ToiletPaper #134

AWS Lambdas lokal debuggen? Serverless-offline to the rescue

Autor: Robert Gruner / Software Engineer / Standort Leipzig

✗ Problem

Ihr seid auch in der Neuzeit angekommen und schreibt euren Business Code jetzt serverless? Super! Die einzelnen Lambda-Funktionen laufen in der Amazon-Infrastruktur und man muss nur ein paar Zeilen JavaScript in der AWS Console schreiben. Aber will man das? Vielleicht doch lieber eine IDE nutzen und TypeScript schreiben? Geht klar. Aber woher weiß ich dann, dass mein Lambda-Code auch funktioniert, ohne ihn zu AWS zu deployen?

✔ Lösung

Nutzt das Tool **serverless** (kurz: sls) in Kombination mit dem Plugin **serverless-offline**! Allgemeint hilft euch sls bei der lokalen Entwicklung der Lambdas, beim Deployment zu AWS etc. Es gibt etliche Plugins – eins davon ist serverless-offline. Dieses Plugin simuliert ein lokales AWS Api-Gateway. Mit diesem Gateway werden die einzelnen Lambda-Funktionen verknüpft und ihr könnt es lokal z. B. mittels Postman oder IDEA Scratch File testen.

➔ Beispiel

1. In eurer sls Konfiguration müsst ihr http-Events für alle Lambdas definieren, welche ins lokale Gateway inkludiert werden sollen. Im TypeScript sieht das für eine *getCoffee* Funktion ungefähr so aus:

```
functions: {
    getCoffee: {
      events: [
        {
          // 'http' simuliert die RestApi von AWS
          http: {
            method: 'get',
            path: '/coffee',
          },
        },
      ],
      handler: 'src/config/handlers/getCoffee',
    },
}
```

2. Ihr legt euch ein **Nodejs** Run/Debug-Profil an, welches *sls offline* startet

```
Working directory: ~/Code/CoffeeLambdas
JavaScript file: ~/.nvm/versions/node/v12.14.1/bin/serverless # NPM-globale binary f. serverless
Application parameters: offline start --aws-profile jambit --stage dev
Environment variables: SLS_DEBUG=* # mehr Output aktivieren, um besser zu sehen, was passiert
```

3. Breakpoint im Code setzen und einen Request gegen das lokale Gateway feuern!
4. Happy Debugging!

✚ Weiterführende Aspekte

- https://www.serverless.com/framework/docs/providers/: Supportet nicht nur AWS, sondern auch Azure, Google etc.
- https://github.com/dherault/serverless-offline: Source des empfohlenen Plugins
- https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api-vs-rest.html: Unterscheidung der Api-Gateway-Typen
- https://www.jetbrains.com/help/idea/run-debug-configuration.html: Doku zur Run-/Debug-Profil-Konfiguration in IntelliJ IDEs
- https://www.jetbrains.com/help/idea/scratches.html: Doku zu Scratch Files