

„Azure DevOps Pipelines as Code”, czyli budowanie pipeline w oparciu o YAML

Jakub Podoba
Senior Microsoft Azure DevOps

softserve

WHOAMI

Obecnie:

- Senior Microsoft Azure DevOps @Soft**Serve**
- CEO @hotbee.eu
- Freelancer

 <https://www.linkedin.com/in/jakubpodoba/>

 jpodoba@hotbee.eu

 @jpodoba1

 <https://hotbee.eu/blog>

 <https://github.com/jpodoba>

Po pracy:

Mąż i ojciec,

Pasjonat lotnictwa,

Zapalony Azurowiec,            



Agenda

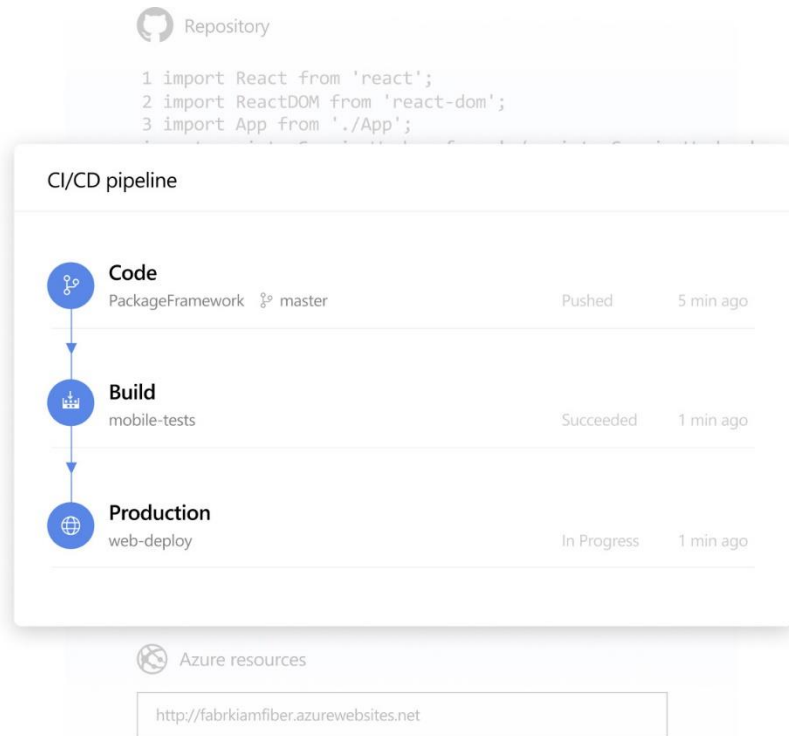
1. Czym są Azure pipelines
2. Klasyczny edytor vs YAML
3. Zalety YAML
4. Koncepcja Multi-Stage Pipelines
5. YAML structure
6. Dema
7. Q & A



Microsoft Azure
User Group Poland

Wrocław

Azure pipelines

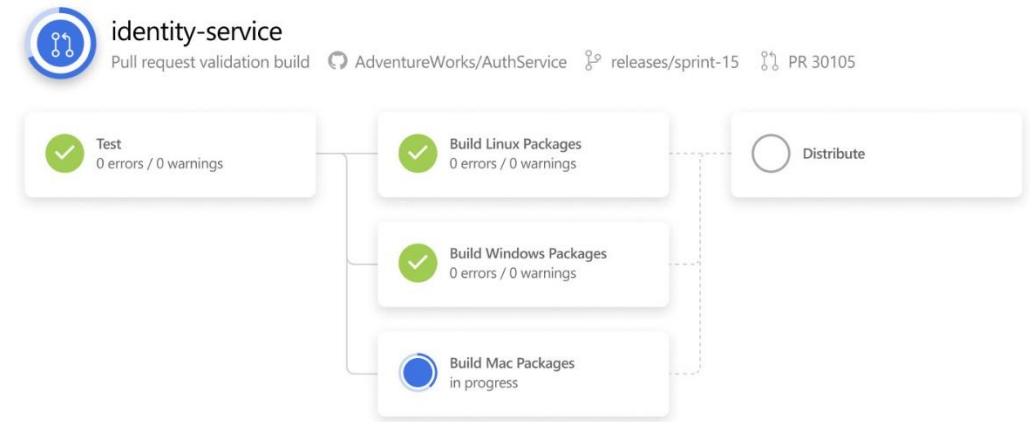


Azure Pipelines

Azure pipelines, dlaczego warto:

Czy warto używać?

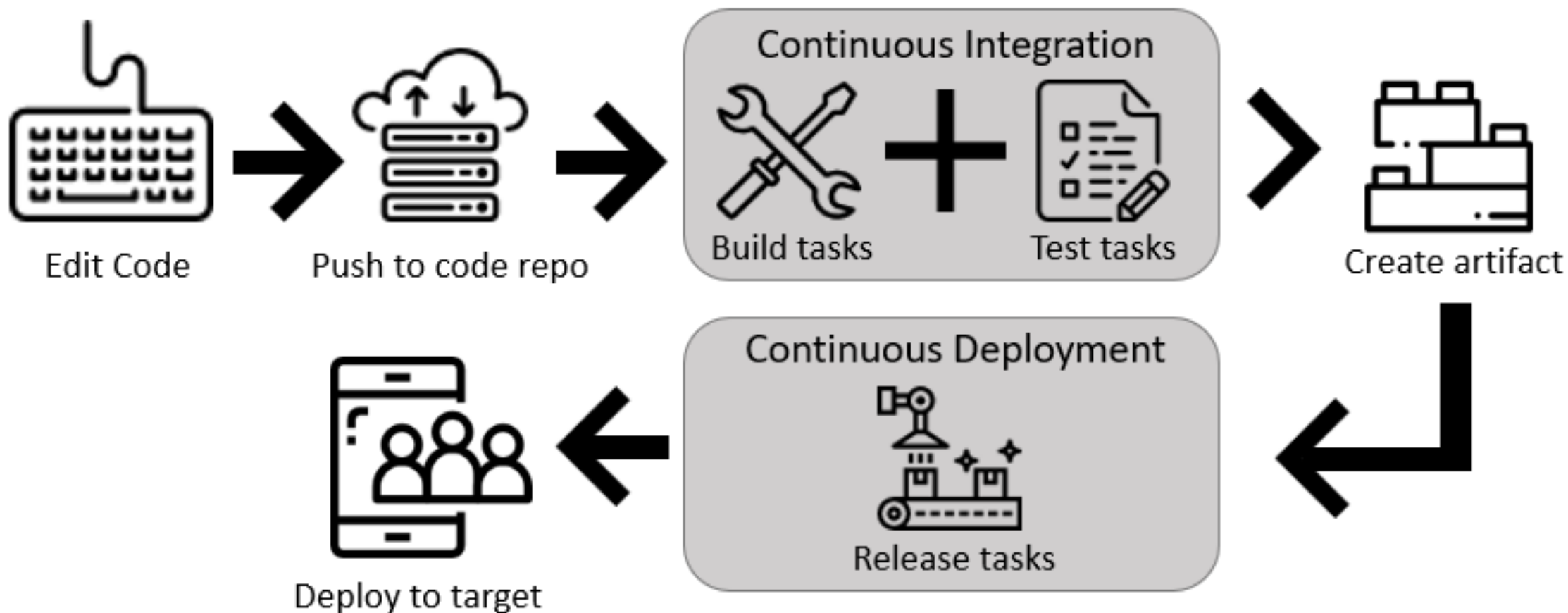
- Darmowe***
- Dowolny język, dowolna platforma,
- Kontenery i platforma Kubernetes,
- Wdrażanie w dowolnej chmurze,
- Możecie robić deploy do różnych systemów jednocześnie,
- Agenci dla systemów Linux, macOS i Windows hostowani przez firmę Microsoft
- Integracja z GitHub



softserve

*** - dla publicznych projektów, bądź 1800min (30h) za darmo na miesiąc dla projektów prywatnych

Klasyczny edytor



Klasyczny edytor - build

The screenshot displays the classic build editor in Azure DevOps. The breadcrumb navigation at the top shows the path: **hotbee** / **Parts Unlimited** / **Pipelines**. A search bar and utility icons are located in the top right corner.

The left sidebar contains a tree view with the following items:

- PU** (Project icon)
- +** (Expand icon)
- Pipeline** (Build pipeline)
- Get sources** (PartsUnlimited, master)
- Phase 1** (Run on agent)
- NuGet restore** (NuGet Installer)
- Build solution** (Visual Studio build)
- Test Assemblies** (Visual Studio Test)
- Copy Files** (Copy files)
- Publish Artifact** (Publish build artifacts)

The main content area is divided into two sections:

Task List:

- Tasks:** Variables, Triggers, Options, Retention, History
- Save & queue** (dropdown), **Discard**, **Summary**, **Queue**, **...**

Configuration Fields:

- Name ***: PartsUnlimitedE2E
- Agent pool ***: Azure Pipelines (with **Pool information** and **Manage** links)
- Agent Specification ***: vs2017-win2016

Parameters ⓘ

This pipeline doesn't have any pipeline parameters. Create them to share the most important settings between tasks and change them in one place.

[Learn more](#)

Klasyczny edytor - release

The screenshot displays the 'Parts Unlimited E2E' release editor. The breadcrumb navigation at the top reads 'hotbee / Parts Unlimited / Pipelines / Releases / PartsUnlimitedE2E'. A search bar is located in the top right corner. Below the breadcrumb, the page title is 'All pipelines > PartsUnlimitedE2E'. On the right side of this section, there are buttons for 'Save', 'Create release', and 'View releases'. A horizontal menu below the title includes 'Pipeline' (selected), 'Tasks', 'Variables', 'Retention', 'Options', and 'History'. The main workspace is divided into two panels: 'Artifacts' on the left and 'Stages' on the right. The 'Artifacts' panel contains a single artifact named 'PartsUnlimitedE2E' and a 'Schedule not set' indicator. The 'Stages' panel contains three stages in a sequence: 'Dev' (1 job, 2 tasks), 'QA' (1 job, 1 task), and 'Production' (1 job, 1 task). Each stage is represented by a box with a lightning bolt icon, a person icon, and a status indicator (a red circle with an exclamation mark for 'Dev' and 'QA', and a green circle with a checkmark for 'Production').

hotbee / Parts Unlimited / Pipelines / Releases / PartsUnlimitedE2E

Search

All pipelines > PartsUnlimitedE2E

Save Create release View releases

Pipeline Tasks Variables Retention Options History

Artifacts | + Add

Stages | + Add

PartsUnlimitedE2E

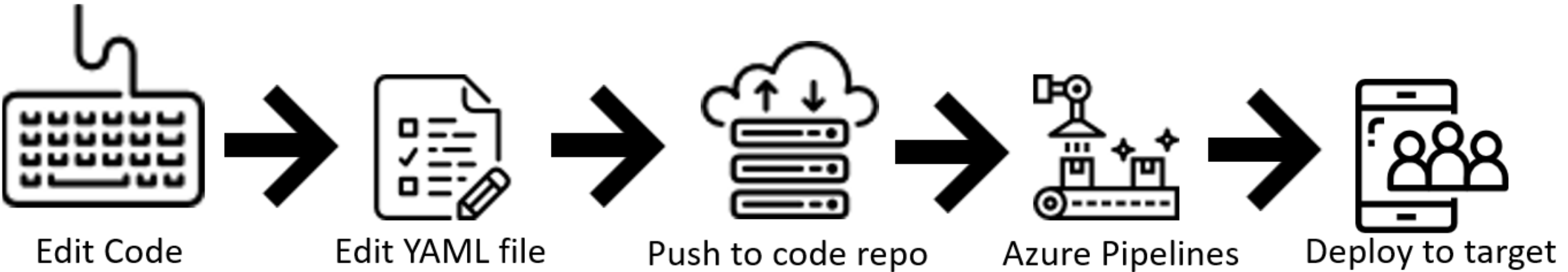
Schedule not set

Dev
1 job, 2 tasks

QA
1 job, 1 task

Production
1 job, 1 task

Azure Pipelines jako kod (YAML)



Azure Pipelines jako kod (YAML)

The screenshot displays the Azure Pipelines web interface for a pipeline named 'PartsUnlimited'. The breadcrumb navigation at the top reads: hotbee / Parts Unlimited YAML / Pipelines / Builds / PartsUnlimited / YAML. The main content area shows the YAML configuration for the pipeline, with line numbers 5 through 39. The configuration includes a trigger for 'master', a pool with VM image 'VS2017-Win2016', and several steps: 'NuGetToolInstaller@0', 'NuGetCommand@2', 'VSBuild@1', 'VSTest@2', and 'AzureRmWebAppDeployment@4'. The 'VSBuild@1' step is expanded, showing its inputs and settings. On the right side, there is a 'Tasks' panel with a search bar and a list of available tasks, including .NET Core, Android signing, Ant, App Center distribute, App Center test, Archive files, ARM Outputs, ARM template deployment, Azure App Service deploy, Azure App Service manage, Azure App Service Settings, Azure CLI, and Azure Cloud Service deployment.

```
5
6 trigger:
7   - master
8
9 pool:
10  - vmImage: 'VS2017-Win2016'
11
12 variables:
13   - solution: '**/*.sln'
14   - buildPlatform: 'Any CPU'
15   - buildConfiguration: 'Release'
16
17 steps:
18   - task: NuGetToolInstaller@0
19
20   - task: NuGetCommand@2
21     inputs:
22       - restoreSolution: '$(solution)'
23
24   - task: VSBuild@1
25     inputs:
26       - solution: '$(solution)'
27       - msbuildArgs: '/p:DeployOnBuild=true /p:WebPublishMethod=Package /p:PackageAsSingleFile=true /p:SkipInvalidConfigurations=true /p:PackageLocation="$(build.artifactstagingdirectory)'
28       - platform: '$(buildPlatform)'
29       - configuration: '$(buildConfiguration)'
30
31   - task: VSTest@2
32     inputs:
33       - platform: '$(buildPlatform)'
34       - configuration: '$(buildConfiguration)'
35
36   - task: AzureRmWebAppDeployment@4
37     inputs:
38       - ConnectedServiceName: ''
39       - WebAppName: ''
40     settings:
41       - Package: '$(build.artifactstagingdirectory)/*.zip'
```

Tasks

- .NET Core**
Build, test, package, or publish a dotnet application...
- Android signing**
Sign and align Android APK files
- Ant**
Build with Apache Ant
- App Center distribute**
Distribute app builds to testers and users via Visu...
- App Center test**
Test app packages with Visual Studio App Center
- Archive files**
Compress files into .7z, .tar.gz, or .zip
- ARM Outputs**
This task reads the output values of an ARM depl...
- ARM template deployment**
Deploy an Azure Resource Manager (ARM) templ...
- Azure App Service deploy**
Deploy to Azure App Service a web, mobile, or AP...
- Azure App Service manage**
Start, stop, restart, slot swap, slot delete, install sit...
- Azure App Service Settings**
Update/Add App settings an Azure Web App for ...
- Azure CLI**
Run Azure CLI commands against an Azure subscr...
- Azure Cloud Service deployment**
Deploy an Azure Cloud Service

Azure Pipelines jako kod (YAML)

The screenshot displays the Azure Pipelines web interface for a pipeline named 'PartsUnlimited'. The breadcrumb navigation at the top reads: hotbee / Parts Unlimited YAML / Pipelines / Builds / PartsUnlimited / YAML. The main editor shows the 'azure-pipelines.yml' file on the 'master' branch. The pipeline configuration is as follows:

```
5
6 trigger:
7   - master
8
9 pool:
10  - vmImage: 'VS2017-Win2016'
11
12 variables:
13   - solution: '**/*.sln'
14   - buildPlatform: 'Any CPU'
15   - buildConfiguration: 'Release'
16
17 steps:
18   - task: NuGetToolInstaller@0
19
20   - task: NuGetCommand@2
21     inputs:
22       - restoreSolution: '$(solution)'
23
24   - task: VSBuild@1
25     inputs:
26       - solution: '$(solution)'
27       - msbuildArgs: '/p:DeployOnBuild=true /p:WebPublishMethod=Package /p:PackageAsSingleFile=true /p:SkipInvalidConfigurations=true /p:PackageLocation="$(build.artifactstagingdirectory)'
28       - platform: '$(buildPlatform)'
29       - configuration: '$(buildConfiguration)'
30
31   - task: VSTest@2
32     inputs:
33       - platform: '$(buildPlatform)'
34       - configuration: '$(buildConfiguration)'
35
36   - task: AzureRmWebAppDeployment@4
37     inputs:
38       - ConnectedServiceName: ''
39       - WebAppName: ''
40       - Package: '$(build.artifactstagingdirectory)/*.zip'
```

On the right side, the 'Tasks' panel is visible, listing various tasks such as .NET Core, Android signing, Ant, App Center distribute, App Center test, Archive files, ARM Outputs, ARM template deployment, Azure App Service deploy, Azure App Service manage, Azure App Service Settings, Azure CLI, and Azure Cloud Service deployment. The 'ConnectedServiceName' input field for the 'AzureRmWebAppDeployment@4' task is highlighted with a red rectangle.

Azure Pipelines jako kod (YAML)

The screenshot displays the Azure Pipelines web interface. The breadcrumb navigation at the top reads: hotbee / Parts Unlimited YAML / Pipelines / PartsUnlimited / 20200424.1. A search bar and utility icons are located in the top right corner.

Jobs in run #20200424.1
PartsUnlimited

Jobs

Job	Duration
Initialize job	8s
Checkout PartsUnlimite...	7s
NuGetToolInstaller	1s
NuGetCommand	35s
VSBuild	34s
VSTest	18s
AzureRmWebAppDep...	<1s
Post-job: Checkout Pa...	<1s

AzureRmWebAppDeployment

View raw log

```
1 Starting: AzureRmWebAppDeployment
2 =====
3 Task      : Azure App Service deploy
4 Description : Deploy to Azure App Service a web, mobile, or API app using Docker, Java, .NET, .NET Core, Node.js, PHP, Python, or Ruby
5 Version    : 4.163.6
6 Author     : Microsoft Corporation
7 Help       : https://aka.ms/azureappservicetroubleshooting
8 =====
9 ##[error]Error: Input required: ConnectedServiceName
10 Finishing: AzureRmWebAppDeployment
```

Demo 1 – pipeline github



softserve

Dlaczego YAML?

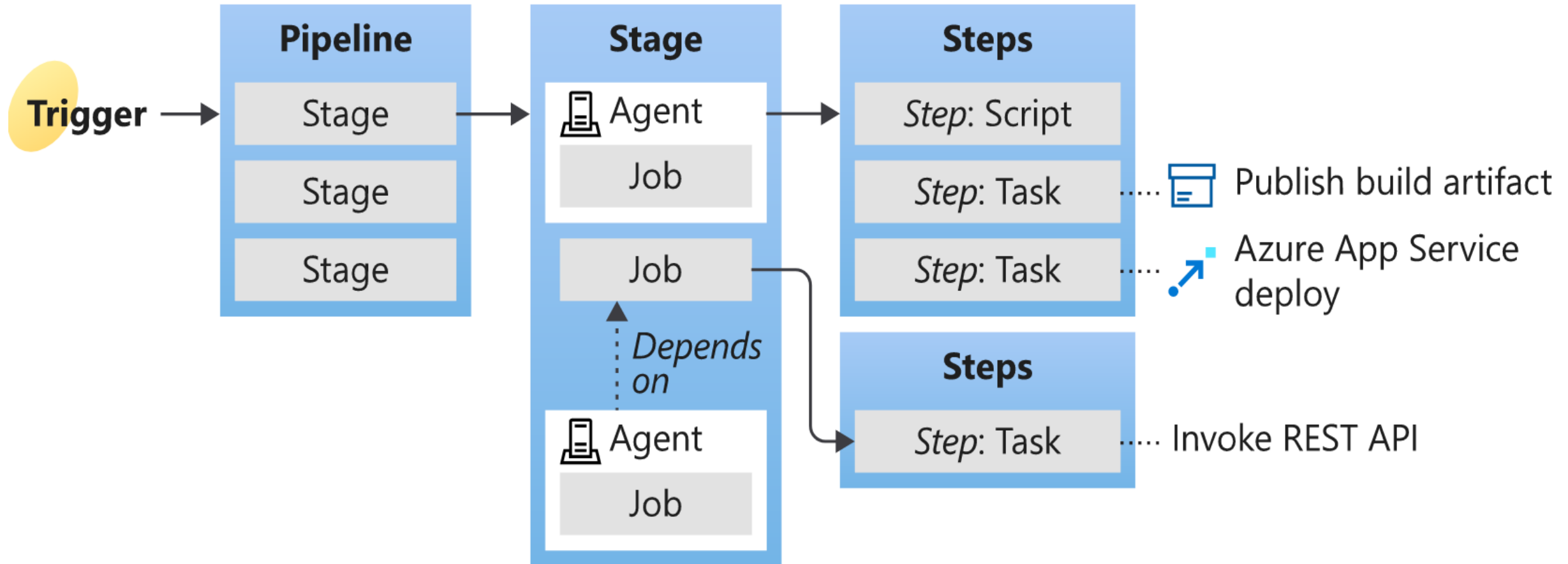
- Znany z poprzednich projektów, puppet, ansible, kubernetes
- Łatwiejszy w utrzymaniu niż edytor graficzny,
- Pipeline jest pisane deklaratywne,
- Wszystko w trzymane w GIT,
- YAML podąża za konkretnym branchem,
- Prosty do kopiowania i utrzymania,
- Dodatek w Visual Studio code,
- Prosty w konwersji z obecnego edytora,

Dostępne feature'y

Feature	YAML	Classic Build	Classic Release	Notes
Agents	Yes	Yes	Yes	Specifies a required resource on which the pipeline runs.
Approvals	Yes	No	Yes	Defines a set of validations required prior to completing a deployment stage.
Artifacts	Yes	Yes	Yes	Supports publishing or consuming different package types.
Caching	Yes	Yes	No	Reduces build time by allowing outputs or downloaded dependencies from one run to be reused in later runs. In Preview, available with Azure Pipelines only.
Conditions	Yes	Yes	Yes	Specifies conditions to be met prior to running a job.
Container jobs	Yes	No	No	Specifies jobs to run in a container.
Demands	Yes	Yes	Yes	Ensures pipeline requirements are met before running a pipeline stage. Requires self-hosted agents.
Dependencies	Yes	Yes	Yes	Specifies a requirement that must be met in order to run the next job or stage.
Deployment groups	Yes	No	Yes	Defines a logical set of deployment target machines.

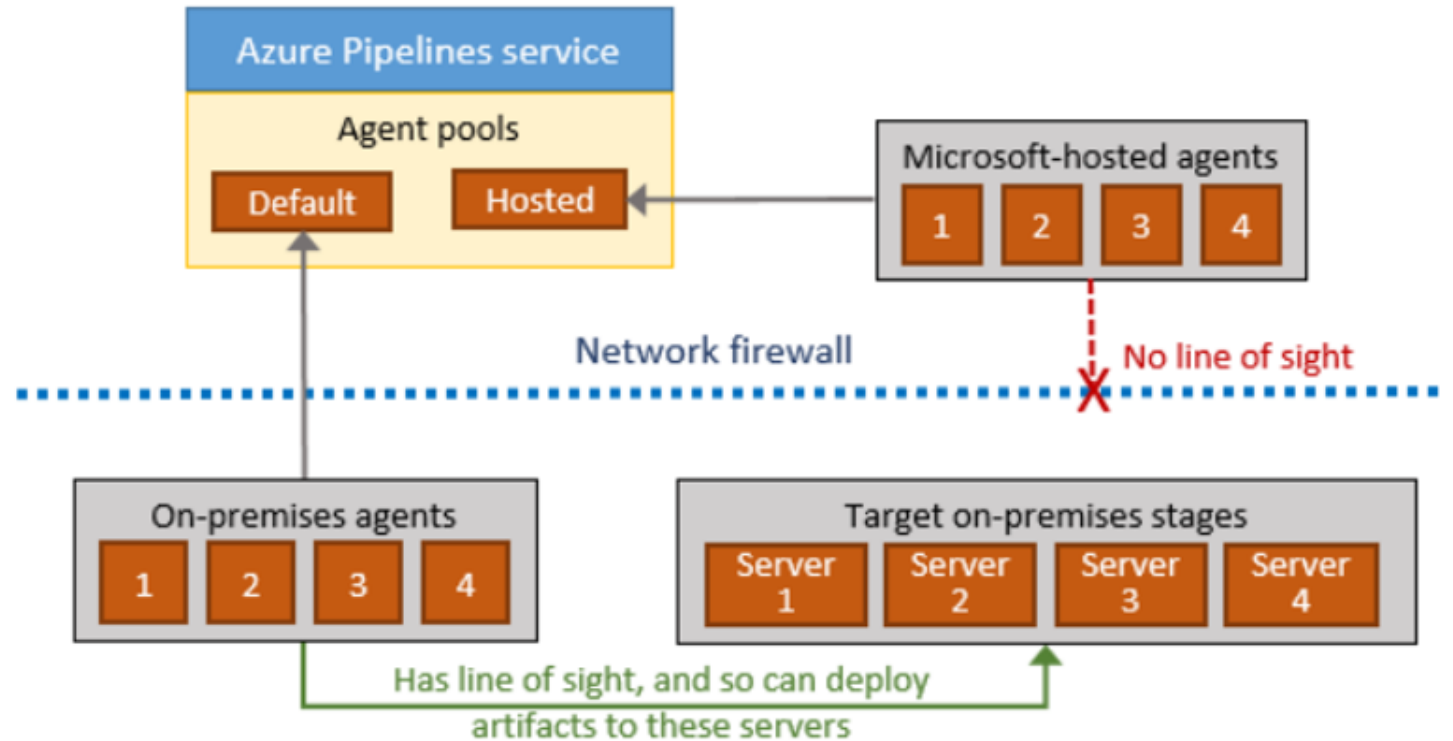
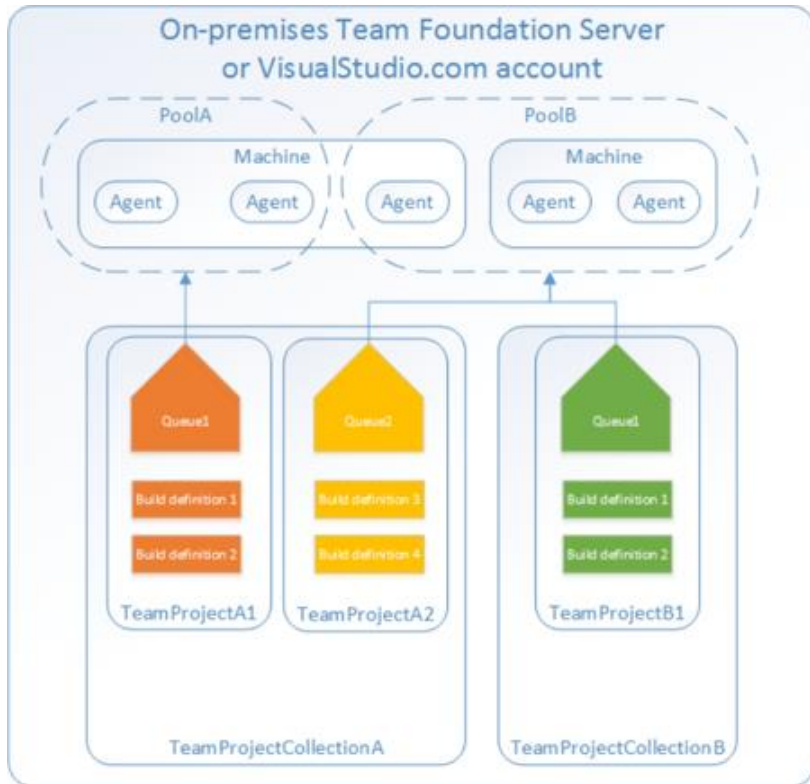
softserve

Koncept Azure DevOps pipeline



softserve

Agenci



softserve

Agenci – hostowani przez Microsoft

Image	Classic Editor Agent Specification	YAML VM Image Label	Included Software
Windows Server 2019 with Visual Studio 2019	<i>windows-2019</i>	<code>windows-latest</code> OR <code>windows-2019</code>	Link
Windows Server 2016 with Visual Studio 2017	<i>vs2017-win2016</i>	<code>vs2017-win2016</code>	Link
Ubuntu 18.04	<i>ubuntu-18.04</i>	<code>ubuntu-latest</code> OR <code>ubuntu-18.04</code>	Link
Ubuntu 16.04	<i>ubuntu-16.04</i>	<code>ubuntu-16.04</code>	Link
macOS X Mojave 10.14	<i>macOS-10.14</i>	<code>macOS-10.14</code>	Link
macOS X Catalina 10.15	<i>macOS-10.15</i>	<code>macOS-latest</code> OR <code>macOS-10.15</code>	Link

softserve

Struktura YAML

- Pipeline
 - Stage A
 - Job 1
 - Step 1.1
 - Step 1.2
 - ...
 - Job 2
 - Step 2.1
 - Step 2.2
 - ...
 - Stage B
 - ...

YAML

```
name: string # build numbering format
resources:
  pipelines: [ pipelineResource ]
  containers: [ containerResource ]
  repositories: [ repositoryResource ]
variables: # several syntaxes, see specific section
trigger: trigger
pr: pr
stages: [ stage | templateReference ]
```

YAML

```
# ... other pipeline-level keywords
jobs: [ job | templateReference ]
```

YAML

```
# ... other pipeline-level keywords
steps: [ script | bash | pwsh | powershell | checkout | task | templateReference ]
```

Demo 2 – prosty pipeline



softserve

Service connection

The screenshot displays the Azure DevOps web interface. The left sidebar contains navigation links: Labs, Overview, Boards, Repos, Pipelines, Test Plans, Artifacts, and Project settings. The main content area is titled 'Service connections' and includes a 'Filter by keywords' search bar. A red arrow points from the 'Service connections*' link in the left sidebar to the 'Service connections' page. Another red arrow points from the 'Service connections*' link to the 'New service connection' modal. The modal is titled 'New service connection' and prompts the user to 'Choose a service or connection type'. It features a search bar and a list of service types: Azure Classic, Azure Repos/Team Foundation Server, Azure Resource Manager, Azure Service Bus, Bitbucket Cloud, Chef, Docker Host, Docker Registry, Generic, GitHub, and GitHub Enterprise Server. A red arrow points from the 'Next' button in the modal to the 'Service connections*' link in the sidebar. A third red arrow points from the 'Project settings' link in the sidebar to the 'Service connections*' link.

Service connections

Filter by keywords

New service connection

Choose a service or connection type

Search connection types

- ☒ Azure Classic
- ☐ Azure Repos/Team Foundation Server
- ☐ Azure Resource Manager
- ☐ Azure Service Bus
- ☐ Bitbucket Cloud
- ☐ Chef
- ☐ Docker Host
- ☐ Docker Registry
- ☐ Generic
- ☐ GitHub
- ☐ GitHub Enterprise Server

[Learn more](#)

Next


Service connection

New service connection


Choose a service or connection type

Search connection types


☐

 Incoming WebHook


☐

 Jenkins


☐

 Jira


☐

 Kubernetes


☐

 Maven


☐

 NuGet


☐

 Other Git


☐

 Python package download


☐

 Python package upload

☐

 SSH

☐

 Service Fabric

[Learn more](#)


Next

New service connection


Choose a service or connection type

Search connection types


☐

 Kubernetes


☐

 Maven


☐

 NuGet


☐

 Other Git


☐

 Python package download


☐

 Python package upload


☐

 SSH


☐

 Service Fabric


☐

 Subversion

☐

 Visual Studio App Center

☐

 npm

[Learn more](#)

Next

Nested YAML

- Steps,
- Jobs,
- Stage,
- Variables,

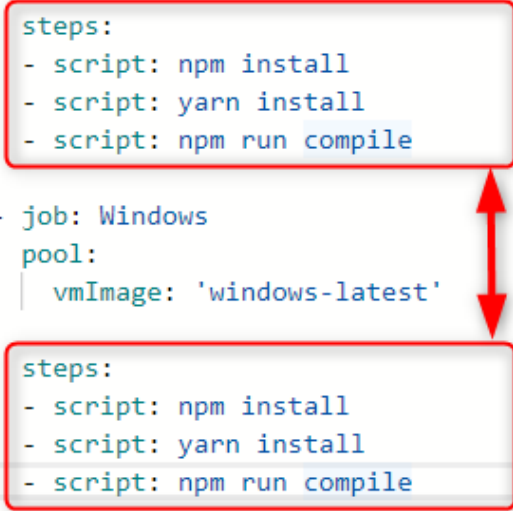
```
1 # File: azure-pipeline-steps01.yml
2
3 jobs:
4 - job: Linux
5   pool:
6     vmImage: 'ubuntu-latest'
7
8   steps:
9     - script: npm install
10    - script: yarn install
11    - script: npm run compile
12
13 - job: Windows
14   pool:
15     vmImage: 'windows-latest'
16
17   steps:
18     - script: npm install
19     - script: yarn install
20     - script: npm run compile
```

softserve

Nested YAML

- Steps,
- Jobs,
- Stage,
- Variables,

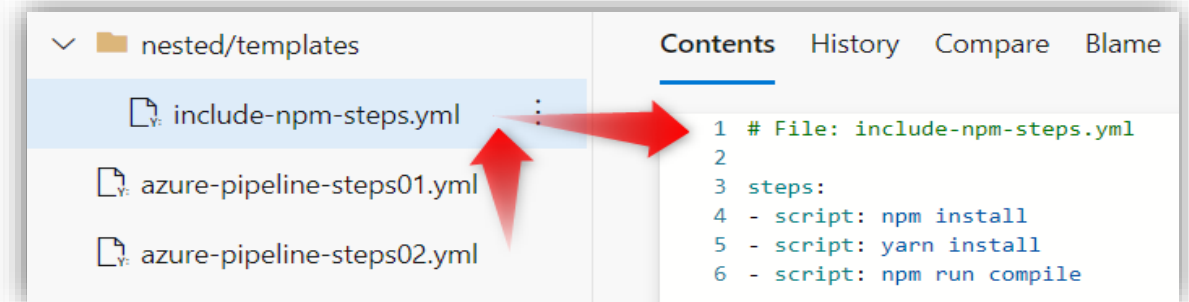
```
1 # File: azure-pipeline-steps01.yml
2
3 jobs:
4 - job: Linux
5   pool:
6     vmImage: 'ubuntu-latest'
7
8   steps:
9     - script: npm install
10    - script: yarn install
11    - script: npm run compile
12
13 - job: Windows
14   pool:
15     vmImage: 'windows-latest'
16
17   steps:
18     - script: npm install
19     - script: yarn install
20     - script: npm run compile
```

A diagram illustrating nested YAML structure. Two red rectangular boxes highlight the 'steps' sections for the 'Linux' and 'Windows' jobs. A red double-headed vertical arrow is positioned between these two boxes, indicating a relationship or comparison between the two sets of steps.

softserve

Nested - steps

```
1 # File: azure-pipeline-steps02.yml
2
3 jobs:
4 - job: Linux
5   pool:
6     vmImage: 'ubuntu-latest'
7   steps:
8     - template: templates/include-npm-steps.yml # Template reference
9 - job: Windows
10  pool:
11    vmImage: 'windows-latest'
12  steps:
13    - template: templates/include-npm-steps.yml # Template reference
```



softserve

Demo 3 – nested pipeline



softserve

Triggers

Classic build pipelines and YAML pipelines

Continuous integration (CI) triggers vary based on the type of repository you build in your pipeline.

- CI triggers in Azure Repos Git
- CI triggers in GitHub
- CI triggers in BitBucket Cloud
- CI triggers in TFVC

```
# A pipeline with no CI trigger
trigger: none
```

Pull request validation (PR) triggers also vary based on the type of repository.

- PR triggers in Azure Repos Git
- PR triggers in GitHub
- PR triggers in BitBucket Cloud

```
# specific path build
trigger:
  branches:
    include:
      - master
      - releases/*
  paths:
    include:
      - docs/*
    exclude:
      - docs/README.md
```

```
# specific tag
trigger:
  tags:
    include:
      - v2.*
    exclude:
      - v2.0
```

softserve

Triggers - scheduled

```
schedules:  
- cron: "0 0 * * *"  
  displayName: Daily midnight build  
  branches:  
    include:  
      - master  
      - releases/*  
    exclude:  
      - releases/ancient/*  
- cron: "0 12 * * 0"  
  displayName: Weekly Sunday build  
  branches:  
    include:  
      - releases/*  
always: true
```

```
# YAML file in the release branch  
schedules:  
- cron: "0 0 * * *"  
  displayName: Daily midnight build  
  branches:  
    include:  
      - master  
  
# YAML file in the master branch with release added to the branches list  
schedules:  
- cron: "0 0 * * *"  
  displayName: Daily midnight build  
  branches:  
    include:  
      - master  
      - release
```

mm	HH	DD	MM	DW	
\	\	\	\	\	Days of week
\	\	\	\		Months
\	\	\			Days
\	\				Hours
\					Minutes

softserve

Triggers - pipeline

```
# this is being defined in app-ci pipeline
resources:
  pipelines:
    - pipeline: securitylib    # Name of the pipeline resource
      source: security-lib-ci # Name of the triggering pipeline
      trigger:
        branches:
          - releases/*
          - master
```

softserve

Variables

- Użytkownika,
- Systemowe, (**predefiniowane**)
- Środowiskowe, (**w zależności od OS**)

- Trzy poziomy definiowania:
- **Root level,**
- **Stage level,**
- **Job level,**

```
variables:  
  global_variable: value    # this is available to all jobs  
  
jobs:  
- job: job1  
  pool:  
    vmImage: 'ubuntu-16.04'  
    variables:  
      job_variable1: value1  # this is only available in job1  
    steps:  
      - bash: echo $(global_variable)  
      - bash: echo $(job_variable1)  
      - bash: echo $JOB_VARIABLE1 # variables are available in the script environment too  
  
- job: job2  
  pool:  
    vmImage: 'ubuntu-16.04'  
    variables:  
      job_variable2: value2  # this is only available in job2  
    steps:  
      - bash: echo $(global_variable)  
      - bash: echo $(job_variable2)  
      - bash: echo $GLOBAL_VARIABLE
```

softserve

Conditions

- „by design” wszystko wyzwalane jest jednocześnie,
- „by design” nic od siebie nie zależy,
- Są po to aby ustalić zależności,
- I ograniczenia, (niezastąpione przy multi stage)

Run for the master branch, if succeeding

```
and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/master'))
```

Run if the branch is not master, if succeeding

```
and(succeeded(), ne(variables['Build.SourceBranch'], 'refs/heads/master'))
```

Run for user topic branches, if succeeding

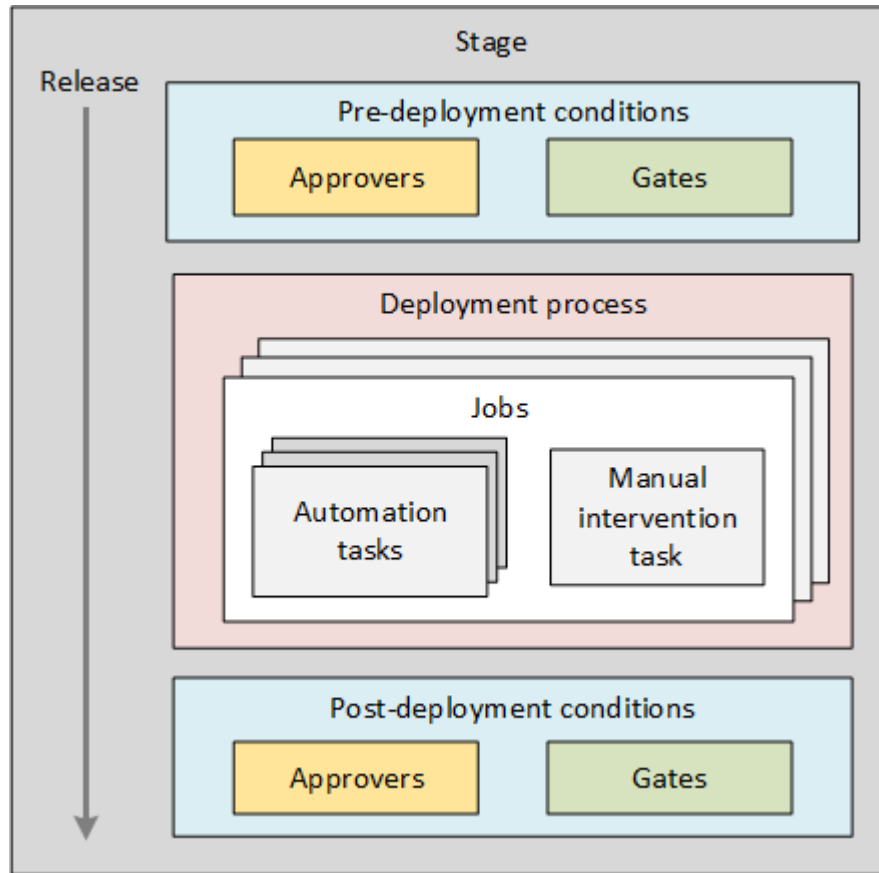
```
and(succeeded(), startsWith(variables['Build.SourceBranch'], 'refs/heads/users/'))
```

- zagadka

```
condition: and(succeeded(), ne(variables['Build.Reason'], 'PullRequest'))
```

softserve

Approvals



The screenshot shows the "Add check" dialog in Azure DevOps. It has a search bar at the top and a list of check types below it. The "Approvals" check type is selected.

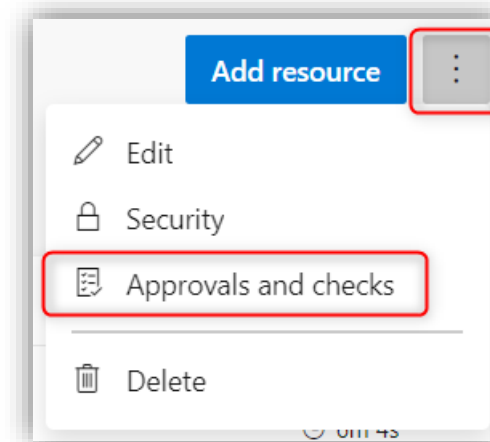
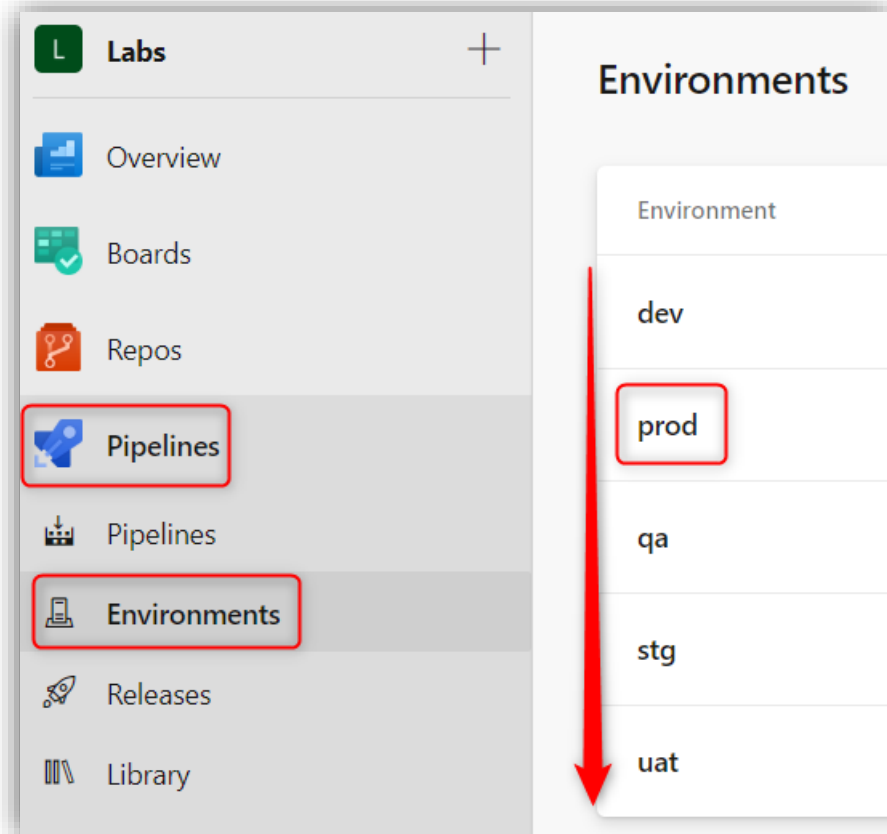
Add check [Close]

Search check types

- ☒ **Approvals**
Approvers should grant approval for deployment
- ☐ **Evaluate artifact (preview)**
Ensure artifacts adhere to custom policies (container images only)
- ☐ **Invoke Azure Function**
Invoke an Azure Function
- ☐ **Invoke REST API**
Invoke a REST API as a part of your pipeline.
- ☐ **Query Azure Monitor alerts**
Observe the configured Azure Monitor rules for active alerts
- ☐ **Required template**
Ensure the pipeline extends one or more YAML templates

softserve

Approvals - manual



softserve

Approvals - manual

```
- stage: PROD
  displayName: 'PROD(CD)'
  condition: and(succeeded('STG'), eq(variables['Build.SourceBranch'], 'refs/heads/master'))
  dependsOn:
  | - BLD
  | - STG
  variables:
  | stage: 'prod'
  jobs:
  - deployment: Primary_NorthErope
    pool:
      vmImage: $(vmImage)
    environment: prod
    strategy:
      runOnce:
        deploy:
          steps:
            - template: 'pipelines/infrastructure/deploy.yml'
              parameters: {type: 'primary', spn: 'azure-devops-cloud4it-spn', location: 'northeurope'}
            - template: 'pipelines/application/deploy.yml'
              parameters: {type: 'primary', spn: 'azure-devops-cloud4it-spn'}
```

softserve

Migracja do YAML

- Prosta dla „build”,
- Prosta dla „release”
 - Ale nie aż tak

The image shows a screenshot of the Azure DevOps interface. In the foreground, a 'Copy to clipboard' dialog box is open, displaying a YAML snippet for an agent pool. The dialog text explains that the build pipeline references undefined variables and provides links to create or edit the build pipeline. The YAML snippet is as follows:

```
pool:
  name: Azure Pipelines
#Your build pipeline references an undefined variable named 'Parameters.RestoreBuildProjects'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
#Your build pipeline references an undefined variable named 'Parameters.RestoreBuildProjects'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
#Your build pipeline references the 'BuildConfiguration' variable, which you've selected to be settable at queue time. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab, and then select the option to make it settable at queue time. See https://go.microsoft.com/fwlink/?linkid=865971
#Your build pipeline references an undefined variable named
```

A 'Copy to clipboard' button is visible at the bottom right of the dialog. In the background, the 'Agent job' configuration page is visible, showing fields for 'Display name' (Agent job 1) and 'Agent pool' (<inherit from pipeline>). A 'View YAML' button is highlighted with a red box in the top right corner of the configuration page.

softserve

Migracja do YAML

The image shows a screenshot of the Azure DevOps web interface. In the background, the 'All pipelines' view for 'Contoso-Pro' is visible. The 'Tasks' tab is selected, showing a list of tasks for the 'DEV' deployment process. The tasks listed are 'Agent job' (Run on agent), 'Collecting parameters' (Azure PowerShell), and 'Deploy Shared Resources' (Azure resource group deployment). The 'Deploy Shared Resources' task is highlighted with a red box. A red arrow points from this task to a 'View YAML' button, which is also highlighted with a red box. In the foreground, a 'Copy to clipboard' dialog box is open. It contains the following text: 'Below is a clipboard-friendly view of your selection. To copy to the clipboard, either right-click and choose 'Copy' from the browser's context menu or press Ctrl+C. [more information about YAML builds]'. Below this text is a code block containing the following YAML content:

```
#Your build pipeline references an undefined variable named 'rgName'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
#Your build pipeline references an undefined variable named 'location'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
#Your build pipeline references an undefined variable named 'regionCode'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
steps:
- task: AzureResourceGroupDeployment@2
  displayName: 'Deploy Shared Resources'
```

 At the bottom right of the dialog box is a blue button labeled 'Copy to clipboard'.

Copy to clipboard

Below is a clipboard-friendly view of your selection. To copy to the clipboard, either right-click and choose 'Copy' from the browser's context menu or press Ctrl+C. [more information about YAML builds]

```
#Your build pipeline references an undefined variable named 'rgName'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
#Your build pipeline references an undefined variable named 'location'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
#Your build pipeline references an undefined variable named 'regionCode'. Create or edit the build pipeline for this YAML file, define the variable on the Variables tab. See https://go.microsoft.com/fwlink/?linkid=865972
steps:
- task: AzureResourceGroupDeployment@2
  displayName: 'Deploy Shared Resources'
```

Copy to clipboard

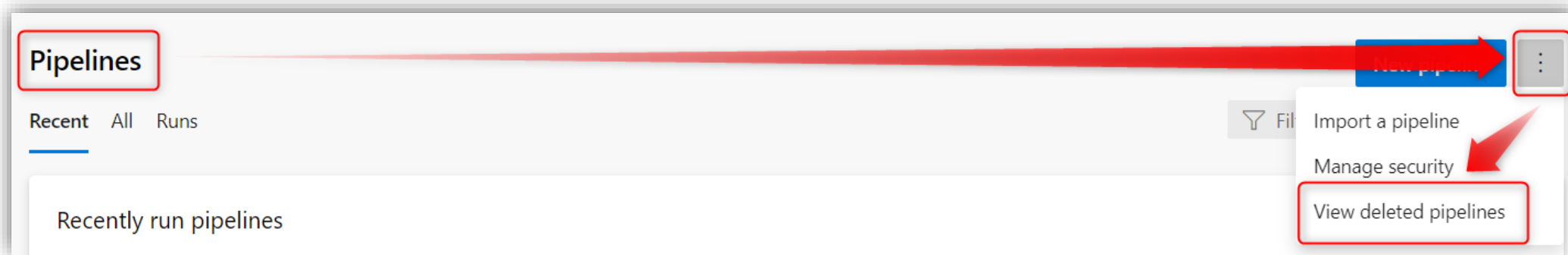
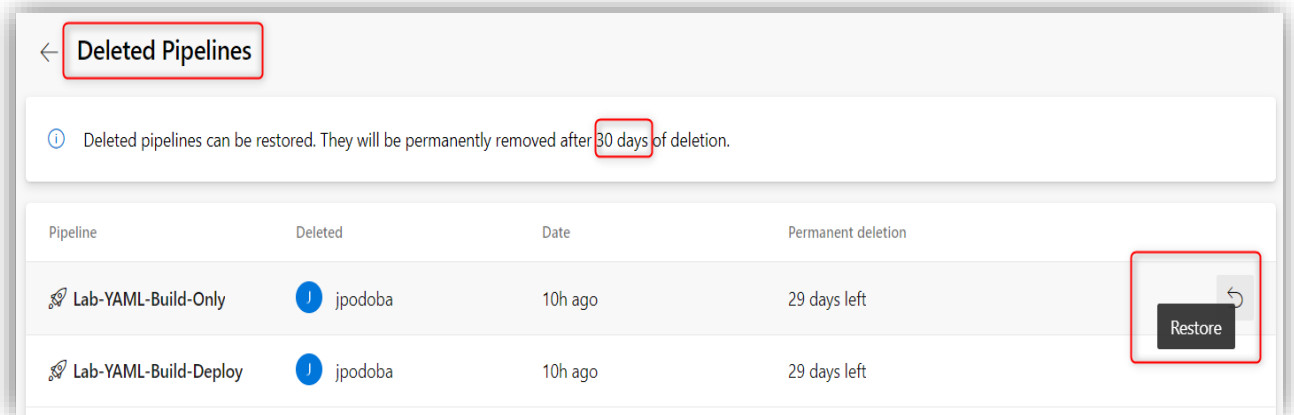
Save Create release View release

View YAML

softserve

Historia i oddzyskiwanie

- „pipeline as code” – jest w GIT,
- proste w śledzeniu zmian,
- proste w cofaniu zmian,
- proste w oddzyskiwaniu po usunięciu,
- Ale co z **pipeline id** ???



softserve

Multi stage pipeline

```
Build
| * Infrastructure
| * Application
```

1

```
Build
| * Infrastructure
| * Application
|-> develop
|   |-> Deploy DEV (Primary Region)
|   |   * Primary Region
|   |-> Deploy QA (Primary Region)
|   |   * Primary Region
```

2

```
Build
| * Infrastructure
| * Application
|-> develop
|   |-> Deploy DEV (Primary Region)
|   |   * Primary Region
|   |-> Deploy QA (Primary Region)
|   |   * Primary Region
|- master
|   |-> Deploy UAT
|   |   * Primary Region
|   |-> Deploy STG
|   |   * Primary Region
|   |-> Deploy PROD
|   |   * Primary Region
|   |   * Secondary Region
```

3

softserve

Multi stage pipeline

```
Build
| * Infrastructure
| * Application
```

1

```
Build
| * Infrastructure
| * Application
|-> develop
|   |-> Deploy DEV (Primary Region)
|   |   * Primary Region
|   |-> Deploy QA (Primary Region)
|   |   * Primary Region
```

2

```
Build
| * Infrastructure
| * Application
|-> develop
|   |-> Deploy DEV (Primary Region)
|   |   * Primary Region
|   |-> Deploy QA (Primary Region)
|   |   * Primary Region
|- master
|   |-> Deploy UAT
|   |   * Primary Region
|   |-> Deploy STG
|   |   * Primary Region
|   |-> Deploy PROD
|   |   * Primary Region
|   |   * Secondary Region
```

3

softserve

Demo 4 – Multi stage pipeline



softserve

Pytania ???



Przydate linki:

- <https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops>
- <https://github.com/microsoft/azure-pipelines-yaml>
- <https://azuredevopslabs.com/labs/azuredevops/yaml/>
- <https://github.com/jpodoba/Presentations/tree/master/2020-04-27-Cloud4it-Group>

Ankieta prelegenta:

<https://tinyurl.com/yamlwroclaw>



Dziękuję za uwagę

