Jenkins를 이용한 CI/CD Pipeline 구축

Jenkins

Microservices

Spring Cloud

프로필

Dowon Lee

지식공유자 인증

5452 ☆ 4.8(420)

멘토링 활성

- **홈**
- 강의
- 로드맵
- 수강후기
- 블로그

강의 (3)

최신순

Spring Cloud로 개발하는 마이크로서비스 애플리케이션(MSA)

Dowon Lee

★★★★★(174)

학습중

+3000명 독점 할인중

Spring Boot를 이용한 RESTful Web Services 개발

Dowon Lee

★★★★★(308)

학습중

+2700명 독점 할인중

웹 애플리케이션 개발을 위한 IntelliJ IDEA 설정

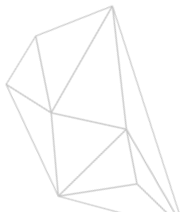
Dowon Lee

★★★★★(228)

학습중

+3500명

목차



Section 5.

Advanced Jenkins

- Delivery Pipeline 사용
- Jenkins Pipeline 구성
- SonarQube 사용
- Jenkins + SonarQube 연동
- Jenkins Master + Slaves 구성

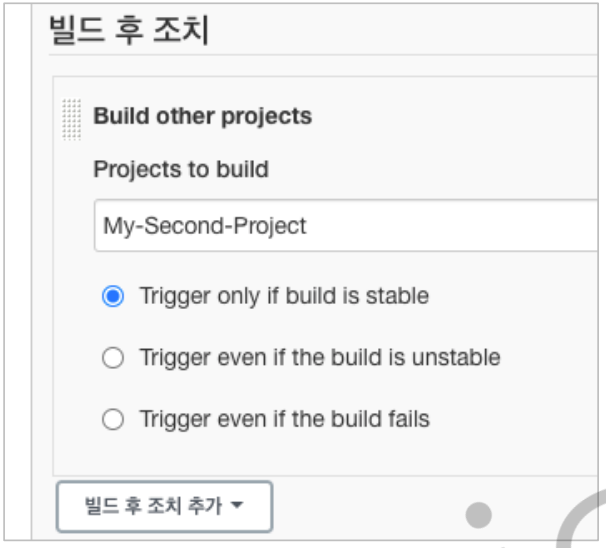
Create a Pipeline

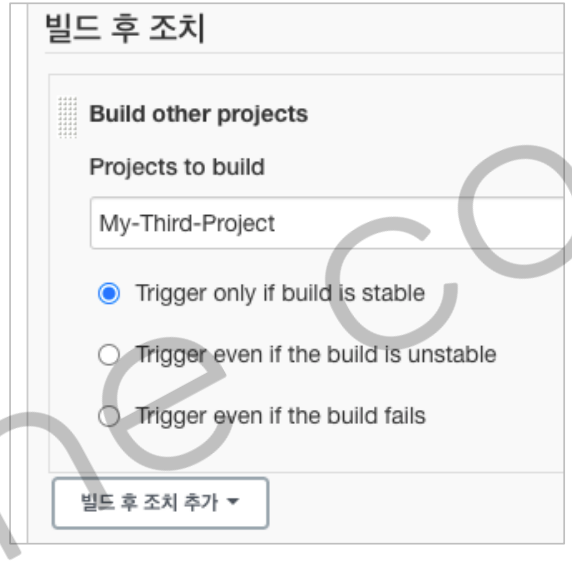
My-First-Project → My-Second-Project → My-Third-Project

- echo

- git
- maven build

- git
- maven build
- deploy on Tomcat

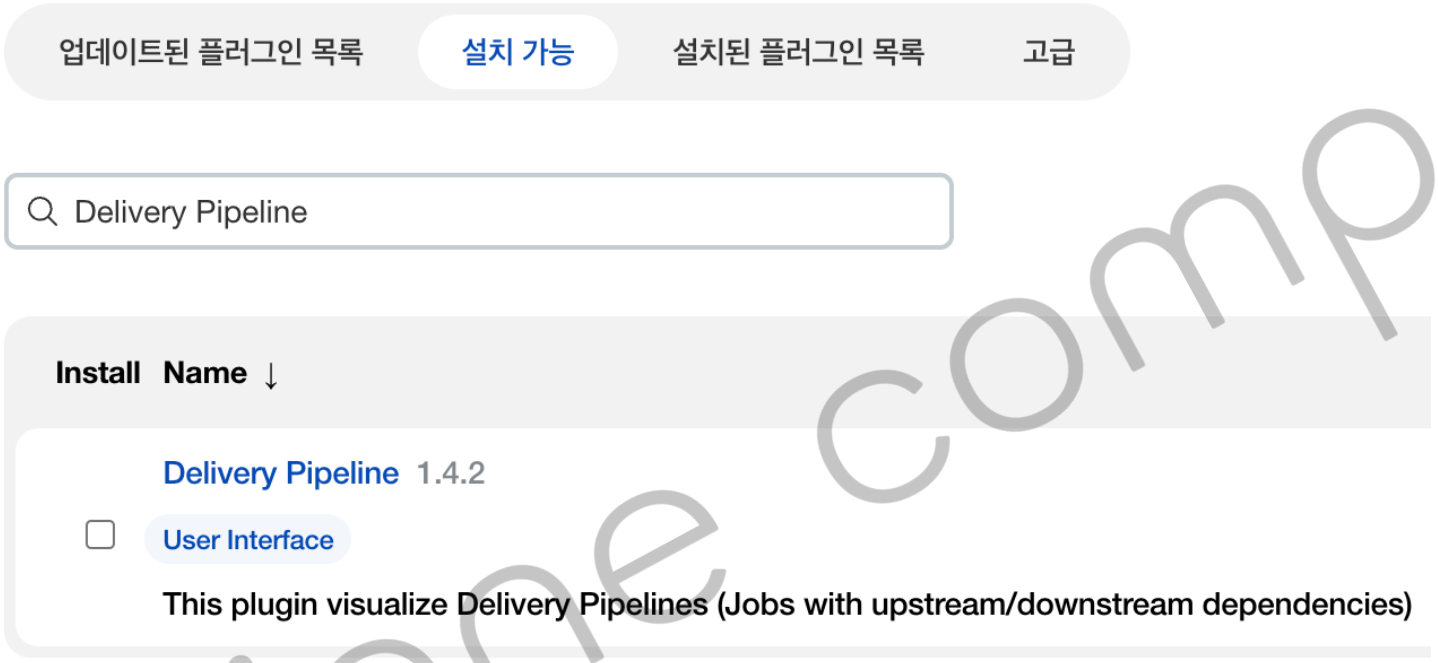




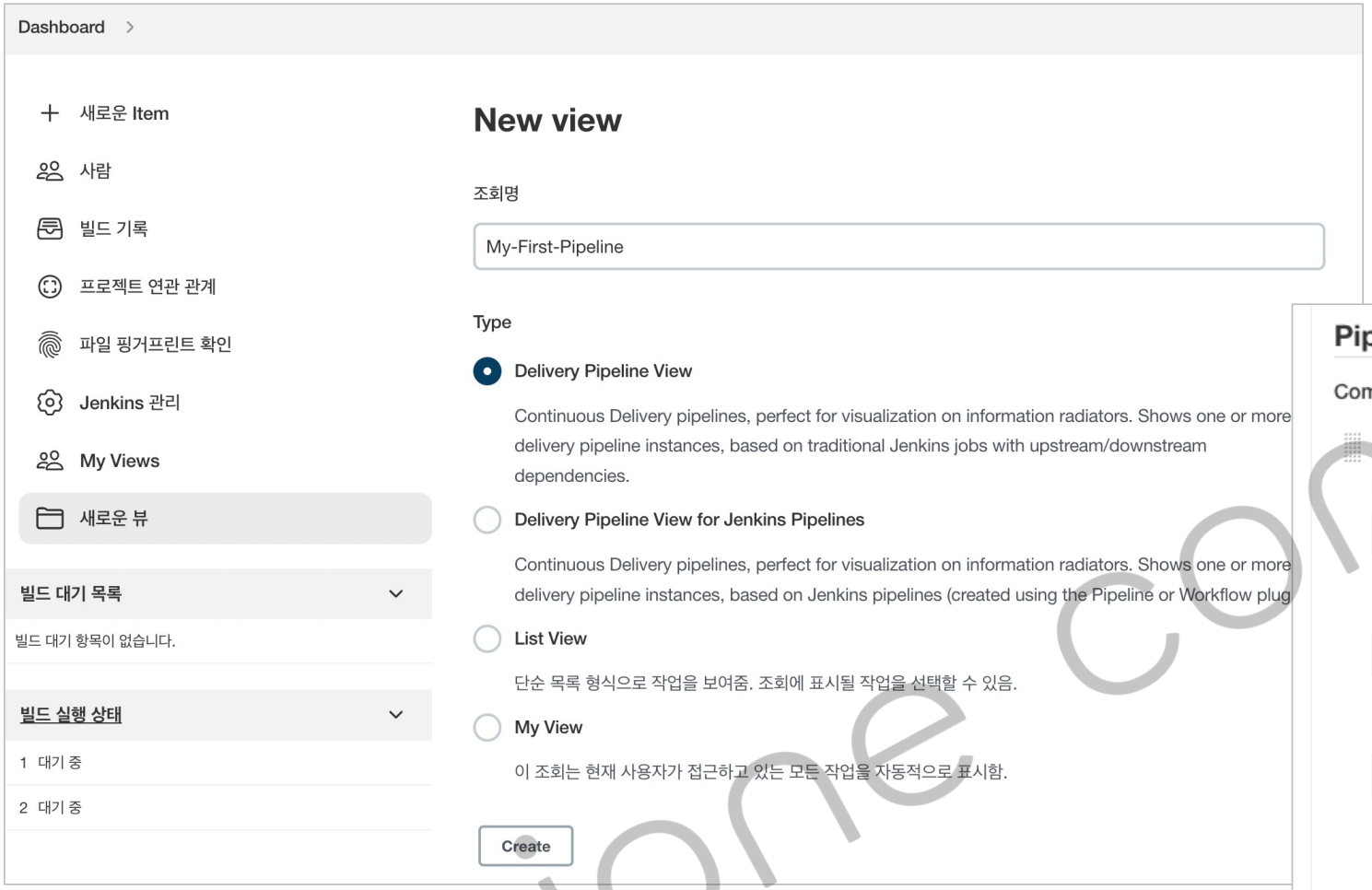
Create a Pipeline

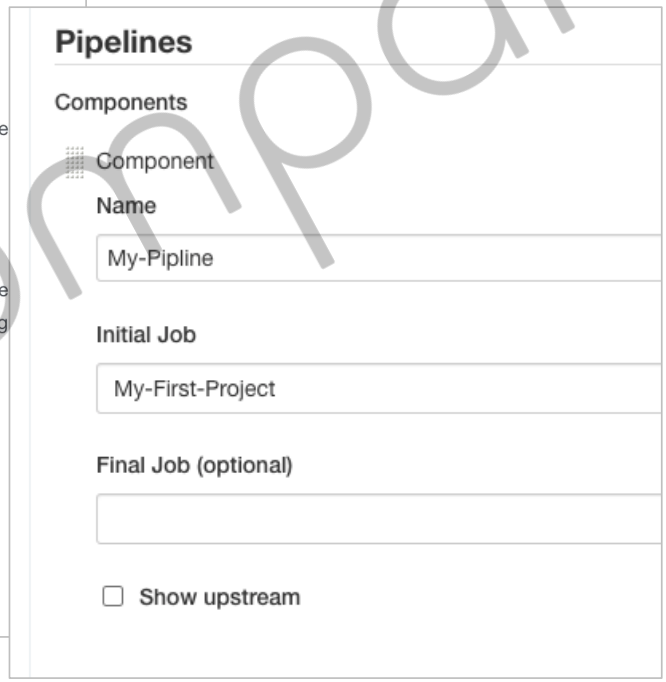
- Manage Jenkins → Plugin Manager→ Available

  - Delivery Pipeline

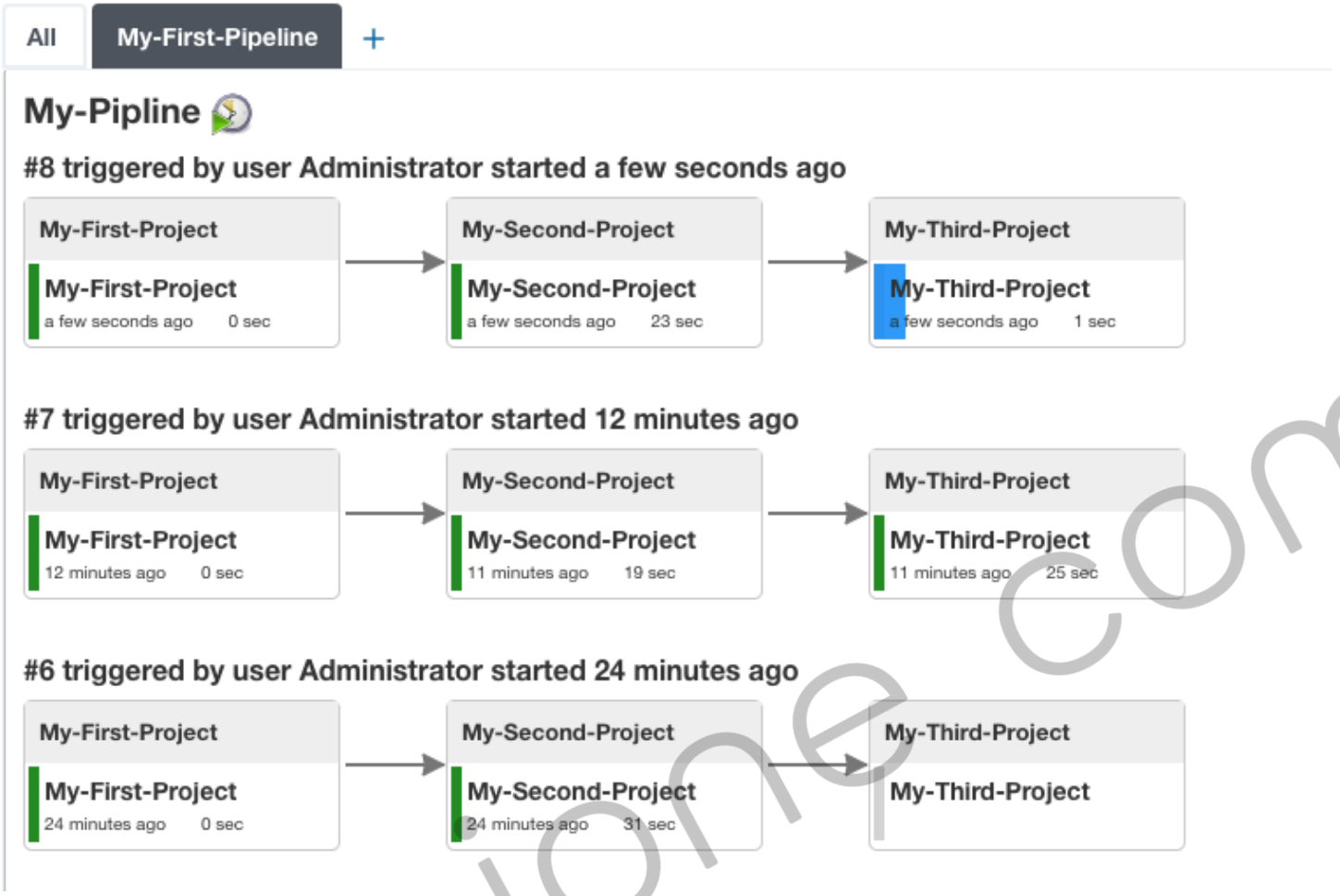


Create a Pipeline





Create a Pipeline



Jenkins Pipeline

- Declarative
- Scripted (Groovy + DSL)
- 차이점
  - 시작 시 유효성 검사 유무
  - 특정 Stage 실행 가능 여부
  - 제어문
  - Option

```
pipeline {
    agent any
    stages {
        steps('build') {
            //
        }

        steps('test') {
            //
        }

        steps('deploy') {
            //
        }
    }
}
```

```
node {
    stage('build') {
        //
    }

    stage('test') {
        //
    }

    stage('deploy') {
        //
    }
}
```

Jenkinsfile

Jenkins Pipeline

- Declarative
  - Groovy script 없이 간단하게 시작

```
pipeline {
    agent any 1)
    stages {
        steps('build') { 2)
            // 3)
        }

        steps('test') { 4)
            // 5)
        }

        steps('deploy') { 6)
            // 7)
        }
    }
}
```

1) → 실행가능한 Agent에서 Pipeline 실행

2) → build 스테이지 선언

3) → build 스테이지에 필요한 작업을 수행

4) → test 스테이지 선언

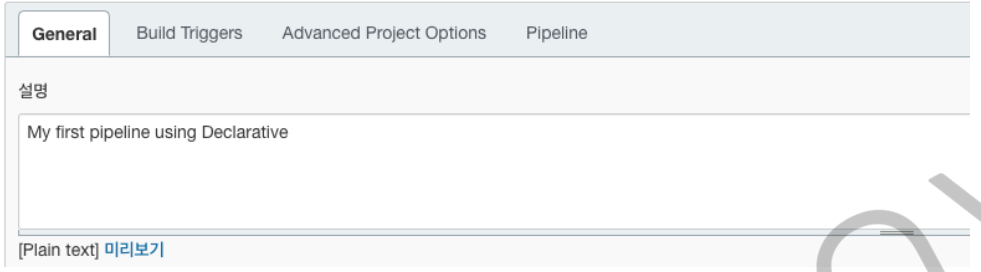
5) → test 스테이지에 필요한 작업을 수행

6) → deploy 스테이지 선언

7) → deploy 스테이지에 필요한 작업을 수행

Exercise 9# Jenkins Pipeline Job 1/4

- Item name → ***My-First-Pipeline***



- Pipeline
  - Script

Exercise 9# Jenkins Pipeline Job 2/4

- Build Now

Exercise 9# Jenkins Pipeline Job 3/4

- Console Output

콘솔 출력

```
Started by user Administrator
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/My-First-Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Compile)
[Pipeline] echo
Compiled successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (JUint)
[Pipeline] echo
```

```
Deployed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Exercise 9# Jenkins Pipeline Job 4/4

- Script 추가
  - ***post***

Pipeline

Definition

Pipeline script

Script

```
22      stage('Deploy') {
23          steps {
24              echo "Deployed successfully!";
25          }
26      }
27  }
28
29  post {
30      always {
31          echo 'This will always run'
32      }
33      success {
34          echo 'This will run when the run finished successfully'
35      }
36      failure {
37          echo 'This will run if failed'
38      }
39      unstable {
40          echo 'This will run when the run was marked as unstable'
41      }
42      changed {
43          echo 'This will run when the state of the pipeline has changed'
44      }
45  }
46  }
```

```
Deployed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
This will always run
[Pipeline] echo
This will run when the run finished successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Exercise 10# Jenkins Pipeline Job 1/3

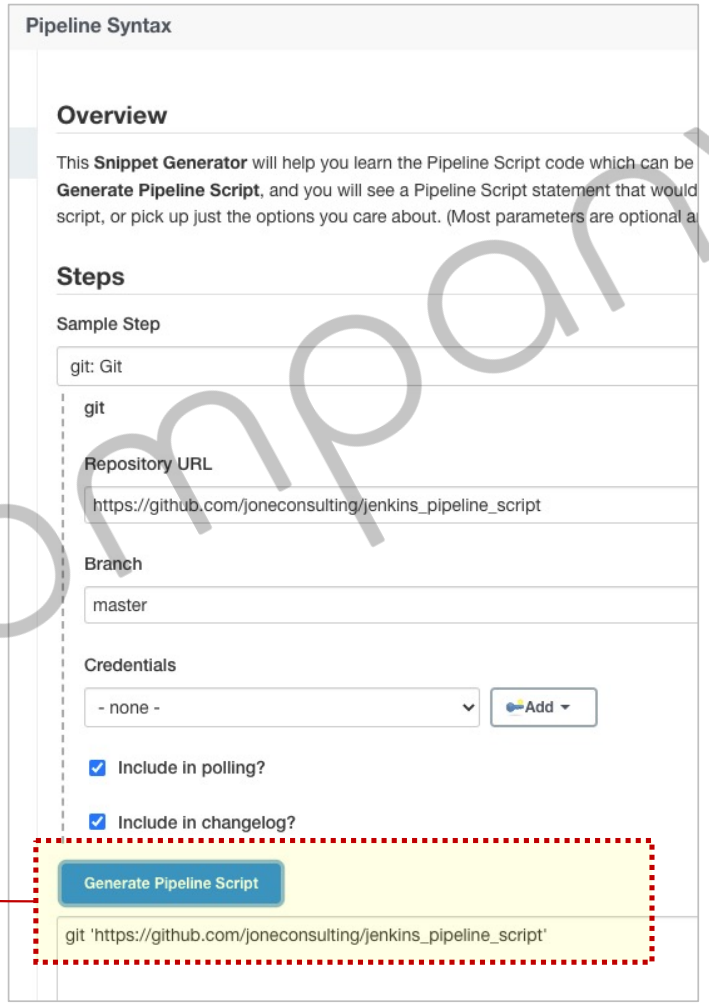
- Github에 저장된 Script 실행
  - https://github.com/joneconsulting/jenkins_pipeline_script

```
ls *.bat
build.bat   deploy.bat  quality.bat unit.bat
(base)  dowonlee  ~/Desktop/git/jenkins_pipeline_script  master
ls *.sh
build.sh   deploy.sh  quality.sh unit.sh
(base)  dowonlee  ~/Desktop/git/jenkins_pipeline_script  master
```

- Pipeline 추가
  - ***Pipeline Syntax*** 이용

Script

```
pipeline {
    agent any
    stages {
        stage('Git clone') {
            steps {
                git 'https://github.com/joneconsulting/jenkins_pipeline_script';
            }
        }
```



Exercise 10 # Jenkins Pipeline Job 2/3

- Build Now


```
Cloning the remote Git repository
Cloning repository https://github.com/joneconsulting/jenkins_pipeline_script
 > git init /var/jenkins_home/workspace/My-First-Pipeline # timeout=10
Fetching upstream changes from https://github.com/joneconsulting/jenkins_pipeline_script
 > git --version # timeout=10
 > git --version # 'git version 2.30.2'
 > git fetch --tags --force --progress -- https://github.com/joneconsulting/jenkins_pipeline_script
timeout=10
 > git config remote.origin.url https://github.com/joneconsulting/jenkins_pipeline_script # timeout=
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 5552317c4dbf21a896b27e707dcbe85168273377 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 5552317c4dbf21a896b27e707dcbe85168273377 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git checkout -b master 5552317c4dbf21a896b27e707dcbe85168273377 # timeout=10
Commit message: "upload batch files"
First time build. Skipping changelog.
```

Exercise 10 # Jenkins Pipeline Job 3/3

- Pipeline 수정 → Build Now

```
9       stage('Compile') {
10          steps {
11              echo "Compiled successfully!";
12              sh './build.sh'
13          }
14      }
15
16      stage('JUint') {
17          steps {
18              echo "JUint passed successfully!";
19              sh './unit.sh'
20          }
21      }
22
23      stage('Code Analysis') {
24          steps {
25              echo "Code Analysis completed successfully!";
26              sh './quality.sh'
27          }
28      }
29
30      stage('Deploy') {
31          steps {
32              echo "Deployed successfully!";
33              sh './deploy.sh'
34          }
35      }
36  }
```

```
JUint passed successfully!
[Pipeline] sh
+ ./unit.sh
Running Unit Test Cases : Wed Oct  6 04:34:20 UTC 2021
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Code Analysis)
[Pipeline] echo
Code Analysis completed successfully!
[Pipeline] sh
+ ./quality.sh
Code Quality Check : Wed Oct  6 04:34:20 UTC 2021
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] echo
Deployed successfully!
[Pipeline] sh
+ ./deploy.sh
Deploying the Project : Wed Oct  6 04:34:21 UTC 2021
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
This will always run
[Pipeline] echo
This will run when the state of the pipeline has changed
[Pipeline] echo
This will run when the run finished successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Exercise 11 # Maven build

- 이전에 작업했던 cicd-web-project 예제를 Maven 빌드 하기

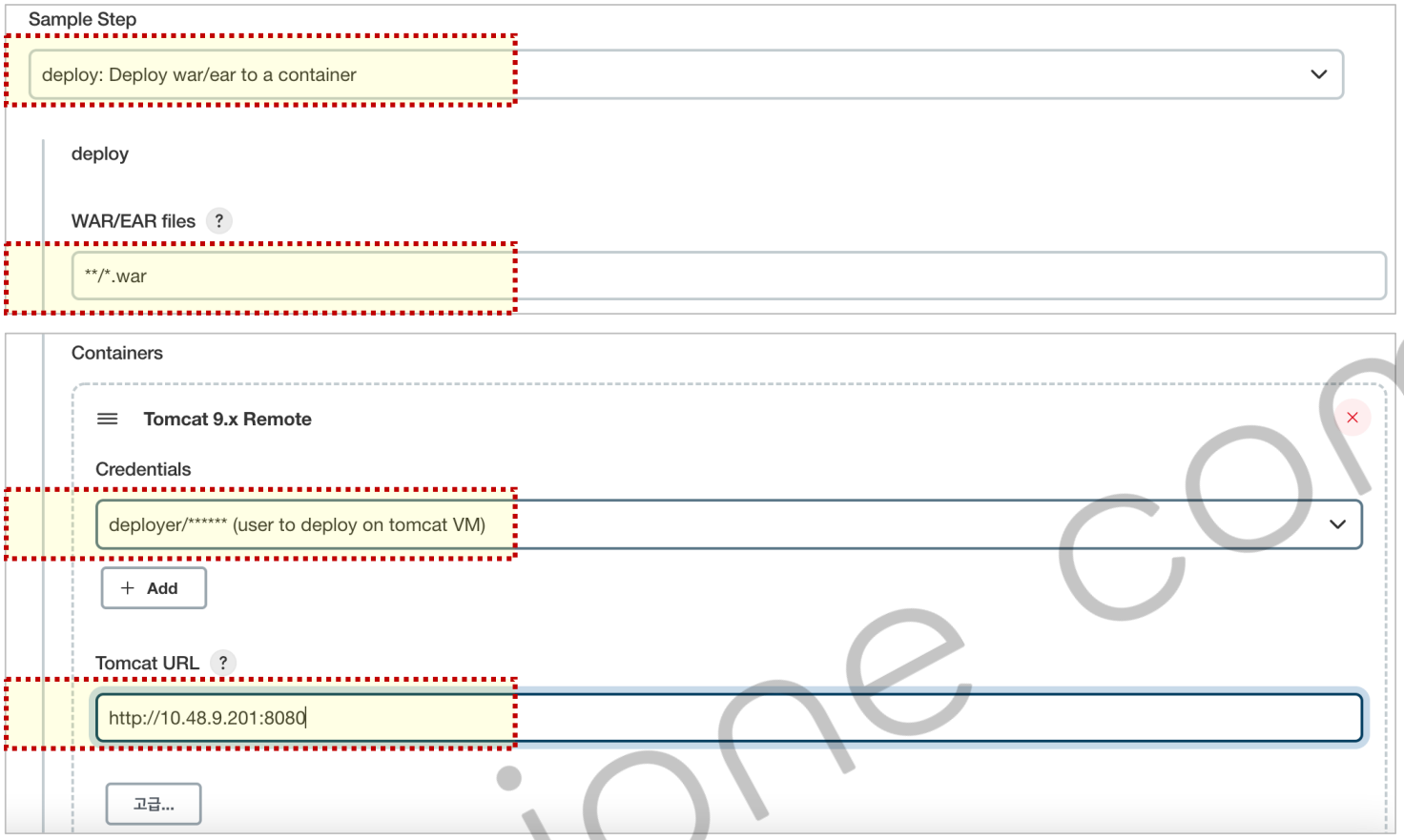
```
pipeline {
    agent any
    tools {
      maven 'Maven3.8.5'
    }
    stages {
        stage('github clone') {
            steps {
                git branch: 'main', url: 'https://github.com/joneconsulting/cicd-web-project.git';
            }
        }

        stage('build') {
            steps {
                sh '''
                    echo build start
                    mvn clean compile package -DskipTests=true
                '''
            }
        }

```

Exercise 12 # Deploy on Tomcat 1/2

- Exercise 11번 작업의 결과물을 tomcat9 서버에 배포하기



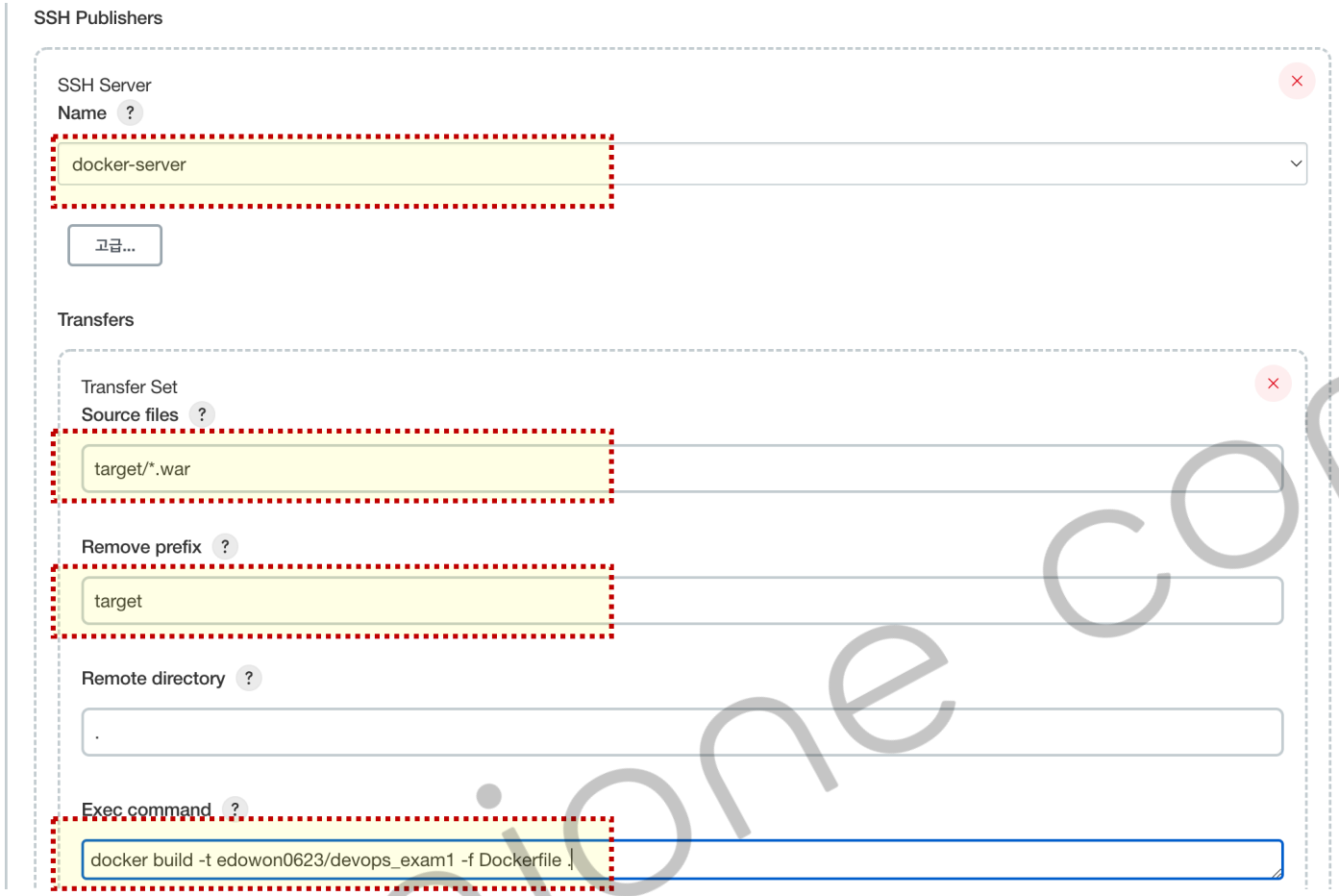
Exercise 12 # Deploy on Tomcat 2/2

- Exercise 11번 작업의 결과물을 tomcat9 서버에 배포하기

```
stage('deploy') {
        steps {
           deploy adapters: [tomcat9(credentialsId: 'deployer_user', path: '', url: 'http://10.48.9.201:8088/')], contextPath:
null, war: '**/*.war'
        }
    }
```

Exercise 13 # Deploy on Docker server 1/2

- Exercise 11번 작업의 결과물을 Docker server에 배포하기



Exercise 13 # Deploy on Docker server 2/2

njone company

- Exercise 11번 작업의 결과물을 Docker server에 배포하기

```
stage('ssh publisher') {
        steps {
           sshPublisher(publishers: [sshPublisherDesc(configName: 'docker-server', transfers: [sshTransfer(cleanRemote:
false, excludes: '', execCommand: 'docker build -t edowon0623/devops_exam1 -f Dockerfile .', execTimeout: 120000,
flatten: false, makeEmptyDirs: false, noDefaultExcludes: false, patternSeparator: '[, ]+', remoteDirectory: '.',
remoteDirectorySDF: false, removePrefix: 'target', sourceFiles: 'target/*.war')], usePromotionTimestamp: false,
useWorkspaceInPromotion: false, verbose: false)])
        }
    }
```