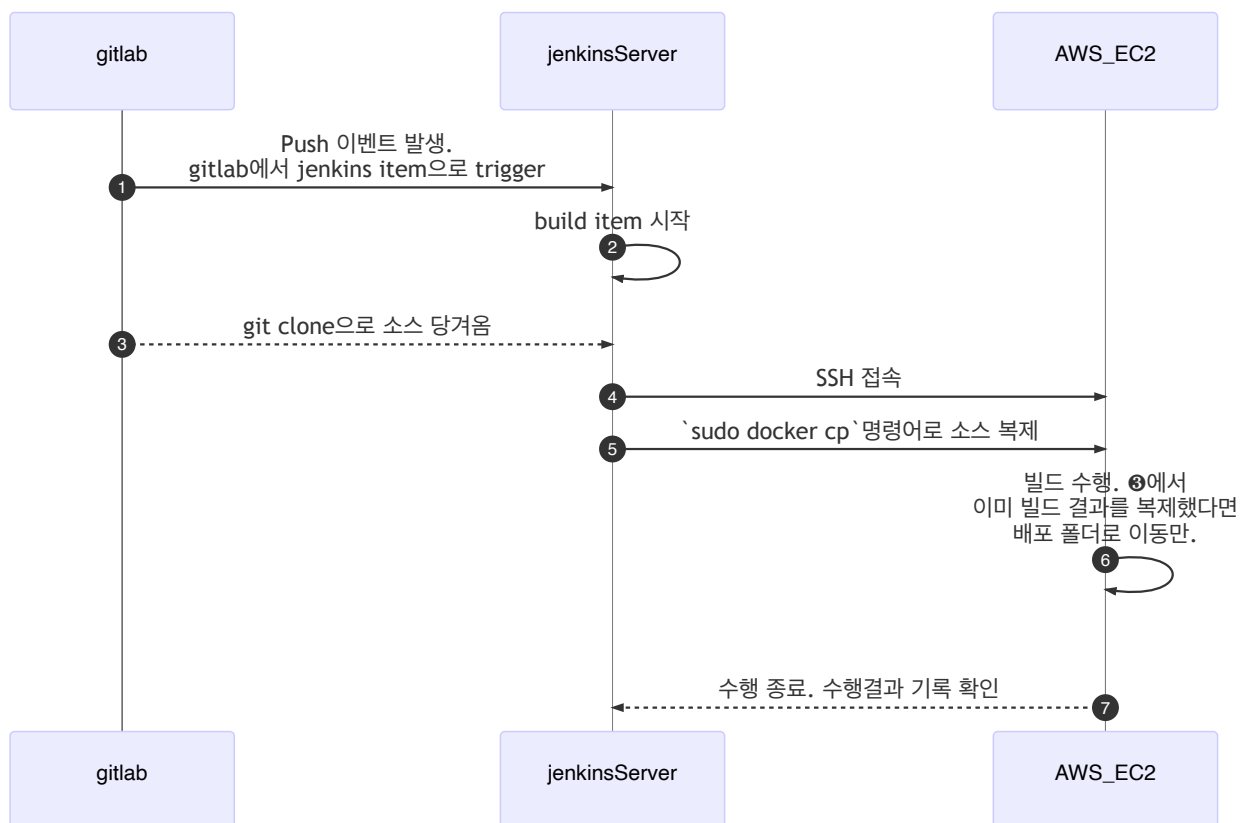


이 들은 docker jenkins 이미지로 설치한 경우 gitlab과의 연동을 정리한 글입니다.

docker image로 jenkins를 설치한 이유는 AWS EC2가 배포 및 운영 서버이기 때문에 jenkins를 바로 EC2에 설치하면 DB나 기타 서버 환경에 영향을 줄 수 있기 때문입니다. (docker를 통해 빈 가상 컴퓨터 하나를 더 만들었다고 생각하시면 됩니다)

gitlab서버는 [https://lab.ssafy.com/sungoonkim/cheuora\\_personel](https://lab.ssafy.com/sungoonkim/cheuora_personel) 이며 jenkins는 AWS EC2에 docker image로 설치하였습니다.

시퀀스 다이어그램을 그려보면 다음과 같습니다.



먼저 AWS\_EC2 서버에 docker image 로 jenkins를 설치합니다. 설치 방법은 아래 링크를 참조 바랍니다.

<http://jmlim.github.io/docker/2019/02/25/docker-jenkins-setup/>

<https://github.com/jenkinsci/docker/blob/master/README.md>

설치후에 유저생성 후 로그인하면 기본 플러그인만 깔린 상태 입니다. 여기에 우리는 두 가지 플러그인을 더 추가합니다.

- gitlab
- publish over SSH

설치는 Jenkins관리--> 플러그인 관리 에서 설치할 수 있습니다. (아래는 설치 후 화면)

먼저 gitlab의 설치 후 화면 입니다.

🔍 gitlab

업데이트된 플러그인 목록

설치 가능

설치된 플러그인 목록

고급

사용가능

이름 ↓

버전

이전 설치 버전

설치 제거

☒

Credentials Plugin

This plugin allows you to store credentials in Jenkins.

2.3.13

설치 제거

☒

Display URL API

Provides the DisplayURLProvider extension point to provide alternate URLs for use in notifications

2.3.3

설치 제거

☒

Git client plugin

Utility plugin for Git support in Jenkins

3.4.2

설치 제거

☒

Git plugin

This plugin integrates [Git](#) with Jenkins.

4.4.1

설치 제거

☒

GitLab Plugin

This plugin allows [GitLab](#) to trigger Jenkins builds and display their results in the GitLab UI.

1.5.13

설치 제거

☒

Matrix Project Plugin

1.17

설치 제거

다음은 publish over SSH 설치 후 화면 입니다.

<input checked="" type="checkbox"/>	<b>Git client plugin</b> Utility plugin for Git support in Jenkins	3.4.2		설치 제거
<input checked="" type="checkbox"/>	<b>Git plugin</b> This plugin integrates <a href="#">Git</a> with Jenkins.	4.4.1		설치 제거
<input checked="" type="checkbox"/>	<b>Infrastructure plugin for Publish Over X</b> Send build artifacts somewhere.	0.22		설치 제거
<input checked="" type="checkbox"/>	<b>JSch dependency plugin</b> Jenkins plugin that brings the JSch library as a plugin dependency, and provides an SSHAuthenticatorFactory for using JSch with the ssh-credentials plugin.	0.1.55.2		설치 제거
<input checked="" type="checkbox"/>	<b>Publish Over SSH</b> Send build artifacts over SSH	1.20.1		설치 제거
	<b>This plugin is up for adoption!</b> We are looking for new maintainers. Visit our <a href="#">Adopt a Plugin</a> initiative for more information.			

이제 Jenkins내에서 작업 item을 생성해야 합니다. 메인 페이지에서 새로운아이템을 클릭하고 "Freestyle Project"를 선택하고 OK를 클릭합니다.

**Enter an item name**

Gitlab\_test  
» 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, 플랫폼 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.

**OK** **Organization**  
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

이제 ❶을 위한 작업을 하겠습니다. gitlab에서 jenkins에 트리거를 전송하려면 jenkins에서 제공하는 secrete token이 필요합니다. 이 코드는 생성된 작업 아이템 설정 중 **빌드유발** 섹션중 'Build when a change is pushed to GitLab. GitLab webhook URL: [http://i02prf9.p.ssafy.io:8080/project/Gitlab\\_test](http://i02prf9.p.ssafy.io:8080/project/Gitlab_test)' 를 클릭 --> '고급' 버튼을 누르면 생성할 수 있습니다.

☒ Build when a change is pushed to GitLab. GitLab webhook URL: [http://i02prf9.p.ssafy.io:8080/project/Gitlab\\_test](http://i02prf9.p.ssafy.io:8080/project/Gitlab_test)

Enabled GitLab triggers

Push Events	<input checked="" type="checkbox"/>
Opened Merge Request Events	<input checked="" type="checkbox"/>
Accepted Merge Request Events	<input type="checkbox"/>
Closed Merge Request Events	<input type="checkbox"/>
Rebuild open Merge Requests	Never
Approved Merge Requests (EE-only)	<input checked="" type="checkbox"/>
Comments	<input checked="" type="checkbox"/>
Comment (regex) for triggering a build	Jenkins please retry a build

**고급...**

Set build description to build cause (eg. Merge request or Git Push) ☒

Build on successful pipeline events ☐

Pending build name for pipeline

Cancel pending merge request builds on update ☐

Allowed branches

- ☒ Allow all branches to trigger this job
- ☐ Filter branches by name
- ☐ Filter branches by regex
- ☐ Filter merge request by label

Secret token

**Generate** Clear

생성된 Secret token값을 복사하고 '저장' 을 눌러 저장하고 빠져 나옵니다.

이제 gitlab에서 push이벤트에 대한 trigger를 만들어 줍니다. 먼저 gitlab페이지에서 Settings-->integration 에 접속합니다.

Sungjoon Kim > cheuora\_personel > Integrations Settings

### Integrations

Webhooks can be used for binding events when something is happening within the project.

**URL**

**Secret Token**

Use this token to validate received payloads. It will be sent with the request in the X-Gitlab-Token HTTP header.

**Trigger**

- ☒ **Push events**  
This URL will be triggered by a push to the repository
- ☐ **Tag push events**  
This URL will be triggered when a new tag is pushed to the repository
- ☐ **Comments**  
This URL will be triggered when someone adds a comment
- ☐ **Confidential Comments**  
This URL will be triggered when someone adds a comment on a confidential issue
- ☐ **Issues events**  
This URL will be triggered when an issue is created/updated/merged
- ☐ **Confidential Issues events**

url에 trigger할 jenkins 작업 item URL를 입력하고 Secret Token 부분에는 아까 복사한 Token를 입력합니다.

push events에는 적용할 브랜치를 적어 줍니다. 빈 값이면 모든 브랜치에 적용됩니다. 저는 master만 적용하도록 하겠습니다.

주의할 점은 URL 값은 jenkins 작업아이템 설정 중 빌드유발 섹션에서 생성된 URL을 입력해야 한다는 것입니다.(여기에서는 [http://i02prf9.p.ssafy.io:8080/project/Gitlab\\_test](http://i02prf9.p.ssafy.io:8080/project/Gitlab_test) 가 되며 웹 페이지에서 다이렉트로 접근하는 URL과는 조금 다름)

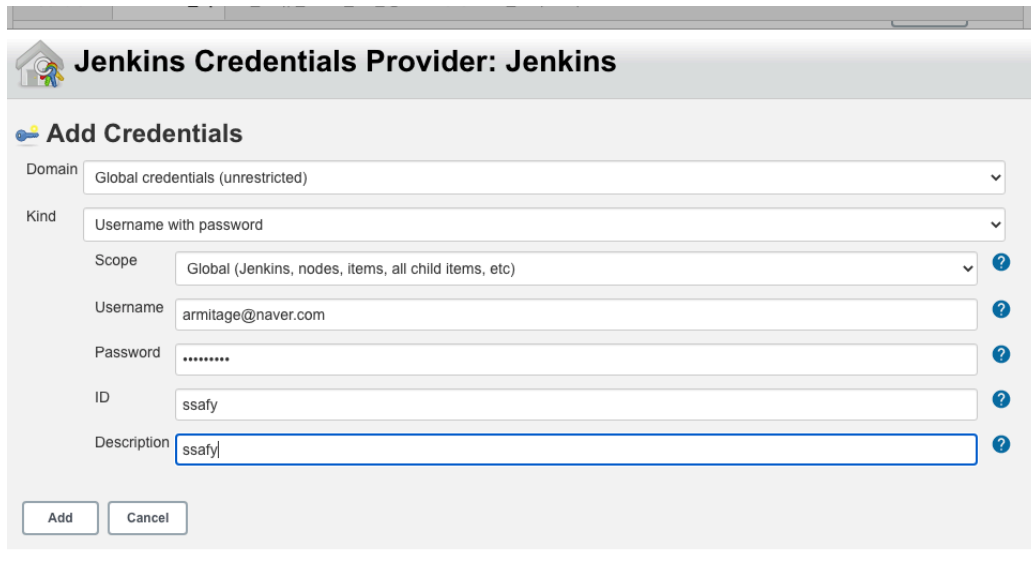
하단의 'Add Webhook' 버튼을 눌러 트리거를 생성하고 push 이벤트로 테스트 해 봅니다. 성공입니다.

이제 ❷ 및 ❸를 위한 작업을 해 봅시다.

생성한 작업아이템 'Gitlab\_test' 의 구성-->소스코드관리 탭을 보시면

gitlab에서 git clone을 하기 위한 url과 Credential을 입력하는 란이 있습니다. Repository URL에는 gitlab의 clone URL을 입력하고 Credential에서는 아직 설정된게 없기 때문에 Add를 클릭합니다.

Kind는 Username with password를 선택하고 gitlab의 ID와 PW를 차례로 입력하고 Add를 클릭합니다.



**Jenkins Credentials Provider: Jenkins**

**Add Credentials**

Domain: Global credentials (unrestricted)

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: armitage@naver.com

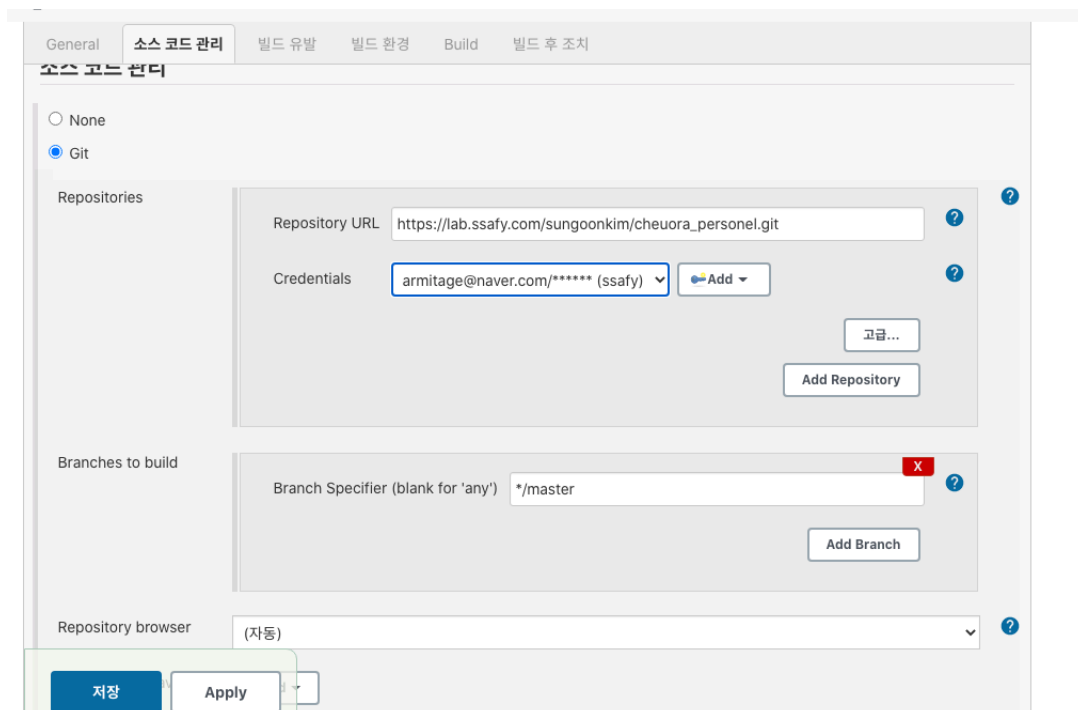
Password: .....

ID: ssafy

Description: ssafy

Buttons: Add, Cancel

credential에서 방금 입력한 id/pw를 선택하고 apply를 클릭합니다.



General | **소스 코드 관리** | 빌드 유발 | 빌드 환경 | Build | 빌드 후 조치

**소스 코드 관리**

☐ None  
☒ Git

Repositories

Repository URL: https://lab.ssafy.com/sungoonkim/cheuora\_personel.git

Credentials: armitage@naver.com/\*\*\*\*\* (ssafy) [Add]

Buttons: 고급..., Add Repository

Branches to build

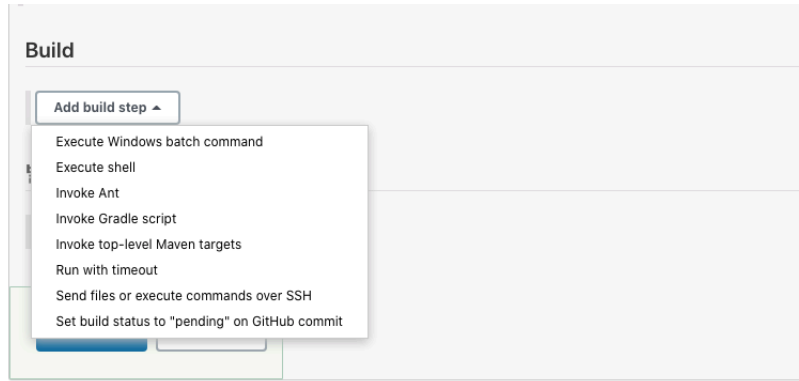
Branch Specifier (blank for 'any'): \*/master

Buttons: Add Branch

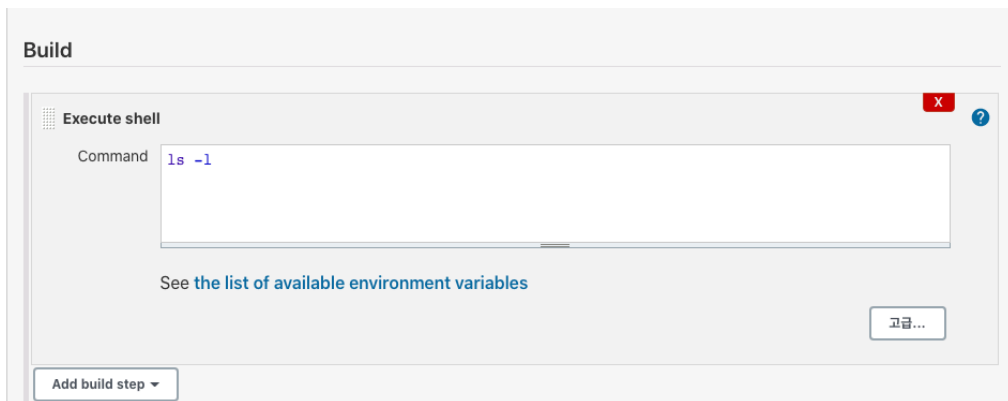
Repository browser: (자동)

Buttons: 저장, Apply

작업 아이템이 수행될때 먼저 수행되는게 **Build** 섹션의 내용입니다. 이 섹션으로 이동하여 Add build step을 선택합니다.



여기에서는 Execute shell을 선택하겠습니다. Build섹션의 내용은 jenkins서버에서 실행되며, 우리가 설치한 jenkins는 docker image를 통한 가상서버에서 실행되는 것입니다. (여기에는 최소 jenkins실행을 위한 OS정도로 깔려 있습니다. vi도 없습니다.) 일단 Build섹션에서는 내용 확인을 위해 `ls -l` 명령을 실행하는 것으로 하겠습니다.



이제 빌드 이후의 작업들인 ④,⑤,⑥을 위한 설정을 해 봅시다. 먼저 jenkins시스템이 SSH서버(여기서는 AWS\_EC2)를 인식하기 위한 설정을 해야 합니다. 이는 홈 페이지 --> Jenkins관리 --> 시스템 설정 --> Publish over SSH 섹션에서 설정합니다.

## Publish over SSH

Jenkins SSH Key		?
Passphrase	<input type="text"/>	?
Path to key	<input type="text"/>	?
Key	<input type="text"/>	?
Disable exec	<input type="checkbox"/>	?
SSH Servers	<input type="button" value="추가"/>	
		<input type="button" value="고급..."/>

key 부분에 AWS EC2접속에 사용되는 \*.pem 파일 내용을 그대로 복사해 넣습니다. 그리고 SSH Server의 추가 버튼을 클릭한 후 서버 접속 정보를 넣습니다.



Passphrase

Path to key

Key 

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAkUIRnM6/cYmqN1eZnuKbNBahzHpkkKjNvHD3OdDN4jGc5rwlgc
vtEqqu5eI
RmtM/k+94bvs0BNSrKgfW7DJMTvusJV7ha+M9B3J2qVMb6+mMeFreYMR/CONOuD
B4iCE0BrGZJb/
-----END RSA PRIVATE KEY-----
```

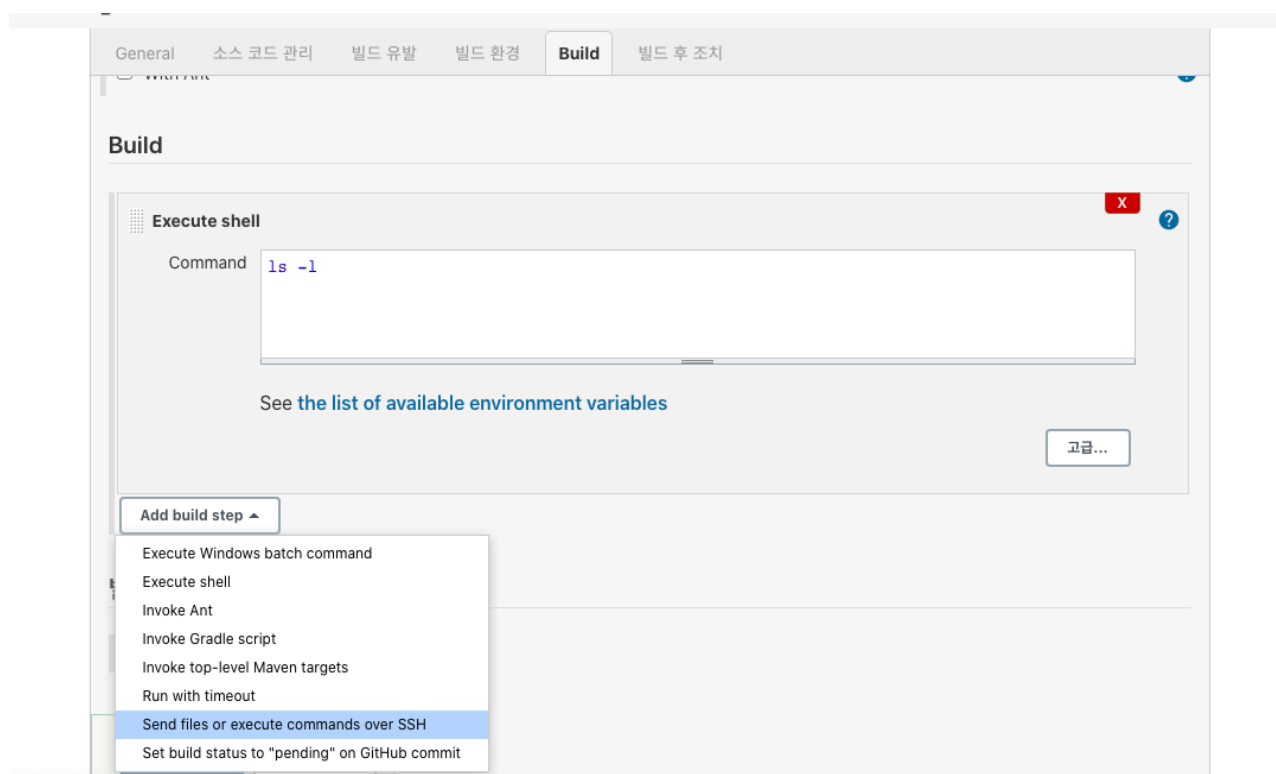
Disable exec ☐

SSH Servers

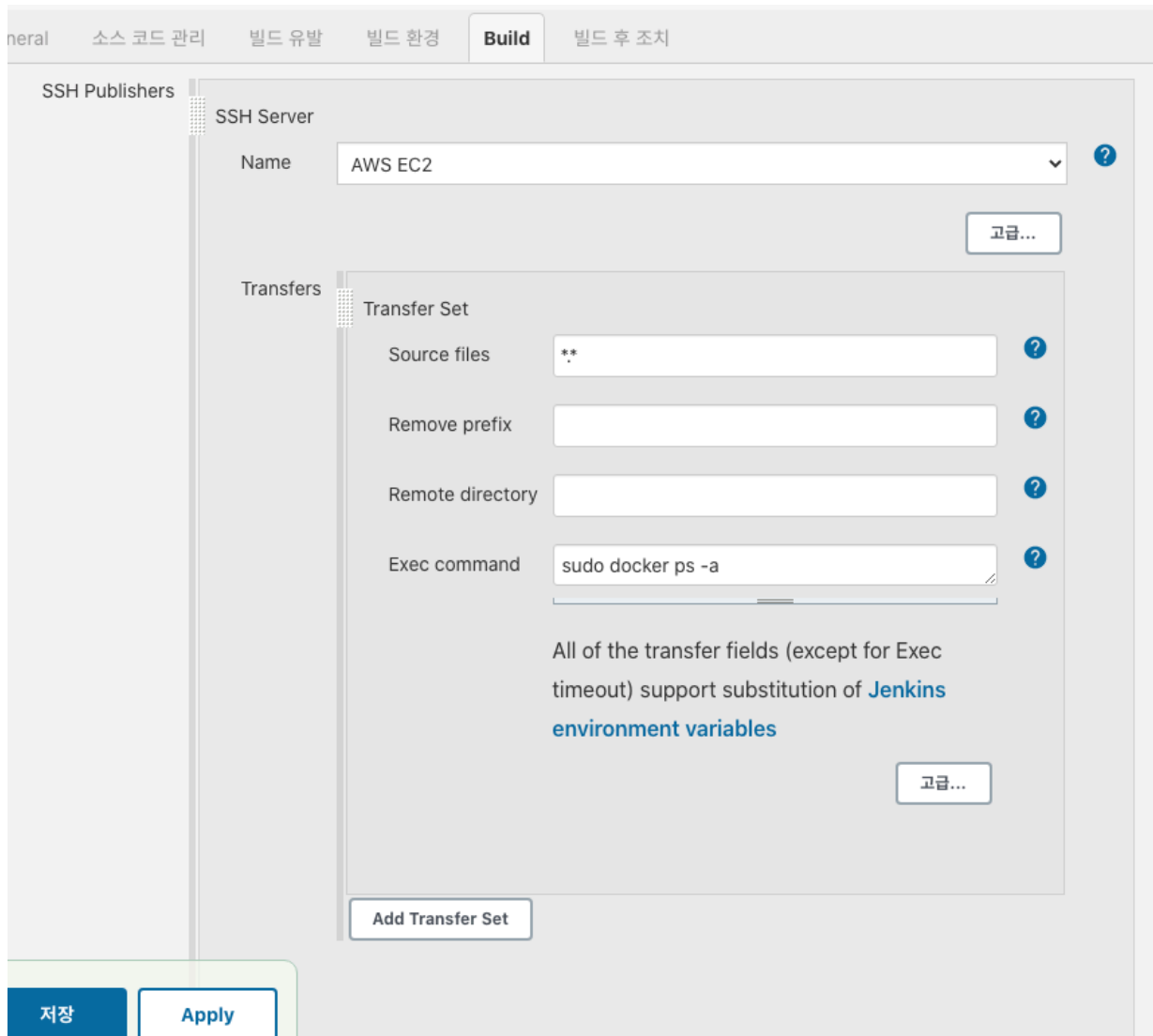
SSH Server	
Name	<input type="text" value="AWS EC2"/> ?
Hostname	<input type="text" value="i02prf9.p.ssafy.io"/> ?
Username	<input type="text" value="ubuntu"/> ?
Remote Directory	<input type="text" value="/home/ubuntu"/> ?

'Test Configuration'버튼으로 접속 테스트를 한 후 성공이 뜨면 '저장' 버튼을 눌러 빠져나옵니다.

빌드 이후의 작업들은 빌드 환경, build스텝 등에 설정을 할 수 있으며, 여기에서는 build스텝에서 "Add build step" 버튼을 눌러 설정하겠습니다. 버튼을 눌러 'Send files or execute commands over SSH' 를 선택합니다.



선택하면 아까 설정한 AWS EC2서버가 기본으로 뜨고 이동시킬 파일 형식 및 수행할 커멘드를 입력합니다. 다시 여기에서 빌드를 하려면 `npm run build` 명령을 입력하셔도 되며 여기에서는 그냥 테스트로 `docker` 명령어를 입력하였습니다.



The image shows the 'SSH Publishers' configuration page in Jenkins. The 'SSH Server' section has 'Name' set to 'AWS EC2'. The 'Transfers' section contains a 'Transfer Set' with the following fields: 'Source files' set to '\*\*', 'Remove prefix' (empty), 'Remote directory' (empty), and 'Exec command' set to 'sudo docker ps -a'. There are help icons (?) next to each field. Below the fields, a note states: 'All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)'. At the bottom of the 'Transfers' section is an 'Add Transfer Set' button. At the bottom of the entire configuration page are '저장' (Save) and 'Apply' buttons.

이 세트의 의미는 git clone한 모든 파일( \*.\* ) 들을 SSH서버로 이동시키고 그냥 `sudo docker ps -a` 명령을 SSH 서버상에서 실행하고 마칩니다.

테스트를 해 보겠습니다.

먼저 gitlab 사이트 입니다.

master

cheuora\_personel / +

History

Find file

Web IDE

Delete 테스트 파일.  
Sungjoon Kim authored 56 minutes ago

2bae743e

README

Add CHANGELOG

Add CONTRIBUTING

Add Kubernetes cluster

Set up CI/CD

Name	Last commit	Last update
README.md	Update README.md	1 month ago
test1.md	test1.md	1 month ago
test2.md	test2	1 month ago
test3.md	test3	1 month ago

README.md

gitlab의 Setting --> Integrations 에서 설치된 Web hook에서 test --> Push event를 선택합니다.

☐ **Job events**  
This URL will be triggered when the job status changes

☐ **Pipeline events**  
This URL will be triggered when the pipeline status

☐ **Wiki Page events**  
This URL will be triggered when a wiki page is creat

**SSL verification**  
☒ **Enable SSL verification**  

Add webhook

**Webhooks (1)**  

http://i02prf9.p.ssafy.io:8080/project/Gitlab\_test

Push Events

SSL Verification: enabled

Edit

Test

Push events

Tag push events

Issues events

Confidential issues events

Note events

Confidential note events

Merge requests events

Job events


Pipeline events

gitlab 페이지에서 성공이 뜨면 이제 jenkins에서 확인

```

Started by GitLab push by Sungoon Kim
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/GitLab_test
The recommended git tool is: NONE
using credential ssaify
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://lab.ssaify.com/sungoonkim/cheuora\_personal.git # timeout=10
Fetching upstream changes from https://lab.ssaify.com/sungoonkim/cheuora\_personal.git
> git --version # timeout=10
> git --version # 'git version 2.11.0'
using GIT_ASKPASS to set credentials ssaify
> git fetch --tags --progress -- https://lab.ssaify.com/sungoonkim/cheuora\_personal.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
skipping resolution of commit remotes/origin/master, since it originates from another repository
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin^{commit} # timeout=10
Checking out Revision 2bae743ed88144609eeb70b99eb575e04c795d3d (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 2bae743ed88144609eeb70b99eb575e04c795d3d # timeout=10
Commit message: "Delete 테스트 파일."
> git rev-list --no-walk a7fb6a3a46736c26807115214fa19d70bc893f4 # timeout=10
[GitLab test $] /bin/sh -xe /tmp/jenkins3481378794783720128.sh
+ ls -l
total 16
-rw-r--r-- 1 jenkins jenkins 475 Sep 3 15:46 README.md
-rw-r--r-- 1 jenkins jenkins 60 Sep 3 15:46 test1.md
-rw-r--r-- 1 jenkins jenkins 24 Sep 3 15:46 test2.md
-rw-r--r-- 1 jenkins jenkins 30 Sep 3 15:46 test3.md
SSH: Connecting from host [0808593a3561]
SSH: Connecting with configuration [AWS EC2] ...
SSH: EXEC: STDOUT/STDERR from command [sudo docker ps -a] ...

```

[view as plain text](#) 빌드 정보 수정 Delete build '#5' Polling Log Git Build Data No Tags 이전 빌드

```
> git config remote.origin.url https://lab.ssafy.com/sungoonkim/cheuora_personel.git # timeout=10
Fetching upstream changes from https://lab.ssafy.com/sungoonkim/cheuora_personel.git
> git --version # timeout=10
> git --version # 'git version 2.11.0'
using GIT_ASKPASS to set credentials ssafy
> git fetch --tags --progress -- https://lab.ssafy.com/sungoonkim/cheuora_personel.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
skipping resolution of commit remotes/origin/master, since it originates from another repository
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 2bae743ed88144609eeb70b99eb575e04c795d3d (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 2bae743ed88144609eeb70b99eb575e04c795d3d # timeout=10
Commit message: "Delete 테스트 파일."
> git rev-list --no-walk a7fb6a3a46736c626807115214fa19d70bc893f4 # timeout=10
[Gitlab_test] $ /bin/sh -xe /tmp/jenkins3481378794783720128.sh
+ ls -l
total 16
-rw-r--r-- 1 jenkins jenkins 475 Sep  3 15:46 README.md
-rw-r--r-- 1 jenkins jenkins 60 Sep  3 15:46 test1.md
-rw-r--r-- 1 jenkins jenkins 24 Sep  3 15:46 test2.md
-rw-r--r-- 1 jenkins jenkins 30 Sep  3 15:46 test3.md
SSH: Connecting from host [0808593ab558]
SSH: Connecting with configuration [AWS EC2] ...
SSH: EXEC: STDOUT/STDERR from command [sudo docker ps -a] ...
CONTAINER ID        IMAGE               COMMAND                  CREATED            STATUS              PORTS
0808593ab558       jenkins/jenkins    "/sbin/tini -- /usr/"   25 hours ago       Up 2 hours
0.0.0.0:8080->8080/tcp, 0.0.0.0:50000->50000/tcp    jenkins
3ddc679c66d2       mariadb             "docker-entrypoint.s..." 5 weeks ago        Exited (0) 2 days ago
mariadb
SSH: EXEC: completed after 210 ms
SSH: Disconnecting configuration [AWS EC2] ...
SSH: Transferred 4 file(s)
Build step 'Send files or execute commands over SSH' changed build result to SUCCESS
Finished: SUCCESS
```