Setting Up the Prerequisites

The main command-line tool which you'll be installing and executing is just called `sam`
The AWS Cloud9 cloud-based integrated development environment also supports SAM tools.

Installing the required tools

If you experience any problems with installing the tools below or want to use an alternative way of managing the packages, check out the *AWS SAM Developer Guide* online.

Python

check whether Python is installed and which version you have by running the following command:

```
python --version
```

If this command prints an error or you are using an outdated version, you will need to upgrade Python. In case you need to upgrade or install Python, get the one-click installer from https://www.python.org.

PIP

In addition to the Python runtime, you will also need to have the `pip` package management tool. Most Python installations already have one. To check whether it is installed on your machine, run the following command:

```
pip --version
```

If this command prints an error, `pip` is missing on your system, so install it using the instructions from https://pip.pypa.io.

AWS

The final prerequisite is the basic AWS command-line tools package. Most developers that access AWS in any way usually have those tools already installed. To check whether your system already has these tools, run the following command:

```
aws --version
```

If the command prints an error, you can download the tools by running the following command:

```
pip install awscli
```

Installing JavaScript and SAM CLI

Installing JavaScript tools

Node.js

you'll be using JavaScript with Node.js for developing Lambda functions.

To check whether Node.js is installed on your machine, run the following command:

```
node --version
```

Get the Node.js installer for your operating system from https://nodejs.org/ if this command prints an error.

Installing the SAM command-line tools

use the `pip` package manager to download `sam`. Run the following command:

```
pip install aws-sam-cli
```

To check whether the installation worked, run the following command:

```
sam --version
```

If you get a response similar to the following, the software is ready:

```
$ sam --version
```

```
SAM CLI, version 0.40.0
```

Configuring Access Credentials

SAM configuration

AWS SAM CLI reuses the credential configurations from AWS command-line tools.

To deploy software to the AWS cloud, you will need an access key ID and a secret key ID associated with your user account. If you do not have these already, here is how you can generate a set of keys:

1. Sign in to the AWS Web Console at https://aws.amazon.com/.
2. Select the Identity and Access Management (IAM) service.
3. In the left-hand IAM menu, select *Users*.
4. Click on the *Add User* button.
5. On the next screen, enter a name for the user account then, in the 'Select AWS access type' section, select *Programmatic access*.
6. Click the Next button to assign permissions, then select Attach existing policies directly.
7. In the list of policies, find the PowerUserAccess and IAMFullAccess policies and tick the checkboxes next to them.

8. You can skip the remaining wizard steps. The final page will show the access key ID and show a link to reveal the secret key as shown in the figure below. Reveal the secret key and copy both keys somewhere.

Once you have the access keys, you may run the following command to save the keys to your local machine:

```
aws configure
```

When the AWS utility asks you about the keys, paste what you copied in the previous step. You will also likely be asked to enter a default region and a default output format.

For the default output format, enter `json`, or just press Enter to keep it unset.

For the region, use `us-east-1`

Verification

Check that your credentials are correctly configured by running the following command line:

```
aws sts get-caller-identity
```

If this command prints a result similar to the following, everything works correctly in the terminal provided above:

```
$ aws sts get-caller-identity

{

    "UserId": "111111111111",

    "Account": "222222222222",

    "Arn": "arn:aws:iam:1111111111:root"

}
```

If you get an error, check out the section *Configuring the AWS CLI* from the AWS CLI user guide for troubleshooting information.

Initialising the Application

The SAM command-line tool

SAM command-line tools can generate sample projects and events so that you can get started easily.

The `sam init` command

to create a simple SAM project in a new subdirectory, run the following command:

```
sam init --runtime nodejs12.x --name app --app-template
hello-world
```

You will be asked for the following permission:

```
Allow SAM CLI to download AWS-provided quick start
templates from Github [Y/n]:
```

Enter `y` as a response. You should get a quick confirmation about the initialization of a new project

Next

Steps to Deploy SAM Applications

Step 1: Build

To prepare the function source for uploading to AWS, execute the following command in the main project directory (`app`). This will be the one that contains the CloudFormation template (`template.yaml`):

```
sam build
```

Step 2: Package

Creating a bucket

create a new S3 bucket using the following command line (replace `BUCKET-NAME` with your chosen name):

```
aws s3 mb s3://BUCKET-NAME
```

For example, to create a bucket called `sam-project-deployment`, run the following command:

```
aws s3 mb s3://sam-project-deployment
```

`sam package` command

SAM has a convenient shortcut to zip up and upload function packages to S3. Just run the following command in the directory containing the CloudFormation template (`template.yaml`) and remember to use your bucket name:

```
sam package --s3-bucket $BUCKET_NAME --output-template-file output.yaml
```

This command will produce another CloudFormation template, saving it to the file called `output.yaml`, as requested in the command parameter `--output-template-file`.

Deploying a SAM Application

Step 3: Deploy

You now have a single CloudFormation template (`output.yaml`) which describes the entire infrastructure and links to a packaged version of the function code. To create a running instance of your application, you'll need to create a new CloudFormation stack based on this template. To do that, run the following command in the directory that contains the packaged template (`output.yaml`):

```
sam deploy --template-file output.yaml --stack-name sam-test-1 --capabilities CAPABILITY_IAM
```

In a few moments, you will see that SAM has started deploying your project:

```
$ sam deploy --template-file output.yaml --stack-name sam-test-1 --capabilities CAPABILITY_IAM
```

```
Waiting for changeset to be created..
```

```
Waiting for stack create/update to complete
```

```
Successfully created/updated stack - sam-test-1
```

That's it for deploying a SAM application

Inspecting a Stack

(output)

Inspecting a stack from the AWS Web Console

1. Sign in to the AWS Web Console, at https://aws.amazon.com/.
2. Find the CloudFormation service.

U can see list of stacks
click on `sam-test-1` (or whatever you called the stack)

Inspecting a stack from the command line

```
aws cloudformation describe-stacks --stack-name sam-test-1
```