AWS - SDK

lundi 20 février 2023 11:38

Formateur : Mohamed AIJJOU

https://www.redhat.com/fr/topics/cloud-native-apps/what-is-SDK

https://aws.amazon.com/fr/developer/tools/

SDK : définition

Un kit de développement logiciel (SDK) est un ensemble d'outils fourni avec une plateforme matérielle (généralement), un système d'exploitation ou un langage de programmation.

Découvrir un kit de développement logiciel Open Source →

Pourquoi utiliser un SDK ?

Il permet aux développeurs de logiciels de créer des applications propres à cette plateforme, ce système ou ce langage de programmation. C'est un peu comme une boîte à outils, ou comme le sachet d'outils fourni avec les éléments d'un meuble à assembler soi-même, mais pour développer une application. Il renferme tous les composants, ou outils de développement, nécessaires pour effectuer la tâche, et son contenu varie selon le fabricant.

https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html

Nous allons créer un compte de service pour l'exécution du SDK : ceci est une bonne pratique de sécurité car ce compte aura très peu de droit

1ère partie : création utilisateur IAM pour SDK

Create an IAM user

To complete this tutorial, you need to use credentials for an IAM user that has read and write access to Amazon S3. To make requests to Amazon Web Services using the SDK for Java 2.x, create an access key to use as credentials.

1. Sign in to the IAM console
2. In the navigation pane on the left, choose **Users**. Then choose **Add user**.
3. Enter *TestSDK* as the **User name** and select the **Programmatic access** checkbox. Choose **Next: Permissions**.
4. Under **Set permissions**, select **Attach existing policies directly**.
5. In the list of policies, select the checkbox for the **AmazonS3FullAccess** policy. Choose **Next: Tags**.
6. Choose **Next: Review**. Then choose **Create user**.
7. On the *Success* screen, choose **Download .csv**.

   The downloaded file contains the Access Key ID and the Secret Access Key for this tutorial. Treat your Secret Access Key as a password; save in a trusted location and do not share it.

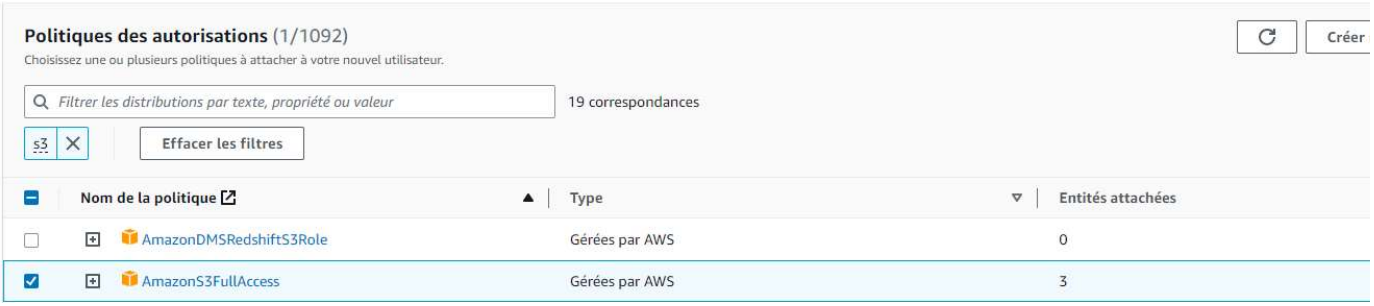
   Note
   You will **not** have another opportunity to download or copy the Secret Access Key.

Régler les autorisations

Ajouter un utilisateur à un groupe existant ou en créer un nouveau. L'utilisation de groupes est une bonne pratique pour gérer les autorisations des utilisateurs par fonctions de tâche. En savoir plus

Options d'autorisations

- ○ Ajouter un utilisateur à un groupe
  Ajouter un utilisateur à un groupe existant ou créer un nouveau groupe. Nous vous recommandons d'utiliser des groupes pour gérer les autorisations utilisateur par fonction de tâche.
- ○ Copier les autorisations
  Copiez toutes les appartenances à un groupe, les stratégies gérées attachées et les stratégies en ligne à partir d'un utilisateur existant.
- ● Attacher directement des politiques
  Attacher une politique gérée directement à un utilisat
  pratique consiste à attacher des politiques à un group
  ajouter l'utilisateur au groupe approprié.



Vérifier et créer

Vérifiez vos choix. Après avoir créé l'utilisateur, vous pouvez afficher et télécharger le mot de passe généré automatiquement, si cette option est activée.

Détails de l'utilisateur

Nom d'utilisateur	Type de mot de passe de la console	Demander la réinitialisation du mot de passe
benoit_SDK	Autogenerated	Oui

Résumé des autorisations

Nom	Type	Utilisé comme
AmazonS3FullAccess	Gérées par AWS	Stratégie des autorisations
IAMUserChangePassword	Gérées par AWS	Stratégie des autorisations

On n'oublie pas de créer les Access Key :

Récupérer les clés d'accès

Clé d'accès

Si vous perdez ou oubliez votre clé d'accès secrète, vous ne pouvez pas la récupérer. Au lieu de cela, créez une clé d'accès et rendez l'ancienne clé inactive.

Clé d'accès	Clé d'accès secrète
AKIAZKAF3QXWNBIGQXAU	K0xhc6NMosI5N4q1EZSmaq5BVMFYaNjHG2V5CfiR Masquer

Configure credentials

Configure your development environment with your Access Key ID and the Secret Access Key. The AWS SDK for Java uses this access key as credentials when your application makes requests to Amazon Web Services.

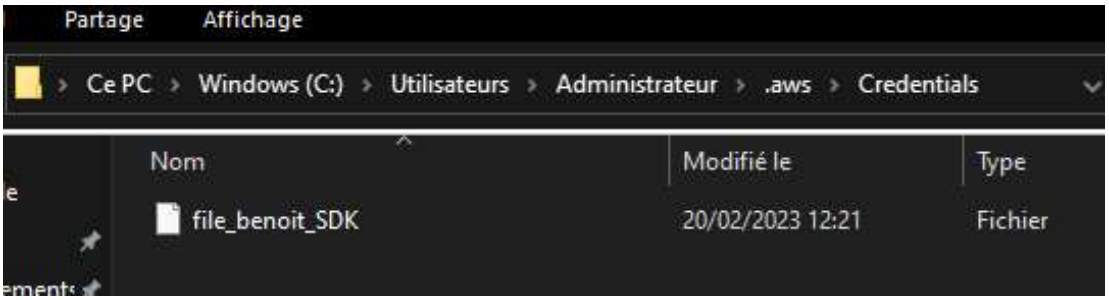
1. In a text editor, create a new file with the following code:

```
[default]
aws_access_key_id = YOUR_AWS_ACCESS_KEY_ID
aws_secret_access_key = YOUR_AWS_SECRET_ACCESS_KEY
region = us-east-1
```

2. In the text file you just created, replace `YOUR_AWS_ACCESS_KEY` with your unique AWS access key ID, and replace `YOUR_AWS_SECRET_ACCESS_KEY` with your unique AWS secret access key.
3. Save the file without a file extension. Refer to the following table for the correct location and file name based on your operating system.

Operating system	File name and location
Windows	`C:\Users\yourUserName\.aws\credentials`
Linux, macOS, Unix	`~/.aws/credentials`

```
 GNU nano 6.4                                  .aws/Credentials/file_benoit_SDK
[default]
aws_access_key_id = AKIAZKAF3QXWNBIGQXAU
aws_secret_access_key = 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
region = eu-west-1
```



Pour ce tp, il ne faut pas oublier de créer un bucket S3 privée

Ensuite, il faut installer maven et java dans notre instance :

```
[ec2-user@ip-10-0-1-69 ~]$ java -version
openjdk version "17.0.6" 2023-01-17 LTS
OpenJDK Runtime Environment Corretto-17.0.6.10.1 (build 17.0.6+10-LTS)
OpenJDK 64-Bit Server VM Corretto-17.0.6.10.1 (build 17.0.6+10-LTS, mixed mode, sharing)
[ec2-user@ip-10-0-1-69 ~]$ sudo yum install maven
```

Suite à cette installation, on clone le projet test pour l'exécution de notre TP (ne pas oublier d'installer git si nécessaire) :

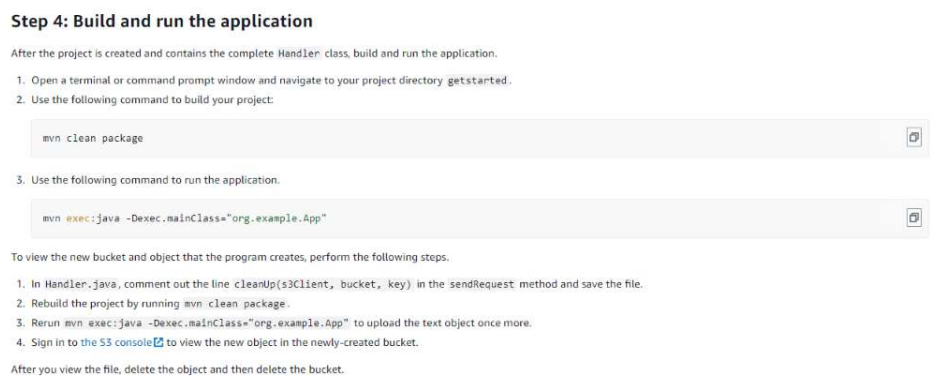
```
[ec2-user@ip-10-0-1-69 ~]$ git clone https://gitlab.com/mohamed_formation_test/s3-spring-demo.git
Cloning into 's3-spring-demo'...
remote: Enumerating objects: 67, done.
remote: Counting objects: 100% (67/67), done.
remote: Compressing objects: 100% (56/56), done.
remote: Total 67 (delta 8), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (67/67), 18.96 KiB | 6.32 MiB/s, done.
Resolving deltas: 100% (8/8), done.
[ec2-user@ip-10-0-1-69 ~]$ ls
s3-spring-demo
```

On va venir modifier notre fichier application.yml pour le faire pointer vers notre bucket :

```
 GNU nano 2.9.8                                  application.yml

amazonProperties:
   endpointUrl: https://s3.eu-west-1.amazonaws.com
   accessKey: AKIAZKAF3QXWNBIGQXAU
   secretKey: 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
   bucketName: bucket-tp-sdk-benoit
spring:
 servlet:
```

```
multipart:
   max-file-size: 100MB
   max-request-size: 100MB
```

On va ensuite lancer notre projet : **mvn clean install** et **mvn spring-boot:run**



Step 4: Build and run the application

After the project is created and contains the complete `Handler` class, build and run the application.

1. Open a terminal or command prompt window and navigate to your project directory `getstarted`.
2. Use the following command to build your project:

```
mvn clean package
```

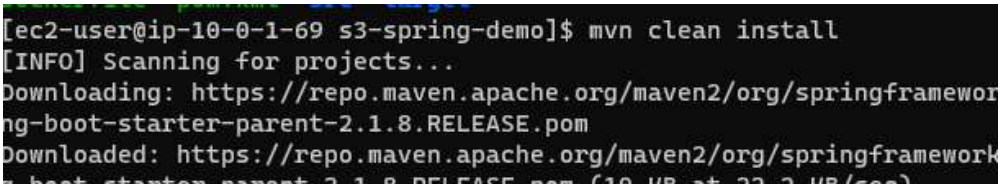
3. Use the following command to run the application.

```
mvn exec:java -Dexec.mainClass="org.example.App"
```

To view the new bucket and object that the program creates, perform the following steps.

1. In `Handler.java`, comment out the line `cleanUp(s3Client, bucket, key)` in the `sendRequest` method and save the file.
2. Rebuild the project by running `mvn clean package`.
3. Rerun `mvn exec:java -Dexec.mainClass="org.example.App"` to upload the text object once more.
4. Sign in to the S3 console to view the new object in the newly-created bucket.

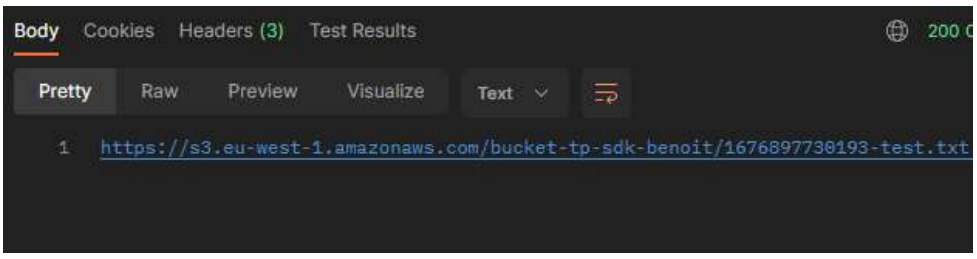
After you view the file, delete the object and then delete the bucket.



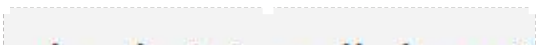
```
[ec2-user@ip-10-0-1-69 s3-spring-demo]$ mvn clean install
[INFO] Scanning for projects...
Downloading: https://repo.maven.apache.org/maven2/org/springframewor
ng-boot-starter-parent-2.1.8.RELEASE.pom
Downloaded: https://repo.maven.apache.org/maven2/org/springframework
```

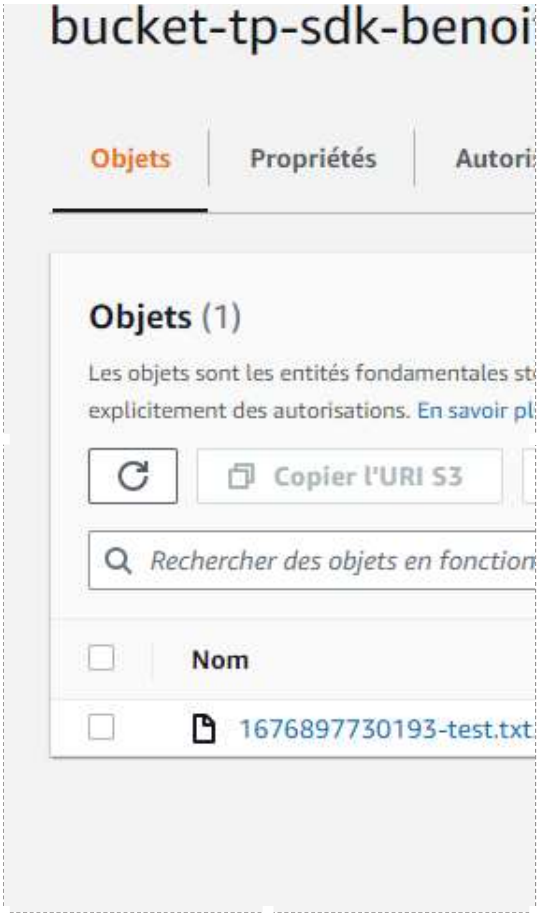
```
[INFO] <<< spring-boot-maven-plugin:2.1.8.RELEASE:run (default-cli) @ SpringBoot-S3Poc <<<
[INFO]
[INFO] --- spring-boot-maven-plugin:2.1.8.RELEASE:run (default-cli) @ SpringBoot-S3Poc ---

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.1.8.RELEASE)

2023-02-20 11:44:13.018  INFO 10753 --- [           main] com.org.tech.s3poc.AppStart
n ip-10-0-1-69.eu-west-1.compute.internal with PID 10753 (/home/ec2-user/s3-spring-demo/tar
er in /home/ec2-user/s3-spring-demo)
2023-02-20 11:44:13.027  INFO 10753 --- [           main] com.org.tech.s3poc.AppStart
```

Si le programme s'est exécuté sans erreur : on va tenter d'envoyer un fichier dans notre bucket avec postman :



Une fois l'objet envoyé : on vérifie sur notre bucket s'il a été réceptionné




bucket-tp-sdk-benoi
Objets
Propriétés
Autori
Objets (1)
Les objets sont les entités fondamentales st
explicitement des autorisations. En savoir pl
Copier l'URI S3
Rechercher des objets en fonction
Nom
1676897730193-test.txt