Creating a Jenkins pipeline to build and deploy a Docker container involves several steps. In this write-up, we'll outline a basic Jenkins pipeline that builds a Docker image and pushes it to a container registry. For demonstration purposes, we'll use Docker Hub as the container registry. Make sure you have Docker and Jenkins properly set up before proceeding.

Step 1: Configure Jenkins Global Credentials

Before creating the pipeline, we need to add Docker Hub credentials to Jenkins. This will allow Jenkins to push the Docker image to Docker Hub. Follow these steps:

1. Go to Jenkins Dashboard.

2. Click on "Manage Jenkins" > "Manage Credentials".

3. Under "Stores scoped to Jenkins", click on "Jenkins".

4. Click on "Global credentials (unrestricted)" > "Add Credentials".

5. Select "Username with password" as the Kind.

6. Provide your Docker Hub username and password.

7. Give the credentials an ID (e.g., "docker-hub-credentials") and a meaningful description.

8. Click "OK" to save the credentials.

Step 2: Create Jenkins Pipeline Script

Now, let's create the Jenkins pipeline script:

Sample code:

```
pipeline {
  agent any
  environment {
    DOCKER_HUB_CREDENTIALS = 'docker-hub-credentials' // Replace with your Docker Hub
credentials ID
    DOCKER_IMAGE_NAME = 'your-docker-hub-username/your-docker-image-name'
  }
  stages {
    stage('Checkout') {
      steps {
        git 'https://github.com/your-repo.git' // Replace with the URL of your source code
repository
      }
    }
```

```
        stage('Build Docker Image') {
            steps {
                script {
                    docker.withRegistry('', DOCKER_HUB_CREDENTIALS) {
                        def customImage = docker.build(DOCKER_IMAGE_NAME)
                    }
                }
            }
        }
        stage('Push Docker Image') {
            steps {
                script {
                    docker.withRegistry('https://index.docker.io/v1/', DOCKER_HUB_CREDENTIALS) {
                        docker.image(DOCKER_IMAGE_NAME).push()
                    }
                }
            }
        }
    }

    post {
        always {
            cleanWs()
        }
    }
}
```

Replace the placeholders `your-docker-hub-username` and `your-docker-image-name` with your actual Docker Hub username and the desired name for your Docker image.

Step 3: Save and Run the Pipeline

1. Open Jenkins and navigate to the Jenkins Dashboard.

2. Click on "New Item" to create a new pipeline.

3. Enter a name for your pipeline (e.g., "Docker_Build_Deploy_Pipeline").

4. Select "Pipeline" as the type and click "OK".

5. Under the "Pipeline" section, select "Pipeline script" and paste the pipeline script from Step 2.

6. Click "Save" to create the pipeline.

Now, Jenkins is ready to execute the pipeline. When you run the pipeline, it will clone the source code from your specified repository, build the Docker image, and push it to Docker Hub.

Please note that this is a basic example of a Jenkins pipeline for Docker, and you may need to modify it based on your specific use case and requirements. For more advanced scenarios, you might want to include additional stages, tests, or environment configurations in your pipeline.