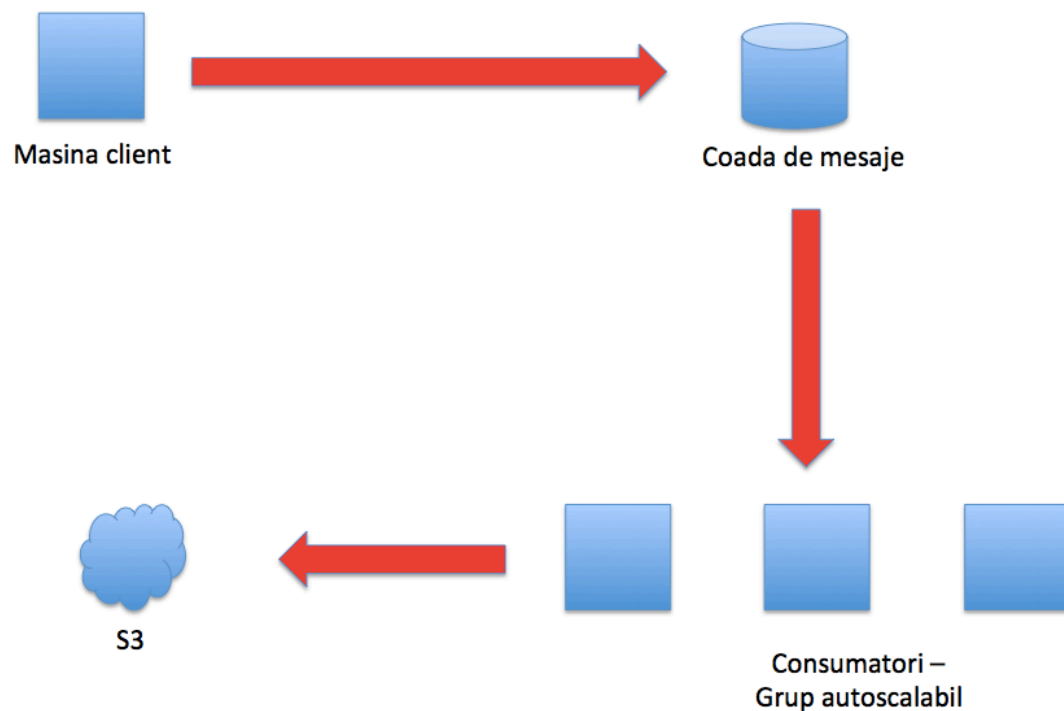


Introducere

Scopul este de a vedea cum se lucreaza cu o coada SQS, sa utilizati tool-ul din linia de comanda pentru a crea un grup de auto scalare si familiarizarea cu API-ul java oferit de amazon.

Pe parcursul laboratorului vom crea urmatoarea infrastructura:

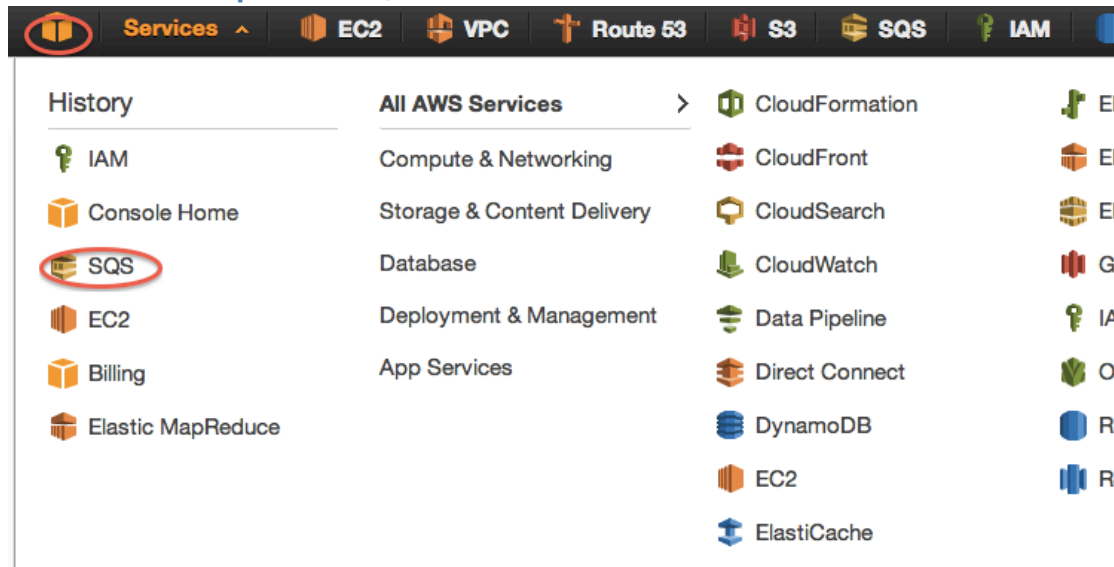


Partea1: Coada SQS

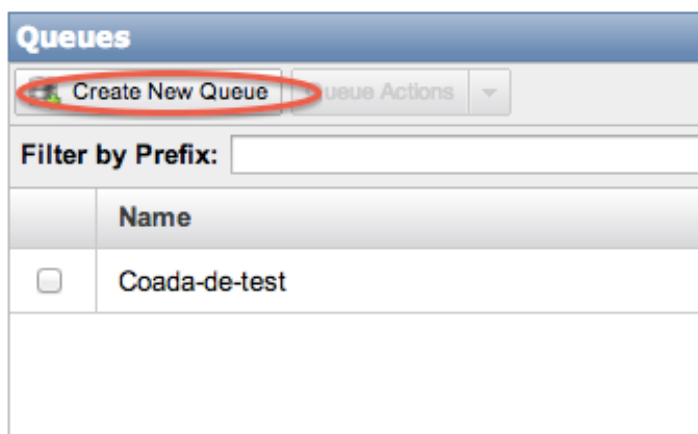
Vom crea o coada folosind consola web:

<https://fmi-unibuc-bigdata.signin.aws.amazon.com/console>

Pas1: Accesati aplicatia SQS



Pas2: Crearea unei cozi



Veti completa formularul urimator (numele cozii):

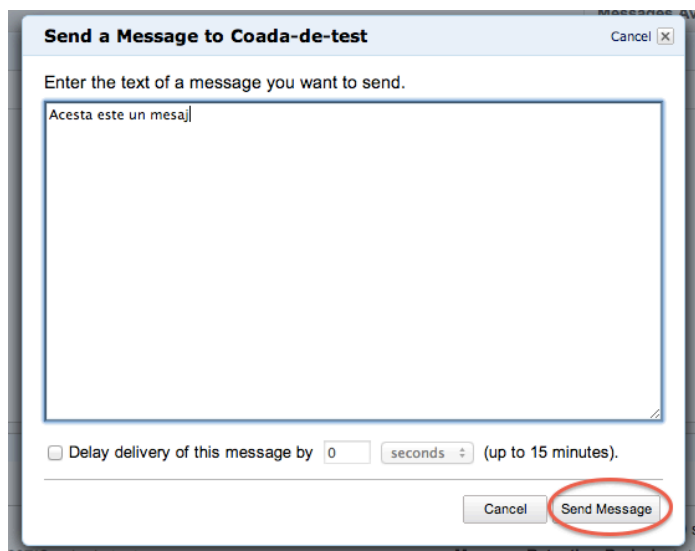
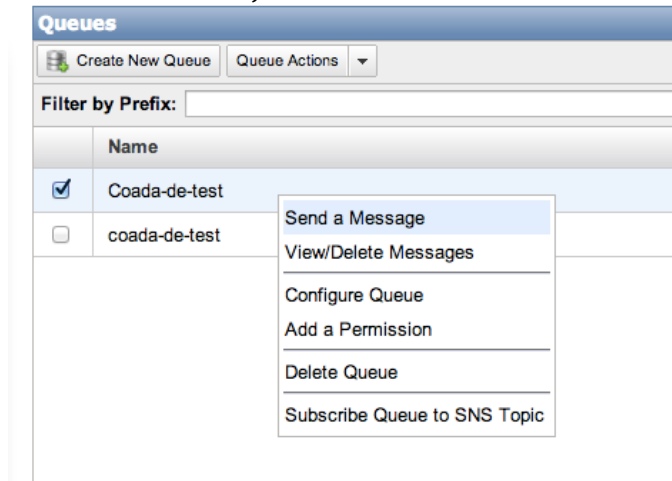
The screenshot shows the 'Create New Queue' form in the AWS SQS console. The form includes a 'Queue Name' field with the value 'coada-de-test' entered. Below the name field, there are several optional attributes to configure the queue:

- Default Visibility Timeout:** 30 seconds (Value must be between 0 seconds and 12 hours).
- Message Retention Period:** 4 days (Value must be between 1 minute and 14 days).
- Maximum Message Size:** 256 KB (Value must be between 1 and 256 KB).
- Delivery Delay:** 0 seconds (Value must be between 0 seconds and 15 minutes).
- Receive Message Wait Time:** 0 seconds (Value must be between 0 and 20 seconds).

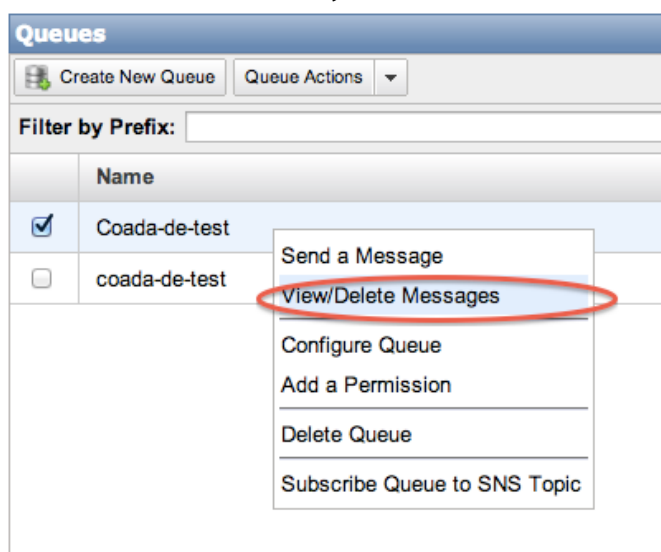
The 'Create Queue' button is highlighted with a red circle.

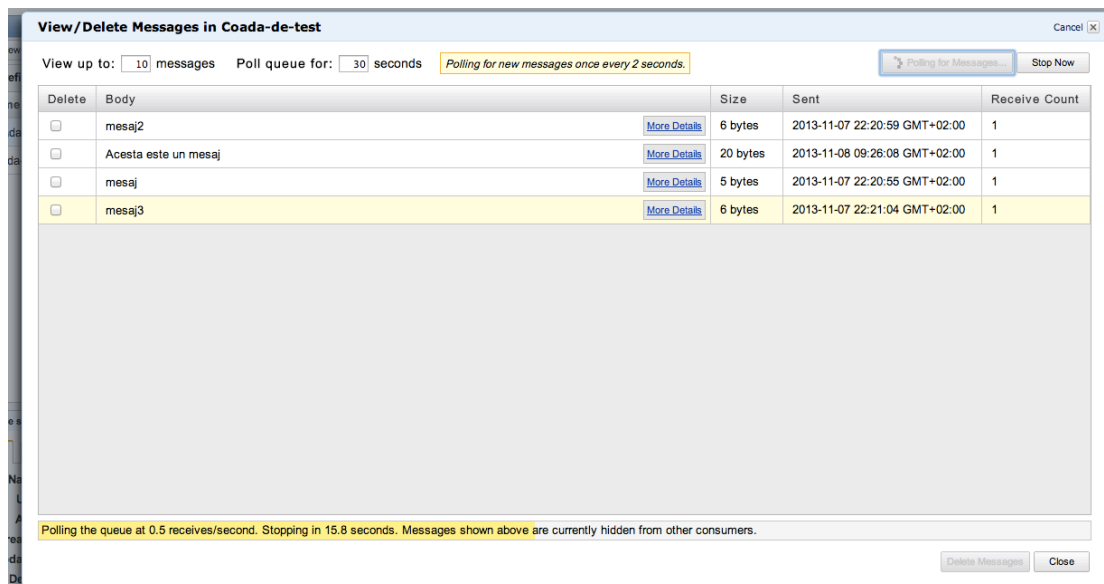
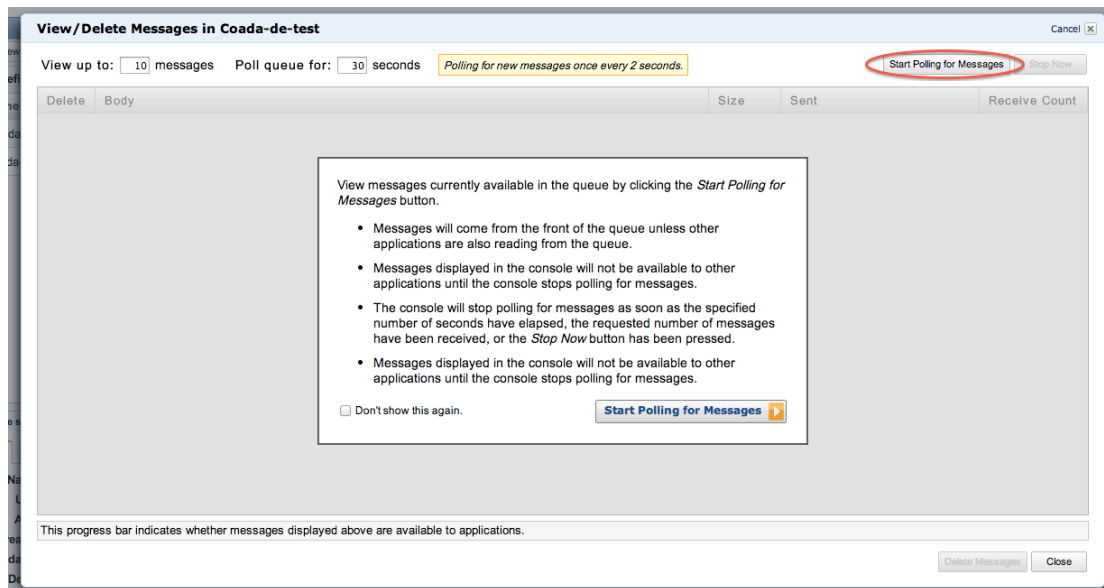
Pas3: Testarea cozii

Trimiteti un mesaj in coada folosind consolva web.



Verificati existenta mesajelor din coada:





Partea 2: Folosirea librăriei java pentru a trimite mesaje

Vom folosi librărie java si vom crea 2 aplicatii:

1. O aplicatie producator: care pune mesajele scrise la consola in coada
2. O aplicatie consumator: un loop care citeste si afiseaza mesaje.

Cele 2 aplicatii constituie un sistem rudimentar de chat.

Pas1: Descarcarea librăriei

<http://aws.amazon.com/sdkforjava/>

AWS SDK for Java

Get started quickly using AWS with the AWS SDK for Java. The SDK helps take the complexity out of coding by providing Java APIs for many AWS services including Amazon S3, Amazon EC2, DynamoDB, and more. The single, downloadable package includes the AWS Java library, code samples, and documentation.


[Getting Started »](#)

[API Documentation »](#)

[Community Forum »](#)

[Developer Blog »](#)

Downloads

[AWS SDK for Java »](#)
[Get the source on GitHub »](#)
[AWS Toolkit for Eclipse »](#)
[Get the source on GitHub »](#)

Follow us on Twitter

[Follow @awsforjava](#)

Dupa dezarhivare libraria se afla in folderul bin/aws-java-sdk-1.6.4.jar

Pas 2: Aplicatia producator

Creati un nou proiect eclipse si includeti libraria descarcata. Scrieti un program care ruleaza un loop infinit in care citeste o linie de la tastatura si o adauga ca un mesaj in coada SQS.

Exemplu de cod (incomplet):

Autentificare

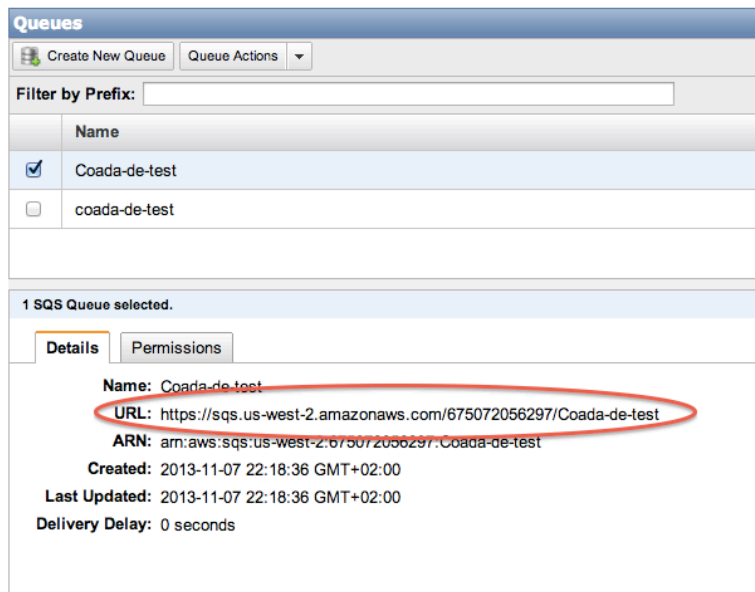
```
String accessKey = "abcdefgh...";
String secretKey = "secret-hbdf6tsf6";
AWSCredentials cred = new BasicAWSCredentials(
    accessKey, secretKey
);
```

Veti primi o lista cu cheile pentru fiecare utilizator.

Trimiterea unui mesaj

```
AmazonSQSClient sqs = new AmazonSQSClient(cred);
SendMessageRequest request = new SendMessageRequest("adresa-coada",
    "mesaj1");
SendMessageResult result = sqs.sendMessage(request);
```

Adresa cozii o gasiti in consola web:



Pas 3: Aplicatia consumator

Veti avea nevoie din nou de codul de autentificare iar citirea mesajelor din coada se face intr-un mod asemanator cu scrierea.

Gasiti documentatie si exemple la:

<http://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/com/amazonaws/services/sqs/AmazonSQSClient.html>

Pas 4: Impachetarea aplicatiilor

Exportati aplicatiile ca "runnable jar". Rulati-le in paralel (in 2 console diferite). Verificati functionarea sistemului de chat.

Partea 3: Folosirea bibliotecii pentru a accesa S3

Modificati aplicatia consumator sa scrie mesajele intr-un fisier din s3, si sa astepte 10 secunde dupa ce a scris fiecare mesaj.

Pentru a scrie in S3 folositi clasa AmazonS3Client:

```
AmazonS3Client s3 = new AmazonS3Client(cred);
```

Folositi aceeasi instanta de credentiale pe care o folosete si AmazonSQSClient;

Scrierea de mesaje in S3 se face asemanator cu trimiterea de mesaje in coada.
`s3.putObject(new PutObjectRequest("bucket-name", "cheie", continut)).`

Gasiti documentatie si exemple la:

<http://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/com/amazonaws/services/s3/AmazonS3Client.html>

Partea 4: Rularea aplicatiilor in cloud

Creati o masina EC2. Copiati si rulati pe ea pachetul cu aplicatia consumator creata la pasul anterior. Rulati producatorul local si scrieti mesaje. Verificati ca mesajele sunt salvate in s3. Deschideti inca o masina de consumator. Verificati ca ambele masini consuma mesaje din coada.

Partea 5: Autoscaling group pentru consumatori

Daca ai ajuns la partea 5, cheama-ma pentru detalii si explicatii.