Continuous Integration and Continuous Delivery with Azure Data Factory and Azure DevOps

Azure Labs by Roque Daudt (rdaudt@yahoo.com)

08 – Pipelines as code

Following the industry trend to support Pipeline as Code, Microsoft announced General Availability of YAM CD features in Azure Pipelines on April 298th, 2020. Again, these lab's goal is not to discuss the many angles of each approach. Instead, they aim to provide an opportunity for practitioners to go hands on with CICD for Azure Data Factory. Still, it is obvious that one of the advantages of Pipeline as code is that the pipeline itself can be subject of version control. That's not a small bonus.

Azure DevOps uses YAML to code pipelines. As the name hints it, YAML is Yet Another Markup language. There is nothing really special about YAML but you will benefit from a quick review before diving into the YAML pipelines. There are many resources out there. One of them is https://learnxinyminutes.com/docs/yaml/. It is not a pre-requisite to run these labs, though.

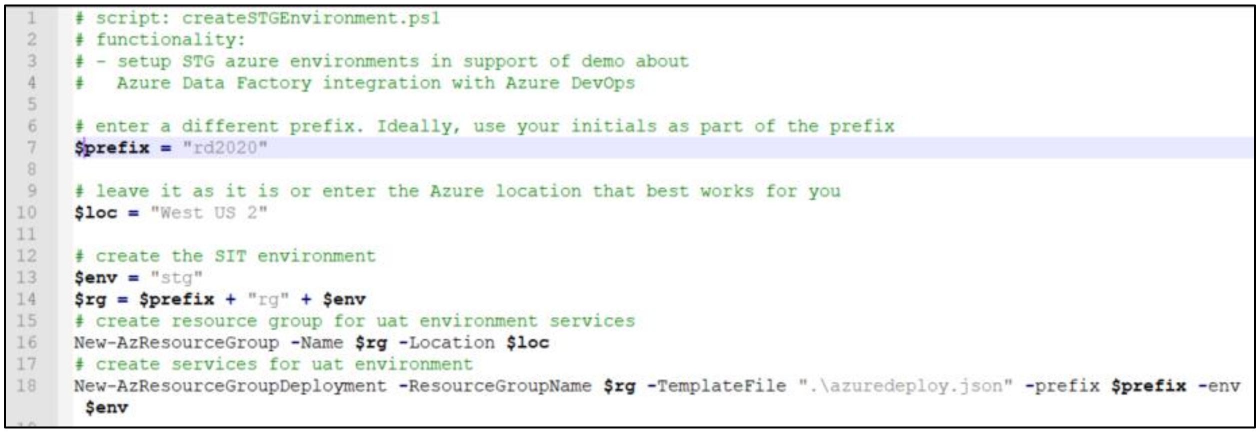
This lab was built based on several articles but mainly on those authored by Alex Volox. See his blog at https://www.alexvolok.com/posts/. It has much more than what is shown in this lab.

It is triggered by any change in the **adf_publish** branch and it deploys the pipeline to **ADF Stg**.

Lab Prep: create STG environment

We could certainly reuse any of the environments that we have provisioned so far for the previous labs. Still, to make sure that we are going to start from a clean slate, we will create another environment. Name in the lab will be staging (**stg**) but it could be anything. We just want to provision a clean, new environment for the lab.

First, you need to customize the script that will create the new environment. As usual, what you need to do is to open the script and enter your own prefix at line 7. The script is **C:\ADFDEVOPS\resources\createSTGEnvironment.ps1**



```
# script: createSTGEnvironment.ps1
# functionality:
# - setup STG azure environments in support of demo about
#   Azure Data Factory integration with Azure DevOps

# enter a different prefix. Ideally, use your initials as part of the prefix
$prefix = "rd2020"

# leave it as it is or enter the Azure location that best works for you
$loc = "West US 2"

# create the SIT environment
$env = "stg"
$rg = $prefix + "rg" + $env
# create resource group for uat environment services
New-AzResourceGroup -Name $rg -Location $loc
# create services for uat environment
New-AzResourceGroupDeployment -ResourceGroupName $rg -TemplateFile ".\azuredeploy.json" -prefix $prefix -env
 $env
```

Save and close.

Start PowerShell. Connect to your Azure account with **Connect-AzAccount**. Enter credentials. Then navigate to **C:\ADFDEVOPS\resources**. Run **Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass** to ensure that you can run the script. Confirm it.

Then, run **.\createSTGEnvironment.ps1**. Once this is completed, browse Azure and ensure that a new resource group was created and that it has a data factory, a key vault and a storage account. In my case, the resource group's name is **rd2020rgstg**.

Lab Prep: configure adf_publish branch

We need to commit some new, supporting folders and files to **adf_publish**. Do as shown below.

Go to Azure DevOps, make sure to be at the **adf_publish** branch.

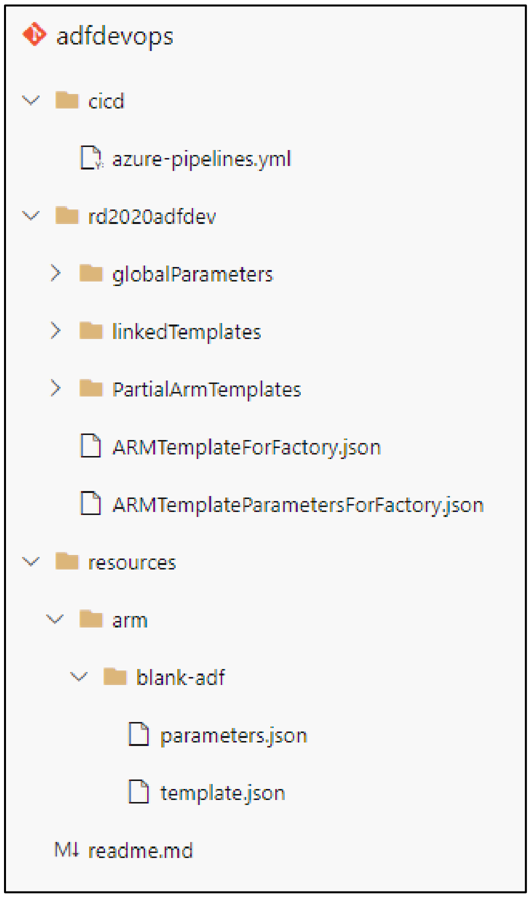
Under the main folder, create folder **cicd** and create an empty file in it named **azure-pipelines.yml**.

Create folders and subfolders **resources/arm/blank-adf**. Create two files in it: **parameters.json** and **template.json**. Copy their contents from files in the resource area, in your workstation:

C:\ADFDEVOPS\resources\parameters.json

C:\ADFDEVOPS\resources\template.json

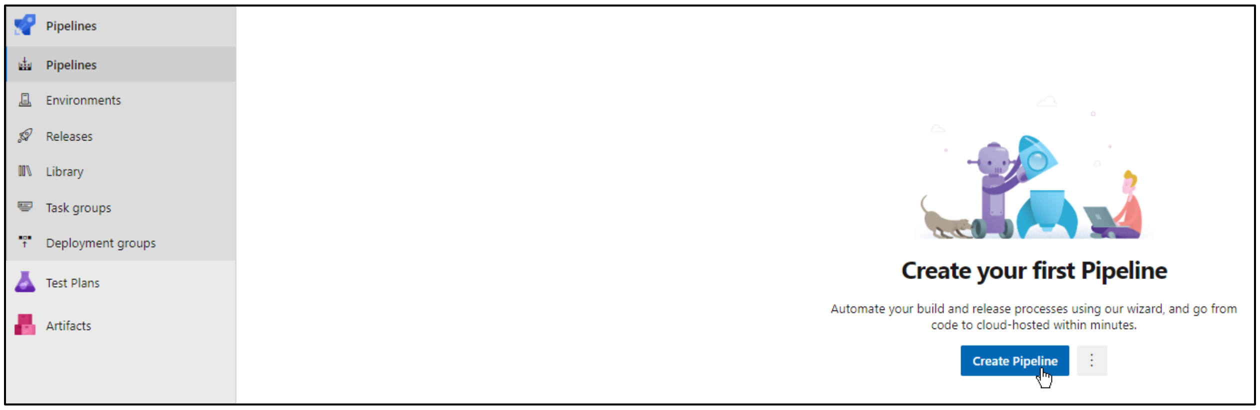
This is how the repo should look like.



```
adfdevops
  cicd
    azure-pipelines.yml
  rd2020adfdev
    globalParameters
    linkedTemplates
    PartialArmTemplates
    ARMTemplateForFactory.json
    ARMTemplateParametersForFactory.json
  resources
    arm
      blank-adf
        parameters.json
        template.json
  readme.md
```

Create a new Pipeline

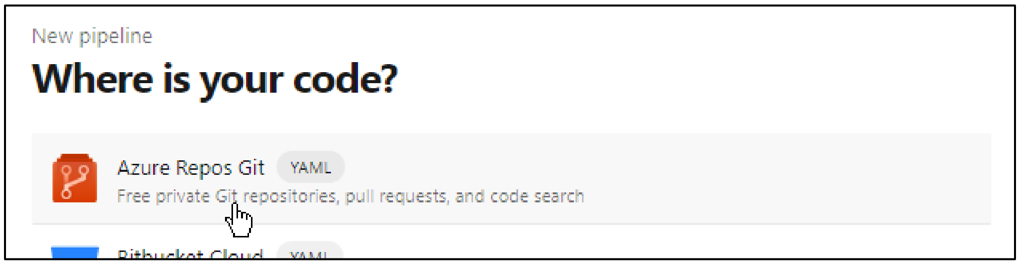
Navigate to **Pipelines / Pipeline** and click **Create Pipeline**.



Create your first Pipeline

Automate your build and release processes using our wizard, and go from code to cloud-hosted within minutes.

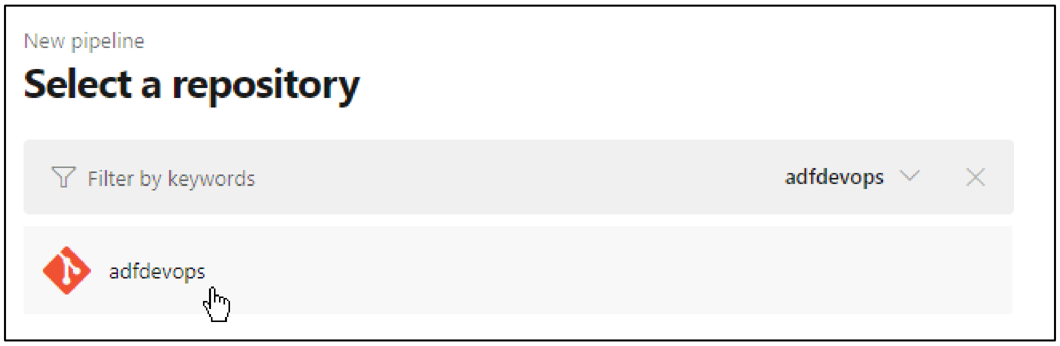
Select **Azure Repos Git**.


New pipeline

Where is your code?

Azure Repos Git YAML
Free private Git repositories, pull requests, and code search

Select the repository

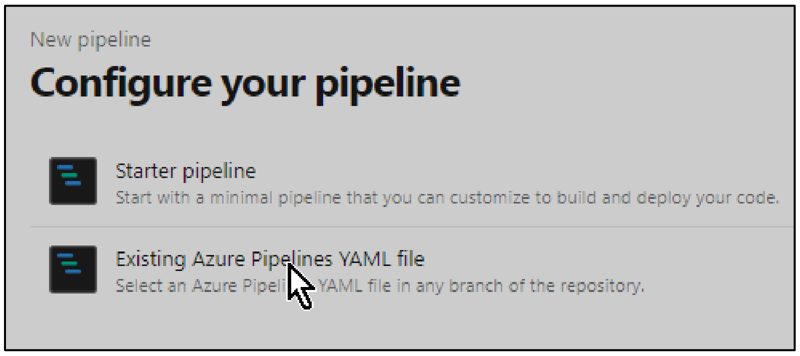

New pipeline

Select a repository

Filter by keywords adfdevops

adfdevops

Select **Existing Azure Pipelines YAML file.**

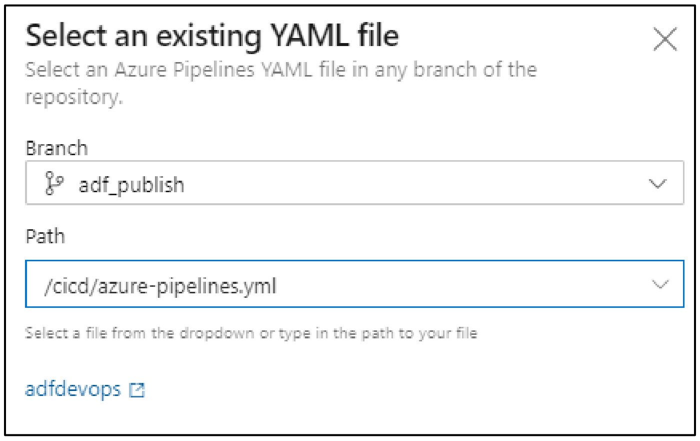

New pipeline

Configure your pipeline

Starter pipeline
Start with a minimal pipeline that you can customize to build and deploy your code.

Existing Azure Pipelines YAML file
Select an Azure Pipeli… YAML file in any branch of the repository.

Select the **adf_publish**_branch and the **path** as shown below.


Select an existing YAML file
Select an Azure Pipelines YAML file in any branch of the repository.

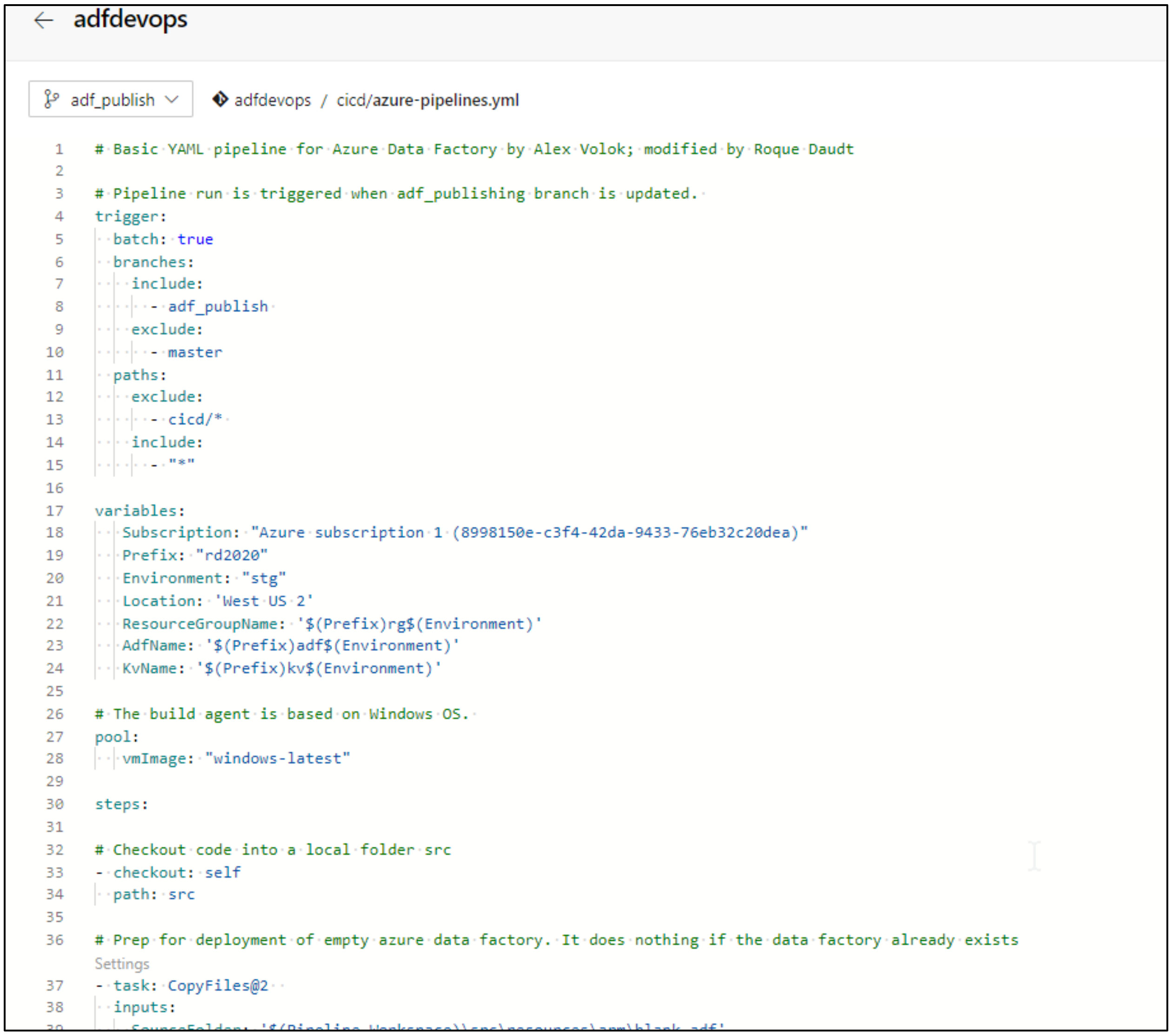
Branch: adf_publish

Path: /cicd/azure-pipelines.yml

Select a file from the dropdown or type in the path to your file

adfdevops

Copy content of file **C:\ADFDEVOPS\resources\adfdevops.yml** into the YAML editor

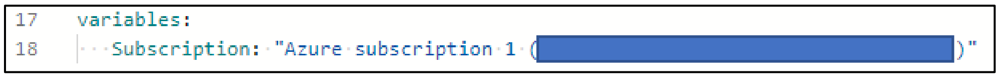

adfdevops

adf_publish   adfdevops / cicd/azure-pipelines.yml

```
1   # Basic YAML pipeline for Azure Data Factory by Alex Volok; modified by Roque Daudt
2
3   # Pipeline run is triggered when adf_publishing branch is updated.
4   trigger:
5     batch: true
6     branches:
7       include:
8         - adf_publish
9       exclude:
10        - master
11    paths:
12      exclude:
13        - cicd/*
14      include:
15        - "*"
16
17  variables:
18     Subscription: "Azure subscription 1 (8998150e-c3f4-42da-9433-76eb32c20dea)"
19     Prefix: "rd2020"
20     Environment: "stg"
21     Location: 'West US 2'
22     ResourceGroupName: '$(Prefix)rg$(Environment)'
23     AdfName: '$(Prefix)adf$(Environment)'
24     KvName: '$(Prefix)kv$(Environment)'
25
26  # The build agent is based on Windows OS.
27  pool:
28     vmImage: "windows-latest"
29
30  steps:
31
32  # Checkout code into a local folder src
33  - checkout: self
34    path: src
35
36  # Prep for deployment of empty azure data factory. It does nothing if the data factory already exists
    Settings
37  - task: CopyFiles@2
38    inputs:
```

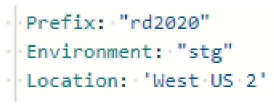
Customize the pipeline with your identifiers, location and subscription:

- At line **18**, replace the content with your subscription name and number, as shown.



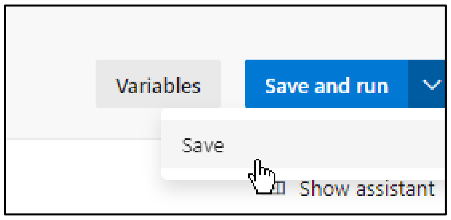
```
17  variables:
18     Subscription: "Azure subscription 1 ([redacted])"
```

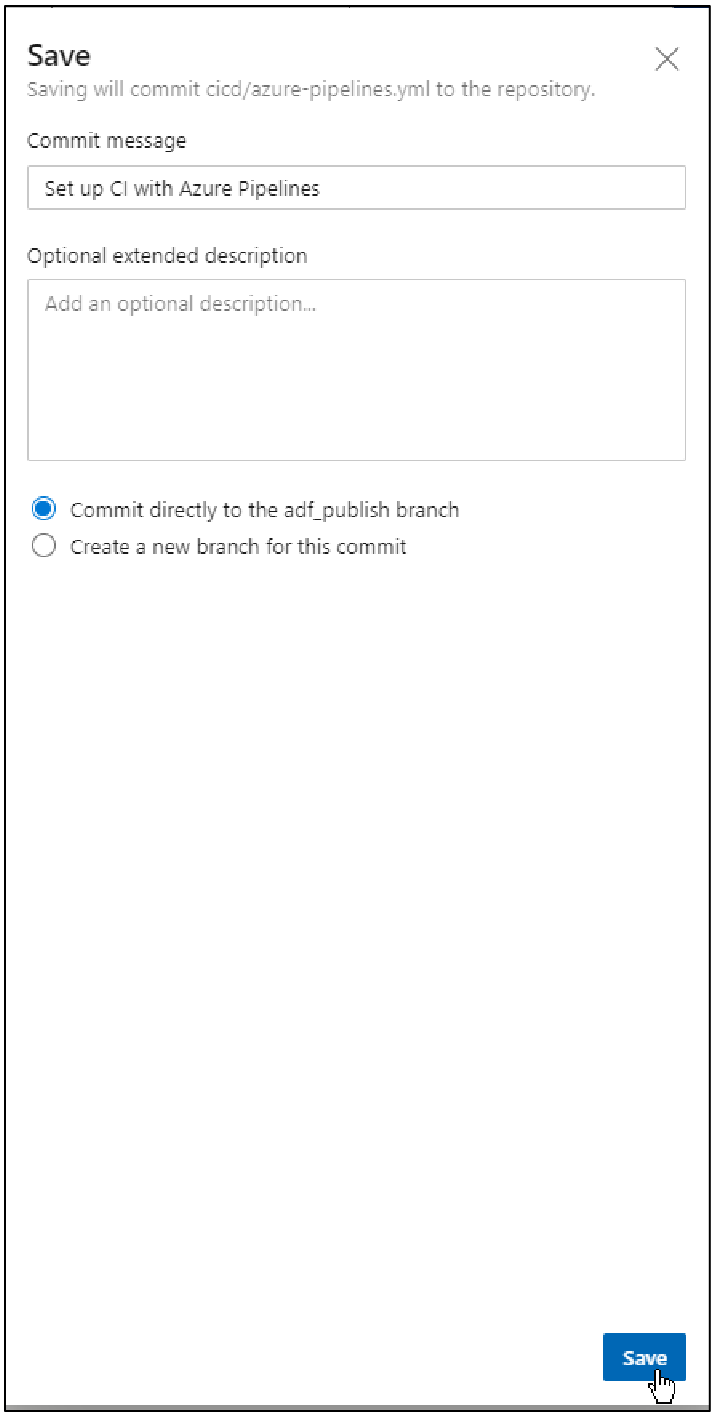
- At lines 19 to 21, enter the prefix, environment and location you entered when provisioning the environment for this lab.



```
   Prefix: "rd2020"
   Environment: "stg"
   Location: 'West US 2'
```

Save the pipeline.





Save

Saving will commit cicd/azure-pipelines.yml to the repository.

Commit message

Set up CI with Azure Pipelines

Optional extended description

Add an optional description...

[✓ Commit directly to the adf_publish branch]

Create a new branch for this commit

Run the pipeline

If we were in a non-lab environment, in order to test the pipeline we would run the integration process again in the Azure Data Factory to update the pipeline:

- create a feature branch

- apply change

- save

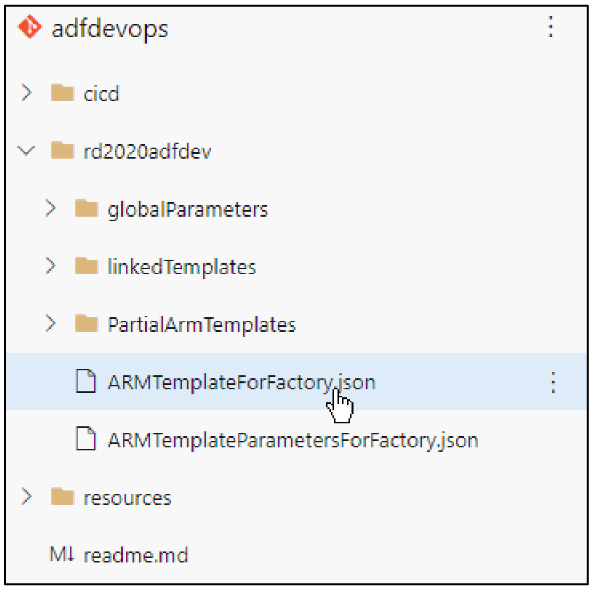
- create a new pull request

- approve/complete

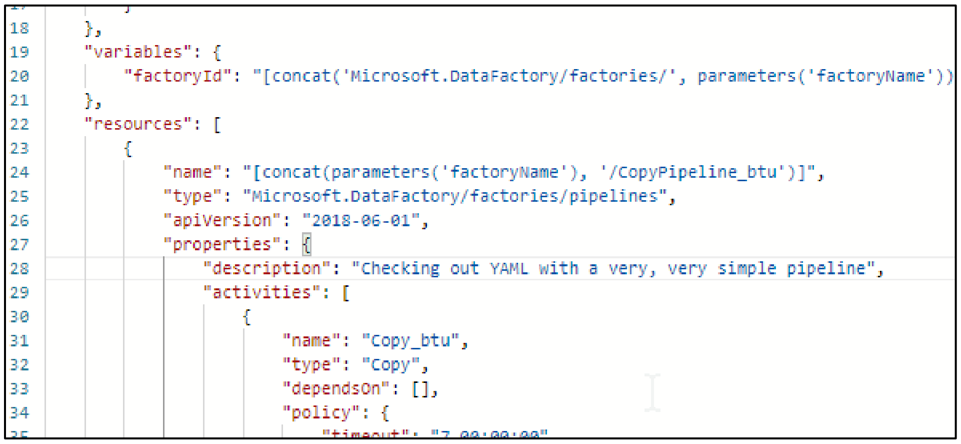
- publish master branch to adf_publish branch so that json files are converted to new ARM files

In this lab, to make things easier and faster, though, we will commit some change directly to an ARM file in adf_publish. Don't do it in a professional setting.

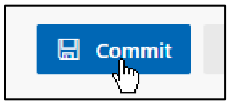
In Azure DevOps go to the **adf_publish** branch and click **ARMTemplateForFactory.json**

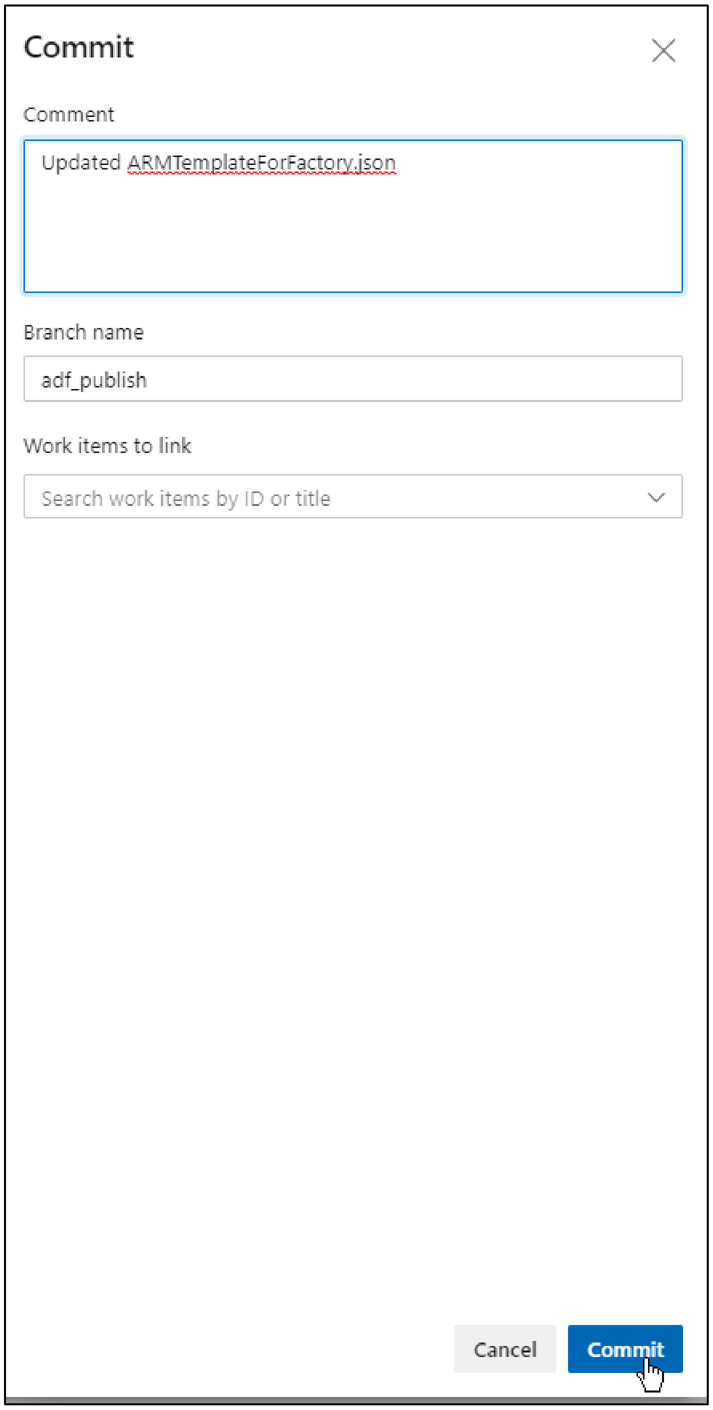


Find and update the pipelines **description**, as shown below.


```
    },
    "variables": {
        "factoryId": "[concat('Microsoft.DataFactory/factories/', parameters('factoryName'))
    },
    "resources": [
        {
            "name": "[concat(parameters('factoryName'), '/CopyPipeline_btu')]",
            "type": "Microsoft.DataFactory/factories/pipelines",
            "apiVersion": "2018-06-01",
            "properties": {
                "description": "Checking out YAML with a very, very simple pipeline",
                "activities": [
                    {
                        "name": "Copy_btu",
                        "type": "Copy",
                        "dependsOn": [],
                        "policy": {
                            "timeout": "7.00:00:00"
```

Click commit



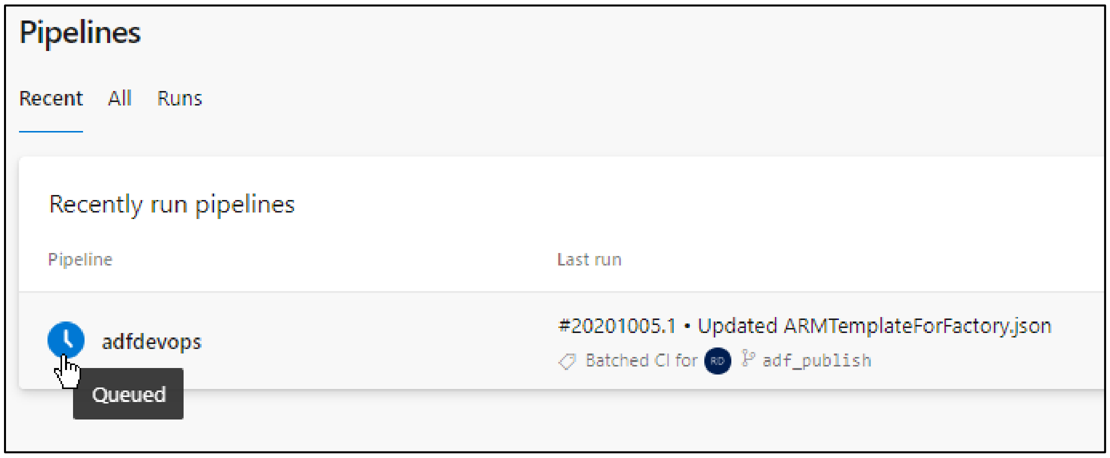


Updated ARMTemplateForFactory.json

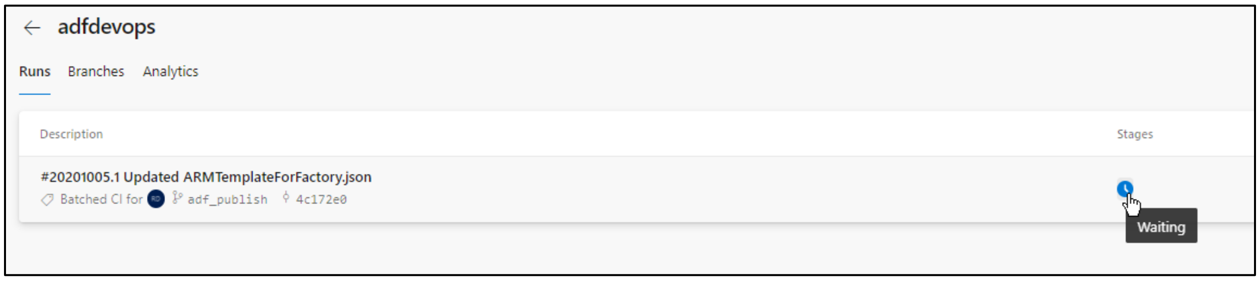
adf_publish

Go to **Pipelines / Pipeline.**

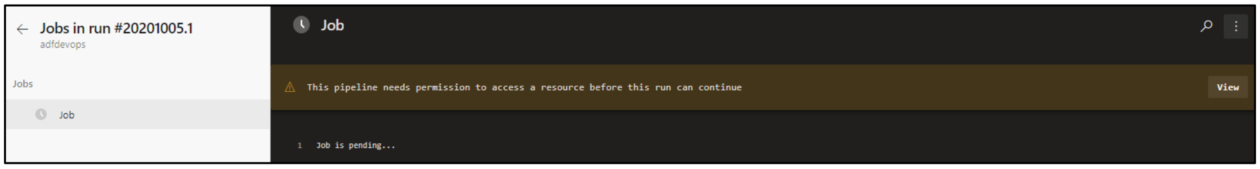
You see that the pipeline was queued



After some time click the pipeline. You will be taken to another view

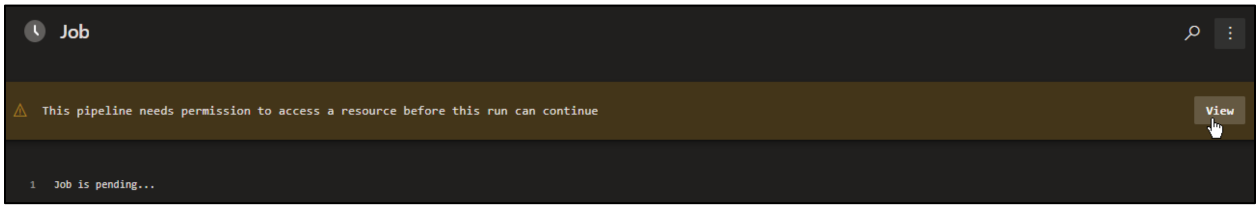


Hover your mouse over the clock icon and now you see that status is **Waiting**. Click the blue icon. See below that is a need to give permissions.



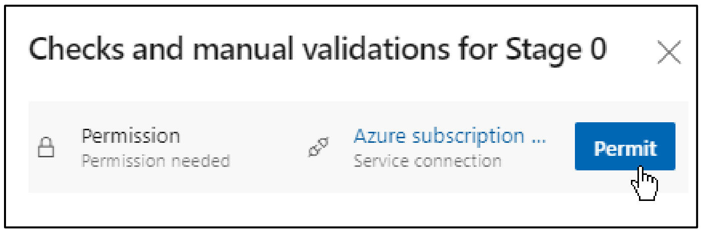
```
This pipeline needs permission to access a resource before this run can continue
1  Job is pending...
```

Click **View**

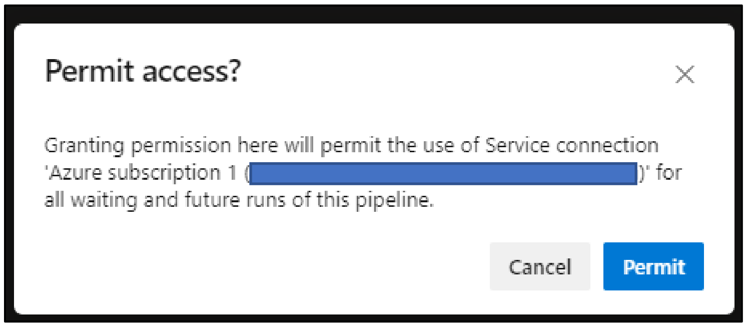


```
This pipeline needs permission to access a resource before this run can continue
1  Job is pending...
```

Click **Permit**

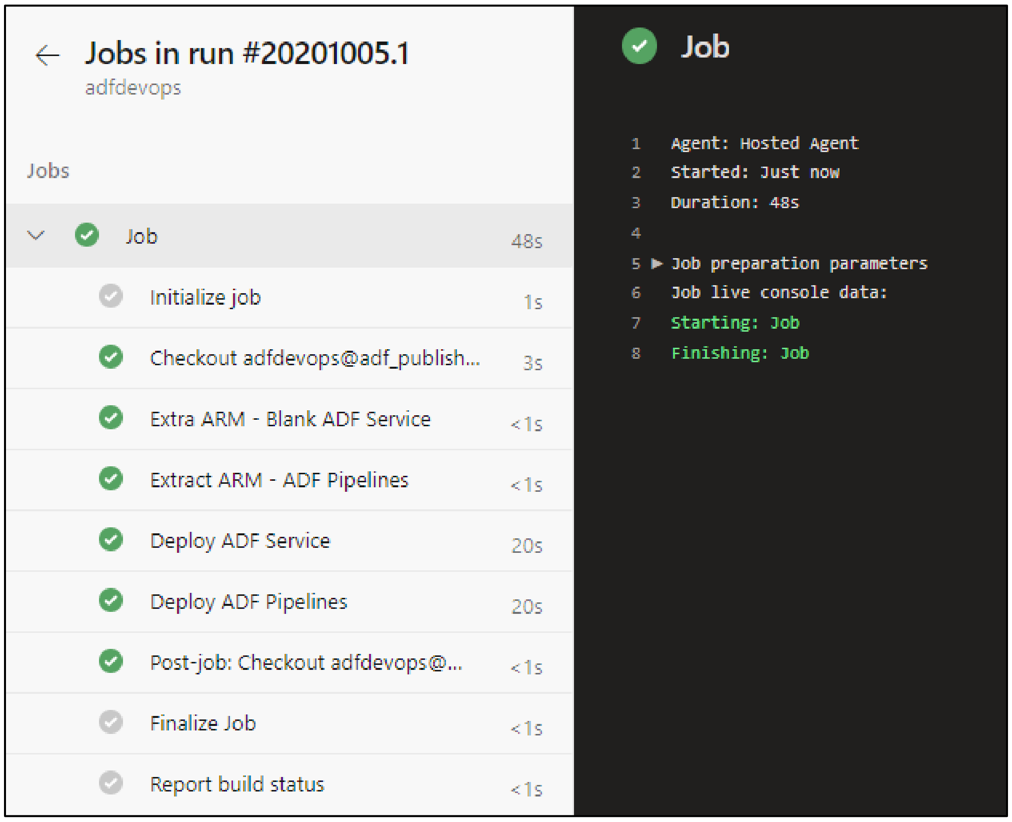


Click **Permit** again


Permit access?

Granting permission here will permit the use of Service connection 'Azure subscription 1 ([redacted])' for all waiting and future runs of this pipeline.

Watch the pipeline run


Jobs in run #20201005.1
adfdevops

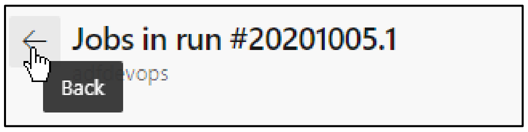
Jobs	
Job	48s
Initialize job	1s
Checkout adfdevops@adf_publish...	3s
Extra ARM - Blank ADF Service	<1s
Extract ARM - ADF Pipelines	<1s
Deploy ADF Service	20s
Deploy ADF Pipelines	20s
Post-job: Checkout adfdevops@...	<1s
Finalize Job	<1s
Report build status	<1s

Job

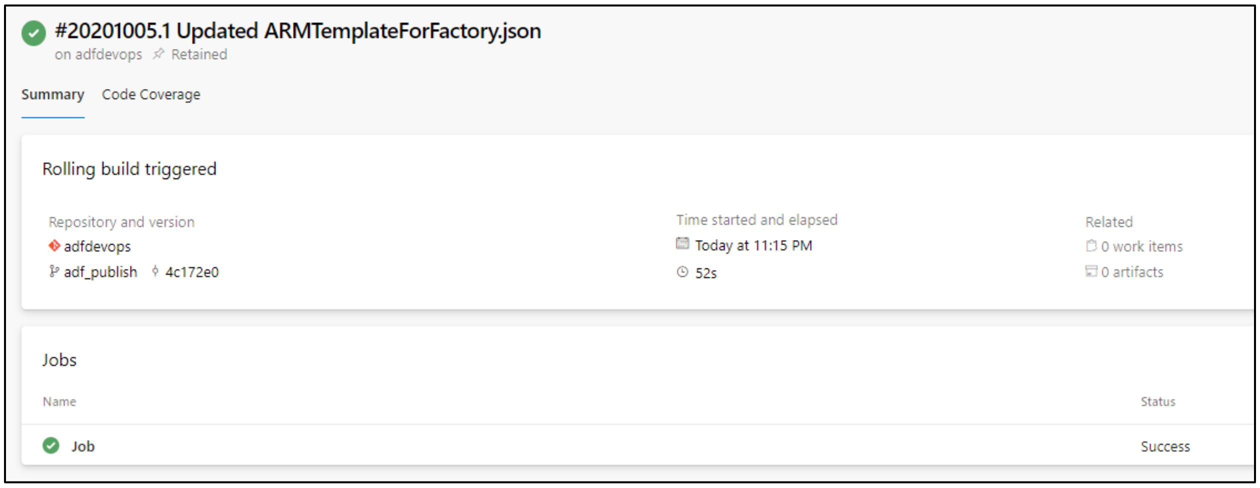
```
Agent: Hosted Agent
Started: Just now
Duration: 48s

▶ Job preparation parameters
Job live console data:
Starting: Job
Finishing: Job
```

Go back


Jobs in run #20201005.1

Confirm the successful execution.


#20201005.1 Updated ARMTemplateForFactory.json
on adfdevops Retained

Summary Code Coverage

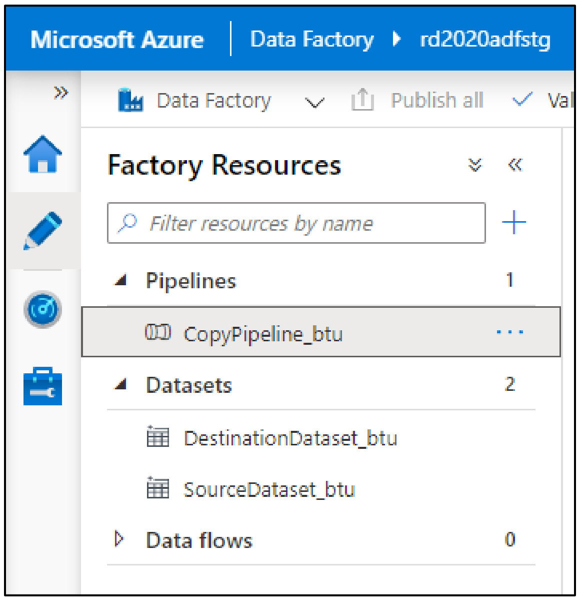
Rolling build triggered

Repository and version	Time started and elapsed	Related
adfdevops	Today at 11:15 PM	0 work items
adf_publish 4c172e0	52s	0 artifacts

Jobs

Name	Status
Job	Success

Check **ADF Stg** and verify that objects were created.


Microsoft Azure | Data Factory ▸ rd2020adfstg

Data Factory | Publish all | Val

Factory Resources

Filter resources by name

- Pipelines 1
  - CopyPipeline_btu
- Datasets 2
  - DestinationDataset_btu
  - SourceDataset_btu
- Data flows 0

Summary

This concludes the series of labs about CICD for Azure Data Factory. Remember to delete the resource groups that you created while running the labs.

References

SQL Player

https://sqlplayer.net/adftools/

Alex Volok Consultancy

https://www.alexvolok.com/