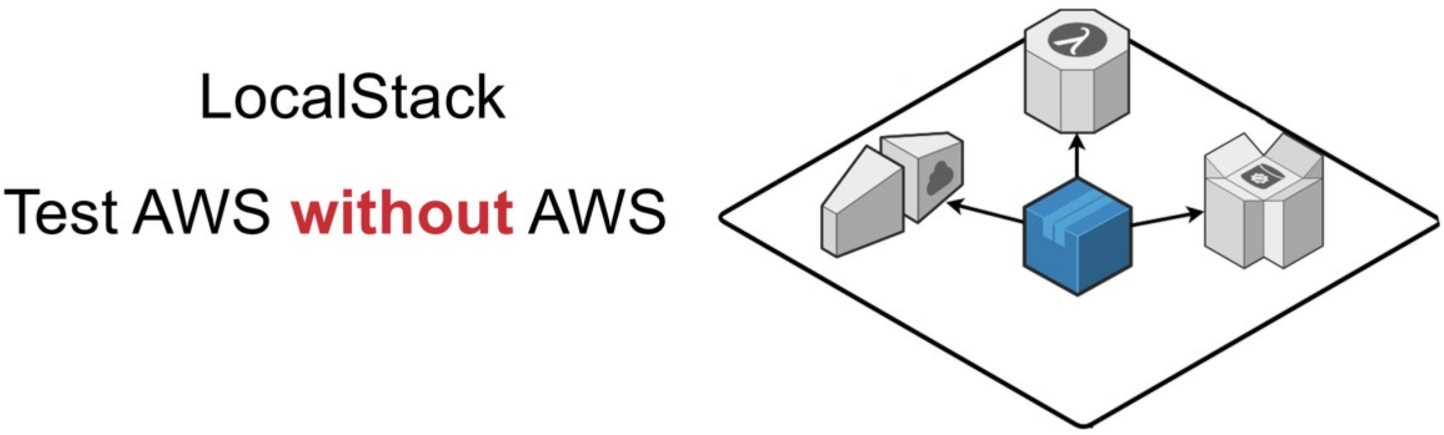


EXPEDIA GROUP TECHNOLOGY — SOFTWARE

Development and Testing of AWS without AWS: Localstack

Localstack provides an AWS-like environment that can be used for development and testing without needing AWS available

Miro Barsocchi
Jun 30 · 4 min read

AWS provides a lot of useful tools and services for developers: Serverless, Lambda functions, SNS queue, DynamoDB, S3 storage. When we start developing something connected to such services, obviously we can use one of the test environments provided by Amazon (LAB). Nonetheless, there is also a “local way” to go: **Localstack**.

Reduce costs and save time!

Localstack provides an AWS-like environment that can be used for development and/or testing purposes.

> ***Will your service use an SNS queue? Then, you can start the development with LocalStack.***

> *Will your app need information by a Lambda? Then, you can develop it using LocalStack.*
>
> *Will your frontend need S3 resources? Then, you can use LocalStack S3.*

How doest it work?

LocalStack gets started inside a Docker container and it contains a lot of the Cloud APIs of AWS.

Installing

The easiest way to install LocalStack is via pip (I know…that's for Mac, but there must also be easy ways for other platforms):

`pip install localstack`

Running in Docker

You can run LocalStack with a command

`localstack start`

or with a docker compose yml file

`docker-compose up`

An example of compose yml file is

```
version: '2.1'
services:
...
localstack:
image: localstack/localstack
ports:
- "4567-4599:4567-4599"
- "${PORT_WEB_UI-8080}:${PORT_WEB_UI-8080}"
environment:
- SERVICES=${SERVICES- }
- DEBUG=${DEBUG- }
- DATA_DIR=${DATA_DIR- }
- PORT_WEB_UI=${PORT_WEB_UI- }
- LAMBDA_EXECUTOR=${LAMBDA_EXECUTOR- }
- KINESIS_ERROR_PROBABILITY=${KINESIS_ERROR_PROBABILITY- }
- DOCKER_HOST=unix:///var/run/docker.sock
volumes:
- "${TMPDIR:-/tmp/localstack}:/tmp/localstack"
```

How did we use it?

We needed to integrate a new service to an already existing system that automatically pushes messages to a Slack channel using:

- SNS queue
- Lambda function
- Dynamo DB
- Nodejs App

SNS queue: a message queue that receives all the triggers

Lambda: serverless function provided by AWS

DynamoDb: a database that stores needed configurations

NodeJs App: an app with all the required logic

To test and verify the behaviour of the whole system, we set up **Localstack** to act as the needed AWS services, so we added a YML file that spun up what we wanted:

```
version: '2.1'
services:
localstack:
container_name: "${LOCALSTACK_DOCKER_NAME-localstack_main}"
image: localstack/localstack
ports:
- "4567-4597:4567-4597"
- "${PORT_WEB_UI-8080}:${PORT_WEB_UI-8080}"
environment:
- SERVICES=lambda,dynamodb
- DATA_DIR=${DATA_DIR- }
- DEBUG=1
- DEFAULT_REGION=us-west-2
- PORT_WEB_UI=${PORT_WEB_UI- }
- LAMBDA_EXECUTOR=${LAMBDA_EXECUTOR- }
- KINESIS_ERROR_PROBABILITY=${KINESIS_ERROR_PROBABILITY- }
- DOCKER_HOST=unix:///var/run/docker.sock
volumes:
- "${TMPDIR:-/tmp/localstack}:/tmp/localstack"
- "/var/run/docker.sock:/var/run/docker.sock"
```

As you can see, we set up the services **Lambda** and **Dynamodb** (we didn't add the SNS topic because it was of less interest)

`- SERVICES=lambda,dynamodb`

After the setup we needed to import and configure data and Lambda for our needs. To do this, we inserted a bash script in the repo that:

1. Creates the lambda package

`gulp zip`

2. Creates a dynamodb table and inserts data in it

```
awslocal dynamodb create-table --table-name MyTableName --attribute-
definitions
AttributeName=slackChannel,AttributeType=S --key-schema
AttributeName=slackChannel,KeyType=HASH --provisioned-throughput
ReadCapacityUnits=5,WriteCapacityUnits=5

awslocal dynamodb put-item --table-name MyTableName --item
'{"somekey":{"somesubkey":{"key":{"S":"value"}}},"somethingelse":
{"S":"anothervalue"}}'
```

3. Deploys the lambda function

```
awslocal lambda create-function --function-name myLambda--handler
index.handler --environment '{"Variables":
{"var1":"val1","var2":"val2"}}' --runtime nodejs10.x --role whatever
--zip-file fileb://build/myLambda.zip
```

N.B.

awslocal is a command but basically it's an alias for `aws --endpoint-url=http://localhost:4568` it acts as the **aws** command, but locally. As you can see, you can also specify the runtime for the Lambda, in this case **nodejs10.x**.

All this setup was in a separate bash script file that, after **localstack** is up and running, populates it with the required function and data.

> This project took me less than I expected — and I admit to be very cautious when it comes to complexity estimation.

Localstack is a very powerful tool, and when you need to test and develop something that requires AWS resources, it can definitely save your time (and some money as well, to be honest!)

During this journey we have learnt about some of the AWS capabilities, localstack, lambda, dynamodb and testing.

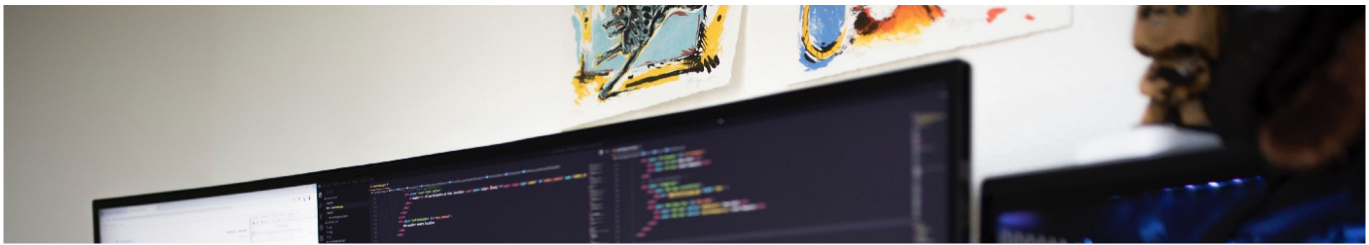
- - - - - -

Ref.

Localstack repo: https://github.com/localstack/localstack

Thanks for reading

Miro Barsocchi: Software tester and also Electronic engineer, radio speaker, actor, surfer, barman, but only two of these are seriously. You can find me on Twitter or Github or elsewhere.



Thanks to Giorgio Delle Grottaglie (hide).

AWS Software Development Testing Localstack Programming