Declarative vs. scripted pipelines: What's the difference?

- Jenkins provides two different syntaxes for pipelines. When DevOps engineers write a Jenkins pipeline, they can choose between declarative and scripted. The differences between the two are both subtle and significant.

- A Jenkins pipeline should be **a simple, easy-to-read and easy-to-manage component**. Excessive code in a scripted pipeline violates one of the fundamental CI/CD principles.

Declarative vs. scripted pipelines

- In contrast to the scripted pipeline, the declarative Jenkins pipeline doesn't permit a developer to inject code.

- Simply put, never put complex code into a pipeline, regardless of whether it seems possible. Jenkins provides several ways to make complex logic available to both scripted and declarative pipelines.

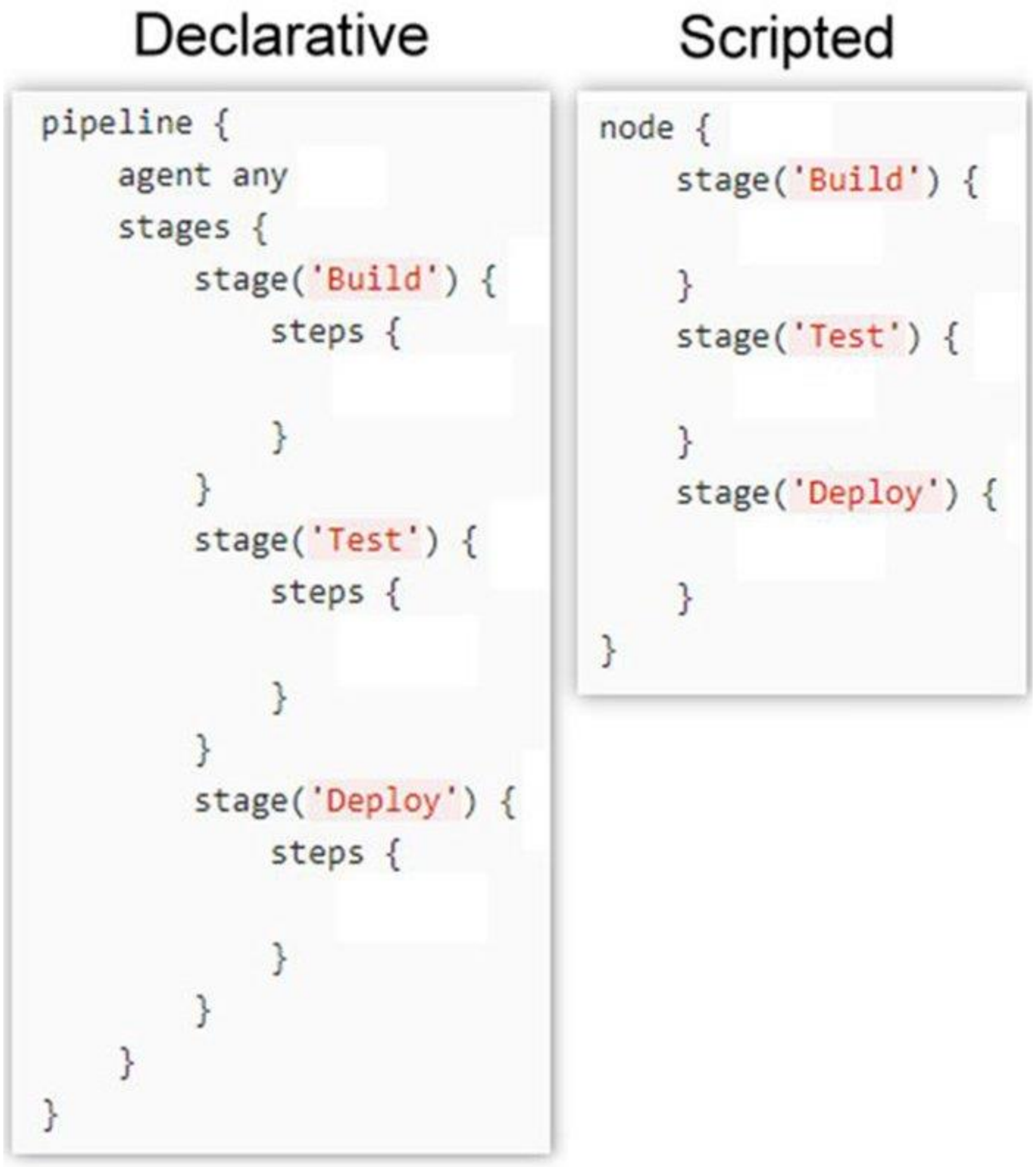
- Scripted vs. declarative pipelines are different only in their programmatic approach. One uses a declarative programming model, while the other uses an imperative programming model.

- But they both run on the same Jenkins pipeline sub-system. There are no differences in the runtime performance, scalability, and problem solvability perspective in the declarative vs. scripted pipeline debate. They only differ in the syntactic approach used to achieve an end goal.

Pipeline syntax differences

- Declarative pipelines always begin with the word pipeline. Scripted pipelines, on the other hand, always begin with the word node. Declarative pipelines break down stages into individual stages that can contain multiple steps.

❖ These are the key differences that allow a DEVOPS ENGINEER to quickly differentiate between a scripted pipeline and a declarative pipeline.



Declarative

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {

            }
        }
        stage('Test') {
            steps {

            }
        }
        stage('Deploy') {
            steps {

            }
        }
    }
}
```

Scripted

```
node {
    stage('Build') {

    }
    stage('Test') {

    }
    stage('Deploy') {

    }
}
```

❖ Declarative Jenkins pipeline is longer compared to the scripted Jenkins pipeline. This is caused by the number of stages.

- Now what about the choice between declarative vs. scripted pipelines for a new project? The answer to that question is most definitely the declarative pipeline.

- The development industry has largely moved toward a declarative programming model for CI/CD pipelines. Both GitHub Actions and GitLab CI support only YAML pipelines, which are very similar to declarative Jenkins pipelines.

- Furthermore, declarative pipelines are easier to maintain, and they tend to have a lower learning curve. The declarative syntax is the best approach to use when new CI/CD workflows are built.