pytest-localstack Documentation

Release 0.2.0

Jaye Doepke

Mar 07, 2019

Contents

pytest-localstack is a plugin for pytest to create AWS integration tests via a Localstack Docker container.

Read The Docs

Requires:

- pytest >= 3.3.0
- Docker

Tested against Python >= 3.6.

CHAPTER 1

Features

- Create pytest fixtures that start and stop a Localstack container.
- Temporarily patch botocore to redirect botocore/boto3 API calls to Localstack container.
- Plugin system to easily extend supports to other AWS client libraries such as aiobotocore.

CHAPTER 2

Example

```
import boto3
import pytest_localstack

localstack = pytest_localstack.patch_fixture(
    services=["s3"],  # Limit to the AWS services you need.
    scope='module',  # Use the same Localstack container for all tests in this module.
    autouse=True,  # Automatically use this fixture in tests.
)

def test_s3_bucket_creation():
    s3 = boto3.resource('s3')  # Botocore/boto3 will be patched to use Localstack
    assert len(list(s3.buckets.all())) == 0
    bucket = s3.Bucket('foobar')
    bucket.create()
    assert len(list(s3.buckets.all())) == 1
```

CHAPTER 3

Services

- apigateway
- cloudformation
- cloudwatch
- dynamodb
- dynamodbstreams
- es
- firehose
- kinesis
- lambda
- redshift
- route53
- s3
- ses
- sns
- ssm
- sqs

CHAPTER 4

Installation

```
$ pip install pytest-localstack
```

CHAPTER 5

TODO

- More detailed docs.
- Break Docker container running out of LocalstackSession.
- Make botocore patching more comprehensible.
- Add common test resource fixture factories i.e. S3 buckets, SQS queues, SNS topics, etc.
- Test this works for non-localhost Docker containers.
- Add other client libraries such as aiobotocore.

CHAPTER 6

Table of Contents

6.1 Usage

`pytest_localstack.`**`patch_fixture`**(*scope='function', services=None, autouse=False, docker_client=None, region_name=None, kinesis_error_probability=0.0, dynamodb_error_probability=0.0, container_log_level=10, localstack_version='latest', auto_remove=True, pull_image=True, container_name=None, **kwargs*)

Create a pytest fixture that temporarily redirects all botocore sessions and clients to a Localstack container.

This is not a fixture! It is a factory to create them.

The fixtures that are created by this function will run a Localstack container and patch botocore to direct traffic there for the duration of the tests.

Since boto3 uses botocore to send requests, boto3 will also be redirected.

Parameters

- **scope** (*str, optional*) – The pytest scope which this fixture will use. Defaults to `"function"`.
- **services** (*list, dict, optional*) – One of
  - A `list` of AWS service names to start in the Localstack container.
  - A `dict` of service names to the port they should run on.

  Defaults to all services. Setting this can reduce container startup time and therefore test time.
- **autouse** (*bool, optional*) – If `True`, automatically use this fixture in applicable tests. Default: `False`
- **docker_client** (`DockerClient`, optional) – Docker client to run the Localstack container with. Defaults to `docker.client.from_env()`.

- **region_name** (*str, optional*) – Region name to assume. Each Localstack container acts like a single AWS region. Defaults to `"us-east-1"`.
- **kinesis_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into Kinesis API responses.
- **dynamodb_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into DynamoDB API responses.
- **container_log_level** (*int, optional*) – The logging level to use for Localstack container logs. Defaults to `logging.DEBUG`.
- **localstack_version** (*str, optional*) – The version of the Localstack image to use. Defaults to `"latest"`.
- **auto_remove** (*bool, optional*) – If `True`, delete the Localstack container when it stops. Default: `True`
- **pull_image** (*bool, optional*) – If `True`, pull the Localstack image before running it. Default: `True`
- **container_name** (*str, optional*) – The name for the Localstack container. Defaults to a randomly generated id.
- ****kwargs*** – Additional kwargs will be passed to the *`LocalstackSession`*.

Returns A `pytest fixture`.

`pytest_localstack.`**`session_fixture`**(*scope='function', services=None, autouse=False, docker_client=None, region_name=None, kinesis_error_probability=0.0, dynamodb_error_probability=0.0, container_log_level=10, localstack_version='latest', auto_remove=True, pull_image=True, container_name=None, **kwargs*)

Create a pytest fixture that provides a LocalstackSession.

This is not a fixture! It is a factory to create them.

The fixtures that are created by this function will yield a *`LocalstackSession`* instance. This is useful for simulating multiple AWS accounts. It does not automatically redirect botocore/boto3 traffic to Localstack (although *`LocalstackSession`* has a method to do that.) The *`LocalstackSession`* instance has factories to create botocore/boto3 clients that will connect to Localstack.

Parameters

- **scope** (*str, optional*) – The pytest scope which this fixture will use. Defaults to `"function"`.
- **services** (*list, dict, optional*) – One of:
  - A `list` of AWS service names to start in the Localstack container.
  - A `dict` of service names to the port they should run on.

  Defaults to all services. Setting this can reduce container startup time and therefore test time.
- **autouse** (*bool, optional*) – If `True`, automatically use this fixture in applicable tests. Default: `False`
- **docker_client** (`DockerClient`, optional) – Docker client to run the Localstack container with. Defaults to `docker.client.from_env()`.

- **region_name** (*str, optional*) – Region name to assume. Each Localstack container acts like a single AWS region. Defaults to `"us-east-1"`.
- **kinesis_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into Kinesis API responses.
- **dynamodb_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into DynamoDB API responses.
- **container_log_level** (*int, optional*) – The logging level to use for Localstack container logs. Defaults to `logging.DEBUG`.
- **localstack_version** (*str, optional*) – The version of the Localstack image to use. Defaults to `"latest"`.
- **auto_remove** (*bool, optional*) – If `True`, delete the Localstack container when it stops. Default: `True`
- **pull_image** (*bool, optional*) – If `True`, pull the Localstack image before running it. Default: `True`.
- **container_name** (*str, optional*) – The name for the Localstack container. Defaults to a randomly generated id.
- ****kwargs*** – Additional kwargs will be passed to the *`LocalstackSession`*.

Returns A `pytest fixture`.

6.2 Internals

6.2.1 LocalstackSession

class pytest_localstack.session.**LocalstackSession**(*docker_client, services=None, region_name=None, kinesis_error_probability=0.0, dynamodb_error_probability=0.0, container_log_level=10, localstack_version='latest', auto_remove=True, pull_image=True, container_name=None, use_ssl=False, **kwargs*)

Run a localstack Docker container.

This class can start and stop a Localstack container, as well as capture its logs. It also implments a plugin system to add factories for the various AWS client libraries (botocore, boto3, etc).

Can be used as a context manager:

```
>>> import docker
>>> client = docker.from_env()
>>> with LocalstackSession(client) as session:
...     s3 = session.boto3.resource('s3')
```

Parameters

- **docker_client** – A docker-py Client object that will be used to talk to Docker.
- **services** (*list|dict, optional*) – One of
  - A list of AWS service names to start in the Localstack container.
  - A dict of service names to the port they should run on.

  Defaults to all services. Setting this can reduce container startup time and therefore test time.
- **region_name** (*str, optional*) – Region name to assume. Each Localstack container acts like a single AWS region. Defaults to 'us-east-1'.
- **kinesis_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into Kinesis API responses.
- **dynamodb_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into DynamoDB API responses.
- **container_log_level** (*int, optional*) – The logging level to use for Localstack container logs. Defaults to `logging.DEBUG`.
- **localstack_version** (*str, optional*) – The version of the Localstack image to use. Defaults to *latest*.
- **auto_remove** (*bool, optional*) – If True, delete the Localstack container when it stops.
- **container_name** (*str, optional*) – The name for the Localstack container. Defaults to a randomly generated id.
- **use_ssl** (*bool, optional*) – If True use SSL to connect to Localstack. Default is False.
- ****kwargs*** – Additional kwargs will be stored in a *kwargs* attribute in case test resource factories want to access them.

6.2.2 Plugins/Hooks

Much like pytest, itself, pytest-localstack uses pluggy to implement a plugin system. These plugins can be used to add additional functionality to pytest-localstack and to trigger callbacks when the Localstack container is started and stopped.

`pytest_localstack.hookspecs.`**`contribute_to_module`**(*pytest_localstack*)

Hook to add additional functionality to the `pytest_localstack` module.

Primarially used to add importable fixture factories at a top level.

`pytest_localstack.hookspecs.`**`contribute_to_session`**(*session*)

Hook to add additional functionality to `LocalstackSession`.

Primarially used to add test resource factories to sessions. See *`pytest_localstack.contrib.botocore`* for an example of that.

`pytest_localstack.hookspecs.`**`session_started`**(*session*)

Hook fired when `LocalstackSession` has started.

`pytest_localstack.hookspecs.`**`session_starting`**(*session*)

Hook fired when `LocalstackSession` is starting.

`pytest_localstack.hookspecs.`**`session_stopped`**(*session*)
Hook fired when `LocalstackSession` has stopped.

`pytest_localstack.hookspecs.`**`session_stopping`**(*session*)
Hook fired when `LocalstackSession` is stopping.

Plugins manager.

See also:

`hookspecs`

`pytest_localstack.plugin.`**`register_plugin_module`**(*module_path*, *required=True*)
Register hooks in a module with the PluginManager by Python path.

Parameters

- **`module_path`** (*`str`*) – A Python dotted import path.
- **`required`** (*`bool, optional`*) – If False, ignore ImportError. Default: True.

Returns The imported module.

Raises `ImportError` – If *required* is True and the module cannot be imported.

6.2.3 Contrib

botocore

Test resource factory for the botocore library.

class `pytest_localstack.contrib.botocore.`**`BotocoreTestResourceFactory`**(*localstack_session*)
Create botocore clients to interact with a *`LocalstackSession`*.

Parameters **`localstack_session`** (*`LocalstackSession`*) – The session that this factory should create test resources for.

`client`(*service_name*, **args*, ***kwargs*)
Create a botocore client that will use Localstack.

Arguments are the same as `botocore.session.Session.create_client()`.

`default_session`
Return a default botocore Localstack Session.

Most applications only need one Session.

`patch_botocore`()
Context manager that will patch botocore to use Localstack.

Since boto3 relies on botocore to perform API calls, this method also effectively patches boto3.

`session`(**args*, ***kwargs*)
Create a botocore Session that will use Localstack.

Arguments are the same as `botocore.session.Session`.

class `pytest_localstack.contrib.botocore.`**`DefaultCredentialProvider`**(*session=None*)
Provide some default credentials for Localstack clients.

`load`()
Return credentials.

class pytest_localstack.contrib.botocore.**LocalstackEndpointResolver**(*localstack_session*, *endpoints*)

Resolve AWS service endpoints based on a LocalstackSession.

construct_endpoint(*service_name*, *region_name=None*)

Resolve an endpoint for a service and region combination.

get_available_endpoints(*service_name*, *partition_name='aws'*, *allow_non_regional=False*)

List the endpoint names of a particular partition.

get_available_partitions()

List the partitions available to the endpoint resolver.

valid_regions

Return a list of regions we can resolve endpoints for.

class pytest_localstack.contrib.botocore.**Session**(*localstack_session*, **args*, ***kwargs*)

A botocore Session subclass that talks to Localstack.

create_client(**args*, ***kwargs*)

Create a botocore client.

pytest_localstack.contrib.botocore.**contribute_to_module**(*pytest_localstack*)

Add `patch_fixture()` to `pytest_localstack`.

pytest_localstack.contrib.botocore.**contribute_to_session**(*session*)

Add `BotocoreTestResourceFactory` to `LocalstackSession`.

pytest_localstack.contrib.botocore.**create_credential_resolver**()

Create a credentials resolver for Localstack.

pytest_localstack.contrib.botocore.**patch_fixture**(*scope='function'*, *services=None*, *autouse=False*, *docker_client=None*, *region_name=None*, *kinesis_error_probability=0.0*, *dynamodb_error_probability=0.0*, *container_log_level=10*, *localstack_version='latest'*, *auto_remove=True*, *pull_image=True*, *container_name=None*, ***kwargs*)

Create a pytest fixture that temporarily redirects all botocore sessions and clients to a Localstack container.

This is not a fixture! It is a factory to create them.

The fixtures that are created by this function will run a Localstack container and patch botocore to direct traffic there for the duration of the tests.

Since boto3 uses botocore to send requests, boto3 will also be redirected.

Parameters

- **scope** (*str, optional*) – The pytest scope which this fixture will use. Defaults to `"function"`.
- **services** (*list, dict, optional*) – One of
  - A `list` of AWS service names to start in the Localstack container.
  - A `dict` of service names to the port they should run on.

  Defaults to all services. Setting this can reduce container startup time and therefore test time.

- **autouse** (*bool, optional*) – If `True`, automatically use this fixture in applicable tests. Default: `False`
- **docker_client** (`DockerClient`, optional) – Docker client to run the Localstack container with. Defaults to `docker.client.from_env()`.
- **region_name** (*str, optional*) – Region name to assume. Each Localstack container acts like a single AWS region. Defaults to `"us-east-1"`.
- **kinesis_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into Kinesis API responses.
- **dynamodb_error_probability** (*float, optional*) – Decimal value between 0.0 (default) and 1.0 to randomly inject ProvisionedThroughputExceededException errors into DynamoDB API responses.
- **container_log_level** (*int, optional*) – The logging level to use for Localstack container logs. Defaults to `logging.DEBUG`.
- **localstack_version** (*str, optional*) – The version of the Localstack image to use. Defaults to `"latest"`.
- **auto_remove** (*bool, optional*) – If `True`, delete the Localstack container when it stops. Default: `True`
- **pull_image** (*bool, optional*) – If `True`, pull the Localstack image before running it. Default: `True`
- **container_name** (*str, optional*) – The name for the Localstack container. Defaults to a randomly generated id.
- ****kwargs** – Additional kwargs will be passed to the *`LocalstackSession`*.

Returns A `pytest fixture`.

boto3

pytest-localstack extensions for boto3.

class `pytest_localstack.contrib.boto3.`**Boto3TestResourceFactory**(*localstack_session*)

Create boto3 clients and resources to interact with a *`LocalstackSession`*.

Parameters **localstack_session** (*`LocalstackSession`*) – The session that this factory should create test resources for.

client(*service_name*)

Return a patched boto3 Client object that will use localstack.

Arguments are the same as `boto3.client()`.

default_session

Return a default boto3 Localstack Session.

Most applications only need one Session.

resource(*service_name*)

Return a patched boto3 Resource object that will use localstack.

Arguments are the same as `boto3.resource()`.

session(**args*, ***kwargs*)

Return a boto3 Session object that will use localstack.

Arguments are the same as `boto3.session.Session`.

`pytest_localstack.contrib.boto3.`**contribute_to_session**(*session*)

Add *`Boto3TestResourceFactory`* to *`LocalstackSession`*.

CHAPTER 7

Change Log

7.1 0.2.0 (2019-03-06)

- Use botocore to determine default AWS region (will us-east-1 fallback).
- Replace use of *pytest.config* with *pytest_configure()* hook.

7.2 0.1.5 (2018-08-17)

- Fix a bug involving our patched botocore Session trying to access *_internal_components* and getting *_components* instead.

7.3 0.1.4 (2018-08-03)

- Fix pinned install requirements conflict between pytest and pluggy.

7.4 0.1.3 (2018-07-17)

- Fix for botocore >= 1.10.58.

7.5 0.1.2 (2018-06-22)

- Broke out LocalstackSession into RunningSession which doesn't start localstack itself.

7.6 0.1.1 (2018-04-23)

- Fixed bug where patched botocore clients wouldn't populated the *_exceptions* attribute.

7.7 0.1.0 (2018-03-13)

- Initial release

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index

V