**nPipeline Code - RK**

1. try and catch block
2. error statement
3. Jenkins default variables
4. assigning values to variables
5. calling the variables which has some values
6. println , echo
7. loop statements (if else , while)
8. halt the pipeline and get input values from user and pass it to script
9. pause the pipeline – sleep
10. proceed or abort the pipeline – get the values
11. sending input as json and receive the o/p and print the values
12. creating and calling funtion in the script
13. comment the code and codes
14. **try and catch block**

stage(‘Creating CR’){

try{

INC\_VALIDATION()

CR\_Creation()

}catch (Exception e){

echo ”there is some issue in the snow portal in creating CR please check the CR portal manually”

throw e

}

}

Stage(‘creating CR\_Task’)

try{

CR\_TASK\_CREATION

}catch (Exception e){

echo ”there is some issue in the snow portal in creating CR please check the CR portal manually”

throw e

}

}

1. error statement

{

if(incidentstate==”[closed]” || incident state==”[resolved]”

echo ”################”

echo “provided incident is invalid”

echo ”################”

error (“hence terminating pipeline”)

}

**Else**

**{**

echo ”################”

echo “provided incident is valid”

echo ”################”

**}**

1. **Jenkins default variables**

**&**

1. **assigning values to variables**

pipeline{

agent any

stages{

stage("Env Build Number"){

steps{

echo "The build number is ${env.BUILD\_NUMBER}"

echo "You can also use and \${BUILD\_NUMBER} -> ${BUILD\_NUMBER}"

}

}

}

}

1. calling the variables which has some values
2. println , echo

string jobDetails JSON = sh returnstdout: true, script:

"curl -f -k -u ${USERNAME}:${PASSWORD} -H 'Content-Type: application/json' -XPOST https://

jobInfo - deserializeJSON(jobDetails JSON)

jobID = jobInfo.id

echo "Printing jobID: $jobID"

}

println "Monitoring Ansible job number: ${jobID}" await TowerJobCompletion (jobID)

//ansibleJobOutput (jobID)

1. loop statements (if else , while)

class Example {

static void main(String[] args) {

// Initializing a local variable

int a = 2

//Check for the boolean condition

if (a<100) {

//If the condition is true print the following statement

println("The value is less than 100");

} else {

//If the condition is false print the following statement

println("The value is greater than 100");

}

}

}

1. halt the pipeline and get input values

from user and pass it to script

**9 . pause the pipeline – sleep**

pipeline {

agent any

stages {

stage("numberone"){

steps {

sh 'echo pipelinejob1'

}

}

stage("numbertwo"){

steps {

sh 'sleep 10'

}

}

stage("numberthree"){

steps {

sh 'echo pipelinejob2'

}

}

}

}

**11. sending input as json and receive the o/p and print the values**

**timeout(time:10, unit: MINUTES)**

**String jobstatus JSON sh returnstdout: true, script: "curl -f k-u ${USERNAME}:$(PASSIIRD) Content-Type: application/json' -GET https://$(Ansible**

**def jobstatus deserialize150(jobstatus)JSON)**

**if(jobstatus.status "pending" jobstatus.status "running" | jobstatus.status waiting"){**

**println\u001b[34mJob is $(jobstatus.status), sleeping for 30 seconds\u001b["**

**sleep(10)**

**} else if (jobstatus.status "successful"){**

**string stdout, sh returnstdout: true, script: "curl-fku $[USERNAME}:$PASSWORD) "Content-Type: application/json" GET https://$(Ansit printin \u001b[mAnsible job output \u001b[on**

**printin stdout1**

**print in "Au001b[34mAnsible job completed successfully\uperblom**

**allDone - "yes"**

**} else {**

**plavbook status - Jobstatus.status.**

**String stdout?sh returnstdout: true, script: -k-u $(env.USERNAME) :$(env.PASSWORD) -H "Content-Type: application/json" GET https:/**

**println stdout?>**

**print in \u001b[31mAnsible job failed\u001b[em"**

**currentBuild.result DEPLOY ISSUE";**

**alldone- "yes"**

**}**

1. **creating and calling funtion in the script**

stage(cal)

{

Calculator()

}

def calculator()

{

X=10+5

echo$ {X}

}

1. **comment the code and codes**

pipeline{

agent any

parameters {

string defaultValue: 'dineshkumar', description: 'enter your name', name: 'Name'

}

/\*

stages{

stage("dineshdemo"){

steps{

echo "hi ${name}, welcome to thusday"

}

} /\*

}

}