Continuous Integration with jenkins-pipeline for Java, Jira, Slack, Maven, Wildfly and Soapui.

schmittjoaopedro Java, jenkins 4 04+00:00 Julho 04+00:00 20194 04+00:00 Julho 04+00:00 2019 5 Minutes

One of the most difficulties that I have found during the Jenkins-pipeline development for a Java application with Maven and Wildfly was: how to integrated this old-fashioned java application with many different tools used by the development team? Today there are many posts demonstrating single pieces of the puzzle, however, there is no guide that unifies all these pieces in a single overall example. Therefore, my idea here is to grab all these different pieces of information that I had collected to solve a specific problem and create an applied scenario using declarative Jenkins-pipeline.

The idea here is to create a continuous integration process. This process uses a beta server to execute integration and functional tests to verify if the application is working correctly. To this, the following steps must be executed:

1. Show the current version of the project managed by Jira;
2. Build the application using maven, where the application tests are executed locally (Unitary and Integration tests);
3. Un-deploy the current war version from the beta server;
4. Copy jar libs for a network folder. This app requires these files to execute some internals;
5. Restart the best server;
6. Deploy new war version on beta;
7. Execute functional tests (Integration, Services, and Load). These are Soapui tests that have their own pipeline on Jenkins;
8. Create a tag on source code to manage the latest stable version;
9. Publish the latest stable war on nexus;
10. Send a notification on Slack about the build status;

The general idea is that the pipeline should execute the build and verify if everything works well before to publish the app on production. In this process, firstly we make the build locally (using maven) and then we publish the war on the beta server. After that, sub-processes on Jenkins are started (using the build directive) and all these are waited to execute completely, even if one of these brokes (directive catchError). It allows to execute all tests and verify the ones that broke, in the other case, the build would be stoped in the first failure and after fixing the broken test we must execute the build again to check if the next tests are ok. Finally, after all, tests had passed, we update the latest tag on git to point for the hash of the latest stable version, and we update the artifact on nexus. At this time, a continuous deployment could verify this build status and make a deploy on production. To communicate the build status, a slack message is sent for the app team.

The following Jenkins-pipeline file describes in details the process:

```
#!groovy

// By default this pipeline makes checkout from the app git repository on the
pipeline {

    // Uses a docker agent with java, maven and git installed
    agent { label 'app-pipeline-deploy' }

    environment {
        // Defines the connection configuration name for the jira-steps-plugi
        JIRA_SITE = 'JIRA-STEP'
    }

    stages {
        stage('Version information from Jira') {
            steps {
                // Changes to the development directory from the project repo
                dir('development') {
                    // Opens a groovy script environment
                    script {
                        // Read the current jira version from a file managed
                        def currentJiraVersion = readFile "jira.version"
                        // Obtain information from jira related with the curr
                        def version = jiraGetVersion id: currentJiraVersion
                        // Log on jenkins console information about the curre
                        def logText =
                            "=================== TESTS ON BETA FOR NEXT JIRA
                            "ID: " + version.data.id + "\n" +
                            "NAME: " + version.data.name + "\n" +
                            "DESCRIPTION: " + version.data.description + "\n'
                            "RELEASED: " + version.data.released + "\n" +
                            "==================================================
                        echo logText
                    }
                }
            }
        }

        stage('Tests and Build') {
            steps {
                dir('development') {
                    // Use groovy to execute shell commands
                    script {
                        // Give full permision for the build on the developme
                        sh 'chmod -R 777 *'
                        // Executes maven tests (unitary and integration) and
                        sh 'mvn -B install'
                    }
                }
            }
        }

        stage('Un-deploy from beta') {
            steps {
                dir('development') {
                    script {
                        // Uses jboss maven plugin to undeploy the current ap
                        sh 'mvn -B -DjbossHost=app-beta.app.com -DjbossUser=x
                    }
                }
            }
        }

        stage('Copy app libs') {
            steps {
                dir('development') {
                    // Uses groovy to extract jar libs from the app and copy
                    script {
                        sh 'mkdir app-lib-temp'
                        sh 'cp app/target/app.war app-lib-temp'
                        sh 'cd app-lib-temp && jar -xf app.war'
                        sh 'cd ..'
                        sh 'rm -rf /applib/*'
                        sh 'cp app-lib-temp/WEB-INF/lib/* /applib'
                    }
                }
            }
        }

        stage('Restart beta') {
            // Uses the jenkins master to restart the beta. It is used becaus
            agent { label 'master' }
            options { skipDefaultCheckout(true) } // Ignore git checkout
            steps {
                script {
                    // Execute ssh command and ignore any response message th
                    sh 'sudo ssh app-beta.app.com \'service jboss-as-beta res
                    // Wait server to restart
                    sh 'sleep 120'
                }
            }
        }

        stage('Deploy on beta') {
            steps {
                dir('development') {
                    // Back to the docker container, execute deploy of the ge
                    script {
                        sh 'mvn -B -DjbossHost=app-beta.app.com -DjbossUser=x
                    }
                }
        }
    }

    // In this stage, sub-builds are started by this build on jenkins.
    // It means that the current build will wait until those builds finis
    // Builds are executed in parallel to speed-up the execution time.
    // Finally, those builds are executed pointing to the fresh version c
    stage('Functional tests') {
        parallel {
            stage('Integration tests') {
                steps {
                    catchError {
                        build 'soapui-test-pipeline-name1'
                    }
                    catchError {
                        build 'soapui-test-pipeline-name2'
                    }
                }
            }
            stage('Services tests') {
                steps {
                    catchError {
                        build 'soapui-test-pipeline-name3'
                    }
                    catchError {
                        build 'soapui-test-pipeline-name4'
                    }
                }
            }
            stage('Load tests') {
                steps {
                    catchError {
                        build 'soapui-test-pipeline-name5'
                    }
                    catchError {
                        build 'soapui-test-pipeline-name6'
                    }
                }
            }
        }
    }

    stage('Create latest tag') {
        // If the build not failed in any stage (including the sub-builds
        when {
            expression { currentBuild.currentResult == 'SUCCESS' }
        }
        steps {
            // Uses credentials configuration from jenkins to execute com
            withCredentials([usernamePassword(credentialsId: 'xxxxxxxx-xx
                sh '''
```

```
                    git push http://$USER_NAME:$USER_PASS@sourcecode.app.
                    git tag -f latest
                    git push http://$USER_NAME:$USER_PASS@sourcecode.app.
                '''
            }
        }
    }

    stage('Publish on Nexus as latest') {
        // If the build not failed in any stage (including the sub-builds
        when {
            expression { currentBuild.currentResult == 'SUCCESS' }
        }
        steps {
            dir('development') {
                script {
                    sh 'mvn -B deploy -DskipTests -Dpmd.skip=true'
                    sh 'mvn -B deploy -DskipTests -Dpmd.skip=true -Dapp.v
                }
            }
        }
    }
}

// Finally, notify the slack group about the build status
post {
success {
    slackSend baseUrl: 'https://app.slack.com/services/hooks/jenkins-ci/
        channel: '#app',
        color: 'good',
        token: 'xxxxxxxxx',
        message: "A new version of the APP is avaiable to production (pip
}
failure {
    slackSend baseUrl: 'https://app.slack.com/services/hooks/jenkins-ci/
        channel: '#app',
        color: 'bad',
        token: 'xxxxxxxxx',
        message: "The app-beta ${currentBuild.fullDisplayName} is broken.
    }
}

}
```

Com as etiquetas :
java,
jenkins,

jira,
maven,
slack,
soapui,
wildfly

Publicado por schmittjoaopedro

Graduado como bacharel em Sistemas de Informação pelo Centro Universitário Católica de Santa Catarina campus Jaraguá do Sul. Formado no Ensino Médio pelo Senai com Técnico em Redes de Computadores Articulado. Atualmente desenvolvedor JEE/Web em Sistemas de Engenharia na WEG. Pesquisador no período de faculdade em Informática pela Católica de Santa Catarina. Contato 47 - 99615 2305 E-mail: schmittjoaopedro@gmail.com Web page: https://joaoschmitt.wordpress.com/ Linkedin: https://www.linkedin.com/in/joao-pedro-schmitt-60847470/ Curriculum lattes: http://lattes.cnpq.br/9304236291664423 Twitter: @JooPedroSchmitt Ver todos os artigos de schmittjoaopedro