

Automation and devops

10 yrkeshögskolepoäng
03/11 – 16/11

Vem är jag?




Max Güclü
max.guclu@sti.se



Vem är du?

Kursmål

Deadline den 16/11-2024, kl 23:55



Kunskaper

- Redogöra för begreppet devops
- Beskriva varför och hur pipelines används för att automatisera leveranser av mjukvara

Färdigheter

- Skapa pipelines som automatiskt testar och bygger en applikation
- Skapa container images
- Ta ansvar för och självständigt skapa och felsöka pipelines i cloudmiljö

Examination

- Projektuppgift i grupp om 3-4
- Pipeline
- Individuell rapport som redogör för:
 - a. Praktiska moment
 - b. Teoretiska kursmål
 - i. Självreflektion och förståelse

Kursupplägg

2 veckor

V1 Måndag Tisdag
V2 Måndag och Torsdag
9-16 på teams

- Teori
- Praktiska moment

Fredagar
13-16 på teams

- Handledning för projektuppgift eller övriga frågor

Moment

Utbildningsmoment:

- Roller & Ansvar i DevOps teams
- Containers & Dockers / Kubernetes
- Continuous integration
- Från källkod till deploy
- Pipelines

Gruppindelning

Vad är DevOps?

Development



Operations

DevOps

Development

Har ansvar för:

Mjukvarudesign
Utveckling
Test & Validering
Kravhantering

Operations

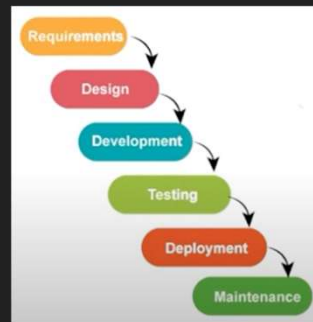
Har ansvar för:

Produktionen
Deployment
Övervakning
Loggning
Avvikelsehantering

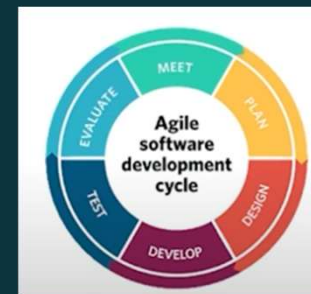
DevOps i systemutveckling

Produktion
Deployment
Övervakning
Loggning
Avvikelsehantering
Mjukvarudesign
Utveckling
Test & Validering
Kravhantering

Vattenfall



Agilt



DevOps kärnprinciper

Kultur

Samarbete
Kommunikation

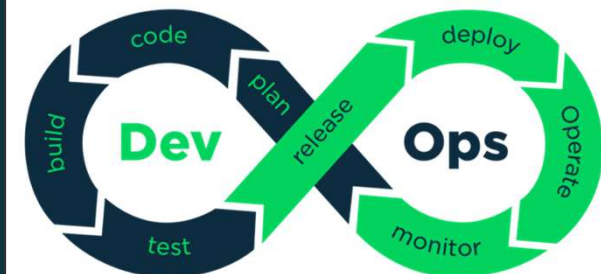
Leverans

Kontinuerlig Integration
Kontinuerlig leverans
Kontinuerlig övervakning

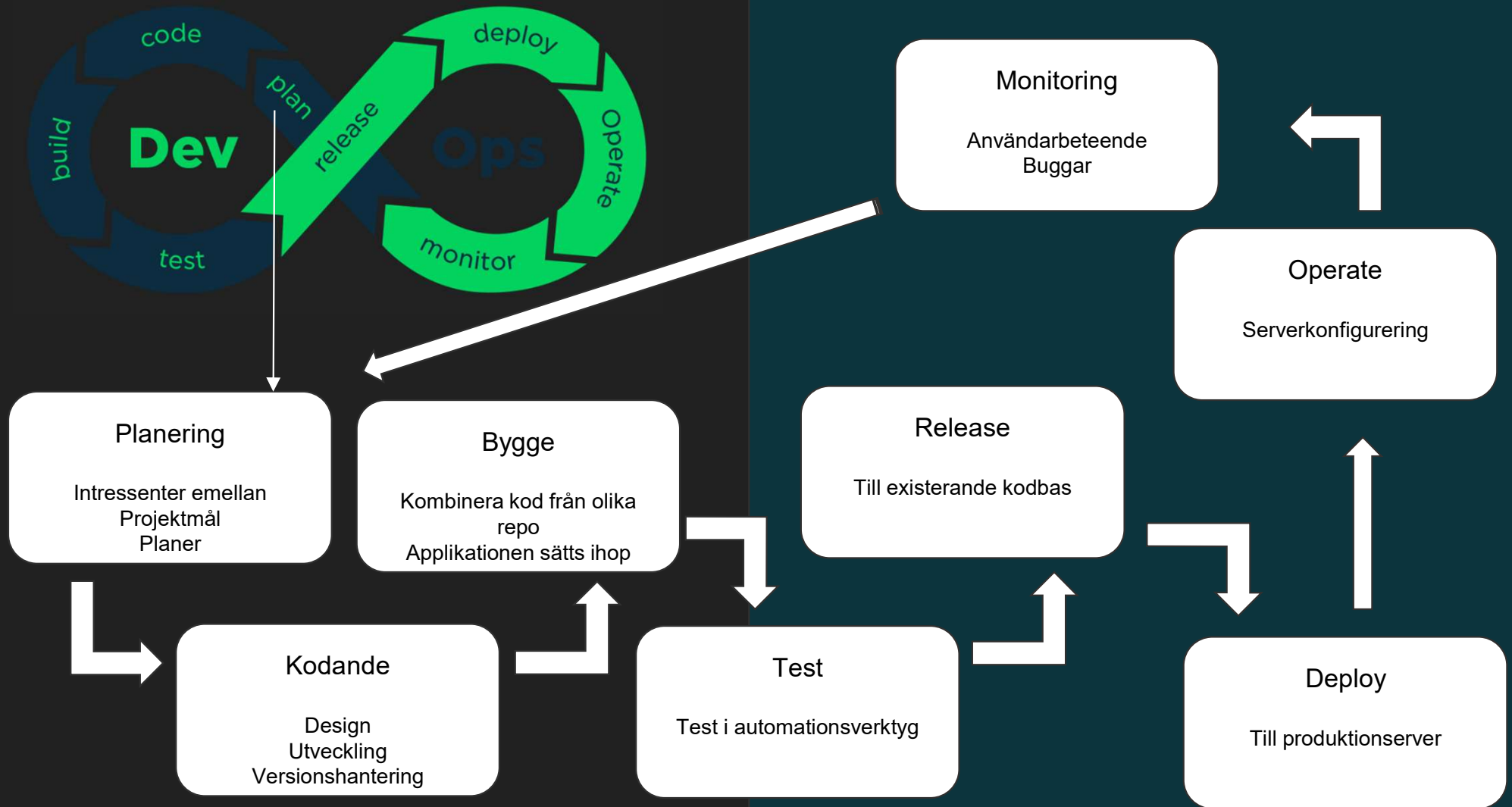
Automation

Bygga
Test
Deploy
Övervakning

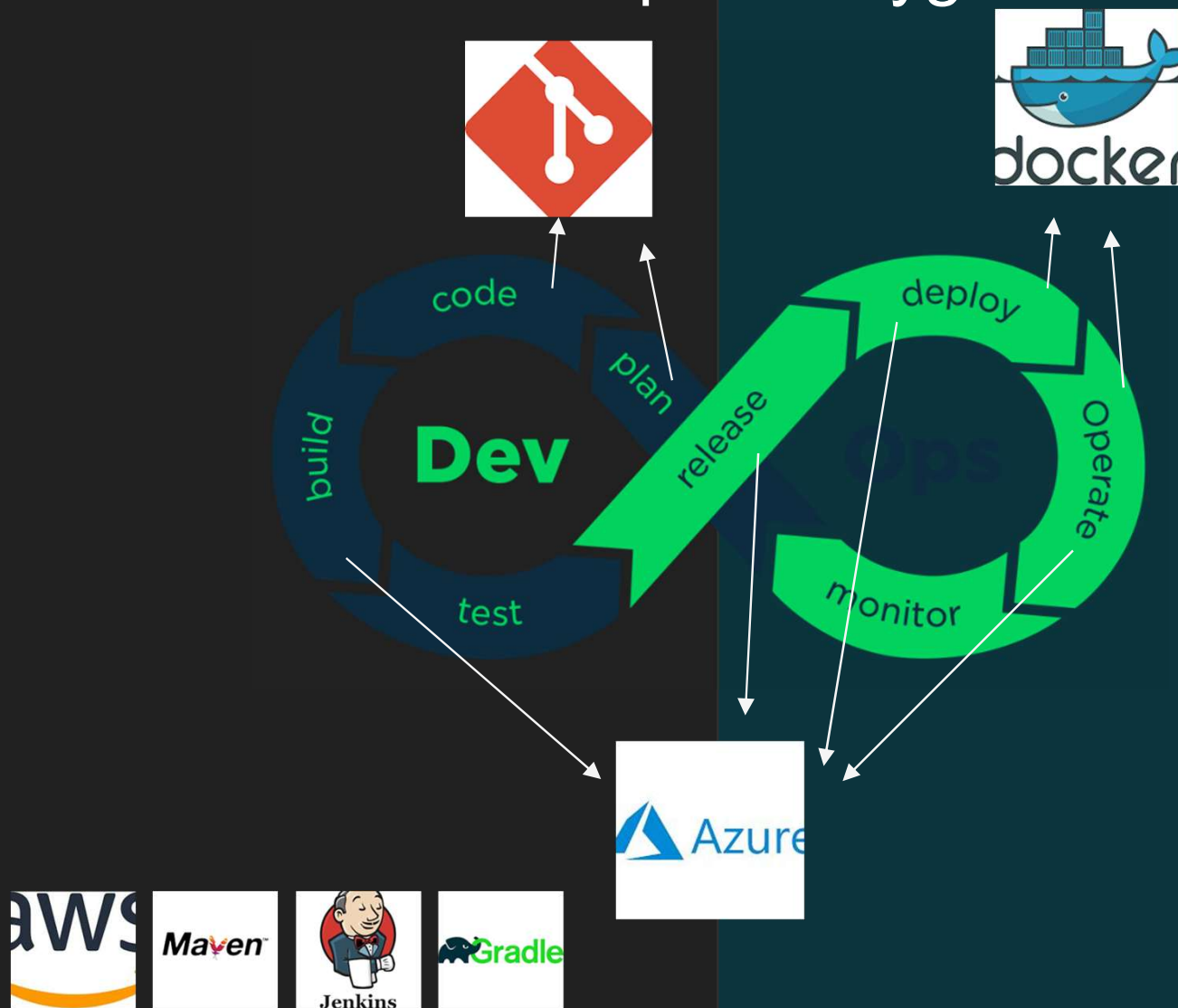
Feedback



DevOps processer



DevOps-verktyg



Projektuppgift

Web-App i grupp om 3-4

Ett API-anrop för att hämta data.

Funktion som behandlar den mottagna datan

Grundläggande UX

En unit-test som verifierar att API-anropet fungerar och att databehandlingen ger förväntat resultat.

Appen ska gå igenom en pipeline med Git -> Github Actions (tester) -> Docker -> Azure

Individuell rapport

Förståelse för begreppet DevOps & dess rön

Förståelse för pipelines i modern mjukvaruutveckling

Självreflektion av projekt och egen insats

Till på onsdag den 5/11 ska ni skicka in en
mindre projektplan på vad ni har tänkt att
skapa och hur

Detta projekt kräver ingen specifik
teknikstack utan kan implementeras med
verktyg och språk du själv väljer. Dock ska
arbetet följa DevOps-principerna som
introducerats i kursen.

DevOps Roller & Ansvar

DevOps Engineer

Säkerställer samarbete Dev - Ops

- Infrastruktur automation
- Konfigurationshantering
- Förståelse för infrastrukturen
- Skriva scripts för t.ex. automation
- Uppdatering av infrastruktur
- Övervakning av system
- Säkerhet

Utvecklare

Skriver och integrerar ny kod

Skapar och underhåller pipelines
Övervakar prestanda

De kan också ta emot feedback

Skriver och underhåller enhetstester,
integrationstester

QA

Ser till att den levererade koden uppfyller kvalitetskraven innan release.

Skriver och underhåller automatiserade tester (som integrationstester, UI-tester och funktionella tester)

säkerställer att kvalitetskontroller är en kontinuerlig del av processen.

QA kan hjälpa till att analysera felrapporter och övervakningsdata för att identifiera och lösa problem proaktivt.

Release manager

- Hanterar release-processen
- Kontrollerat och konsekvent sätt
- Koordinerar med andra team
- Planerar och hanterar versionshantering och release-schema

System Administration

Ansvarar för att systemen och infrastrukturen är stabila och fungerar
Ser till att applikationerna körs smidigt och tar ansvar för att övervaka systemens prestanda, tillgänglighet och säkerhet.
Hanterar och optimerar servrar, nätverk och andra komponenter i infrastrukturen.
Tar hand om incidenter som uppstår i drift och arbetar för att minimera avbrott och snabbt åtgärda problem.

Security Engineer

Säkerställer att säkerhetsstandarder och bästa praxis följs
Implementerar säkerhetspolicys och övervakar potentiella hot
Utför säkerhetsgranskningar och penetrationstester
Samarbetar med DevOps Engineer och System Admin för att säkerställa att systemen är säkra och skyddade

DevOps Roller & Ansvar forts.

Produktägare

Ansvarar för att prioritera funktionalitet och arbetsuppgifter baserat på affärsmål. Samarbetar nära med utvecklingsteamet för att säkerställa att produkten uppfyller användarnas och företagets behov.

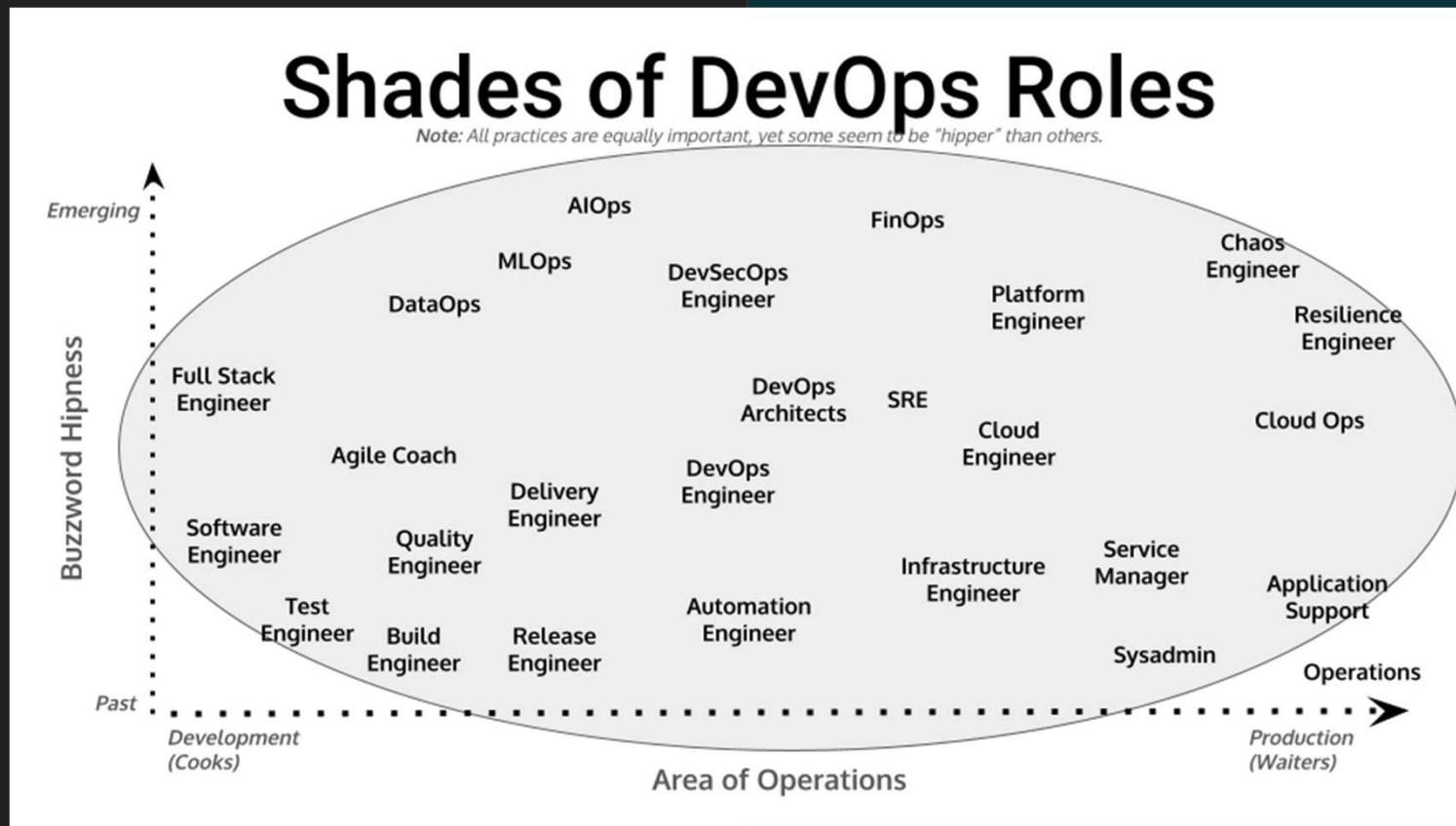
Site Reliability Engineer

Ett mer specifikt operativt fokus på att optimera driftsäkerhet och prestanda i applikationer. SREs ansvarar ofta för att bygga och hantera tillförlitliga driftsmiljöer och hantera driftshändelser.

Incident manager

Hanterar och prioriterar incidenter som påverkar produktionen och arbetar för att snabbt lösa driftproblem. Samlar in lärdomar från incidenter och förbättrar framtida hanteringsprocesser.

DevOps Roller & Ansvar forts.





Github Actions

Vad?

- Githubs automationsverktyg
- Loggar
- Secrets hantering
- Stöd för alla språk
- CI/CD
- Bygge
- Test
- Deploy

Hur funkar det?

- **Workflows**
- Hosted "runners"
- Workflow Syntax
- Continuous Integration workflows
- Deployment workflows
- Automating workflows
- Code Scanning workflows
- Pages workflows

Hur gör man för att skapa en workflow?

- I ert repo, gå till '.github/workflows', eller skapa en mapp '.github/workflows' i root
- Gå till Actions fliken i ert repo
- Tryck på New workflow
 - Färdiga mallar som:
 - Deploy
 - Security analysis
 - greetings
 - Pages



Github Actions

- Checkout
- Bygga
- Installera SDK
- Ladda ned och upp artefakter
- Cache beroenden
- Deploy
- Sköts av gh

Linting
Security scan
Merge pull
Enforce branch naming
Post slack meddelanden
Epost

Github Marketplace
Gå in på github marketplace och sök på azure



GitHub Actions

YAML

```
name: GitHub Actions Demo
run-name: ${{ github.actor }} is testing out GitHub Actions 🚀
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "🎉 The job was automatically triggered by a ${{ github.event_name }} event."
      - run: echo "🐧 This job is now running on a ${{ runner.os }} server hosted by GitHub!"
      - run: echo "🌍 The name of your branch is ${{ github.ref }} and your repository is ${{ github.repository }}."
      - name: Check out repository code
        uses: actions/checkout@v4
      - run: echo "💡 The ${{ github.repository }} repository has been cloned to the runner."
      - run: echo "📄 The workflow is now ready to test your code on the runner."
      - name: List files in the repository
        run: |
          ls ${{ github.workspace }}
      - run: echo "🍏 This job's status is ${{ job.status }}."
```

← GitHub Actions Demo

✅ Max-comerit is testing out GitHub Actions 🚀 #1

Explore-GitHub-Actions

succeeded 2 minutes ago in 8s

- > ✅ Set up job
- > ✅ Run echo "🎉 The job was automatically triggered by a push event."
- > ✅ Run echo "🐧 This job is now running on a Linux server hosted by GitHub!"
- > ✅ Run echo "🌍 The name of your branch is refs/heads/main and your repository is Max-comerit/WeatherAPI-React."
- > ✅ Check out repository code
- > ✅ Run echo "💡 The Max-comerit/WeatherAPI-React repository has been cloned to the runner."
- > ✅ Run echo "📄 The workflow is now ready to test your code on the runner."
- > ✅ List files in the repository
- > ✅ Run echo "🍏 This job's status is success."
- > ✅ Post Check out repository code
- > ✅ Complete job



YAML

YAML Ain't Markup Language

Vad

Se det som en konfigurationsfil
Redogör för regler och inställningar
Går att läsas av människor & datorer

I CI/CD används dessa som instruktioner som
automatiserar vissa uppgifter

Hur

Första och sista kod är "---"
Key value pairs
Separerade av kolon och space
Inga tabs för indragningar, endast space

key: value

<https://learnxinyminutes.com/docs/yaml/>



Github Actions - YAML



Vilka actions ska jag välja?

Vad ska ditt workflow göra?

Exempelvis:

Bygga och testa koden

Distribuera koden

`actions/checkout@v2` behövs nästan alltid - tillgängliggör repot till yml filen
`actions/setup-[miljö]` t.ex. `actions/setup-node@v2` - tillgängliggör runtime-miljö. Går inte att köra npm install annars t.ex.

Lathund!

Så här tänker du för att välja rätt actions

Behöver du koden? – Börja med `actions/checkout`.

Behöver du en specifik miljö? – Använd `actions/setup-[miljö]` för att installera rätt runtime.

Behöver du installera beroenden varje gång? – Använd `actions/cache` för att spara tid.

Behöver du spara filer mellan steg? – Använd `actions/upload-artifact` och `actions/download-artifact`.



YAML-Struktur

<https://github.com/marketplace?type=actions>

Indelat i sektioner

```
name: Mitt Workflow      # Namnet på workflow (valfritt)
on:                      # När workflow ska köras
  push:                  # Trigger när kod pushas till specifika grenar
    branches:
      - main
  pull_request:          # Trigger när en pull request skapas
```

```
jobs:                   # Definierar de olika jobben som ska köras
  test: # Första jobbet för att köra tester
    runs-on: ubuntu-latest # Välj vilket operativsystem jobbet körs på
```

```
  steps:                # Steg som ska köras i jobbet
```

```
    - name: Check out code
      uses: actions/checkout@v2

    - name: Set up Node.js
      uses: actions/setup-node@v2
      with:
```

```
        node-version: '14'
```

```
    - name: Install dependencies
      run: npm install
```

```
    - name: Run tests
      run: npm test
```

```
  steps:
    - name: Kör flera kommandon
      run: |
        echo "Steg 1: Bygg applikationen"
        npm run build
        echo "Steg 2: Kör testerna"
        npm test
        echo "Steg 3: Distribution"
        echo "Distributionen är klar!"
```

```
  build: # Andra jobbet för att bygga applikationen
    runs-on: ubuntu-latest
    needs: test # Detta jobb körs bara om 'test' lyckas
    steps:
      - name: Check out code
        uses: actions/checkout@v2
```

Sektioner:

- Namn
 - Sätter namn på workflow
- Trigger
 - Startar med "on"
 - Finns olika konfigurationer
 - I exempel push till main & Pull-request
- Jobs
 - Definierar vad som ska göras
 - Sker i steg
 - "Build" namnet på job
 - Skapar beroenden
 - Kör tester
 - runs-on:
 - Väljer miljö för virtuella maskinen
 - steps:
 - Namn på steget
 - uses: vilken action
 - med
 - run: kommando
- Flera jobs kan följa
 - Beroenden definieras med needs



Tester

Github Actions - YAML

```
jobs:
  test:
    runs-on: ubuntu-latest
    steps:
      - name: Check out the code
        uses: actions/checkout@v2

      - name: Set up environment
        uses: actions/setup-[language]@v2
        with:
          [language]-version:
            'version'

      - name: Install dependencies
        run: [install_command]

      - name: Run tests
        run: [test_command]
```

```
- name: Set up Node.js
  uses: actions/setup-node@v2
  with:
    node-version: '14'

- name: Install dependencies
  run: npm install

- name: Run tests
  run: npm test
```


Övningsuppgifter, 4 st

1. Skapa en enkel yaml workflow

Heter hello world
Triggar på push
Skriver ut "hello world"

2. Skapa en workflow med env.variabler

Valfritt namn
Triggar på workflow dispatch
definera 2 miljövariabler i job och använd dem i steps
t.ex. Greeting.
Kör några Echos där ni testat på en del miljövariabler och github variabler

3. Skapa en workflow med flera jobs

Valfritt namn
Triggar på något annat än en pull / push
definera 2 miljövariabler i job och använd dem i steps
t.ex. Greeting. Använd gärna en secret också (finns i repo settings)

Skapa beroenden mellan 2 jobs "needs"

4. Skapa en workflow som testat

Skapa ett nytt repo och app
Skapa en enkel pytest som kör någon 1+1 logik
Skapa workflow som kombinerar tidigare labbuppgifter
Dvs variabler och echos som redogör för det som sker.

Skapa 2 jobs, en som testat, och om test är godkänt, så bygger den applikationen