Microsoft

AZ-400.2 Module 01: Implementing Continuous Integration in an Azure DevOps Pipeline

Lesson 01: Continuous Integration Overview

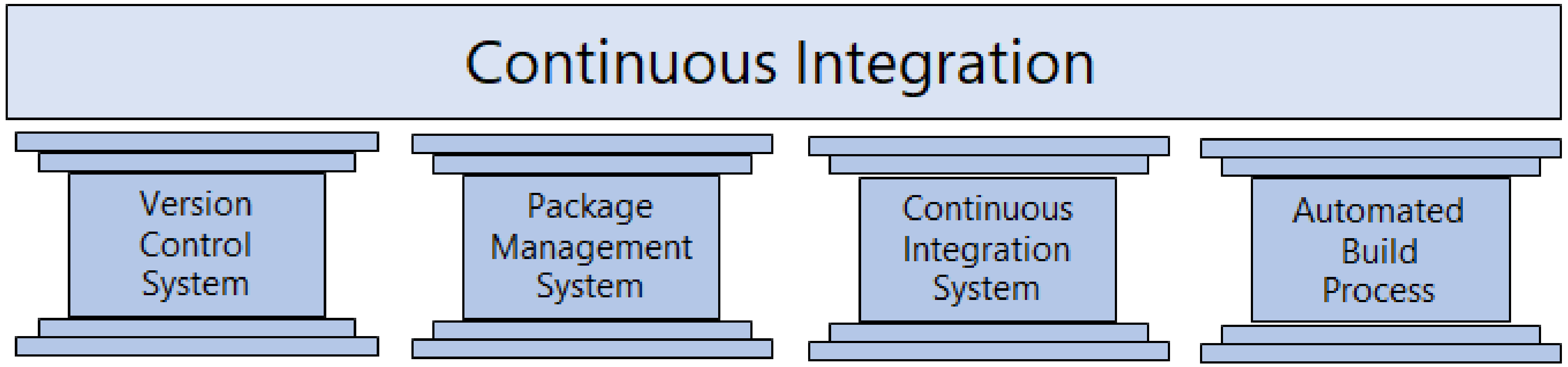
Lesson 1 Overview

- Introduction to Continuous Integration
- The Four Pillars of Continuous Integration
- Benefits of Continuous Integration
- Continuous Integration Implementation Challenges
- Implementing Continuous Integration in Azure DevOps
- Using Variables to Avoid Hard-coded Values
- Build Number Formatting and Build Status
- Build Authorizations, Timeouts, and Badges
- Configuring Build Retention
- Lab - Enabling Continuous Integration with Azure Pipelines

Video: Introduction to Continuous Integration

- Continuous Integration (CI) is the process of automating the build and testing of code.
- CI encourages developers to share their code and unit tests by merging their changes into the shared version control repository.
- When a change is detected it triggers an automated build system. The code is built using a build definition. Developers respond to any issues or bugs.
- CI keeps the master branch clean ensuring bugs are caught earlier in the development cycle, which makes them less expensive to fix.

The Four Pillars of Continuous Integration



- A Version Control System manages changes to your source code over time.
- A Package Management System is used to install, uninstall and manage software packages.
- A Continuous Integration System merges all developer working copies to a shared mainline several times a day.
- An Automated Build Process creates a software build including compiling, packaging, and running automated tests.

Benefits of Continuous Integration

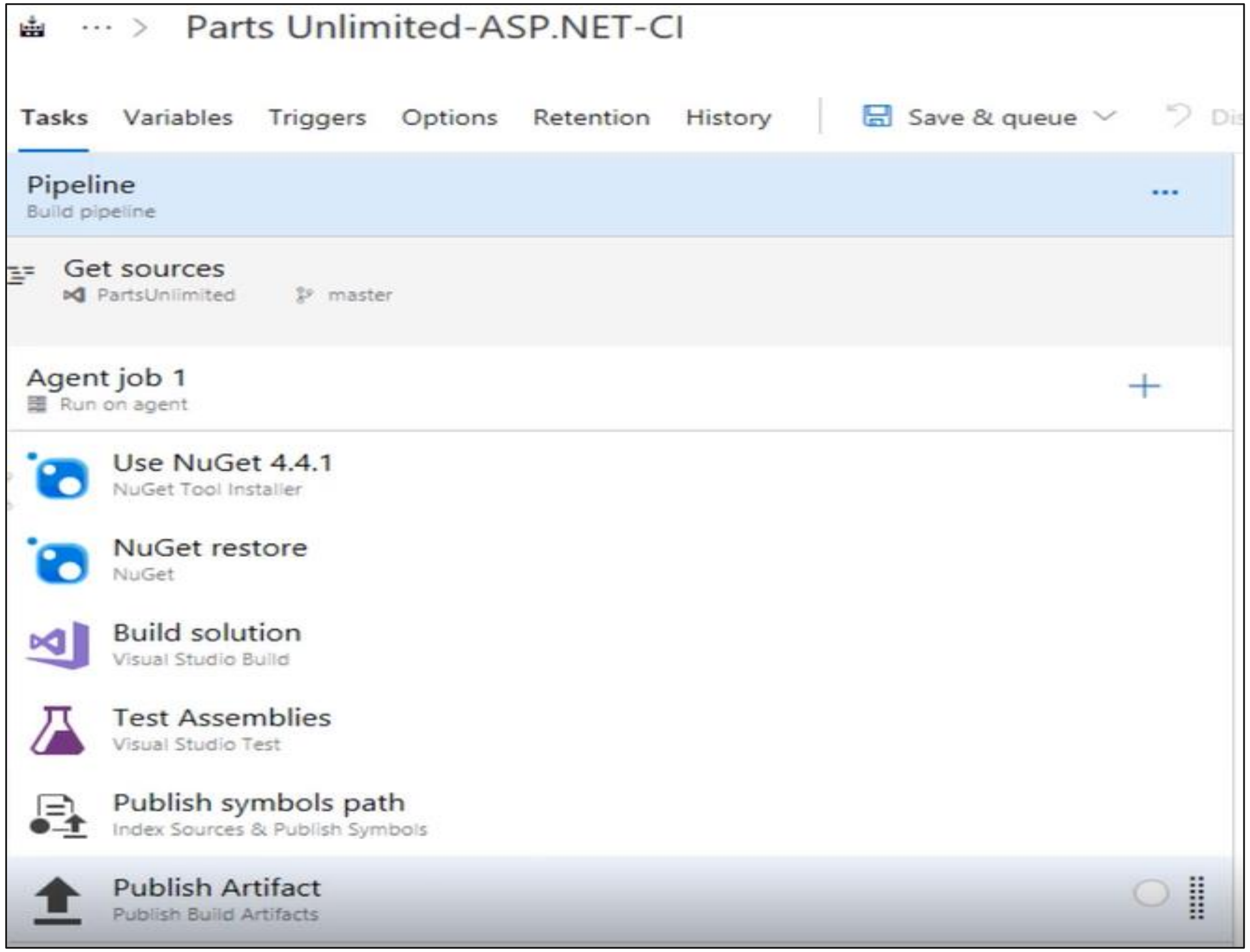
- Improving code quality based on rapid feedback
- Triggering automated testing for every code change
- Reducing build times for rapid feedback and early detection of problems (risk reduction)
- Better management of technical debt and code analysis
- Reducing long, difficult, and bug-inducing merges
- Increasing confidence in codebase health long before production deployment
- Rapid feedback to the developer

Discussion: Continuous Integration

- Have you tried to implement continuous integration in your organization?

- If you where successful, what lessons did you learn?
- If you were not successful, what were the challenges?

Demonstration: Implementating Continuous Integration in Azure DevOps



Pipeline
Build pipeline

Get sources
PartsUnlimited master

Agent job 1
Run on agent

Use NuGet 4.4.1
NuGet Tool Installer

NuGet restore
NuGet

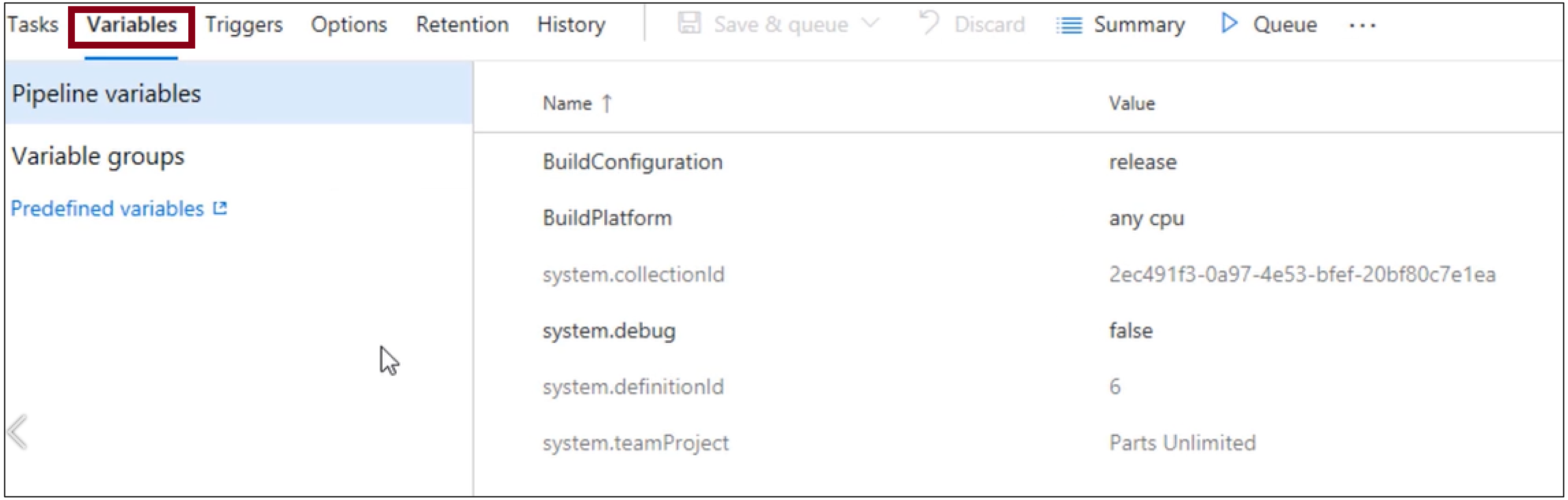
Build solution
Visual Studio Build

Test Assemblies
Visual Studio Test

Publish symbols path
Index Sources & Publish Symbols

Publish Artifact
Publish Build Artifacts

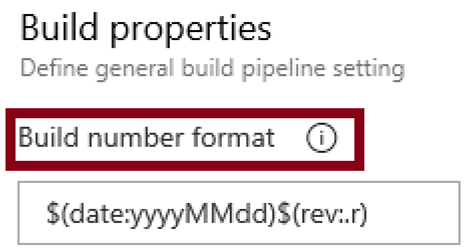
Demonstration: Using Variables to Avoid Hard-coded Values



Name ↑	Value
BuildConfiguration	release
BuildPlatform	any cpu
system.collectionId	2ec491f3-0a97-4e53-bfef-20bf80c7e1ea
system.debug	false
system.definitionId	6
system.teamProject	Parts Unlimited

Build Number Formatting and Build Status

- Build number formatting



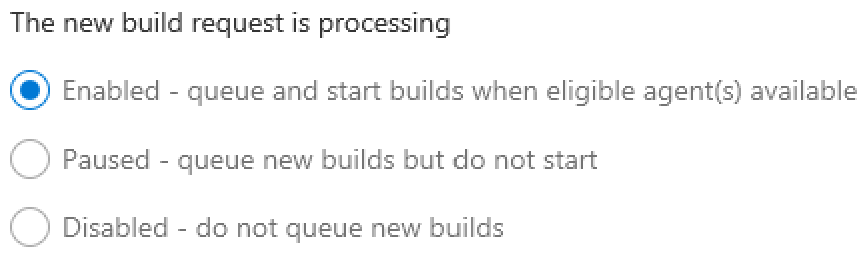
Build properties

Define general build pipeline setting

Build number format

```
$(date:yyyyMMdd)$(rev:.r)
```

- Build status (enabled, paused, disabled)



The new build request is processing

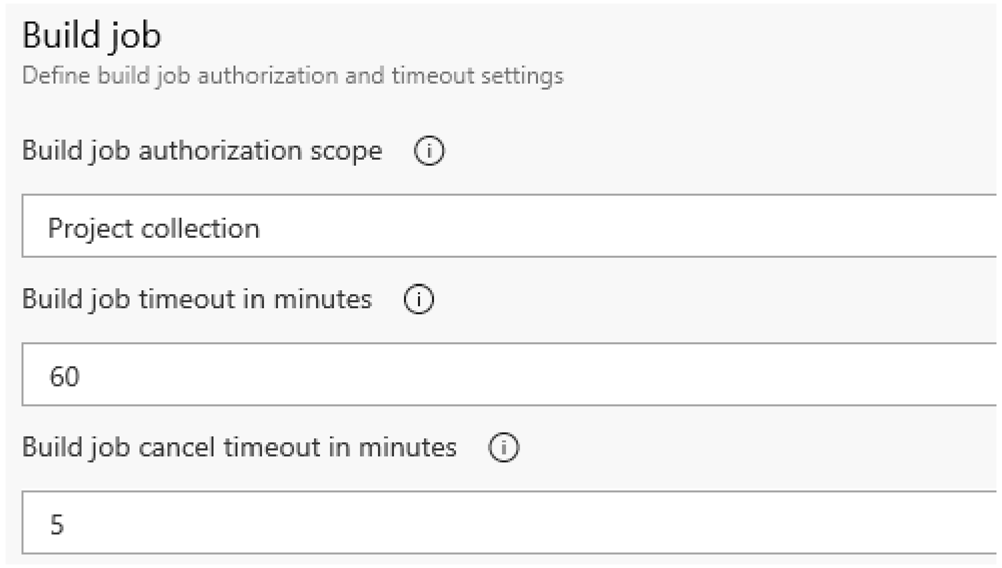
[✓ Enabled - queue and start builds when eligible agent(s) available]

Paused - queue new builds but do not start

Disabled - do not queue new builds

Authorization and Timeouts, and Badges

- Authorization and Timeouts (scope, job timeout, cancel job timeout)



Build job

Define build job authorization and timeout settings

Build job authorization scope

Project collection

Build job timeout in minutes

60

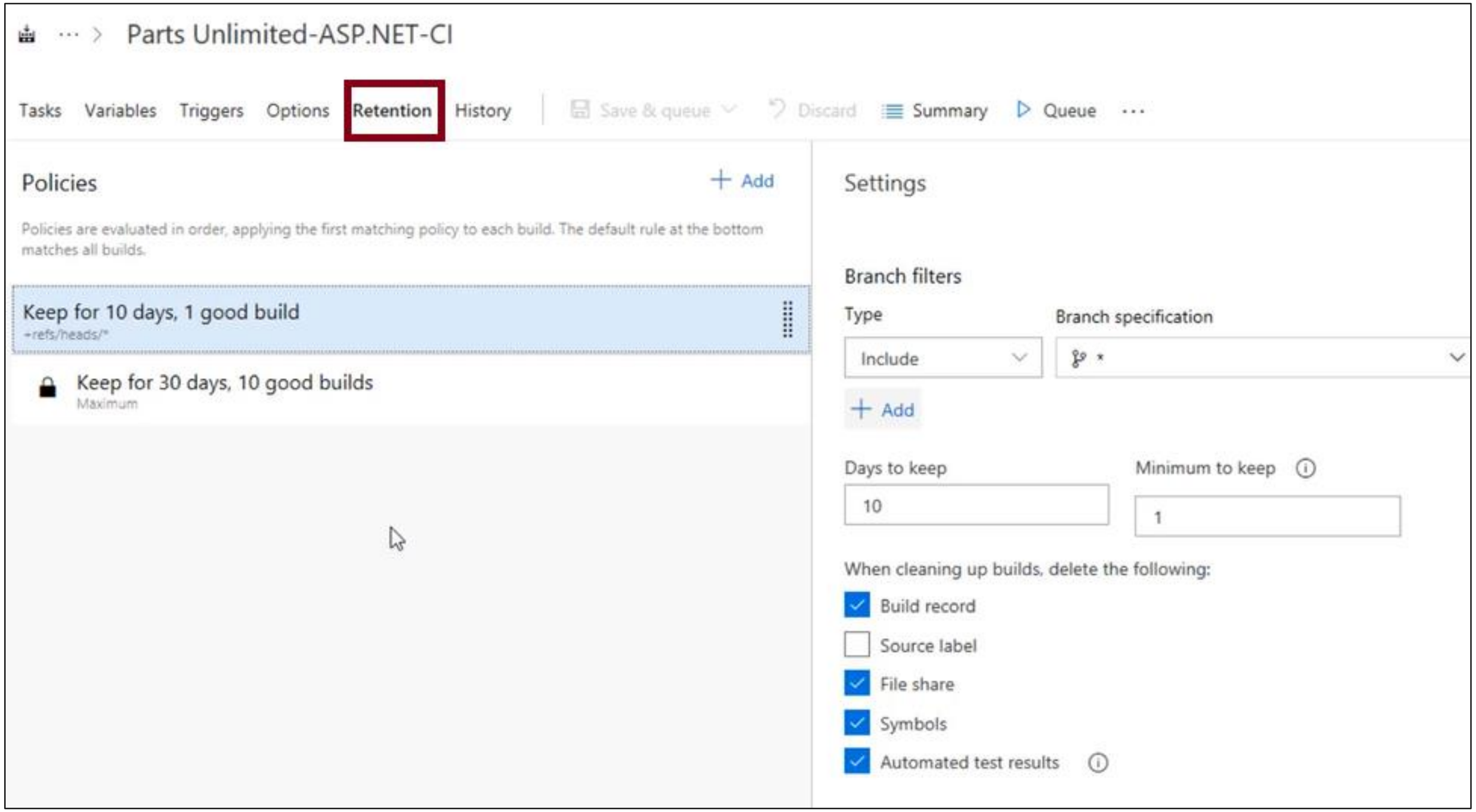
Build job cancel timeout in minutes

5

- Badges

Azure Pipelines
succeeded

Video: Configuring Build Retention



Policies

Policies are evaluated in order, applying the first matching policy to each build. The default rule at the bottom matches all builds.

Keep for 10 days, 1 good build
+refs/heads/*

Keep for 30 days, 10 good builds
Maximum

Settings

Branch filters

Type	Branch specification
Include	*

Days to keep: 10

Minimum to keep: 1

When cleaning up builds, delete the following:

- ✓ Build record
- ☐ Source label
- ✓ File share
- ✓ Symbols
- ✓ Automated test results

Lab: Enabling Continuous Integration with Azure Pipelines

In this hands-on lab, you will learn how to configure continuous integration with Azure Pipelines. You will perform the following tasks:

- Creating a basic build pipeline from a template
- Tracking and reviewing a build
- Invoking a continuous integration build

✓ Note that you must have already completed the prerequisite labs in the Welcome section.

Lesson 02: Implementing a Build Strategy

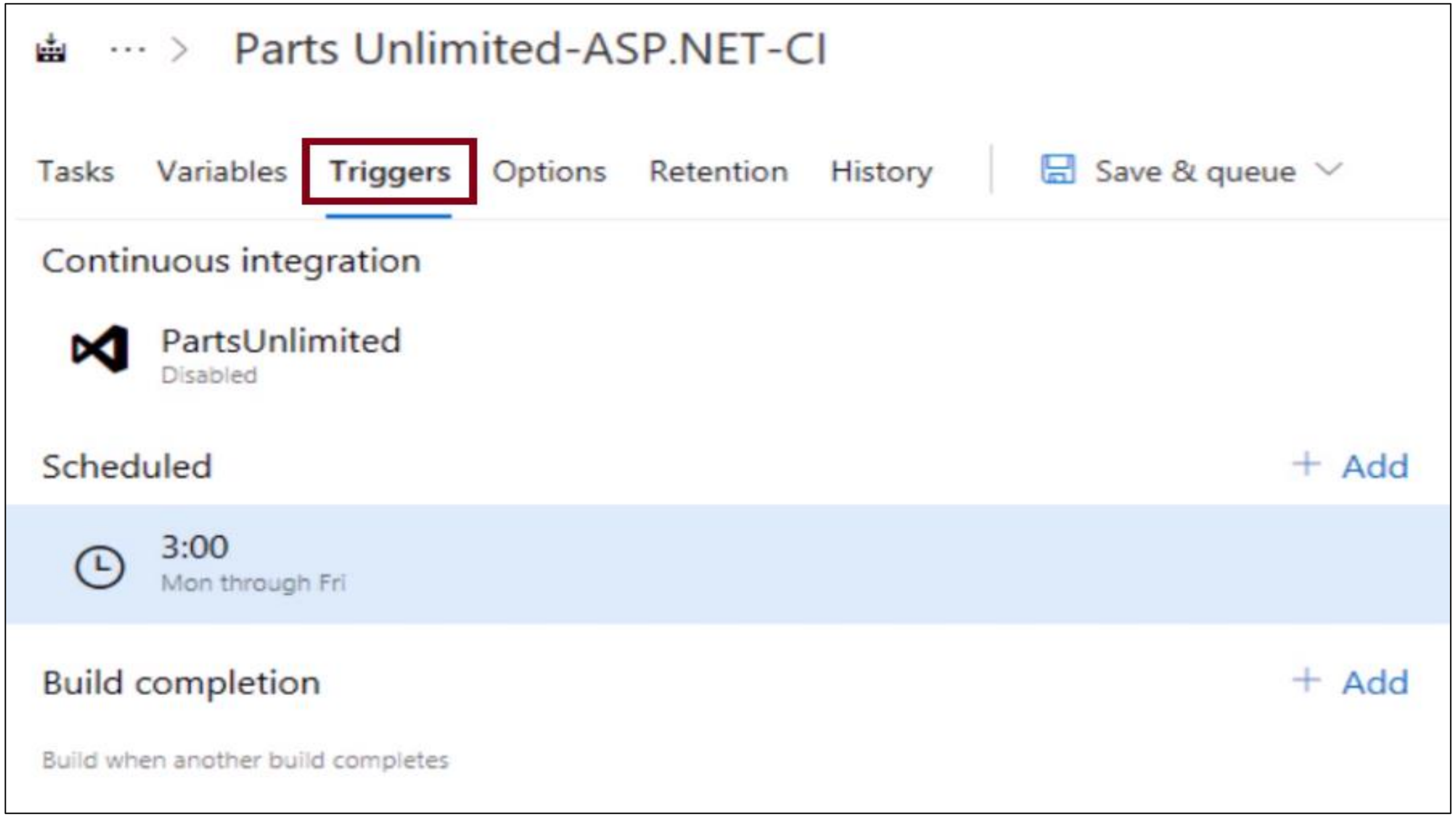
Lesson 2 Overview

- Automated Build Workflows
- Implementing Build Triggers
- Working with Hosted Agents
- Implementing a Hybrid Build Process
- Configuring Agent Demands
- Implementing Multi-Agent Builds
- Build-Related Tooling
- Creating a Jenkins Build Job and Triggering CI

Video: Automated Build Workflows

- Azure DevOps can automate a custom workflow that's as large and complex as you need
- Agile teams normally require more than one type of build
- Builds are typically triggered automatically when code is committed
- Builds can also be scheduled – such as a daily build

Video: Implementing Build Triggers



Continuous integration

PartsUnlimited
Disabled

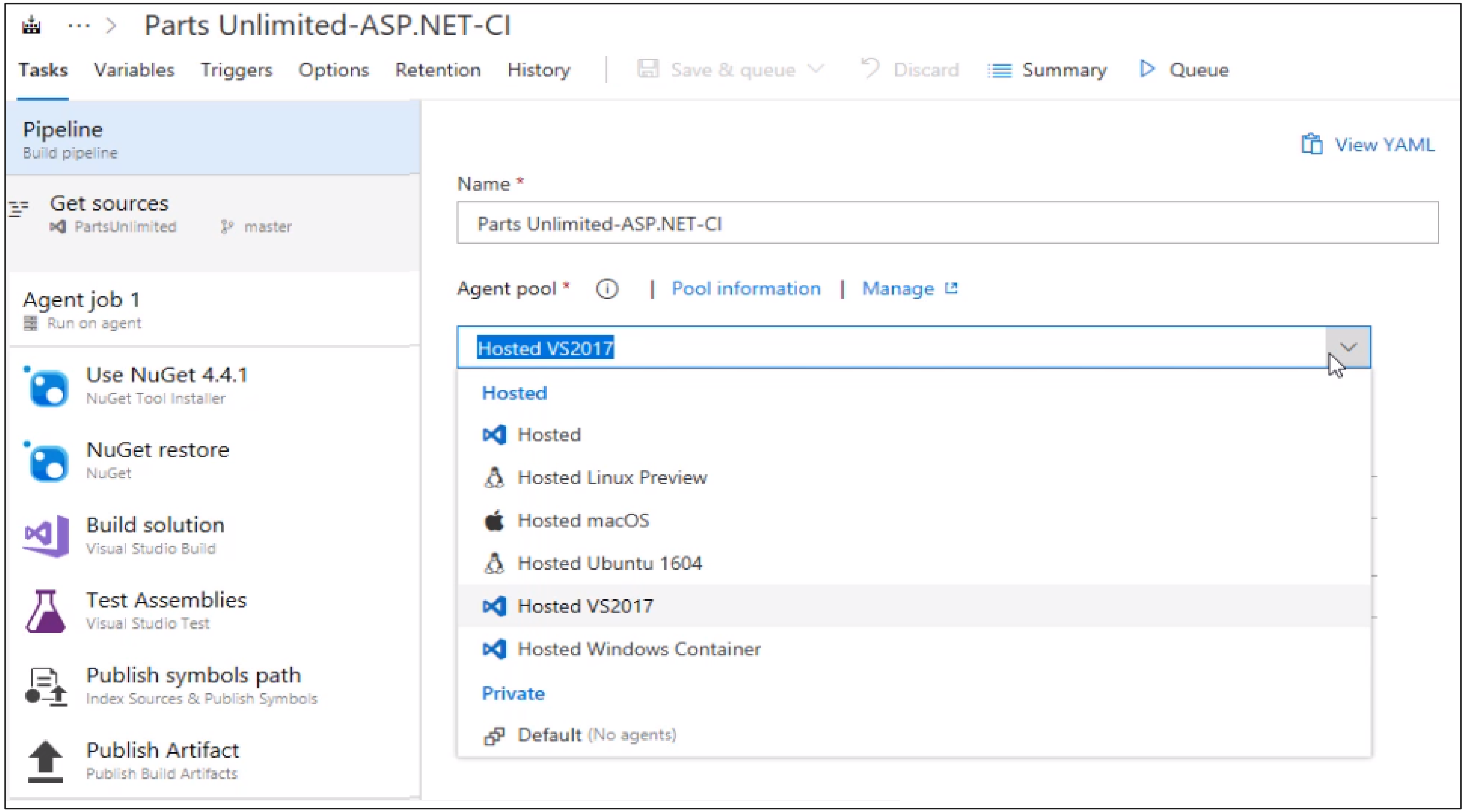
Scheduled

3:00
Mon through Fri

Build completion

Build when another build completes

Video: Working with Hosted Agents



Name *

Parts Unlimited-ASP.NET-CI

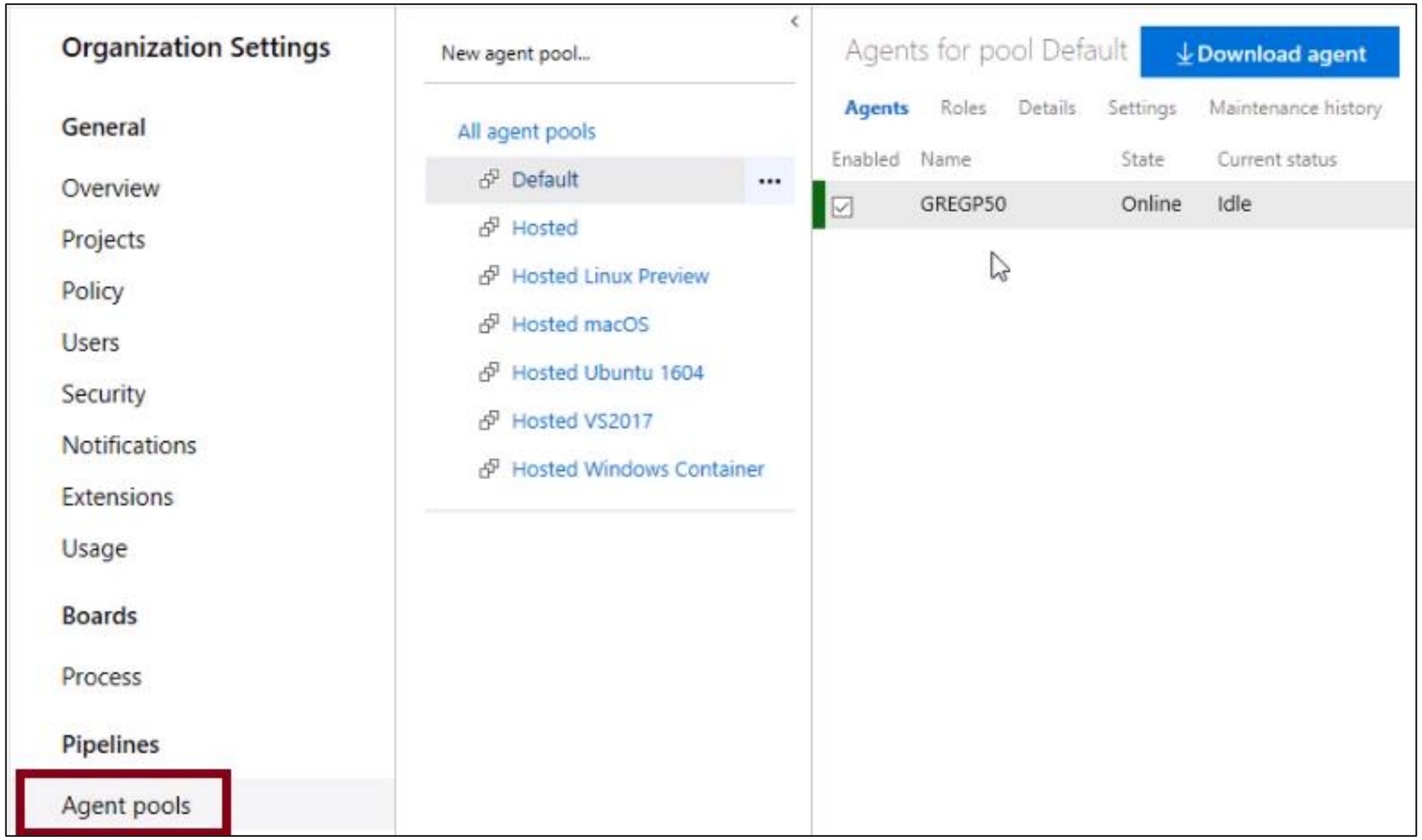
Agent pool *

Hosted VS2017

Hosted
- Hosted
- Hosted Linux Preview
- Hosted macOS
- Hosted Ubuntu 1604
- Hosted VS2017
- Hosted Windows Container

Private
- Default (No agents)

Video: Implementing a Hybrid Build Process



Organization Settings

General

- Overview
- Projects
- Policy
- Users
- Security
- Notifications
- Extensions
- Usage

Boards

- Process

Pipelines

- Agent pools

New agent pool...

All agent pools

- Default
- Hosted
- Hosted Linux Preview
- Hosted macOS
- Hosted Ubuntu 1604
- Hosted VS2017
- Hosted Windows Container

Agents for pool Default

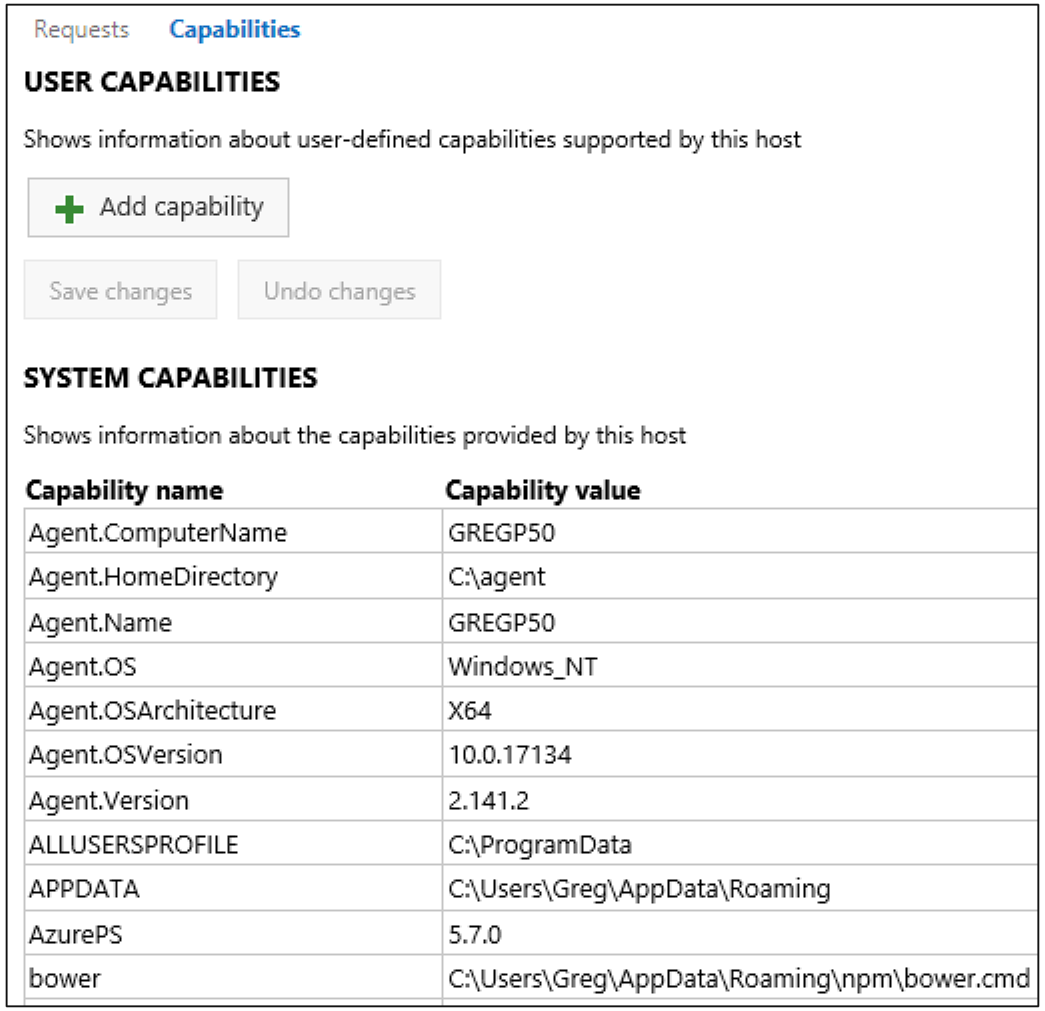
Download agent

Agents | Roles | Details | Settings | Maintenance history

Enabled	Name	State	Current status
✓	GREGP50	Online	Idle

Configuring Agent Demands

- User Capabilities
- System Capabilities
- Agents can have different authorization and timeout settings



USER CAPABILITIES

Shows information about user-defined capabilities supported by this host

SYSTEM CAPABILITIES

Shows information about the capabilities provided by this host

Capability name	Capability value
Agent.ComputerName	GREGP50
Agent.HomeDirectory	C:\agent
Agent.Name	GREGP50
Agent.OS	Windows_NT
Agent.OSArchitecture	X64
Agent.OSVersion	10.0.17134
Agent.Version	2.141.2
ALLUSERSPROFILE	C:\ProgramData
APPDATA	C:\Users\Greg\AppData\Roaming
AzurePS	5.7.0
bower	C:\Users\Greg\AppData\Roaming\npm\bower.cmd

Implementing Multi-Agent Builds

Adding multiple jobs to a pipeline lets you:

- Break your pipeline into sections that need different agent pools, or self-hosted agents
- Publish artifacts in one job and consume them in one or more subsequent jobs
- Build faster by running multiple jobs in parallel
- Enable conditional execution of tasks

✓ You can configure the number of parallel jobs

Discussion: Build-Related Tooling

Azure DevOps can be integrated with a wide range of existing tooling that is used for builds or associated with builds.

- Which build-related tools do you currently use?
- What do you like or don't like about the tools?

Lab: Creating a Jenkins Build Job and Triggering CI

In this hands-on lab, you will learn how to create a build job in Jenkins and to enable continuous integration. You will learn how to:

- Provision Jenkins on Azure VM using the Jenkins template available on the Azure Marketplace
- Configure Jenkins to work with Maven and Azure DevOps
- Create a build job in Jenkins
- Configure Azure Pipeline to integrate with Jenkins
- Configure a CD pipeline in Azure Pipelines to deploy the build artifacts

✓ In this lab, you will try two approaches to triggering continuous integration

Module 1: Review Questions

1. What are the four pillars of continuous integration?
2. The build numbers that Azure DevOps generates for you are currently integers. If you would prefer the build to include the date, how would you change this?
3. You want to take your build server offline to make a configuration change. You want it to complete any build that it is currently processing, but you want to queue any new build requests. What should you do?
4. You want to set a maximum time that builds should not run for more than 5 minutes. What configuration change should you make?
5. The hands-on lab for Creating a Jenkins Build Job and Triggering CI described two methods that could be used to enable continuous integration for Jenkins. What were the two methods?