**Azure DevOps Variables**

* Azure DevOps Variables <https://learn.microsoft.com/en-us/azure/devops/pipelines/process/variables?view=azure-devops&tabs=yaml%2Cbatch>
* predefined variables <https://learn.microsoft.com/en-us/azure/devops/pipelines/build/variables?view=azure-devops&tabs=yaml>
* Creating user defined variables <https://learn.microsoft.com/en-us/azure/devops/pipelines/process/variables?view=azure-devops&tabs=yaml%2Cbatch#user-defined-variables>
* Azure DevOps Pipeline Parameters <https://learn.microsoft.com/en-us/azure/devops/pipelines/yaml-schema/parameters-parameter?view=azure-pipelines>

---

trigger:

- master

variables:

goals: package

parameters:

- name: mavenGoal

displayName: Maven Goal

type: string

default: package

values:

- package

- clean package

- clean install

- install

jobs:

- job: buildjob

displayName: Build and Package Game of life

pool:

vmImage: ubuntu-22.04

steps:

- task: Maven@3

inputs:

mavenPOMFile: 'pom.xml'

goals: ${{ parameters.mavenGoal }}

publishJUnitResults: true

testResultsFiles: '\*\*/surefire-reports/TEST-\*.xml'

javaHomeOption: 'JDKVersion'

jdkVersionOption: '1.8'

- task: CopyFiles@2

inputs:

Contents: "\*\*/target/gameoflife.war"

TargetFolder: $(Build.ArtifactStagingDirectory)

- task: PublishBuildArtifacts@1

inputs:

pathToPublish: $(Build.ArtifactStagingDirectory)

artifactName: GameOfLifeArtifacts

Azure DevOps parameters can be set during the execution of the pipeline  


* Azure DevOps Pipeline values that are passed before execution is parameter and variables are set and modified during pipeline execution.
* Setting variables from scripts in Azure DevOps Pipelines <https://learn.microsoft.com/en-us/azure/devops/pipelines/process/set-variables-scripts?view=azure-devops&tabs=powershell>

**Making Pipelines reusable using templates**

* <https://learn.microsoft.com/en-us/azure/devops/pipelines/process/templates?view=azure-devops> for official docs
* We have created a reusable template with a name generic-maven-java.yaml

parameters:

- name: javaVersion

type: string

default: '1.11'

- name: mavenGoal

type: string

default: package

- name: artifactPath

type: string

default: '\*\*/\*.jar'

- name: artifactName

type: string

default: generic

steps:

- task: Maven@3

inputs:

mavenPOMFile: 'pom.xml'

goals: ${{ parameters.mavenGoal }}

publishJUnitResults: true

testResultsFiles: '\*\*/surefire-reports/TEST-\*.xml'

javaHomeOption: 'JDKVersion'

jdkVersionOption: ${{ parameters.javaVersion}}

- task: CopyFiles@2

inputs:

Contents: ${{ parameters.artifactPath }}

TargetFolder: $(Build.ArtifactStagingDirectory)

- task: PublishBuildArtifacts@1

inputs:

pathToPublish: $(Build.ArtifactStagingDirectory)

artifactName: ${{ parameters.artifactName }}

* In the spring petclinic pipeline is now as simple as

---

trigger:

- main

pool:

vmImage: ubuntu-22.04

extends:

template: generic-maven-java.yaml

parameters:

javaVersion: '1.17'

mavenGoal: 'package'

artifactPath: '\*\*/target/spring-petclinic\*.jar'

artifactName: 'SPCArtifacts'

* Now lets try to reuse the template which is different git repo with the help of azure devops pipeline resources. <https://learn.microsoft.com/en-us/azure/devops/pipelines/process/templates?view=azure-devops#use-other-repositories> for usage
* Azure DevOps Pipline resources <https://learn.microsoft.com/en-us/azure/devops/pipelines/process/resources?view=azure-devops&tabs=schema>
* Our yaml

---

trigger:

- master

pool:

vmImage: ubuntu-22.04

resources:

repositories:

- repository: BuildTemplates

name: BuildTemplates

type: git

extends:

template: generic-maven-java.yaml@BuildTemplates

parameters:

javaVersion: '1.8'

mavenGoal: 'package'

artifactPath: '\*\*/target/gameoflife.war'

artifactName: 'GameOfLifeArtifacts'