1. **Setting your credentials for use by the AWS SDK for Java:**

For Linux, macOS, or Unix:

Set up ~/.aws/credentials this file.

This file will be automatically generated after you installed the aws sdk in your eclipse.

This file should contain lines in the following format:

```
[default]
aws_access_key_id = your_access_key_id
aws_secret_access_key = your_secret_access_key
```

Substitute your own AWS credentials values for the values

your_access_key_id and*your_secret_access_key*.

For Windows:

C:\Users*USERNAME* \.aws\credentials

This file will be automatically generated after you installed the aws sdk in your eclipse.

Set the AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY

environment variables.

To set these variables on Linux, macOS, or Unix, use **export** :

```
export AWS_ACCESS_KEY_ID=your_access_key_id
export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

To set these variables on Windows, use **set** :

```
set AWS_ACCESS_KEY_ID=your_access_key_id
set AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

2. **Setting the AWS Region for use by the AWS SDK for Java:**

Set the AWS Region in the AWS config file on your local system, located at:

- ~/.aws/config on Linux, macOS, or Unix
- C:\Users\USERNAME\.aws\config on Windows

If you don’t have this file you can create one with the following contents:

This file should contain lines in the following format:

```
[default]
region = your_aws_region
```

Substitute your desired AWS Region (for example, "us-west-2") for *your_aws_region*.

Set the AWS_REGION environment variable.

On Linux, macOS, or Unix, use **export** :

```
export AWS_REGION=your_aws_region
```

On Windows, use **set** :

```
set AWS_REGION=your_aws_region
```

3. **Install AWS cli**

Linux:

1.

$ curl -O https://bootstrap.pypa.io/get-pip.py

The script downloads and installs the latest version of pip and another required package named setuptools.

$ python get-pip.py --user

Run the script with Python

$ pip install awscli --upgrade --user

Use pip to install the AWS CLI.

$ aws --version

Verify that the AWS CLI installed correctly.

2.

Instructions for Ubuntu 17.10. All instructions should be same for other versions too.

1) **Install AWS CLI.**

   $ sudo apt-get install awscli

   Or $ sudo apt install awscli

   Or $ sudo aptitude install awscli

2) **Configure AWS.**

   aws configure

   i) Enter your Access key ID.

   ii) Enter your Secret access key.

   iii) Set default location as us-west-2 not US_WEST_2.

iv) Set Output format as json.

3) Create a Security Group.

aws ec2 create-security-group --group-name 546cc-sg --description "security group for development environment in EC2"

4) Set Protocols.

$ aws ec2 authorize-security-group-ingress --group-name 546cc-sg --protocol tcp --port 22 --cidr 0.0.0.0/0

5) Create a .pem file.

$ aws ec2 create-key-pair --key-name 546cc-key --query 'KeyMaterial' --output text > 546-key.pem

6) Change the access permissions of pem file to 400.

$ chmod 400 546-key.pem

7) Create an Instance.

$ aws ec2 run-instances --image-id ami-6e1a0117 --security-group-ids sg-04c44a3cab20bd362 --count 1 --instance-type t2.micro --key-name 546cc-key --query 'Instances[0].InstanceId'

8) Get a public IP address.

$ aws ec2 describe-instances --instance-ids "i-0588b13f89a128ac1" --query 'Reservations[0].Instances[0].PublicIpAddress'

9) **Connect with ssh.**

$ ssh -i 546-key.pem ubuntu@54.187.76.217

54.187.76.217 returned in step 8. Depending on the instance the prefix could be ubuntu, root and ec2.

Windows:

1.Download the appropriate MSI installer.:

64bit: https://s3.amazonaws.com/aws-cli/AWSCLI64.msi

32bit: https://s3.amazonaws.com/aws-cli/AWSCLI32.msi

2. Run the downloaded MSI installer.

3. Follow the instructions that appear.

The CLI installs to C:\Program Files\Amazon\AWSCLI (64-bit) or C:\Program Files (x86)\Amazon\AWSCLI (32-bit) by default. To confirm the installation, use the aws --version command at a command prompt (cmd)

MacOS:

Same as Linux

4. **AWS cli EC2**

Configure AWS: (If you have done 1 and 2 you can skip this step)
$ aws configure:
Type your AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY.
Region set as us-west-2
Format set as json

Create security group:
$aws ec2 create-security-group --group-name ${your security group name} --description "security group for development environment in EC2"
It will return the ${security group id}

Set security group inbound rules:
$ aws ec2 authorize-security-group-ingress --group-name ${your security group name} --protocol tcp --port 22 --cidr *0.0.0.0/0*

Create key pair:
$aws ec2 create-key-pair --key-name ${your key name} --query 'KeyMaterial' --output text > ${your key name}.pem

Set permission for the key.pem file:
$chmod 400 ${your key name}.pem

Run instance:

$aws ec2 run-instances --image-id *${your image id}*
--security-group-ids *${your security group id}* --count 1 --instance-type *t2.micro* --key-name ${your key name} --query 'Instances[0].InstanceId'
It will return the ${instance id}

Get public ip address of the instance:
aws ec2 describe-instances --instance-ids *"${your instance id}"* --query 'Reservations[0].Instances[0].PublicIpAddress'
It will return the ${public ip address}

Connect to the instance:
ssh -i ${your key name}.pem ubuntu(this name may be root/ec2_user/ubuntu, check the name in your image id)@*${your public ip address}*

***5.* Java code for EC2:**

```
import java.util.List;

import com.amazonaws.services.ec2.AmazonEC2;
import com.amazonaws.services.ec2.AmazonEC2ClientBuilder;
import com.amazonaws.services.ec2.model.Instance;
import com.amazonaws.services.ec2.model.RunInstancesRequest;
import com.amazonaws.services.ec2.model.RunInstancesResult;
import com.amazonaws.services.ec2.model.StartInstancesRequest;
import com.amazonaws.services.ec2.model.StopInstancesRequest;
```

```
import
com.amazonaws.services.ec2.model.TerminateInstancesRequest;

public class AwsExample {

	public static void createinstance() {

		final AmazonEC2 ec2 =
AmazonEC2ClientBuilder.defaultClient();

		System.out.println("create an instance");

		String imageId = "ami-8f78c2f7";  //image id of the instance
		int minInstanceCount = 1; //create 1 instance
		int maxInstanceCount = 1;

		RunInstancesRequest rir = new
RunInstancesRequest(imageId,
				minInstanceCount, maxInstanceCount);
		rir.setInstanceType("t2.micro"); //set instance type

		RunInstancesResult result = ec2.runInstances(rir);

		List<Instance> resultInstance =
				result.getReservation().getInstances();
```

```
        for(Instance ins : resultInstance) {
            System.out.println("New instance has been created:" +
                    ins.getInstanceId());//print the instance ID
        }
    }

    public static void startinstance(String instanceId) {
        final AmazonEC2 ec2 =
AmazonEC2ClientBuilder.defaultClient();
        StartInstancesRequest request = new StartInstancesRequest().
                withInstanceIds(instanceId);//start instance using the
instance id
        ec2.startInstances(request);

    }

    public static void stopinstance(String instanceId) {
        final AmazonEC2 ec2 =
AmazonEC2ClientBuilder.defaultClient();
        StopInstancesRequest request = new StopInstancesRequest().
                withInstanceIds(instanceId);//stop instance using the
instance id
        ec2.stopInstances(request);

    }
```

```
    public static void terminateinstance(String instanceId) {
        final AmazonEC2 ec2 =
AmazonEC2ClientBuilder.defaultClient();
        TerminateInstancesRequest request = new
TerminateInstancesRequest().
                withInstanceIds(instanceId);//terminate instance using
the instance id
        ec2.terminateInstances(request);

    }

    public static void main(String[] args) {
        AwsExample aws = new AwsExample();
        aws.createinstance();
    }


}
```

6. Java code for S3:

```
import java.nio.file.Paths;
import java.util.List;

import com.amazonaws.AmazonClientException;
import com.amazonaws.AmazonServiceException;
import com.amazonaws.auth.AWSCredentials;
```

```
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.ec2.AmazonEC2;
import com.amazonaws.services.ec2.AmazonEC2ClientBuilder;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import com.amazonaws.services.s3.model.Bucket;

/**
 * Create an Amazon S3 bucket.
 *
 * This code expects that you have AWS credentials set up per:
 *
http://docs.aws.amazon.com/java-sdk/latest/developer-guide/setup-cre
dentials.html
 */
public class createBucket
{
   public static Bucket getBucket(String bucket_name) {
            AWSCredentials credentials = null;
            try {
               credentials = new
ProfileCredentialsProvider().getCredentials();
            } catch (Exception e) {
               throw new AmazonClientException(
```

```
                    "Cannot load the credentials from the credential
profiles file. " +
                    "Please make sure that your credentials file is at the
correct " +
                    "location (~/.aws/credentials), and is in valid format.",e);
            }


            AmazonS3 s3 = AmazonS3ClientBuilder.standard()
                        .withCredentials(new
AWSStaticCredentialsProvider(credentials)).build();
        Bucket named_bucket = null;
       List<Bucket> buckets = s3.listBuckets();
       for (Bucket b : buckets) {
          if (b.getName().equals(bucket_name)) {
             named_bucket = b;
          }
       }
       return named_bucket;
    }

    public static Bucket createBucket(String bucket_name) {
            AWSCredentials credentials = null;
            try {
               credentials = new
ProfileCredentialsProvider().getCredentials();
```

```
        } catch (Exception e) {
            throw new AmazonClientException(
                    "Cannot load the credentials from the credential
profiles file. " +
                    "Please make sure that your credentials file is at the
correct " +
                    "location (~/.aws/credentials), and is in valid format.",e);
        }


        AmazonS3 s3 = AmazonS3ClientBuilder.standard()
                    .withCredentials(new
AWSStaticCredentialsProvider(credentials)).build();
       Bucket b = null;
      if (s3.doesBucketExist(bucket_name)) {
        System.out.format("Bucket %s already exists.\n",
bucket_name);
        b = getBucket(bucket_name);
      } else {
        try {
            b = s3.createBucket(bucket_name);
        } catch (AmazonS3Exception e) {
            System.err.println(e.getErrorMessage());
        }
      }
      return b;
```

```
    }

    public static void putObject(String bucket_name, String file_path) {
            final AmazonS3 s3 = AmazonS3ClientBuilder.defaultClient();

            String key_name =
Paths.get(file_path).getFileName().toString();
        try {
            s3.putObject(bucket_name, key_name, file_path);
        } catch (AmazonServiceException e) {
            System.err.println(e.getErrorMessage());
            System.exit(1);
        }
    }

    public static void deleteObject(String bucket_name, String
object_key) {

            final AmazonS3 s3 = AmazonS3ClientBuilder.defaultClient();
            try {
                    s3.deleteObject(bucket_name, object_key);
             } catch (AmazonServiceException e) {
                System.err.println(e.getErrorMessage());
                System.exit(1);
            }
    }
```

```
   public static void main(String[] args)
   {

      /* String bucket_name = "todoabuckte";

       System.out.format("\nCreating S3 bucket: %s\n", bucket_name);
       Bucket b = createBucket(bucket_name);
       if (b == null) {
          System.out.println("Error creating bucket!\n");
       } else {
          System.out.println("Done!\n");
       }*/
            //putObject("111test-bucket100",
"/Users/njjry/Desktop/devenv-key.pem");
            deleteObject("111test-bucket100", "AwsExample.java");
   }
}
```

7. Java code for SQS:

```
public class sqsExample
{
      private static final String QUEUE_NAME = "bbQueue";
      public static void main(String[] args)
```

```
    {
        sqsExample se = new sqsExample();
        se.receiveMsg();

    }

    public static void createQueue() {
        final AmazonSQS sqs = AmazonSQSClientBuilder.defaultClient();
        try {
            CreateQueueResult create_result =
sqs.createQueue(QUEUE_NAME);
            System.out.println("created a queue");
        } catch (AmazonSQSException e) {
            if (!e.getErrorCode().equals("QueueAlreadyExists")) {
                throw e;
            }
        }
    }

    public static void sendMsg() {
        final AmazonSQS sqs = AmazonSQSClientBuilder.defaultClient();
        String queueUrl =
sqs.getQueueUrl(QUEUE_NAME).getQueueUrl();
        SendMessageRequest send_msg_request = new
SendMessageRequest()
            .withQueueUrl(queueUrl)
```

```
                .withMessageBody("hello world")
                .withDelaySeconds(5);
        sqs.sendMessage(send_msg_request);
        System.out.println("send a msg");
    }

    public static void receiveMsg() {
        final AmazonSQS sqs = AmazonSQSClientBuilder.defaultClient();
        String queueUrl =
sqs.getQueueUrl(QUEUE_NAME).getQueueUrl();

        List<Message> messages =
sqs.receiveMessage(queueUrl).getMessages();
        // delete messages from the queue
        for (Message m : messages) {
            sqs.deleteMessage(queueUrl, m.getReceiptHandle());

        }
        System.out.println("deleted msgs");
    }
}
```