SharedLibraryJenkinswithresources

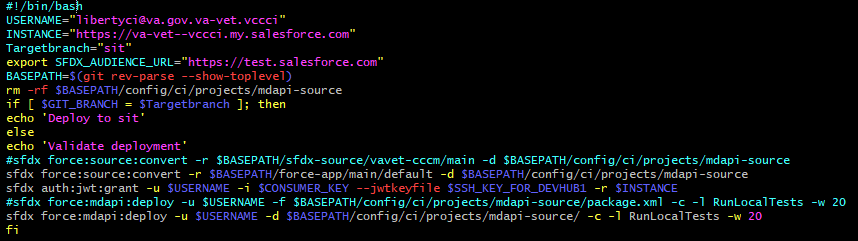
In the shared repo

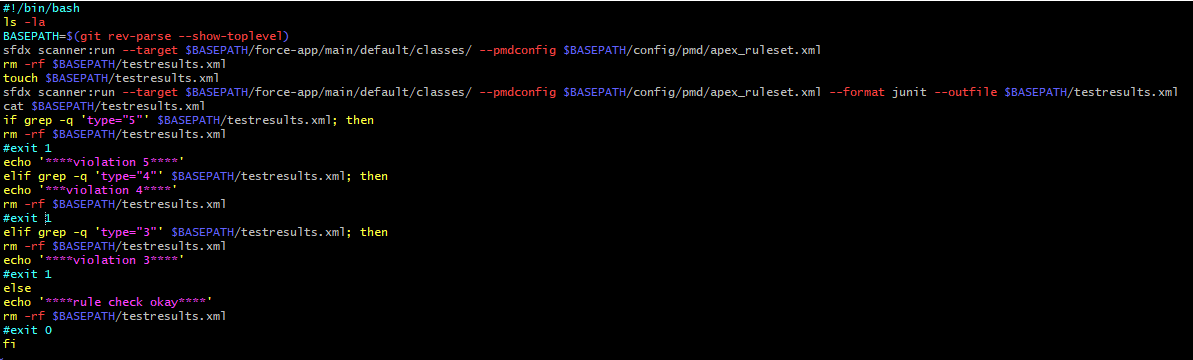
\*\*This repository is what will be configured in Jenkins @ manage Jenkins > configure system to let Jenkins know the repo to use

1. A shared location/repo for the var file i.e .groovy script located in vars directory. Vars directory is at root location locations u normally have Jenkinsfile. Then u create the below .groovy script e.g testsharedpipeline.groovy



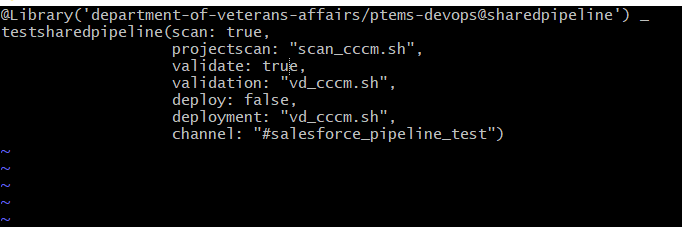
1. In the resources directory same level with vars directory above the scripts to be called are located





In the repo that needs deployment done

1. Create a Jenkinsfile at root level



@Library('department-of-veterans-affairs/ptems-devops@sharedpipeline') \_

Above tells jenkins to go to the library name “department-of-veterans-affairs/ptems-devops” configured in Jenkins at this sharedpipeline' branch. Normally if you don’t want Jenkins to go to the configured branch normally main or master you can specify which branch by using the @sharedpipeline to tell Jenkins don’t u the configured default branch in Jenkins but use the one I specify.

Testsharedpipeline(

Tells Jenkins that sthe .grrovyscript I want you to use in the vars directory in the shared repo because they may be more than one groovy script located

scan: true,

projectscan: "scan\_cccm.sh",

This says that if the scan is set to true in the above Jenkinsfile then the scan\_cccm.sh script will run but if set to false scan\_cccm.sh script will not be executed

LANGUAGE BEHIND the entire doc

Created a variable in DevOps-PTEMS repo - Vars directory ..that will house our master variable for the projects

which is basically what gets executed

So we have a call() method.

Tells Jenkins what method to run when the var is called from any Jenkinsfile.

Also have a parameter for the variable, A map called ptemspipeline

The ptemspipeline) parameter allows u to pass named parameters to the call method.

Named parameters in this case are etc scan, deploy. validation etc

loads the the script in resources folder as a string scriptcontent

take string (scriptcontent) and write it out using WriteFile to the workspace we are now running on

make the it executable and run it