AWS code commit

Myfolder -> C:\Users\Administrator\Documents\Awscodecommit local repo

Try this. Open eclipse --> Go to File --> switch workspace --> other --> choose your workspace by clicking the Browse button. Hope this will work.

workspace : c:user/admin/workspace

C:aws (pem)

repository

awsrepo - repo name

localrepo desktop

git clone HTTPS REPO

<https://aws.amazon.com/blogs/devops/introducing-git-credentials-a-simple-way-to-connect-to-aws-codecommit-repositories-using-a-static-user-name-and-password/>

1.Aws configure

**2.git clone** [**https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo**](https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo) **my-demo-repo(localrepo)**

It will ask username and pwd

3.Git add

4.Git commit

From the command prompt or terminal, switch to your local repo directory and run the **git remote add** command to add the AWS CodeCommit repository as a remote repository for your local repo.

5.git remote add origin <https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo>

6.git push origin master

git remote add origin <https://git-codecommit.us-east-1.amazonaws.com/v1/repos/awsrepo>

**git remote –v**

which should create o/p

origin <https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo> (fetch)

origin <https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo> (push)

git config --global credential.helper "!aws codecommit credential-helper $@"

git config --global credential.UseHttpPath true

The Git credential helper writes the following to the .gitconfig file:

[credential]

helper = !aws codecommit credential-helper $@

UseHttpPath = true

Git clone

Maven

\*\*\*\*\*\*\*\*\*\*

MAVEN\_HOME C:\Program Files\apache-maven-3.5.2

C:\Users\Administrator>cd C:\Users\Administrator\Desktop\Struts2Hibernate3Spring

3Tile2Integration\target

[INFO] Installing C:\Users\Administrator\Desktop\Struts2Hibernate3Spring3Tile2In

tegration\target\target\Struts2Hibernate3Spring3Tile2Integration-0.0.1-SNAPSHOT.

war to C:\Users\Administrator\.m2\repository\sangeetha\Struts2Hibernate3Spring3T

ile2Integration\0.0.1-SNAPSHOT\Struts2Hibernate3Spring3Tile2Integration-0.0.1-SN

APSHOT.war

[INFO] Installing C:\Users\Administrator\Desktop\Struts2Hibernate3Spring3Tile2In

tegration\target\pom.xml to C:\Users\Administrator\.m2\repository\sangeetha\Stru

ts2Hibernate3Spring3Tile2Integration\0.0.1-SNAPSHOT\Struts2Hibernate3Spring3Tile

2Integration-0.0.1-SNAPSHOT.pom

Build commNDS



version: 0.2

environment\_variables:

plaintext:

JAVA\_HOME: "/usr/lib/jvm/java-8-openjdk-amd64"

PATH: "/usr/bin/mvn"

phases:

install:

commands:

- apt-get update -y

- apt-get install -y maven

pre\_build:

commands:

- echo nothing to do in the pre\_build phase

build:

commands:

- echo Build started on `date`

- mvn clean

- mvn install

post\_build:

commands:

- echo completed on `date`

artifacts:

files:

- target/messageUtil.jar

discard-paths: yes

"

Step 1 :Docker

\*\*\*\*\*\*\*\*\*\*

Pre-requisites

Install Docker - Installed

Install apache tomcat – yet to

Sudo apt-get update && apt-get install -y apache2

Create Dockerfile

* Mkdir mywebapp
* Cd mywebapp
* Nano Dockerfile

**From tomcat:8 – jre8**

**MAINTAINER** [**email**](mailto:sanveenv@gmail.com) **id**

**ADD mywebapp.war /usr/local/tomcat/webapp/**

**EXPOSE 8080**

**Deploy war file to docker image**

* Prepare a Dockerfile with the following content.
* Copy the war file from out from the target folder.
* From tomcat:8-jre8
* **MAINTAINER mailid**
* ADD dockerwar.war /usr/local/tomcat/webapps/

Now run the follow command to docker image name webserver with your Dockerfile  in current directory. First time to build docker image will require download and may take longer times.

* docker build -t webserver .
* docker run -it --rm -p 8080:8080 --name dockerwar webserver

Open a browser with URL <http://192.168.59.103:8080/dockerwar/>

AWS Code Deploy

I have created 3 ubuntu instances

Keypaid ecs

Cdinstance1

Cdinstance2

Cdinstance3

Sample deployment wizard

Green/Blue deployment

**We're setting up your environment now. It might take a few minutes.**

**Sample environment:**

We are launching a Classic load balancer (BlueGreenLoadBalancer) and an Auto Scaling group (BlueGreenAutoScalingGroup) of three t2.micro EC2 instances.

**Sample application:**

After your environment is launched, we will install a sample application using an AWS CodeDeploy in-place deployment.

You can download the application code:

<https://s3.amazonaws.com/aws-codedeploy-us-east-1/samples/latest/SampleApp_Linux.zip>



**Congratulations! Your environment is ready.**

**Sample environment:**

We launched a Classic load balancer (BlueGreenLoadBalancer-dnayo1q) and an Auto Scaling group (CodeDeployBGStack-dnayo1q-BlueGreenAutoScalingGroup-WN5SD56NS20Y) of three t2.micro EC2 instances.

**Sample application:**

We installed a sample application using an AWS CodeDeploy in-place deployment.

You can download the application code:

[https://s3.amazonaws.com/aws-codedeploy-us-east-1/samples/latest/SampleAp HYPERLINK "https://s3.amazonaws.com/aws-codedeploy-us-east-1/samples/latest/SampleApp\_Linux.zip"p HYPERLINK "https://s3.amazonaws.com/aws-codedeploy-us-east-1/samples/latest/SampleApp\_Linux.zip"\_Linux.zip](https://s3.amazonaws.com/aws-codedeploy-us-east-1/samples/latest/SampleApp_Linux.zip)

You can view the sample web page using your web browser:

[http://BlueGreenLoadBalancer-dnayo1q-1141187827.us-east-1.elb.amazonaws.com](http://bluegreenloadbalancer-dnayo1q-1141187827.us-east-1.elb.amazonaws.com/)

**Sample blue/green deployment:**

This is the new application revision that will be installed by the blue/green deployment.

You can download the application code:

<https://s3.amazonaws.com/aws-codedeploy-us-east-1/samples/latest/SampleApp2_Linux.zip>

After the application revision is installed, you can view it using your web browser:

[http://BlueGreenLoadBalancer-dnayo1q-1141187827.us-east-1.elb.amazonaws.com](http://bluegreenloadbalancer-dnayo1q-1141187827.us-east-1.elb.amazonaws.com/)

**Cleanup:**

After you finish examining the sample application, you can clean up your sample environment resources. In the Auto Scaling console, delete the Auto Scaling group prefixed with CodeDeploy\_BlueGreenDemoFleet-dnayo1q. In the AWS CloudFormation console, delete the stack named CodeDeployBGStack-dnayo1q.

To continue the sample deployment, choose **Start blue/green deployment.**



Pulling Docker images from ECR for your build environment does not use the service role. You will need to add permissions for CodeBuild to pull the image in the ECR repository policy. See this sample for instructions and the specific policy/permissions:

<https://docs.aws.amazon.com/codebuild/latest/userguide/sample-ecr.html>

buildspec.yml

<https://forums.aws.amazon.com/thread.jspa?threadID=244603>