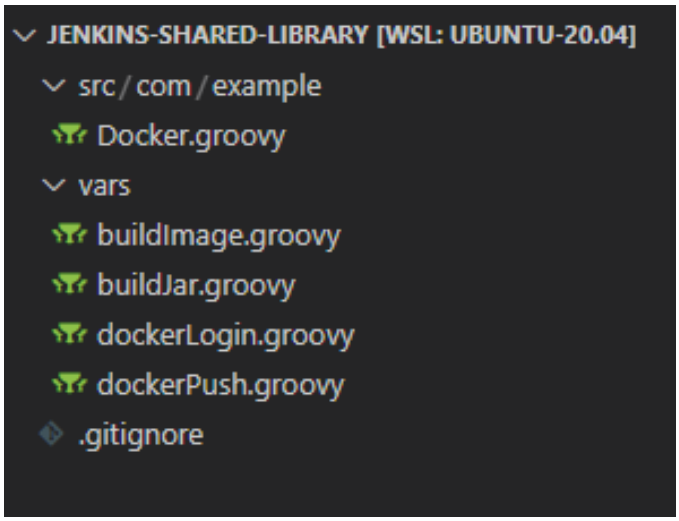
# Demo Project: Create a Jenkins Shared Library



## Step 1: Create the Git Repository for the Shared Library

1. **Create a new Git repository:**
   - Name the repository, e.g., `jenkins-shared-library`.
2. **Define the project structure:**
   Organize the repository with the following folder structure:

```
jenkins-shared-library/
|-- vars/
|   |-- buildJar.groovy
|   |-- buildImage.groovy
|   |-- dockerLogin.groovy
|   |-- dockerPush.groovy
|-- src/
    |-- com/
        |-- example/
            |-- Docker.groovy
```




3. **Initialize the repository:**
   - Clone the repository to your local machine.
   - Create the necessary directories (`vars`, `src/com/example/`).
4. **Add shared library files:**
   - Create Groovy files in the `vars` folder for reusable pipeline steps.
   - Create a Groovy class (e.g., `Docker.groovy`) under `src/com/example/` to define common methods.
5. **Commit and push changes**

---

## Step 2: Configure Jenkins to Use the Shared Library

1. **Access Jenkins Global Configuration:**
   - Log in to Jenkins.
   - Navigate to **Manage Jenkins > System**.
2. **Add the Shared Library:**
   - Scroll to the **Global Trusted Pipeline Libraries** section.
   - Click **Add**.
   - Enter the following details:
     - **Name:** `jenkins-shared-library` (must match the repository name).

- **Default version:** `main` (or the branch name you will use).
- **Retrieval method:** Choose **Modern SCM**.
- **Source Code Management:** Select **Git** and provide the repository URL.

3. **Save the configuration.**

---

# Step 3: Create Functions in the Jenkins Shared Library

1. **Example Function in `vars/buildJar.groovy`:**

```
#!/user/bin/env groovy

def call () {
    echo "building the application for branch $BRANCH_NAME"
    sh 'mvn package'
}
```

2. **Example Docker Functions in `src/com/example/Docker.groovy`:**

```
#!/user/bin/env groovy
package com.example

class Docker implements Serializable {

    def script

    Docker(script) {
        this.script = script
    }

    def buildDockerImage(String imageName) {
        script.echo "building the docker image..."
        script.sh "docker build -t $imageName ."
        }

    def dockerLogin() {
        script.withCredentials([script.usernamePassword(credentialsId: 'docker-hub-repo',
passwordVariable: 'PASS', usernameVariable: 'USER')]) {
            script.sh "echo '${script.PASS}' | docker login -u '${script.USER}' --password-stdin"
        }
    }

    def dockerPush(String imageName) {
        script.sh "docker push $imageName"
    }
}
```

3. **Commit and push the changes**

---

# Step 4: Use the Shared Library in a Jenkins Pipeline

## Globally in Jenkins

1. **Access Jenkins Pipeline Configuration:**
   - Navigate to the project that needs the Shared Library in Jenkins Pipeline.
   - Click **Configure**.
2. **Update the pipeline to use the library:**

```
#!/user/bin/env groovy

@Library('jenkins-shared-library')

def gv

pipeline {
    agent any
    tools {
        maven 'maven-3.9'
    }
    stages {
        stage("init") {
            steps {
                script {
                    gv = load "script.groovy"
                }
            }
        }
        stage("build jar") {
            steps {
                script {
                    buildJar ()
                }
            }
        }
        stage("build and push image") {
            steps {
                script {
                    buildImage ('your-image-name')
                    dockerLogin()
                    dockerPush ('your-image-name')
                }
            }
        }
        stage("deploy") {
            steps {
                script {
                    gv.deployApp()
                }
            }
        }
    }
}
```

3. **Save and run the pipeline.**

Stage View

| | Declarative: Checkout SCM | Declarative: Tool Install | init | build jar | build and push image | deploy |
|---|---|---|---|---|---|---|
| Average stage times: (Average full run time: ~13s) | 617ms | 106ms | 361ms | 6s | 2s | 177ms |
| #12 Dec 25 15:42 1 commit | 637ms | 108ms | 352ms | 6s | 2s | 231ms |

eduardobautistamaciel/demo-app

Last pushed less than a minute ago

*Add a description*

*Add a category*

Tags

This repository contains 3 tag(s).

| Tag | OS | Type | Pulled | Pushed |
|---|---|---|---|---|
| jma-3.0 | | Image | --- | a few seconds ago |

## For a Specific Project in Jenkins

1. **Update the pipeline for the specific project:**

```
#!/user/bin/env groovy

library identifier: 'jenkins-shared-library@main', retriever: modernSCM([
    $class: 'GitSCMSource',
    remote: 'https://gitlab.com/twn-devops-projects/jenkins/jenkins-shared-library',
    credentialsId: 'gitlab-credentiales'
])

def gv

pipeline {
    agent any
    tools {
        maven 'maven-3.9'
    }
    stages {
        stage("init") {
            steps {
                script {
                    gv = load "script.groovy"
                }
            }
        }
        stage("build jar") {
            steps {
                script {
                    buildJar ()
                }
            }
        }
        stage("build and push image") {
            steps {
                script {
```

2. Navigate to **Manage Jenkins > System:**
    - Scroll to the **Global Trusted Pipeline Libraries** and remove the section.
2. **Run the pipeline for this project.**

Stage View

| | Declarative: Checkout SCM | Declarative: Tool Install | init | build jar | build and push image | deploy |
|---|---|---|---|---|---|---|
| Average stage times: (Average full run time: ~15s) | 661ms | 80ms | 315ms | 6s | 5s | 237ms |
| #5 Dec 25 16:14 No Changes | 661ms | 80ms | 315ms | 6s | 5s | 237ms |

## eduardobautistamaciel/demo-app

Last pushed 1 minute ago

*Add a description*

*Add a category*

### Tags

This repository contains 3 tag(s).

| Tag | OS | Type | Pulled | Pushed |
|---|---|---|---|---|
| jma-3.0 | | Image | — | a few seconds ago |