Working with Containerization using Docker

- Understanding VM and Containers
- What is Docker and its Benefits
- Docker Architecture
- Steps to Create Docker Image
- Build and Publish Docker Image to Docker Hub using Azure Pipeline
- Build and Publish Docker Image to Azure Container Registry using Azure Pipeline
- Deploying to Web App
- Deploying a Docker Container in VM or Local Machine using Docker Compose

Create an Image Manually

Step 1: Update the Code

1. Create a New Project – HelloWebApp.Web

   dotnet new mvc -n HelloWebApp.Web

2. Add dockerfile as below Dockerfile

```
FROM mcr.microsoft.com/dotnet/core/aspnet:3.1 AS base
WORKDIR /app
EXPOSE 80

FROM mcr.microsoft.com/dotnet/core/sdk:3.1 AS build
WORKDIR /src
COPY ["HelloWorldApp.Web/HelloWorldApp.Web.csproj", "HelloWorldApp.Web/"]
RUN dotnet restore "HelloWorldApp.Web/HelloWorldApp.Web.csproj"
COPY . .
WORKDIR "/src/HelloWorldApp.Web"
RUN dotnet build "HelloWorldApp.Web.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "HelloWorldApp.Web.csproj" -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "HelloWorldApp.Web.dll"]
```

3. Build images

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA. http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

docker build -t sandeepsoni/hellowebapp:v1 .

4. Run the docker image locally

docker run --rm -p 8080:80 sandeepsoni/hellowebapp:v1

What is Multistage Build

- Multi-stage builds are a new feature requiring Docker 17.05 or higher on the daemon and client.
- Before the multi-stage build feature was added, a script was required to copy the published output of your build container onto your disk and then the runtime container would read in that output. This was tedious to implement and not super-efficient. Now, containers can share build artifacts from different stages within a single Dockerfile.
- Using this feature, you can build a .NET Core app in an SDK image and then copy the published app into a runtime image all in the same Dockerfile.

Stop at a specific build stage:

When you build your image, you don't necessarily need to build the entire Dockerfile including every stage. You can specify a target build stage.

The following command assumes you are using the previous Dockerfile but stops at the stage named build-env:

$ docker build **--target build-env** -t sandeepsoni/hello-world:**dev** .

$ docker build -t sandeepsoni/hello-world:**prod .**

Azure Pipeline for Build and Publish Docker Image to Docker Hub

Azure Pipelines can be used to build images for any repository containing a Dockerfile. Building of both Linux and Windows containers is possible based on the agent platform used for the build.

Create a New DevOps Project and Commit the Code.

Create a New Service Connection

Organization Properties → Service Connection → **New Service Connection → Docker Registry** → Select **Docker Hub** radio button, provide your Docker Hub Username and Password → Verify

Create a New Pipeline

Project → Pipelines → New Pipeline → Azure Repos → Docker

Edit the YAML as below

```
trigger:
- none
```


Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA. http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

```
resources:
- repo: self

variables:
  tag: 'ver1'
  dockerHubId: 'sandeepsoni'
  imageName: "helloworldapp.web"
  dockerHub: "Docker Hub"

stages:
- stage: Build
  displayName: Build image
  jobs:
  - job: Build
    displayName: Build
    pool:
      vmImage: 'ubuntu-latest'
    steps:
    - task: Docker@2
      displayName: "Build an image"
      inputs:
        command: 'build'
        Dockerfile: '**/Dockerfile'
        containerRegistry: $(dockerHub)
        repository: '$(dockerHubId)/$(imageName)'
        tags: '$(Build.BuildId)'

    - task: Docker@2
      displayName: "Push image"
      inputs:
        command: push
        containerRegistry: $(dockerHub)
        repository: $(dockerHubId)/$(imageName)
        tags: '$(Build.BuildId)'
```

Azure Container Registry

- **Azure Container is a private registry allows you to store and manage docker container images**
- Use container registries in Azure with your existing container development and deployment pipelines.

Use Azure Container Registry to:

1. Store and manage container images across all types of Azure deployments
2. Use familiar, open-source Docker command line interface (CLI) tools
3. Keep container images near deployments to reduce latency and costs
4. Simplify registry access management with Azure Active Directory
5. Maintain Windows and Linux container images in a single Docker registry

Create Container Registry Using Portal

1. **Create a resource → Containers → Azure Container Registry.**
2. Under **Admin user**, select **Enable**. Take note of the following values:
    - Login server
    - Username
    - password
3. Login to ACR

    docker login --**username** dssdemo --**password** X3bI/uVbrNJ8lgfLXqjDV4zQVWRJgOI1 **dssdemo.azurecr.io**
4. Tag Local Images

    docker image **tag** sandeepsoni/hellowebapp:v1 dssdemo.azurecr.io/hellowebapp:v1
5. Push images to ACR

    docker **push** dssdemo.azurecr.io/hellowebapp:v1

Create a New Pipeline

1. Project → Pipelines → New Pipeline → Azure Repos → Docker (Build and push an image to Azure container registry)
2. Select Your Azure Subscription → Continue → Login with Azure Credentails.

Note: we can as well create a Azure Container Registry Connection using **Docker Template as done for Docker Hub**

3. Next Dialog: Select ACR Name, ImageName and Dockerfile path → **Validate and configure**

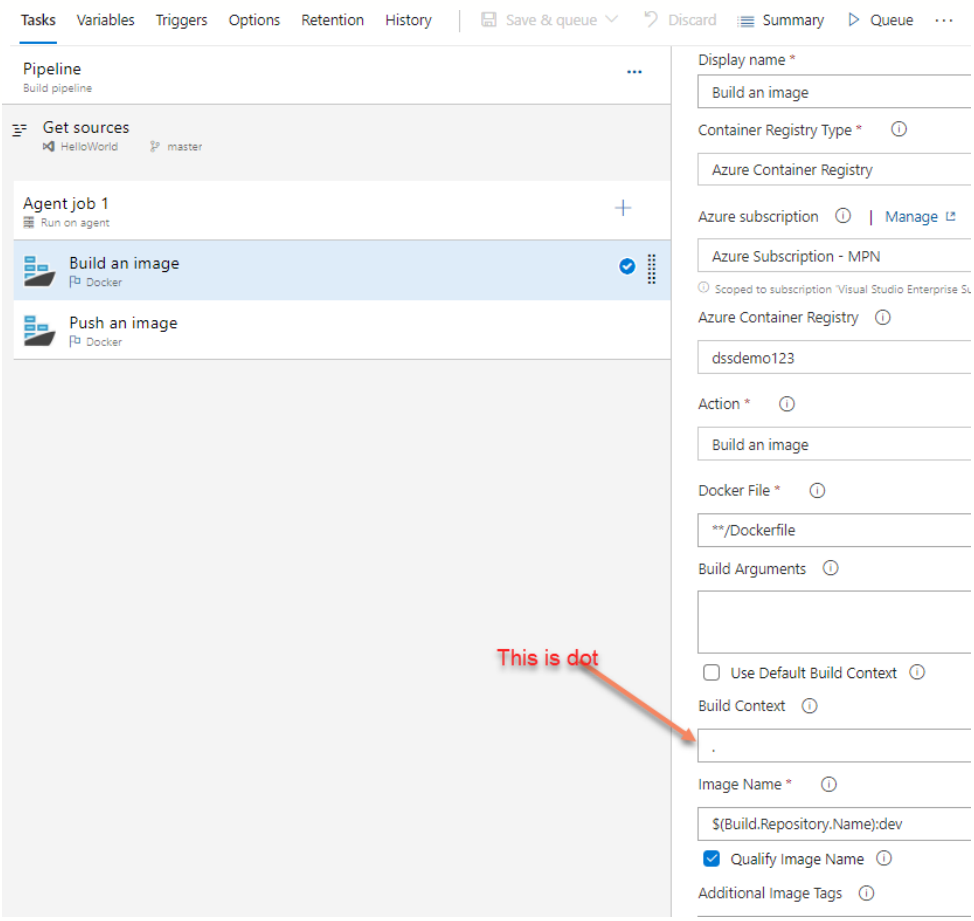
Note: Azure Pipelines created a pipeline for you, using the *Docker container template*.

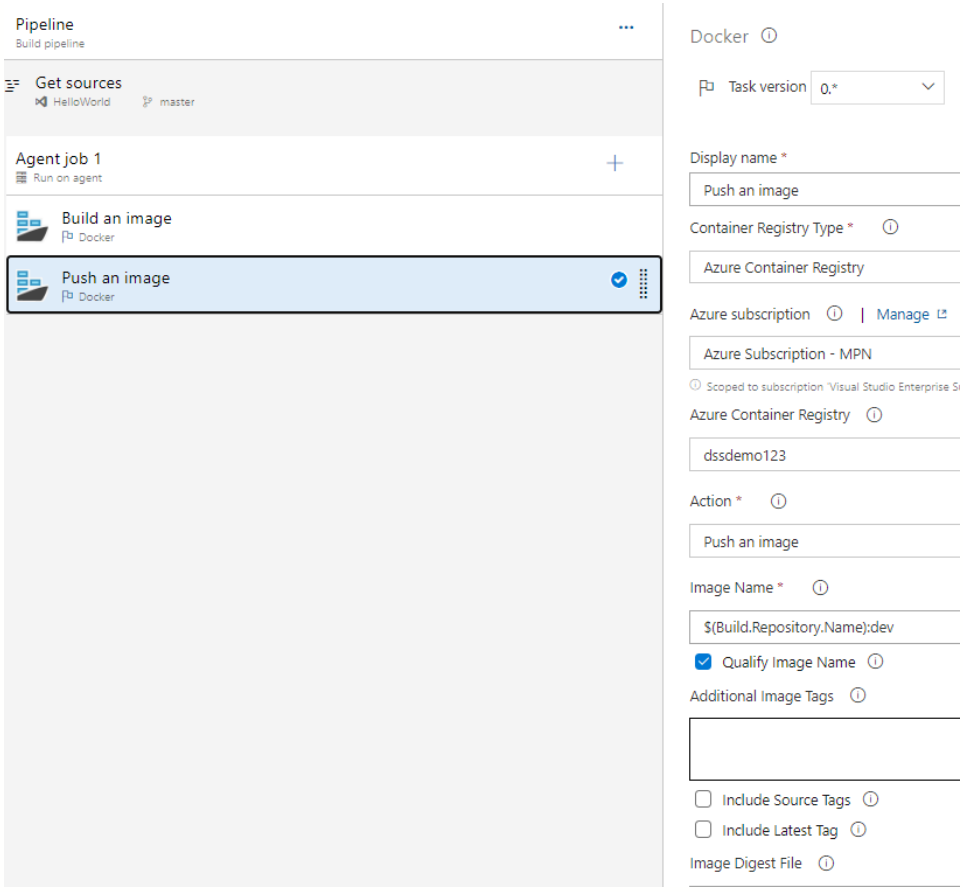
Azure-pipeline.yml

```
# Docker
# Build and push an image to Azure Container Registry
# https://docs.microsoft.com/azure/devops/pipelines/languages/docker
```


Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA. http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

```
trigger:
- master

resources:
- repo: self

variables:
  # Container registry service connection established during pipeline creation
  dockerRegistryServiceConnection: '487ebce8-02ce-48f8-b549-60b2b3aa6e12'
  imageRepository: 'helloworldapp.web'
  containerRegistry: 'dssdemo.azurecr.io'
  dockerfilePath: '$(Build.SourcesDirectory)/HelloWorldApp.Web/dockerfile'
  tag: '$(Build.BuildId)'

  # Agent VM image name
  vmImageName: 'ubuntu-latest'

stages:
- stage: Build
  displayName: Build and push stage
  jobs:
  - job: Build
    displayName: Build
    pool:
      vmImage: $(vmImageName)
    steps:
    - task: Docker@2
      displayName: Build and push an image to container registry
      inputs:
        command: buildAndPush
        repository: $(imageRepository)
        dockerfile: $(dockerfilePath)
        containerRegistry: $(dockerRegistryServiceConnection)
        tags: |
          $(tag)
```

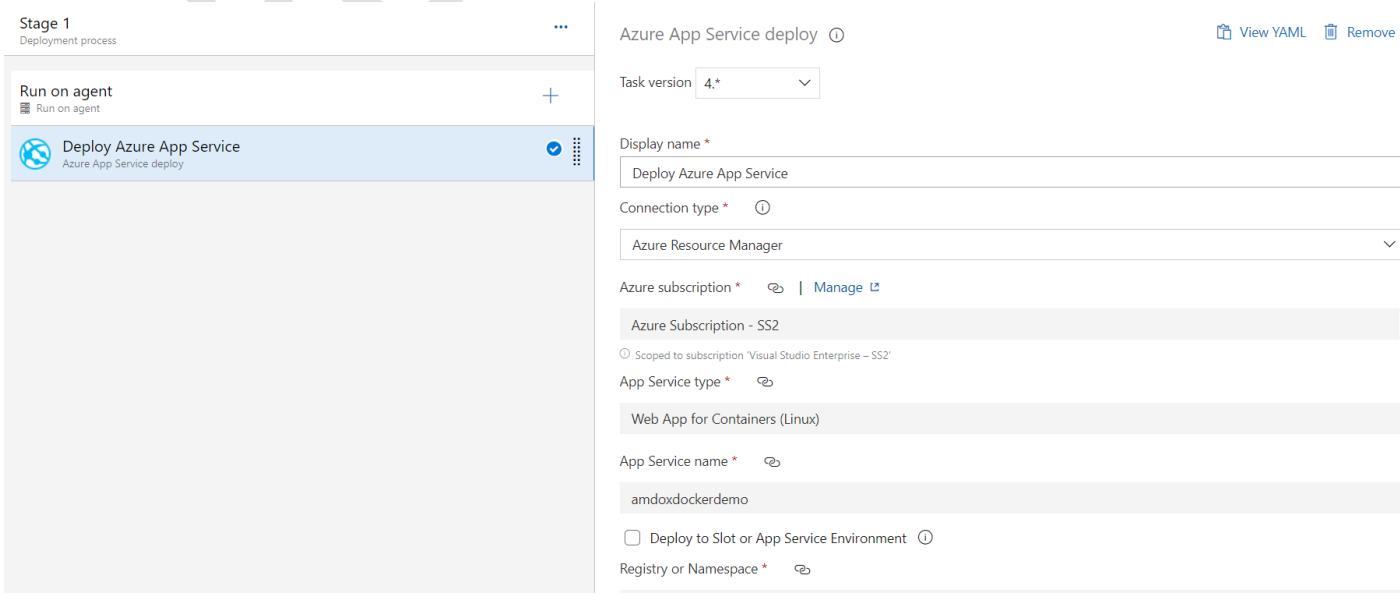

Tasks
Variables
Triggers
Options
Retention
History
Save & queue
Discard
Summary
Queue
Pipeline
Build pipeline
Get sources
HelloWorld
master
Agent job 1
Run on agent
Build an image
Docker
Push an image
Docker
Display name *
Build an image
Container Registry Type *
Azure Container Registry
Azure subscription
Manage
Azure Subscription - MPN
Azure Container Registry
dssdemo123
Action *
Build an image
Docker File *
**/Dockerfile
Build Arguments
This is dot
Use Default Build Context
Build Context
.
Image Name *
$(Build.Repository.Name):dev
Qualify Image Name
Additional Image Tags

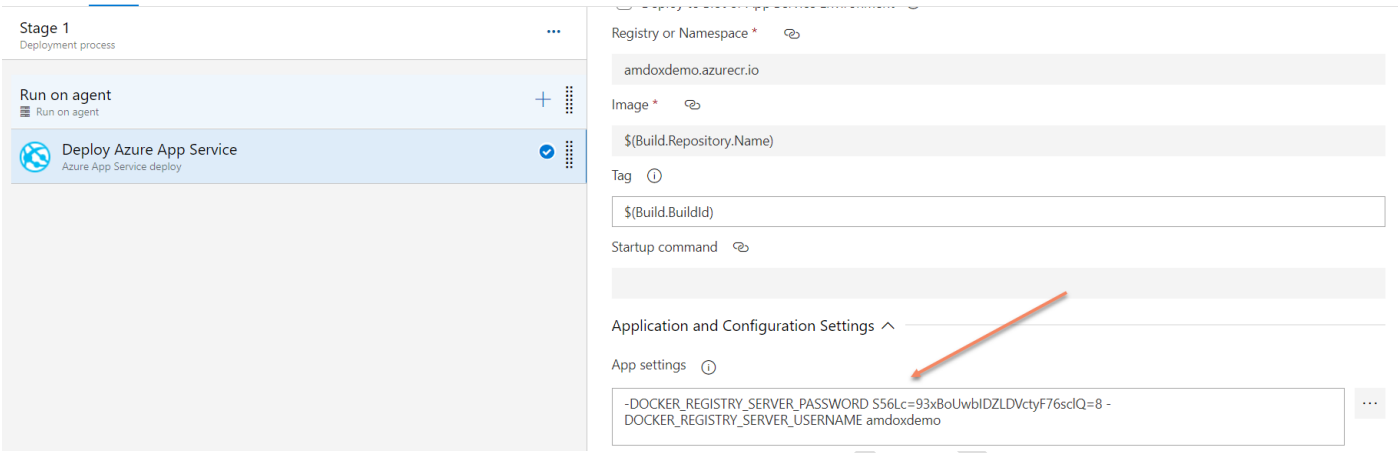


Deploying to Web App

Following Properties must be set in AppSettings of Web App:

-DOCKER_REGISTRY_SERVER_PASSWORD S56Lc=93xBoUwbIDZLDVctyF76sclQ=8 -DOCKER_REGISTRY_SERVER_USERNAME amdoxdemo





You can automatically deploy your application to an Azure Web App for Linux Containers after every successful build.

You must supply an Azure service connection to the AzureWebAppContainer task. Add the following YAML snippet to your existing **azure-pipelines.yaml** file. Make sure you add the service connection details in the variables section as shown below-

```
variables:
  ## Add this under variables section in the pipeline
   azureSubscription: <Name of the Azure subscription>
   appName: <Name of the Web App>
   containerRegistry: <Name of the Azure container registry>

## Add the below snippet at the end of your pipeline
- task: AzureWebAppContainer@1
  displayName: 'Azure Web App on Container Deploy'
  inputs:
   azureSubscription: $(azureSubscription)
   appName: $(appName)
   containers: $(containerRegistry)/$(imageRepository):$(tag)
```

Note: Azure Pipe doesn't push image to App Service, it only sets the property…

Note: Once the image is deployed to AppService, goto to Azure Portal → App Service → Select the App Service → Container Settings → Set Full Image Name and Tag → Save

Running in Any VM or Local Machine using Docker Compose

- A command-line tool and YAML file format with metadata for defining and running multi-container applications.
- You define a single application based on multiple images with one or more .yml files that can override values depending on the environment.

- After you have created the definitions, you can deploy the entire multi-container application by using a single command (docker-compose up) that creates a container per image on the Docker host.

1. Create a File: docker-compose.yml

```
version: '3.4'
services:
 hellowebapp:
  image: sandeepsoni/helloworldapp
  ports:
  - '8080:80'
```

2. Execute the following command to execute the application

```
docker-compose up
```