

Jenkins 2

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

Custom Console Output

```
Started by user Jay\_patel
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/dockerdelete-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Delete docker images acc to ist)
[Pipeline] sh
+ sudo docker image prune -f
Total reclaimed space: 0B
[Pipeline] sh
+ echo Successfully delete the docker images
Successfully delete the docker images
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Pipeline dockerdelete-pipeline

Create a jenkins pipeline Job to delete redundant docker images daily at 1 AM UTC.(6:30 am ist)

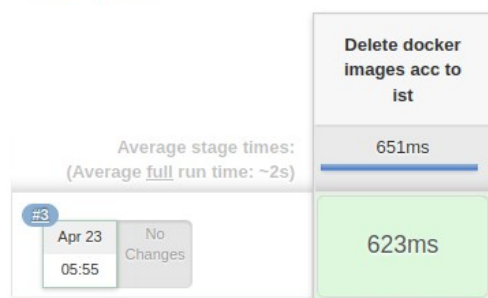
 [edit description](#)

Disable Project



[Recent Changes](#)

Stage View



General
Build Triggers
Advanced Project Options
Pipeline

Build Triggers

☐ Build after other projects are built
☒ Build periodically

Schedule
30 06 * * *

⚠ Spread load evenly by using 'H 06 * * *' rather than '30 06 * * *'
Would last have run at Wednesday, 22 April, 2020 6:30:39 AM IST; would next run at Thursday, 23 April, 2020 6:30:39 AM IST.

☐ GitHub hook trigger for GITScm polling
☐ Poll SCM
☐ Disable this project
☐ Quiet period
☐ Trigger builds remotely (e.g., from scripts)

Advanced Project Options

Save
Apply
Advanced...

Pipeline

Definition
Pipeline script

Script

```

1 pipeline {
2   agent any
3   triggers {
4     cron('30 06 * * *')
5   }
6   stages {
7     stage('Delete docker images acc to ist') {
8       steps {
9         sh "sudo docker image prune -f "
10        sh 'echo "Successfully delete the docker images"'
11      }
12    }
13  }
14 }
15

```

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

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

```

1  #!groovy
2  library identifier: 'jenkinsdemo@master', retriever: modernSCM(
3      [class: 'GitSCMSource',
4        remote: 'https://github.com/jaypttn/jenkinsdemo'])
5  pipeline {
6      agent any
7      options{
8          timestamps()
9          ansiColor('xterm')
10     }
11     stages {
12         stage('List All files and dir of current path') {
13             steps {
14                 script{
15                     sh ''' ls -l
16                     '''
17                     myscript.info("Successfully Executed")
18                     def ret = sh(script: 'lll', returnStatus: true)
19                     myscript.warning(ret)
20                 }
21             }
22         }
23         stage('Git Commit Id') {
24             steps {
25                 script{
26                     def gitId=sh(script:'git rev-parse HEAD', returnStdout: true)
27                     myscript.gitCommitId(gitId)
28                 }
29             }
30         }
31     }

```

Branch: master **jenkinsdemo** / vars / myscript.groovy


Find file Copy path

 jaypttn Jenkins file

ac9cc41 43 minutes ago

1 contributor

12 lines (10 sloc) | 271 Bytes

Raw Blame History  

```

1  def info(message) {
2      echo "\033[4;32mINFO:\033[0m \033[1;33m${message}\033[0m\n"
3  }
4
5  def warning(message) {
6      echo "WARNING: \033[1;31;43m${message}\033[0m"
7  }
8
9  def gitCommitId(message)
10 {
11     echo "\033[4;32m[Git Commit ID] \033[0m \033[1;33m ${message} \033[0m\n"
12 }

```

Pipeline

Definition Pipeline script from SCM

SCM Git

Repositories

Repository URL https://github.com/jayptn/jenkinsdemo.git

Credentials jayptn/***** (github cred) Add

Advanced...

Add Repository

Branches to build

Branch Specifier (blank for 'any') */master Add Branch

Repository browser (Auto)

Additional Behaviours Add

Save Apply

```
[Pipeline] ansiColor
[Pipeline] {
[Pipeline] stage
[Pipeline] { (List All files and dir of current path)
[Pipeline] script
[Pipeline] {
[Pipeline] sh
06:52:47 + ls -l
06:52:47 total 16
06:52:47 -rw-r--r-- 1 jenkins jenkins 84 Apr 23 06:52 hello.groovy
06:52:47 -rw-r--r-- 1 jenkins jenkins 847 Apr 23 06:52 Jenkinsfile
06:52:47 -rw-r--r-- 1 jenkins jenkins 37 Apr 23 06:52 README.md
06:52:47 drwxr-xr-x 2 jenkins jenkins 4096 Apr 23 06:52 vars
[Pipeline] echo
06:52:47 INFO: Successfully Executed
06:52:47
[Pipeline] sh
06:52:47 + lll
06:52:47 /var/lib/jenkins/workspace/jenkins-ansi-shared@tmp/durable-4fdd7d78/script.sh: 1: /var/lib/jenkins
/workspace/jenkins-ansi-shared@tmp/durable-4fdd7d78/script.sh: lll: not found
[Pipeline] echo
06:52:47 WARNING: 127
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
```

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

```
stage('Git Commit Id') {
    steps {
        script{
            def gitId=sh(script:'git rev-parse HEAD', returnStdout: true)
            myscript.gitCommitId(gitId)
        }
    }
}
```

```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Git Commit Id)
[Pipeline] script
[Pipeline] {
[Pipeline] sh
06:52:47 + git rev-parse HEAD
[Pipeline] echo
06:52:47 [Git Commit ID]_ 32fa0ad2b12ba9b4c94d96c8d801235e9e812125
06:52:47
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // ansiColor
[Pipeline] }
[Pipeline] // timestamps
[Pipeline] }
```

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

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

Resources was not available(EKS) so Prashant told us to leave this question.