Installing the AWS SAM CLI on Windows

The following steps help you to install and conﬁgure the required prerequisites for using the AWS SAM CLI on your Windows host:

Create an AWS account.

Conﬁgure IAM permissions.

Install Docker. Note: Docker is only a prerequisite for testing your application locally.

Install the AWS SAM CLI.

Step 1: Create an AWS Account

If you don't already have an AWS account, see [aws.amazon.com](https://aws.amazon.com/) and choose Create an AWS Account.

Step 2: Create an IAM User with Administrator Permissions

If you don't already have an IAM user with administrator permissions, see [Creating Your First IAM Admin](https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started_create-admin-group.html) [User and Group](https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started_create-admin-group.html) in the *IAM User Guide*.

In addition, you must set up AWS credentials to enable the AWS SAM CLI to make AWS service calls. For example, the AWS SAM CLI makes calls to Amazon S3 and AWS CloudFormation. For more information about setting up AWS credentials.

Step 3: Install Docker

Docker is only a prerequisite for testing your application locally and building deployment packages using the --use-container ﬂag. You can skip this section or install Docker at a later time if you don't plan to use these features initially.

Docker is an application that runs containers on your Linux machines. AWS SAM provides a local environment that's similar to AWS Lambda to use as a Docker container. You can use this container to build, test, and debug your serverless applications.

You must have Docker installed and working to be able to run serverless projects and functions locally with the AWS SAM CLI. The AWS SAM CLI uses the DOCKER\_HOST environment variable to contact the Docker daemon. The following steps describe how to install, conﬁgure, and verify a Docker installation to work with the AWS SAM CLI.

Install Docker.

Docker Desktop supports the most recent Windows operating system. For legacy versions of Windows, the Docker Toolbox is available. Choose your version of Windows for the correct Docker installation steps:

Conﬁgure your shared drives.

The AWS SAM CLI requires that the project directory, or any parent directory, is listed in a shared drive. Choose your version of Windows below for the correct shared drive instructions:

Verify the installation.

After Docker is installed, verify that it's working. Also conﬁrm that you can run Docker commands from the AWS SAM CLI (for example, docker ps). You don't need to install, fetch, or pull any containers—the AWS SAM CLI does this automatically as required.

If you run into issues installing Docker, see the [Logs and troubleshooting](https://docs.docker.com/docker-for-windows/troubleshoot/) section of the *Docker installation guide* for additional troubleshooting tips.

Step 4: Install the AWS SAM CLI

Windows Installer (MSI) ﬁles are the package installer ﬁles for the Windows operating system. Follow these steps to install the AWS SAM CLI using the MSI ﬁle.

Install the AWS SAM CLI [64-bit](https://github.com/awslabs/aws-sam-cli/releases/latest/download/AWS_SAM_CLI_64_PY3.msi).

If you operate on 32-bit machine, execute the following command: pip install aws- sam-cli

Verify the installation.

After completing the installation, verify it by opening a new command prompt or PowerShell prompt. You should be able to invoke sam from the command line.

**sam --version**

You should see output like the following after successful installation of the AWS SAM CLI:

SAM CLI, version 0.33.0

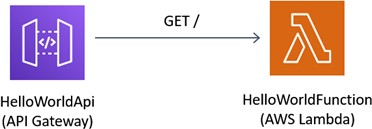
You're now ready to start development.

Deploying a Hello World Application

In this guide, you download, build, and deploy a sample Hello World application using AWS SAM. You then test the application in the AWS Cloud, and optionally test it locally on your development host.

This application implements a simple API backend. It consists of an API Gateway endpoint and a Lambda function. When you send a GET request to the API Gateway endpoint, the Lambda function is invoked.

This function returns a hello world message.



The following is a preview of commands that you run to create your Hello World application.

#Step 1 - Download a sample application

**sam init**

#Step 2 - Build your application

**cd sam-app sam build**

**#step 3 deploy**

**sam deploy --guided**

#Step 3 - Deploy your application

**sam deploy --guided**

Step 4: Testing Your Application Locally (Optional)

When you're developing your application, you might also ﬁnd it useful to test locally. The AWS SAM CLI provides the sam local command to run your application using Docker containers that simulate the execution environment of Lambda. There are two options to do this:

Host your API locally

Invoke your Lambda function directly

This step describes both options.

Host Your API Locally

Command to run:

**sam local start-api**

Troubleshooting

SAM CLI error: "no such option: --app-template"

When executing sam init, you see the following error:

Error: no such option: --app-template

This means that you are using an older version of the AWS SAM CLI that does not support the --app- template parameter. To ﬁx this, you can either update your version of AWS SAM CLI to 0.33.0 or later, or omit the --app-template parameter from the sam init command.

SAM CLI error: "no such option: --guided"

When executing sam deploy, you see the following error:

Error: no such option: --guided

This means that you are using an older version of the AWS SAM CLI that does not support the --guided

parameter. To ﬁx this, you can either update your version of AWS SAM CLI to 0.33.0 or later, or omit the

--guided parameter from the sam deploy command.

OUTPUT:

#### Host Your API Locally

Command to run:

**sam local start-api**

2019-07-12 15:27:58 Mounting HelloWorldFunction at http://127.0.0.1:3000/hello [GET] 2019-07-12 15:27:58 You can now browse to the above endpoints to invoke your functions.

You do not need to restart/reload SAM CLI while working on your functions, changes will be reflected instantly/automatically. You only need to restart SAM CLI if you update your AWS SAM template

2019-07-12 15:27:58 \* Running on http://127.0.0.1:3000/ (Press CTRL+C to quit)

Fetching lambci/lambda:python3.7 Docker container image......................................................................................

2019-07-12 15:28:56 Mounting /<working-development-path>/sam-app/.aws-sam/build/ HelloWorldFunction as /var/task:ro,delegated inside runtime container

START RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72 Version: $LATEST END RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72

**curl http://127.0.0.1:3000/hello**

Example output:

2019-07-12 15:29:57 Invoking app.lambda\_handler (python3.7)

2019-07-12 15:29:57 Found credentials in shared credentials file: ~/.aws/credentials

Fetching lambci/lambda:python3.7 Docker container image......

2019-07-12 15:29:58 Mounting /<working-development-path>/sam-app/.aws-sam/build/ HelloWorldFunction as /var/task:ro,delegated inside runtime container

START RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72 Version: $LATEST END RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72

REPORT RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72 Duration: 7.92 ms Billed Duration: 100 ms Memory Size: 128 MB Max Memory Used: 22 MB

2019-07-01 14:08:42 Found credentials in shared credentials file: ~/.aws/credentials 2019-07-01 14:08:42 Invoking app.lambda\_handler (python3.7)

Fetching lambci/lambda:python3.7 Docker container

2019-07-01 14:09:39 Mounting /<working-development-path>/sam-app/.aws-sam/build/ HelloWorldFunction as /var/task:ro,delegated inside runtime container

START RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72 Version: $LATEST END RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72

REPORT RequestId: 52fdfc07-2182-154f-163f-5f0f9a621d72 Duration: 3.51 ms Billed Duration: 100 ms Memory Size: 128 MB Max Memory Used: 22 MB

{"statusCode":200,"body":"{\"message\": \"hello world\"}"}

**sam local generate-event apigateway aws-proxy --body "" --path "hello" --method GET > api- event.json**

**diff api-event.json event.json**

**sam local invoke "HelloWorldFunction" -e events/event.json**

{"statusCode":200,"body":"{\"message\": \"hello world\"}"}

Clean Up

To delete the AWS CloudFormation stack created with this tutorial using the AWS Management Console, follow these steps:

Sign in to the AWS Management Console and open the AWS CloudFormation console.

In the left navigation pane, choose Stacks.

In the list of stacks, choose sam-app (or the name of stack you created).

Choose Delete.

When done, the status of the of the stack will change to DELETE\_COMPLETE.

Alternatively, you can delete the AWS CloudFormation stack by executing the following AWS CLI command:

**aws cloudformation delete-stack --stack-name *sam-app* --region *region***

Verify Deleted Stack

For both methods of deleting the AWS CloudFormation stack, you can verify it was deleted by going to the, choosing Stacks in the left navigation pane, and choosing Deleted in the dropdown to the right of the search text box. You should see your stack name sam-app (or the name of the stack you created) in the list of deleted stacks.

Conclusion

In this tutorial, you've done the following:

Created, built, and deployed a serverless application to AWS with AWS SAM.

Tested your application locally by using the AWS SAM CLI and Docker.

Deleted the AWS resources that you no longer need.

When done, the status of the of the stack will change to DELETE\_COMPLETE.

Alternatively, you can delete the AWS CloudFormation stack by executing the following AWS CLI command:

**aws cloudformation delete-stack --stack-name *sam-app* --region *region***

Verify Deleted Stack

For both methods of deleting the AWS CloudFormation stack, you can verify it was deleted by going to the [https://console.aws.amazon.com/cloudformation](https://console.aws.amazon.com/cloudformation/), choosing Stacks in the left navigation pane, and choosing Deleted in the dropdown to the right of the search text box. You should see your stack name sam-app (or the name of the stack you created) in the list of deleted stacks.