

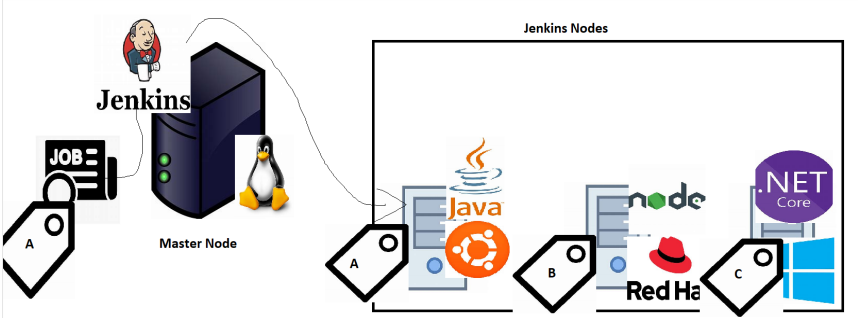


FEBRUARY 28, 2023

# DevOps Classroomnotes 28/Feb/2023

## Jenkins Multi node configuration

- Multi node configuration using pem files/private files



- Configure Game of life

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

☐ Execute concurrent builds if necessary

JDK  
JDK to be used for this project  
JDK\_8\_UBUNTU

☒ Restrict where this project can be run  
Label Expression  
MAVEN\_JDK8  
Label MAVEN\_JDK8 matches 1 node. Permissions or other restrictions provided by plugins may further reduce

Advanced...

Dashboard

Check File Fingerprint

Manage Jenkins

My Views

Build Queue

Build Executor Status

gol-fs

N/A

N/A

N/A

Atom feed for all

Atom feed for failures

Atom feed for just latest builds

1 Idle

2 Idle

1 Idle

2 Idle

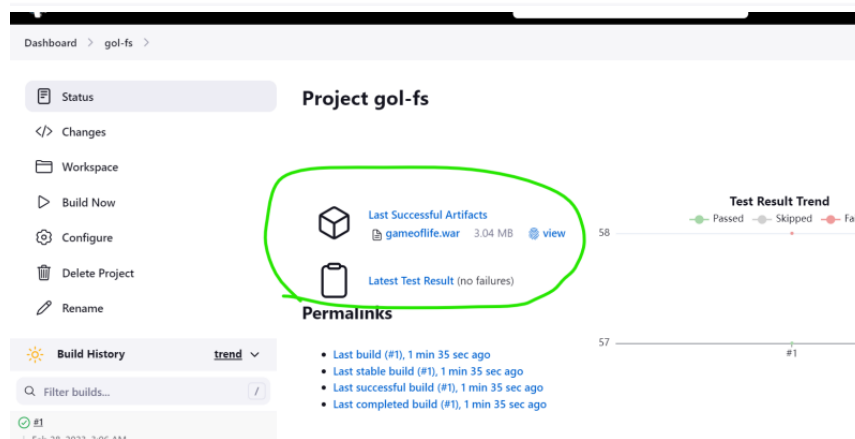
REST API Jenkins 2.37

```

Dashboard > gol-fs > #1

[01:34mINFO[m] Building war: /home/ubuntu/remote/workspace/gol-fs/gameoflife-web/target/gameoflife.war
[01:34mINFO[m] WEB-INF/web.xml already added, skipping
[01:34mINFO[m] [!m-----[!m
[01:34mINFO[m] [!mReactor Summary for gameoflife 1.0-SNAPSHOT:[!m
[01:34mINFO[m]
[01:34mINFO[m] gameoflife ..... [!;32mSUCCESS[m [ 15.302 s]
[01:34mINFO[m] gameoflife-build ..... [!;32mSUCCESS[m [ 5.524 s]
[01:34mINFO[m] gameoflife-core ..... [!;32mSUCCESS[m [ 8.933 s]
[01:34mINFO[m] gameoflife-web ..... [!;32mSUCCESS[m [ 5.472 s]
[01:34mINFO[m] [!m-----[!m
[01:34mINFO[m] [!;32mBUILD SUCCESS[m
[01:34mINFO[m] [!m-----[!m
[01:34mINFO[m] Total time: 39.684 s
[01:34mINFO[m] Finished at: 2023-02-28T03:07:22Z
[01:34mINFO[m] [!m-----[!m
Archiving artifacts
Recording test results
[Checks API] No suitable checks publisher found.
Finished: SUCCESS

```



## Jenkins 2

- This is all about Pipeline as code
- Pipeline as a code refers to creating CI/CD Pipeline in some file format and version control (generally closer to code)
- Jenkins 1 was predominantly UI oriented, where as in Jenkins 2 the concept of pipelines were introduced natively.
- Now we write our build steps in these pipelines and then version control.
- Jenkins has created 2 types of pipelines
  - Scripted Pipeline:
    - This pipeline allows to use Groovy Language directly
    - This is a different approach
  - Declarative Pipeline
    - This pipeline internally uses Groovy but we use Jenkins DSL (Domain Specific Language)
    - Similar for the benefit of classic jenkins users

## Groovy

- There are two popular Java Based Languages
  - Scala (Big Data Purposes)
  - Groovy (Scripting purposes)

## Lets Quickly build Gameoflife on node1 using

### Scripted pipeline

- Lets look at below steps for an overview

The image shows two screenshots from the Jenkins web interface. The top screenshot is the 'Enter an item name' dialog where 'gol-scriptedpipeline' is entered. Below this, three project types are listed: 'Freestyle project', 'Pipeline', and 'Multiconfiguration project'. The 'Multiconfiguration project' option is highlighted with a green circle and an 'OK' button. The bottom screenshot shows the 'Configure' page for the 'gol-scriptedpipeline' project. The 'Definition' is set to 'Pipeline script'. The 'Script' field contains a Groovy pipeline script. The 'Save' button is highlighted with a green circle.

**Enter an item name**

gol-scriptedpipeline

» Required field

**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any for something other than software build.

**Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pip and/or organizing complex activities that do not easily fit in free-style job type.

**Multiconfiguration project**  
Projects that need a large number of different configurations, such as testing on multiple operating systems, architectures, or hardware configurations.

**OK**

**Configure**

Definition: Pipeline script

Script ?

```

1 node('MAVEN_JDK8')
2 {
3   stage('vcs')
4   {
5     git 'https://github.com/wakaleo/game-of-life.git'
6   }
7   stage('build')
8   {
9     sh 'mvn package'
10  }
11  stage('postbuild')
12  {
13    archiveArtifacts artifacts: '**/target/gameoflife.war', followSymlinks: false
14    junit '**/surefire-reports/TEST-*.xml'
15  }
16 }

```

Use Groovy Sandbox ?

**Save** **Apply**

```

node('MAVEN_JDK8')
{
  stage('vcs')
  {
    git 'https://github.com/wakaleo/game-of-life.git'
  }
  stage('build')
  {
    sh 'export PATH="/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin:$PATH" && mvn package'
  }
  stage('postbuild')
  {
    archiveArtifacts artifacts: '**/target/gameoflife.war', followSymlinks: false
    junit '**/surefire-reports/TEST-*.xml'
  }
}

```

- Declarative

```

pipeline {
  agent { label 'MAVEN_JDK8' }
  stages {
    stage('VCS') {
      steps {
        git 'https://github.com/wakaleo/game-of-life.git'
      }
    }
    stage('build') {
      steps {
        sh 'export PATH="/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin:$PATH" && mvn package'
      }
    }
  }
}

```

## Leave a Reply

Enter your comment here...

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)



## About continuous learner

devops & cloud enthusiastic learner

[VIEW ALL POSTS](#)

◀ [PREVIOUS POST](#)

[Azure Classroomnotes 28/Feb/2023](#)

[NEXT POST](#)

[AWS Classroomnotes 28/Feb/2023](#)

POWERED BY [WORDPRESS.COM](#).