Jenkins Job-DSL Workshop

Anton Weiss
Otomato
http://otomato.link

Install the Plugin

- Manage Jenkins -> Configure System -> Manage Plugins -> Available
- Search for ‘job-dsl’
- Install

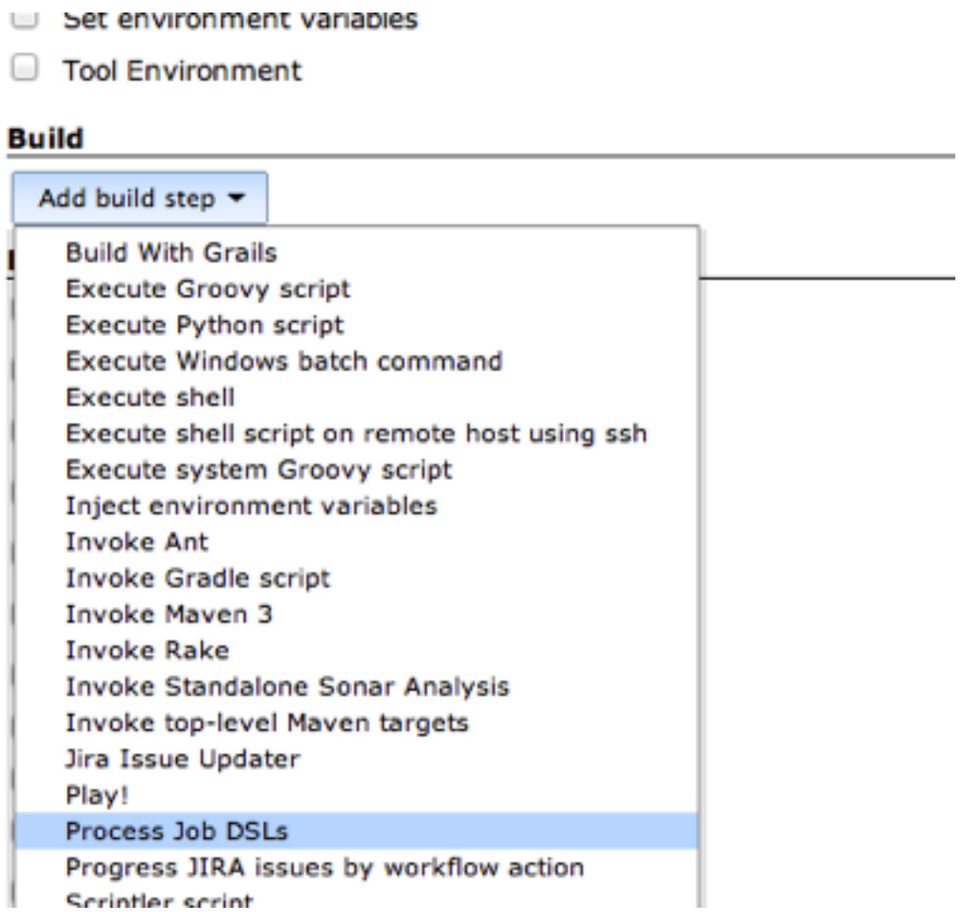
1. Creating the Seed Job

- New Item -> Freestyle Project
- Name it “job-dsl-workshop”

1. Creating the Seed Job

- Configure the ‘job-dsl-workshop’ job:
- Build: Add build step:
- From the pull down menu, select "Process Job DSLs". You should be presented with two radio buttons. The default will be "Use the provided DSL script" and a text input box will be displayed below it.

1. Creating the Seed Job



1. Creating the Seed Job

- Copy the following DSL Script block into the input box:

```
job('DSL-Tutorial-1') {
    scm {
        git('https://github.com/antweiss/mvnJunitExampe.git')
    }
    triggers {
        scm('H/15 * * * *')
    }
    steps {
        maven('-e clean test')
    }
}
```

- Click the "Save" button. You'll be shown the overview page for the new Seed job you just created.

2. Run the Seed Job

- The Seed Job is now all set up and can be run, generating the Job we just scripted.

- Click on ‘Build Now’

- Look at the build result to see a link to the new Job which has been created by the running of your DSL script in the Seed Job. You should see this in the section called "Generated Jobs".

- Follow this link to your new job. You can run this new script-generated job manually or wait the 15 minutes for the scm trigger to kick in.

Exercise

- Change the seed job script to:
  - Checkout code from https://github.com/antweiss/jenkins-groovy-scripts.git
  - Instead of calling maven - call ‘ls’ shell command:

Example:

```
steps {
        shell('echo Hello World!')
    }
```

Rerun the seed job.

3.Create Jobs for Multiple Branches

```
def project = 'antweiss/mvnJunitExampe'
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}-${branchName}".replaceAll('/','-')
    job(jobName) {
        scm {
            git("https://github.com/${project}.git", branchName)
        }
        steps {
            maven(“clean test -Dproject.name=${project}/${branchName}")
        }
    }
}
```

Exercise

- Use the original seed job script from slide 6
- Add a post-build step to archive junit reports:

```
job('example-1') {
    steps {
                …
    }
    publishers {
        archiveJunit('**/target/surefire-reports/*.xml')
    }
}
```

- Run the seed job
- Run the resulting job
- Verify the unit test results are getting archived

Exercise:

- Change the seed job to generate a new job that:
  - Has following environment variables defined: Name, Nickname
    - Example:

```
job('example-1') {
    environmentVariables(FOO: 'bar', TEST: '123')
}
```

  - Runs a shell step to output these variables
  - Runs another shell step

4. Generated DSL

- Click on ‘See Job DSL API for syntax reference.’
- Explore:
  - job
  - listView

Exercise:

- Create a new seed job with a script that:
  - Creates 2 jobs “one” and “two” (loop on a list with .each()) - each with 2 parameters:
    - A boolean named “FLAG”
    - A string named “INPUT”
    - Each job:
      - checkout code from https://github.com/antweiss/jenkins-examples.git
      - call Maven with ‘clean install -Dproject.name=example_${jobName}’
  - Creates a list view with these 2 jobs.