rm/
https://docs.docker.com/engine/reference/commandline/rmi/

Änderungen abspeichern und hochladen

Syntax:

```
docker save [--output] IMAGE [IMAGE…]
docker import [OPTIONS] file|URL|- [REPOSITORY[:TAG]]
  → Container speichern und importieren
docker commit [OPTIONS] CONTAINER [REPOSITORY[:TAG]]
  → Container in einem neuen Image speichern
docker login
docker push Login-Name/NAME[:TAG]
  → Docker-Hub Login und hochladen von Images
```

Bsp.:

```
docker save --output busybox.tar busybox
docker import busybox.tar
```

* https://docs.docker.com/engine/reference/commandline/save/
https://docs.docker.com/engine/reference/commandline/import/
https://docs.docker.com/engine/reference/commandline/commit/
https://docs.docker.com/engine/reference/commandline/login/
https://docs.docker.com/engine/reference/commandline/push/

Vielen Dank für Ihre Aufmerksamkeit!

sebastian.staeubert@imise.uni-leipzig.de
Benjamin.Baum@med.uni-goettingen.de