SITECORE

Data Exchange Framework 6.0 container deployment guide

A guide to deploying Sitecore Data Exchange Framework to Docker or Azure Kubernetes Services

Table of Contents

© Copyright 2021, Sitecore® - all rights reserved.

1. Introduction to Data Exchange Framework for containers

Sitecore Data Exchange Framework enables you to synchronize data between Salesforce and third-party systems.

This guide shows you how to add the Data Exchange Framework to Sitecore container installations for Docker and Azure Kubernetes Service.

© Copyright 2021, Sitecore® - all rights reserved.

2. Prepare to deploy DEF to Sitecore containers

This section explains what you need to prepare for deploying the Sitecore Data Exchange Framework (DEF) to Sitecore containers for Docker and Azure Kubernetes Service (AKS).

2.1. Requirements

Before you deploy DEF to Docker or AKS, the following requirements must be met:

- Docker Desktop must be installed and running. For instructions on how to set up the Docker environment, see the Containers in Sitecore development documentation.
- If the installation is done on Docker, you must have the Sitecore Docker container files deployed on a local machine. For instructions on how to prepare the Sitecore containers, see the *Installation Guide for Developer Workstation with Containers* on the Sitecore download site.
- If the installation is done on Kubernetes, you must the Sitecore AKS container files deployed on a local machine. For instructions on how to prepare a Sitecore environment with Kubernetes, see the *Installation Guide for Production Environment with Kubernetes* on the Sitecore download site.

© Copyright 2021, Sitecore® - all rights reserved.

3. Add Data Exchange Framework module to Sitecore in Docker

To add the Sitecore Data Exchange Framework (DEF) module in Docker, you must do the following in this order:

- Prepare the installation files
- Build the Docker images
- Update the Solr indexes

3.1. Prepare the installation files

To prepare the files you need for the installation:

1. Download the DEF container deployment package from the Sitecore Developer Portal. Extract it to your local workstation with the folder structure intact.
2. Go to the folder that you extracted the DEF container deployment package to. Go to the folder for the Windows version and topology you are using, for example, `DEF.Asset\compose\ltsc2019\xp1`.
3. Open the `.env-example` file in an editor. The file looks, for example, like this:

   ```
   SITECORE_DEV_DOCKER_REGISTRY=ideftdevacr*******/
   TOPOLOGY=xp0
   RUN_UID=ABCD1234
   DEF_IMAGE=scr.sitecore.com/sxp/modules/sitecore-def-xp0-assets:6.0.0.0-2009
   TOOL_IMAGE=scr.sitecore.com/tools/sitecore-docker-tools-assets:10.1.0-1809
   ```

   Copy all the variables in the file to the clipboard.
4. Go to the Sitecore container deployment folder on your local machine. Go to the folder for the Windows version and topology you are using, for example, `compose\ltsc2019\xp1`.
5. Open the `.env` file in an editor, and paste in the variables from the DEF `.env-example` file.
6. From the DEF `compose\<version>\<topology>` folder, copy the `docker-compose.override.yml` file to the Sitecore container deployment `compose\<version>\<topology>` folder (where the `docker.compose.yml` file is).
7. If you are not using the tenant service in DEF, open the `docker-compose.override.yml` file, and remove the sections for the `xdbautomationworker` and `id` images.

© Copyright 2021, Sitecore® - all rights reserved.

3.2. Build the Docker images

When you have prepared the installation files, you must create Docker files for each role, and build the Docker images.

To build the images:

1. Go to the Sitecore container deployment folder on your local machine. Go to the folder for the Windows version and topology you are using, for example, `compose/ltsc2019/xp1`. Create a subfolder and name it `module`.
2. In the `module` folder, create these subfolders:
   - `mssql`
   - `mssql-init`

     NOTE
     The `mssql-init` image is only necessary if you are deploying to Azure Kubernetes Services (AKS).

   - `cm`
   - `cd`
   - `id`
   - `xdbautomationworker`

     NOTE
     You only need the `xdbautomationworker` and `id` modules if you are using the tenant service. If you are not using the tenant service, you do not have to create the folders and files for these two modules.

3. In each subfolder, create a new file and name it `Dockerfile`.
4. In the `mssql` folder, in the `Dockerfile` file, enter the following instructions:

```
# escape=`

ARG BASE_IMAGE
ARG DEF_IMAGE

FROM ${DEF_IMAGE} as def
FROM ${BASE_IMAGE}

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

# Deploy DEF dacpac file
COPY --from=def \module\db \def_data
RUN C:\DeployDatabases.ps1 -ResourcesDirectory C:\def_data; `
Remove-Item -Path C:\def_data -Recurse -Force;
```

5. If you are deploying to AKS, in the `mssql-init` folder, in the `Dockerfile` file, enter the following instructions:

© Copyright 2021, Sitecore® - all rights reserved.

```
# escape=`
ARG BASE_IMAGE
ARG DEF_IMAGE

FROM ${DEF_IMAGE} as def

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

FROM ${BASE_IMAGE} AS ce

COPY --from=def C:\module\db C:\resources\ce
```

6. In the `cm` folder, in the `Dockerfile` file, enter the following instructions:

```
# escape=`

ARG BASE_IMAGE
ARG DEF_IMAGE
ARG TOOL_IMAGE

FROM ${DEF_IMAGE} as def
FROM ${TOOL_IMAGE} as tooling
FROM ${BASE_IMAGE}

ARG SITECORE_ROOT

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

WORKDIR C:\inetpub\wwwroot

# Add DEF module
COPY --from=def \module\cm\content ${SITECORE_ROOT}
COPY --from=def \module\transforms\ C:\transforms\

COPY --from=tooling \tools\ \tools\
RUN C:\tools\scripts\Invoke-XdtTransform.ps1 -Path C:\inetpub\wwwroot -XdtPath \transforms
\cm
```

7. In the `cd` folder, in the `Dockerfile` file, enter the following instructions:

```
# escape=`

ARG BASE_IMAGE
ARG DEF_IMAGE

FROM ${DEF_IMAGE} as def
FROM ${BASE_IMAGE}

ARG SITECORE_ROOT

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

WORKDIR C:\inetpub\wwwroot

# Add DEF module
COPY --from=def \module\cm\content ${SITECORE_ROOT}
```

8. If you are using the tenant service, in the `id` folder, in the `Dockerfile` file, enter the following instructions:

© Copyright 2021, Sitecore® - all rights reserved.

```
# escape=`

ARG BASE_IMAGE
ARG DEF_IMAGE
ARG TOOL_IMAGE

FROM ${DEF_IMAGE} as def
FROM ${TOOL_IMAGE} as tooling
FROM ${BASE_IMAGE}

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

WORKDIR C:\Identity

# Add DEF module
COPY --from=def \module\transforms\ C:\transforms\

COPY --from=tooling \tools\ \tools\
RUN C:\tools\scripts\Invoke-XdtTransform.ps1 -Path C:\Identity -XdtPath c:\transforms\id
```

9. If you are using the tenant service, in the `xdbautomationworker` folder, in the `Dockerfile` file, enter the following instructions:

```
# escape=`

ARG BASE_IMAGE
ARG DEF_IMAGE

FROM ${DEF_IMAGE} as def
FROM ${BASE_IMAGE}

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

# Add DEF MA module
COPY --from=def \module\xdbautomationworker\content C:\service\
```

10. In the Windows console, go to the folder containing the `docker-compose.override.yml` file. Run the command `docker-compose build`.
11. When the build completes, run the command `docker-compose up -d`.

NOTE
Some modifications to Sitecore deployments, such as adding connection strings or changing the web configuration files, require you to use configuration transforms to change the configuration files. For information on how to apply configuration transforms, see the Sitecore container development documentation.

3.3. Update the Solr indexes

When the Docker compose command has finished, you must update your Solr indexes.

To update the indexes:

© Copyright 2021, Sitecore® - all rights reserved.

1. When the Docker compose command finishes, browse to your Sitecore URL, for example, `https://xp0cm.localhost/`. Open the control panel and click **Populate Solr Managed Schema**.
2. After Sitecore has populated the Solr Schema, click **Indexing Manager**.
3. In the **Indexing Manager** dialog, select the indexes you want to update, and click **Rebuild**. When the indexes have been rebuilt, click **Close**.

© Copyright 2021, Sitecore® - all rights reserved.

4. Add Data Exchange Framework module to Sitecore in Azure Kubernetes Service

To add Data Exchange Framework (DEF) in Azure Kubernetes Service (AKS) you must do the following in this order:

- Build images and push them to Azure
- Prepare configuration files for deployment
- Deploy the containers
- Update Solr indexes

4.1. Build images and push them to Azure

To build the images for DEF and push them to Azure:

1. Prepare the installation files as explained in Add Data Exchange Framework module to Sitecore in Docker.
2. Build the images for DEF as explained in Add Data Exchange Framework module to Sitecore in Docker.
3. Tag the images with the `docker tag` command. For example:

```
docker tag sitecore-def-xp0-assets:6.0.0.01525.109-10.0.19042.804-2009 $registry/
experimental/def/sitecore-xp1-cm:sc101def
```

4. Push the images to your Azure registry with the `docker push` command. For example

```
docker push $registry/experimental/def/sitecore-xp1-cm:sc101def
```

4.2. Prepare configuration files for deployment

To prepare configuration files in your installation for deployment:

1. Open the folder where you extracted the Data Exchange Framework container deployment package.
2. Navigate to the `DEF.Asset\k8s\<version>` folder, for example, `DEF.Asset\k8s\ltsc2019`. Copy the `overrides` subfolder to the Sitecore Experience Platform (SXP) container deployment package, in the `k8s\<version>` folder (on the same level as the `xp1` folder).

© Copyright 2021, Sitecore® - all rights reserved.

3. In the SXP container deployment package, in each of the `overrides\<topology>`, `overrides\<topology>\init`, and `overrides\<topology>\secrets` folders, locate the `kustomization.yaml` file. In each file, update the `bases` parameter with the appropriate folder names for your installation, for example, `../../xp1`.

   NOTE
   The `bases` parameter contains the placement of the original Sitecore container deployment files that the `kustomization.yaml` files override.

4. In each `kustomization.yaml` file, in the `images:` section, update the `newName` and `newTag` parameters with the values for the images you pushed to the Azure Registry, for example, `mssql-init`, `cm`, `cd`, `xdbautomationworker`, and `id`.

   NOTE
   To find the values you need for the `mssql-init` image, go to the Azure Container Registry, search for your `sitecore-xp1-mssql-init-def-test` image, and take the values from that image.

5. In the `overrides\<topology>\secrets` folder, in the `sitecore-tenant-service-connection-string.txt` file , update the connection string details. The file contains an example of how the connection string should look.

4.3. Deploy the containers

Before you deploy the containers to Kubernetes, you must prepare the AKS cluster configuration and deploy the ingress controller. For information on how to do this, see the *Installation Guide for Production Environment for Kubernetes* which is available on the Sitecore download page.

To deploy the containers and the necessary Kubernetes components:

1. Open the Windows console, and navigate to the `k8s\<version>` folder.
2. Deploy the secrets using this command:

   ```
   kubectl apply -k ./overrides/<topology>/secrets/
   ```

3. Run the `external` folder. Use this command:

   ```
   kubectl apply -k ./<topology>/external/
   ```

4. Wait for all containers to have the status *Ok/Running*. You can check the status with this command:

   ```
   kubectl get pods -o wide
   ```

5. Run the `init` folder. Use this command:

© Copyright 2021, Sitecore® - all rights reserved.

```
kubectl apply -k ./overrides/<topology>/init/
```

6. Wait for all containers to have the status *Completed*. You can check the status with this command:

```
kubectl get pods
```

7. To create persistent volumes, run this command:

```
kubectl apply -f ./<topology>/volumes/azurefile
```

8. Run the Sitecore containers with the SFCRM changes. Use this command:

```
kubectl apply -k ./overrides/<topology>/
```

9. Wait for all containers to have the status *Ok/Running*. You can check the status with the `kubectl get pods` command.
10. Obtain the external IP address. Use this command:

```
kubectl get service -l app=nginx-ingress
```

11. Update the local host file. For information on how to do this, see the *Installation Guide for Production Environment for Kubernetes*, which is available on the Sitecore download page.

4.4. Update Solr indexes

To update your Solr indexes:

1. Browse to your Sitecore URL, for example, `https://cm.globalhost/`. Open the control panel and click **Populate Solr Managed Schema**.
2. After Sitecore has populated the Solr Schema, click **Indexing Manager**.
3. In the **Indexing Manager** dialog, select the indexes you want to update, and click **Rebuild**. When the indexes have been rebuilt, click **Close**.

© Copyright 2021, Sitecore® - all rights reserved.