lambda-toolkit Documentation

Release 0.3

Lucio Veloso

Aug 09, 2017

Getting started:

CHAPTER 1

lambda-toolkit

- Now compatible with Python 2 and 3.
- Now compatible with Linux, Mac and Windows.

Top Features

- Invoke locally lambdas in your machine with simulated events or real events
- Easy import and deploy your lambdas in different regions (v0.3.0)
- Supporting Lambda Environment Variables (Even in local execution) (v0.3.3)

Contributing

We are open to contributions. Also let we know if you need some feature that is not implemented or it's not working properly. Please feel free to open an Issue in the Github project requesting fixes or new features.

Easy to use and install

We provide a step-by-step installation guide with installation manual and how to use.

Access now here.

Documentation

http://lambda-toolkit.readthedocs.io/

GitHub Project

https://github.com/lucioveloso/lambda-toolkit

Installation:

$ pip install lambda-toolkit -U

CHAPTER 2

Lambda-toolkit offers a lambda proxy

Using lambda-toolkit proxy you're able to debug real events in real time in your own machine.

See how it works.

After you create a proxy, you're able even to put breakpoints in your code, like the example below:

CHAPTER 3

Lambda-toolkit offers lambda tailing

Basically you're able to install the lambda-toolkit using pip. You can also install it manually, cloning the project from GitHub.

Hint: If you're not a developer and you're not planning to contribute developing in lambda-toolkit, **we do recommend to use pip** installation mode..

General Prerequisites

Basically, to start to use the lambda-toolkit, you must have at least:

- Python 2 or 3
- Package Management System (`pip`)
- AWS Credentials

Python

Python is required to use lambda-toolkit. By this way, you should make sure that your system has a proper Python installation available:

```
$ python --version
Python x.x.x
```

If your system does not support Python, you can install here.

Package Management System (pip)

pip is required to use lambda-toolkit. By this way, you should make sure that your system has a proper pip installed:

```
$ pip --version
pip 9.0.1 from /Library/Python/2.7/site-packages/pip-9.0.1-py2.7.egg (python 2.7)
```

If you system does not has pip installed, you can install here.

AWS Credentials

Lambda-toolkit can read yours credentials from the **system environment variables** or **credential files**.

If you're already using aws cli for example, credential files were generated when the command `aws configure` was executed.

Hint: Lambda-toolkit tries first to read the environment variables, and if it is not configured, lambda-toolkit reads the credential files.

Warning: We do recommend to use credential files.

Using credential files

Make sure that you have the files:

- `~/.aws/credentials`
- `~/.aws/config`

For example:

~/.aws/credentials:

```
[default]
aws_access_key_id = AAAAAAAAAAAAAAAAAAAA
aws_secret_access_key = AWS_ACCESS_SECRET_KEY__KEEP_IT_SAFE
```

~/.aws/config:

```
[default]
region = eu-west-1
```

Hint: As you can see, the example shows how to create a default credential. You can create others, and then, you can use the environment variable `AWS_PROFILE` to choice a specific one. You can also overwrite your region option, setting the environment variable `AWS_REGION`.

Using environments variables

Specify the env variables below:

- `AWS_ACCESS_KEY_ID`
- `AWS_SECRET_ACCESS_KEY`
- `AWS_REGION`

For example:

```
$ export AWS_ACCESS_KEY_ID="AAAAAAAAAAAAAAAAAAAA"
$ export AWS_SECRET_ACCESS_KEY="AWS_ACCESS_SECRET_KEY__KEEP_IT_SAFE"
$ export AWS_REGION="us-east-1"
```

Hint: Remember that you must `export` the variables.

Warning: Lambda-toolkit only uses the environment variables, if the 3 variables are available.

Installing

Installing using pip

The installation with pip is quick and simple. For common installation, use the command below:

```
$ sudo -H pip install lambda-toolkit
$ lt --help
```

Hint: If you want uninstall lambda-toolkit, just run `sudo pip uninstall lambda-toolkit`. To update, run `sudo pip install lambda-toolkit -U`

Cloning the repository manually

Installing from repository is not to common users, but it is also another option. To install from repository you also need to have the `git` client installed.

The first step is clone the repository:

```
$ git clone https://github.com/lucioveloso/lambda-toolkit
```

Install the requirements using pip:

```
$ pip install -r lambda-toolkit/requirements-user.txt
$ lambda-toolkit/bin/lt --help
```

And then, you are able to run the lambda-toolkit from current user:

Creating a lambda project

The first step to start to use the lambda-toolkit is creating or importing a lambda project. Lambda-toolkit provides a command called `project`, that you can invoke to create a lambda project running the command `lt project create`:

```
$ lt project create -p myLambdaProject
Initializing lambda-toolkit CLI (v0.2.10) - Region: eu-west-1 - Auth: file
[INFO] Creating the project lambda-toolkit folder '/Users/glucio/lambda-toolkit/
↪lambdas/myLambdaProject_eu-west-1'
[INFO] Project 'myLambdaProject' [python2.7] has been created.
```

Now I have my project pre-configured in `lambdas/myLambdaProject_eu-west-1`:

```
$ ls /Users/glucio/lambda-toolkit/lambdas/myLambdaProject_eu-west-1
__init__.py index.py
```

Note: Lambda-toolkit created also the file `__init__.py`. Lambda-toolkit creates this file to make sure that it will be possible to load this project as module.

Now, if we list our environment, we already can see this project:

```
$ lt list
Initializing lambda-toolkit CLI (v0.2.10) - Region: eu-west-1 - Auth: file
[INFO]
[INFO] User Projects (Lambda Functions):
[INFO] - Project:     lambda-list-buckets      Deployed: True                     ␣
↪Runtime:  python2.7
[INFO] - Project:     myLambdaProject          Deployed: False                    ␣
↪Runtime:  python2.7
```

To see the project command help, just type `lt project --help`:

```
Manage Projects
******************************************

 Examples:

 $ lt project list
 $ lt project create -p myProject [--runtime nodejs6.10]
 $ lt project delete -p myProject
 $ lt project deploy -p myProject [--rolename arn:xxx:yyyy]
 $ lt project undeploy -p myProject
 $ lt project deploy-all [--rolename arn:xxx:yyyy]
 $ lt project undeploy-all
 $ lt project import -p myAwsProject
 $ lt project import-all
 $ lt project list-aws
```

Hint: For all the commands, you can use `--help`. For example: `lt invoke --help` or `lt queue --help`.

Listing AWS lambda projects

To list all existing AWS lambda projects in your AWS environment, you can use the command `lt project list-aws`:

```
$ lt project list-aws
Initializing lambda-toolkit CLI (v0.2.10) - Region: eu-west-1 - Auth: file
[INFO] AWS Projects (Lambda Functions):
[INFO] - Project:     abcd                      Imported: False                    ␣
↪Runtime:  python2.7
[INFO] - Project:     s3_resources              Imported: False                    ␣
↪Runtime:  python2.7
[INFO] - Project:     lambda-list-buckets       Imported: True                     ␣
↪Runtime:  python2.7
```

Importing an existing AWS lambda project

To import an existing lambda project in your AWS environment, you can use the command `lt project import`:

```
$ lt project import -p s3_resources
Initializing lambda-toolkit CLI (v0.2.10) - Region: eu-west-1 - Auth: file
[INFO] Creating the project lambda-toolkit folder '/Users/glucio/lambda-toolkit/
↪lambdas/s3_resources_eu-west-1'
[INFO] Project s3_resources imported.
```

Warning: Note that if you import an proxy that you already have inside lambda-toolkit, it will overwrite your local project. It can be very useful if you wish to update your local project from the lambda in the AWS, but it also can make you lose data.

Deploying a project in AWS

To deploy a project to your AWS environment, you can use the command `lt project deploy`:

```
$ lt project deploy -p myLambdaProject
Initializing lambda-toolkit CLI (v0.2.10) - Region: eu-west-1 - Auth: file
[INFO] Lambda project myLambdaProject was created and deployed.
```

Note: Note that I didn't provide the option `--rolename` that is required. It happened due I previously had configured a default role using the command `lt role --set-default`.

Warning: If you already have a lambda project with this name in your AWS environment, it will be overwritten.

Setting a default role

To set a default role to be used always that you do not provide a rolename to deploy a proxy or a lambda project, you can use the command `lt role`:

```
$ lt role set-default --rolename arn:aws:iam::432811670411:role/service-role/
↪myRoleLambda
Initializing lambda-toolkit CLI (v0.2.10) - Region: eu-west-1 - Auth: file
[INFO] Role 'arn:aws:iam::123456789000:role/service-role/myRoleLambda' is set as␣
↪default now.
```

Hint: If you define a default role, but you use the `--rolename` to deploy a lambda project, the rolename that you specify will preponderate.

Invoking a local lambda

To invoke a lambda project locally in your machine, you can use the command `lt project invoke`:

```
$ lt invoke local --projectname myLambdaProject --event-file helloworld.json
Initializing lambda-toolkit CLI (v0.2.10) - Region: eu-west-1 - Auth: file
[INFO] Importing project myLambdaProject
Hi, I'm here. Lambda-proxy is working. =)
AWS Event ID: 11111111-1111-1111-1111-111111111111
Event Body: {"key3": "value3", "key2": "value2", "key1": "value1"}
```

Hint: You can customize or add new events file in the folder `~/lambda-toolkit/invoke/events/`

Invoking a remote lambda

To invoke a lambda project remotely, you can use the command `lt project invoke remote`:

```
$ lt invoke remote --event-file helloworld.json --projectname myLambdaProject
Initializing lambda-toolkit CLI (v0.2.11) - Region: eu-west-1 - Auth: file
[INFO] Invoking the project myLambdaProject
START RequestId: 5a380d00-66c6-11e7-8119-9b430b7e8688 Version: $LATEST
Hi, I'm here. Lambda-proxy is working. =)
AWS Event ID: 5a380d00-66c6-11e7-8119-9b430b7e8688
Event Body: {"key3": "value3", "key2": "value2", "key1": "value1"}
END RequestId: 5a380d00-66c6-11e7-8119-9b430b7e8688
REPORT RequestId: 5a380d00-66c6-11e7-8119-9b430b7e8688      Duration: 0.57 ms      ␣
↪Billed Duration: 100 ms         Memory Size: 128 MB     Max     Memory Used: 29 MB
```

Hint: You can invoke remotely your lambda-toolkit proxy, providing the argument `--proxyname` instead `--projectname`.

Tailing a remote lambda

To tail a remote lambda project, you can use the command `lt tail cloudwatch`:

```
$ lt tail cloudwatch --loggroupname "/aws/lambda/myLambdaProject"
Initializing tail-toolkit CLI (v0.0.5) - Region: eu-west-1
Collecting logs in real time, starting from 5 minutes ago
START RequestId: 8b690d74-66de-11e7-b54e-2d48a73dcaf9 Version: $LATEST
Hi, I'm here. Lambda-proxy is working. =)
AWS Event ID: 8b690d74-66de-11e7-b54e-2d48a73dcaf9
Event Body: {"account": "123456789000", "region": "eu-west-1", "detail": {"state":
↪"running", "instance-id": "i-03169cf0533d7d000"}, "detail-type": "EC2 Instance␣
↪State-change Notification", "source": "aws.ec2", "version": "0", "time": "2017-07-
↪12T08:46:05Z", "id": "812d642c-5f46-4588-9dde-bfa4478a4e78", "resources": [
↪"arn:aws:ec2:eu-west-1:123456789000:instance/i-03169cf0533d7d000"]}
END RequestId: 8b690d74-66de-11e7-b54e-2d48a73dcaf9
REPORT RequestId: 8b690d74-66de-11e7-b54e-2d48a73dcaf9       Duration: 0.69 ms      ␣
↪Billed Duration: 100 ms         Memory Size: 128 MB     Max Memory Used: 29 MB
*************
```

Important: Please note that tail can be used to any log group name in your cloudwatch environment. To tail your lambda functions you should append the lambda log group prefix `/aws/lambda/<your lambda function name>`

Note: If you want to debug your remote lambda function, you should use the `receiver` command instead the `tail`.

Debugging a Lambda

To debug real events in real time, you should have at least:

- A queue (to store the events)
- A proxy (to forward the events to the queue)
- A lambda project (to process the events)

Creating a queue

To create a queue, you can use the command `lt project queue create`:

```
$ lt queue create -q myQueue
Initializing lambda-toolkit CLI (v0.2.11) - Region: eu-west-1 - Auth: file
[INFO] The queue 'myQueue.fifo' has been created.
```

Hint: In any moment, you can use `lt list` to see all resources in your lambda-toolkit environment.

Deploying a proxy

To deploy a proxy, you can use the command `lt project proxy deploy`:

```
$ lt proxy deploy -p myProxy -q myQueue
Initializing lambda-toolkit CLI (v0.2.11) - Region: eu-west-1 - Auth: file
[INFO] Lambda proxy myProxy created proxying requests to myQueue.fifo
```

Verifying requisites

Let's list our environment, to check if we already have the **queue**, the **proxy** and the **lambda project**.

```
$ lt list
Initializing lambda-toolkit CLI (v0.2.11) - Region: eu-west-1 - Auth: file
[INFO]
[INFO] User Projects (Lambda Functions):
[INFO] - Project:     s3_resources             Deployed: True                  ␣
↪Runtime:  python2.7
[INFO] - Project:     myLambdaProject          Deployed: True                  ␣
↪Runtime:  python2.7
[INFO] - Project:     lambda-list-buckets      Deployed: True                  ␣
↪Runtime:  python2.7
[INFO]
[INFO] SQS (Queues):
[INFO] - Queue name:  myQueue.fifo             Used by:  myProxy
[INFO]
[INFO] Proxies (Lambda proxies):
[INFO] - Proxy name:  myProxy                  Queue:    myQueue.fifo          ␣
↪Runtime:  python2.7
```

Note: Note that now we have the proxy `myProxy` forwarding the event to `myQueue`.

Debugging

Now that you already have the all requisites to run a `receiver`, we can execute it to start to collect our real data in real time:

```
$ lt receiver collect --projectname myLambdaProject --sqsname myQueue
Initializing lambda-toolkit CLI (v0.2.11) - Region: eu-west-1 - Auth: file
[INFO] Importing project myLambdaProject
[INFO] Starting the receiver using the queue myQueue.fifo
.........
```

Attention: You can run this command inside your IDE in Debug Mode. By this way you will be able to set break points and debug all data. If you want to just see the logs in real time, you can use the `lt tail` instead the `lt receiver`.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search