CI/CD Documentation for Azure Data Factory with Azure DevOps

Overview

This documentation describes the implementation of a Continuous Integration and Continuous Deployment (CI/CD) process for Azure Data Factory using Azure DevOps as both the repository and automation engine (deployment pipeline).

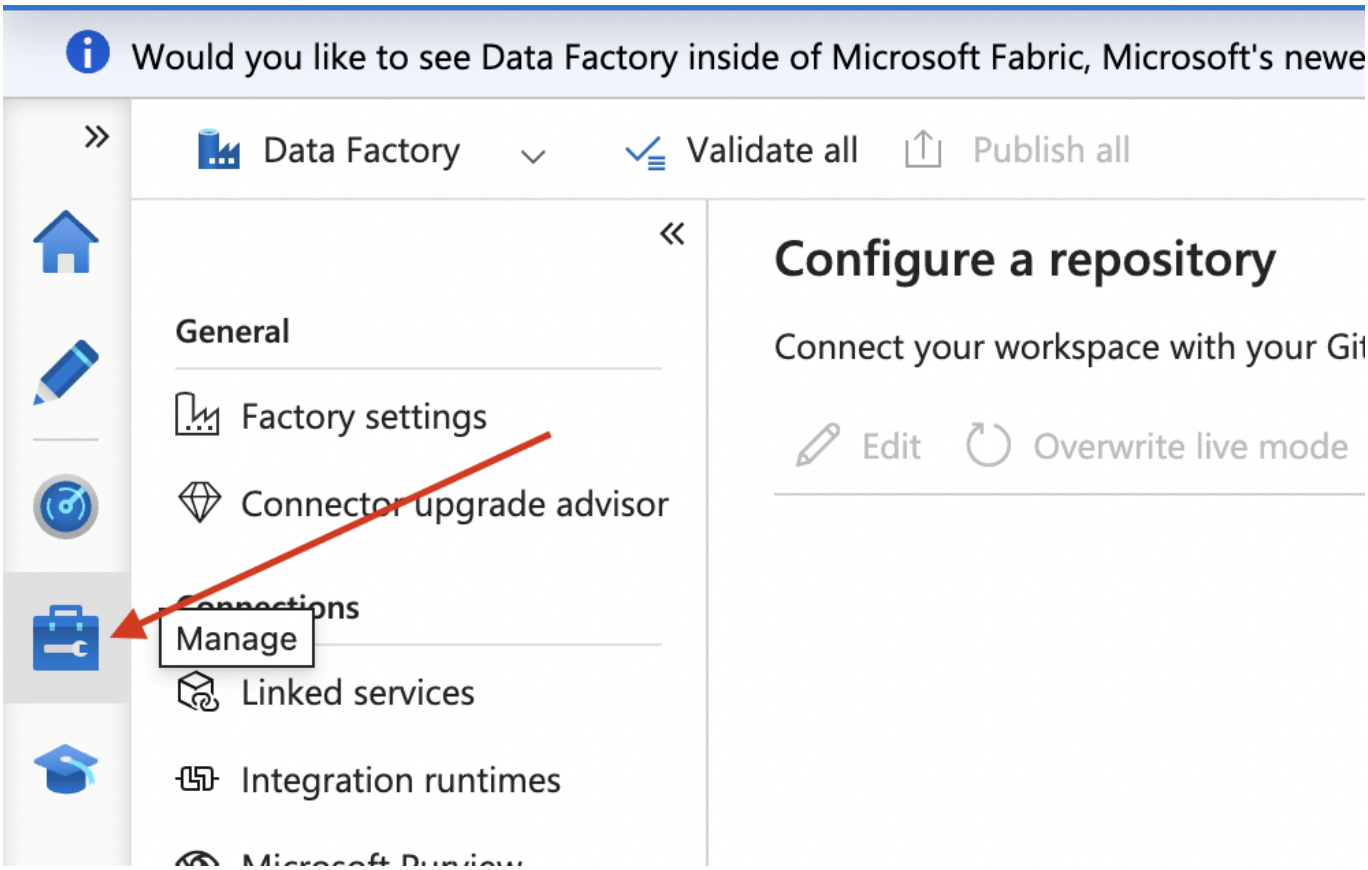
Architecture Overview

- Git Repository: Azure DevOps Repos
- Environments: `Dev` and `QA`
- Pipelines:
  - CI/CD: Automatic deployment to QA environment
- Deployed Resources:
  - Main Artifacts
  - Secondary Artifacts

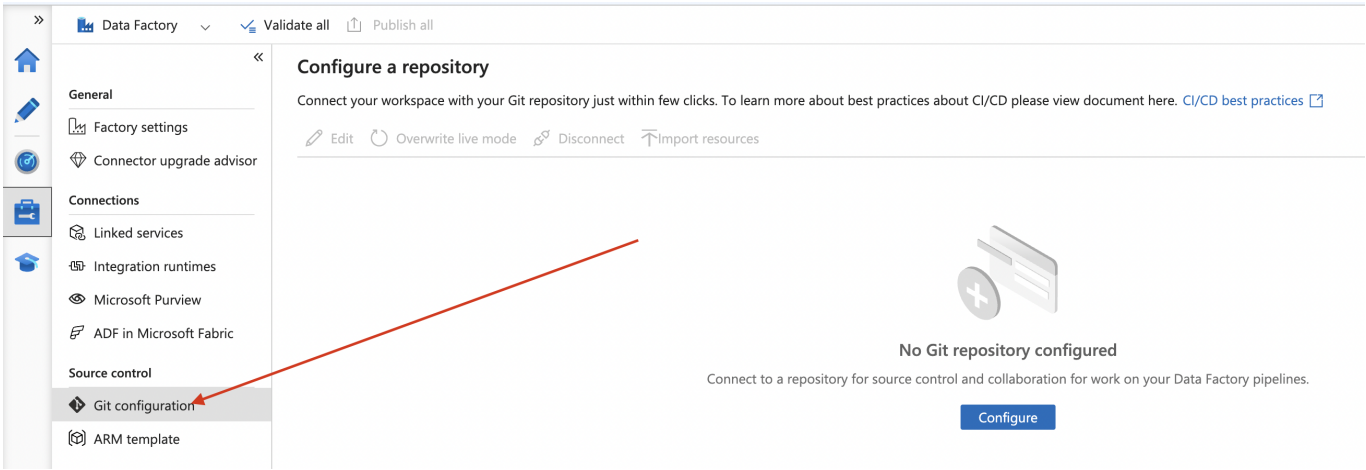
Git Configuration

The Git integration with Azure DevOps is straightforward and easy to set up. It's important to understand the roles of the **collaboration** and **publish** branches. The **collaboration** branch is where all artifacts are organized in a folder-based structure during **development**. In contrast, the **publish** branch is used by the deployment pipeline and contains all necessary artifacts derived from the **collaboration** branch but transformed into a JSON template format that simplifies the deployment process. Therefore, it's considered a good practice to keep these two branches separate.

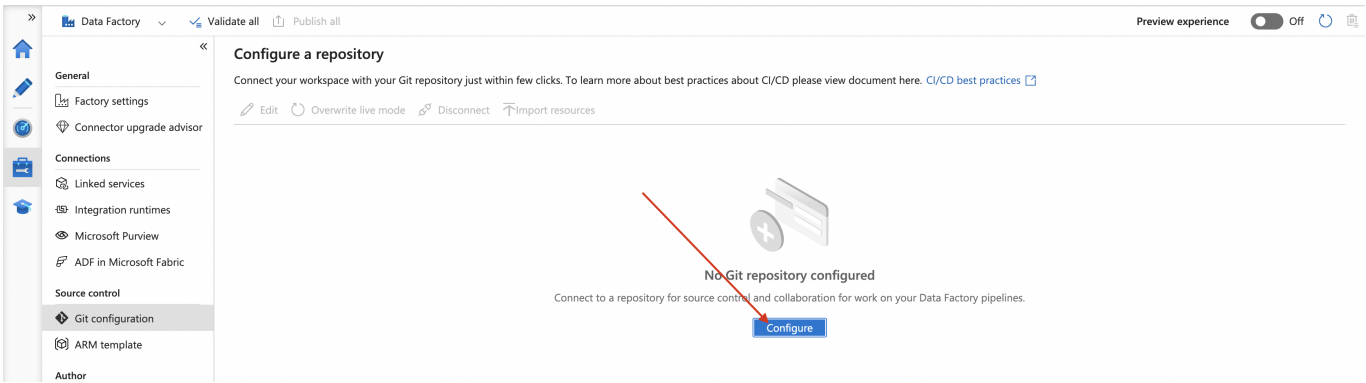
Step 1: Open Data Factory and Click Manage



Step 2: Click Git Configuration



Step 3: Click Configure



Step 4: Configure both repository and Microsoft Entra ID. When done, click continue

Configure a repository

Specify the settings that you want to use when connecting to your repository.

Repository type *

Azure DevOps Git

Cloud Cloud (cross-tenant sign-in) Server (on-premise)

Microsoft Entra ID

Continue Cancel

Step 5: Configure your repository with its corresponding data, have in mind what's been explained about collaboration and publish branch

Configure a repository

Specify the settings that you want to use when connecting to your repository.

(•) Select repository ( ) Use repository link

Azure DevOps organization name *
mate2003-4

Project name *
Emperor

Repository name *
Emperor

Collaboration branch *
main

Publish branch *
adf_publish

Root folder *
/

Custom comment
[x] Use custom comment

Import existing resources
[x] Import existing resources to repository

Import resource into this branch

Final step: click on Apply and that's it!

Repository Structure

```
-adf_publish
/ (root)
├── datafactory/
│   ├── notebooks/
│   ├── pipelines/
│   ├── datasets/
│   ├── linkedServices/
│   └── sqlScripts/
├── azure-pipelines.yml
```

CI/CD Pipeline in Azure DevOps

- Location: `/azure-pipelines.yml`
- Goals:
- Deploy artifacts into QA workspace

```yaml
name: Release-$(rev:r)

trigger:
  branches:
    include:
    - adf_publish

resources:
  repositories:
  - repository: jkDataFactory
    type: git
    name: jkDataFactory
    ref: adf_publish

variables:
  - group: DF_variables

stages:
- stage: Release
  displayName: Release stage
  jobs:
    - job: Release
      displayName: Release job
```

```
      pool:
        vmImage: 'Windows-2019'
      steps:
        - checkout: jkDataFactory

        - task: AzurePowerShell@5
          displayName: Stop Triggers
          inputs:
            azureSubscription: '$(azureSubscription)'
            ScriptType: 'InlineScript'
            Inline:
              $triggersADF = Get-AzDataFactoryV2Trigger -DataFactoryName
              "$(DeployDataFactoryName)" -ResourceGroupName
"$(DeploymentResourceGroupName)";
              $triggersADF | ForEach-Object { Stop-AzDataFactoryV2Trigger
-
              ResourceGroupName "$(DeploymentResourceGroupName)" -
DataFactoryName
              "$(DeployDataFactoryName)" -Name $_.name -Force }
            azurePowerShellVersion: 'LatestVersion'

        - task: AzureResourceManagerTemplateDeployment@3
          inputs:
            deploymentScope: 'Resource Group'
            azureResourceManagerConnection: '$(ServiceConnectionName)'
            subscriptionId: '$(subscriptionID)'
            action: 'Create Or Update Resource Group'
            resourceGroupName: '$(resourceGroupName)'
            location: 'Australia East'
            templateLocation: 'Linked artifact'
            csmFile:
'$(System.DefaultWorkingDirectory)/$(SourceDataFactoryName)/ARMTemplateFor
Factory.json'
            csmParametersFile:
'$(System.DefaultWorkingDirectory)/$(SourceDataFactoryName)/ARMTemplatePar
ametersForFactory.json'
            overrideParameters: '-factoryName $(DeployDataFactoryName)'
            deploymentMode: 'Incremental'

        - task: AzurePowerShell@5
          displayName: Restart Triggers
          inputs:
            azureSubscription: '$(azureSubscription)'
            ScriptType: 'InlineScript'
            Inline:
              $triggersADF = Get-AzDataFactoryV2Trigger -DataFactoryName
"$(DeployDataFactoryName)" -ResourceGroupName
"$(DeploymentResourceGroupName)";
              $triggersADF | ForEach-Object { Start-AzDataFactoryV2Trigger
-ResourceGroupName "$(DeploymentResourceGroupName)" -DataFactoryName
"$(DeployDataFactoryName)" -Name $_.name -Force }
            azurePowerShellVersion: 'LatestVersion'
```

Key notes

- Pipeline has 3 tasks, two of them (AzurePowerShell@5) are in other words, a traffic light functionality to avoid pipeline to run too many times when not desired. But both tasks won't affect pipeline's main goal. for more information visit: https://techcommunity.microsoft.com/blog/azuredatablog/azure-data-factory-ci-cd-using-yaml-template/3107341
- Main task: AzureResourceManagerTemplateDeployment@3. This is task will deploy all corresponding artifacts into the QA workspace.
- DO NOT CHANGE:

Library Variables

Check the following part of the code above:

```
variables:
  - group: DF_variables
```

Variable group name is `DF_variables` you should create a **Variable Group** under Library Section in Azure Devops with that name. Variables:

- azureSubscription = it's value must be your **Suscription Name**
- ServiceConnectionName = it's value must be your **Service Connection Name** linked to the corresponding **QA Data Factory workspace**
- subscriptionID = it's value must be your **Suscription ID**
- resourceGroupName = it's value must be your **Resource Group Name** which is where the **QA Data Factory workspace** is located at
- DeployDataFactoryName = it's value must be your **QA Data Factory Workspace Name** which is the one you where you want to deploy artifacts

overrideParameters Feature

In some scenarios you would like to override some parameters as **Linked Service's connection strings** so this feature allows you to change the values you specify for those artifacts you'd like to change between different workspaces.

How to do it?

The logic behind this is: Linked Services have unique values and they are most likely to be different between workspaces. So there is an **ARMTemplateParametersForFactory.json** file that will contain all artifacts definition and it's main value, you are suppose to copy the corresponding name of the artifact that might differ between environments and copy it's value from the Azure Portal GUI.

- **IMPORTANT:** This should be done BEFORE publishing any changes to the ADO repo.

Step 1:

Head to `datafactoryname/ARMTemplateParametersForFactory.json` and copy the name of the artifact you desire

adf_publish / jk-napsy-df / ARMTemplateParametersForFactory.json

ARMTemplateParametersForFactory.json

Contents History Compare Blame

You updated adf_publish Jun 12

```
1  {
2      "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentParameters.json#",
3      "contentVersion": "1.0.0.0",
4      "parameters": {
5          "factoryName": {
6              "value": "jk-napsy-df"
7          }
8          "HadiTest_connectionString": {
9              "value": ""
10         },
11         "hadi_linked_service_connectionString": {
12             "value": ""
13         },
14         "hadi_linked_service_noConnectionString_connectionString": {
15             "value": ""
16         },
17         "test_connectionString": {
18             "value": ""
19         },
20         "test_blob_storage_connectionString": {
21             "value": ""
22         },
23         "DbTest_password": {
24             "value": ""
25         },
26         "AzureKeyVaultTest_properties_typeProperties_baseUrl": {
27             "value": "https://dfkvdev.vault.azure.net/"
28         },
29         "DbTest_properties_typeProperties_server": {
30             "value": "synapsedbserver.database.windows.net"
31         },
32         "DbTest_properties_typeProperties_database": {
33             "value": "synapsedb"
34         },
35         "DbTest_properties_typeProperties_userName": {
36             "value": "root123"
37         }
38     }
39 }
```

Step 2:

Go to the Pipeline and add the corresponding artifact with its corresponding value as shown in the picture

```yaml
- task: AzureResourceManagerTemplateDeployment@3
  inputs:
    deploymentScope: 'Resource Group'
    azureResourceManagerConnection: '$(ServiceConnectionName)'
    subscriptionId: '$(subscriptionID)'
    action: 'Create Or Update Resource Group'
    resourceGroupName: '$(resourceGroupName)'
    location: 'Australia East'
    templateLocation: 'Linked artifact'
    csmFile: '$(System.DefaultWorkingDirectory)/$(SourceDataFactoryName)/ARMTemplateForFactory.json'
    csmParametersFile: '$(System.DefaultWorkingDirectory)/$(SourceDataFactoryName)/ARMTemplateParametersForFactory.json'
    overrideParameters: '-factoryName $(DeployDataFactoryName) HadiTest_connectionString itscorrespondingvalue'
    deploymentMode: 'Incremental'
```

Connections

- `Service Connection`: Service connection in Azure DevOps with RBAC permissions

Resources

- Official Data Factory CI/CD Documentation

Notes

- Pipeline won't work the first time: when runnning the pipeline for the first time it will ask for an authorization it only takes a click into the UI