

# Automatyzacja w procesie wytwarzania oprogramowania

Piotr Olasik, Witold Pacholik, Wojciech Grzywocz



Politechnika  
Śląska

Wydział Matematyki Stosowanej

25.06.2025



# Agenda

## 1 Wprowadzenie

- Cel
- Co to Jenkins?

## 2 Rezultaty projektu

- Utworzenie projektu z aplikacją kalkulatora
- Automatyzacja z Github Actions
- Automatyzacja z Jenkins
- Testy

## 3 Podsumowanie

- Podsumowanie i wnioski



# Cel

## Cel

- Celem projektu jest przedstawienie różnych możliwości wykorzystania narzędzi do automatyzacji w procesie wytwarzania oprogramowania. Narzędzia te mają ułatwiać pracę w całym procesie wytwarzania oprogramowania i są podstawowymi usługami używanymi przez osoby związane z DevOps.




# Co to Jenkins?

Jenkins jest open-source'owym narzędziem do automatyzacji, które umożliwia ciągłą integrację i ciągłe dostarczanie (CI/CD) oprogramowania. Jest to serwer automatyzacji napisany w języku Java, który pozwala developerom na:

- **Automatyzację budowania** – Jenkins może automatycznie kompilować kod źródłowy po każdej zmianie w repozytorium
- **Uruchamianie testów** – Automatyczne wykonywanie testów jednostkowych, integracyjnych i funkcjonalnych
- **Deployment aplikacji** – Wdrażanie aplikacji na różne środowiska (testowe, staging, produkcyjne)
- **Integrację z wieloma narzędziami** – Git, Docker, Maven, Gradle, oraz setkami pluginów












# Utworzenie projektu z aplikacją kalkulatora

 **python-calculator** Public

[Watch](#) 1 [Fork](#) 1 [Star](#) 0

[main](#) 1 Branch 0 Tags  [Add file](#) [Code](#)

 **PiotruOlasik** Update Jenkinsfile ✓ 0e2a852 - 9 hours ago 30 Commits

	replace test step with pytest --cov	last month
	test jenkinsa	2 days ago
	Add files via upload	last month
	Update Jenkinsfile	9 hours ago
	Update README.md	last month
	testowane	2 days ago
	Update requirements.txt	last month
	poprawa testow	2 days ago

README

## Python Calculator

Prosty projekt w Pythonie z podstawowymi funkcjami kalkulatora. Cechy automatyzacja

About

No description, website, or topics provided

- Readme
- Activity
- 0 stars
- 1 watching
- 1 fork

Report repository




Releases

No releases published

Packages

No packages published

Contributors 3

-  Grzywocz-W
-  witek171 Witold Pacholik
-  PiotruOlasik



# Plik z testami

Code

Blame

24 lines (17 loc) · 421 Bytes

```
1      # test_main.py
2
3      import pytest
4      from main import add, subtract, multiply, divide
5
6      def test_add():
7          assert add(2, 3) == 5
8
9      def test_subtract():
10         assert subtract(5, 3) == 2
11
12     def test_multiply():
13         assert multiply(4, 2) == 8
14
15     def test_divide():
16         assert divide(10, 2) == 5
17
18     ## test
19     def test_divide():
20         assert divide(10, 2) == 5
21
22     def test_divide_by_zero():
23         with pytest.raises(ValueError):
24             divide(5, 0)
```



# Wygląd aplikacji

## Prosty Kalkulator

<input type="text" value="5"/>	<input type="button" value="+"/>	<input type="text" value="10"/>
<input type="button" value="Oblicz"/>		

Wynik: 15.0



# Automatyzacja z Github Actions

Code

Blame

26 lines (19 loc) · 524 Bytes

```
1   name: Run Python tests
2
3   on: [push, pull_request]
4
5   jobs:
6     test:
7       runs-on: ubuntu-latest
8
9       steps:
10        - name: Checkout repo
11          uses: actions/checkout@v3
12
13        - name: Set up Python
14          uses: actions/setup-python@v5
15          with:
16            python-version: '3.10'
17
18        - name: Install dependencies
19          run: |
20            python -m pip install --upgrade pip
21            pip install -r requirements.txt
22
23        - name: Run tests with coverage
24          run: |
25            pytest --cov=. --cov-report=term-missing
```





# Automatyzacja z Jenkins

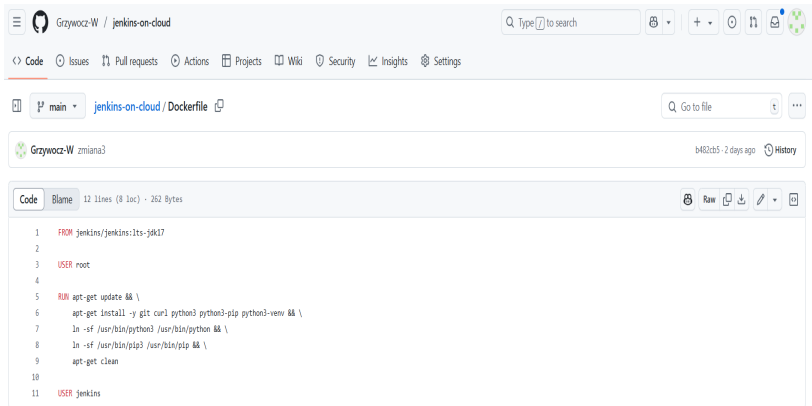
Jenkins został uruchomiony na wirtualnej maszynie z systemem ubuntu na chmurze Microsoft Azure. Konfiguracja po stronie Azure:

Subskrypcja	Azure for Students
Grupa zasobów	(nowy) ZSI
Nazwa maszyny wirtualnej	Jenkins-on-linux
Region	East US
Opcje dostępności	Nie jest wymagana żadna nadmiarowość infrastruktury
Opcje strefy	Strefa wybrana samodzielnie
Typ zabezpieczeń	Standardowe
Obraz	Ubuntu Server 24.04 LTS - Gen2
Architektura maszyny wirtualnej	x64
Rozmiar	Standard B1s (vcpu: 1, 1 GiB pamięci)
Włącz hibernację	Nie
Typ uwierzytelniania	Klucz publiczny SSH
Nazwa użytkownika	azureuser
Format klucza SSH	RSA
Nazwa pary kluczy	Jenkins-on-linux_key
Publiczne porty ruchu przychodzącego	SSH
Azure Spot	Nie



# Automatyzacja z Jenkins

Jenkins jest de facto dockerowym kontenerem z Pythonem. Na początku utworzyliśmy Dockerfile i repozytorium z nim na github



The screenshot shows a GitHub repository page for 'Grzywocz-W / jenkins-on-cloud'. The file 'Dockerfile' is selected, showing 12 lines of code. The code defines a Docker image for Jenkins using Python.

```
1 FROM jenkins/jenkins:lts-jdk17
2
3 USER root
4
5 RUN apt-get update && \
6     apt-get install -y git curl python3 python3-pip python3-venv && \
7     ln -sf /usr/bin/python3 /usr/bin/python && \
8     ln -sf /usr/bin/pip3 /usr/bin/pip && \
9     apt-get clean
10
11 USER jenkins
```



# Automatyzacja z Jenkins

Repozytorium z Dockerfile zostało sklonowane na chmurę, a następnie utworzono Docker image i kontener na maszynie Ubuntu. Dzięki temu mieliśmy dostępnego Jenkins na porcie 8080, do którego mogliśmy się zalogować przez przeglądarkę podając adres ip i port z maszyny ubuntu.

```
info: Adding group 'docker' (GID 114) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for dbus (1.14.10-4ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

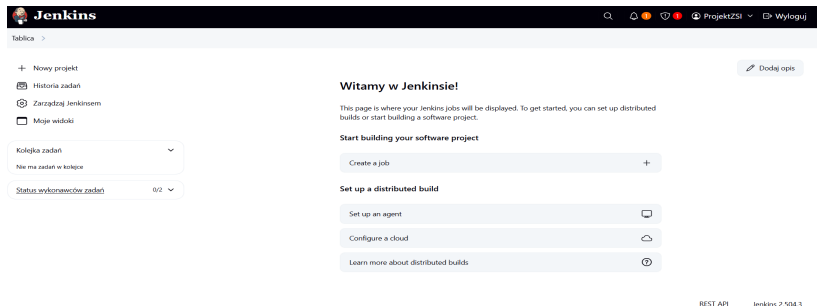
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
azureuser@jenkins-on-linux:~$
azureuser@jenkins-on-linux:~$ sudo systemctl start docker
azureuser@jenkins-on-linux:~$ sudo systemctl enable docker
azureuser@jenkins-on-linux:~$ sudo usermod -aG docker $USER
azureuser@jenkins-on-linux:~$ newgrp docker
azureuser@jenkins-on-linux:~$ git clone https://github.com/Grzywocz-W/jenkins-on-cloud.git
Cloning into 'jenkins-on-cloud'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
azureuser@jenkins-on-linux:~$
```



# Automatyzacja z Jenkins

Po utworzeniu konta mogliśmy doinstalować potrzebne pluginy i zapoznać się z Jenkinsem i jego funkcjami w sposób praktyczny:



# Automatyzacja z Jenkins

Następnie w repozytorium projektu utworzyliśmy Jenkinsfile ze schematem pipeline'a dla Jenkinsa:

```

Code Blame 46 lines (41 loc) · 1.11 KB
1 pipeline {
2   agent any
3
4   stages {
5     stage('Clone') {
6       steps {
7         sh 'rm -rf python-calculator'
8         sh 'git clone https://github.com/Grzywocz-W/python-calculator.git'
9       }
10    }
11
12    stage('Install dependencies') {
13      steps {
14        dir('python-calculator') {
15          sh '''
16            python3 -m venv venv
17            . venv/bin/activate
18            pip install --upgrade pip
19            pip install -r requirements.txt
20            ...
21          '''
22        }
23      }
24    }
25
26    stage('Run tests') {
27      steps {
28        dir('python-calculator') {
29          sh '''
30            . venv/bin/activate
31            pytest test_main.py --maxfail=1 --disable-warnings --exitfirst
32            ...
33          '''
34        }
35      }
36    }
37  }

```



# Automatyzacja z Jenkins

Kolejnym krokiem było utworzenie pipeline'a po stronie Jenkins:

## Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- ☐ Uruchamiaj, gdy inne zadania zostaną zakończone ?
- ☐ Buduj cyklicznie ?
- ☐ GitHub Branches
- ☐ GitHub Pull Requests ?
- ☒ GitHub hook trigger for GITScm polling ?
- ☐ Pobierz z repozytorium kodu ?
- ☐ Wyzwalaj budowanie zdalnie (np. przez skrypt) ?

## Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

### Definition

Pipeline script from SCM



# Automatyzacja z Jenkins

## Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

### Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

`https://github.com/PiotruOlasik/python-calculator.git`

Credentials ?

- none -

+ Add

Save Zastosuj



# Automatyzacja z Jenkins

Branches to build ?

Branch Specifier (blank for 'any') ?

\*/main

Add Branch

Repository browser ?

(Automatyczny)

Additional Behaviours

Dodaj ▾

Script Path ?

Jenkinsfile

☒ Lightweight checkout ?

Save Zastosuj





# Automatyzacja z Jenkins

Następnie utworzyliśmy w repozytorium projektu na githubie webhooka aby automatyzacja w Jenkins przebiegała poprawnie

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, *x-www-form-urlencoded*, etc). More information can be found in [our developer documentation](#).

**Payload URL \***

**Content type \***

**Secret**

**SSL verification**  
By default, we verify SSL certificates when delivering payloads.  
☒ **Enable SSL verification** ☐ **Disable (not recommended)**

**Which events would you like to trigger this webhook?**  
☒ Just the push event.  
☐ Send me **everything**.  
☐ Let me select individual events.

☒ **Active**  
We will deliver event details when this hook is triggered.



# Testy

Na koniec wykonując push w git do repozytorium sprawdziliśmy czy automatyzacja testów działa i czy testy wykonują się poprawnie:

Sukces		python-pytest-pipeline	1 dzień 17 godz #1	—	21 sek	
		python-test-python-calculator-main	3 godz 28 min #1	—	14 sek	



# Testy

## All workflows

Showing runs from all workflows

Filter workflow runs

10 workflow runs				Event ▾	Status ▾	Branch ▾	Actor ▾
✓	Update Jenkinsfile	main	8 hours ago 14s	...			
Run Python tests #11: Commit <a href="#">0e2a852</a> pushed by PiotruOlasik							
✓	Merge pull request #2 from Grzywocz-W/main	main	yesterday 14s	...			
Run Python tests #10: Commit <a href="#">0ccde50</a> pushed by PiotruOlasik							



# Podsumowanie i wnioski

## Podsumowanie i wnioski

- Projekt stanowi cenne wprowadzenie dla osób, które chciałyby zgłębić tajniki automatyzacji w procesie wytwarzania oprogramowania i osób które chcą rozwijać się w kierunku DevOps



# Podsumowanie i wnioski

## Podsumowanie i wnioski

- Projekt stanowi cenne wprowadzenie dla osób, które chciałyby zgłębić tajniki automatyzacji w procesie wytwarzania oprogramowania i osób które chcą rozwijać się w kierunku DevOps
- Przedstawiliśmy podstawowe możliwości wykorzystania narzędzi do automatyzacji w procesie wytwarzania oprogramowania, automatyzując testy dla naszej aplikacji.



# Podsumowanie i wnioski

## Podsumowanie i wnioski

- Projekt stanowi cenne wprowadzenie dla osób, które chciałyby zgłębić tajniki automatyzacji w procesie wytwarzania oprogramowania i osób które chcą rozwijać się w kierunku DevOps
- Przedstawiliśmy podstawowe możliwości wykorzystania narzędzi do automatyzacji w procesie wytwarzania oprogramowania, automatyzując testy dla naszej aplikacji.
- Projekt również ma na celu podkreślenie jak bardzo ważne w obecnej branży IT jest wykorzystanie narzędzi do automatyzacji gdyż znacznie skracają one czas wdrożenia aplikacji oraz już teraz są popularnym narzędziem, które z pewnością będzie się dynamicznie rozwijać w przyszłości w miarę postępu technologicznego.



# Dziękujemy!

## Dziękujemy

