Supercharged Configuration As Code

Bulk Updates with Job DSL and System Groovy

Who Am I

- Alan Beale

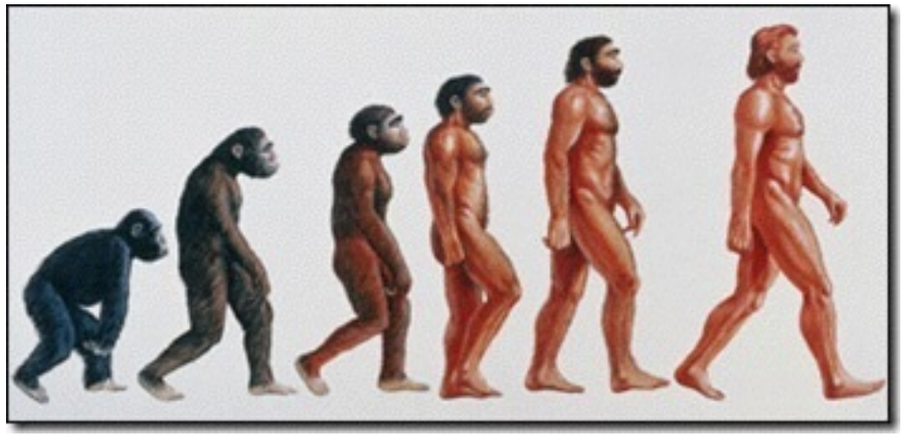
  Build Engineer @ Chicago Trading Company

Any opinions expressed are my own and do not represent the opinions of my employer

Observations about build jobs

- Jobs Change over Time
- Most jobs are like other jobs

Jobs Change Over Time



- Correctness Changes
- Bug Fixes
- Complexity Increase

Most Jobs are Similar

- Small number of themes with structured variation

How do these observation help?

- Job DSL enables us to using coding techniques to create job configurations
- System groovy scripts enable use to automate repeated calls to jobs

Player Piano

Our metaphor for bulk job regeneration

The Piano Roll

- Controls the sound of the piano
- Orchestrates the automation of job recreation

Job DSL for gradle build

```
job {

    name "git-gradle-simple-build"

    scm {

        git('/Users/alanbeale/sandbox/samples/java/basic',
                '*/master'  )

    }

    steps {

        gradle ( "build", null, true )

    }

}
```

Evolution Example

```
job {
    name "git-gradle-build-with-worspace-option"
    scm {
        git('/Users/alanbeale/sandbox/samples/java/basic',
            '*/master')
    }
    steps {
        gradle ( "build", null, true
        ){ node -> //hudson.plugins.gradle.Gradle
            node/useWorkspaceAsHome(true)
        }
    }
}
```

Add Gerrit Trigger

```
job {

    name "git-gradle-build-with-gerrit-trigger"

    scm {
        git(
            '/Users/alanbeale/sandbox/samples/java/basic',
            '$GERRIT_BRANCH'
        ){ node ->
            node/extensions {
                'hudson.plugins.git.extensions.impl.BuildChooserSetting'{
                    buildChooser(buildChooserAttrs){
                        separator('#')
                    }
                }
            }
            def remoteConfig = node/userRemoteConfigs/
            'hudson.plugins.git.UserRemoteConfig'
            remoteConfig << refspec('$GERRIT_REFSPEC')
        }

}
```

continued

```
triggers {
    gerrit {
        events {
            PatchsetCreated
            DraftPublished
        }
        project(
            "plain: /Users/alanbeale/sandbox/samples/java/basic",
            'ant:**'
        )
    }
}
steps {
    gradle ("build", null, true)
    { node -> //hudson.plugins.gradle.Gradle
        node/useWorkspaceAsHome(true)
    }
}
}
```

Repeated elements

- Job Name
- Git URL
- Gerrit Trigger Project

Towards Job Factory

- Factor out common operations
- Parameterize calling Job DSL

Create Utility class

- jobdsl.JobUtilities methods
  - setupGradleGerritJobName()
  - setupGerritScm()
  - setupGerritTrigger()
  - setupGradleBuild()

generator job DSL

```
job {
    name "gradle-gerrit-job-generator"
    parameters {
        stringParam("PROJECT_NAME","sample/java/library")
    }
    steps {
        dsl {
            removeAction('IGNORE')
            ignoreExisting(false)
            text('''
import jobdsl.JobUtilities
import javaposse.jobdsl.dsl.Job
def gerritProjectName= "${PROJECT_NAME}"
def simpleJavaVerifyJob = job {}
JobUtilities.setupGradleGerritJobName(simpleJavaVerifyJob, gerritProjectName)
JobUtilities.setupGerritScm( simpleJavaVerifyJob, gerritProjectName)
JobUtilities.setupGerritTrigger( simpleJavaVerifyJob, gerritProjectName)
JobUtilities.setupGradleBuild(simpleJavaVerifyJob)
''')
```

Part 2 : System Groovy

Regeneration script

```
import jenkins.model.*

import hudson.model.*

def jenkins = jenkins.model.Jenkins.instance
def jobGeneratorName='gradle-gerrit-job-generator'
def jobGenerator = jenkins.getItem(jobGeneratorName)
def projectParamName ='PROJECT_NAME'
def projectParamValue=‘sample/java/library'

if(jobGenerator){
    boolean buildQueued =
        jobGenerator.scheduleBuild(  5,
            new Cause.UserIdCause(),
            new ParametersAction(
                new StringParameterValue(projectParamName, projectParamValue)
            )
        )
}
```

Regenerate from existing job name

```
import jenkins.model.*

import hudson.model.*

def jenkins = jenkins.model.Jenkins.instance
def jobGeneratorName='gradle-gerrit-job-generator'
def jobGenerator = jenkins.getItem(jobGeneratorName)
def projectParamName ='PROJECT_NAME'
def jobs = jenkins.getItems()

def gerritJobs = jobs.findAll { job ->
 job.name.endsWith('gradle-gerrit')
}

def projectNames =gerritJobs.collect {  item ->
    item.triggers.values().find {
        it.class.simpleName =='GerritTrigger'
    }. gerritProjects[0].pattern
}

if(jobGenerator){
    projectNames.each {  projectName ->
        boolean buildQueued =
            jobGenerator.scheduleBuild( 5,
                new Cause.UserIdCause(),
                new ParametersAction(
                    new StringParameterValue( projectParamName,projectName)
                )
            )
    }
}
```

Example of Job Themes

- Continuos Integration Jobs
- Gerrit Verification Jobs
- Release Jobs
- Privileged SCM Task Automation

Summary

- Job DSL enables precise control
- Define parameterized Job Generator (the piano)

Summary

- System Groovy Jobs
- automates calling job generator in bulk

(piano roll)

Thank You To Our Sponsors

Platinum

CloudBees

Gold

BlazeMeter
THE LOAD TESTING CLOUD

keynote

zend

XebiaLabs
Deliver Faster

ravello systems

elasticbox

JFrog

perfecto mobile

black diamond SOFTWARE

Silver

SOASTA
Test Faster. Release Sooner.

YAHOO!

SERENA

choose digital

|opensourcearchitect|

Sauce LABS

Corporate

cloudera
Ask Bigger Questions

Thank you

- Source Code at https://github.com/bealeaj1214/jenkins-juc-job-dsl-bulk-regeneration