

**Czas: 8 godz. lekcyjnych 9:00 ~ 17:00**

*Przerwy co godzinę 5 minut + obiadowa ok 45 min.*

**o mnie:**

- <https://www.linkedin.com/in/adam-jurkiewicz-python-linux/>
  - <https://blog.jurkiewicz.tech/about>
  - <https://jurkiewicz.tech>
  - <https://blog.jurkiewicz.pro>
  - <https://jurkiewicz.pro>
- 

**Wrzutki waszych zadań jako printscreeny:**

---

**Plan pracy:**

- 1szy dzień
  - linux / podstawy i przypomnienie (lynx i curl się przydadzą 2giego dnia)
  - git z Terminala
  - git i github
  - github actions, workflow przykładowy
- 2gi dzien
  - python Flask project
  - testy pyTest
  - docker podstawy wykorzystania compose
  - Jenkins konfiguracja
  - zarządzanie użytkownikami
  - podstawowe pipeliny
  - Jenkinsfile -> przykłady
  - połączenie z githubem i webhooki
  - wykonanie
  - **inne githuby - omówienie, testy (na przykładzie Gitea - OnPremis self-hosted)**

**Ustalenia IP dla uczestników:**

...

Trener Public IP address: 20.238.32.128 + 🌟 (G:adam@jurkiewicz.tech) ✨

Zdalni (W) Artur Drzewakowski Public IP address: 4.210.114.146 + 🌟  
(G:a7r4t1ur@gmail.com) ✨ (L) Mariusz Grecki Public IP address: 20.166.62.29 + 🌟  
(G:mariusz.grecki87@gmail.com) ✨ (W) Marek Kwatera Public IP address: 4.210.124.59 + 🌟

(G: am.kwatera@gmail.com) ✨ (W) Maciej Konobrocki Public IP address: 4.210.124.144 + 🌟  
(G: maciekkono@gmail.com) ✨ (W) Aleksander Chrzanowski Public IP address: 98.71.97.115  
+ 🌟 (G:aleksander.chrzanowski.dev@gmail.com) ✨

Sala (W) Daniel Stepanian Public IP address: 20.238.115.237 + 🌟 (G: entukio@gmail.com) ✨ (W) Grzegorz Sanecki Public IP address: 4.210.108.59 + 🌟  
(G:sanecki.grzegorz@gmail.com) ✨ (L) Mateusz Zawadzki Public IP address: 40.69.95.200 +  
(G:mzawadzki.developer@gmail.com) ✨

(W) Daniel Sternik Public IP address: 74.234.81.201 (G:dxs2@wp.pl)

l: jenkinsadmin p: jenkinsadmin

...

Klucze prywatne dla Windows w prywatnym repo Adama:

<https://git.jurkiewicz.tech/adamjurkiewicz/szkolenia-podstawowe-informacje/src/branch/main/Kluczessh>

Artykuł o Putty: <https://blog.jurkiewicz.pro/2025/11/11/Putty-dla-Windows-w-10-minut-z-kluczem-ssh/>

Git manager:

<https://github.com/sourcegit-scm/sourcegit>

Różne serwisy git: <https://github.com/> <https://about.gitlab.com/> <https://bitbucket.org/> <https://codeberg.org/> <https://about.gitea.com/>

Github:

Trener: <https://github.com/Adam-Jurkiewicz-Pythonista/alsx-deops-1213>

MK <https://github.com/Marek883-dev/ALX-Devops> MG: <https://github.com/MariuszGrecki/alx-devops-1213> Maciej K: <https://github.com/MaciejKonobrocki/alx-devops> AD: <https://github.com/a7r4t1ur/testowe742> Mateusz Zawadzki: <https://github.com/ethereume/test> GS: <https://github.com/grzegorz-gsa/repo-kurs> entukio: [https://github.com/entukio/alx\\_devops](https://github.com/entukio/alx_devops) ACH: <https://github.com/a-chrzanowski/alsx-devops-1213#> DS: <https://github.com/dast835/alx-repo>

---

---

## Zadanie:

Sklonować repo: `git@github.com:Adam-Jurkiewicz-ythonista/alsx-deops-1213.git`

Utworzyć branch o nazwie `devel/XYRRRR` (XY - to wasze inicjały, a RRRR to rok urodzenia)

## Bazujący na main !!!

`devel/AJ1974` - to mój przykład

W nim utworzyć plik o nazwie `infoXYRRRR.txt`

I zrobić commit i push

<https://github.com/Adam-Jurkiewicz-Pythonista/alsx-deops-1213>

I New PullRequest

---

## Fork z GitHuba.....

<https://github.com/Adam-Jurkiewicz-Pythonista/alx-acions-01-init> i clone tego do linux'a

---

## I screena z Source git do plików

### Do github actions

.github/workflow/plik.yml

```
name: Test Log Output env: REPO_NAME: ${ github.event.repository.name } ..
on: [push] jobs: print-logs: runs-on: ubuntu-latest steps:
- name: Wypisz powitanie run: echo "Witaj w GitHub Actions!" - name:
Test echo run: echo "sprawdzamy - $REPO_NAME"
```

---

```
``` name: CI
```

```
on: push: branches: [ main ] pull_request: branches: [ main ]
```

```
jobs: test: runs-on: ubuntu-latest
```

```
steps:
```

```
- uses: actions/checkout@v4
```

```
- name: Set up Python 3.12
  uses: actions/setup-python@v5
  with:
    python-version: '3.12'
```

```
- name: Install dependencies
  run: |
    python -m pip install --upgrade pip
    pip install -r requirements.txt
    pip install ruff pytest
```

```
- name: Lint with Ruff
  run: ruff check .
```

```
- name: Run tests
  run: pytest tests/ -v
```

```
```
```

### Zadanie:

[https://doc.jurkiewicz.tech/34gIjMchQQyXGvTlZ0W\\_-A?both](https://doc.jurkiewicz.tech/34gIjMchQQyXGvTlZ0W_-A?both)

### YAML Linter offline

- <https://github.com/adrienverge/yamllint> sudo apt install yamllint

---

## Jenkins podstawy:

Prosty Jenkinsfile:

```
...
```

## podstawowy - tylko wypisanie na ekranie

// <https://www.jenkins.io/doc/book/pipeline/jenkinsfile/>

```
pipeline { agent any
  stages {
    stage('Build') {
      steps {
        echo 'Building..'
      }
    }
    stage('Test') {
      steps {
        echo 'Testing..'
      }
    }
    stage('Deploy') {
      steps {
        echo 'Deploying....'
      }
    }
  }
}
```

} Teraz Jenkinsfile dla połączenia Githuba z repozytorium: // flask uruchomiony z linii poleceń

```
pipeline { agent any
  stages {

    stage('Checkout') {
      steps {
        checkout scm
      }
    }

    stage('Run with venv') {
      options {
        timeout(time: 5, unit: 'MINUTES')
      }

      steps {
        script {
          sh '''
            python3 -m venv venv
            . venv/bin/activate
            pip install -r requirements.txt
          '''
        }
      }
    }
  }
}
```

```

        python3 app.py
    '''
    }
}

stage('Verify Deployment') {
    steps {
        sh 'sleep 5'
        sh 'curl -f http://localhost:5000 || exit 1'
    }
}

post {
    always {
        sh "docker image prune -f"
    }
}
}''' —

```

WebHook:

<http://x.x.x.x:8080/github-webhook/>

---

## Sukces

''' // flask uruchomiony z linii poleceń

```

pipeline { agent any
    stages {

        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Run with venv') {
            options {
                timeout(time: 1, unit: 'MINUTES')
            }

            steps {
                script {
                    sh '''
                    python3 -m venv venv
                    . venv/bin/activate
                    pip install -r requirements.txt
                    python3 app.py

```

```

    ...
    }
  }
}

}

\\ abort zmieniamy w sukces na końcu post { aborted { script { // jeśli chcesz traktować
timeout/abort jako sukces currentBuild.result = 'SUCCESS' }}}
}
...

```

---

#### **Na koniec szkolenia - ankieta:**

<https://www.alx.pl/ankiety/7856vbzx>

#### **Polecajki obok szkolenia:**

<https://www.markdownguide.org/getting-started/>

<https://blog.jurkiewicz.pro/2025/11/17/Gdy-pamietasz-ten-edytor-do-emerytury-niedaleko/>

---

```

``` // flask uruchomiony z linii poleceń

```

```

pipeline { agent any

```

```

  stages {

```

```

    stage('Checkout') {
      steps {
        checkout scm
      }
    }
  }

```

```

    stage('Run with venv') {
      options {
        timeout(time: 1, unit: 'MINUTES')
      }

```

```

      steps {
        script {
          sh '''
            python3 -m venv venv
            . venv/bin/activate
            pip install -r requirements.txt
            nohup python3 app.py > app.log 2>&1 &
            sleep 3
          '''
        }
      }
    }
  }
}

```

```

    }
}

stage('Verify Deployment') {
    steps {
        sh 'sleep 5'
        sh 'curl -f http://localhost:5000 || exit 1'
    }
}
}
} ``` —

```

### Skrypt z raportem:

Do pliku: Jenkins\_raport ``` pipeline { agent any

```

environment {
    PYTHON = 'python'
    VENV_DIR = '.venv'
}

stages {
    stage('Checkout') {
        steps {
            checkout scm
        }
    }

    stage('Setup Python env') {
        steps {
            sh """
            ${PYTHON} -m venv ${VENV_DIR}
            . ${VENV_DIR}/bin/activate
            pip install --upgrade pip
            pip install -r requirements.txt
            """
        }
    }

    stage('Run pytest') {
        steps {
            sh """
            . ${VENV_DIR}/bin/activate
            pytest -v --junitxml=reports/junit-report.xml
            """
        }
    }
}

post {
    always {
        // Wciągnięcie wyników testów do zakładki "Test Result"
    }
}

```

```

        // junit 'reports/junit-report.xml'
        // blokada, bo wymaga odpowiedniej konfiguracji GitHub
        // https://stackoverflow.com/questions/67162746/how-to-get-rid-of-
noisy-warning-no-suitable-checks-publisher-found
    }
    success {
        echo 'Tests passed 🎉'
    }
    failure {
        echo 'Tests failed ❌'
    }
}
}
...

```

---

the configuration of GitHub App credentials, see this guide for more details. If not disabled in the job configuration, this plugin will publish test results to GitHub through GitHub checks API.

In the Details view of each check (example), test results will be displayed.

checks

In order to disable the checks feature, set the property `skipPublishingChecks` to true: