

Jenkins Pipeline

Jenkins Pipeline을 사용해 자동 배포 사이클을 구축했다.

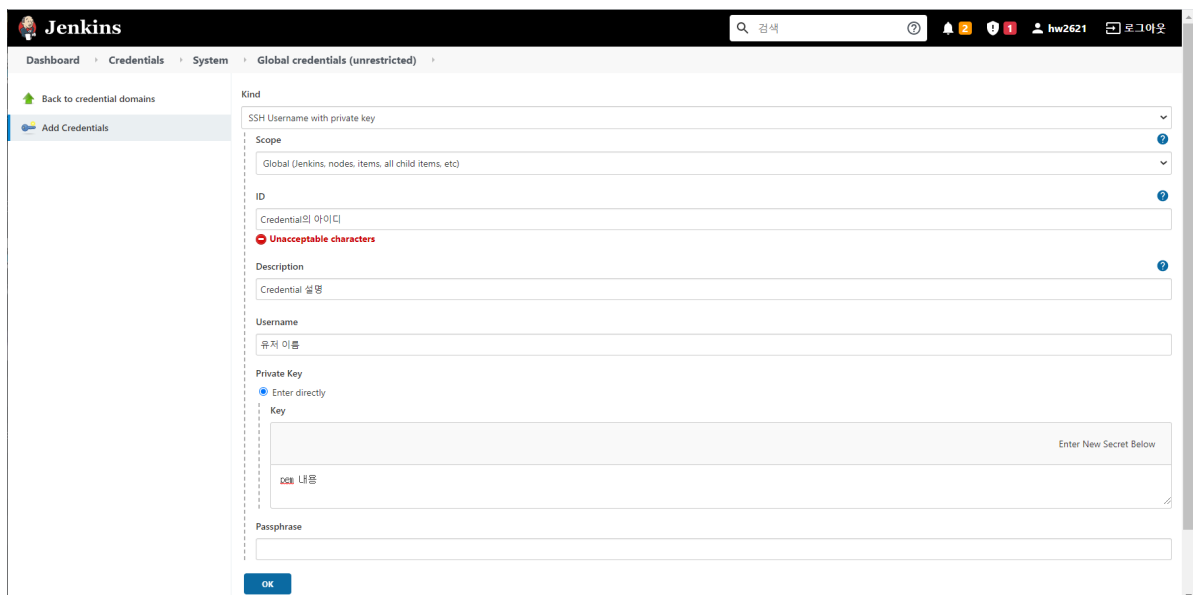
배포 사이클은 다음과 같다.

clone -> test -> build -> (staging -> release 는 생략) -> deploy

SSH Credential 생성

deploy에서는 우분투 서버의 배포 스크립트를 실행시켜야 한다.

때문에 ssh 연결을 위한 Credential을 생성해야 한다.



The screenshot shows the Jenkins web interface for adding a new credential. The breadcrumb trail is Dashboard > Credentials > System > Global credentials (unrestricted). The left sidebar has 'Back to credential domains' and 'Add Credentials'. The main form is titled 'Kind' and 'SSH Username with private key'. It includes fields for 'Scope' (set to 'Global (Jenkins, nodes, items, all child items, etc)'), 'ID' (with a warning 'Unacceptable characters'), 'Description' (set to 'Credential 설명'), 'Username' (set to '유저 이름'), 'Private Key' (set to 'Enter directly'), and 'Passphrase'. The 'Key' section has a large text area for the private key and a 'Key' label. The 'Passphrase' field is empty. An 'OK' button is at the bottom left.

Jenkins Pipeline 생성

1. 새로운 아이템 추가에서 Pipeline을 선택

Enter an item name

» Required field



Freestyle project

이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

다양한 환경에서의 테스트, 플레폼 특성 빌드, 기타 등등 처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

2. 빌드 트리거를 통해 언제 Pipeline을 실행시킬 지 설정한다

Dashboard > zuul >

General	Build Triggers	Advanced Project Options	Pipeline
<input type="checkbox"/> Throttle builds			
<input type="checkbox"/> 오래된 빌드 삭제			
<input type="checkbox"/> 이 빌드는 매개변수가 있습니다			
Build Triggers			
<input type="checkbox"/> Build after other projects are built			
<input type="checkbox"/> Build periodically			
<input type="checkbox"/> Build when a change is pushed to GitLab. GitLab webhook URL: http://f4f003.p.ssafy.io:30000/project/zuul			
<input type="checkbox"/> GitHub hook trigger for GITScm polling			
<input type="checkbox"/> Poll SCM			
<input type="checkbox"/> 빌드 안함			
<input type="checkbox"/> Quiet period			
<input type="checkbox"/> 빌드를 원격으로 유발 (예: 스크립트 사용)			
Advanced Project Options			
Display Name			
Pipeline			
Definition			
Pipeline script			
Script			
<button>저장</button> <button>Apply</button>			

스크립트 작성

Jenkins Pipeline을 작성하는 방법은 2가지가 있다.

1. 선언적 방식
2. 스크립트 방식

두 방법의 자세한 차이는 젠킨스 공식 문서를 참고하자.

다음은 스크립트 방식을 사용해 zuul 서비스를 우분투 서버에 배포하는 코드이다.

```
node {
  //Step 1. checkout git
  stage('Clone') {
    git branch: 'master', changelog: false, credentialsId: 'gitlab', poll:
false, url: 'https://lab.ssafy.com/s04-field/field-team3.git'
  }
  //Step 2. Build
  stage('Build zuul') {
```

```

        sh """
            mkdir -p
            /var/jenkins_home/workspace/zuul/backend/zuul/src/main/resources
            cp /var/jenkins_home/common/zuul/application.yml
            /var/jenkins_home/workspace/zuul/backend/zuul/src/main/resources
            cd /var/jenkins_home/workspace/zuul/backend/zuul/
            chmod +x ./gradlew
            ./gradlew clean
            ./gradlew build
        """
    }
    //Step 3. Deploy
    stage("SSH deploy") {
        withCredentials([sshUserPrivateKey(credentialsId: 'ec2-ubuntu-server',
            keyFileVariable: 'identity', passphraseVariable: '', usernameVariable:
            'userName'))] {
            def remote = [:]
            remote.name = "ubuntu"
            remote.host = "j4f003.p.ssafy.io"
            remote.allowAnyHosts = true
            remote.user = "ubuntu"
            remote.identityFile = identity
            sshCommand remote: remote, command: 'sh /home/ubuntu/CICD/deploy-
zuul.sh'
        }
    }
}

```

이제 단계별로 살펴보자.

Step 1

- 미리 설정한 Gitlab Credential을 통해 Gitlab에서 master 브랜치를 클론한다.

Step 2

- 해당 서비스가 저장된 디렉토리로 이동 후 빌드한다.
- 이 때, 배포용 application.yml 을 복사해 resources 폴더 하위에 위치시킨다.
- gradle을 통해 빌드한다.

Step 3

- ssh 연결을 통해 배포할 서버에 접속한다.
 - CredentialsId: 위 ssh credential에서 설정한 아이디
 - host: 배포할 서버의 주소
 - user: 배포할 서버의 사용자
- 배포할 서버에 저장된 스크립트를 실행한다

