
Introduction

About This Guide

Use this guide to build SAS software as a container and to deploy it to a supported cloud container service or platform.

- To use this guide successfully, you should have a working knowledge of the following:
 - SAS installation and configuration
 - Linux operating system and commands
 - Docker
 - A cloud container service or platform on which the SAS container can run, such as Amazon EC2, Google Cloud Container Service, and OpenStack
- The latest version of this guide is available at [SAS Analytics for Containers product page](#).

Benefits of Running SAS as a Container

- Moving to a container infrastructure benefits SAS programmers who want to move their code to the cloud.
- All software resources that are required to run SAS are included in the container, including the libraries and functions.
- Because containers are cloud-native, it is easy to build and test SAS as a container and then to move it to any cloud environment without making any additional changes.
- Users can access their customized container of SAS software using SAS Studio, a browser-based interface, or Jupyter Notebook, an open-source notebook-style interface. The IT department can deploy and manage SAS software using SAS Analytics for Containers.
- (Optional) Other software can be added to the container besides SAS.

Supported SAS Products and Technologies

- SAS Analytics for Containers, and related add-ons including the following software:
 - SAS/CONNECT: Cross-platform computing connections and parallel processing
 - SAS/ETS software: Econometrics and time series analysis
 - SAS/IML: Interactive matrix programming and exploratory analysis with integration to R

2

- SAS/OR software: Operations research and optimization modeling
- SAS/QC software: Statistical process control
- SAS Analytics for Hadoop Containers, which includes the previously listed software, plus the following software:
 - SAS/ACCESS Interface to Hadoop for connectivity between SAS and Hadoop
 - SAS In-Database Code Accelerator for Hadoop: Publish and run SAS DS2 code in parallel inside Hadoop
 - SAS Scoring Accelerator for Hadoop: Automate scoring processes within Hadoop for faster model performance

Related Resources

Documentation from SAS:

- [Product information page](#)
- [SAS Studio](#)
- [SAS and Hadoop documentation](#)

Third-party resources:

- [Docker](#)
- [Oracle Cloud Container Service](#)
- [Amazon EC2 Container Service](#)
- [Microsoft Azure Container Service](#)
- [Google Cloud Platform Container Clusters](#)
- [OpenStack](#)

SAS Technical Support

Technical support is available to all customers who license SAS software. However, we encourage you to engage your designated on-site SAS support personnel as your first support contact. If your on-site SAS support personnel cannot resolve your issue, have them contact SAS Technical Support to report your problem.

Before you call, explore the SAS Support website at support.sas.com/techsup/. This site offers access to the SAS Knowledge Base, as well as SAS communities, Technical Support contact options, and other support materials that might answer your questions.

When you contact SAS Technical Support, you are required to provide information, such as your SAS site number, company name, email address, and phone number, that identifies you as a licensed SAS software customer.

First Steps

How Deployment Works

Here is an overview of the process to build a container that includes SAS Studio.

- 1 To start, SAS Deployment Wizard is run to install SAS Studio on a supported Linux platform. Then, a TAR file that includes the installation directory is created manually.

TIP The Linux machine on which SAS Studio is installed and the machine on which you plan to run the container should have the same Linux distribution and version.

- 2 A recommendation is to create a script that starts SAS Studio. In a later step, this start-up script can be added to the container.

TIP For the container to run successfully, SAS Studio must start properly. Using a start-up script might help ensure a successful start-up.

- 3 A Dockerfile that contains the instructions for the installation is needed.
- 4 These files are needed in the directory in which Docker will be run:
 - the TAR file
 - the Dockerfile
 - an optional start-up script

Then, the Docker build and run commands must be run.

- 5 After the container is running, the IP address of the container can be shared with the data scientists, the analysts, and others who want to use SAS Studio in a web browser.

SAS Software Checklist

- Run SAS Deployment Wizard to install SAS Studio on a supported Linux 64-bit operating system. During the installation, change the default location for the SAS Studio installation to `/usr/local/SASHome`.

A supported Linux 64-bit operating system is required. A best practice is to install the SAS software on a Linux kernel and distribution that closely matches the environment for the version of Docker that you plan to use. For a list of supported Linux platforms for Docker, see the [Docker installation guide](#).

For details about how to install SAS Studio, refer to the installation and configuration instructions provided by the product.

- Create a TAR file that includes the SASHome directory. In a later step, this TAR file is required in order to build the Docker image. Here is an example:

```
tar - cvf SASHomeTar.tar /usr/local/SASHome
```

- It is recommended that you create a script that can start SAS Studio. When the container runs, this script is used to start SAS Studio.

```
#!/bin/bash
/usr/local/SASHome/SASFoundation/9.4/utilities/bin/setuid.sh
/usr/local/SASHome/sas/studioconfig/sasstudio.sh start
tail -f /dev/null
```

Note: You can use other methods to start SAS Studio in the container. However, the start-up script is used in the examples throughout this document.

Docker Checklist

- If you do not have Docker, install it. For more information, see the [Docker installation guide](#).
Swarm mode is supported. Running Docker in Swarm mode takes advantage of native cluster management and orchestration features for Docker.
- Create a Dockerfile that contains the order of instructions for building the Docker image. Here is an example:

```
FROM centos
MAINTAINER your name

# Install libraries and clean all
RUN yum -y install numactl-libs.x86_64 \
    passwd \
    libXp \
    libpng12 \
    libXmu.x86_64 \
    && yum clean all

# Add group
RUN useradd -m sas
RUN groupadd -g 1001 sasstaff

# Add sas user
RUN usermod -a -G sasstaff sas

# Set default password by pointing to /etc/passwd
RUN echo -e "mypassword" | /usr/bin/passwd --stdin sas

# Make the SASHome directory and add the TAR file
RUN mkdir -p /usr/local/SASHome
ADD SASHomeTar.tar /
RUN chown -R sas:sasstaff /usr/local/SASHome
EXPOSE 38080
```

```
# Add startup script to start SAS Studio
ADD startup.sh /
ENTRYPOINT ["/startup.sh"]
```

This Dockerfile starts with a CentOS parent image, and the following declarations are used to the build the new Docker image:

- The first RUN instruction installs the required libraries and cleans up the cache. The subsequent RUN instructions add a group named `sasstaff`, add a user named `sas` to the group, and then set the password `mypassword` for the user.

Note: SAS Studio uses operating system authentication. In the preceding example, the user name and password combination of `sas` and `mypassword` will be used to log on to SAS Studio via a web browser.

- The RUN and ADD instructions make an empty directory named `SASHome`, copy the `SASHomeTar.tar` file to the `SASHome` directory, and then change the owner and group of the `SASHome` directory. In addition, the EXPOSE declaration instructs the container to forward port 38080 to the associated host for connectivity.
- A start-up script named `startup.sh` starts SAS Studio automatically.

Note: Depending on your software requirements, you might need to specify an operating system version for the parent image.

Build and Run a Container

Before You Begin

- Make sure that you have completed the [checklists for the SAS software and Docker on page 3](#).
- Examples in the following instructions use variables that might differ for your environment.

Create and Run the Docker Image

- 1 Create a directory where you want to run Docker. Here is an example:

```
mkdir SASDocker
cd SASDocker
```

- 2 Add the following files to the `SASDocker` directory:

- the Dockerfile
- the TAR file that you created, which contains the `SASHome` directory

6

- the start-up script that starts SAS Studio

3 To build the Docker image, run the following command:

```
docker build -t sa4c:v1 .
```

Note: To run the command successfully, you must be in the directory where the Dockerfile resides, which is the SASDocker directory in our example.

4 To view the built images, run the following command:

```
docker images
```

5 To run the container, run the following command:

```
docker run -d -p 38080:38080 sa4c:v1
```

After a successful build, execute the Docker images to view the ID that was created in the Docker repository. Next, you can run the container (See step 5) or refer to the name of the container image.

Note: SAS Studio must be running in the container.

Examples of Running Docker

- To show all running containers and to view container IDs:

```
docker ps -a
```

- To force a container to remain running, add `-f /dev/null` at the end of the command:

```
docker run -d -p 38080:38080 containerID tail -f /dev/null
```

- To start the container without the ENTRYPOINT declaration:

```
docker run -d -p 38080:38080 sa4c:v1 tail -f /dev/null
```

- To execute a start-up script, which was added to the container:

```
docker exec containerID ./startup.sh
```

- To show start-up commands that can be used by an orchestration layer:

```
docker exec containerID /usr/local/SASHome/SASFoundation/9.4/utilities/bin/setuid.sh  
docker exec containerID /usr/local/SASHome/sas/studioconfig/sasstudio.sh start
```

For more information about running Docker, see the Docker run reference documentation:

<https://docs.docker.com/engine/reference/run/>

Third-Party Integration

Jupyter Integration

Data scientists and analysts can use Jupyter Notebook as the user interface to SAS Analytics for Containers. To learn more about how to integrate with Jupyter Notebook, see the following:

https://github.com/sassoftware/sas_kernel

Using Hadoop Data

SAS Analytics for Containers includes software that enables users to access Hadoop data.

- The basic SAS Analytics for Containers offering includes SAS/ACCESS Interface to Hadoop, which enables you to access Hadoop data through Hive and HiveServer2 and from HDFS. The benefits of using SAS/ACCESS Interface to Hadoop include the following:
 - Access to Hadoop data as SAS data sets without requiring the programmer to have specific Hadoop skills such as writing MapReduce code.
 - Hive interaction with Hadoop where data is pulled from Hadoop, as well as the ability to push procedures to run inside Hadoop, with no data movement.
 - Explicit and implicit pass-through, which enables the data scientist to run HiveQL without leaving the SAS environment.
- To gain the most value from the distributed compute power of Hadoop, the SAS Accelerators for Hadoop can be included with the SAS Analytics for Containers with Hadoop offering. This allows the data scientist to publish models to the Hadoop cluster and to submit work to Hadoop via the SAS DS2 language.
- Taking advantage of the processing power of Hadoop is accomplished by SAS In-Database Code Accelerator for Hadoop and SAS Scoring Accelerator for Hadoop. Both solutions enable the data scientist to send SAS models and DATA step code to run inside the Hadoop data cluster. This workload is managed by YARN (Yet Another Resource Negotiator) and the Hadoop ecosystem. The SAS developer does not have to learn all the nuances of working with Hadoop, such as creating and scheduling MapReduce jobs. Instead, the SAS data scientist writes programs only in the SAS language.

Troubleshooting

SAS Studio Stops Unexpectedly

Explanation

Container does not remain running and SAS Studio stops.

Resolution

When the container is run, SAS Studio must start appropriately. Make sure that you are using a proper method for starting SAS Studio, such as a start-up script.

Container Does Not Start on a Macintosh Computer

Explanation

A container that is built with Docker for a Macintosh computer does not run.

Resolution

Build the Docker image on a Linux machine, and then copy the image to a Macintosh computer.

To run the container, you need an IP address and a port that points to where the container is running. The method to collect this information varies according to the particular cloud provider. Some providers assign a single address to the container and use a dynamic method of producing a different port for each container that is built. Integration with the orchestration layer is necessary in order to obtain this information. As a simple example, Docker can provide an IP address with a fixed port (Docker for a Macintosh computer provides an IP address of 192.168.99.100). After the container and SAS Studio are started, a browser can connect via this address:

<http://192.168.99.100:38080>