YAML Generic Blueprint-based Pipeline Quick Reference

WORKSAFEBC

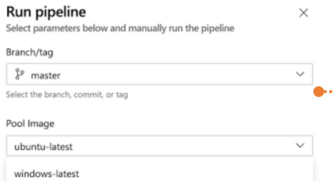
WORKING TO MAKE A DIFFERENCE

2021.04.30 v1.2 WS - See https://aka.ms/yaml for Azure DevOps Pipeline documentation

Parameters

You can specify parameters in templates and in the pipeline, as shown. They can be restricted to a subset of values.

- key:value pairs
- string : any
- ${{}} to dereference @ **queue** time



Run pipeline

Select parameters below and manually run the pipeline

Branch/tag: master

Select the branch, commit, or tag

Pool Image: ubuntu-latest

windows-latest

Triggers

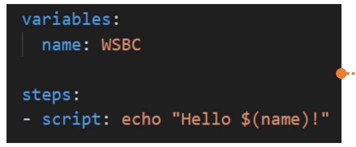
Use triggers to run a pipeline automatically.

- Continuous integration (CI) trigger
- Pull request validation (PR) trigger
- Comment triggers (GitHub only)
- Scheduled triggers
- Pipeline triggers (YAML)
- Build completion triggers (Classic Build)

Variables

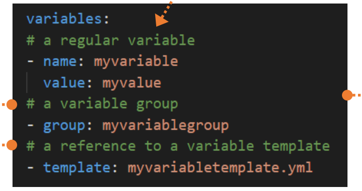
Variables are different from runtime parameters, which are typed and available during template parsing.

- key:value pairs
  - string : string
  - $(key) to dereference @ **run** time
- Inline variables
  - hardcoded
- Variable groups
- Template variables
- Pipeline variables
- Predefined variables
  - $(Build.SourceBranch)
  - https://docs.microsoft.com/en-us/azure/devops/pipelines/build/variables



```yaml
variables:
  name: WSBC

steps:
- script: echo "Hello $(name)!"
```

Variables switch to **key:value** if mixing with groups, templates



```yaml
variables:
# a regular variable
- name: myvariable
  value: myvalue
# a variable group
- group: myvariablegroup
# a reference to a variable template
- template: myvariabletemplate.yml
```

Resources

Resource is anything used by a pipeline that lives outside the pipeline.

- CI/CD pipelines that produce artifacts (Azure Pipelines, Jenkins, etc.)
- Code repositories (Azure Repos Git repos, GitHub, Bitbucket Cloud)
- Container image registries (Azure Container Registry, Docker Hub, etc.)
- Package feeds (GitHub packages)

Templates

Define reusable content, logic, and parameters.

- Variable template
- Logic template

azure-pipeline-single-job Blueprint Extract

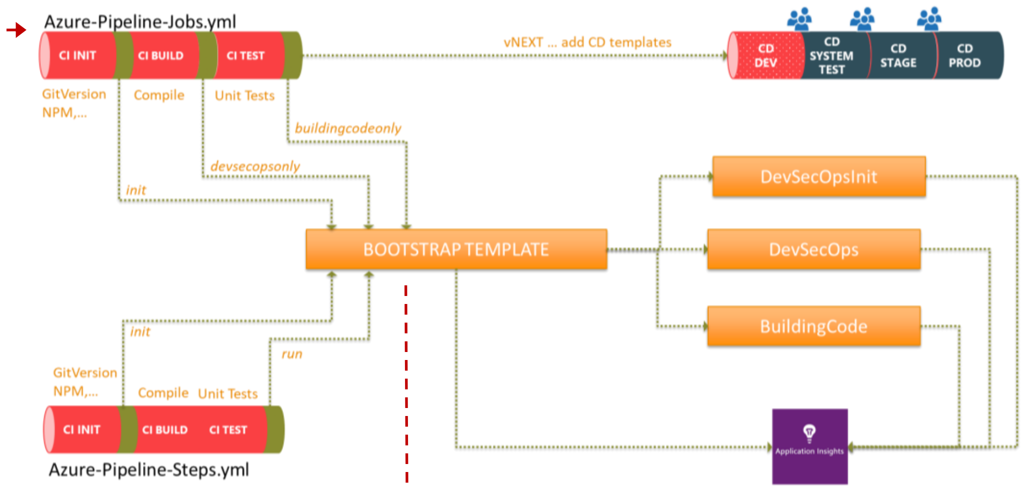
```yaml
# ---------------------------------------------------------------------------
# PIPELINE INITIALIZATION
# ---------------------------------------------------------------------------
# batch   - Batch if pipeline is running.
# '*'     - Trigger on all branches. Artifact filters will be used in CD.
# paths   - Exclude pipeline/ folder which contains pipeline yml files.
# ---------------------------------------------------------------------------
parameters:
- name: image
  displayName: 'Pool Image'
  type: string
  default: windows-latest
  values:
  - windows-latest
  - ubuntu-latest
  - macOS-latest

trigger:
  batch: true
  branches:
    include:
    - '*'
  paths:
    exclude:
    - pipeline

# Semantic version as per Azure DevOps Naming Conventions.
name:
$(portfolioName)_$(productName)_$(GITVERSION_MAJORMINORPATCH)_$(date:yyyyMMdd).$(d
ate:HHmmss).$(Build.SourceBranchName)

# Configure the default agent pool and image to use for your pipeline
pool:
  vmImage:              $(image)

# VARIABLES
variables:
  BuildConfiguration:   Release
  BuildPlatform:        'Any CPU'
  templateVersion:      1.0.7
  portfolioName:        'TODO REPLACE WITH PORTFOLIO NAME'
  productName:          'TODO REPLACE WITH PRODUCT NAME'
  productGuid:          'TODO REPLACE WITH A NEW GUID WITHOUT BRACKETS'

#Repository resources
resources:
  repositories:
    # ========================================================================
    # SKULL & CROSS-BONES - DO NOT COMMENT OUT, OR REMOVE THIS REPOSITORY
    - repository: CDTemplates
      type: git
      name: 'Common-Engineering-System/AzureDevOps.Automation.Pipeline.Templates'
    # ========================================================================

# ---------------------------------------------------------------------------
# START OF BUILD and TEST STAGE
# - GitVersion task looks at your Git history and works out the semantic
#   version of the commit being built.
# ---------------------------------------------------------------------------
stages:
- stage: ContinuousIntegration
  displayName: Continuous Integration
  jobs:
  - job: ContinuousIntegration
    steps:
    - task: GitVersion@5
      inputs:
        runtime: core
        updateAssemblyInfo: false

# ---------------------------------------------------------------------------
# PREREQUISITES
# - Run jobs/steps that have to run before the build here, for example NuGet
# ---------------------------------------------------------------------------
#   TODO Insert your scripts, steps, and tasks here and remove this comment

# ===========================================================================
# BOOTSTRAP VALIDATION, STAGE: CI_BOOTSTRAP_INIT
# SKULL & CROSS-BONES - DO NOT COMMENT OUT, OR REMOVE THIS SECTION
# ===========================================================================
    - template: Templates/Bootstrap.yml@CDTemplates
      parameters:
        bootstrapMode:    'init'
        applicationType:  'TODO REPLACE WITH APP TYPE' # dotnet, angular
        applicationGuid:  $(productGuid)
        portfolioName:    $(portfolioName)
        productName:      $(productName)
        sourcesDirectory: $(Build.SourcesDirectory)
# ===========================================================================

# ---------------------------------------------------------------------------
# CONTINUOUS INTEGRATION BUILD
# - Run jobs/steps/tasks to build your solution here.
# - Move initialisations (NPM, NuGet,...) to PREREQUISITES section
# ---------------------------------------------------------------------------
#  TODO Insert your scripts, steps, and tasks here and remove these comments
#    - task: MSBuild@1
#      inputs:
#        solution: '**/*.sln'

# ---------------------------------------------------------------------------
# CONTINUOUS INTEGRATION TEST
# - Run jobs/steps/tasks to test your solution here
# ---------------------------------------------------------------------------
#   TODO Insert your scripts, steps, and tasks here and remove this comment

# ---------------------------------------------------------------------------
# PUBLISH
# - Publish the build and test artifacts
# ---------------------------------------------------------------------------
#   TODO Insert build and test artifact publication tasks

# ===========================================================================
# BOOTSTRAP VALIDATION, STAGE: CI_BOOTSTRAP
# SKULL & CROSS-BONES - DO NOT COMMENT OUT, OR REMOVE THIS SECTION
# ===========================================================================
    - template: Templates/Bootstrap.yml@CDTemplates
      parameters:
        bootstrapMode:    'run'
        applicationType:  'TODO REPLACE WITH APP TYPE' # dotnet, angular
        applicationGuid:  $(productGuid)
        portfolioName:    $(portfolioName)
        productName:      $(productName)
        sourcesDirectory: $(Build.SourcesDirectory)
# ===========================================================================
```

Inject at **queue** time

Inject at **queue** time

Conditional template injection

Only run DevSecOps and Building Code validations for pull request validation builds, or if forceCheck is 'true'



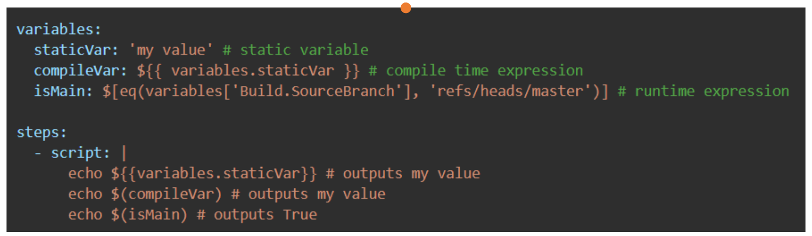
Bootstrap.xml Extract

```yaml
# ---------------------------------------------------------------------------
# WorkSafeBC Multi-Stage YAML-Based CI Pipeline Design Practice
# ---------------------------------------------------------------------------
# MODES
#   init                 Runs initialisation steps
#   run                  Runs validation steps
#   runDevSecOpsOnly     Runs only DevSecOps validation steps
#   runBuildingCodeOnly  Runs only Building Code validation steps
# ---------------------------------------------------------------------------
parameters:
  bootstrapMode:    'invalid'
  applicationType:  'unknown'
  applicationGuid:  'unknown'
  portfolioName:    'unknown'
  productName:      'unknown'
  sourcesDirectory: '$(Build.SourcesDirectory)'
  verbose:          'false'
  forceCheck:       'false'
  forceToolbox:     'false'
  previews:         'false'

steps:

# ===========================================================================
# BOOTSTRAP TOOLBOX
# ===========================================================================
- ${{ if or(eq( parameters.bootstrapMode, 'init' ), eq( parameters.forceToolbox, 'true' )) }}:
  - task: UniversalPackages@0
    displayName: Bootstrap download toolbox
    inputs:
      command: 'download'
      downloadDirectory: '$(System.DefaultWorkingDirectory)/Toolbox'
      feedsToUse: 'internal'
      vstsFeed: '9e070486-893d-44c7-8840-2e82011555b6/60506877-90cb-47b3-aa4c-cdd1b05db462'
      vstsFeedPackage: '84827d02-43dd-4518-8292-b17e681cb764'
      vstsPackageVersion: '*'

# ===========================================================================
# Alert if we supress
# ===========================================================================
- ${{ if and(ne(variables['Build.SourceBranchName'], 'merge'), eq( parameters.forceCheck, 'false')) }}:
  - script: echo Bootstrap Validations are supressed!
    displayName: Bootstrap Validation Supression Alert

# ===========================================================================
# BOOTSTRAP INIITIALISATION STAGE
# ===========================================================================
- ${{ if and(eq( parameters.bootstrapMode, 'init' ),
          or(eq(variables['Build.SourceBranchName'],'merge'), eq( parameters.forceCheck, 'true'))) }}:
  - template: DevSecOps/SecurityScansInit.yml
    parameters:
      applicationType:       ${{parameters.applicationType}}
      portfolioName:         ${{parameters.portfolioName}}
      productName:           ${{parameters.productName}}
      sonarQubeCliSources:   ${{parameters.sourcesDirectory}}
      sonarQubeService:      'SonarQube Production'
      sonarQubeProjectKey:   SQv2.${{parameters.portfolioName}}.${{parameters.productName}}
      sonarQubeProjectName:  ${{parameters.portfolioName}}.${{parameters.productName}}
      verbose:               ${{parameters.verbose}}
      previews:              ${{parameters.previews}}

…
```

Expressions

Expressions can be evaluated at compile time or at run time.

- Functions
  - and, eq, ge, gt, in, le, lt, ne, not, or, xor
  - coalesce, contains,, containsValue, counter, endsWith, format, join, length, lower, notIn, replace, startsWith, upper
- Job Status
  - Always, canceled, failed, succeeded, succeededOrFailed



```yaml
variables:
  staticVar: 'my value' # static variable
  compileVar: ${{ variables.staticVar }} # compile time expression
  isMain: $[eq(variables['Build.SourceBranch'], 'refs/heads/master')] # runtime expression

steps:
  - script: |
      echo ${{variables.staticVar}} # outputs my value
      echo $(compileVar) # outputs my value
      echo $(isMain) # outputs True
```