Install the Codeless Schema Extensions module using Docker

June 3, 2024

Table of Contents

1. Install the Sitecore Codeless Schema Extensions module using Docker

The Codeless Schema Extensions module enables business users to extend the xConnect schema without requiring code development.

This guide describes how to build custom Docker images that include the Codeless Schema Extensions module, using an out-of-the-box Sitecore Container Deployment Package. This involves downloading the deployment package then building the custom images that include the Sitecore Codeless Schema Extensions module.

If you are installing Codeless Schema Extensions on your existing Sitecore Container Deployment Package, use this guide as a reference to add the necessary custom configurations.

BEFORE YOU BEGIN

Make sure:

- You are using Sitecore Experience Platform 10.2 or later.
- Docker is installed.
- You have a container registry configured to host the Docker images built during the `docker-compose build` step.

To deploy Sitecore Codeless Schema Extensions, you must build custom Docker images for all the relevant roles and use them in your deployment. For more information about Sitecore development with Docker, see containers in Sitecore development.

1.1. Prepare for deployment

To prepare your system for the deployment:

1. Create a new folder named `deploy`.
2. Download the Sitecore Container Deployment Package for the Sitecore version you want to use, then unzip and navigate to the `compose\[windows-version]\[desired-topology]` folder. For example: `C:\Users\admin\Downloads\compose\ltsc2019\xp1`.
3. Copy the contents of this folder to your `deploy` folder. After copying, ensure that the `docker-compose.yml` file is present in the `deploy` folder.
4. Download the Sitecore Codeless Schema Extensions Container Deployment Package, then unzip and navigate to the `compose\[windows-version]\[desired-topology]` folder.
5. Copy the contents of this folder to your `deploy` folder.

1.2. Build custom images

To build custom images:

1. Create a new folder named `build`.
2. In the `build` folder, create a new folder named `cm`. In the `cm` folder, create a new `Dockerfile` and paste the instructions from the following code snippet depending on your desired topology.
   Repeat for the other Sitecore roles in your desired topology. When finished, there will be a folder and `Dockerfile` for every role in your desired topology. For example: `build\cm\Dockerfile`.

For XP Single (XP0):

`cm`:

```
# escape=`

ARG BASE_IMAGE
ARG GP_IMAGE

FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]
WORKDIR C:\inetpub\wwwroot
COPY --from=gp \module\cm\content C:\inetpub\wwwroot
```

`xdbautomationworker`:

```
# escape=`

ARG GP_IMAGE
ARG BASE_IMAGE

FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]
WORKDIR C:\service
COPY --from=gp \module\xconnect\content\App_Data\jobs\continuous\AutomationEngine\
C:\service
```

`xconnect`:

```
# escape=`

ARG BASE_IMAGE
ARG GP_IMAGE

FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

COPY --from=gp \module\xconnect\content\App_Data\Config C:\inetpub\wwwroot\App_Data\Config
COPY --from=gp \module\xconnect\content\App_Data\Models C:\inetpub\wwwroot\App_Data\Models
```

`xdbsearchworker`:

```
# escape=`
```

```
ARG BASE_IMAGE
ARG GP_IMAGE

FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

COPY --from=gp \module\xconnect\content\App_Data\jobs\continuous\IndexWorker\ C:\service
```

`cortexprocessingworker`

```
# escape=`

ARG BASE_IMAGE
ARG GP_IMAGE

FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

COPY --from=gp \module\xconnect\content\App_Data\jobs\continuous\ProcessingEngine\
C:\service
```

For XP Scaled (XP1):

Replace `<ROLE>` with the appropriate role name. For example, for `cm`: `COPY --from=gp \module\cm\content C:\inetpub\wwwroot`

`cm`, `cd`:

```
# escape=`

ARG BASE_IMAGE
ARG GP_IMAGE

FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]
WORKDIR C:\inetpub\wwwroot
COPY --from=gp \module\<ROLE>\content C:\inetpub\wwwroot
```

`xdbautomationworker`:

```
# escape=`

ARG GP_IMAGE
ARG BASE_IMAGE

FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]
WORKDIR C:\service
COPY --from=gp \module\xdbautomation\content\App_Data\jobs\continuous\AutomationEngine\
C:\service
```

`cortexprocessing`, `cortexprocessingworker`, `cortexreporting`, `xdbautomation`, `xdbautomationrpt`, `xdbcollection`, `xdbrefdata`, `xdbsearch`, `xdbsearchworker`:

```
# escape=`

ARG BASE_IMAGE
ARG GP_IMAGE
```

```
FROM ${GP_IMAGE} as gp
FROM ${BASE_IMAGE}

COPY --from=gp \module\<ROLE>\content C:\inetpub\wwwroot
```

3. In the `build` folder, create a new `docker-compose.yml` file with the following contents based on your desired topology:

For XP0:

```
services:
  cm:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp0-cm:${GP_VERSION}-${SITECORE_VERSION}
    build:
      context: ./cm
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-cm:${SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xconnect:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp0-xconnect:${GP_VERSION}-${SITECORE_VERSION}
    build:
      context: ./xconnect
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-xconnect:${SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xdbautomationworker:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp0-xdbautomationworker:${GP_VERSION}-${SITECORE_VERSION}
    build:
      context: ./xdbautomationworker
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-xdbautomationworker:${SITECORE_VERSION}
        GP_IMAGE: ${GP_IMAGE}:${GP_VERSION}
  xdbsearchworker:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp0-xdbsearchworker:${GP_VERSION}-${SITECORE_VERSION}
    build:
      context: ./xdbsearchworker
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-xdbsearchworker:${SITECORE_VERSION}
        GP_IMAGE: ${GP_IMAGE}:${GP_VERSION}
  cortexprocessingworker:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp0-cortexprocessingworker:${GP_VERSION}-${SITECORE_VERSION}
    build:
      context: ./cortexprocessingworker
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-cortexprocessingworker:${SITECORE_VERSION}
        GP_IMAGE: ${GP_IMAGE}:${GP_VERSION}
```

For XP1:

```
services:
  cm:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-cm:${GP_VERSION}-${SITECORE_VERSION}
    build:
```

```
      context: ./cm
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-cm:${SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  cd:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-cd:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./cd
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-cd:${SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xdbcollection:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-xdbcollection:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./xdbcollection
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-xdbcollection:$
{SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xdbsearch:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-xdbsearch:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./xdbsearch
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-xdbsearch:${SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  cortexprocessing:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-cortexprocessing:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./cortexprocessing
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-cortexprocessing:$
{SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  cortexreporting:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-cortexreporting:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./cortexreporting
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-cortexreporting:$
{SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xdbautomation:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-xdbautomation:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./xdbautomation
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-xdbautomation:$
{SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xdbautomationrpt:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-xdbautomationrpt:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./xdbautomationrpt
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-xdbautomationrpt:$
{SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xdbrefdata:
```

```
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-xdbrefdata:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./xdbrefdata
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-xdbrefdata:${SITECORE_VERSION}
        GP_IMAGE : ${GP_IMAGE}:${GP_VERSION}
  xdbautomationworker:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-xdbautomationworker:$
{GP_VERSION}-${SITECORE_VERSION}
    build:
      context: ./xdbautomationworker
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-xdbautomationworker:$
{SITECORE_VERSION}
        GP_IMAGE: ${GP_IMAGE}:${GP_VERSION}
  xdbsearchworker:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-xdbsearchworker:${GP_VERSION}-$
{SITECORE_VERSION}
    build:
      context: ./xdbsearchworker
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-xdbsearchworker:$
{SITECORE_VERSION}
        GP_IMAGE: ${GP_IMAGE}:${GP_VERSION}
  cortexprocessingworker:
    image: ${REGISTRY}sitecore-gp-standard-integration-xp1-cortexprocessingworker:$
{GP_VERSION}-${SITECORE_VERSION}
    build:
      context: ./cortexprocessingworker
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp1-cortexprocessingworker:$
{SITECORE_VERSION}
        GP_IMAGE: ${GP_IMAGE}:${GP_VERSION}
```

4. In the `build` folder, create a file named `.env` and add the following parameters:

> **NOTE**
> For more information about these variables, see the Sitecore image reference.

Variable	Value
SITECORE_DOCKER_REGISTRY	`scr.sitecore.com/sxp/`
SITECORE_VERSION	`<sitecore-version>-<windows-version>`
GP_IMAGE	The Codeless Schema Extensions module asset image. `GP_IMAGE=scr.sitecore.com/sxp/modules/gp-<sitecore_topology>-assets`
GP_VERSION	The version of the Codeless Schema Extensions module.
REGISTRY	Destination container registry of the custom images.

Example:

```
SITECORE_DOCKER_REGISTRY=scr.sitecore.com/sxp/
SITECORE_VERSION=10.3-ltsc2019
GP_IMAGE=scr.sitecore.com/sxp/modules/gp-xp0-assets
GP_VERSION=1.0-1809
REGISTRY=https://registry.hub.docker.com/repository/docker/tangalin/tobytest/
```

5. In the `build` folder, open PowerShell and run the following command: `docker-compose build`. This builds the images patched with the Codeless Schema Extensions files for each of the roles.

1.3. Run Sitecore Codeless Schema Extensions module on top of Sitecore

> **NOTE**
> For more information about installing Sitecore, see the installation guide for your version of Sitecore.

To run the Codeless Schema Extensions module on top of Sitecore:

1. In the `deploy` folder, ensure the following variables are set in the `.env file`:

Variable	Value
GP_VERSION	The version of the Codeless Schema Extensions module.
SITECORE_VERSION	`<sitecore-version>-<windows-topology>`

2. Check the `docker-compose.override.yml` file and ensure that the image names match the names of the custom images built in the previous section.
3. In the `deploy` folder, run the following command: `docker-compose up -d`.