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)

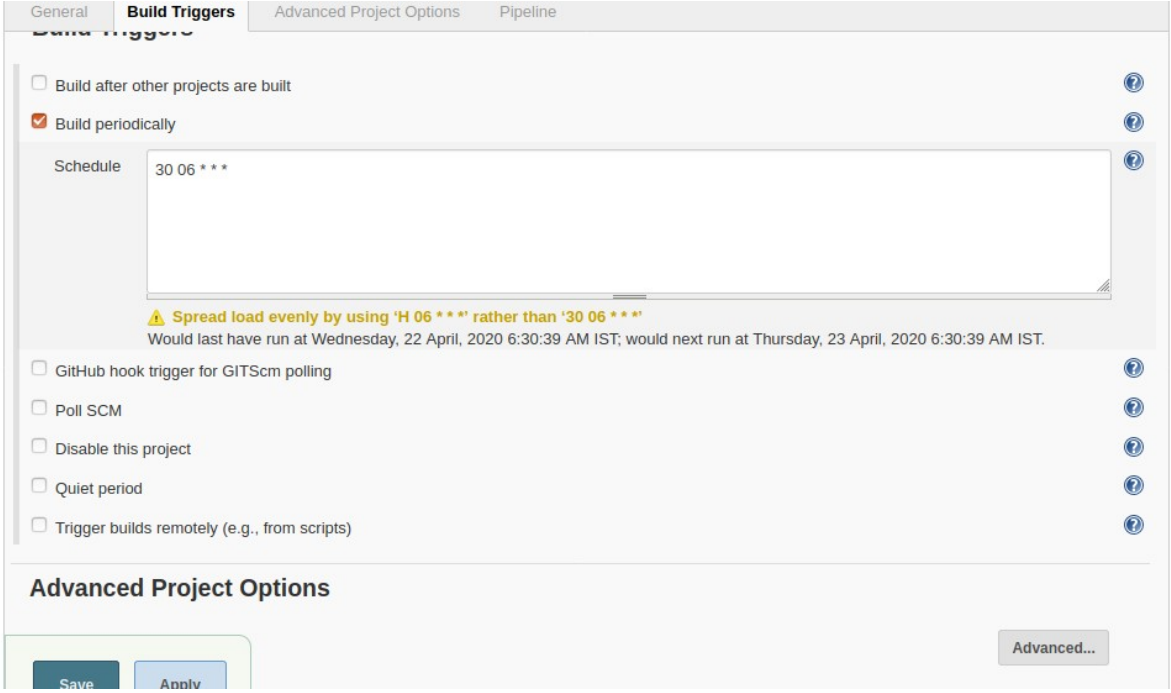
edit description

Disable Project

Recent Changes

Stage View

	Delete docker images acc to ist
Average stage times: (Average full run time: ~2s)	651ms
#3 Apr 23 05:55 No Changes	623ms



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).

```
#!groovy
library identifier: 'jenkinsdemo@master', retriever: modernSCM(
        [$class: 'GitSCMSource',
         remote: 'https://github.com/jaypttn/jenkinsdemo'])
pipeline {
   agent any
    options{
       timestamps()
       ansiColor('xterm')
   }
   stages {
      stage('List All files and dir of current path') {
         steps {
             script{
                    sh ''' ls -l
                    '''
                    myscript.info("Successfully Executed")
                    def ret = sh(script: 'lll', returnStatus: true)
                    myscript.warning(ret)
             }
         }
      }
      stage('Git Commit Id') {
         steps {
              script{
                    def gitId=sh(script:'git rev-parse HEAD', returnStdout: true)
                    myscript.gitCommitId(gitId)
             }
         }
      }
   }
```

Branch: master ▾ **jenkinsdemo** / vars / **myscript.groovy** Find file Copy path

jaypttn Jenkins file ac9cc41 43 minutes ago

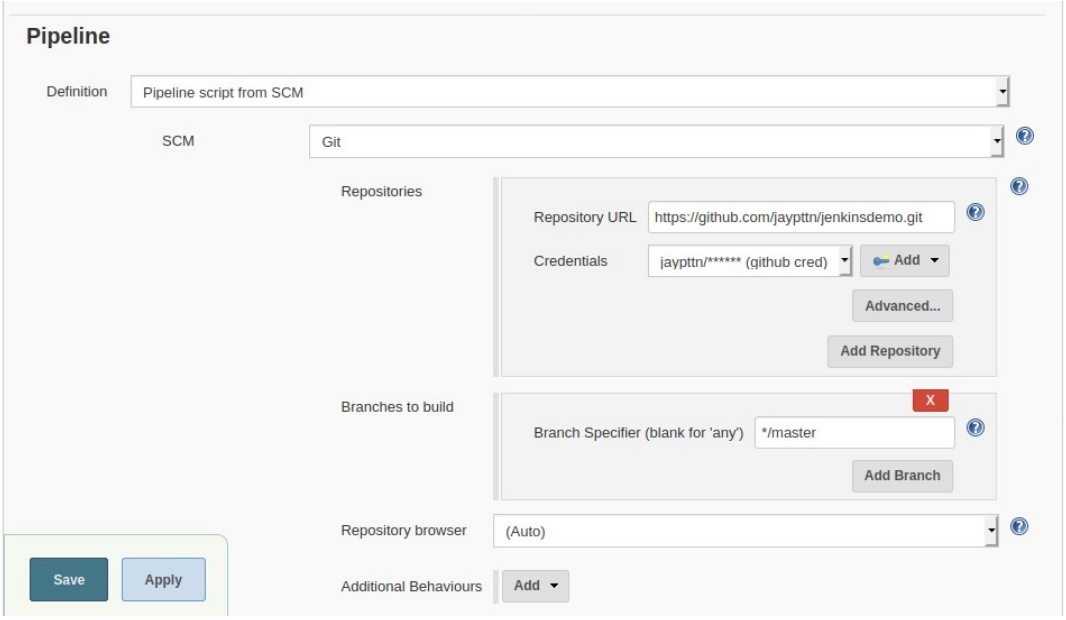
1 contributor

12 lines (10 sloc) | 271 Bytes Raw Blame History

```
def info(message) {
        echo  "\033[4;32mINFO:\033[0m \033[1;33m${message}\033[0m\n"
}

def warning(message) {
    echo "WARNING: \033[1;31;43m${message}\033[0m"
}

def gitCommitId(message)
{
      echo  "\033[4;32m[Git Commit ID] \033[0m \033[1;33m ${message} \033[0m\n"
}
```




```
[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
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
 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.

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