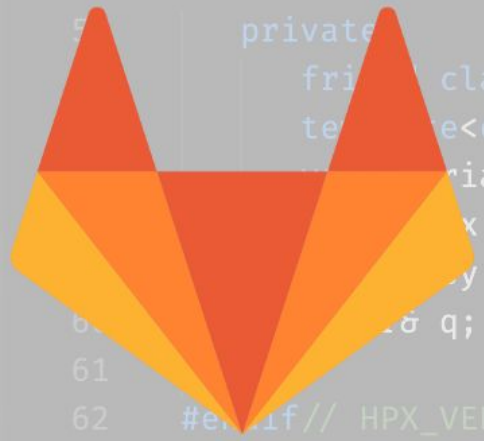
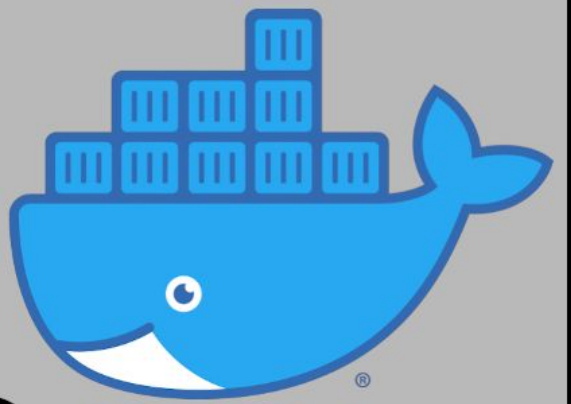


Gitlab Continuous Integration (CI) einrichten



Was erwartet Euch in diesem Video?

Eine kleine Agenda

1. Was ist Continuous Integration (CI) überhaupt? Was ist der Verwendungszweck?
2. Wie funktioniert eine CI Pipeline?
3. Was benötige ich für eine CI Pipeline in Gitlab? Welche Schritte sind für notwendig?
4. Beispiel Projekt mit Python
5. Server Konfiguration

Was ist Continuous Integration (CI)?

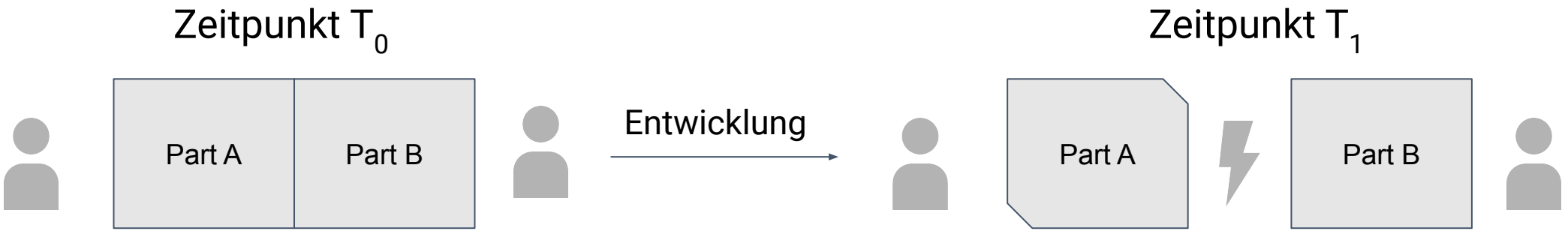
Continuous Integration (CI) verbessert die Software Qualität

Automatisierung

</>

Software Entwicklung

Beispiel



Wie funktioniert eine CI Pipeline?

Eine CI Pipeline kann individuell gestaltet werden

Implementierung von neuer Funktion

Code wird “gepushed” (Git)

CI Pipeline startet die Tests, um die Funktionalitäten des Programms zu überprüfen

Falls die Tests nicht funktionieren, wird z.B. eine Email an den Entwickler verschickt

Was benötigt ihr für eine CI Pipeline?

Ein eigener Server ist nicht immer notwendig

Grundlegendes

- gitlab Account
- Git Know How
- Software Projekt
- Tests o.ä. zum überprüfen

Option A

- Aufsetzen eines eigenen Servers
- Serverkapazitäten (bspw. RAM) sind der limitierende Faktor
- keine Zeitlimitierung in Nutzung

Option B

- Benutzung der gitlab.com Resources
- Konfiguration von eigenem Server nicht notwendig
- Zeitliche Limitierung in Nutzung (12/2020: 400 min)
- Einfacherer Umsetzung

CI Pipeline mit Python Projekt

Ein einfaches Projekt erleichtert das Verständnis

Next Steps

1. Projekt anlegen
2. Funktionen und Tests (pytest) schreiben
3. Datei `.gitlab-ci.yml` schreiben
4. Code zu Gitlab pushen

Server Konfiguration

Die Konfiguration ist etwas komplizierter aber gut machbar!

Voraussetzungen

- eigener Server z.B. VPS mit unix Betriebssystem
- Terminal für SSH Zugriff o. ä.
- Kenntnisse von Docker sind hilfreich

Schritte

1. Updaten des Servers
2. Anlegen von neuem Benutzer
3. root Zugriff per SSH deaktivieren
4. Login ohne Passwort aktivieren
5. Installation von Docker
6. Registrierung von Gitlab Runner

Server Konfiguration (1 / 6)

Nun die einzelnen Schritte im Überblick

Schritte

1. SSH Login
2. Updaten des Servers
3. Anlegen von neuem Benutzer
4. SSH Login mit neuem Benutzer

```
# SSH Login
~ ssh root@IP-Adresse
# Update Server
root@server:~$ apt-get update &&
apt-get upgrade
```

Server Konfiguration (2 / 6)

Nun die einzelnen Schritte im Überblick

Schritte

1. SSH Login
2. Updaten des Servers
3. Anlegen von neuem Benutzer
4. SSH Login mit neuem Benutzer

```
# Neuen Benutzer anlegen
root@server:~$ useradd -m -s
/bin/bash youtube
# Passwort ändern
root@server:~$ passwd youtube
# Benutzer zu der sudo Gruppe
# hinzufügen
root@server:~$ usermod -aG sudo
youtube
```

Server Konfiguration (3 / 6)

Root Zugriff per SSH deaktivieren

```
# Bearbeiten der SSH Konfiguration
youtube@server:~$ sudo vim /etc/ssh/sshd_config

# In vim
/PermitRootLogin
yes mit no ersetzen

# Auf der Tastatur
:wq
# SSH Service neu starten
root@server:~$ sudo systemctl restart ssh
```

Server Konfiguration (4 / 6)

Login ohne Passwort aktivieren + Installation von Docker

```
# Auf deinem Laptop / PC
~ ssh-copy-id youtube@server-IP
# Login mit youtube ohne Passwort

# Installation von Docker
youtube@server:~$ sudo apt-get install -y docker.io

# Benutzer der docker Gruppe hinzufügen
root@server:~$ usermod -aG docker $USER

# Einmal aus und wieder einloggen
```

Server Konfiguration (5 / 6)

Überprüfung der Docker Installation

```
# Testen der Docker Installation
youtube@server:~$ docker ps

# Beispiel Container starten
youtube@server:~$ docker run hello-world
```

Server Konfiguration (6 / 6)

Gitlab Runner installieren

```
# Gitlab Runner starten
# Quelle: https://docs.gitlab.com/runner/install/docker.html
youtube@server:~$ docker run -d --name gitlab-runner --restart
always -v /srv/gitlab-runner/config:/etc/gitlab-runner \
     -v /var/run/docker.sock:/var/run/docker.sock \
     gitlab/gitlab-runner:latest

# Runner registrieren -> In gitlab.com oder deiner Gitlab Instanz
eintragen, Quelle: https://docs.gitlab.com/runner/register/
youtube@server:~$ docker run --rm -it -v
/srv/gitlab-runner/config:/etc/gitlab-runner gitlab/gitlab-runner
register
```

Hilfreiche Links

Das ist erst der Start für eine CI Pipeline!

- https://docs.gitlab.com/ee/ci/README.html
- https://docs.pytest.org/en/stable/
- https://docs.docker.com/
- https://docs.gitlab.com/runner/install/docker.html
- https://docs.gitlab.com/runner/register/

Quellen:

- https://www.flaticon.com/authors/becr
- https://www.flaticon.com/autho
- https://www.flaticon.com/authors/freepik