RATIO

Provision Sitecore in Azure and Continuous Integration using Azure DevOps

04/12/2019

Alina Fodor – Solution Architect

Contents

KISS Principal

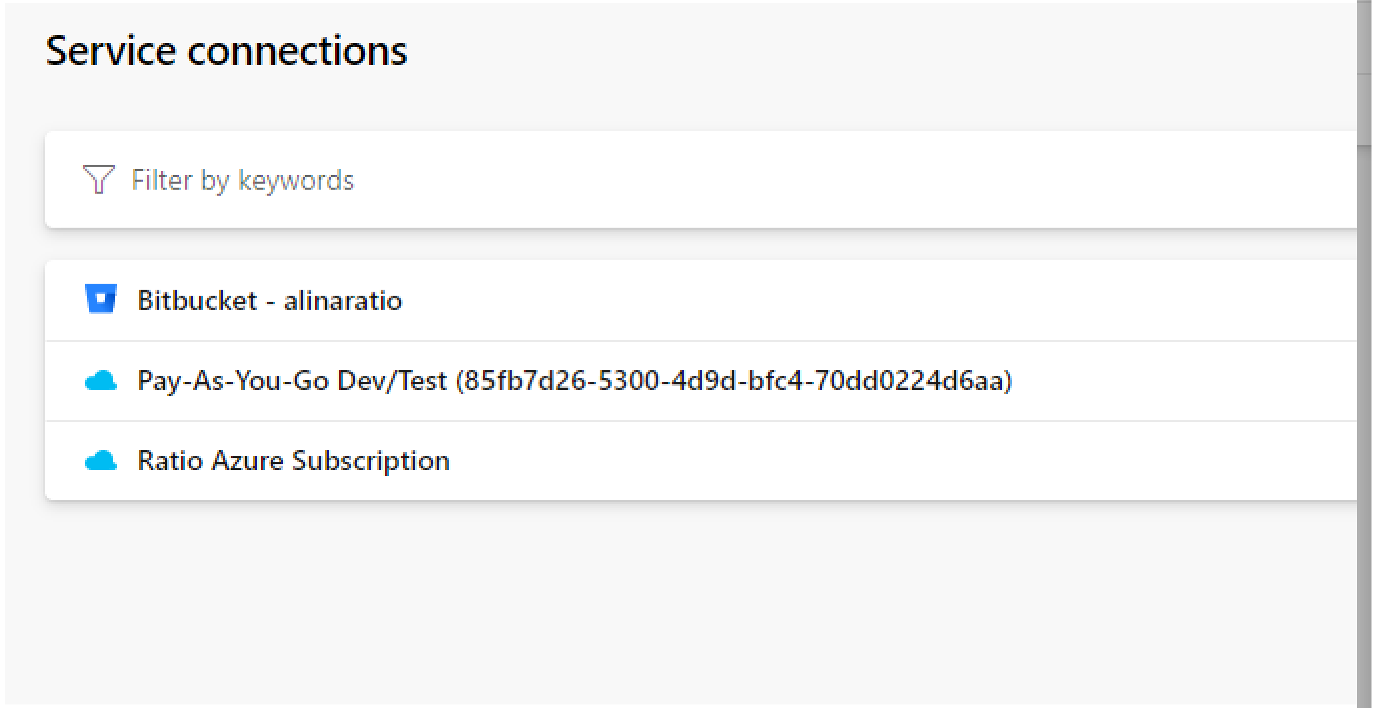
Keep It Simple Stupid

The KISS principle states that most systems work best if they are kept simple rather than made complicated; therefore, simplicity should be a key goal in design, and unnecessary complexity should be avoided.

Connect to Bitbucket and Azure

Service connections in Azure Pipelines are available for use in all your tasks.

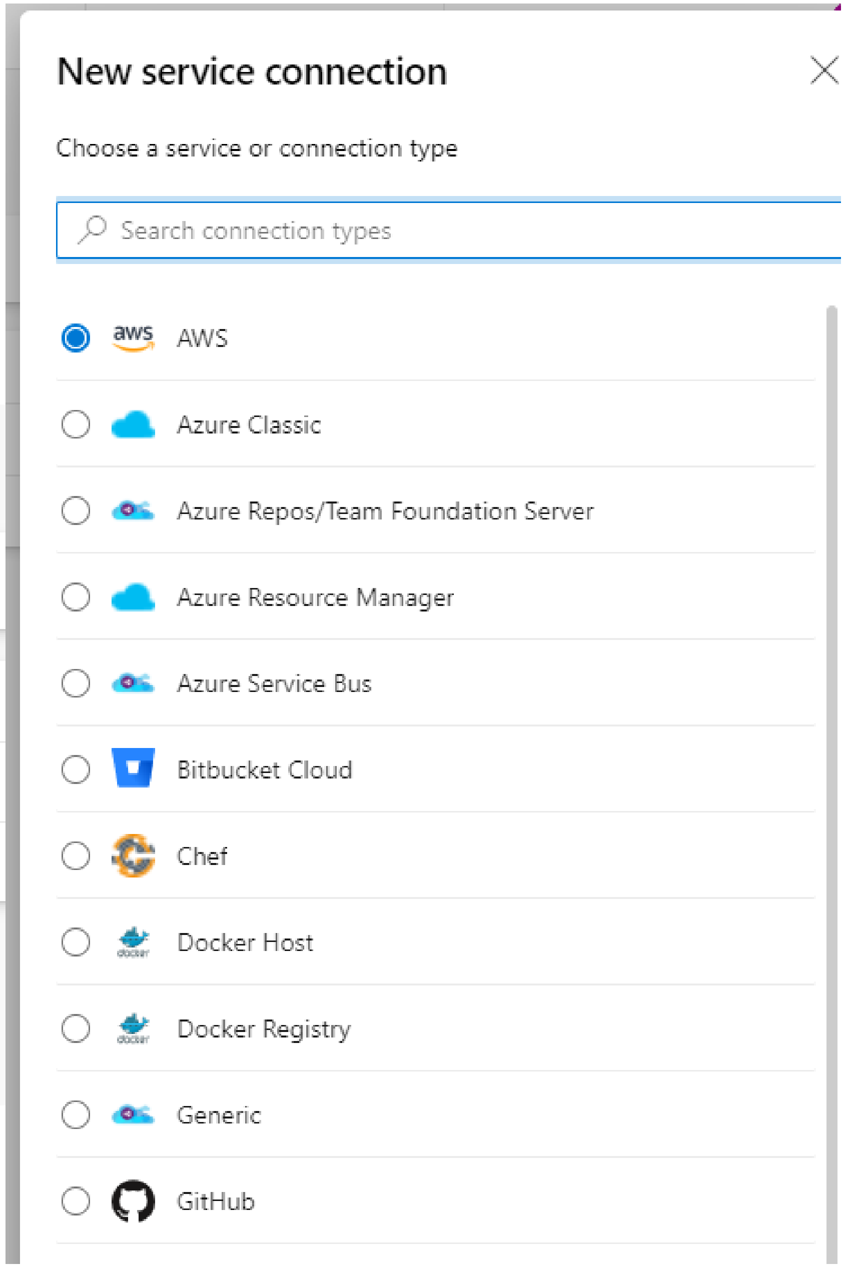
Service connection are created on Project Level but can ne set to be shared across all pipelines.



Service connections

Filter by keywords

- Bitbucket - alinaratio
- Pay-As-You-Go Dev/Test (85fb7d26-5300-4d9d-bfc4-70dd0224d6aa)
- Ratio Azure Subscription



New service connection

Choose a service or connection type

Search connection types

- [✓ AWS]
- Azure Classic
- Azure Repos/Team Foundation Server
- Azure Resource Manager
- Azure Service Bus
- Bitbucket Cloud
- Chef
- Docker Host
- Docker Registry
- Generic
- GitHub

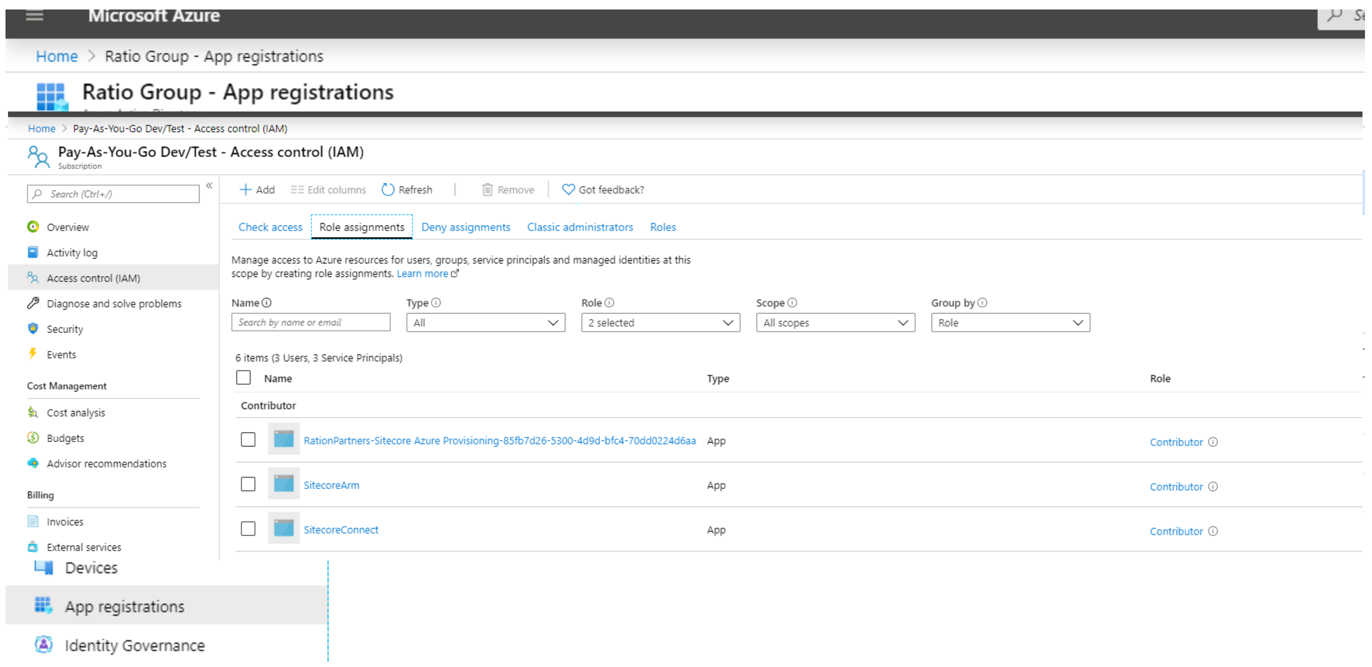
App Registration and Service Principals

An OAuth 2.0 Authorization Grant flow defines the conversation protocol, which allows the client/resource to access/protect a resource's data, respectively.

An Azure AD application is defined by its one and only application object, which resides in the Azure AD tenant where the application was registered, known as the application's "home" tenant.

To access resources that are secured by an Azure AD tenant, the entity that requires access must be represented by a security principal.

When an application is given permission to access resources in a tenant (upon registration or consent), a service principal object is created.



Manage access to Azure resources for users, groups, service principals and managed identities at this scope by creating role assignments. Learn more

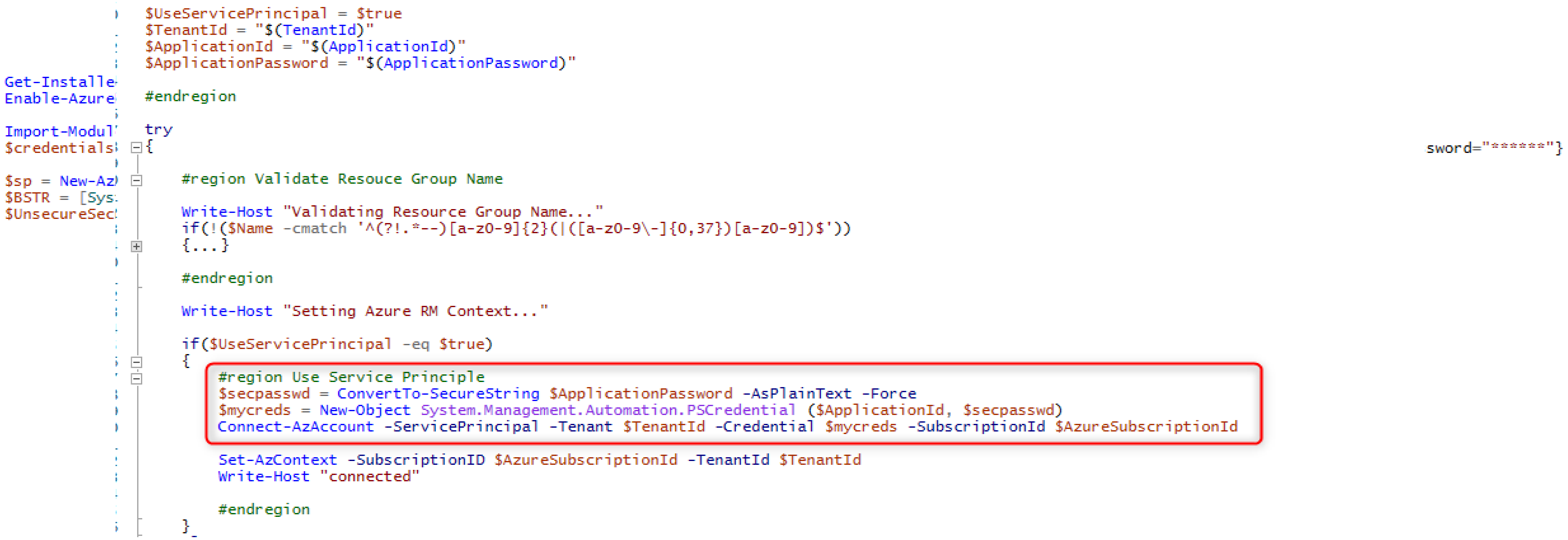
Name: Search by name or email
Type: All
Role: 2 selected
Scope: All scopes
Group by: Role

6 items (3 Users, 3 Service Principals)

Name	Type	Role
Contributor		
RationPartners-Sitecore Azure Provisioning-85fb7d26-5300-4d9d-bfc4-70dd0224d6aa	App	Contributor
SitecoreArm	App	Contributor
SitecoreConnect	App	Contributor

App Registration and Service Principals

A Service Principal can be created using PowerShell. You can connect to Azure and access a subscription to read/write resources.

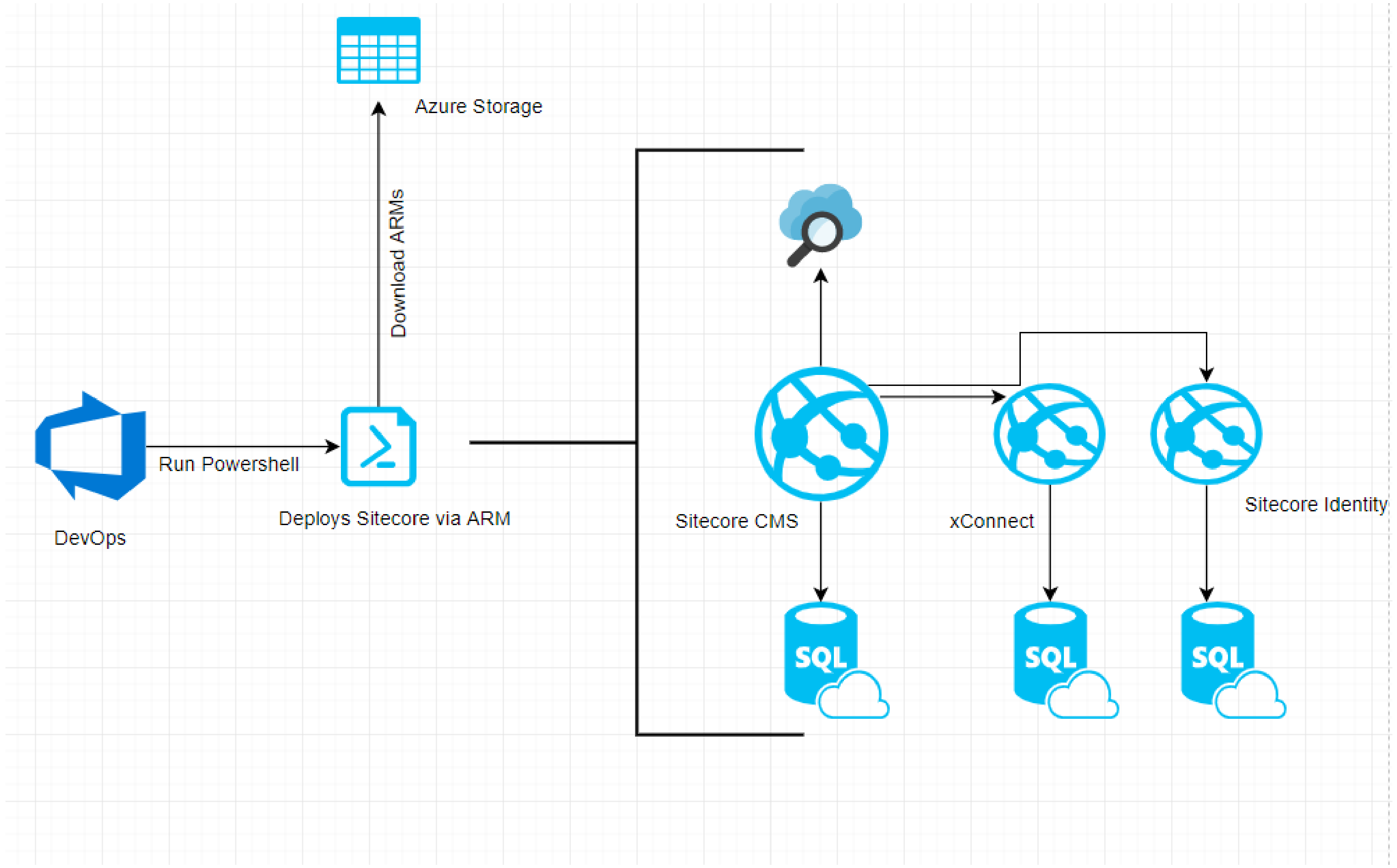


```
Get-Installe
Enable-Azure

Import-Modul
$credentials
                                                                                    sword="******"}
$sp = New-Az
$BSTR = [Sys
$UnsecureSec
```

```
$UseServicePrincipal = $true
$TenantId = "$(TenantId)"
$ApplicationId = "$(ApplicationId)"
$ApplicationPassword = "$(ApplicationPassword)"

#endregion

try
{

    #region Validate Resouce Group Name

    Write-Host "Validating Resource Group Name..."
    if(!($Name -cmatch '^(?!.*--)[a-z0-9]{2}(|([a-z0-9\-]{0,37})[a-z0-9])$'))
    {...}

    #endregion

    Write-Host "Setting Azure RM Context..."

    if($UseServicePrincipal -eq $true)
    {
        #region Use Service Principle
        $secpasswd = ConvertTo-SecureString $ApplicationPassword -AsPlainText -Force
        $mycreds = New-Object System.Management.Automation.PSCredential ($ApplicationId, $secpasswd)
        Connect-AzAccount -ServicePrincipal -Tenant $TenantId -Credential $mycreds -SubscriptionId $AzureSubscriptionId

        Set-AzContext -SubscriptionID $AzureSubscriptionId -TenantId $TenantId
        Write-Host "connected"

        #endregion
    }
```

Provision Sitecore in Azure

https://github.com/Sitecore/Sitecore-Azure-Quickstart-Templates

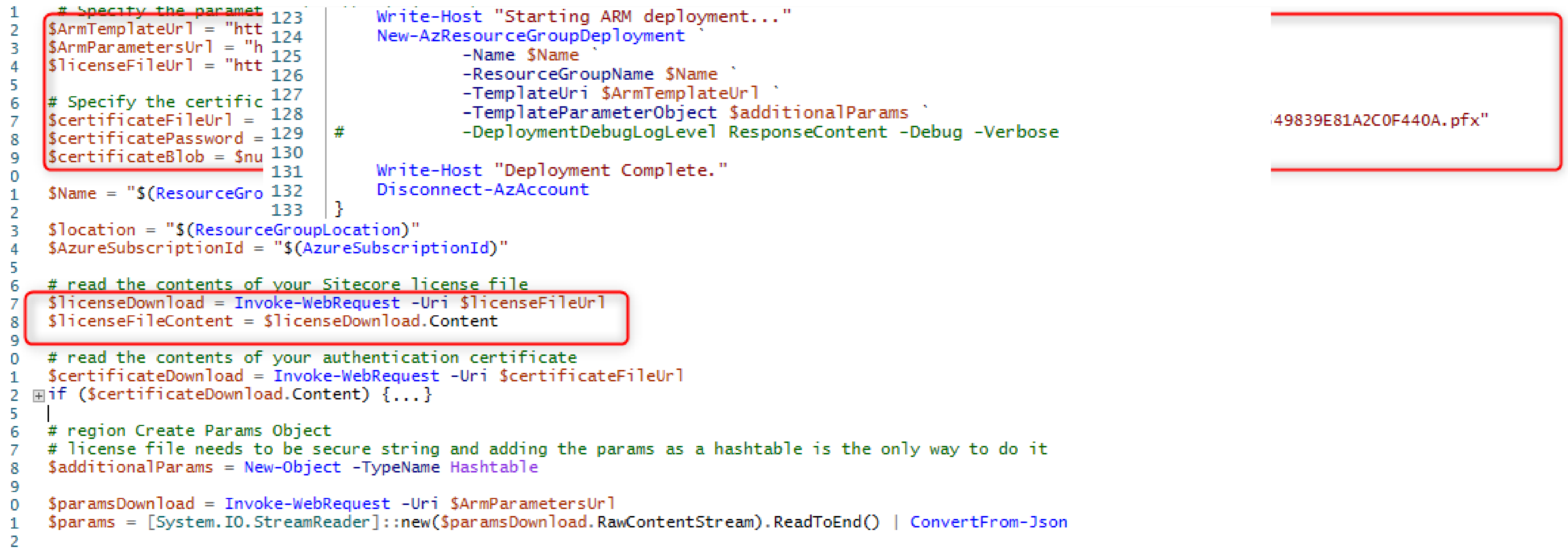


DevOps Release

To provision Sitecore 9.2 in Azure is enough to run a PowerShell that deploys Sitecore ARM on the desired subscription.

Pre-requirements

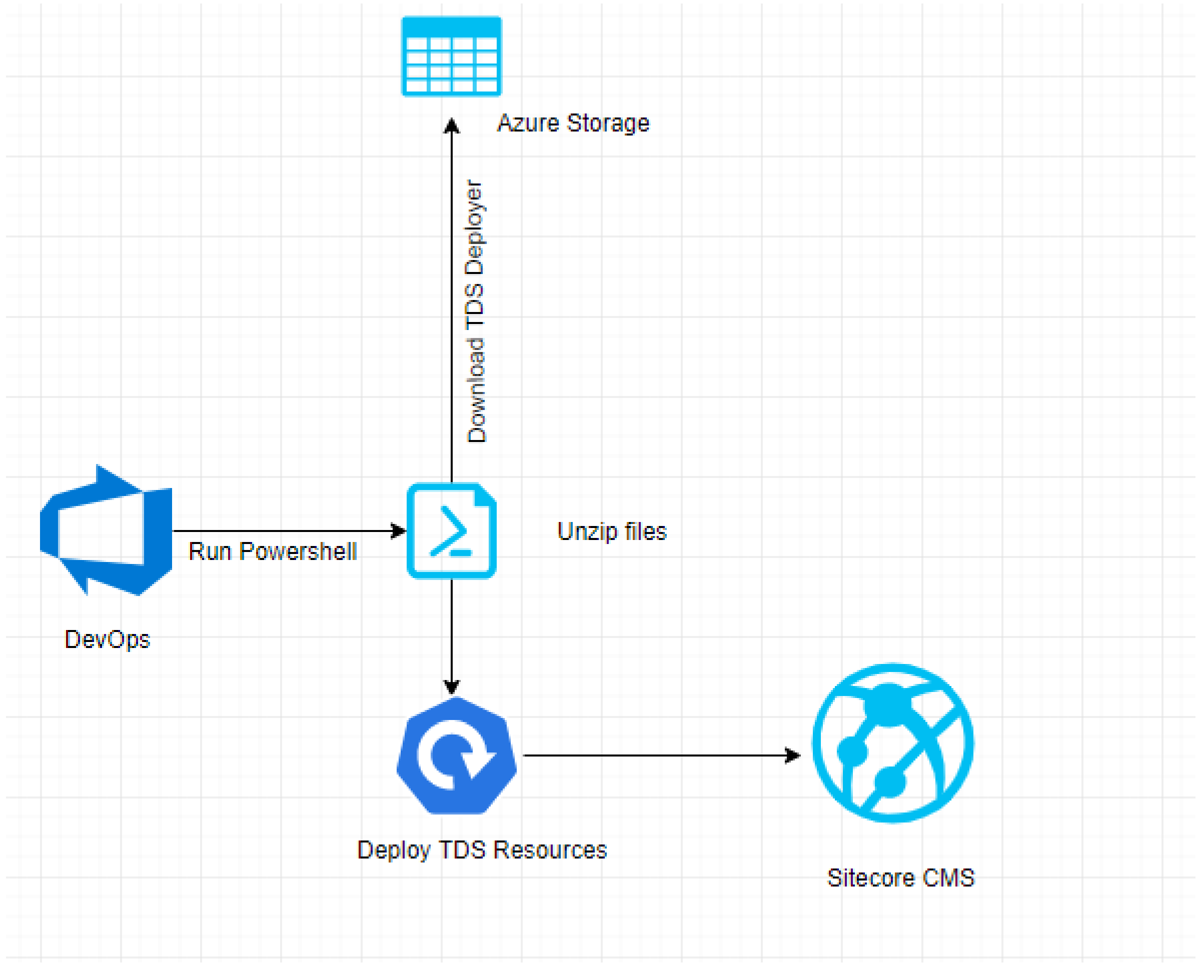
- Upload all Sitecore Web Deployer Packages in Azure Storage
- Upload all Sitecore ARMs in Azure Storage
- Upload a Sitecore License in Azure Storage



```
# Specify the paramet
$ArmTemplateUrl = "htt
$ArmParametersUrl = "h
$licenseFileUrl = "htt

# Specify the certific
$certificateFileUrl =                                                       49839E81A2C0F440A.pfx"
$certificatePassword =
$certificateBlob = $nu

$Name = "$(ResourceGro

$location = "$(ResourceGroupLocation)"
$AzureSubscriptionId = "$(AzureSubscriptionId)"

# read the contents of your Sitecore license file
$licenseDownload = Invoke-WebRequest -Uri $licenseFileUrl
$licenseFileContent = $licenseDownload.Content

# read the contents of your authentication certificate
$certificateDownload = Invoke-WebRequest -Uri $certificateFileUrl
if ($certificateDownload.Content) {...}

# region Create Params Object
# license file needs to be secure string and adding the params as a hashtable is the only way to do it
$additionalParams = New-Object -TypeName Hashtable

$paramsDownload = Invoke-WebRequest -Uri $ArmParametersUrl
$params = [System.IO.StreamReader]::new($paramsDownload.RawContentStream).ReadToEnd() | ConvertFrom-Json
```

```
    Write-Host "Starting ARM deployment..."
    New-AzResourceGroupDeployment `
            -Name $Name `
            -ResourceGroupName $Name `
            -TemplateUri $ArmTemplateUrl `
            -TemplateParameterObject $additionalParams `
#           -DeploymentDebugLogLevel ResponseContent -Debug -Verbose

    Write-Host "Deployment Complete."
    Disconnect-AzAccount
}
```

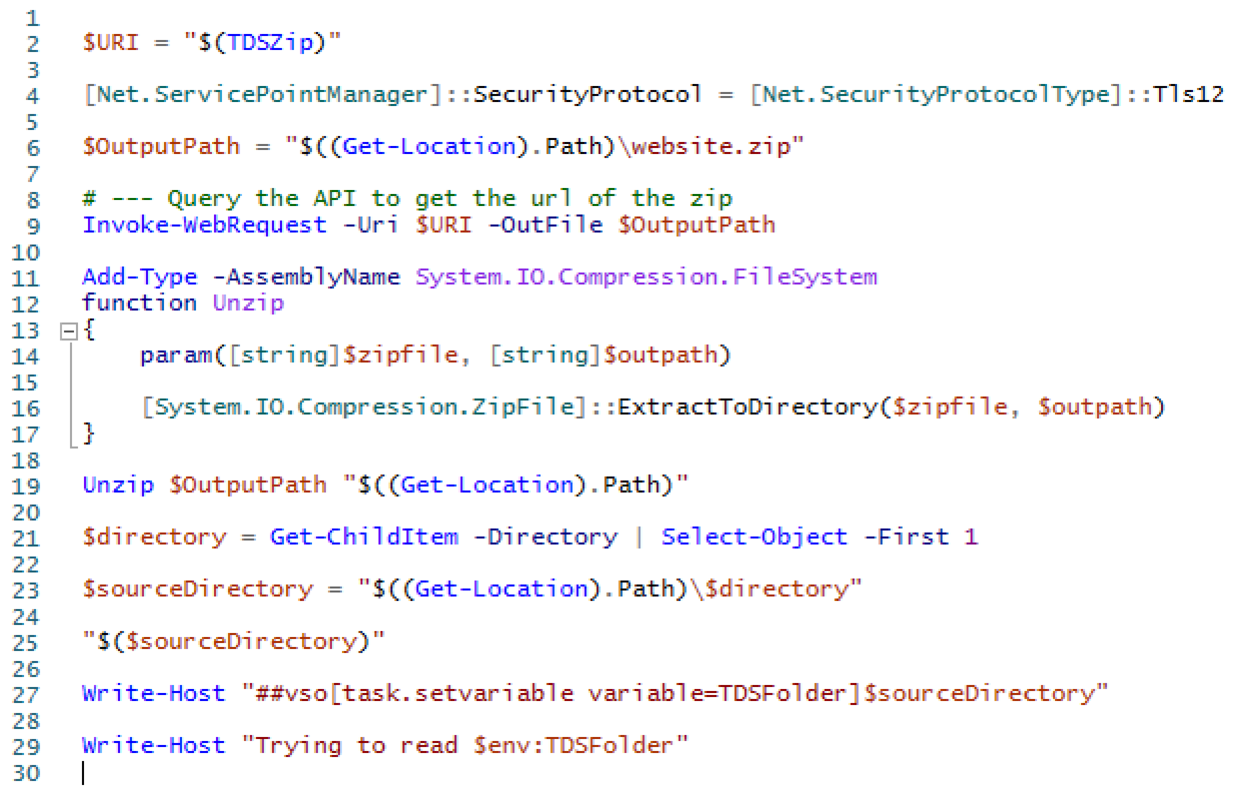
DevOps Install TDS Package Deployer



DevOps Install TDS Package Deployer

Pre-requirements

- Upload all TDS Package Deployer files in Azure Storage



```
$URI = "$(TDSZip)"

[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12

$OutputPath = "$((Get-Location).Path)\website.zip"

# --- Query the API to get the url of the zip
Invoke-WebRequest -Uri $URI -OutFile $OutputPath

Add-Type -AssemblyName System.IO.Compression.FileSystem
function Unzip
{
    param([string]$zipfile, [string]$outpath)

    [System.IO.Compression.ZipFile]::ExtractToDirectory($zipfile, $outpath)
}

Unzip $OutputPath "$((Get-Location).Path)"

$directory = Get-ChildItem -Directory | Select-Object -First 1

$sourceDirectory = "$((Get-Location).Path)\$directory"

"$($sourceDirectory)"

Write-Host "##vso[task.setvariable variable=TDSFolder]$sourceDirectory"

Write-Host "Trying to read $env:TDSFolder"

```

RATIO

Demo time !!!

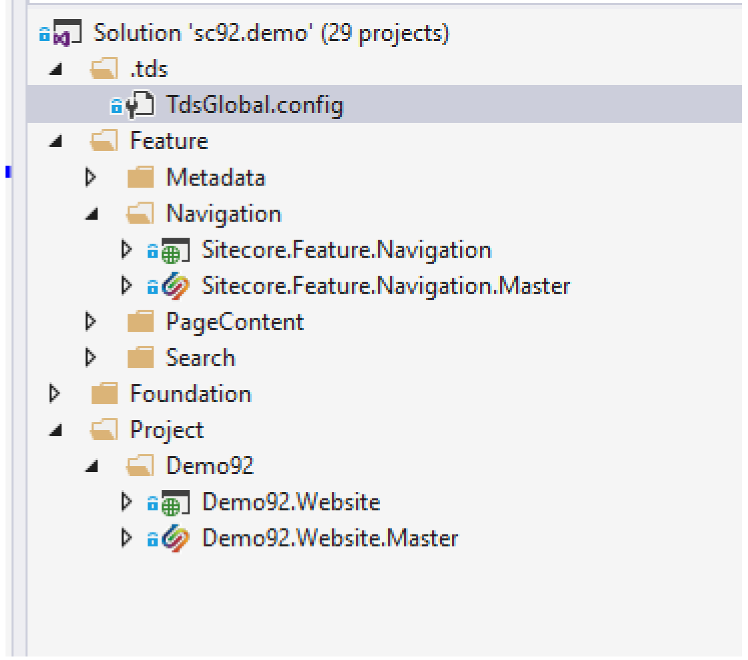
Setup Sitecore Solution with TDS

Using Team Development for Sitecore (TDS), it makes it extremely easy as it will automatically deploy your web project to your Sitecore instance root on build.

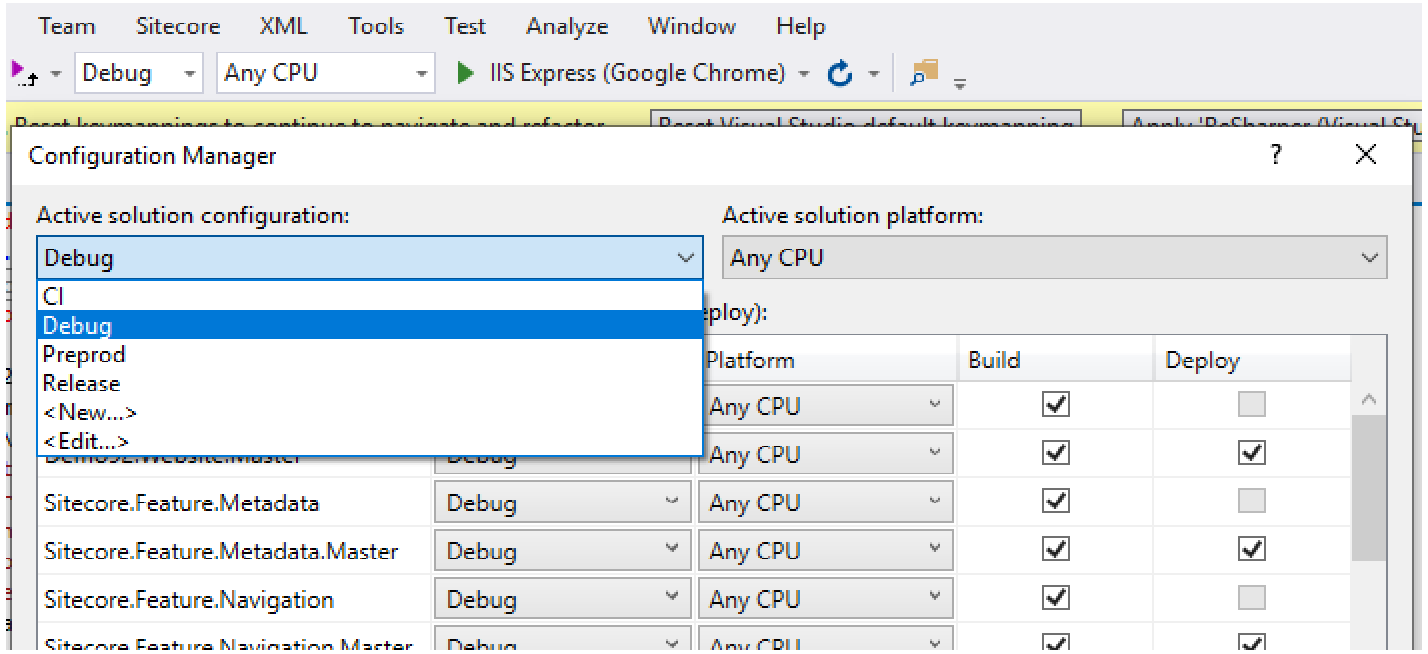
Each project in Visual Studio is either

- a web application project with an associated TDS project – connecting a TDS project to a Web Project that follows Helix architecture will ensure copy of **views**, **bin**, **configs** on **build**.
- code library referenced by another project.

Define multiple solution configurations that allows defining environment specific configurations.



- Solution 'sc92.demo' (29 projects)
  - .tds
    - TdsGlobal.config
  - Feature
    - Metadata
    - Navigation
      - Sitecore.Feature.Navigation
      - Sitecore.Feature.Navigation.Master
    - PageContent
    - Search
  - Foundation
  - Project
    - Demo92
      - Demo92.Website
      - Demo92.Website.Master



Configuration Manager

Active solution configuration: Debug

- CI
- Debug
- Preprod
- Release
- <New...>
- <Edit...>

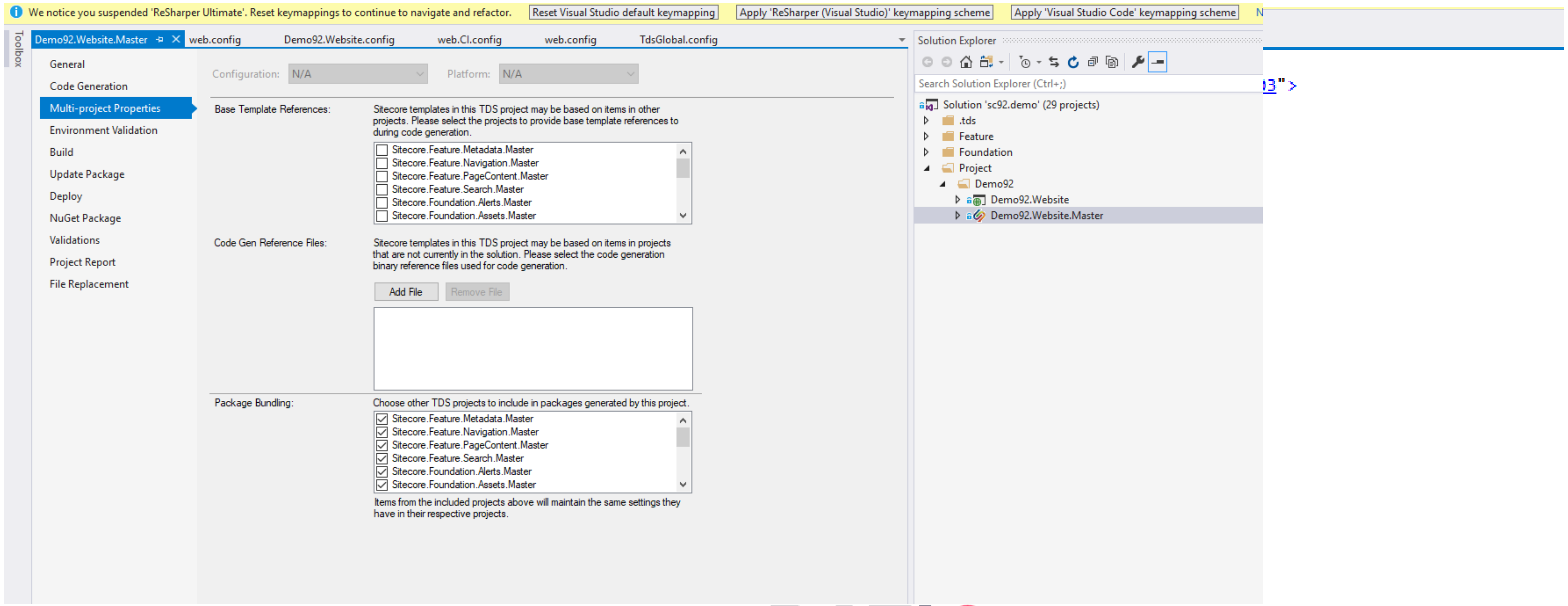
Active solution platform: Any CPU

Project	Configuration	Platform	Build	Deploy
		Any CPU	✓	
		Any CPU	✓	✓
Sitecore.Feature.Metadata	Debug	Any CPU	✓	
Sitecore.Feature.Metadata.Master	Debug	Any CPU	✓	✓
Sitecore.Feature.Navigation	Debug	Any CPU	✓	
Sitecore.Feature.Navigation.Master	Debug	Any CPU	✓	✓

Setup Sitecore Solution For Multi-Environments Build

Adding TDS Global config to a solution you can control project properties across multiple Sitecore TDS projects.
For CI configuration, set up TDS to deploy all custom files to a folder **_Deploy** and to generate a **TDS package** without code files.

With the **Package Bundling** option we can generate a single TDS package with items pulled from all TDS projects.



Configuration: N/A Platform: N/A

Base Template References: Sitecore templates in this TDS project may be based on items in other projects. Please select the projects to provide base template references to during code generation.

- ☐ Sitecore.Feature.Metadata.Master
- ☐ Sitecore.Feature.Navigation.Master
- ☐ Sitecore.Feature.PageContent.Master
- ☐ Sitecore.Feature.Search.Master
- ☐ Sitecore.Foundation.Alerts.Master
- ☐ Sitecore.Foundation.Assets.Master

Code Gen Reference Files: Sitecore templates in this TDS project may be based on items in projects that are not currently in the solution. Please select the code generation binary reference files used for code generation.

Package Bundling: Choose other TDS projects to include in packages generated by this project.

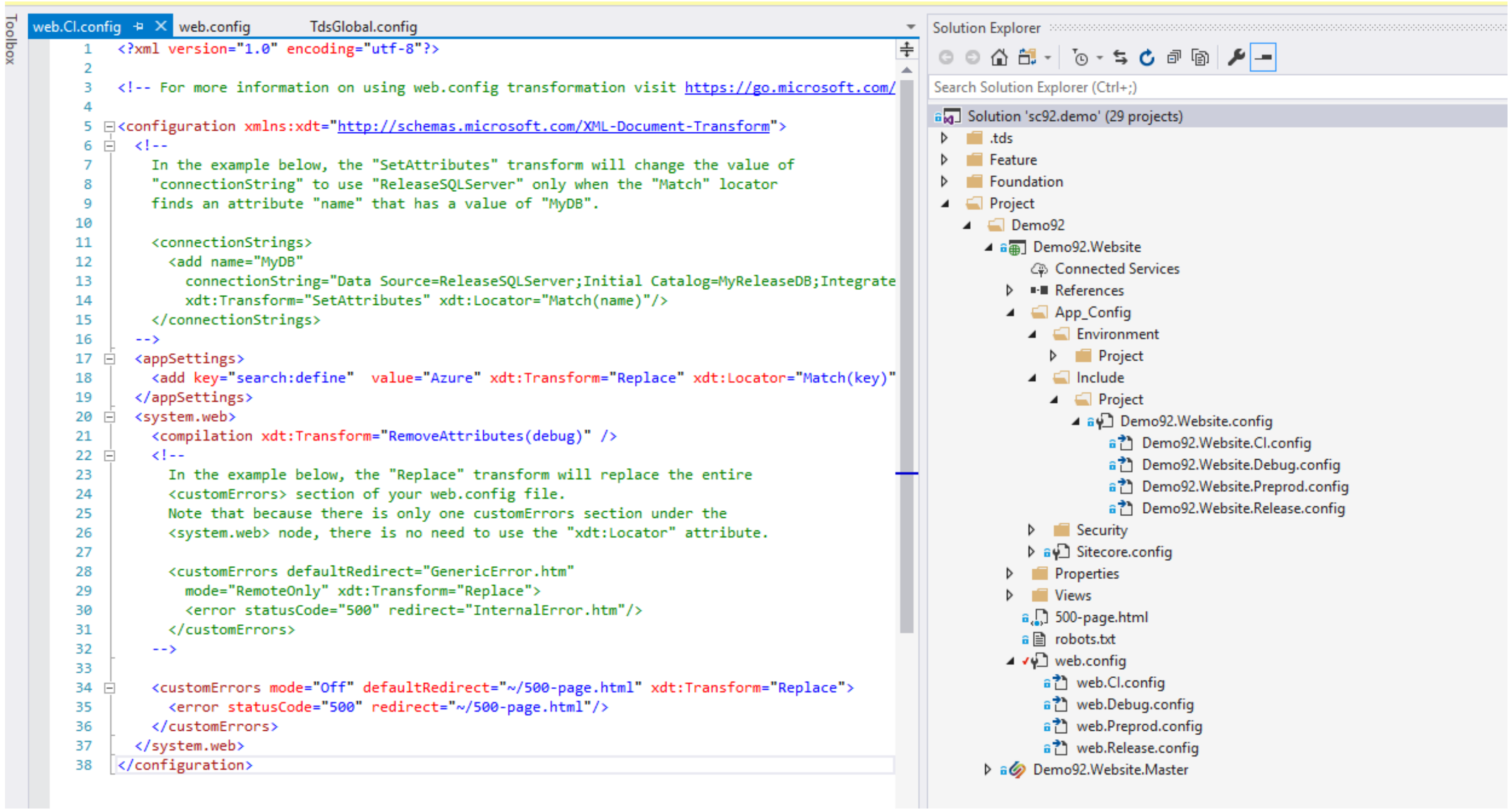
- ☑ Sitecore.Feature.Metadata.Master
- ☑ Sitecore.Feature.Navigation.Master
- ☑ Sitecore.Feature.PageContent.Master
- ☑ Sitecore.Feature.Search.Master
- ☑ Sitecore.Foundation.Alerts.Master
- ☑ Sitecore.Foundation.Assets.Master

Items from the included projects above will maintain the same settings they have in their respective projects.

Setup Sitecore Solution For Multi-Environments Build

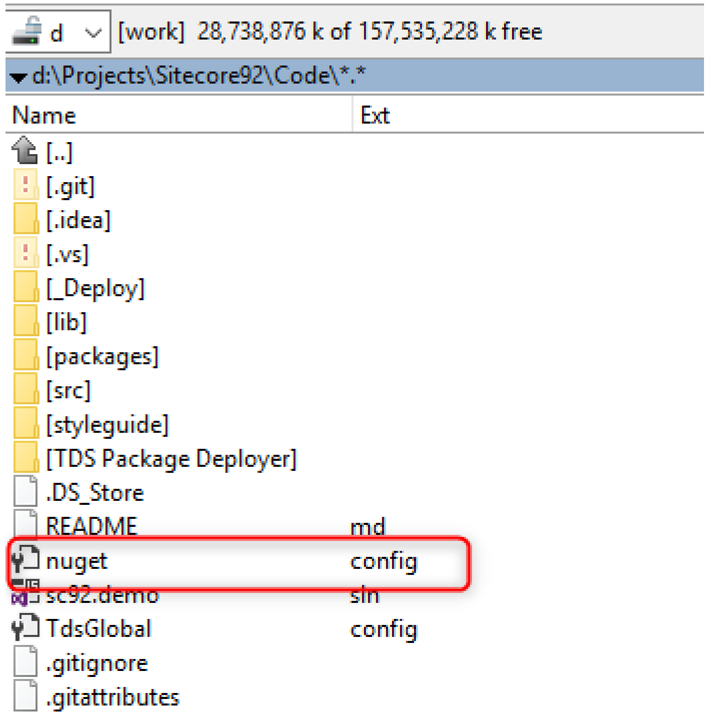
Use **config transformations** for environment specific configurations. Use visual studio Add Config Transform extension to generate configuration configs.

Define **nuget.config** in the solution root to define Sitecore nuget feeds and other nuget feeds – will be used to restore packages on CI build.



```xml
<?xml version="1.0" encoding="utf-8"?>

<!-- For more information on using web.config transformation visit https://go.microsoft.com/

<configuration xmlns:xdt="http://schemas.microsoft.com/XML-Document-Transform">
  <!--
    In the example below, the "SetAttributes" transform will change the value of
    "connectionString" to use "ReleaseSQLServer" only when the "Match" locator
    finds an attribute "name" that has a value of "MyDB".

    <connectionStrings>
      <add name="MyDB"
        connectionString="Data Source=ReleaseSQLServer;Initial Catalog=MyReleaseDB;Integrate
        xdt:Transform="SetAttributes" xdt:Locator="Match(name)"/>
    </connectionStrings>
  -->
  <appSettings>
    <add key="search:define"  value="Azure" xdt:Transform="Replace" xdt:Locator="Match(key)"
  </appSettings>
  <system.web>
    <compilation xdt:Transform="RemoveAttributes(debug)" />
    <!--
      In the example below, the "Replace" transform will replace the entire
      <customErrors> section of your web.config file.
      Note that because there is only one customErrors section under the
      <system.web> node, there is no need to use the "xdt:Locator" attribute.

      <customErrors defaultRedirect="GenericError.htm"
        mode="RemoteOnly" xdt:Transform="Replace">
        <error statusCode="500" redirect="InternalError.htm"/>
      </customErrors>
    -->

    <customErrors mode="Off" defaultRedirect="~/500-page.html" xdt:Transform="Replace">
      <error statusCode="500" redirect="~/500-page.html"/>
    </customErrors>
  </system.web>
</configuration>
```

```
Solution 'sc92.demo' (29 projects)
  .tds
  Feature
  Foundation
  Project
    Demo92
      Demo92.Website
        Connected Services
        References
        App_Config
          Environment
            Project
          Include
            Project
              Demo92.Website.config
                Demo92.Website.CI.config
                Demo92.Website.Debug.config
                Demo92.Website.Preprod.config
                Demo92.Website.Release.config
          Security
          Sitecore.config
        Properties
        Views
        500-page.html
        robots.txt
        web.config
          web.CI.config
          web.Debug.config
          web.Preprod.config
          web.Release.config
      Demo92.Website.Master
```



d [work] 28,738,876 k of 157,535,228 k free

d:\Projects\Sitecore92\Code*.*

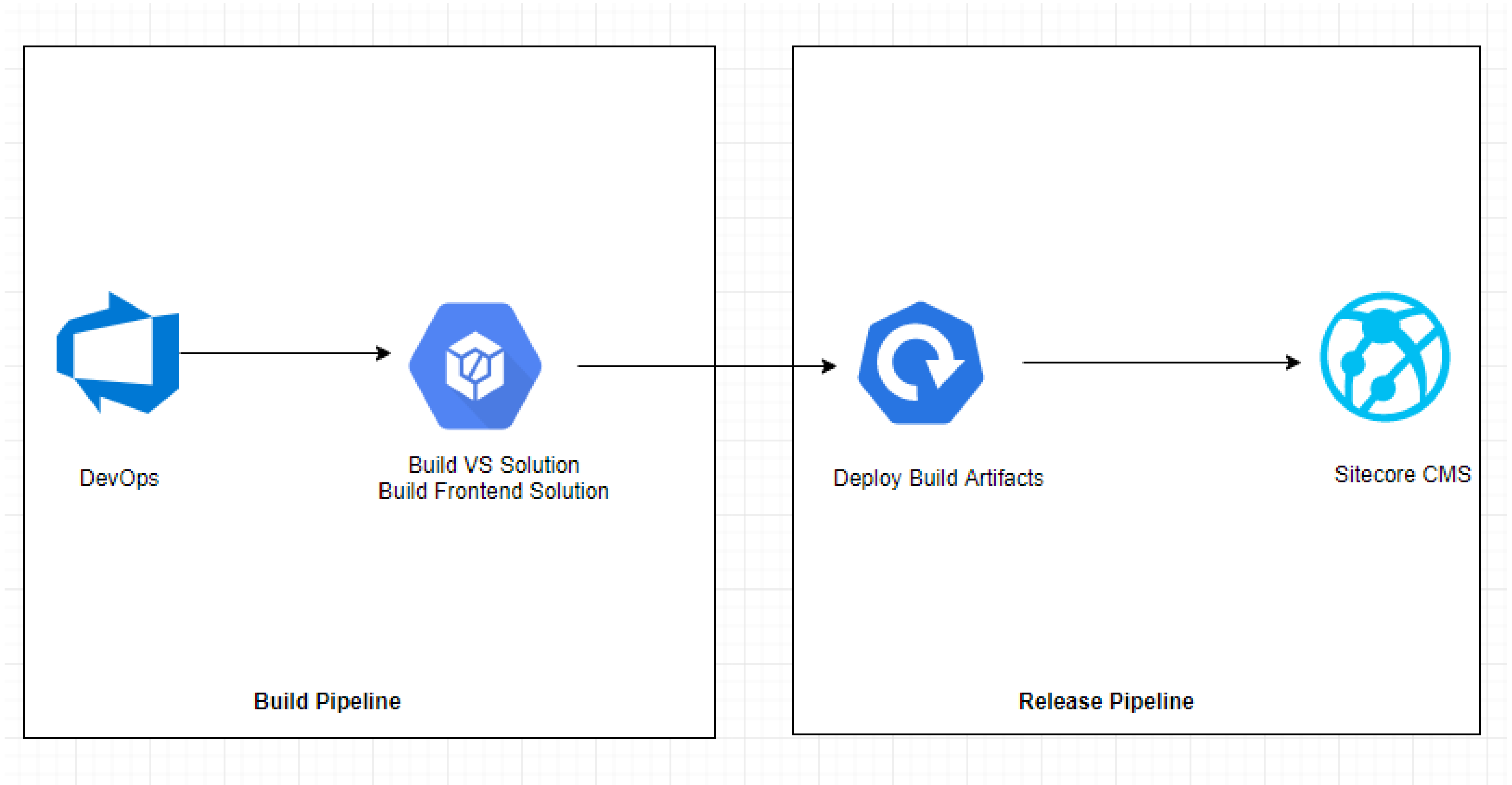
Name	Ext
[..]	
[.git]	
[.idea]	
[.vs]	
[_Deploy]	
[lib]	
[packages]	
[src]	
[styleguide]	
[TDS Package Deployer]	
.DS_Store	
README	md
nuget	config
sc92.demo	sln
TdsGlobal	config
.gitignore	
.gitattributes	

Demo time !!!

DevOps Pipelines

To deploy our custom code, we will need to:

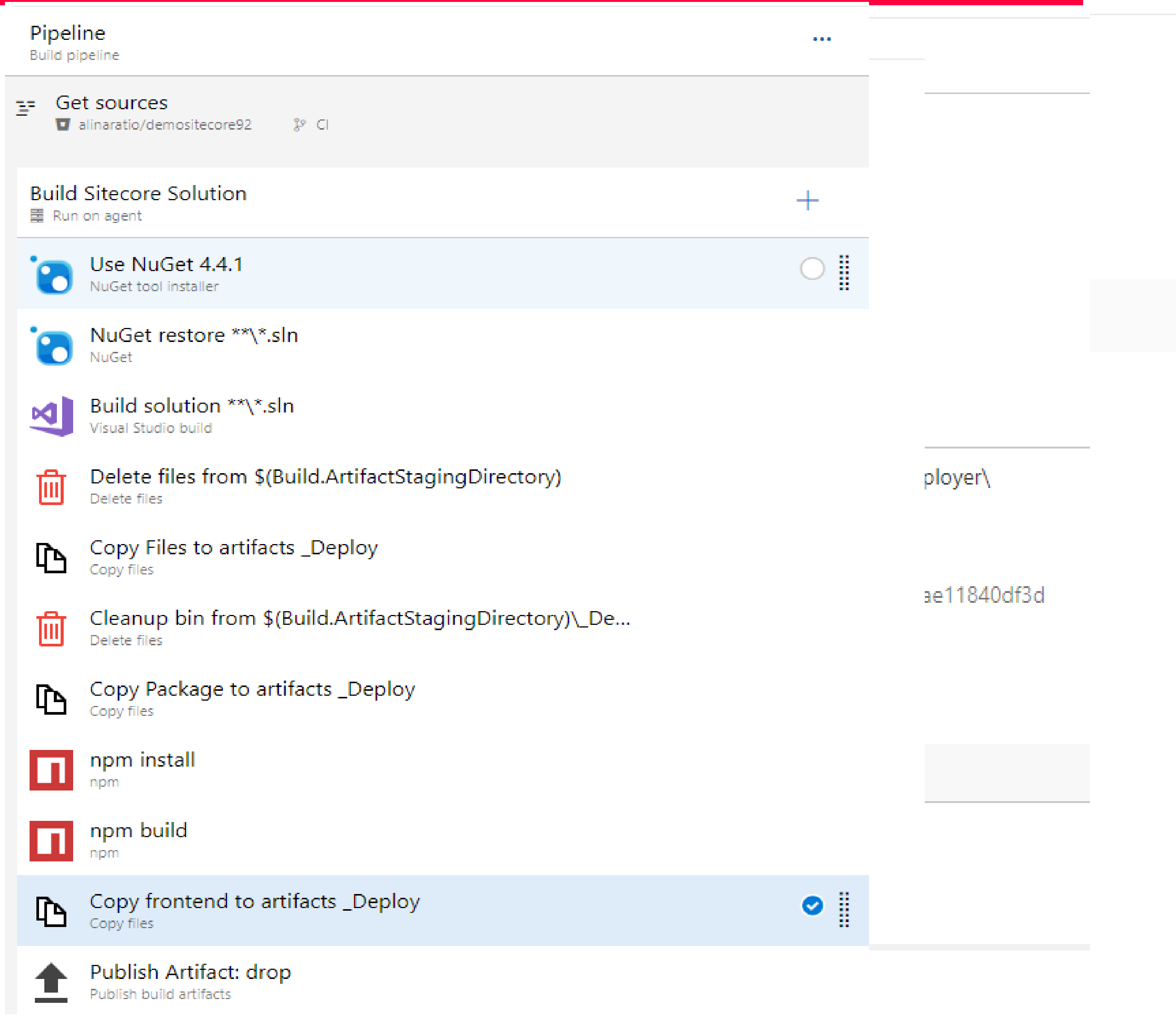
- Generate the folder _Deploy and TDS update package.
- Copy _Deploy folder content under Sitecore 9.2 website root. Sitecore 9.2 is an Azure Web App.
- Install update package using TDS package deployer.



DevOps Build Pipeline

To set up the Build Pipeline specific to **Sitecore**:

- Select Repository & branch
- Define Build variables that can be used in Task definition
- Define if Build will be triggered on each commit
- Define Build tasks



Pipeline
Build pipeline

Get sources
alinaratio/demositecore92 CI

Build Sitecore Solution
Run on agent

Use NuGet 4.4.1
NuGet tool installer

NuGet restore ***.sln
NuGet

Build solution ***.sln
Visual Studio build

Delete files from $(Build.ArtifactStagingDirectory)
Delete files

Copy Files to artifacts _Deploy
Copy files

Cleanup bin from $(Build.ArtifactStagingDirectory)_De...
Delete files

Copy Package to artifacts _Deploy
Copy files

npm install
npm

npm build
npm

Copy frontend to artifacts _Deploy
Copy files

Publish Artifact: drop
Publish build artifacts

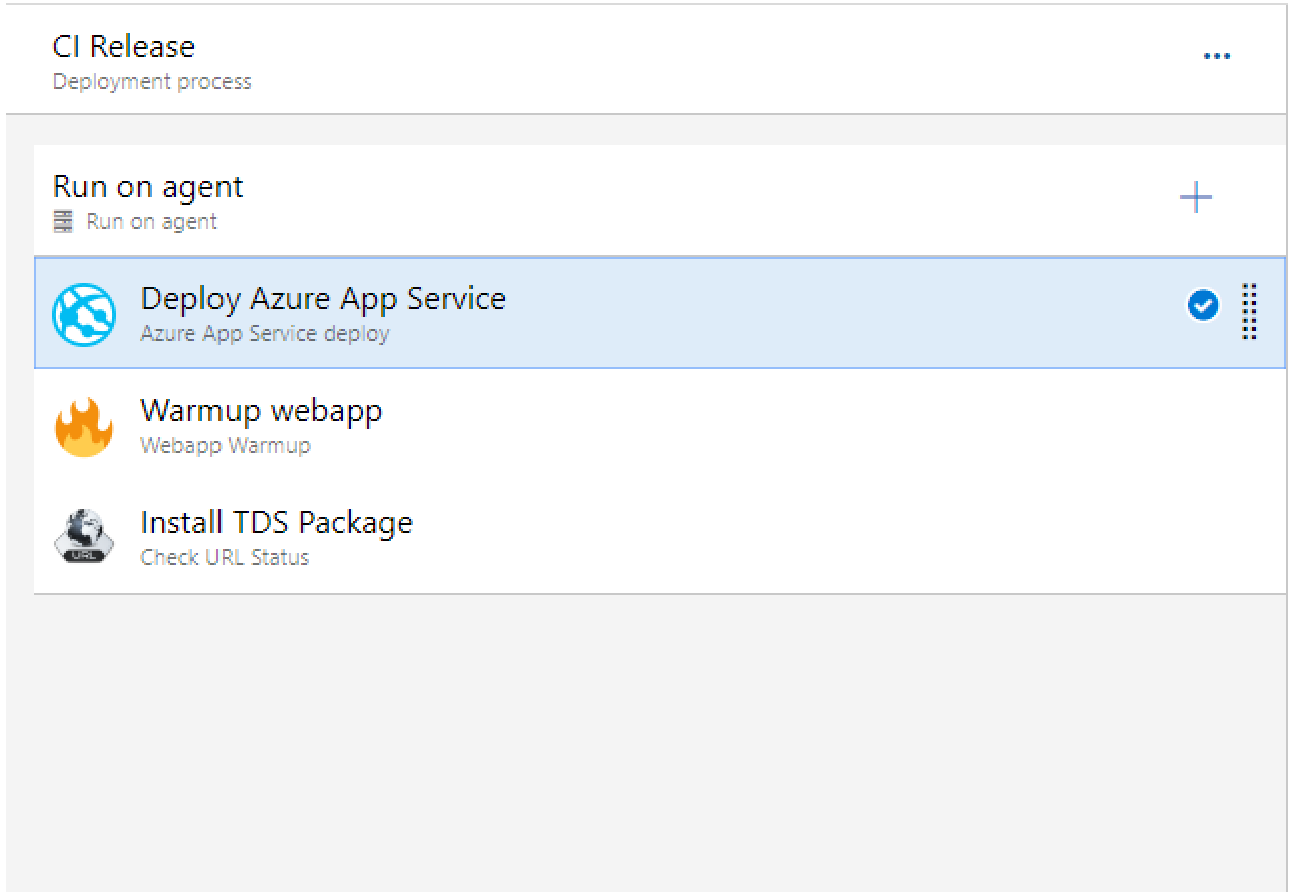
ployer\

ae11840df3d

DevOps Release Pipeline

To set up the Release Pipeline specific to **Azure Web Apps**:

- Connect Build to Release pipeline
- Set up if Release is automatically created on successful build
- Define Release tasks



CI Release
Deployment process

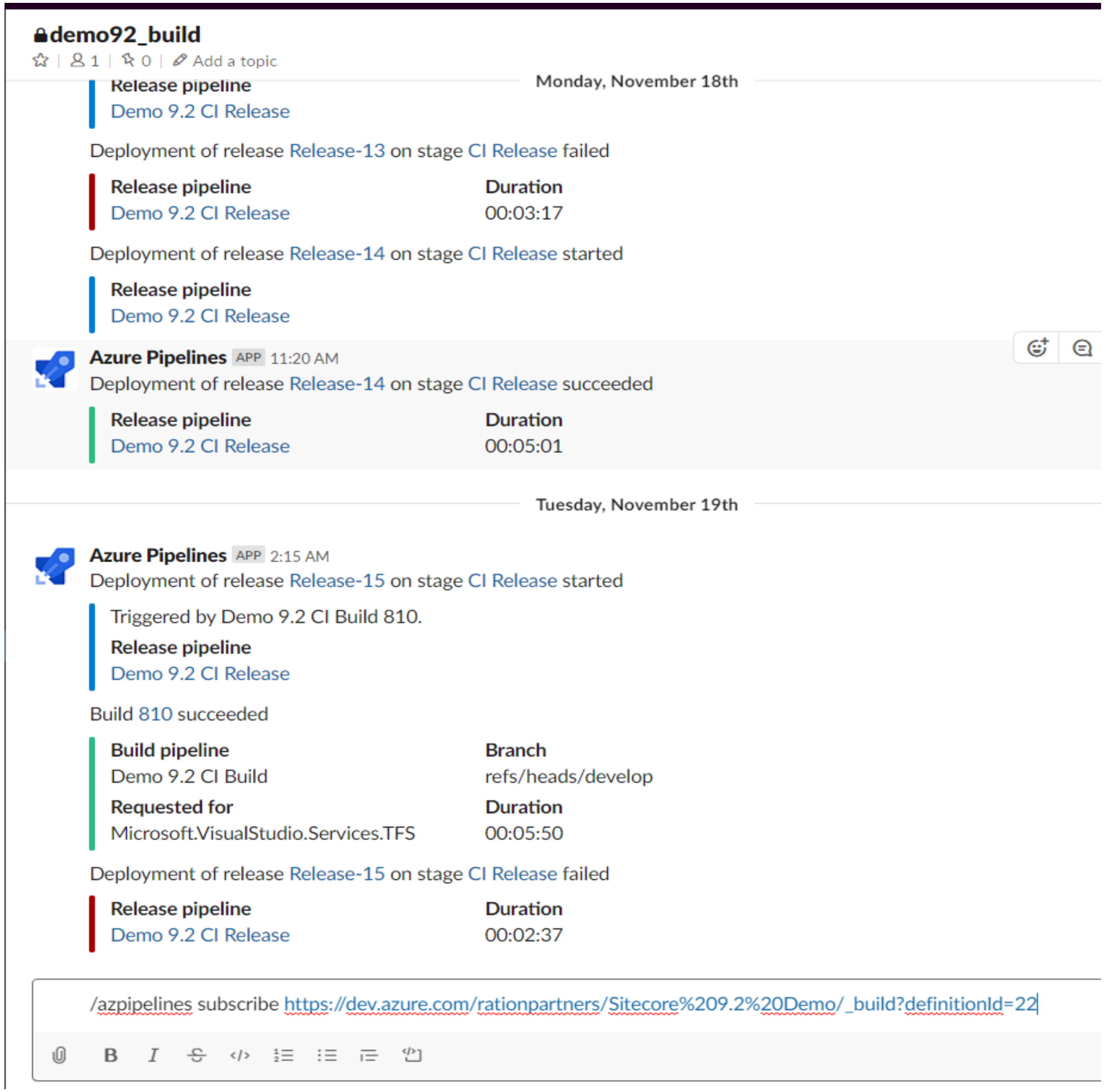
Run on agent
Run on agent

Deploy Azure App Service
Azure App Service deploy

Warmup webapp
Webapp Warmup

Install TDS Package
Check URL Status

Monitor your builds and releases on Slack



demo92_build

1 | 0 | Add a topic

Monday, November 18th

Release pipeline
Demo 9.2 CI Release

Deployment of release Release-13 on stage CI Release failed

Release pipeline	Duration
Demo 9.2 CI Release	00:03:17

Deployment of release Release-14 on stage CI Release started

Release pipeline
Demo 9.2 CI Release

Azure Pipelines APP 11:20 AM
Deployment of release Release-14 on stage CI Release succeeded

Release pipeline	Duration
Demo 9.2 CI Release	00:05:01

Tuesday, November 19th

Azure Pipelines APP 2:15 AM
Deployment of release Release-15 on stage CI Release started

Triggered by Demo 9.2 CI Build 810.
Release pipeline
Demo 9.2 CI Release

Build 810 succeeded

Build pipeline	Branch
Demo 9.2 CI Build	refs/heads/develop
Requested for	**Duration**
Microsoft.VisualStudio.Services.TFS	00:05:50

Deployment of release Release-15 on stage CI Release failed

Release pipeline	Duration
Demo 9.2 CI Release	00:02:37

```
/azpipelines subscribe https://dev.azure.com/rationpartners/Sitecore%209.2%20Demo/_build?definitionId=22
```

Demo time !!!

RATIO

That’s all Folks! Thank you!