29.04May.Adv Jenkins

4 May 2023 07:09 PM

CT
Types
- Integration
- Unit
- Acceptance
- Load Testing

Test Driven Development
BDD
Selenium

CI
Dev1 --> Github --> Build --> Test --> Report(Surefire)

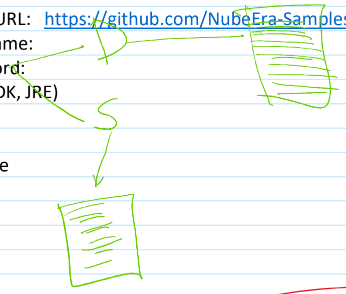
JaCoCo Report
- Code Quality(java file, classes)
- HTML, CSV, XML

Surefire Test Cases Report
- XML
- txt

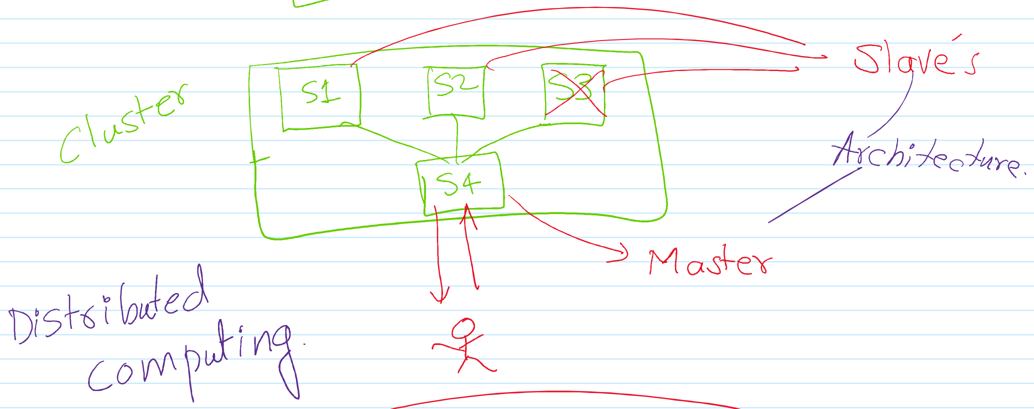
Job
Practical 1: CI Server(Jenkins) Clone Github Repo(private/public)
  a. Public Repo
  b. Private Repo
    a. Manage Jenkins --> Security --> Credentails --> System --> Add Cred.
    b. Using Github (PAT)

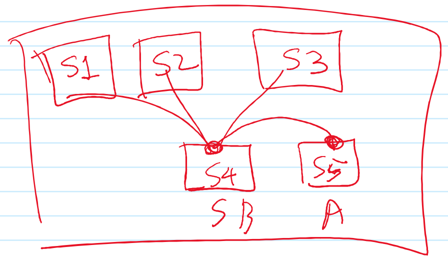
Practical 2: CI Server(Jenkins) Clone, **Configure** Build, Build Test --> Generate Report

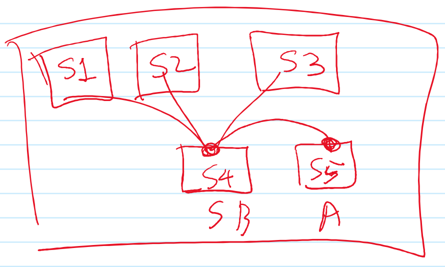
Practical 1: CI Server(Jenkins) Clone Github Repo(private/public)
1. Github
   a. Repo. URL: https://github.com/NubeEra-Samples/Java-Hello-Welcome-World.git
   b. UserName:
   c. Password:
2. Install Java(JDK, JRE)

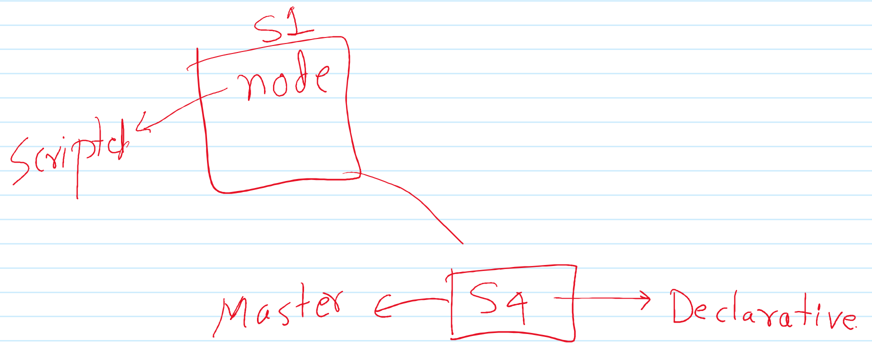


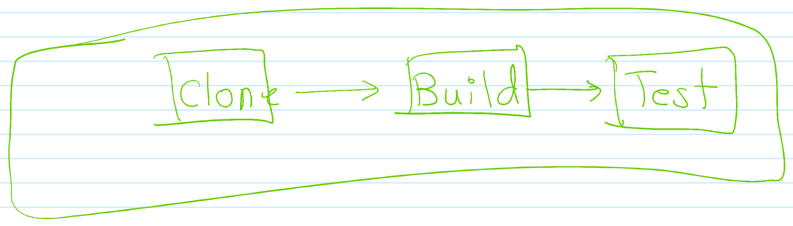
Freestyle vs Pipeline



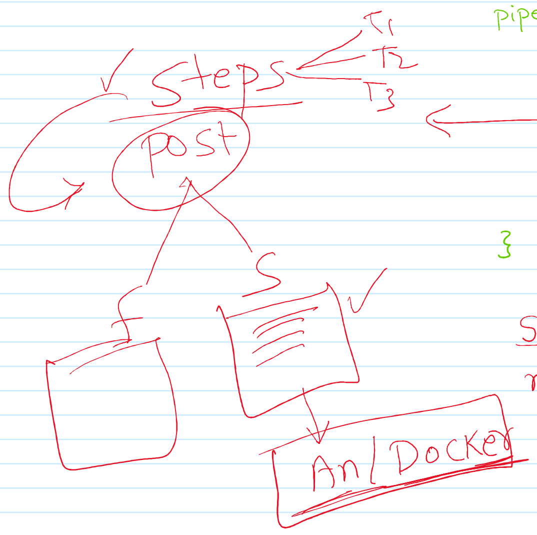








Declarative



```
pipeline {
    agent  any
    stages {
        Clone
        Build
        test
    }
}
```

tools { Maven M3 → GTC
Java J2
Ansible A5 }

Scripted

```
node {
    Stage("Clone") {
    }
    Stage ("Build") {
    }
}
```

Declarative

pipeline = whole pipeline

agent = defines the machine that will handle this pipeline

~~pipeline~~ = whole pipeline.

agent = defines the machine that will handle this pipeline.

Stages = declares the Stages of the pipeline.

Steps = small operations inside a particular stage.

Pre

post = when Steps executed < S / F > post

Jenkins → Create User Jenkins

win

bat

Lin/OSX

sh

```
pipeline {
   agent any
   stages {
      stage("Greeting") {
         steps {
            sh 'echo Welcome to Jenkins Pipeline'
         }
      }
      stage("Bye") {
         steps {
            sh 'echo Thanks to Jenkins Pipeline'
         }
      }
   }
}
```

Mar/2023

pwd

PAT

<u>Scripted vs Declarative Pipeline</u>

Syntax:
S: based on Groovy Scripting
D: YAML

Flexibility:
S: More flexibility and control over the pipeline workflow
D: Simpler & more structured syntax

Error Handling:
S: Allow for more granular error, recovery mechanisms
D: Simpler error, easier to understand

Code Reuse:
Readability: D:

Jenkins
- All Temporary Execution with folder is available in workspace (PipelineJobName,PipelineJobName@tmp)

Linux --> mvn --> Java(JRE --> JVM--> .class DVM = Dalvik Virtual Machine --> apk/jar/ear/war)

-Dmaven.test.failure.ignore=true

```
pipeline {
   agent any
   tools {
      maven "M3"
   }

   stages {
      stage('Check JRE') {
         steps {
            sh "java -version"
         }
      }
      stage('Check JDK') {
         steps {
            sh "javac -version"
         }
      }
      stage('Check MVN') {
         steps {
            sh "mvn --version"
         }
      }
      stage('Check Git') {
         steps {
            sh "git --version"
         }
      }
      stage('Cloning') {
         steps {
            git "https://github.com/NubeEra-Samples/JavaMvnJUnit.git"
         }
      }
      stage('Building and Test') {
         steps {
            sh "mvn clean"
            sh "mvn -Dmaven.test.failure.ignore=true package"
            //bat "mvn -Dmaven.test.failure.ignore=true clean package"
         }
         post {
            success {
               junit '**/target/surefire-reports/TEST-*.xml'
               archiveArtifacts 'target/*.jar'
            }
         }
      }

   }
}
```

M3
Path of MAVE_HOME

```
pipeline {
   agent any
   tools {
      maven "M3"
   }

   stages {
      stage('Check JRE,JDK, MVN, GIT') {
         steps {
            sh "java -version"
            sh "javac -version"
            sh "mvn --version"
            sh "git --version"
         }
      }
      stage('Cloning') {
         steps {
            git "https://github.com/NubeEra-Samples/JavaMvnJUnit.git"
         }
      }
      stage('Building and Test') {
         steps {
            sh "mvn clean"
            sh "mvn -Dmaven.test.failure.ignore=true package"
            //bat "mvn -Dmaven.test.failure.ignore=true clean package"
         }
         post {
            success {
               junit '**/target/surefire-reports/TEST-*.xml'
               archiveArtifacts 'target/*.jar'
            }
         }
      }
   }
}
```

Private Repo

1. Generate Token from Github

Top Right Icon --> Settings --> Developer Settings -->PAT --> Create new Classic PAT --> Name, Permission

PAT: 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

2. Private Repo:
   https://oauth:PAT@github.com/User-OrgName/RepoName.git

https://oauth:PAT@github.com/NubeEra-Samples/JavaMvnJUnit.git