A Guide to Navigation Jenkins Pipelines: Reading Code Without Groovy Know-How

Software delivery procedures can be efficiently automated and coordinated with the help of Jenkins pipelines. However, if you're unfamiliar with Groovy, the scripting language used to define pipeline stages and steps, interacting with Jenkins pipelines might be challenging. This post will discuss Jenkins pipelines in a way that is easy to understand for people who have never used Groovy before.

Understanding the Basics:

First, start by familiarizing yourself with the basic concepts of Jenkins pipelines. A pipeline is essentially a set of instructions that define the steps to be executed during the software delivery process. It includes stages, steps, and other elements that help streamline the workflow.

Explore Declarative Syntax:

Jenkins supports both Scripted and Declarative pipeline syntax. Declarative pipelines are recommended for beginners as they offer a more structured and readable way to define pipelines and come with a set of built-in steps and directives that cover common use cases, such as checking out code, running tests, and deploying artifacts. This means you can accomplish many tasks without having to write custom Groovy code, making it more approachable for those who are not yet proficient in Groovy.

For example:

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                // Your build steps here
            }
        }
        stage('Test') {
            steps {
                // Your test steps here
            }
        }
        stage('Deploy') {
            steps {
                // Your deployment steps here
            }
        }
    }
    post {
        success {
            echo 'Pipeline succeeded! Do additional post-success tasks here.'
        }
        failure {
            echo 'Pipeline failed! Handle post-failure tasks here.'
        }
    }
}
```

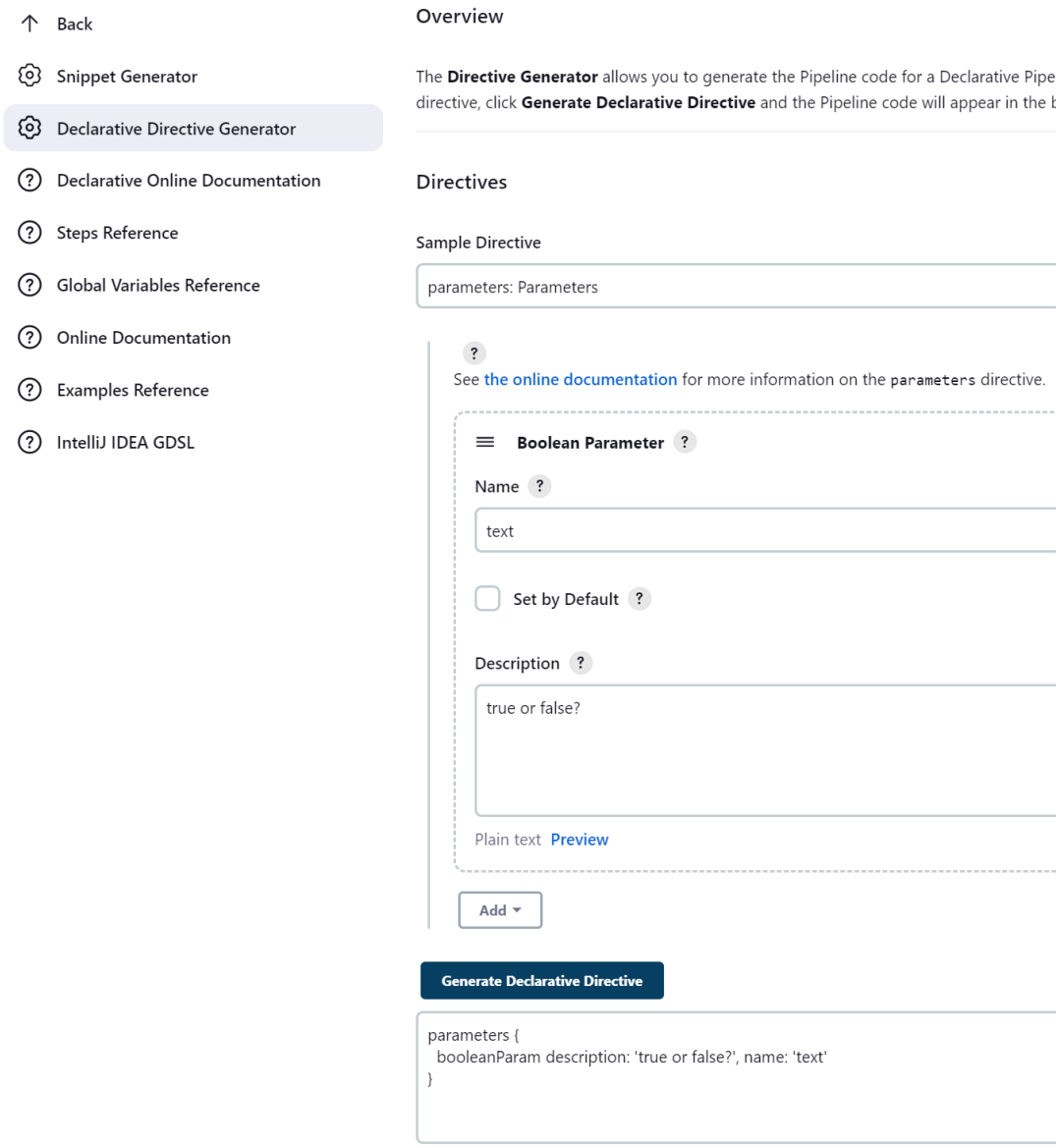
Incremental Learning:

Approach Groovy learning incrementally. Begin by focusing on understanding the simpler constructs within the code, such as method calls and variable assignments. Gradually delve into more complex aspects as you become more comfortable with the syntax.

Useful Tools for Code Comprehension:

Pipeline Syntax:

Jenkins provides a Pipeline Syntax tool that aids in generating Groovy code snippets for different pipeline steps. In the Pipeline Syntax tool, you'll find a dropdown menu that lists different pipeline steps, Choose the step that corresponds to the action you want to perform in your pipeline and generate the corresponding Groovy code snippet for the selected step. This tool can be found on the left pane of any pipeline job.



Script Console:

The Script Console allows you to interactively execute Groovy scripts directly within the Jenkins environment. It's an excellent way to experiment with Groovy code snippets without the need to modify Jenkinsfiles or pipelines. This tool can be found under Manage Jenkins and scroll down until you see the Script Console option.