let it run - Vom manuellen Eingriff zur Automatisierung mit GitLab CI

Iryna Kosiakovska von Puzzle ITC

PUZZLE ITC
changing IT for the better

Nice to meet you



Iryna Kosiakovska
kosiakovska@puzzle-itc.de



Sebastian Preisner
preisner@puzzle-itc.de

puzzle.ch/team



Agenda

- Zeitaufwand: Manuelle Arbeit pro Tag
- Kurze Einführung in CI/CD
- Welche gängigen CI/CD-Tools gibt es?
- Die Komponenten von GitLab CI/CD



Was machen wir eigentlich den ganzen Tag?



Zeitaufwand: Manuelle Arbeit pro Tag (Teil 1)

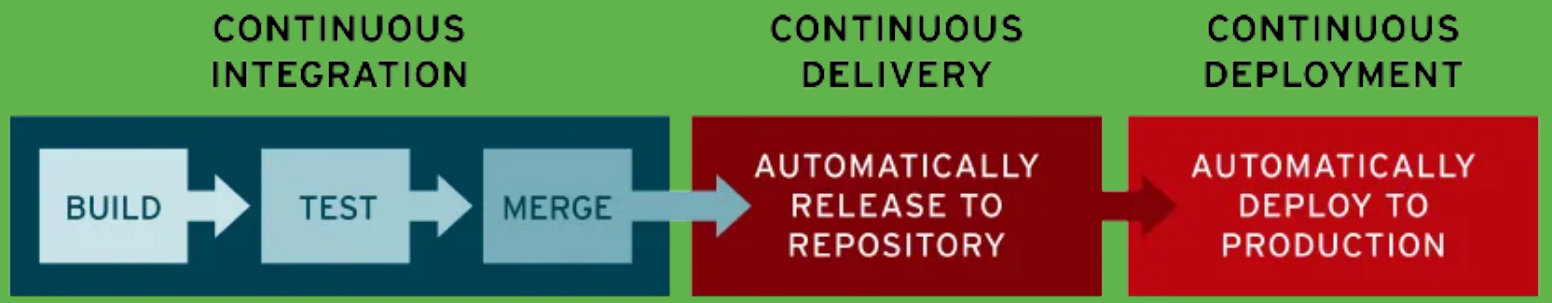
Schritt	Manueller Aufwand pro Durchlauf	Aufwand pro Tag (bei 4 Durchläufen)	Problem bei manueller Ausführung
Linting & SAST	ca. 1–2 Min.	5–8 Min.	Man vergisst es oft oder tippt den Befehl falsch ein.
Unit & Integration Tests	ca. 2–5 Min.	10–20 Min.	Lokale Tests dauern; währenddessen ist der PC oft ausgelastet.
Build erstellen	ca. 3–10 Min.	12–40 Min.	"Auf meinem PC lief es!" – lokale Umgebungen weichen oft von Prod ab.

Zeitaufwand: Manuelle Arbeit pro Tag (Teil 2)

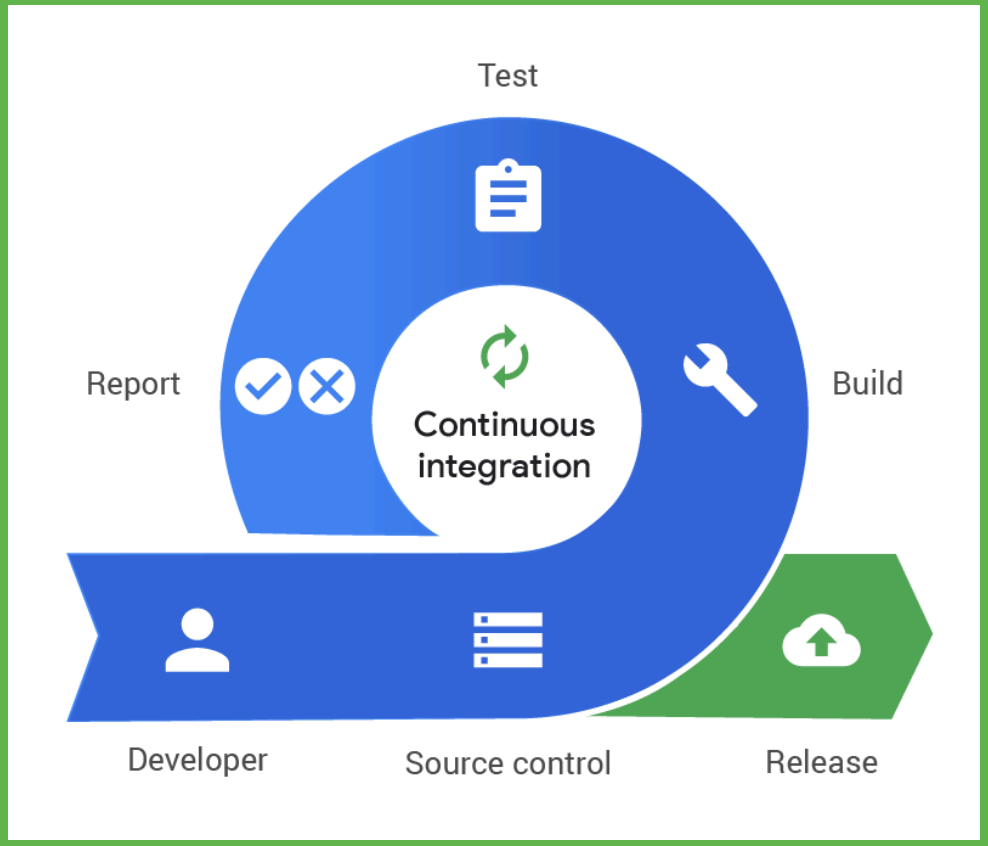
Schritt	Manueller Aufwand pro Durchlauf	Aufwand pro Tag (bei 4 Durchläufen)	Problem bei manueller Ausführung
Deployment	ca. 5–15 Min.	20–60 Min.	Manuelles Kopieren via FTP/SSH ist extrem fehleranfällig.
E2E Tests	ca. 5–10 Min.	20–40 Min.	Wird manuell oft komplett weggelassen, weil es zu lange dauert.
Gesamtzeit	ca. 16 – 42 Min.	ca. 1 bis 3 Stunden	Enormer Fokus-Verlust und "Context Switching".

Lass die Maschine ackern

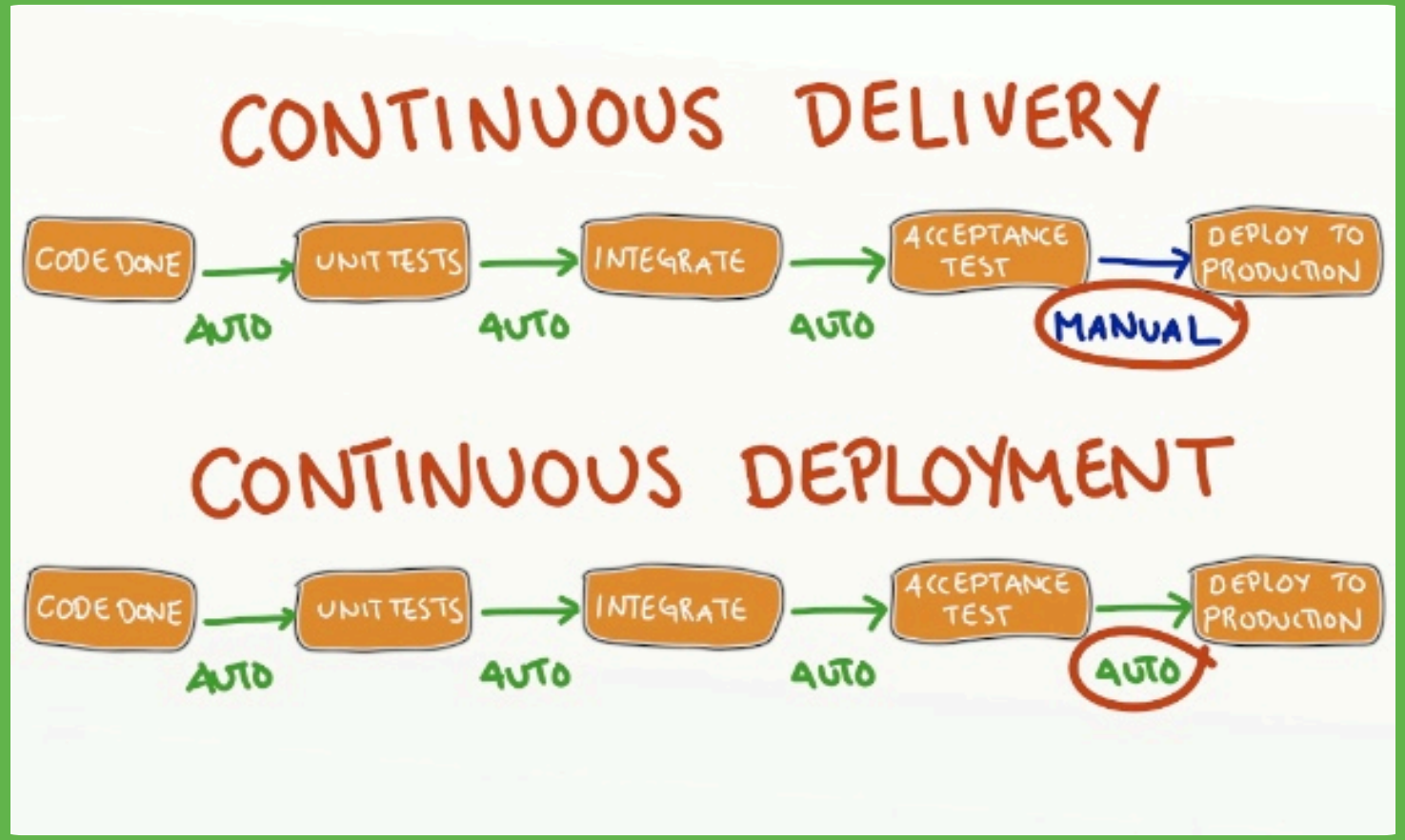
Was ist CI/CD?


CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
CONTINUOUS DEPLOYMENT
BUILD
TEST
MERGE
AUTOMATICALLY RELEASE TO REPOSITORY
AUTOMATICALLY DEPLOY TO PRODUCTION

Was ist Continuous Integration?



Was ist das „CD“ in CI/CD?



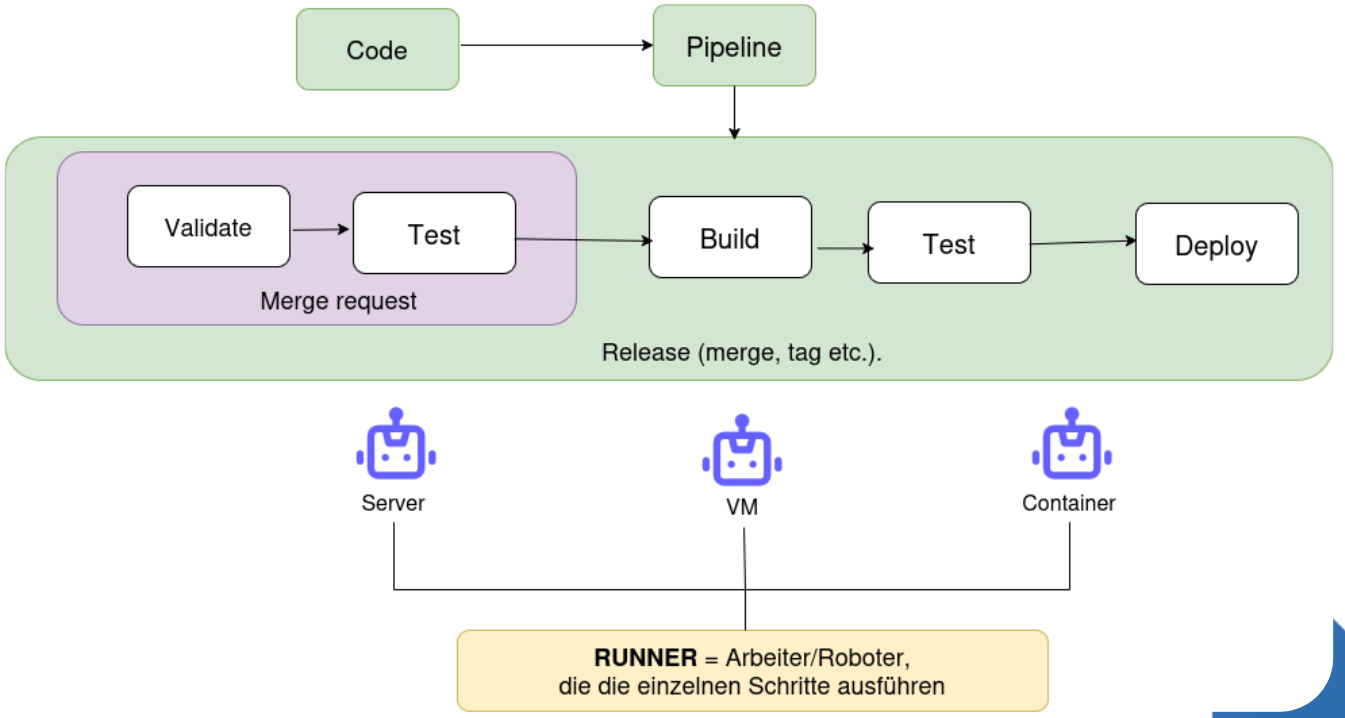
Tooling Übersicht

Tool	Deployment-Modell	Am besten für...
	Cloud & Self-hosted	Teams, die alles in einer Plattform wollen (DevSecOps).
	Cloud	Schnelle Projekte und Open-Source auf GitHub.
	Self-hosted	Komplexe Legacy-Systeme mit speziellen Anforderungen.
	Kubernetes	Moderne Cloud-Native Infrastrukturen (GitOps).

and many more!

GitLab CI/CD

Die Zutaten


Code
Pipeline
Validate
Test
Merge request
Build
Test
Deploy
Release (merge, tag etc.).
Server
VM
Container
RUNNER = Arbeiter/Roboter,
die die einzelnen Schritte ausführen

1. Pipeline in `.gitlab-ci.yml`:

```yaml
stages:
  - build
  - test
  - deploy

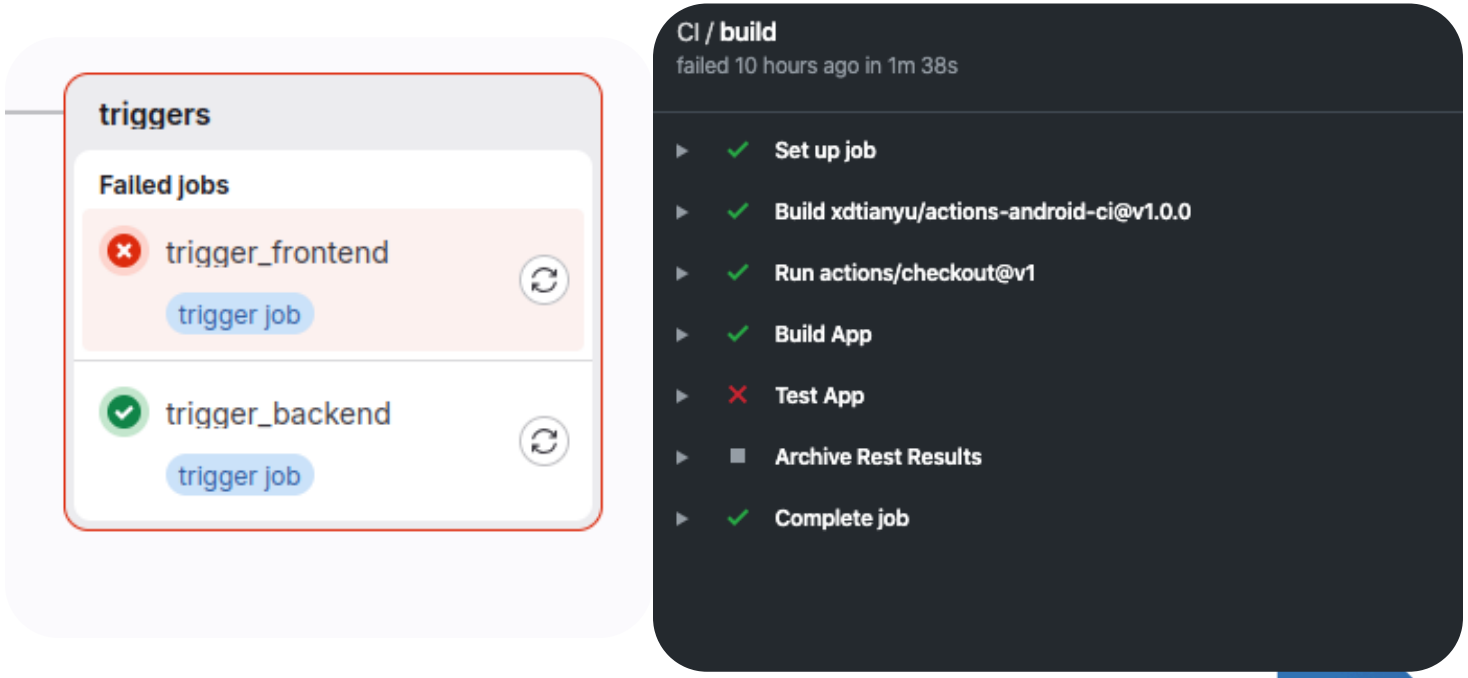
build_job:
  stage: build
  script:
    - echo "Building the application..."

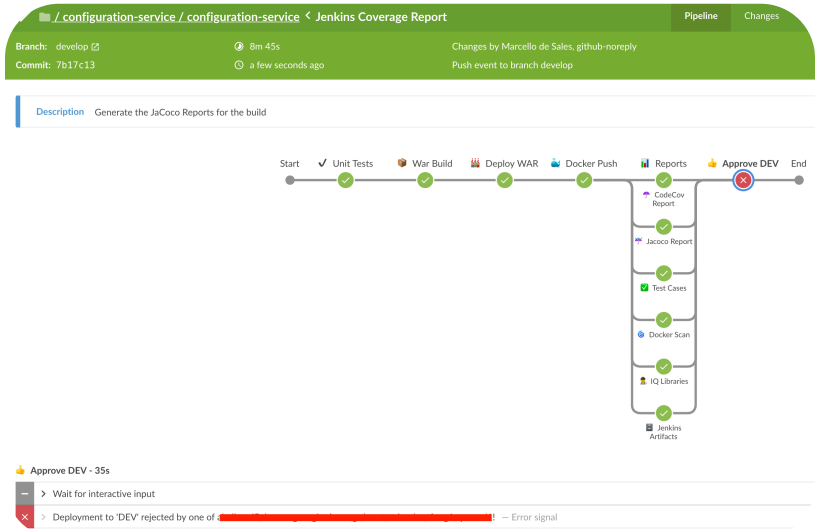
test_job:
  stage: test
  image: python:3.9-slim
  script:
    - echo "Running tests..."

deploy_job:
  stage: deploy
  script:
    - echo "Deploying to production..."
```

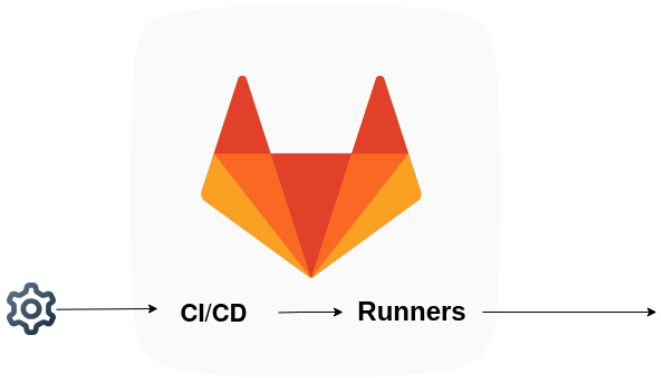


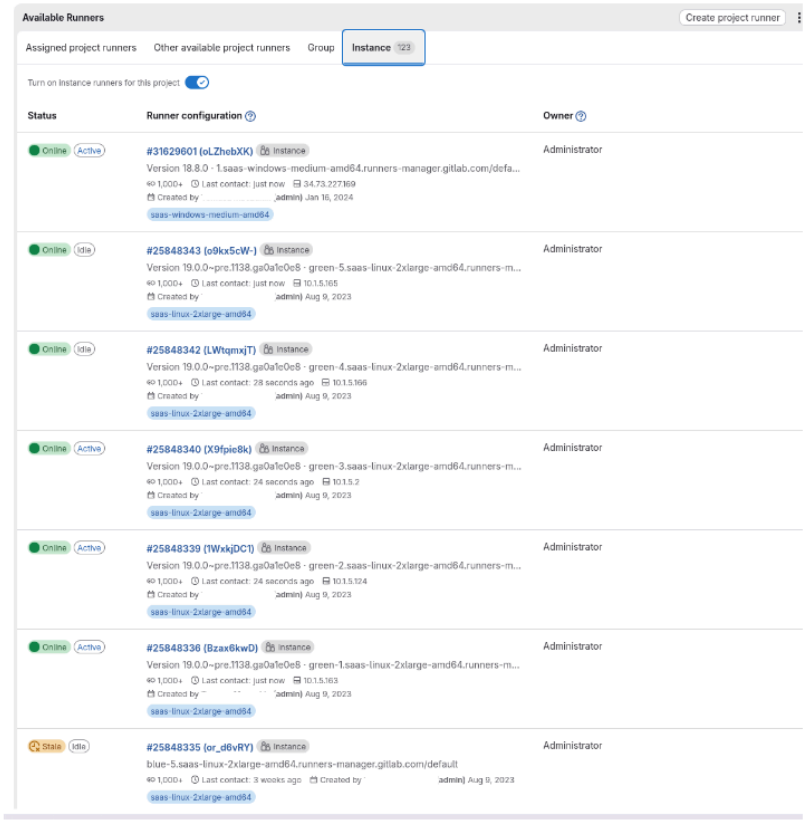
Failed und Successful Pipeline





GitLab Runner:





Fragen

Workshop

Dankeschön!

Mehr Informationen zu Puzzle:

www.puzzle-itc.de