Microsoft
tech·days
Kistamässan Stockholm
24-25 oktober 2018

# Continuous Delivery

# Azure Pipelines

Mathias Olausson, CTO, Solidify
mathias.olausson@solidify.se
@molausson

# Agenda

- What is Continuous Delivery?
- Setting up a delivery pipeline using Azure Pipelines
- Release Management
- Deployment

# What do we want?

More innovation

Faster

At lower cost

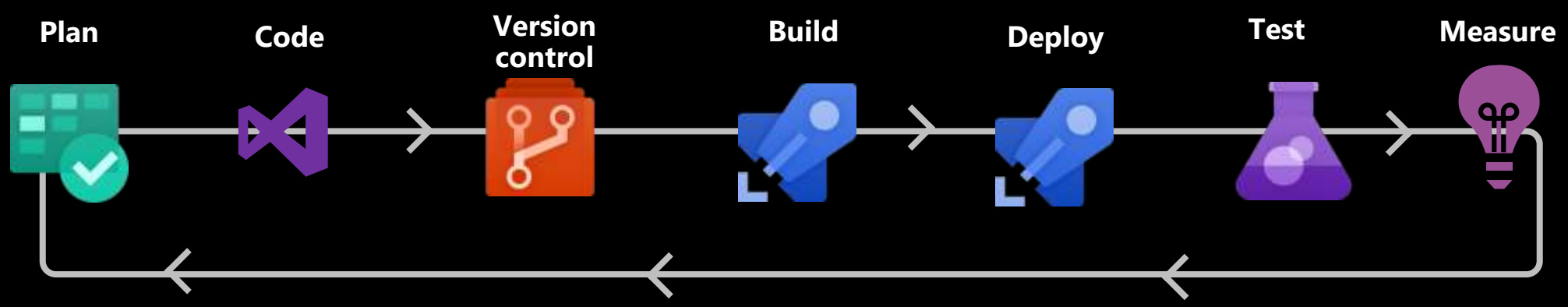
# How?

*"Continuous Delivery is a software development discipline where you build software in such a way that the software can be released to production at any time"*

Deliver what the customer wants!

# How?

# Continuous Delivery



# Azure DevOps

## Azure Boards

Deliver value to your users faster using proven agile tools to plan, track, and discuss work across your teams.

## Azure Pipelines

Build, test, and deploy with CI/CD that works with any language, platform, and cloud. Connect to GitHub or any other Git provider and deploy continuously.

## Azure Repos

Get unlimited, cloud-hosted private Git repos and collaborate to build better code with pull requests and advanced file management.

## Azure Test Plans

Test and ship with confidence using manual and exploratory testing tools.

## Azure Artifacts

Create, host, and share packages with your team, and add artifacts to your CI/CD pipelines with a single click.

# Azure Pipelines

Cloud-hosted pipelines for Linux, Windows and macOS, with unlimited minutes for open source

## Any language, any platform, any cloud

Build, test, and deploy Node.js, Python, Java, PHP, Ruby, C/C++, .NET, Android, and iOS apps. Run in parallel on Linux, macOS, and Windows. Deploy to Azure, AWS, GCP or on-premises

## Extensible

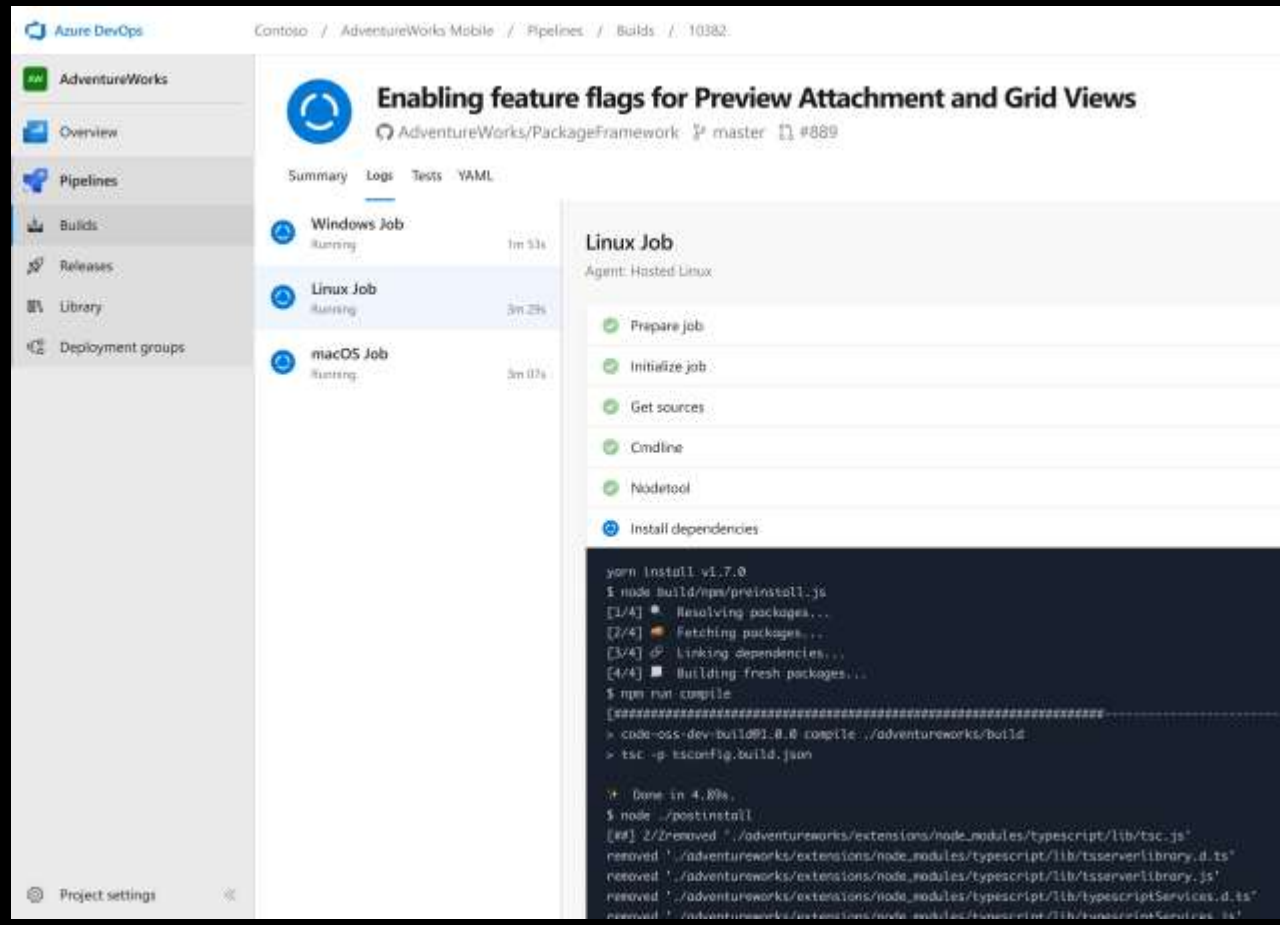
Explore and implement a wide range of community-built build, test, and deployment tasks, along with hundreds of extensions from Slack to SonarCloud. Support for YAML, reporting and more

## Containers and Kubernetes

Easily build and push images to container registries like Docker Hub and Azure Container Registry. Deploy containers to individual hosts or Kubernetes.

## Best-in-class for open source

Ensure fast continuous integration/continuous delivery (CI/CD) pipelines for every open source project. Get unlimited build minutes for all open source projects with up to 10 free parallel jobs across Linux, macOS and Windows



# C:\> Building the right things

From code to user

Is this what you wanted?

What worked?

- Fast delivery
- Version control
- CI, build once
- CD pipeline (process)
- Anyone can deploy anytime

What's missing?

- No validation
- Only one stage
- No configuration
- No managed environments
- No governance

# Building things right – CD practices

- Version control strategy
- Build process
- Build promotion strategy
- Deployment stages
- Validation
- Governance
- Release notes

# Setting up the release process

# Version control strategy

- Branching model
  - Use what makes sense for your needs
    - GitFlow
    - Feature/topic branches
    - Trunk based development
- Set branch policies
  - Protect master (or other branches)
  - Common requirements
    - Require code review
    - Require work item
    - CI build must pass
- Code integration using pull-requests

# Build process

- Core steps
  - Restore packages
  - Compile
  - Run unit tests (L0-1)
  - Run code analysis
  - Package artifacts
- Azure Pipelines run on hosted or private agents
  - And on Window, Mac or Linux
- Very extensible
  - But… be aware of task implementations
- And free for OSS!
  - 10 pipelines for public Azure DevOps project

# Build promotion strategy

- Which builds to use where
  - PR builds?
  - CI builds?
  - Release builds?
- Assign a build version
  - Follow SemVer
  - Use build variables to set build version
    - $(majorVersion).$(minorVersion)$(rev:.r)
  - Or use build tasks like GitVersion

# Deployment stages

- Configure artifacts
  - Build, repo, nuget, external, …
- Define stages for your process
  - Dev, test, prod, …
- Configure deployment model
  - Fan out/in, incremental roll-out, …
- Configure triggers for workflow
- Configure deployment process

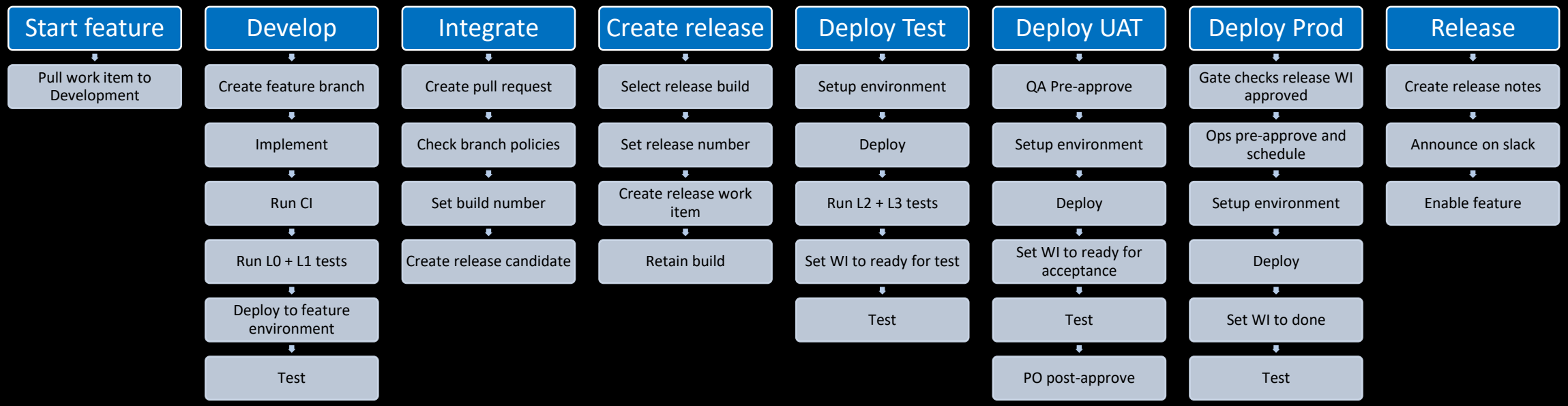
# Governance

- Only authorized staff can change deployment process
- Four eyes principle
- Provide end to end traceability

# Release notes

- Gives feedback about what has been released
- Automated release notes keeps us honest
  - And saves time

# A release process



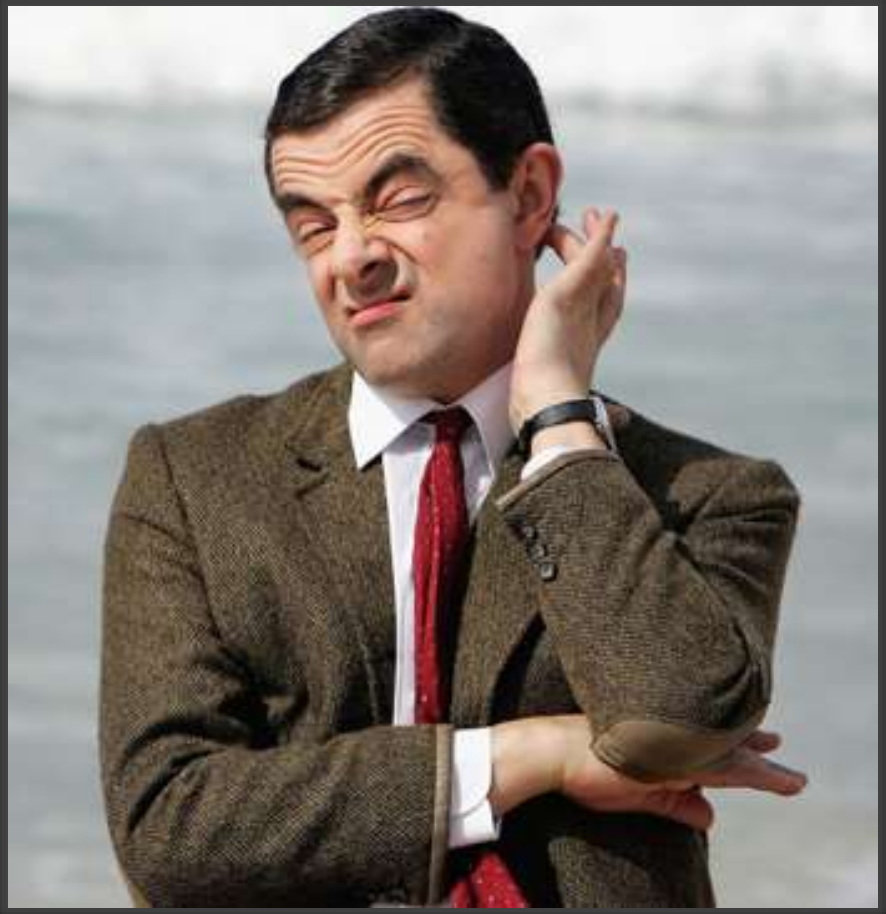
# C:\> Building things right

From work items to release notes

Is this what you wanted?

What worked?

- Version control
- CI, build once
- CD pipeline
- Anyone can deploy anytime
- Many stages
- Release management
- Governance



What's missing?

- No validation
- No managed environments

# Buildings things

- Managing (heterogeneous) environments
- Deployment strategies
- Infrastructure as code
- Validation

# Buildings things

Azure DevOps lets developers choose the tools that are right for them

Mix and match to create workflows with tools from Microsoft, open source or your favorite 3rd party tools

Target any cloud, on-prem or both and deploy to the servers you need

# Deployment strategies

- Apply DDD*
- Deal with infrastructure early
- Separate configuration from implementation

*deployment driven development

# Infrastructure as code

- Use what makes sense for your infrastructure (engineers)
  - DSC, ARM, Terraform,
    AZ CLI, Powershell, …
- Keep it close to the source
- Benefits of IOC
  - Repeatable
  - Reduce risk
  - Traceable
  - Elastic

```
Node $env:COMPUTERNAME
{
    WindowsFeature WebServerRole
    {
        Ensure = "Present"
        Name = "Web-Server"
    }

    WindowsF
    {
        Ensu
        Name
        Depe
    }
```

```
"resources": [
  {
    "name": "[parameters('databaseServer')]",
    "type": "Microsoft.Sql/servers",
    "location": "[resourceGroup().location]",
    "tags": {
      "displayName": "SqlServer"
    },
    "apiVersion": "2014-04-01-preview",
    "properties": {
      "administratorLogin": "[parameters('administratorLogin')]",
      "administratorLoginPassword": "[parameters('administratorLoginPassword')]"
    },
    "resources": [
      {
        "name": "[parameters('databaseName')]",
        "type": "databases",
        "location": "[resourceGroup().location]",
        "tags": {
          "displayName": "Database"
        },
```

# Pipelines as code

- We can now define Azure Pipelines in code!
  - Currently only for build
  - No round-trip to visual designer
  - No local runner
- And… Keep it close to the source
- Benefits of PiC
  - Maintainability
  - Reusability
  - Edit using any tool

azure-pipelines.yml

```yaml
# .NET Desktop
# Build and run tests for .NET Desktop or Windows classic desktop solutions.
# Add steps that publish symbols, save build artifacts, and more:
# https://docs.microsoft.com/azure/devops/pipelines/apps/windows/dot-net

pool:
  vmImage: 'VS2017-Win2016'

variables:
  solution: '**/*.sln'
  buildPlatform: 'Any CPU'
  buildConfiguration: 'Release'

steps:
- task: NuGetToolInstaller@0

- task: NuGetCommand@2
  inputs:
    restoreSolution: '$(solution)'

- task: VSBuild@1
  inputs:
    solution: '$(solution)'
    platform: '$(buildPlatform)'
    configuration: '$(buildConfiguration)'

- task: VSTest@2
  inputs:
    platform: '$(buildPlatform)'
    configuration: '$(buildConfiguration)'
```

# Validation

- Faster releases require automation
- Should be done as a team
- Run automated tests as early as possible
  - “Shift-left”

# C:\> Building things

Any tool, any platform, any cloud

Is this what you wanted?

# CD practices

- Version control
  - Code, infrastructure and process
- CI, build once
- CD pipeline
  - Anyone can deploy anytime
  - Multiple stages
- Governance
  - Workflows, approvals, gates
- Deployment process
- Validation
- Configuration
- Dynamic environments

Continuous Delivery

Azure Pipelines

helps us

**deliver what the customer wants**

by

**building the right things right**

# Resources

- Azure DevOps
  - https://azure.microsoft.com/en-us/services/devops/
- Azure Pipelines
  - https://azure.microsoft.com/en-us/services/devops/pipelines/
- Hands-on labs
  - https://www.azuredevopslabs.com/

# Thank you!

Please evaluate my session in the TechDays app!