Serverless Framework

Context

Documentation: https://www.serverless.com/framework/docs/

Functions

- Independent unit of deployment
- Code written & deployed to the cloud
- Performs a single job
- e.g.
  - Saving user to database
  - Process file in database
  - Perform scheduled task

Events

- Triggers **functions** from occurring
- e.g.
  - Endpoint request
  - CRON trigger
  - On upload

Resources

- Infrastructure **components** which **functions** use
- e.g.
  - AWS S3 Bucket
  - AWS SNS

Services

- Unit of organization
- e.g. project file
- Defines grouping of function, events, & resources

Setup

1. Install global serverless **npm** package

```
npm install -g serverless
```

2. Run setup command

```
serverless
```

Extra: Upgrade package

```
npm update -g serverless
```

Process

Endpoint Set-up

Define endpoint in `serverless.yml` that will trigger your serverless function

```
functions:
  hello:
    handler: handler.hello
    # Add the following lines:
    events:
      - http:
          path: hello
          method: post
```

Deploy the Service

Deploy all changes within your service at the same time via **CloudFormation**

```
serverless deploy -v
```

▼ List of Options...

- `--config` or `-c` names configuration file other than `serverless.yml`
- `--noDeploy` or `-n` skips deployment steps and leaves artifacts in `.serverless` directory
- `--stage` or `-s` stage in your service that you want to deploy to
- `--region` or `-r` region in that stage you want to deploy to
- `--package` or `-p` path to pre-packaged directory and skip package step
- `--verbose` or `-v` shows all stack events during deployment

- `--force` force deployment
- `--function` or `-f` invoke `deploy function`
- `--conceal` hides secrets from output

Can be found in `sls deploy` output

Test the Service

Target URL Endpoint

```
curl -X POST https://TARGET_ENDPOINT.com/endpoint
```

Invoke Service Function

```
serverless invoke -f FUNCTION_NAME -l
```

Fetch Function Logs

```
serverless logs -f FUNCTION_NAME -t
```

Removing the Service

Removes functions, events, & resources created

```
serverless remove
```

AWS Specific Setup

Credentials

1. Create AWS account; *if needed*
2. Create **IAM User** & **Access Key**
   - Stored in environment variables
3. Configure IAM permissions

Using AWS Access Keys

Method 1 (Quick Setup)

Exported as *environment variables* to be accessible to *serverless* & *AWS SDK*

```
export AWS_ACCESS_KEY_ID=<your-key-here>
export AWS_SECRET_ACCESS_KEY=<your-secret-key-here>
# AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY are now available for serverless to use
serverless deploy

# 'export' command is valid only for unix shells. In Windows - use 'set' instead of 'export'
```

▼ Notes: If you are using a self-signed certificate you'll need to do one of the following...

```
# String example:
# if using the 'ca' variable, your certificate contents should replace the newline character with '\n'
export ca="-----BEGIN CERTIFICATE-----...-----END CERTIFICATE-----"
# or multiple, comma separated
export ca="-----BEGIN CERTIFICATE-----...-----END CERTIFICATE-----,-----BEGIN CERTIFICATE-----...-----END CERTIFICATE-----"

# File example:
# if using the 'cafile' variable, your certificate contents should not contain '\n'
export cafile="/path/to/cafile.pem"
# or multiple, comma separated
export cafile="/path/to/cafile1.pem,/path/to/cafile2.pem"

# 'export' command is valid only for unix shells. In Windows - use 'set' instead of 'export'
```

Method 2 (AWS Profiles)

Serverless Framework provides a way to configure AWS profiles using `serverless config credentials`

```
serverless config credentials --provider aws --key AWS_ACCESS_ID --secret AWS_SECRET_KEY --profile test-profile-name
```

▼ List of Options...

- `--provider` or `-p` The provider (in this case `aws`). **Required**.
- `--key` or `-k` The `aws_access_key_id`. **Required**.
- `--secret` or `-s` The `aws_secret_access_key`. **Required**.
- `--profile` or `-n` The name of the profile which should be created.
- `--overwrite` or `-o` Overwrite the profile if it exists.

Create New Service

```
serverless create --template aws-nodejs --path test-service
```

▼ List of Options...

- `--template` or `-t` from an available template **Required if none specified**
- `--template-url` Creates service using custom template (*URL*) **Required if none specified**
- `--path` Creates service in new folder
- `--name` or `-n` Names services under `serverless.yml`