

ASSESSMENT - 22

Jenkins 2

**TO
THE
NEW**™




1. Create a Jenkins pipeline Job to delete redundant docker images daily at 1 AM UTC.

- Create a pipeline job


Enter an item name

ques_1

» Required field

**Freestyle project**

This is the central feature of Jenkins. Jenkins will build y something other than software build.

**Pipeline**

Orchestrates long-running activities that can span multi organizing complex activities that do not easily fit in free

- Add cron

Build Triggers

☐ Build after other projects are built

☒ Build periodically

Schedule

0 1 * * *

- Fetch the Jenkinsfile from the git repository.

Definition Pipeline script from SCM

SCM Git

Repositories

Repository URL

Credentials

[Add](#)

[Advanced...](#)

[Add Repository](#)

Branches to build

Branch Specifier (blank for 'any')

[Add Branch](#)

Repository browser (Auto)

Additional Behaviours [Add](#)

[Save](#) [Apply](#) Path

- Jenkinsfile in repository.

26 lines (26 sloc) | 581 Bytes

```

1 pipeline {
2     agent any
3     stages {
4         stage('listing all docker images') {
5             steps {
6                 sh '''
7                 docker image ls
8                 '''
9             }
10        }
11        stage('Deleting Redundant images'){
12            steps{
13                sh '''
14                docker image prune -af
15                '''
16            }
17        }
18        stage('listing remaining docker images') {
19            steps {
20                sh '''
21                docker image ls
22                '''
23            }
24        }
25    }
26 }
```

- Build the job.
- Output of stage 1

```
[Pipeline] { (listing all docker images)
[Pipeline] sh
+ docker image ls
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
garimal998/nginx_image	nginx	5e9581f7e627	3 days ago	127MB
garimal998/tomcat_image	tomcat	881370d659b0	7 days ago	508MB
nginx	latest	ed21b7a8aee9	13 days ago	127MB
alpine	latest	a187dde48cd2	2 weeks ago	5.6MB
ubuntu	latest	4e5021d210f6	3 weeks ago	64.2MB
centos	latest	470671670cac	2 months ago	237MB

- Output of stage 2

```
[Pipeline] { (Deleting Redundant images)
[Pipeline] sh
+ docker image prune -af
Deleted Images:
untagged: garimal998/nginx_image:nginx
untagged: garimal998/nginx_image@sha256:f0e6b7a8ef990a541e8b600d9109a41cd3b7a47129c2dc6600c89e10803c2145
deleted: sha256:5e9581f7e627f8f8be4961c504cfa7ef0afe902c5f8e415a0eddee5e11f4e1f90
deleted: sha256:49686afd65d9875bda0b439a0636e1cc7f443015d383bdf67f24f7bad1975536
deleted: sha256:88ac93ec5c0a73d2b691f51e6837f36ba823c2488bcca01b86f14be34f98069a
deleted: sha256:47772eae38f5466aac9257acd6a219c4a1768d8c7893732612416708904d0d93
untagged: alpine:latest
untagged: alpine@sha256:b276d875eed9c7d3f1cfa7edb06b22ed22b14219a7d67c52c56612330348239
deleted: sha256:a187dde48cd289ac374ad8539930628314bc581a481cdb41409c9289419ddb72
deleted: sha256:beee9f30bc1f711043e78d4a2be0668955d4b761d587d6f60c2c8dc081efb203
untagged: centos:latest
untagged: centos@sha256:fe8d824220415eed5477b63addf40fb06c3b049404242b31982106ac204f6700
untagged: ubuntu:latest
untagged: ubuntu@sha256:bec5a2727be7fff3d308193cfde3491f8fba1a2ba392b7546b43a051853a341d
deleted: sha256:4e5021d210f65ebe915670c7089120120bc0a303b90208592851708c1b8c04bd
```

- Output of stage 3

```
Total reclaimed space: 5.596MB
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (listing remaining docker images)
[Pipeline] sh
+ docker image ls
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
garimal998/tomcat_image	tomcat	881370d659b0	7 days ago	508MB
nginx	latest	ed21b7a8aee9	13 days ago	127MB

2. Create a shared library function to convert error and success output into a colorful output and use it in the upcoming questions(Hint: use ANSI color).

- Install AnsiColor plugin

Updates	Available	Installed	Advanced	
Enabled		Name ↓	Version	Previously i
<input checked="" type="checkbox"/>	AnsiColor	Adds ANSI coloring to the Console Output	0.6.3	

- Configure the Global Pipeline Libraries in manage jenkins

Global Pipeline Libraries


Sharable libraries available to any Pipeline jobs running on this system. These libraries will be trusted, meaning they run without "sandbox"

Library	
Name	<input type="text" value="my-shared-library"/>
Default version	<input type="text" value="master"/> <small>Cannot validate default version until at</small>
Load implicitly	<input type="checkbox"/>
Allow default version to be overridden	<input checked="" type="checkbox"/>
Include @Library changes in job recent changes	<input checked="" type="checkbox"/>


- Create a pipeline job

Enter an item name

» Required field



Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system something other than software build.



Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (former organizing complex activities that do not easily fit in free-style job type.

Pipeline script from SCM

SCM

Git

Repositories

Repository URL

Credentials

 Add ▼

Branches to build

Branch Specifier (blank for 'any')

Repository browser

Additional Behaviours

 Add ▼

Apply

Path

Jenkinsfile

- Create a git repository with a folder named vars. The structure should be somewhat like.

The screenshot shows the GitHub repository page for 'ga-sudo / Jenkins2'. The repository has 20 commits, 1 branch, 0 packages, and 0 releases. The 'Code' tab is selected. Below the repository information, there are buttons for 'Branch: master', 'New pull request', 'Create new file', and 'Upload files'. The file list shows the following structure:

File/Folder	Action
vars	Update logs.groovy
Jenkinsfile	Update Jenkinsfile

- In the vars folder we shall write the groovy scripts that'll be pulled into the jenkins file at the runtime.


The screenshot shows the 'vars' folder in the 'Jenkins2' repository. The file list shows the following structure:

File/Folder	Action
logs.groovy	Update logs.groovy

logs.groovy

Branch: **master** ▾

Jenkins2 / [vars](#) / **logs.groovy**

 **ga-sudo** Update logs.groovy

1 contributor

23 lines (17 sloc) | 411 Bytes

```
1  def loadColors() {
2      RED='\033[0;31m'
3      GREEN='\033[0;32m'
4      NC='\033[0m'
5  }
6
7  def info(message){
8      loadColors()
9      sh ""set +x;echo -e "${GREEN}[INFO] - ${message} ${NC}" ""
10 }
11
12 def warn(message){
13     loadColors()
14     sh ""set +x;echo -e "${RED}[WARN] - ${message} ${NC}" ""
15 }
16
17 def gitCommitId(message){
18     loadColors()
19     sh ""set +x;echo -e "${GREEN}[GIT COMMIT ID] - ${message} ${NC}" ""
20 }
21
```


My jenkins file

Branch: **master** ▾

Jenkins2 / Jenkinsfile

 **ga-sudo** Update Jenkinsfile

1 contributor

21 lines (21 sloc) | 470 Bytes

```
1  #!groovy
2  library identifier: 'Jenkins2@master', retriever: modernSCM(
3      [$class: 'GitSCMSource' ,
4         remote: 'https://github.com/ga-sudo/Jenkins2'])
5  pipeline {
6      agent any
7      options{
8          timestamps()
9          ansiColor( 'xterm' )
10     }
11     stages {
12         stage( 'Coloured Outputs with git commit id' ) {
13             steps {
14                 script{
15                     logs.info "SUCCESS"
16                     logs.warn "WARNING"
17                 }
18             }
19         }
20     }
21 }
```


Output:

```
-----
[Pipeline] { (Coloured Outputs with git commit id)
[Pipeline] script
[Pipeline] {
[Pipeline] sh
09:45:38 + set +x
09:45:38 -e [INFO] - SUCCESS
[Pipeline] sh
09:45:38 + set +x
09:45:38 -e [WARN] - WARNING
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // ansiColor
[Pipeline] }
[Pipeline] // timestamps
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

3. Create a function in the same shared library to output git commitID.

- Jenkins file

Branch: master ▾ **Jenkins2 / Jenkinsfile**

 **ga-sudo** Update Jenkinsfile

1 contributor

23 lines (23 sloc) | 592 Bytes

```
1  #!groovy
2  library identifier: 'Jenkins2@master', retriever: modernSCM(
3      [$class: 'GitSCMSource' ,
4          remote: 'https://github.com/ga-sudo/Jenkins2'])
5  pipeline {
6      agent any
7      options{
8          timestamps()
9          ansiColor( 'xterm' )
10     }
11     stages {
12         stage( 'Coloured Outputs with git commit id' ) {
13             steps {
14                 script{
15                     logs.info "SUCCESS"
16                     logs.warn "WARNING"
17                     def gitId=sh(script: 'git rev-parse HEAD' , returnStdout: true)
18                     logs.gitCommitId(gitId)
19                 }
20             }
21         }
22     }
23 }
```

- Output:

```
[Pipeline] { (Coloured Outputs with git commit id)
[Pipeline] script
[Pipeline] {
[Pipeline] sh
09:49:00 + set +x
09:49:00 -e [INFO] - SUCCESS
[Pipeline] sh
09:49:00 + set +x
09:49:00 -e [WARN] - WARNING
[Pipeline] sh
09:49:01 + git rev-parse HEAD
[Pipeline] sh
09:49:01 + set +x
09:49:01 -e [GIT COMMIT ID] - cb8ac824f9deb319c756e50aa2c7f37f81ed299d
```

4. Take a sample react application and deploy it on EKS

- a. You can use this repo or any other sample (<https://github.com/gothinkster/react-redux-realworld-example-app>).
- b. Create a Dockerfile for react application
- c. Build and publish image to ECR (create ECR repo of your name) and image must have the git commit id in its name.
- d. Deploy this image on EKS.
- e. Send Slack notification/Mail/google chat notification for build pass, abort and fail.

- Switch to user jenkins, login to aws ecr registry and authenticate docker for ecr.

```
jenkins@garima:~$ aws ecr get-login-password --region us-east-1 | docker login -
-username AWS --password-stdin 044650439222.dkr.ecr.us-east-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/.docker/co
nfig.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
jenkins@garima:~$ █
```

- Dockerfile

```
jenkins@garima:~$ cat Dockerfile
FROM mhart/alpine-node:11 AS builder
WORKDIR /app
COPY . .
RUN yarn run build

FROM mhart/alpine-node
RUN yarn global add serve
WORKDIR /app
COPY --from=builder /app/build .
CMD ["serve", "-p", "80", "-s", "."]
```

- Create ecr repo

✓ Successfully created repository garima

ECR > Repositories

Repositories (1) 🔄 View push co

🔍 Find repositories

Repository name ▲	URI
<input type="radio"/> garima	📦 044650439222.dkr.ecr.us-east-1.amazonaws.com/garima

- Create a pipeline job.

Enter an item name

ques_4

» Required field



Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build tool, or something other than software build.



Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines or organizing complex activities that do not easily fit in free-style job type.

Pipeline script from SCM


SCM

Git

Repositories

Repository URL

Credentials

 Add ▼

Branches to build

Branch Specifier (blank for 'any')

Repository browser

Additional Behaviours

Add ▼

Apply

Path

Jenkinsfile

● Jenkinsfile

```
pipeline {
  agent any
  stages {
    stage('Docker Build') {
      steps {
        sh 'docker build -t 044650439222.dkr.ecr.us-east-1.amazonaws.com/garima:${git rev-parse HEAD} .'
      }
    }
    stage('pushing image to ECR') {
      steps {
        sh 'docker push 044650439222.dkr.ecr.us-east-1.amazonaws.com/garima:${git rev-parse HEAD}'
      }
    }
    stage('Deploying on eks') {
      steps {
        sh 'kubectl apply -f deployment.yml'
        sh 'sleep 10'
        sh 'kubectl get pods'
        sh 'kubectl get svc'
        sh 'kubectl describe pods'
      }
    }
  }
  post {
    success {
      emailx body: 'Success', recipientProviders: [[class: 'DevelopersRecipientProvider'], [class: 'RequesterRecipientProvider']]
    }
    failure {
      emailx body: 'failed', recipientProviders: [[class: 'DevelopersRecipientProvider'], [class: 'RequesterRecipientProvider']]
    }
  }
}
```

Branch: **ques_4** ▼

Jenkins2 / Dockerfile

 **ga-sudo** Create Dockerfile


1 contributor

8 lines (8 sloc) | 237 Bytes

```
1 FROM node:13.12.0-alpine
2 WORKDIR /app
3 ENV PATH /app/node_modules/.bin:$PATH
4 RUN apk add git
5 RUN git clone https://github.com/gothinkster/react-redux-realworld-example-app /app
6 RUN npm install --silent
7 CMD [ "npm" , "start" ]
8 EXPOSE 4100
```

Branch: ques_4 ▾

Jenkins2 / deployment.yaml

 ga-sudo Create deployment.yaml

1 contributor

34 lines (34 sloc) | 542 Bytes

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: react-deployment
5    labels:
6      app: react
7  spec:
8    replicas: 2
9    selector:
10     matchLabels:
11       app: react
12   template:
13     metadata:
14       labels:
15         app: react
16   spec:
17     containers:
18     - name: dreact
19       image: 044650439222.dkr.ecr.us-east-1.amazonaws.com/garima
20       ports:
21       - containerPort: 4100
22   ---
```



```
22 ---
23 apiVersion: v1
24 kind: Service
25 metadata:
26   name: my-service
27 spec:
28   type: NodePort
29   selector:
30     app: react
31   ports:
32     - protocol: TCP
33       port: 4100
34       targetPort: 4100
```
