How To - DSL Jobs for Jenkins Jobs

1. Plugin installation in Jenkins -

There are few plugins need to install in Jenkins to use Docker within Jenkins jobs and Pipeline. Go to Manage Jenkins >> Manage Plugins >> Available tab >> search for "Job DSL Plugin" then select and install Plugin.

2. Job DSL Plugin Introduction -

The Job-DSL-Plugin allows the programmatic creation of projects using a DSL. Pushing job creation into a script allows you to automate and standardize your Jenkins installation, unlike anything possible before.

After installing the plugin, you'll get a new Build Step entry named "Process Job DSLs". Type the DSL directly in the text box, or point to a file in the workspace.

2.1 Basic -

The DSL allows the definition of a job, and then offers a useful set of functions to configure common Jenkins items. A *configure* is available to give direct access to the *config.xml* before generating the job. The script is groovy code, which can be very powerful.

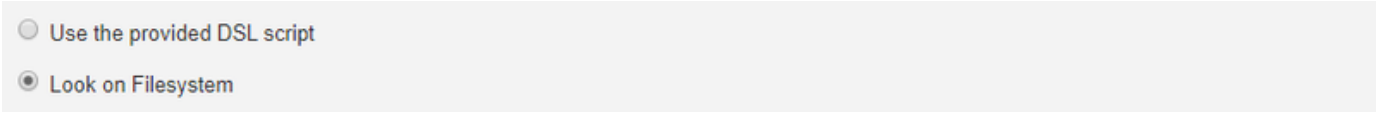
2.2 Features -

- DSL - Scriptable via Groovy
- DSL - Direct control of XML, so that anything possible in a config.xml is possible via the DSL
- DSL - Helper methods for common job configurations, e.g. scm, triggers, build steps
- Plugin - DSL can be put directly in a job
- Plugin - DSL can be put into SCM and polled using standard SCM triggering
- Plugin - Multiple DSLs can be referenced at a time
- Plugin - Tracks Templates used, will update derivative jobs when template is change

3. Creating a Seed Job -

From Jenkins main page, select New Job then provide the name and select a Free Style job then Save the job. On the configure screen, scroll down to the "Build: Add build step" pull down menu and select option "Process Job DSLs".

There are two ways to provide the DSL scripts in Jenkins job-



3.1 Use the Provided DSL script -

If user select this option then a text area will open, where user can write or paste DSL script.

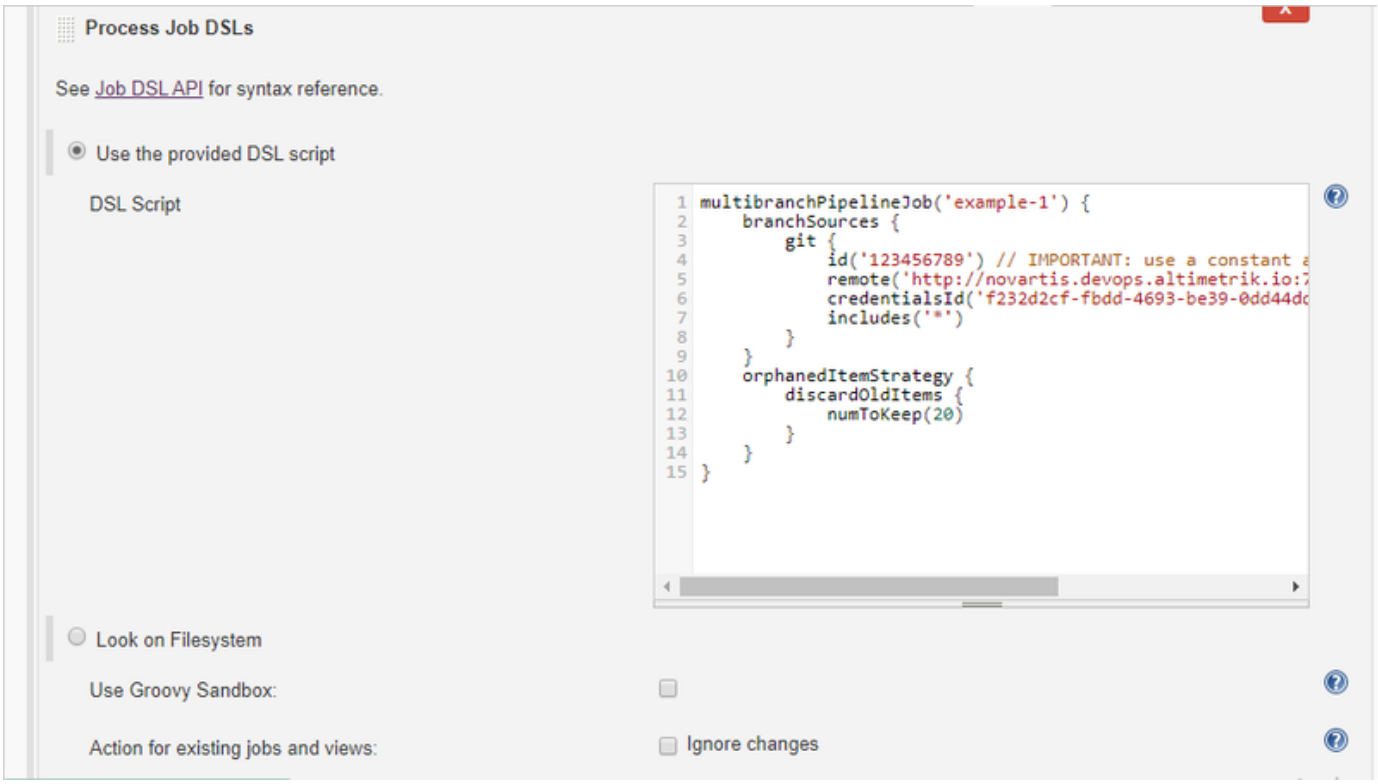
3.2 Look on Filesystem -

In this option, all the groovy scripts are checked-in into SCM system like GitLab or BitBucket. During SCM step all the scripts will be stored in workspace and any script names are mentioned here will be executed.

Newline separated list of DSL scripts, located in the Workspace can be mentioned. Scripts are executed in the same order as specified. The execution order of expanded wildcards is unspecified.

4. Working Example To Create FreeStyle Job -

Below mentioned code will create Freestyle job for repository/project i.e. quidryan/aws-sdk-test mentioned, so it will create multiple Jenkins jobs with respect to each branch present in this repository. Here this example has been presented with the reference of **3.1 Use the Provided DSl Script** option.

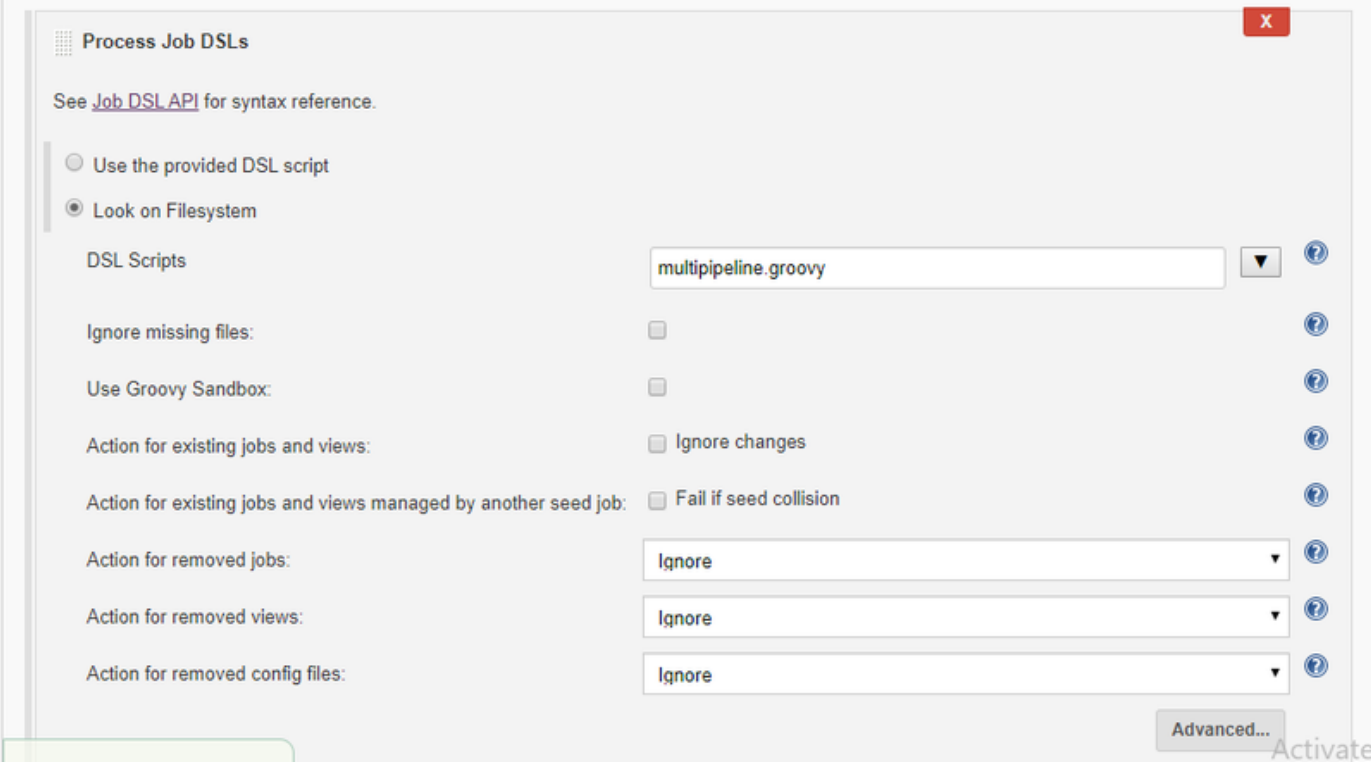



```
def project = 'quidryan/aws-sdk-test'
def branchApi = new
URL("https://api.github.com/repos/${project}/branches")
def branches = new
groovy.json.JsonSlurper().parse(branchApi.newReader())
branches.each {
    def branchName = it.name
    def jobName = "${project}-${branchName}".replaceAll('/','-')
    job(jobName) {
        scm {
            git("git://github.com/${project}.git", branchName)
        }
        steps {
            maven("test -Dproject.name=${project}/${branchName}")
        }
    }
}
```

Jenkins Job created with above examples, as this repository have 3 branches (Master, Feature1 and Feature2) -

5. Working Example To Create Multi Branch Pipeline -

Below mentioned example will create the multi-pipeline jobs with respect to remote repository is given. Here this example has been presented with the reference of **3.2 Look on Filesystem** option. For this need to create a groovy script and need to store in repository's root folder and need to mention the name like "multipipeline.groovy".



DSL Script (multipipeline.groovy) code -

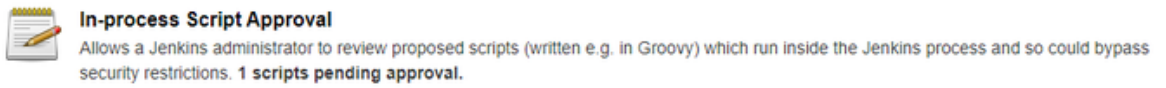
```
multibranchPipelineJob('example-2') {
    branchSources {
        git {
            id('123456789') // IMPORTANT: use a constant and unique
identifier

remote('http://novartis.devops.altimetrik.io:7080/altimetrik-poc/demo-so
nar.git')
   credentialsId('f232d2cf-fbdd-4693-be39-0dd44ddf1663')
            includes('*')
        }
    }
    orphanedItemStrategy {
        discardOldItems {
            numToKeep(20)
        }
    }
}
```

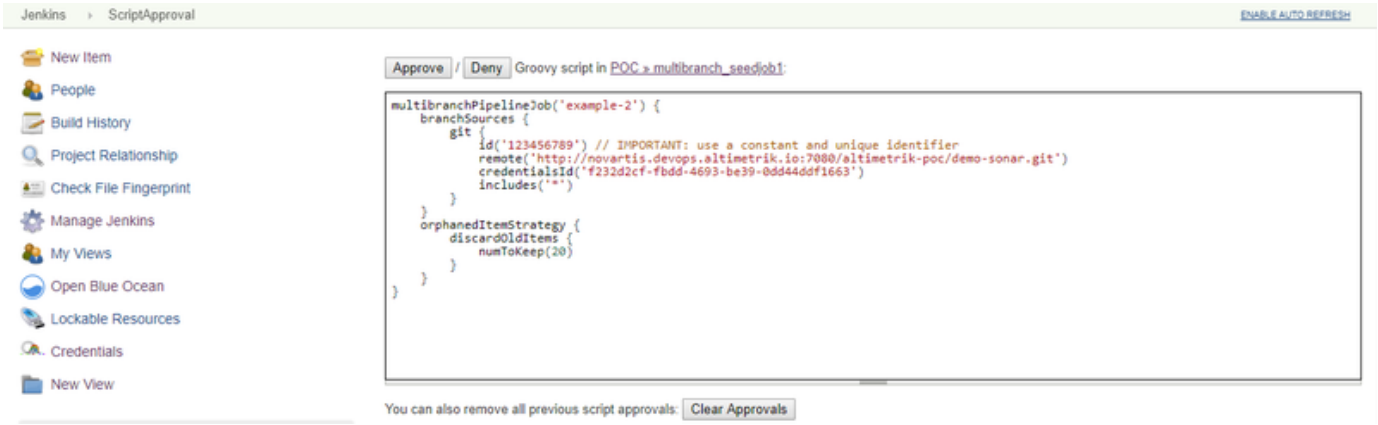
Once you are trying to "Build Now" the job then it will be failed and Console Output will show below mentioned error-

```
Processing DSL script multipipeline.groovy
ERROR: script not yet approved for use
Finished: FAILURE
```

This error is coming as Script is not approved in Jenkins. To approve the script, go to Jenkins Manage Jenkins In-process Script Approval.

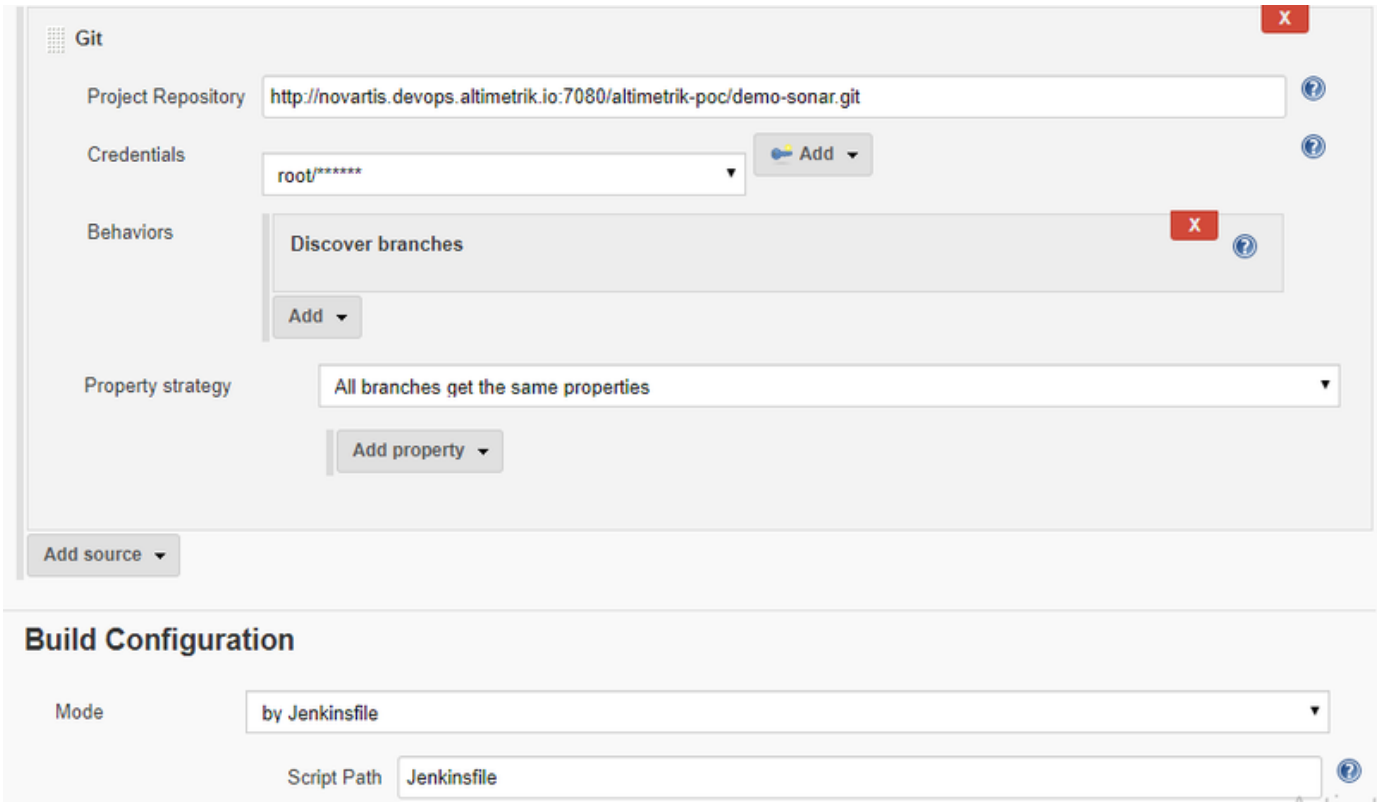


Here admin has right to Approve or Deny and can have multiple approval clearance required. Once script is approved and try to run the job again.



A new seed job has been created i.e. example-2 (multibranch-pipeline job). Go inside of this job and click on "Scan Multibranch Pipeline Now" and refresh the page. It will detect all the branches and create the job accordingly as below.

example-2 job configure screen-shot as below-



Here any branch can be build and Jenkinsfile will be picked based on the "Repository/branch name". All branch can have different Jenkinsfile

based on the requirement and job will automatically picked the correct Jenkinsfile with respect to branch.

6. References -

- https://jenkinsci.github.io/job-dsl-plugin/#path/multibranchPipelineJob
- https://gist.github.com/djfdyuruiry/e3c891c6204bea602e770f9bf7a0cb1c
- https://issues.jenkins-ci.org/browse/JENKINS-49615
- https://artem.services/?p=877&lang=en
- https://tech.gogoair.com/jenkins-jobs-as-code-with-groovy-dsl-c8143837593a