Microsoft

Azure DevOps Pipeline Decorators

January 2023

Steve Keeler
Cloud Solution Architect

Agenda

- Overview
- Pre-requisites
- Authoring
- Publishing
- Installing

Pipeline Decorators

- Actions
  - Inject steps at the start of every job
  - Inject steps at the end of every job
  - Apply steps consistently across the organization
  - Ensure steps are run in *every* pipeline, without the need for pipeline authors to include them
- Scenarios
  - Run a code scan to detect secrets stored in code
  - Scan packages against a list of known vulnerabilities
  - Run a virus scan on build outputs

Pre-requisites

1. An Azure DevOps organization
2. A publisher in the Visual Studio Marketplace
3. A code editor, e.g., Visual Studio Code
4. The latest version of Node.js
5. TypeScript compiler 4.0.2 or later
6. Cross-platform CLI for Azure DevOps (tfx-cli)
7. A home directory for your project

Install Node packages

Windows Command Shell or PowerShell

```
npm install -g typescript
npm install -g tfx-cli
```

Create publisher

- Extensions are provided by a publisher
- Create a publisher if you don’t already have one:
  - Sign in at Visual Studio Marketplace Publishing Portal
  - If you aren’t already a member of an existing publisher, you’re prompted to create a publisher
  - If not prompted to create a publisher, select Publish extensions link at top of page
  - Specify an identifier for your publisher; for example: mycompany-myteam
    Use it as the publisher attribute in your extension manifest file
  - Specify a display name for your publisher
  - Review the Marketplace Publisher Agreement and select Create

Create a pipeline decorator project

Windows Command Shell

```
REM Change 'pipeline-decorators' to your project name
md pipeline-decorators
cd pipeline-decorators
echo node_modules > .gitignore
npm init --yes
npm install vss-web-extension-sdk --save
```

Create a pipeline decorator project

PowerShell

```
# Change 'pipeline-decorators' to your project name
New-Item -ItemType Directory -Path pipeline-decorators
Set-Location -Path pipeline-decorators
Set-Content -Path .gitignore -Value "node_modules"
npm init --yes
npm install vss-web-extension-sdk --save
```

Sample code

- The sample implements the following pipeline decorator targets:
  - Post-checkout task
  - Pre-job task
  - Post-job task
- The following slides show sample source code for:
  - Extension manifest file
  - Decorator definition files
- These files (and more) are also available in GitHub repo:
  - https://github.com/devopsincanada/pipeline-decorators

Create manifest file

- Create a manifest file named `vss-extension.json`, based on the sample shown in the next slide
- Update these attributes with values specific to your extension:
  - id
  - version
  - name
  - publisher
  - description
  - contributions
  - files

vss-extension.json

```json
{
  "manifestVersion": 1,
  "id": "pipeline-decorators",
  "version": "0.1.0",
  "name": "Pipeline Decorators",
  "publisher": "steve-keeler-microsoft",
  "targets": [
    {
      "id": "Microsoft.VisualStudio.Services"
    }
  ],
  "description": "Demonstrate pipeline decorators",
  "categories": [
    "Azure Pipelines"
  ],
  "contributions": [
    {
      "id": "pipeline-decorators-pre-job-task",
      "type": "ms.azure-pipelines.pipeline-decorator",
      "targets": [
        "ms.azure-pipelines-agent-job.pre-job-tasks"
      ],
      "properties": {
      "template": "pre-job-task.yml"
      }
    },
    {
      "id": "pipeline-decorators-post-job-task",
      "type": "ms.azure-pipelines.pipeline-decorator",
      "targets": [
        "ms.azure-pipelines-agent-job.post-job-tasks"
      ],
      "properties": {
      "template": "post-job-task.yml"
      }
    },
    {
      "id": "pipeline-decorators-post-checkout-task",
      "type": "ms.azure-pipelines.pipeline-decorator",
      "targets": [
        "ms.azure-pipelines-agent-job.post-checkout-tasks"
      ],
      "properties": {
      "template": "post-checkout-task.yml"
      }
    }
  ],
  "files": [
    {
      "path": "pre-job-task.yml",
      "addressable": true,
      "contentType": "text/plain"
    },
    {
      "path": "post-job-task.yml",
      "addressable": true,
      "contentType": "text/plain"
    },
    {
      "path": "post-checkout-task.yml",
      "addressable": true,
      "contentType": "text/plain"
    }
  ]
}
```

Targets

Target	Description
`ms.azure-pipelines-agent-job.pre-job-tasks`	Run before other tasks in a classic build or YAML pipeline. Due to differences in how source code checkout happens, this target runs before checkout in a YAML pipeline but after checkout in a classic build pipeline.
`ms.azure-pipelines-agent-job.post-checkout-tasks`	Run after the last checkout task in a classic build or YAML pipeline.
`ms.azure-pipelines-agent-job.post-job-tasks`	Run after other tasks in a classic build or YAML pipeline.
`ms.azure-pipelines-agent-job.pre-task-tasks`	Run before specified task in a classic build or YAML pipeline.
`ms.azure-pipelines-agent-job.post-task-tasks`	Run after specified task in a classic build or YAML pipeline.

https://learn.microsoft.com/azure/devops/extend/develop/add-pipeline-decorator?view=azure-devops#targets

Create decorator file

Create a file named `post-checkout-task.yml` based on the sample →

Use any pre-existing Azure Pipeline tasks

Optionally, create your own custom tasks (extensions) and use them in your decorator

post-checkout-task.yml

```yaml
steps:

- task: CmdLine@2
  condition: not(eq(variables.skipPostCheckoutTask,'true'))
  displayName: 'Post-Checkout Task'
  inputs:
  script: |
    echo 'Pipeline decorator post-checkout task'
```

Create decorator file

Create a file named
`pre-job-task.yml`
based on the sample →

Use any pre-existing Azure Pipeline tasks

Optionally, create your own custom tasks (extensions) and use them in your decorator

pre-job-task.yml

```yaml
steps:

- task: CmdLine@2
  condition: not(eq(variables.skipPreJobTask,'true'))
  displayName: 'Pre-Job Task'
  inputs:
  script: |
    echo 'Pipeline decorator pre-job task'
```

Create decorator file

Create a file named `post-job-task.yml` based on the sample →

Use any pre-existing Azure Pipeline tasks

Optionally, create your own custom tasks (extensions) and use them in your decorator

post-job-task.yml

```yaml
steps:

- task: CmdLine@2
  condition: not(eq(variables.skipPostJobTask,'true'))
  displayName: 'Post-Job Task'
  inputs:
  script: |
    echo 'Pipeline decorator post-job task'
```

Build extension

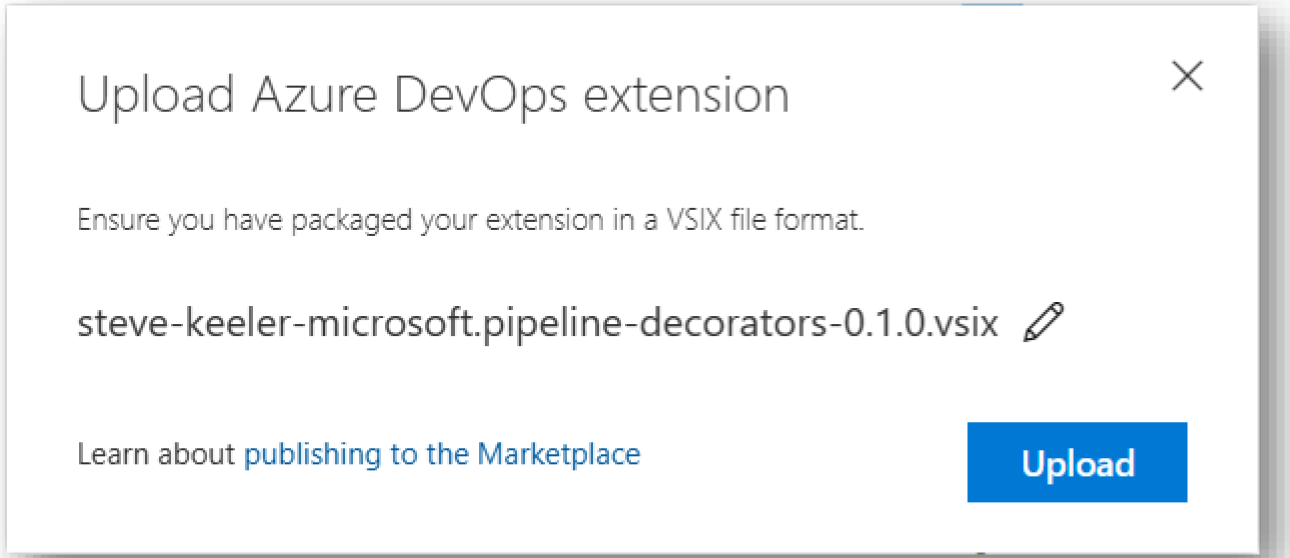
```
npx tfx-cli extension create
```

Results

```
TFS Cross Platform Command Line Interface v0.12.0
Copyright Microsoft Corporation

=== Completed operation: create extension ===
 - VSIX:steve-keeler-microsoft.pipeline-decorators-0.1.0.vsix
 - Extension ID: pipeline-decorators
 - Extension Version: 0.1.0
 - Publisher: steve-keeler-microsoft
```

Publish extension

- Navigate to https://marketplace.visualstudio.com/manage
- Select your publisher
- Select “New extension”, then “Azure DevOps”
- Upload your compiled `vsix` package


Upload Azure DevOps extension

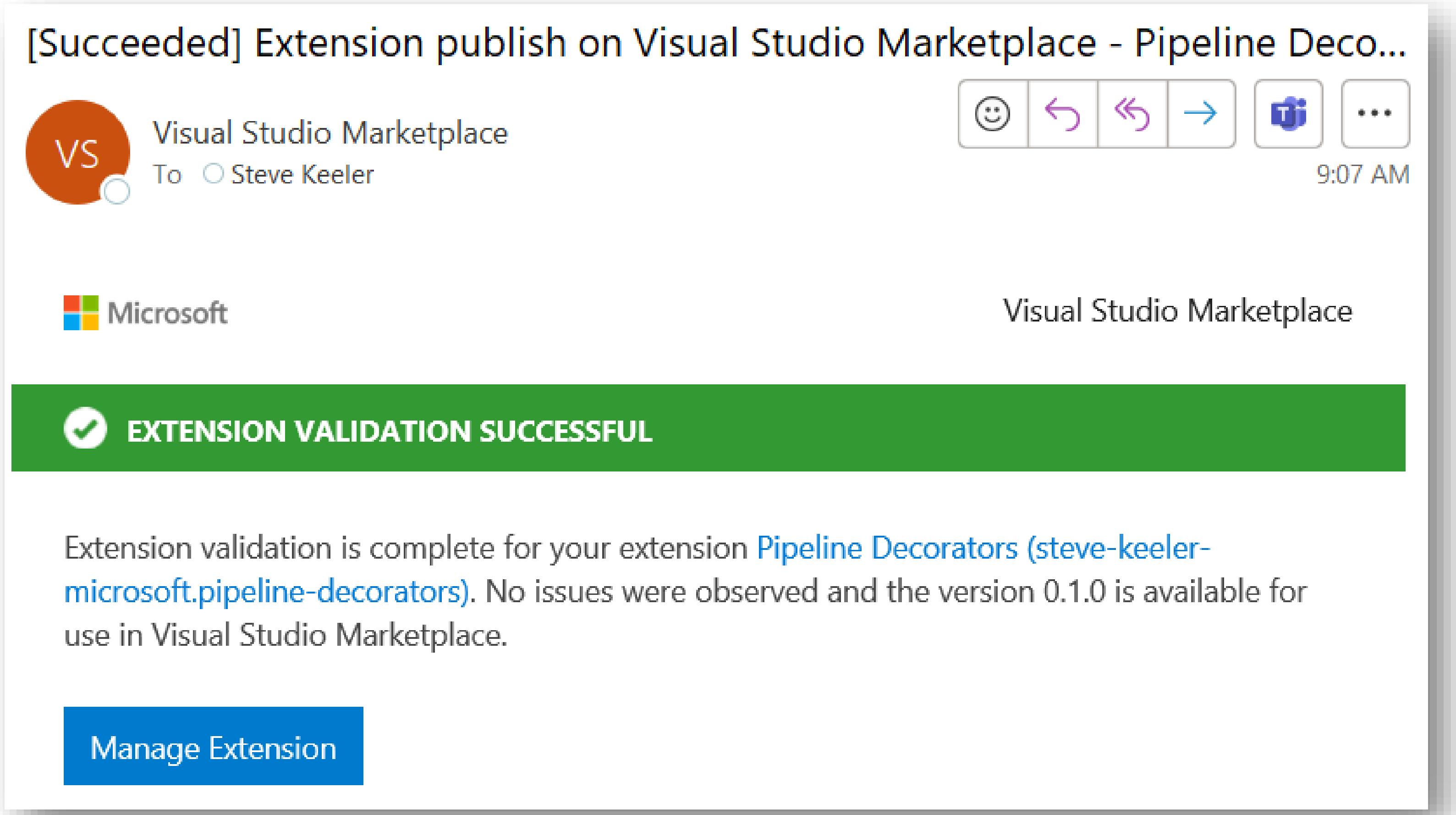
Ensure you have packaged your extension in a VSIX file format.

steve-keeler-microsoft.pipeline-decorators-0.1.0.vsix

Learn about publishing to the Marketplace

Extension validation

- You will receive an email once validation is complete



[Succeeded] Extension publish on Visual Studio Marketplace - Pipeline Deco...

Visual Studio Marketplace
To Steve Keeler
9:07 AM

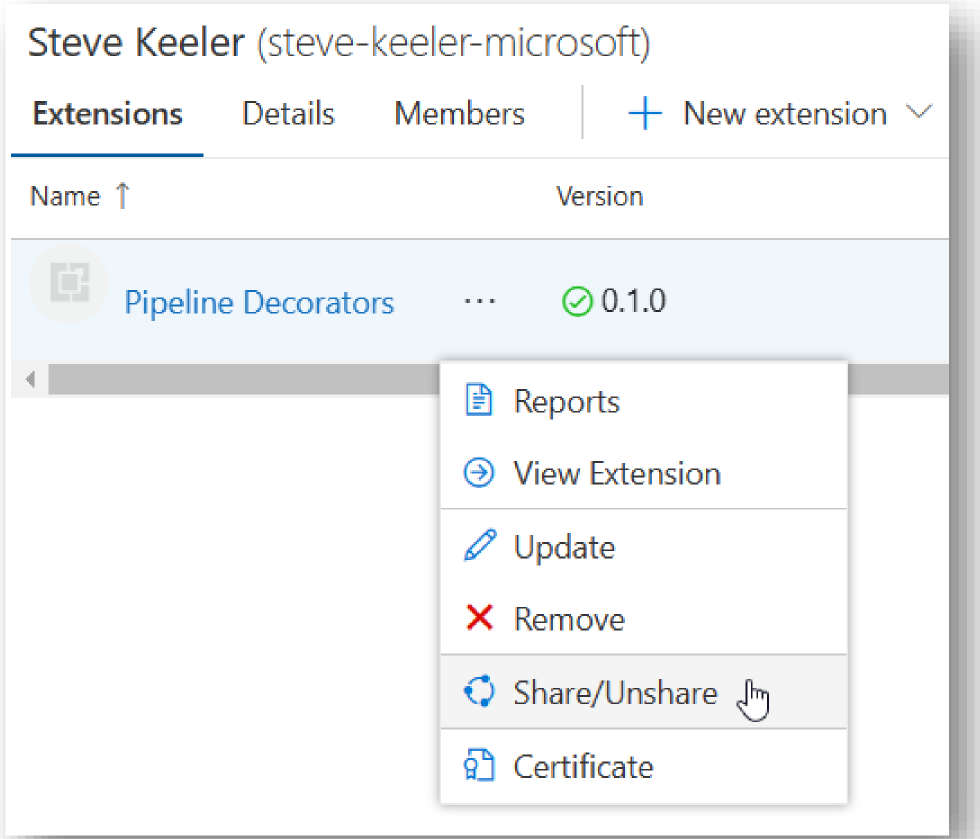
Microsoft Visual Studio Marketplace

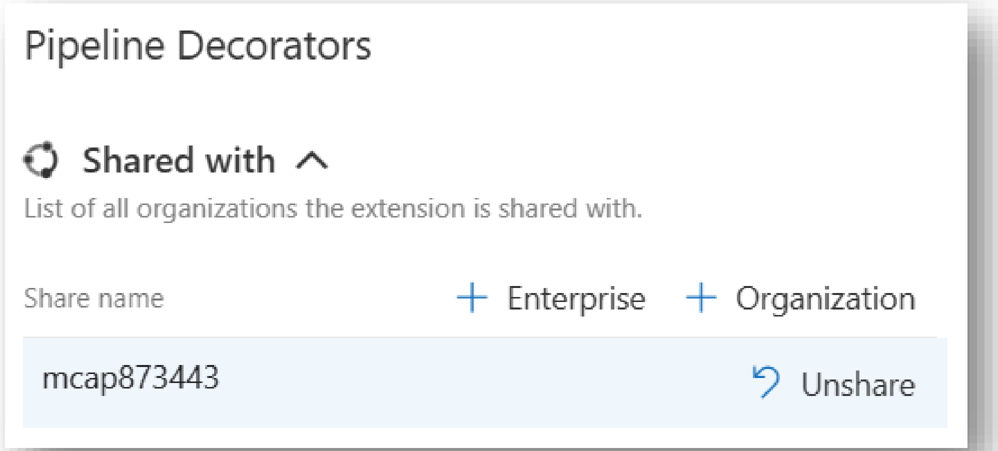
EXTENSION VALIDATION SUCCESSFUL

Extension validation is complete for your extension Pipeline Decorators (steve-keeler-microsoft.pipeline-decorators). No issues were observed and the version 0.1.0 is available for use in Visual Studio Marketplace.

Share extension

- Navigate to https://marketplace.visualstudio.com/manage
- Share your extension from the context menu

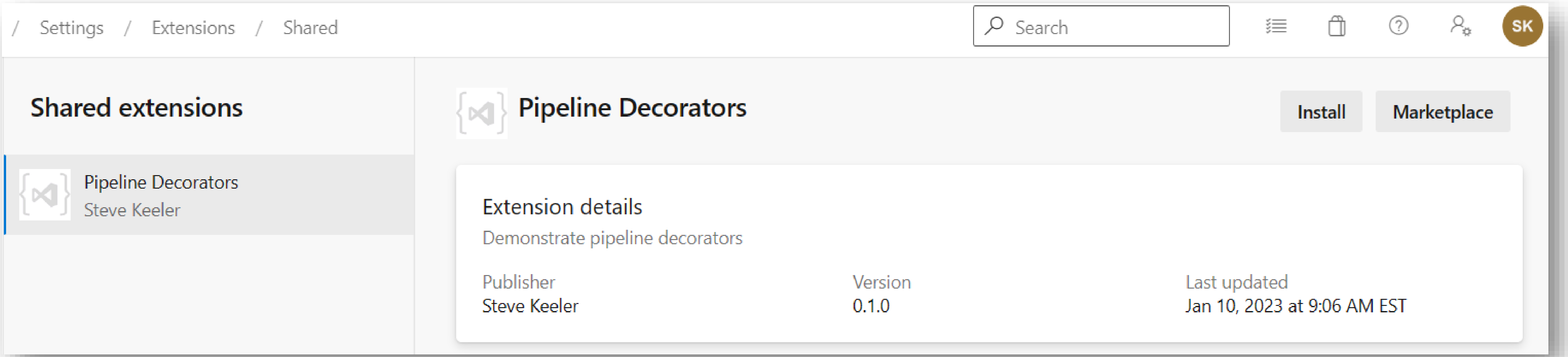




List of all organizations the extension is shared with.

Install extension

- Navigate to Shared Extensions in your Azure DevOps organization
  https://dev.azure.com/<YourDevOpsOrg>/_settings/extensions?tab=shared
- Select the custom extension and click "Install"



Shared extensions

Pipeline Decorators
Steve Keeler

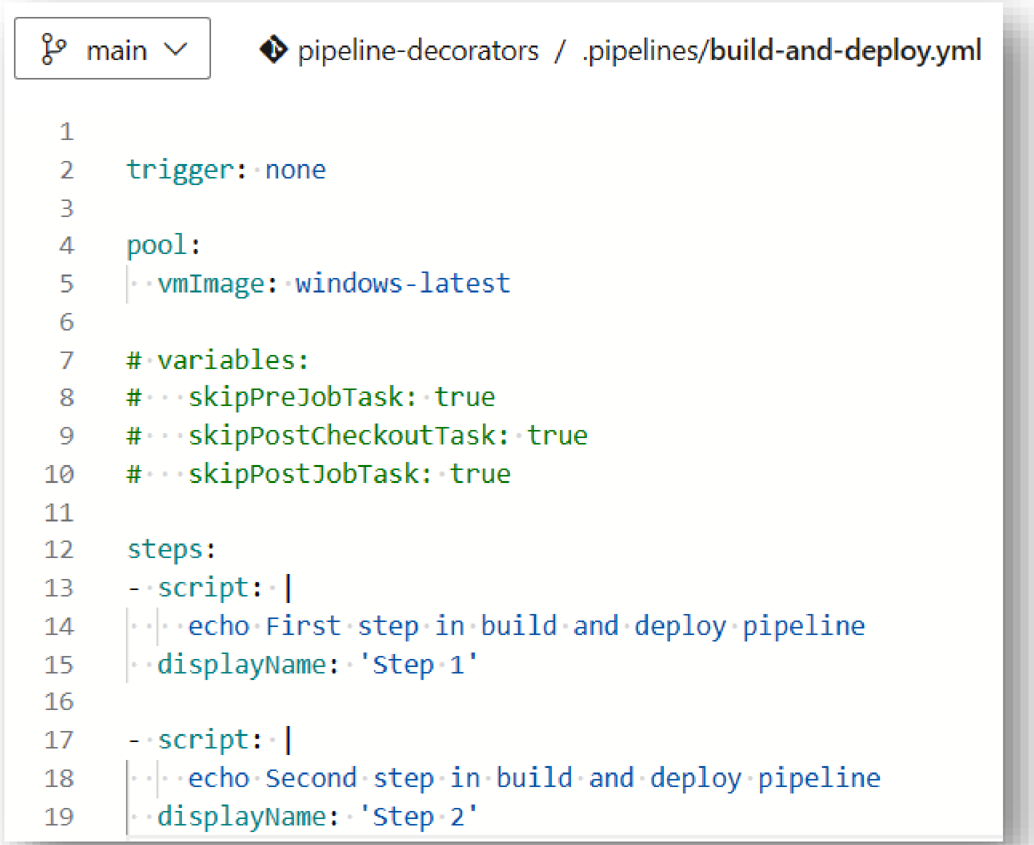
Pipeline Decorators

Extension details
Demonstrate pipeline decorators

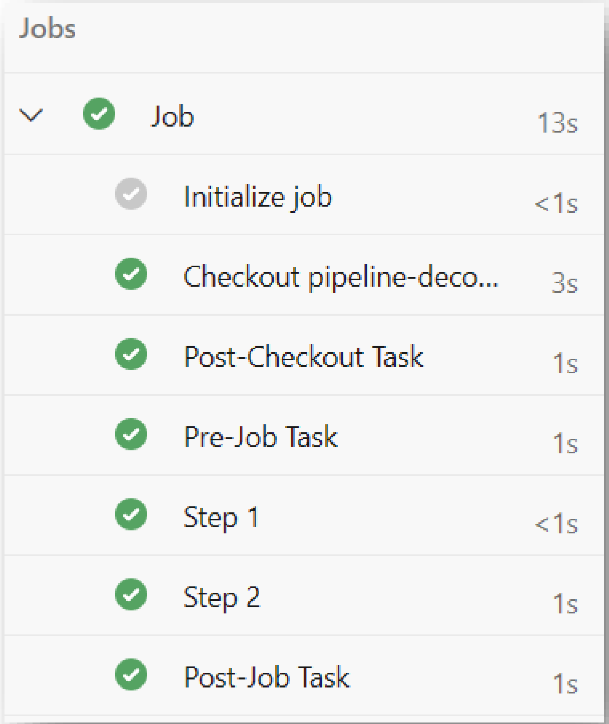
Publisher	Version	Last updated
Steve Keeler	0.1.0	Jan 10, 2023 at 9:06 AM EST

Test extension

Once the decorator is installed in your organization, all pipelines automatically run it. For example:



```yaml

trigger: none

pool:
  vmImage: windows-latest

# variables:
#   skipPreJobTask: true
#   skipPostCheckoutTask: true
#   skipPostJobTask: true

steps:
- script: |
    echo First step in build and deploy pipeline
  displayName: 'Step 1'

- script: |
    echo Second step in build and deploy pipeline
  displayName: 'Step 2'
```



Jobs	
Job	13s
Initialize job	<1s
Checkout pipeline-deco...	3s
Post-Checkout Task	1s
Pre-Job Task	1s
Step 1	<1s
Step 2	1s
Post-Job Task	1s

References

Author a pipeline decorator

https://docs.microsoft.com/azure/devops/extend/develop/add-pipeline-decorator

Extension manifest reference

https://docs.microsoft.com/azure/devops/extend/develop/manifest

Creating custom pipeline task extensions

https://docs.microsoft.com/azure/devops/extend/develop/add-build-task

YAML schema reference

https://docs.microsoft.com/azure/devops/pipelines/yaml-schema

Automate publishing an extension

https://intellitect.com/demystified-azure-pipeline-vsix-extension

Microsoft