**Java example** demonstrating how to integrate **AWS SNS (Simple Notification Service) with**

**SQS (Simple Queue Service)**. This code will:

1. **Create an SNS Topic**
2. **Create an SQS Queue**
3. **Subscribe the SQS Queue to the SNS Topic**
4. **Publish a Message to the SNS Topic**
5. **Receive the Message from SQS**

**1. Add AWS SDK Dependency**

If you are using **Maven**, add this to pom.xml:

<dependencies>

<!-- AWS SDK for SNS -->

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>sns</artifactId>

<version>2.20.0</version>

</dependency>

<!-- AWS SDK for SQS -->

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>sqs</artifactId>

<version>2.20.0</version>

</dependency>

<!-- SLF4J (for logging) -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>2.0.7</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-simple</artifactId>

<version>2.0.7</version>

</dependency>

</dependencies>

**2. Java Code for AWS SNS + SQS Integration**

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.sns.SnsClient;

import software.amazon.awssdk.services.sns.model.\*;

import software.amazon.awssdk.services.sqs.SqsClient;

import software.amazon.awssdk.services.sqs.model.\*;

import java.util.HashMap;

import java.util.Map;

public class SnsSqsIntegration {

public static void main(String[] args) {

// AWS Region

Region region = Region.US\_EAST\_1; // Change as needed

// Create SNS and SQS Clients

try (SnsClient snsClient = SnsClient.builder()

.region(region)

.credentialsProvider(ProfileCredentialsProvider.create())

.build();

SqsClient sqsClient = SqsClient.builder()

.region(region)

.credentialsProvider(ProfileCredentialsProvider.create())

.build()) {

// 1. Create SNS Topic

String topicArn = createSnsTopic(snsClient, "MySNSTopic");

// 2. Create SQS Queue

String queueUrl = createSqsQueue(sqsClient, "MySQSQueue");

// 3. Get SQS Queue ARN

String queueArn = getQueueArn(sqsClient, queueUrl);

// 4. Subscribe SQS to SNS Topic

subscribeSqsToSns(snsClient, topicArn, queueArn);

// 5. Publish Message to SNS Topic

publishToSns(snsClient, topicArn, "Hello from AWS SNS to SQS!");

// 6. Receive Message from SQS

receiveMessagesFromSqs(sqsClient, queueUrl);

} catch (SnsException | SqsException e) {

System.err.println("Error: " + e.awsErrorDetails().errorMessage());

}

}

// Method to Create SNS Topic

public static String createSnsTopic(SnsClient snsClient, String topicName) {

CreateTopicRequest request = CreateTopicRequest.builder().name(topicName).build();

CreateTopicResponse response = snsClient.createTopic(request);

System.out.println("SNS Topic Created: " + response.topicArn());

return response.topicArn();

}

// Method to Create SQS Queue

public static String createSqsQueue(SqsClient sqsClient, String queueName) {

CreateQueueRequest request = CreateQueueRequest.builder().queueName(queueName).build();

CreateQueueResponse response = sqsClient.createQueue(request);

System.out.println("SQS Queue Created: " + response.queueUrl());

return response.queueUrl();

}

// Method to Get SQS Queue ARN

public static String getQueueArn(SqsClient sqsClient, String queueUrl) {

GetQueueAttributesRequest request = GetQueueAttributesRequest.builder()

.queueUrl(queueUrl)

.attributeNames(QueueAttributeName.QUEUE\_ARN)

.build();

GetQueueAttributesResponse response = sqsClient.getQueueAttributes(request);

return response.attributes().get(QueueAttributeName.QUEUE\_ARN);

}

// Method to Subscribe SQS to SNS Topic

public static void subscribeSqsToSns(SnsClient snsClient, String topicArn, String queueArn) {

SubscribeRequest request = SubscribeRequest.builder()

.topicArn(topicArn)

.protocol("sqs")

.endpoint(queueArn)

.build();

SubscribeResponse response = snsClient.subscribe(request);

System.out.println("SQS Subscribed to SNS. Subscription ARN: " + response.subscriptionArn());

}

// Method to Publish Message to SNS Topic

public static void publishToSns(SnsClient snsClient, String topicArn, String message) {

PublishRequest request = PublishRequest.builder()

.topicArn(topicArn)

.message(message)

.build();

PublishResponse response = snsClient.publish(request);

System.out.println("Message Published. Message ID: " + response.messageId());

}

// Method to Receive Messages from SQS

public static void receiveMessagesFromSqs(SqsClient sqsClient, String queueUrl) {

ReceiveMessageRequest request = ReceiveMessageRequest.builder()

.queueUrl(queueUrl)

.maxNumberOfMessages(5)

.waitTimeSeconds(10)

.build();

ReceiveMessageResponse response = sqsClient.receiveMessage(request);

if (response.messages().isEmpty()) {

System.out.println("No messages received.");

} else {

response.messages().forEach(msg -> {

System.out.println("Received Message: " + msg.body());

// Delete message after processing

deleteMessageFromQueue(sqsClient, queueUrl, msg.receiptHandle());

});

}

}

// Method to Delete a Message from SQS

public static void deleteMessageFromQueue(SqsClient sqsClient, String queueUrl, String receiptHandle) {

DeleteMessageRequest request = DeleteMessageRequest.builder()

.queueUrl(queueUrl)

.receiptHandle(receiptHandle)

.build();

sqsClient.deleteMessage(request);

System.out.println("Message deleted from SQS.");

}

}

**How This Code Works**

1. **Creates an SNS Topic** → "MySNSTopic".
2. **Creates an SQS Queue** → "MySQSQueue".
3. **Gets the ARN of the SQS Queue**.
4. **Subscribes the SQS Queue to the SNS Topic**.
5. **Publishes a Message to SNS**.
6. **Receives the Message from SQS**.
7. **Deletes the Message from SQS after reading it**.

**Run the Code**

* Make sure your **AWS credentials** are configured properly (~/.aws/credentials).
* **Update the AWS region if needed**.
* **Run the Java program**, and it will create an SNS-SQS integration.

**Expected Output**

SNS Topic Created: arn:aws:sns:us-east-1:123456789012:MySNSTopic

SQS Queue Created: https://sqs.us-east-1.amazonaws.com/123456789012/MySQSQueue

SQS Subscribed to SNS. Subscription ARN: arn:aws:sns:us-east-1:123456789012:MySNSTopic:abcd1234

Message Published. Message ID: 12345678-abcd-efgh-ijkl-123456789012

Received Message: Hello from AWS SNS to SQS!

Message deleted from SQS.

**Conclusion**

* This **automates message delivery** from SNS to SQS.
* **SQS acts as a buffer**, ensuring messages are **not lost**.
* You can use this for **event-driven architectures**.

🚀 **Now you have a working AWS SNS + SQS Java integration!** 🚀

{

"Version": "2012-10-17",

"Id": "SQSPolicyForSNS",

"Statement": [

{

"Effect": "Allow",

"Principal": "\*",

"Action": "sqs:SendMessage",

"Resource": "arn:aws:sqs:us-east-1:093547934696:MySQSQueue",

"Condition": {

"ArnEquals": {

"aws:SourceArn": "arn:aws:sns:us-east-1:093547934696:MySNSTopic"

}

}

}

]

}