Java + AWS Lambda: montando um ambiente de desenvolvimento local com LocalStack

Thomás da Costa

Objetivo

Iremos demonstrar a configuração de um ambiente de desenvolvimento para criação de aplicações Serverless com AWS Lambda e Java e mostrar um exemplo de “Hello World” com Lambda.

Requisitos

Para o nosso tutorial, vamos precisar das seguintes ferramentas:

- LocalStack — https://localstack.cloud/
- AWS CLI — https://aws.amazon.com/pt/cli/
- SAM CLI — https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/serverless-sam-cli-install.html
- Java 11 — https://adoptium.net/
- Maven — https://maven.apache.org/install.html
- Docker — https://hub.docker.com/search/?type=edition&offering=community
- IntelliJ — https://www.jetbrains.com/pt-br/idea/
- AWS Toolkit — https://plugins.jetbrains.com/plugin/11349-aws-toolkit

Instalando LocalStack

Efetue a leitura do artigo abaixo para a instalação do LocalStack:

LocalStack: ambiente local para testar a sua aplicação AWS
Introdução
thomsdacosta.medium.com

Instalando o AWS SAM CLI

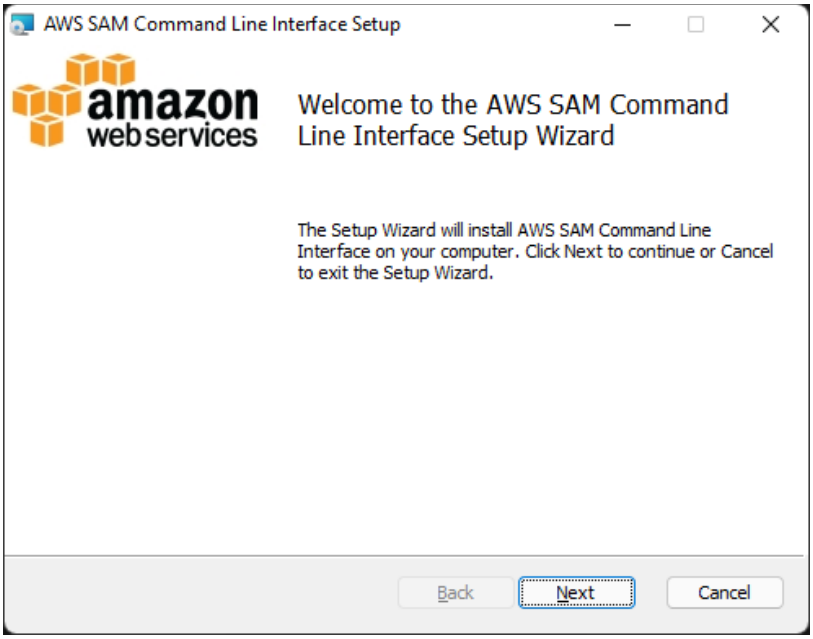
A Serverless Application Model Command Line Interface (SAM CLI) é uma extensão da AWS CLI que adiciona funcionalidade para criar e testar aplicações Lambda. Além disso, podemos gerar o pacote de instalação da aplicação Serverless. Utiliza o Docker em conjunto com uma imagem do Amazon Linux para executar as funções especificas dentro do seu ambiente. Para maiores detalhes sobre o AWS SAM consulte o link abaixo:

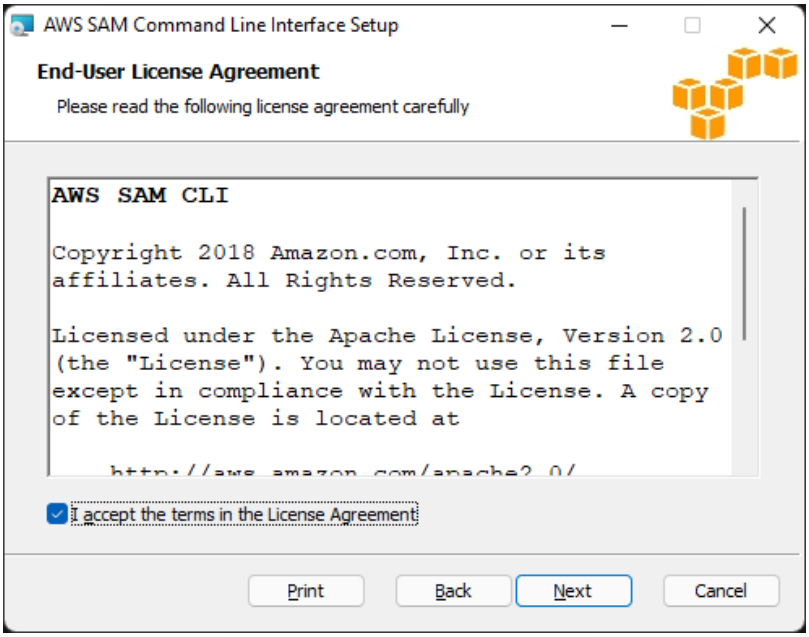
What is the AWS Serverless Application Model (AWS SAM)?
The AWS Serverless Application Model (AWS SAM) is an open-source framework that you can use to build serverless…
docs.aws.amazon.com

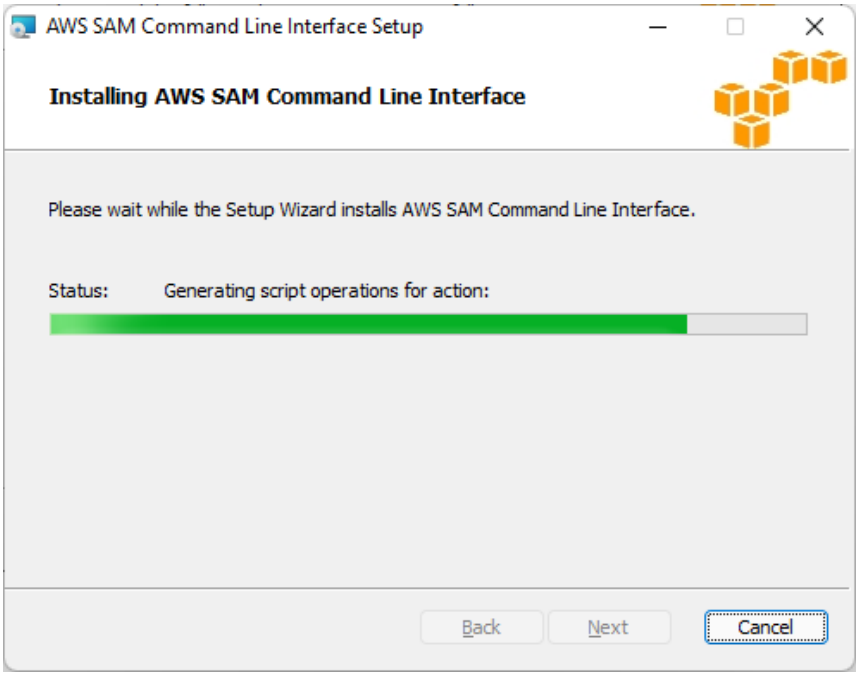
Instale o AWS SAM CLI de acordo com o seu sistema operacional:

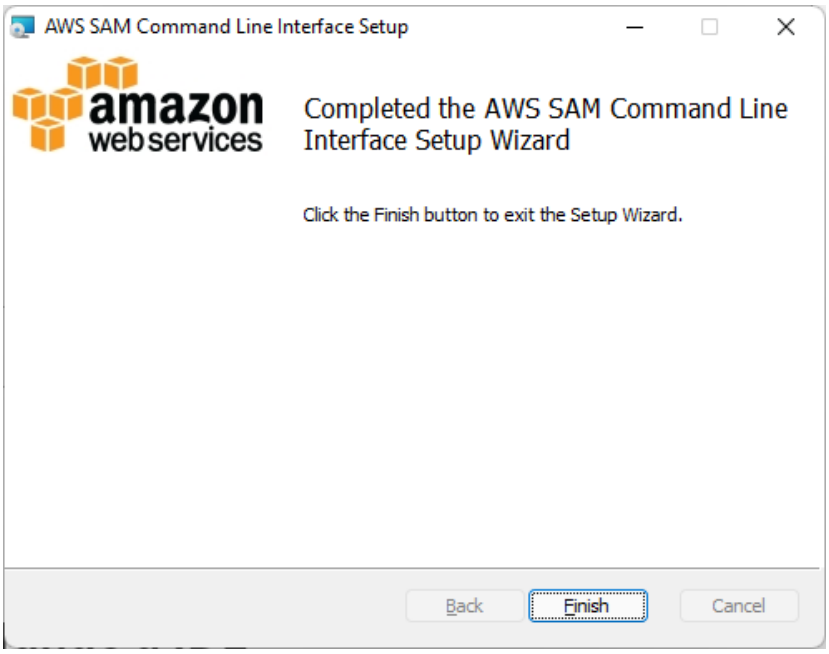
Installing the AWS SAM CLI
AWS SAM provides you with a command line tool, the AWS SAM CLI, that makes it easy for you to create and manage…
docs.aws.amazon.com

Execute o instalador e siga os passos de instalação. As imagens abaixo são referentes a instalação do Windows:



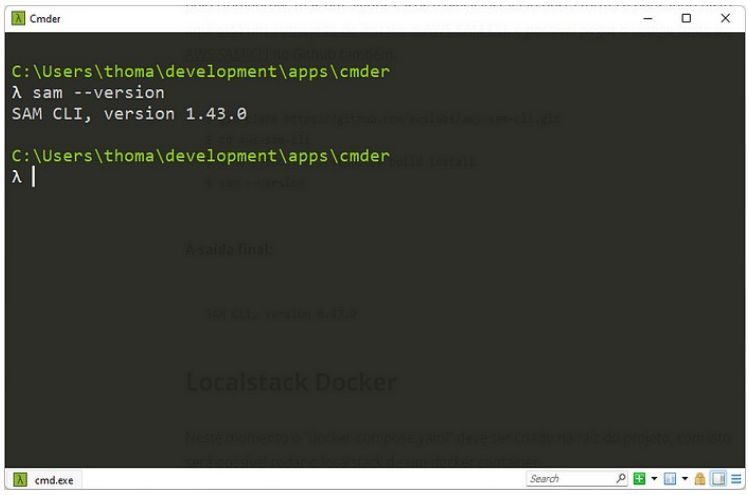






Verifique se a instalação está correta executando o comando abaixo em algum terminal do seu sistema operacional:

```
sam --version
```



Configurando a IDE

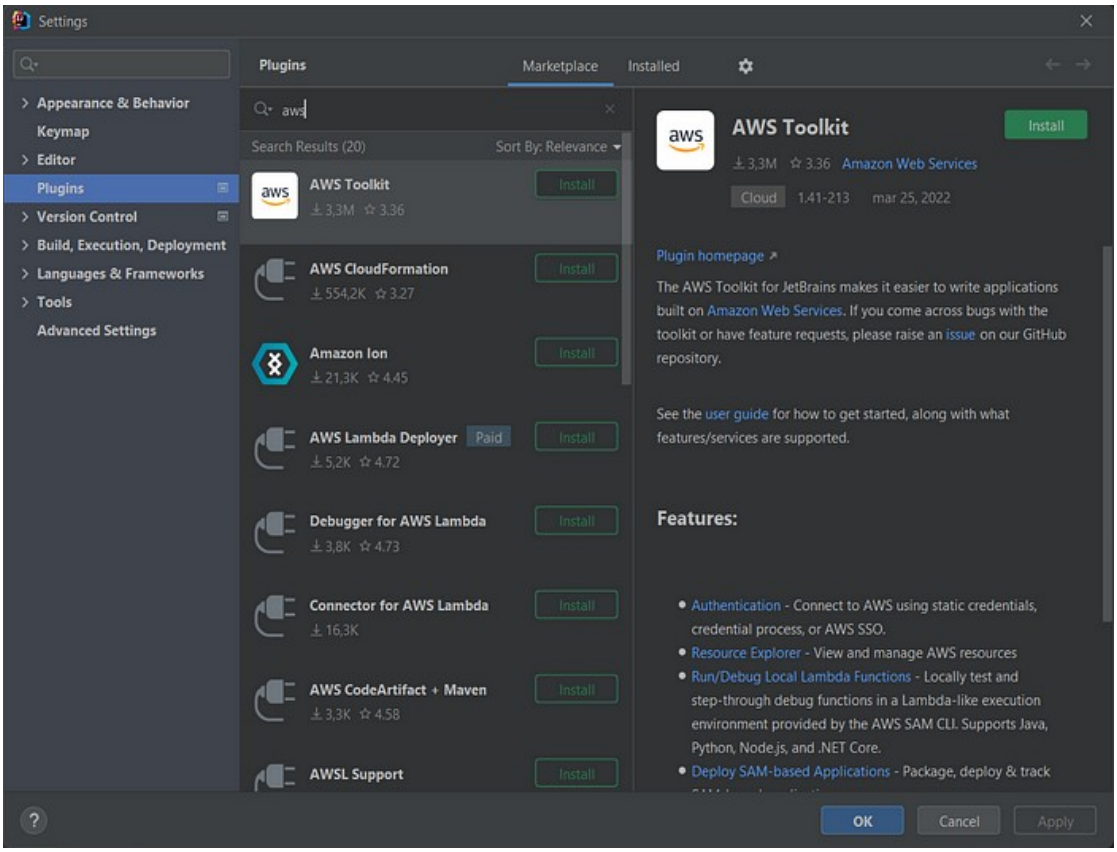
O **AWS Toolkit** é um **plugin** do **IntelliJ** que ajuda na utilização e desenvolvimento de aplicações usando os **recursos** da **AWS**. No nosso exemplo, o plugin será utilizado na criação do **projeto** com **Lambda**, na execução, compilação e depuração do código criado.

AWS Toolkit for IntelliJ

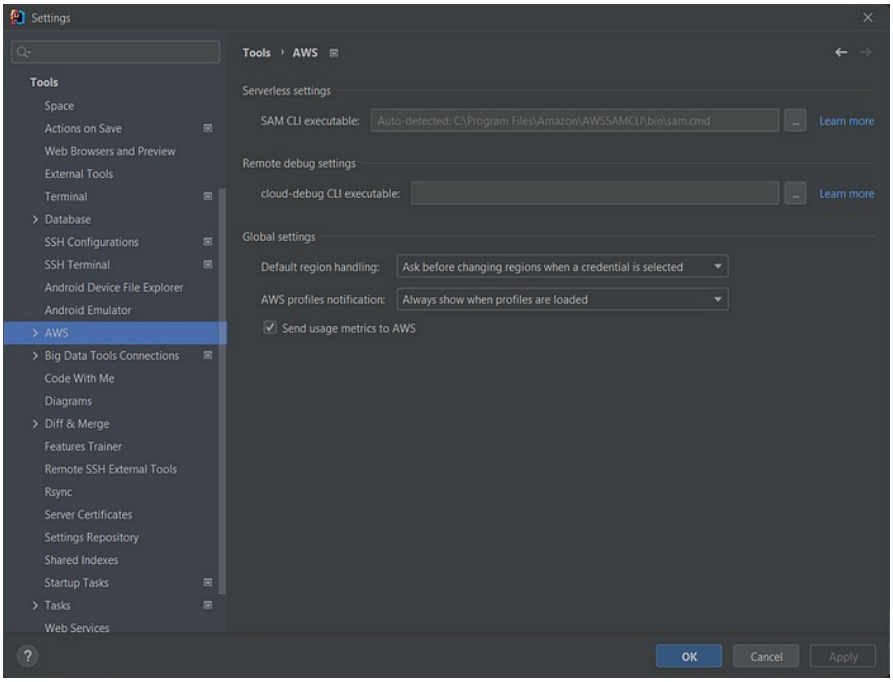
The AWS Toolkit for IntelliJ is an open source plug-in for the IntelliJ IDEA that will make it easier to create, debug…

aws.amazon.com

Para instalar o plugin, selecione *File->Settings->Plugins* e pesquise por **AWS Toolkit**:

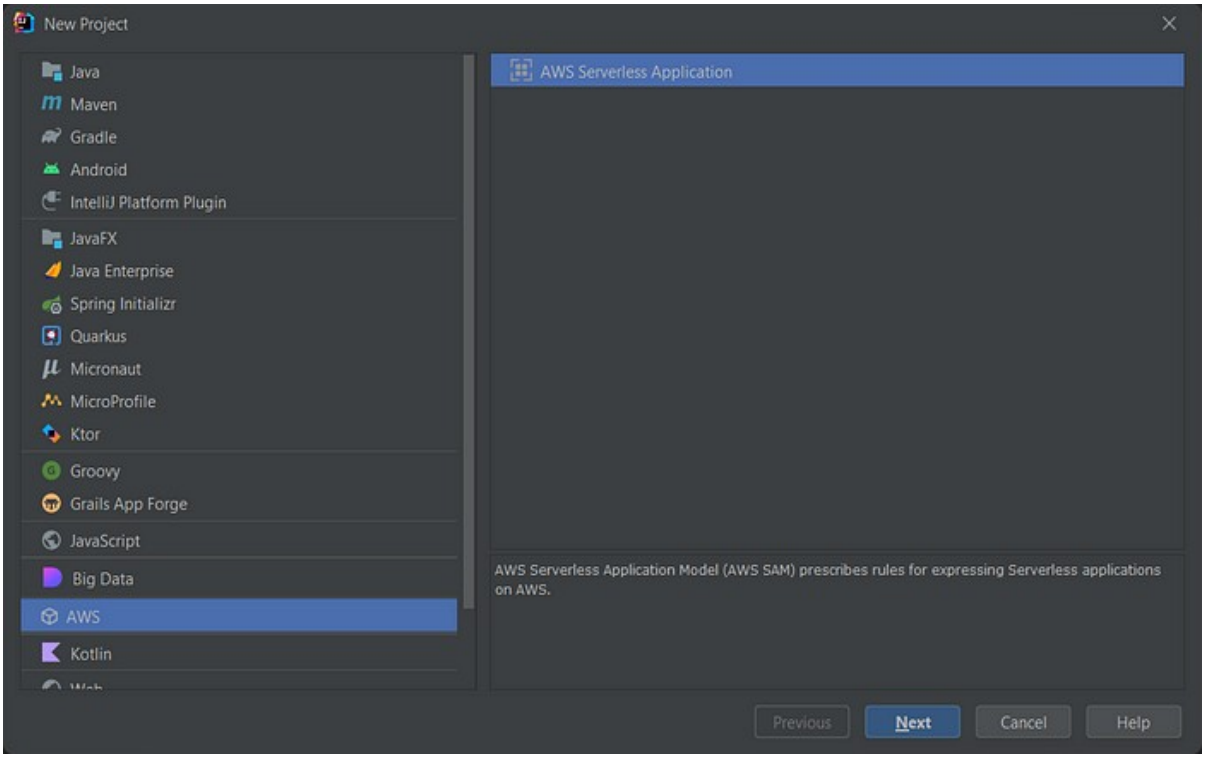


Em *File->Settings->Tools->AWS* configure a **localização** da **instalação** do **SAM CLI**. Normalmente a IDE detectará automaticamente a instalação:

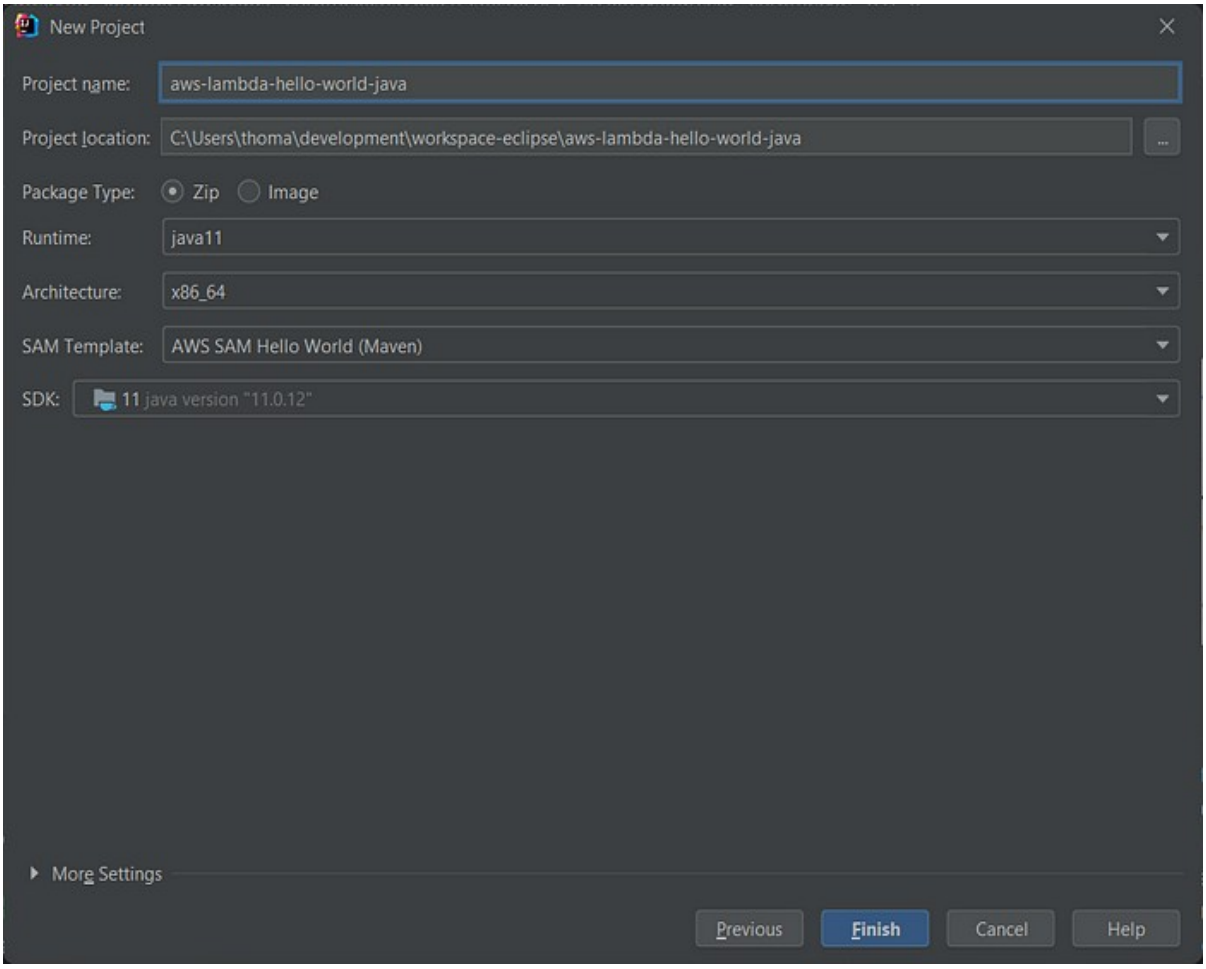


Criando um projeto

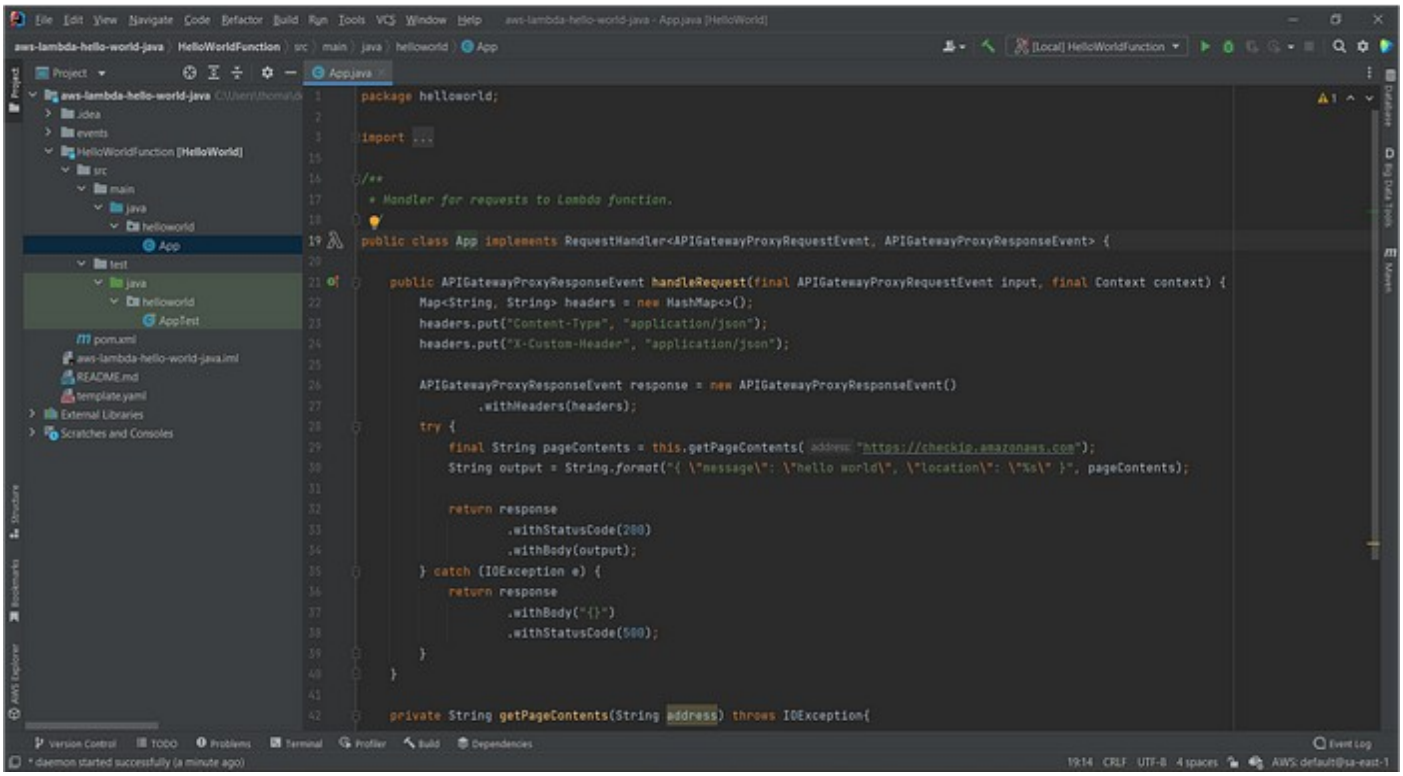
Após instalado o **plugin**, vamos **criar** o **primeiro** projeto **Lambda**. Selecione *File->New->Project->AWS:*



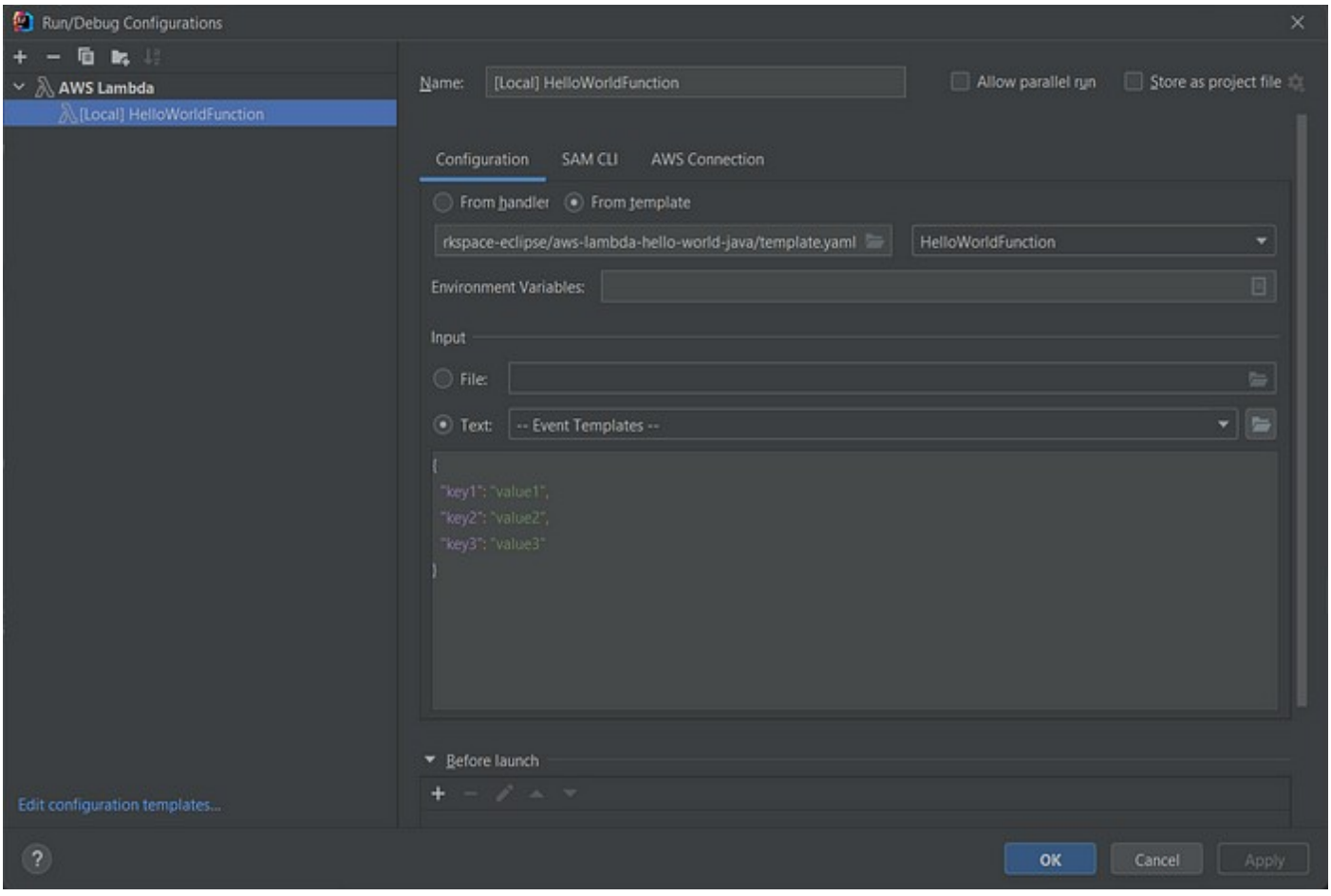
Coloque o nome do seu projeto e selecione as opções conforme mostrado abaixo:



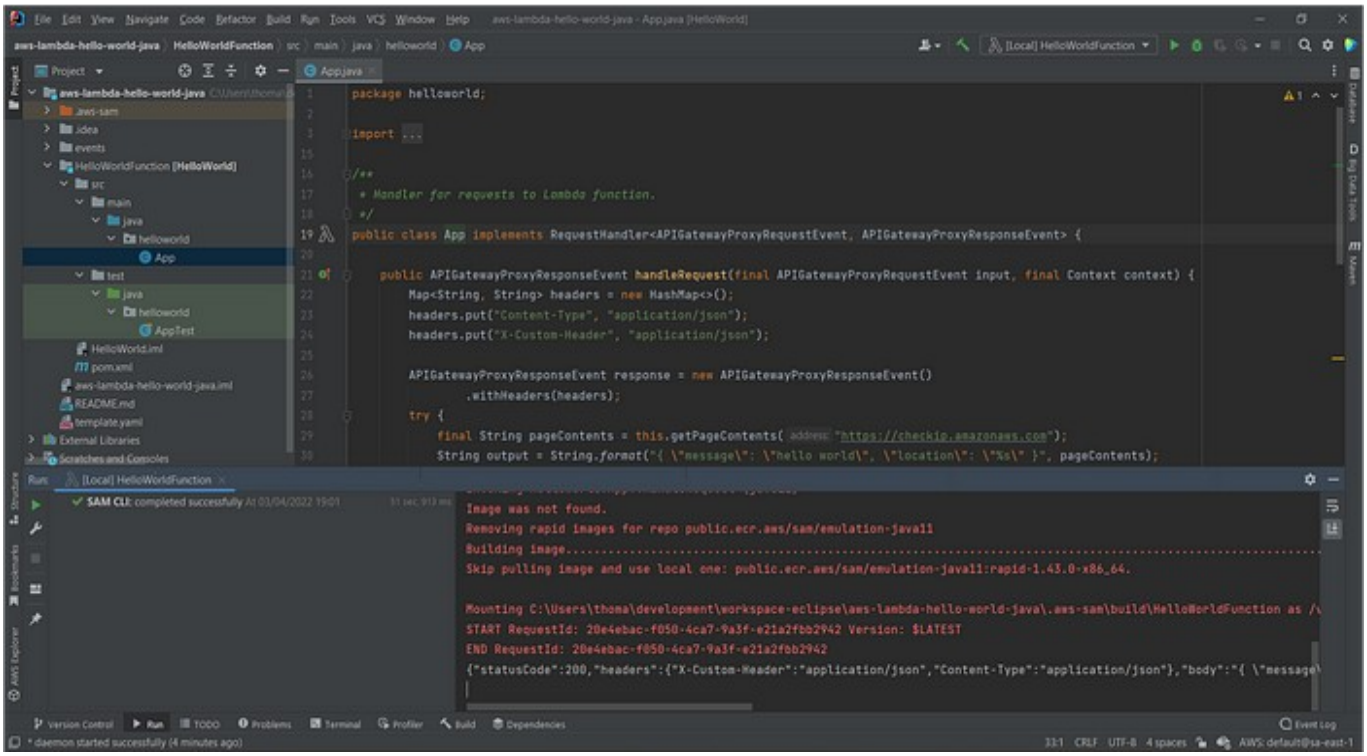
Após isso o projeto foi criado com um exemplo de uma aplicação Lambda:



Para executar o Lambda, devemos efetuar a configuração de execução conforme imagem abaixo:



Aplicação em execução e retornando a resposta:



Implantando no LocalStack

Para a **implantação** no **LocalStack**, devemos gerar o **pacote** que contém o **código** do **Lambda,** executando o seguinte comando:

```
mvn clean install
```

O pacote da instalação é gerado no seguinte diretório:

```
aws-lambda-hello-world-java\HelloWorldFunction\target\HelloWorld-1.0.jar
```

O Lambda precisa de um **IAM Role** para a sua **execução**:

```
aws --endpoint http://localhost:4566 --profile localstack \
  iam create-role \
  --role-name lambda-execution \
  --assume-role-policy-document "{\"Version\": \"2012-10-
17\",\"Statement\": [{ \"Effect\": \"Allow\", \"Principal\":
{\"Service\": \"lambda.amazonaws.com\"}, \"Action\": \"sts:AssumeRole\"}
]}"
```

Com a IAM Role criada, vamos incluir a policy de execução:

```
aws --endpoint http://localhost:4566 --profile localstack \
  iam attach-role-policy \
  --role-name lambda-execution \
  --policy-arn
arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
```

Efetuamos a **implantação** do **Lambda** no **LocalStack** com o seguinte comando:

```
aws --endpoint http://localhost:4566 --profile localstack \
  lambda create-function \
  --function-name HelloWorld \
  --zip-file fileb://HelloWorld-1.0.jar \
  --handler helloworld.App \
  --runtime java11 \
  --role arn:aws:iam::000000000000:role/lambda-execution
```

Executando o **Lambda** e verificando se foi instalado corretamente:

```
aws --endpoint http://localhost:4566 --profile localstack \
  lambda invoke \
  --function-name HelloWorld out.txt \
  --log-type Tail
```

Caso tenha **executado** com **sucesso**, o arquivo **out.txt** terá o seguinte conteúdo:

```
{"body":"{ \"message\": \"hello
world\", \"location\": \"200.0.0.0\" }","headers":{"Content-
Type":"application/json","X-Custom-Header":"application/
json"},"statusCode":200}
```

Proximo Artigo

Em um próximo artigo, iremos abordar com mais detalhe o desenvolvimento de uma aplicação serverless com Lambda, explicando os seus gatilhos e integração com outros recursos da AWS.

Até a próxima!

Prof. Thomás da Costa
thomasdacosta.com.br