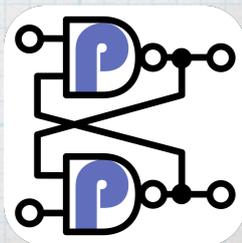


# Docker Crash Kurs

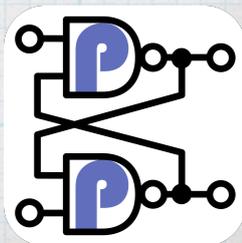
Linuxwochen Wien 2018

Peter Pfläging  
[pflaeging.net](http://pflaeging.net)  
[peter@pflaeging.net](mailto:peter@pflaeging.net)



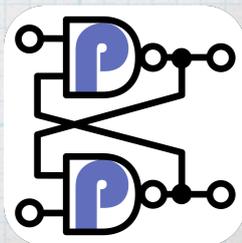
# Vorwort

- \* Über mich: Peter Pfläging
  - \* IT-Architect
  - \* Fotograf
  - \* Musiker
  - \* System Programmier
  - \* Cloud Services (<https://www.stickiebox.org>)
- \* Das ist ein Workshop!
  - \* Mitarbeit erwünscht
  - \* Variabler Inhalt



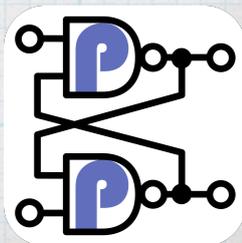
# Agenda

- \* Was ist eigentlich Docker?
- \* Vorteile / Nachteile
- \* Schnellstart in die Docker Welt
- \* Prinzipien
- \* Tools
- \* Höherwertige Dienste



# Was ist eigentlich Docker?

- \* Container in Linux / Windows / OSX
  - \* Keine Virtualisierung!
- \* Container sind klein und atomar:
  - \* EIN Service ergibt EINEN Container
- \* Build
  - \* From Source
  - \* Layered
- \* Secure by design
  - \* Basiert auf CGroups



- \* Inhalt:

- \* „Mini“ Linux

- \* Eine Applikation

- \* Pakete

- \* Abgrenzung:

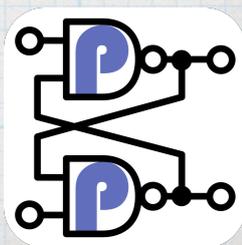
- \* Eigenes virtuelles Filesystem

- \* Eigene Netzwerkkonfiguration / virt.

- \* Nur dedizierte Ports zugänglich

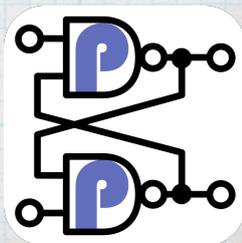
- \* Container laufen überall ( i386 / x86\_64 / arm / ...)

- \* Aussehen immer gleich ( Develop / Operation)



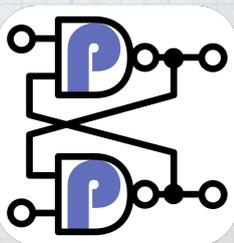
# Vorteile

- \* Isolation von Applikationen
  - \* Ohne Virtualisierungsoverhead
- \* Einheitliche Pakete über alle Linuxe ;-)
- \* Security
  - \* Separation & Isolation
- \* Gleiche Environments von
  - \* Dev -> Test -> QM -> Production
- \* Schnelle Update möglich



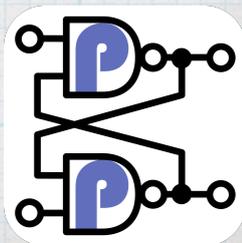
# Nachteile

- \* Neues Paradigma
  - \* KEINE Virtualisierung
- \* Ideal nur bei „neueren“ Applikationen
- \* Deklarative Umgebungen sind aufwändiger
- \* Architekturelles Umdenken beim Aufbau der Strukturen und Appl. notwendig



# Schnellstart in die Docker Welt

- \* Installation auf allen Systemen parallel möglich
- \* Empfohlen: CentOS minimal
- \* **ACHTUNG:** Firewalld / SELinux!



- \* Installation „Docker CE“

- \* <https://docs.docker.com/install/linux/docker-ce/centos/>

- \* Wir starten das Zeug

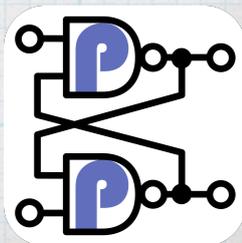
- \* Kurzüberblick Commandline

- \* `docker ps`

- \* `docker version`

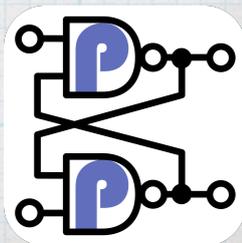
- \* `docker images`

- \* `docker start / restart / stop / logs`



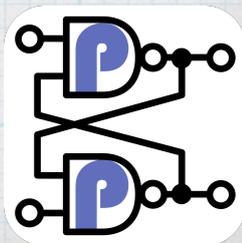
# Prinzipien

- \* Ein Container basiert auf einem Image
  - \* Immer nur eine Applikation im Container
- \* Images bestehen aus Images (Layer)
- \* Ein Container kann jederzeit entsorgt werden
  - \* Auf Persistenz achten!
  - \* Sicherheit!
- \* Container haben virtuelle Netze zwischen und mit dem Host
  - \* eigenes DNS
  - \* eigenes DHCP



# Tools

- \* Docker-compose
  - \* Einfachste Art mehrere Container zu verbinden und zu koppeln
  - \* <https://docs.docker.com/compose/>
- \* Docker bash\_completion rules
  - \* CentOS: `yum install bash-completion bash-completion-extra`
- \* Portainer als Web-GUI für lokalen Docker
  - \* <https://portainer.io>
- \* Services selbst erstellen mit:
  - \* Dockerfiles (z.B.: <https://github.com/pflaeging/django-docker>)



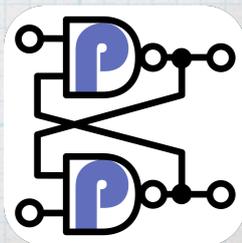
- \* Docker Registry

- \* <https://hub.docker.com>

- \* Oder Eigene!

- \* Beispiel:

- \* `docker run -p 18080:8000 --name django pflaeing/django-docker:0.6.4`



# Höherwertige Dienste

1. Docker

2. Portainer

3. Docker-Compose / Docker-Swarm

4. Rancher / Mesa

5. Kubernetes

6. OpenShift

