Lab: Finished Pipeline

In this exercises you will:

- Review Pipeline resources.
- Create and run a Pipeline in another feature branch.
- Move some steps to `post` sections.
- Add a `when` directive to skip Deploy stages when the pipeline is not running on the master branch.
- Save the Pipeline to the master branch.

Before you begin this lab, ensure that your Pipeline matches the solution from the previous lab: Multi-environment Pipeline. You can use any text editor or the Blue Ocean Visual Editor along with the Blue Ocean Code Editor to make necessary changes.

Task: Review Pipeline resources

The Jenkins Pipeline features in this lab are not accessible via the Blue Ocean Visual Editor. The rest of this lab requires some combination of the Classic Web UI and the Blue Ocean Code Editor.

Snippet Generator

The Snippet Generator helps you construct proper Pipeline code for Scripted and Declarative Pipelines. It is useful for developing Pipeline code both in a text editor and in Blue Ocean when you need options that must be entered in the Blue Ocean Code Editor manually.

Learn more about the Snippet Generator:

1. From the Jenkins Dashboard, select `pipeline-lab`.
2. Select **Pipeline Syntax** in the left navigation bar and open **Snippet Generator**.
3. Select a step from the **Sample Step** dropdown menu, add some value, and select **Generate Pipeline Script** to generate a code snippet.

   For example, select **echo: Print Message**, enter the **Message** `Placeholder`, and select **Generate Pipeline Script**.

4. Review the resulting code snippet. In this example, it is `echo 'Placeholder'`.
5. Repeat this for a few different steps to view the resulting code snippets.

Declarative Directive Generator

The **Declarative Directive Generator** in **Pipeline Syntax** helps you code the syntax for directives correctly. Learn more about the Declarative Directive Generator in the same way as you review the Snippet Generator. Refer to the Declarative Pipeline Syntax for available directives.

Task: Create and run a Pipeline in another feature branch

1. In Blue Ocean, edit your pipeline from the `master` branch and select the edit icon.
2. Save the pipeline. In the **Save** dialog, provide the description `Start Finished Pipeline` and commit to a new branch named `finished-pipeline`.

Task: Move some steps to `post` sections

Edit your pipeline from the `finished-pipeline` branch:

1. Open the Blue Ocean Code Editor with `Ctrl+S` or `Cmd+S`.
2. Add a post section to any stage that contains any of the following steps: `archiveArtifacts`, `stash`, or `junit`.
    a. Move the `archiveArtifacts` and `stash` steps in the Build stages into `post` sections using the `{ success { … } }` condition.

    For example, the Build Java 7 stage looks like the following with a `post` section:

```
stage('Build Java 7') {
  agent {
    node {
      label 'java7'
    }
  }
  steps {
    sh './jenkins/build.sh'
  }
  post {
    success {
      archiveArtifacts 'target/*.jar'
      stash(name: 'Java 7', includes: 'target/**')
    }
  }
}
```

    The Build Java 8 stage looks like:

```
stage('Build Java 8') {
  agent {
    node {
      label 'java8'
    }
  }
  steps {
    sh './jenkins/build.sh'
  }
  post {
    success {
      stash(name: 'Java 8', includes: 'target/**')
    }
  }
}
```

b. Move the `junit` steps in the Backend and Frontend Test stages into `post` sections using the `{ always { … } }` condition.

For example, the Backend Java 7 and Frontend Java 7 stages look like the following with their respective `post` sections:

```
stage('Backend Java 7') {
  agent {
    node {
      label 'java7'
    }
  }
  steps {
    unstash 'Java 7'
    sh './jenkins/test-backend.sh'
  }
  post {
    always {
      junit 'target/surefire-reports/**/TEST*.xml'
    }
  }
}
stage('Frontend Java 7') {
  agent {
    node {
      label 'java7'
    }
  }
  steps {
    unstash 'Java 7'
    sh './jenkins/test-frontend.sh'
  }
  post {
    always {
      junit 'target/test-results/**/TEST*.xml'
    }
  }
}
```

Repeat the same steps to add `post` sections to Backend Java 8 and Frontend Java 8.

3. Select **Update** to apply your changes.

These steps will no longer be visible in the Blue Ocean Visual Editor, but they exist in the `Jenkinsfile`.

4. Save the pipeline. In the **Save** dialog, provide the description `Move steps to post sections` and save to the default branch (`finished-pipeline` branch).

Task: Add a `when` directive

Edit your pipeline from the `finished-pipeline` branch:

1. Open the Blue Ocean Code Editor with `Ctrl+S` or `Cmd+S`.

2. Add a when directive to specify the branch condition at the start of the two Deploy stages. This tells the run to skip the two Deploy stages when the pipeline runs on a branch other than `master`.
    a. Specify the `branch` condition for the Confirm Deploy stage.

```
stage('Confirm Deploy') {
    when {
      branch 'master'
    }
    steps {
      input(message: 'Okay to deploy to staging?', ok: 'Yes')
    }
  }
```

    b. Specify the `branch` condition for the Fluffy Deploy stage.

```
  stage('Fluffy Deploy') {
    when {
      branch 'master'
    }
    agent {
      node {
        label 'java7'
      }
    }
    steps {
      unstash 'Java 7'
      sh './jenkins/deploy.sh staging'
    }
  }
```

3. Select **Update** to apply your changes.

    These directives are not visible in the Blue Ocean Visual Editor, but they exist in the `Jenkinsfile`.

4. Save the pipeline. In the **Save** dialog, provide the description `Skip Deploy when not on master` and save to the default branch (`finished-pipeline` branch).
5. Watch the pipeline run and verify that both Deploy stages were skipped.

Task: Save a Pipeline to the master branch

Edit your pipeline from the `finished-pipeline` branch:

1. Save the pipeline. In the **Save** dialog, provide the description `Change to Finshed Pipeline in master` and save to the `master` branch.
2. Watch the pipeline run and verify that both Deploy stages were included in the run.

Solution

Click here to see the solution
Jenkinsfile (final)

```
pipeline {
  agent none
  stages {
    stage('Fluffy Build') {
      parallel {
        stage('Build Java 8') {
          agent {
            node {
              label 'java8'
            }
          }
          steps {
            sh './jenkins/build.sh'
          }
          post {
            success {
              stash(name: 'Java 8', includes: 'target/**')
            }
          }
        }
        stage('Build Java 7') {
          agent {
            node {
              label 'java7'
            }
          }
          steps {
            sh './jenkins/build.sh'
          }
          post {
            success {
              archiveArtifacts 'target/*.jar'
              stash(name: 'Java 7', includes: 'target/**')
            }
          }
        }
      }
    }
    stage('Fluffy Test') {
      parallel {
        stage('Backend Java 8') {
          agent {
            node {
              label 'java8'
            }
          }
          steps {
            unstash 'Java 8'
            sh './jenkins/test-backend.sh'
          }
          post {
            always {
              junit 'target/surefire-reports/**/TEST*.xml'
            }
          }
        }
```

```
        stage('Frontend Java 8') {
          agent {
            node {
              label 'java8'
            }
          }
          steps {
            unstash 'Java 8'
            sh './jenkins/test-frontend.sh'
          }
          post {
            always {
              junit 'target/test-results/**/TEST*.xml'
            }
          }
        }
        stage('Performance Java 8') {
          agent {
            node {
              label 'java8'
            }
          }
          steps {
            unstash 'Java 8'
            sh './jenkins/test-performance.sh'
          }
        }
        stage('Static Java 8') {
          agent {
            node {
              label 'java8'
            }
          }
          steps {
            unstash 'Java 8'
            sh './jenkins/test-static.sh'
          }
        }
        stage('Backend Java 7') {
          agent {
            node {
              label 'java7'
            }
          }
          steps {
            unstash 'Java 7'
            sh './jenkins/test-backend.sh'
          }
          post {
            always {
              junit 'target/surefire-reports/**/TEST*.xml'
            }
          }
        }
        stage('Frontend Java 7') {
          agent {
            node {
              label 'java7'
            }
          }
          steps {
```

```
            unstash 'Java 7'
            sh './jenkins/test-frontend.sh'
          }
          post {
            always {
              junit 'target/test-results/**/TEST*.xml'
            }
          }
        }
        stage('Performance Java 7') {
          agent {
            node {
              label 'java7'
            }
          }
          steps {
            unstash 'Java 7'
            sh './jenkins/test-performance.sh'
          }
        }
        stage('Static Java 7') {
          agent {
            node {
              label 'java7'
            }
          }
          steps {
            unstash 'Java 7'
            sh './jenkins/test-static.sh'
          }
        }
      }
    }
    stage('Confirm Deploy') {
      when {
        branch 'master'
      }
      steps {
        input(message: 'Okay to deploy to staging?', ok: 'Yes')
      }
    }
    stage('Fluffy Deploy') {
      when {
        branch 'master'
      }
      agent {
        node {
          label 'java7'
        }
      }
      steps {
        unstash 'Java 7'
        sh './jenkins/deploy.sh staging'
      }
    }
  }
}
```