

Docker 란?

도커는 컨테이너 기반의 오픈소스 가상화 플랫폼이다.

화물선에 담긴 컨테이너를 떠올리면, 딱 맞는 규격에 여러 화물을 실어 쉽게 운송할 수 있다는 사실을 알 수 있다.

도커 또한 비슷한 개념이다. 다양한 프로그램들과 실행환경을 컨테이너로 규격화시켜 프로그램의 배포 및 관리를 단순화할 수 있다

도커를 사용하면 많은 이점이 있다. 가상머신 처럼 다른 프로세스와 격리되어 사용할 수 있지만, 성능 저하가 거의 없다.

환경에 구애받지 않고 실행할 수 있으며 배포도 간단하다. 이러한 방식은 최근 **MSA** 방식과 잘 어울린다고 할 수 있다.

오픈 소스이기 때문에 특정 회사에 종속되지 않는다는 점도 장점이다. 복잡한 기술을 몰라도 사용할 수 있으며, 레이어를 이용할 수 있다.

Docker 컨테이너는 애플리케이션의 모든 코드 및 종속성을 표준 형식으로 패키징할 수 있게 해주는 컨테이너입니다.

이를 통해 애플리케이션이 컴퓨팅 환경 전반에서 빠르고 안정적으로 실행될 수 있죠.

Docker 컨테이너는 라이브러리, 시스템 도구, 코드, 런타임 등 애플리케이션 실행에 필요한 모든 것을 담고 있는 인기 있는 경량의 독립형 실행 컨테이너입니다.

Docker는 개발자가 컨테이너화된 애플리케이션을 빠르게 빌드, 테스트 및 배포할 수 있게 해주는 소프트웨어 플랫폼이기도 합니다.

클라우드 서버 접속

ssh root@서버ip

도커 설치 스크립트 다운로드

```
# apt-get update
# apt-get install curl
# curl https://get.docker.com > docker-install.sh
# chmod 755 docker-install.sh
```

도커 설치

```
# ./docker-install.sh
```

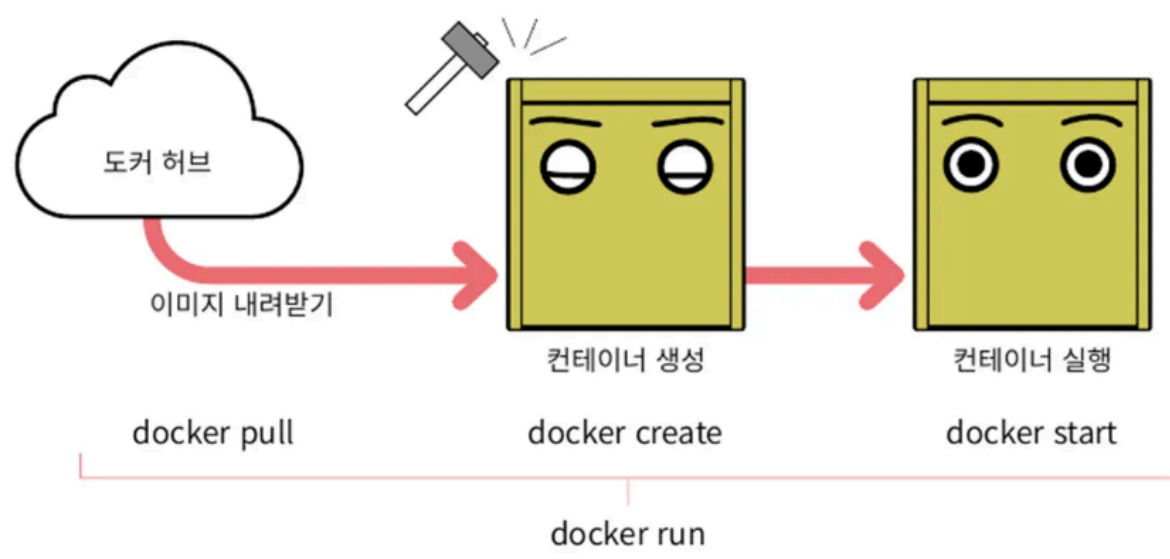
도커 버전 확인

```
# docker -v
```

도커 이미지 목록 확인하기

```
docker image ls
docker images
```

docker run 커맨드



위의 3개의 명령을 합친게 **run**

컨테이너 생성하고 실행하는 커맨드: docker container run

docker image pull, docker container create, docker container start 기능을 하나로 합친 명령어

docker run (옵션) 이미지 (인자)

주요 옵션

옵션 형식	내용
--name 컨테이너_이름	컨테이너 이름을 지정함
-p 호스트_포트번호:컨테이너_포트번호	포트 번호를 지정함
-v 호스트_디스크:컨테이너_디렉터리	볼륨을 마운트함
--net=네트워크_이름	컨테이너를 네트워크에 연결함
-e 환경변수_이름=값	환경변수를 설정함
-d	백그라운드로 실행함
-i	컨테이너에 터미널(키보드)을 연결함
-t	특수 키를 사용 가능하도록 함
-help	사용 방법 안내 메시지를 출력함

-p는 --publish, -v는 --volume, -e는 --env, -d는 --detach, -i는 --interactive, -t는 --tty의 생략형이다.

docker container [하위_커맨드] [옵션]

주요 하위 커맨드

하위 커맨드	내용	생략 가능 여부	주요 옵션
start	컨테이너를 실행	O	-i
stop	컨테이너를 정지	O	거의 사용하지 않음
create	도커 이미지로부터 컨테이너를 생성	O	--name -e -p -v
run	도커 이미지를 내려받고 컨테이너를 생성해 실행함(다운 로드는 필요한 경우에만). docker image pull, docker container create, docker container start라는 세 개의 명령을 하나로 합친 것과 같다.	O	--name -e -p -v -d -i -t
rm	정지 상태의 컨테이너를 삭제	O	-f -v
exec	실행 중인 컨테이너 속에서 프로그램을 실행	O	-i -t
ls	컨테이너 목록을 출력	*1	-a
cp	도커 컨테이너와 도커 호스트 간에 파일을 복사	O	거의 사용하지 않음
commit	도커 컨테이너를 이미지로 변환	O	거의 사용하지 않음

docker image [하위_커맨드] [옵션]

주요 하위 커맨드

하위 커맨드	내용	생략 가능 여부	주요 옵션
pull	도커 허브 등의 리포지토리에서 이미지를 내려받음	O	거의 사용하지 않음
rm	도커 이미지를 삭제	*2	거의 사용하지 않음
ls	내려 받은 이미지의 목록을 출력	X	거의 사용하지 않음
build	도커 이미지를 생성	O	-t

테스트

docker run hello-world

다시 한번 도커 이미지 목록 확인하기

docker image ls

docker images

모든 컨테이너 확인 (실행중&미실행중인거 포함)

`docker ps` : 실행중인거만 확인하는건

`docker ps -a`

컨테이너 먼저 삭제후 이미지 삭제

`docker container stop 0cc083767d4c` : 반드시 **stop** 후 삭제

`docker container rm 0cc083767d4c`

실습 2

`docker run --name apaex1 -d httpd`

```
root@bitcamp-jenkins:~# docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
634ece1eb3a8   httpd      "httpd-foreground"      About a minute ago    Up 59 seconds    80/tcp        apaex1
root@bitcamp-jenkins:~# docker images
REPOSITORY    TAG        IMAGE ID      CREATED        SIZE
httpd         latest    0de612e99135  5 weeks ago   148MB
root@bitcamp-jenkins:~#
```

`docker container stop apaex1` (name 으로 stop 해도됨)

`docker container rm apaex1`

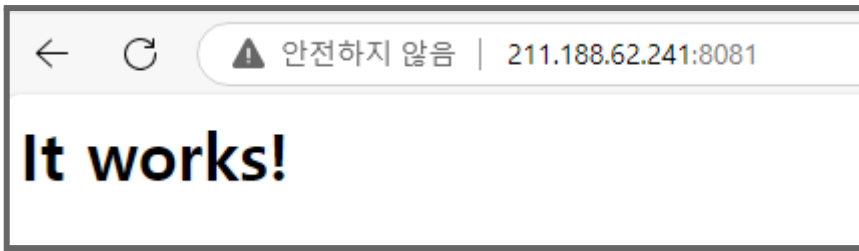
`docker image rm 0d`

실습 3 - (-p 로 포트 지정후 브라우저로 실행해보자)

내부포트 **80**을 외부포트 **8080** 으로 변경해서 **httpd**이미지 내려받기

`docker run --name apach1 -d -p 8080:80 httpd`

`docker run --name apach2 -d -p 8081:80 httpd`



```
root@bitcamp-jenkins:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                    NAMES
5ada5bcc8d37   httpd     "httpd-foreground"      44 seconds ago Up 43 seconds  0.0.0.0:8081->80/tcp     apach2
9ea849dde52b   httpd     "httpd-foreground"      2 minutes ago  Up 2 minutes  0.0.0.0:8080->80/tcp     apach1
root@bitcamp-jenkins:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
httpd         latest   0de612e99135  5 weeks ago   148MB
root@bitcamp-jenkins:~#
```

apach2 로 접속해보자

`docker exec -i -t apach2 bash` : 접속(빠져나가는건 `exit`)

`index.html` 은 `htdocs` 폴더에 있으므로 이동

`cd htdocs`

```
root@bitcamp-jenkins:~# docker exec -i -t apach2 bash
root@5ada5bcc8d37:/usr/local/apache2# ls
bin build cgi-bin conf error htdocs icons include logs modules
root@5ada5bcc8d37:/usr/local/apache2# ls bin
ab apxs dbmmanage envvars-std htcacheclean htdigest httpd
apachectl checkgid envvars fcgidstarter htdbm htpasswd httxt2dbm
root@5ada5bcc8d37:/usr/local/apache2# ls
bin build cgi-bin conf error htdocs icons include logs modules
root@5ada5bcc8d37:/usr/local/apache2# cd htdocs
root@5ada5bcc8d37:/usr/local/apache2/htdocs# ls
index.html
root@5ada5bcc8d37:/usr/local/apache2/htdocs# cat index.html
<html><body><h1>It works!</h1></body></html>
root@5ada5bcc8d37:/usr/local/apache2/htdocs# vi index.html
bash: vi: command not found
```

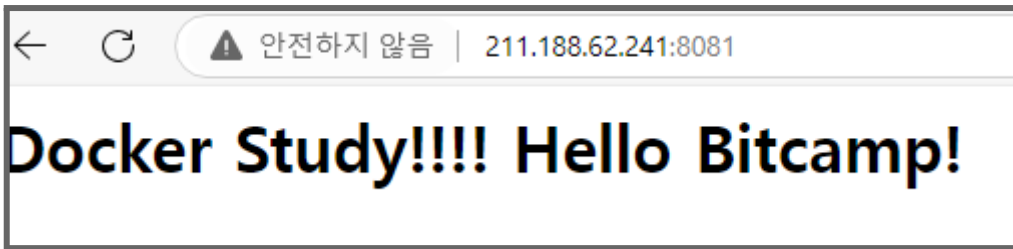
vi 내려받기

`apt-get update`

`apt-get install vim`

vi `index.html` 로 들어가서 마음대로 편집후 저장

211.188.62.241:8081 : 수정된 내용 확인(ip 는 각자 자기 서버 IP)



연습끝난후 모두 삭제하기

```
docker container stop apach1
docker container stop apach2
docker container rm apach1 apach2
```

```
docker image rm 0d
```

```
root@bitcamp-jenkins:~# docker container stop apach1
apach1
root@bitcamp-jenkins:~# docker container stop apach2
apach2
root@bitcamp-jenkins:~# docker container rm apach1 apach2
apach1
apach2
root@bitcamp-jenkins:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
httpd         latest   0de612e99135   5 weeks ago   148MB
root@bitcamp-jenkins:~# docker image rm 0d
Untagged: httpd:latest
Untagged: httpd@sha256:10381816bb7e60ae3a9db3784f2966a8910b6ff07c4da54bd2d62d2671c8ab6e
Deleted: sha256:0de612e991352926be994158377c9f14147fe31e3ae92927f3b1b37dbf88ab10
Deleted: sha256:3795bd2a2e330f77b5c07052dcf338a9251c4803bc5d740cd9e058a30bb5add1
Deleted: sha256:500029e2c565c90e9c3f458942ce64587a2a378486d5b2d389ec4db7d27f301c
Deleted: sha256:2d5a8b1049b135735434609b2b93404099cfedec72a09f3406d7f290fb9ed711
Deleted: sha256:a946867c364480b5bd05bc4323a57c4c88d4a6f2813d966a649182daa3e2bd7a
Deleted: sha256:a1403f0b4ad2ef14d87d4214b2afd699aefefef6a0fcf3493704b1231ab2ef5b
Deleted: sha256:5f1ee22ffb5e68686db3dcb6584eb1c73b5570615b0f14fabb070b96117e351d
root@bitcamp-jenkins:~#
```

실습 4 : mysql 설치

```
docker run -d --name mydb -e MYSQL_ROOT_PASSWORD=1234 -p 3307:3306 -e
MYSQL_DATABASE=mydata mysql:8
```

실행중인 **mysql** 에 접속하기

docker exec -i -t mydb bash

mysql 접속

mysql -u root -p 엔터
1234 엔터

mysql>

exit 로 빠져나간후 **mysql** 컨테이너와 이미지 모두 삭제하기

```
root@bitcamp-jenkins:~# docker container stop mydb
mydb
root@bitcamp-jenkins:~# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
root@bitcamp-jenkins:~# docker container rm mydb
mydb
root@bitcamp-jenkins:~# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
root@bitcamp-jenkins:~# docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
root@bitcamp-jenkins:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mysql         8         9755c387b39e   5 weeks ago    769MB
root@bitcamp-jenkins:~# docker image rm 97
Untagged: mysql:8
Untagged: mysql@sha256:0917ecc5863323a48203dda0bb7d58582d958da62914024c474bf2e8c5f5ee73
Deleted: sha256:9755c387b39e71d6f2f8f04b57c9461bf14c8023cb8105768a6e79936f62ed02
Deleted: sha256:445ba40e24756bc8761055c244938331ee2e2781821fe3b8d9678de3b5168d51
Deleted: sha256:6df09fc689f6b0cd56ac0c27380cd2807fc11ca8e829ae0ea34be8a8aeb96061
Deleted: sha256:b0bb848ce2bcf91d756a5bbfeed22868e2153b6c23438ea61b60efa1350fad49
Deleted: sha256:96ec31e33f5716ab300f41597750797f5dda638f98c8ff480e9ca65e2a27fcb8
Deleted: sha256:9845770646767ced47c3afc72bd32166ed3d992116728871f73d853524c35cf2
Deleted: sha256:9dbd58bfeb95ed8068d22d9c48b5c72f72f375b181369aaa12fd136ff55cab4b
Deleted: sha256:8b8b78d61a959def6dc399e0b7d1b61a86bdb3f40ed26e2ce2e404d60528d5ad
Deleted: sha256:46895b037bf203a08ad55ac8e4de05703262f22aa22cb8402ee91f66fb15195a
Deleted: sha256:5c83fc46f1b715d8cede1c70402d9ab19f1ae0246c35ef410f4d8ab2ac1bad14
Deleted: sha256:d7b2257a2277cf62be373f5e32a77274c67cdcdb474af84f2f01b95d8f6ebd0d
root@bitcamp-jenkins:~#
```

오후에 배포프로그램인 젠킨스 설치

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

CI/CD 의미(지속적인 통합과 배포)

<https://tecoble.techcourse.co.kr/post/2021-08-14-ci-cd/>

제킨스 이미지 다운및 실행

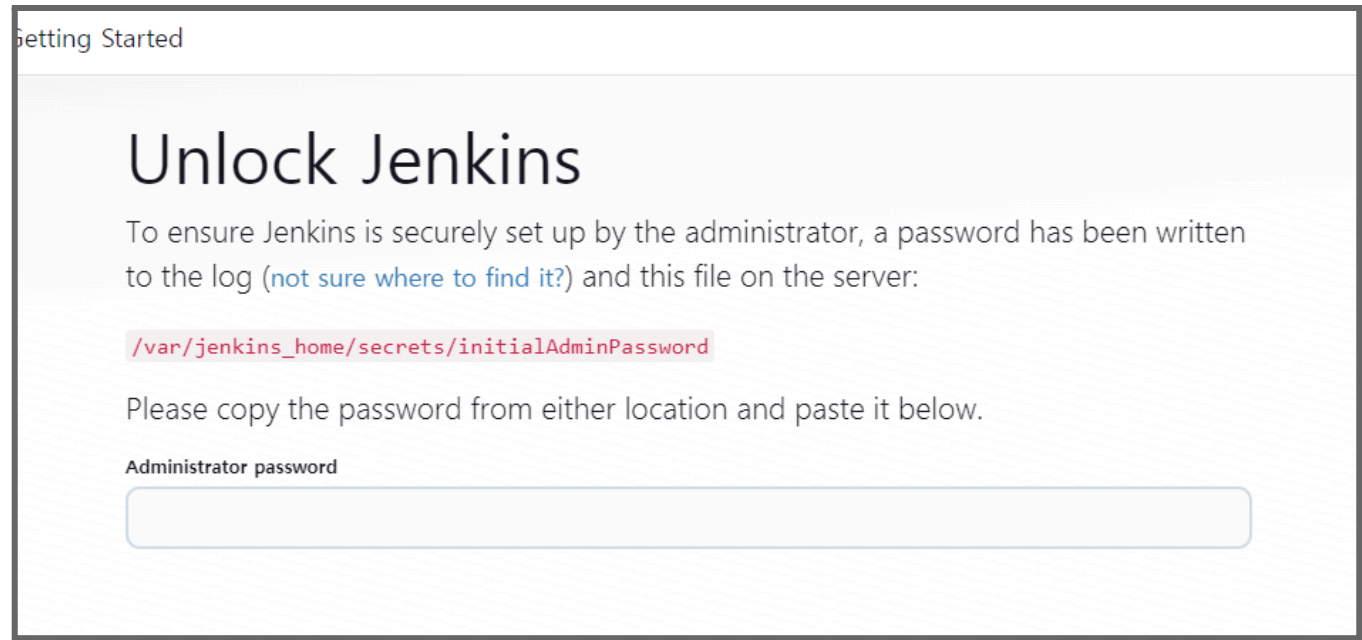
docker run -d -v jenkins_home:/var/jenkins_home -p 8080:8080 -p 50000:50000 --name jenkins-server --restart=on-failure jenkins/jenkins:lts-jdk17

```
root@bitcamp-jenkins:~# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
406132f43a22   jenkins/jenkins:lts-jdk17          "/usr/bin/tini -- /u..." 49 seconds ago Up 46 seconds 0.0.0.0:8080->8080/tcp, 0.0.0.0:50000->50000/tcp jenkins-server
root@bitcamp-jenkins:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
jenkins/jenkins  lts-jdk17  82a2134e1742   3 weeks ago   468MB
root@bitcamp-jenkins:~#
```

볼륨도 확인해보자

docker volume ls

211.188.62.241 : 8080 (각자 자기 서버 IP 로 8080)



제킨스 비번을 알아보는 방법

서버에서

docker logs jenkins-server


```
*****
*****
*****
jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:
480fcc067d744d1ab3b5de162fb9953
This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
*****
*****
*****
025-03-04 05:18:37.987+0000 [id=34] INFO jenkins.InitReactorRunner$1#onAttained:
```

비번을 복사후 젠킨스 초기화면에 넣어준다

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/jenkins_home/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Getting Started

<input checked="" type="checkbox"/> Folders	<input checked="" type="checkbox"/> OWASP Markup Formatter	<input type="checkbox"/> Build Timeout	<input type="checkbox"/> Credentials Binding	<div>** Icons API Folders OWASP Markup Formatter ** ASM API ** JSON Path API ** Structs ** Pipeline: Step API ** Token Macro</div>
<input type="checkbox"/> Timestampers	<input type="checkbox"/> Workspace Cleanup	<input type="checkbox"/> Ant	<input type="checkbox"/> Gradle	
<input type="checkbox"/> Pipeline	<input type="checkbox"/> GitHub Branch Source	<input type="checkbox"/> Pipeline: GitHub Groovy Libraries	<input type="checkbox"/> Pipeline Graph View	
<input type="checkbox"/> Git	<input type="checkbox"/> SSH Build Agents	<input type="checkbox"/> Matrix Authorization Strategy	<input type="checkbox"/> PAM Authentication	
<input type="checkbox"/> LDAP	<input type="checkbox"/> Email Extension	<input type="checkbox"/> Mailer	<input type="checkbox"/> Dark Theme	

이름 및 비밀번호를 모두 **admin** 으로 통일

Create First Admin User

계정명

admin

암호

.....

암호 확인

.....

이름

admin

이메일 주소

admin@admin.com



Instance Configuration

Jenkins URL:

http://211.188.62.241:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

설치 완료

+ 새로운 Item

빌드 기록

Jenkins 관리

My Views

빌드 대기 목록

빌드 대기 항목이 없습니다.

빌드 실행 상태

0/2

Jenkins 관리

Building on the built-in node can be a security issue. You should set up distributed builds. See [the document](#).

System Configuration

System

환경변수 및 경로 정보등을 설정합니다.

Tools

Configure tools, their locations and automatic installers.

Clouds

Add, remove, and configure cloud instances to provision agents on-demand.

Appearance

Configure the look and feel of Jenkins

Jenkins 관리 - Tools - add JDK

JDK installations

✓ Add JDK

☰ JDK

Name

! Required

JAVA_HOME

☐ Install automatically ?

Add JDK

java home 의 위치 알아보는 방법

docker inspect jenkins-server

```

"OpenStdin": false,
"StdinOnce": false,
"Env": [
  "PATH=/opt/java/openjdk/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin",
  "LANG=C.UTF-8",
  "JENKINS_HOME=/var/jenkins_home",
  "JENKINS_SLAVE_AGENT_PORT=50000",
  "REF=/usr/share/jenkins/ref",
  "JENKINS_VERSION=2.492.1",
  "JENKINS_UC=https://updates.jenkins.io",
  "JENKINS_UC_EXPERIMENTAL=https://updates.jenkins.io/experimental",
  "JENKINS_INCREMENTALS_REPO_MIRROR=https://repo.jenkins-ci.org/incrementals",
  "COPY_REFERENCE_FILE_LOG=/var/jenkins_home/copy_reference_file.log",
  "JAVA_HOME=/opt/java/openjdk"
],
"Cmd": null,
"Image": "jenkins/jenkins:lts-jdk17",
"Volumes": {
  "/var/jenkins_home": {}
}

```

Add JDK

JDK

Name

openjdk

JAVA_HOME

/opt/java/openjdk

☐ Install automatically ?

프로젝트 실행(간단한 테스트)

새로운 **Item** 클릭

New Item

Enter an item name

test1

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, for example, build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipeline (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 프로젝트

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment

≡ Execute shell ?

Command

See [the list of available environment variables](#)

```
echo "Jenkins Test!!!!"
```



Jenkins

Dashboard > test1 >



상태

test1



변경사항

고정링크



작업공간



지금 빌드



구성



Project 삭제



Rename

상태

test1

</> 변경사항

고정링크

작업공간

지금 빌드

구성

Project 삭제

Rename

Builds

...

Today

✓ #1 오전 5:48

▼

Dashboard > test1 > #1 > 콘솔 출력

상태

✓ 콘솔 출력

</> 바뀐점

Console Output

빌드 정보 수정

Delete build '#1'

Timings

Started by user [admin](#)

Running as SYSTEM

Building in workspace /var/jenkins_home/workspace/test1

[test1] \$ /bin/sh -xe /tmp/jenkins2154813262027559920.sh

+ echo Jenkins Test!!!!

Jenkins Test!!!!

Finished: SUCCESS

스프링 프로젝트 만들어서 **Git** 연동해놓기

jenkins관리 - Tools - Maven 설정

Maven installations

Add Maven

≡ Maven

Name

maven3.9

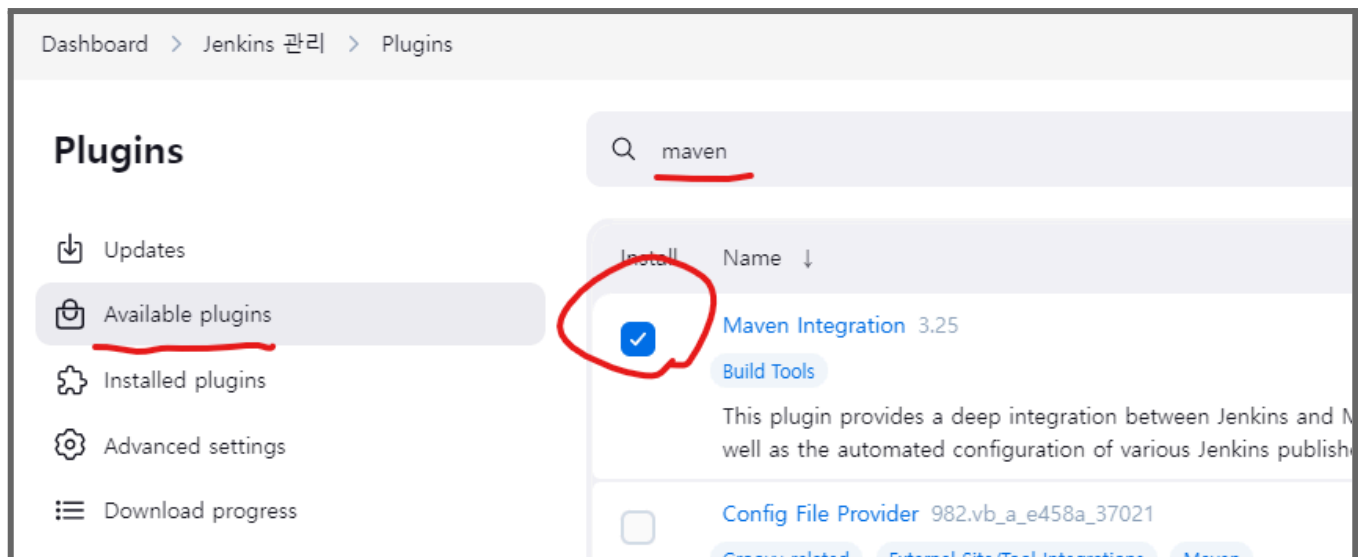
☒ Install automatically ?

≡ Install from Apache

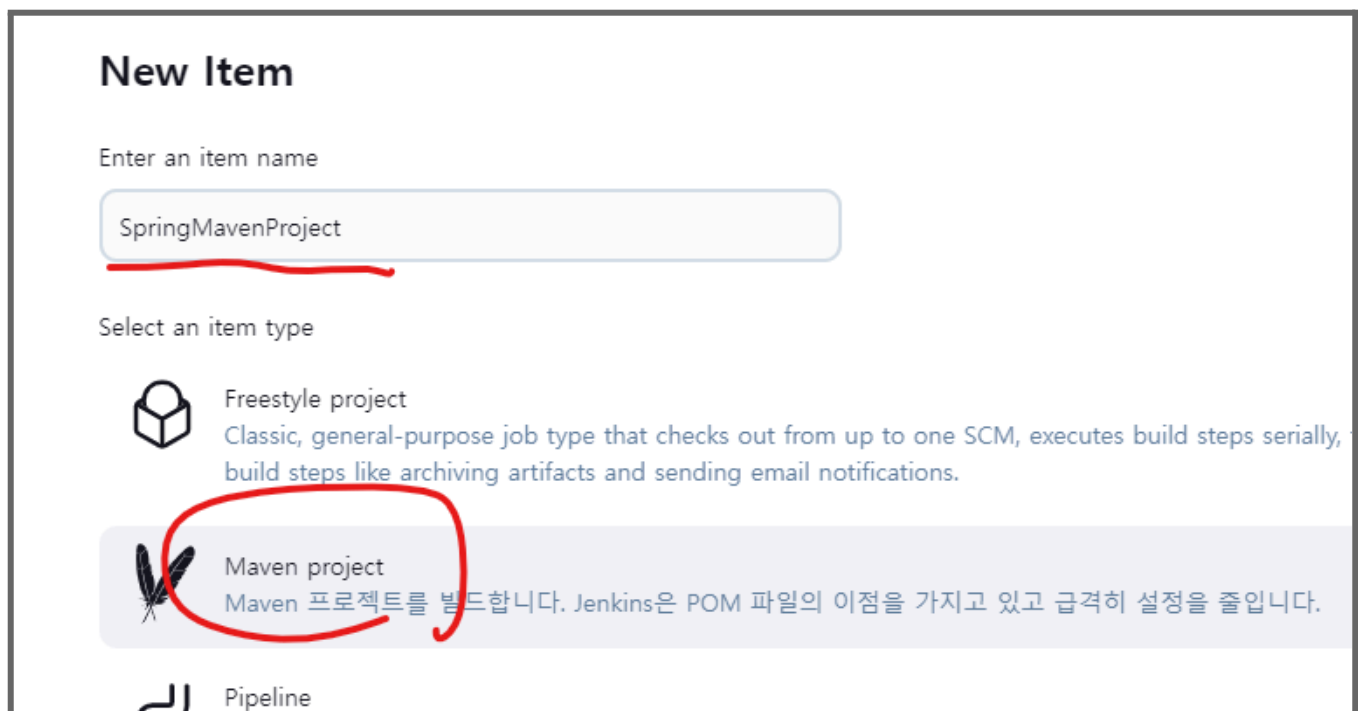
Version

3.9.9

Add Installer ▼



선택후 우측상단에 있는 **[install]** 클릭



Configure

General

General

소스 코드 관리

Triggers

Environment

설명

스프링 메이븐 프로젝트 배포 연습

Connect and manage your code repository to automatically pull the latest code for your builds.

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/zipsy327/mavenwar.git

Credentials ?

- none -

+ Add

고급 ▾

Add Repository

Branches to Build ?

Branch Specifier (blank for 'any') ?

*/main


Build

Root POM ?


pom.xml


Goals and options ?


clean compile package


 **Jenkins**


Dashboard > SpringMavenProject >


 상태

 변경사항

 작업공간

 지금 빌드

 구성

 Maven project 삭제

Maven project SpringMavenProject

스프링 메이븐 프로젝트 배포 연습

고정링크

Console Output 에서 **success** 나오면 일단은 빌드 성공

```
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/
[INFO] Replacing main artifact /var/jenkins_home/workspace/SpringMave
[INFO] The original artifact has been renamed to /var/jenkins_home/wo
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 11.559 s
[INFO] Finished at: 2025-03-04T06:35:32Z
[INFO] -----
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/jenkins_home/workspace/SpringMavenProject/po
[JENKINS] Archiving /var/jenkins_home/workspace/SpringMavenProject/ta
SNAPSHOT.war
channel stopped
Finished: SUCCESS
```

컴파일 성공후 생성된 **war** 파일 확인하기

Dashboard > SpringMavenProject > Workspace of SpringMavenProject on Built-In Node

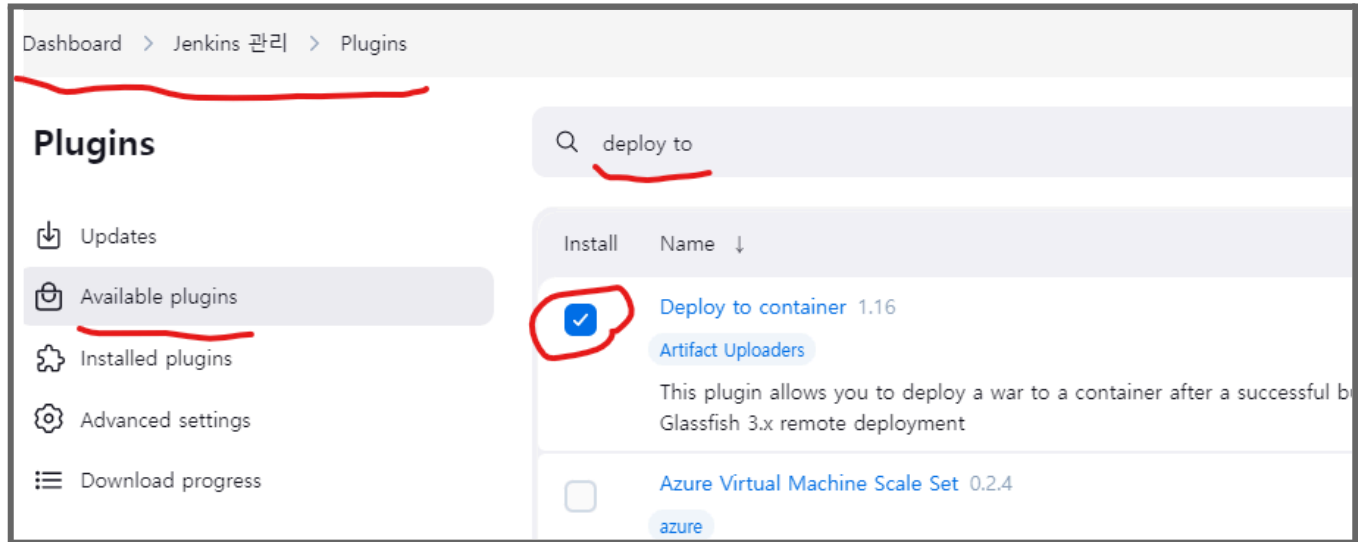
Workspace of SpringMavenProject on Built-In N

SpringMavenProject / target /

- classes
- generated-sources/annotations
- generated-test-sources/test-annotations
- maven-archiver
- maven-status/maven-compiler-plugin
- SpringMavenTest-0.0.1-SNAPSHOT
- surefire-reports
- test-classes/com/example/demo
- SpringMavenTest-0.0.1-SNAPSHOT.war ✓ 2025. 3. 4. 오전 6:35:32 21.55 Mi
- SpringMavenTest-0.0.1-SNAPSHOT.war.original 2025. 3. 4. 오전 6:35:31 15.59 Mi

위의 **war** 파일은 직접 실행할수는 없다
톰켓을 생성후 그 톰켓안에 **war** 파일을 넣어줘야 한다

docker 안에 **Tomcat** 을 설치하고 프로젝트 배포하기



서버에서 **Tomcat** 설치

docker run -d -it --name tomcat -p 8090:8080 tomcat:10

일단 톰캣이 정상 작동하는지 확인하기

<http://211.188.62.241:8090>

404 오류남



설정값을 변경해줘야 한다

토캣에 접속하는 방법

`docker exec -it tomcat bash`

```
root@bitcamp-jenkins:~# docker exec -it tomcat bash
root@f0ce1aadc8e8:/usr/local/tomcat# pwd
/usr/local/tomcat
root@f0ce1aadc8e8:/usr/local/tomcat# ls
bin          CONTRIBUTING.md  LICENSE        NOTICE        RUNNING.txt    webapps
BUILDING.txt  filtered-KEYS    logs           README.md      temp           webapps.dist
conf         lib             native-jni-lib  RELEASE-NOTES  upstream-KEYS  work
root@f0ce1aadc8e8:/usr/local/tomcat# ls webapps
root@f0ce1aadc8e8:/usr/local/tomcat# ls webapps.dist
docs  examples  host-manager  manager  ROOT
root@f0ce1aadc8e8:/usr/local/tomcat#
```

배포를 하기위해서는 토캣의 **webapps** 에 **war** 파일을 넣어야한다

그런데 **webapps** 에 있어야할 폴더및 파일들이 **webapps.dist** 에 들어있다

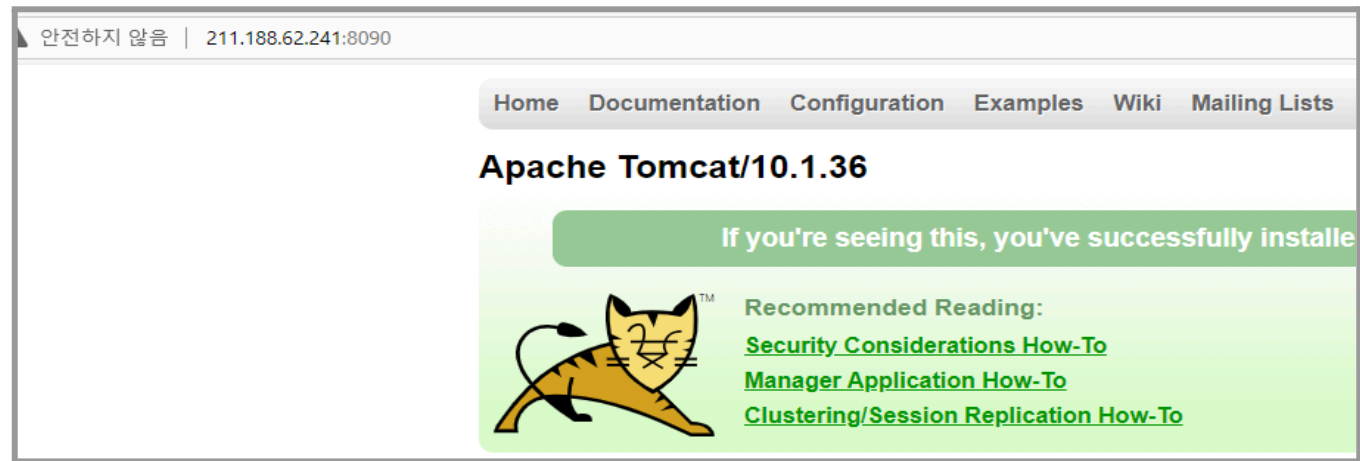
그곳의 모든것들을 **webapps** 로 복사해주어야 한다

`cp -R ./webapps.dist/* ./webapps`

(-R : 하위 디렉토리및 파일까지도 모두 복사)

```
root@f0ce1aadc8e8:/usr/local/tomcat# cp -R ./webapps.dist/* ./webapps
root@f0ce1aadc8e8:/usr/local/tomcat# ls webapps
docs  examples  host-manager  manager  ROOT
root@f0ce1aadc8e8:/usr/local/tomcat#
```

8090 포트로 다시 확인



webapps/manager/META-INF 로 이동

context.xml 내용 확인하기

cat context.xml

```
root@f0c1aadc8e8:/usr/local/tomcat/webapps/manager/META-INF# ls
context.xml
root@f0c1aadc8e8:/usr/local/tomcat/webapps/manager/META-INF# cat context.xml
<?xml version="1.0" encoding="UTF-8"?>
```

vi 에디터를 이용해서

vi context.xml 에 가서 보안에 관련된 Value 부분 주석처리할것

수정하고 싶은데 vi 가 설치 안되었음 설치하기

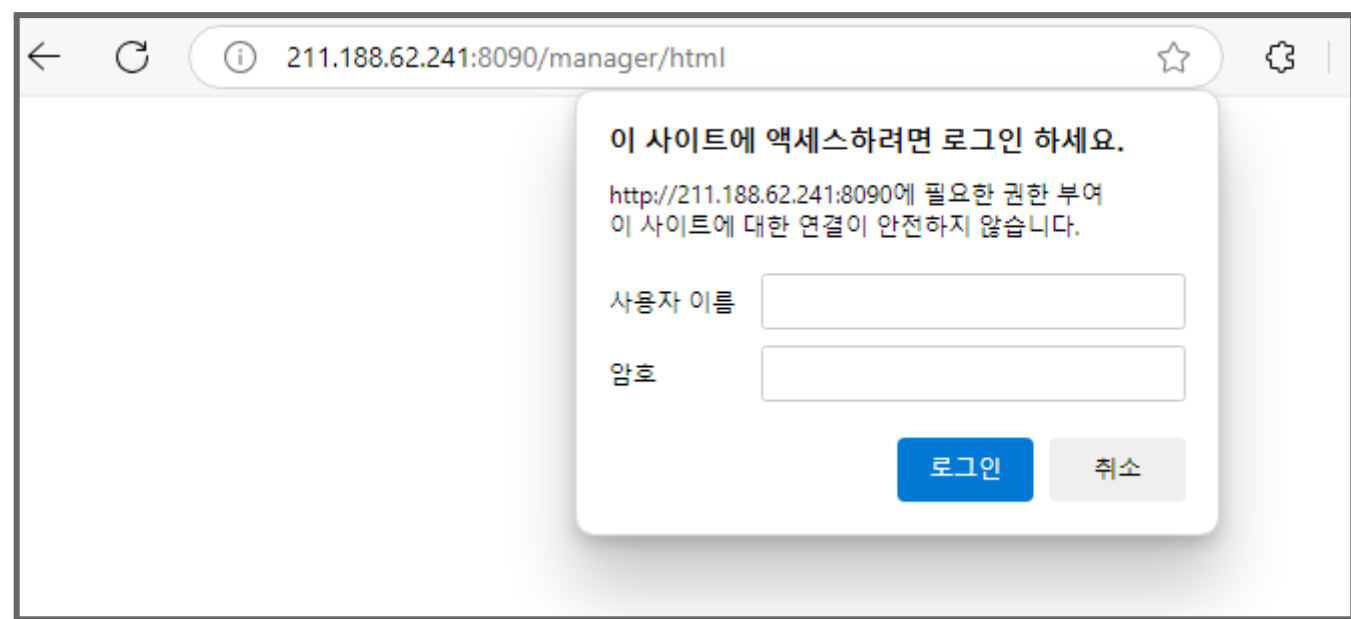
apt-get update

apt-get install vim

주석처리할 부분

```
<!-- <Valve className="org.apache.catalina.valves.RemoteAddrValve"
allow="127.\d+.\d+.\d+|::1|0:0:0:0:0:0:0:1" />-->
```

211.188.62.241:8090/manager/html 로 접속



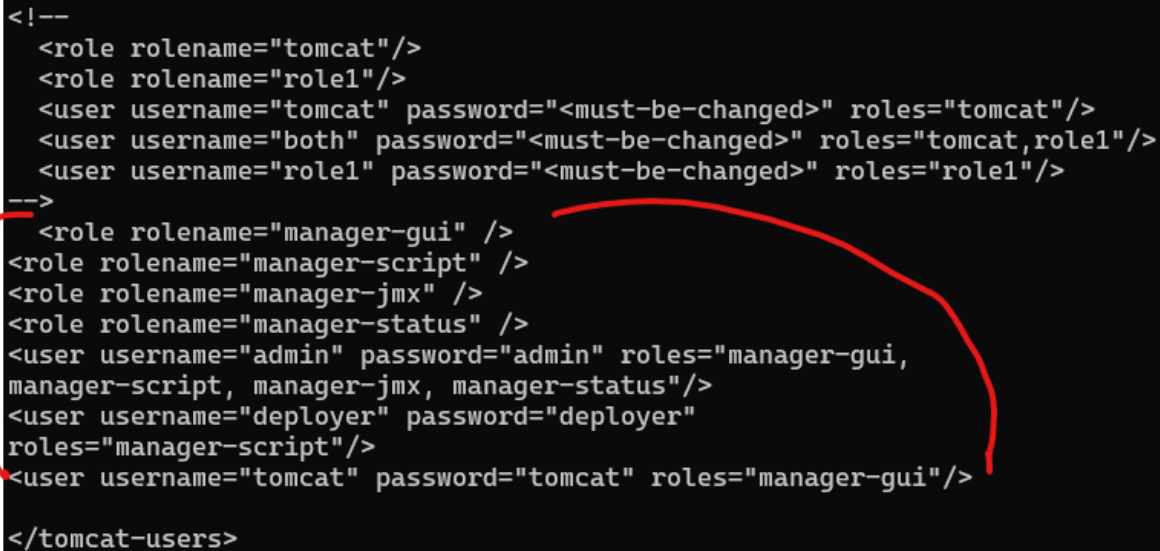
접속할 계정이 없음. 등록해야한다

tomcat/conf 로 이동

cat tomcat-users.xml : 안에 내용 확인하기

vi 로 들어가서 아래 내용을 추가

```
<role rolename="manager-gui" />
<role rolename="manager-script" />
<role rolename="manager-jmx" />
<role rolename="manager-status" />
<user username="admin" password="admin" roles="manager-gui,
manager-script, manager-jmx, manager-status"/>
<user username="deployer" password="deployer"
roles="manager-script"/>
<user username="tomcat" password="tomcat" roles="manager-gui"/>
```



```
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
  <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
  <user username="role1" password="<must-be-changed>" roles="role1"/>
-->
  <role rolename="manager-gui" />
  <role rolename="manager-script" />
  <role rolename="manager-jmx" />
  <role rolename="manager-status" />
  <user username="admin" password="admin" roles="manager-gui,
manager-script, manager-jmx, manager-status"/>
  <user username="deployer" password="deployer"
roles="manager-script"/>
  <user username="tomcat" password="tomcat" roles="manager-gui"/>
</tomcat-users>
```

admin/admin 으로 톰캣 접속

211.188.62.241:8090/manager/html

☆

⚙

이 사이트에 액세스하려면 로그인 하세요.

http://211.188.62.241:8090에 필요한 권한 부여 이 사이트에 대한 연결이 안전하지 않습니다.

사용자 이름

admin

암호

.....


로그인

취소

←

↺

⚠ 안전하지 않음 | 211.188.62.241:8090/manager/html



Tomcat 웹 애플리케이션 매니저

메시지:

OK

매니저

애플리케이션들의 목록을 표시

HTML 매니저 도움말

매니저 도...

애플리케이션들

경로	버전	표시 이름	실행 중	세션들	명령들
/	지정 안됨	Welcome to Tomcat	true	0	<div>시작 중지 다시 로드 배치된 것을 제거</div> <div>세션들을 만료시키기 idle 값 ≥ 30 분</div>
/docs	지정 안됨	Tomcat Documentation	true	0	<div>시작 중지 다시 로드 배치된 것을 제거</div> <div>세션들을 만료시키기 idle 값 ≥ 30 분</div>
/examples	지정 안됨	Servlet and JSP Examples	true	0	<div>시작 중지 다시 로드 배치된 것을 제거</div> <div>세션들을 만료시키기 idle 값 ≥ 30 분</div>
/host-manager	지정 안됨	Tomcat Host Manager Application	true	0	<div>시작 중지 다시 로드 배치된 것을 제거</div> <div>세션들을 만료시키기 idle 값 ≥ 30 분</div>
/manager	지정 안됨	Tomcat Manager Application	true	1	<div>시작 중지 다시 로드 배치된 것을 제거</div> <div>세션들을 만료시키기 idle 값 ≥ 30 분</div>

다시 젠킨스 설정

Dashboard - 배포프로젝트선택 - 구성

빌드 후 조치

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs

Deploy war/ear to a container

WAR/EAR files ?

**/*.war

Context path ?

mini

Context path 에 **mini** 를 넣으면 서버**ip:8090/mini** 하면 프로젝트 실행됨

Context path 에 **/** 를 넣으면 서버**ip:8090** 하면 프로젝트 실행됨

☰ Deploy war/ear to a container

WAR/EAR files ?

**/*.war

Context path ?

mini

Containers

☰ Tomcat 9.x Remote

Credentials

- none -

+ Add



Jenkins



Kind

Username with password

Scope ?

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

Username ?

deployer

☐ Treat username as secret ?

Password ?

.....

ID ?

deployer_user1

Description ?

deployer on VM

다시 선택



안전하지 않음 | 211.188.62.241:8090/mini/

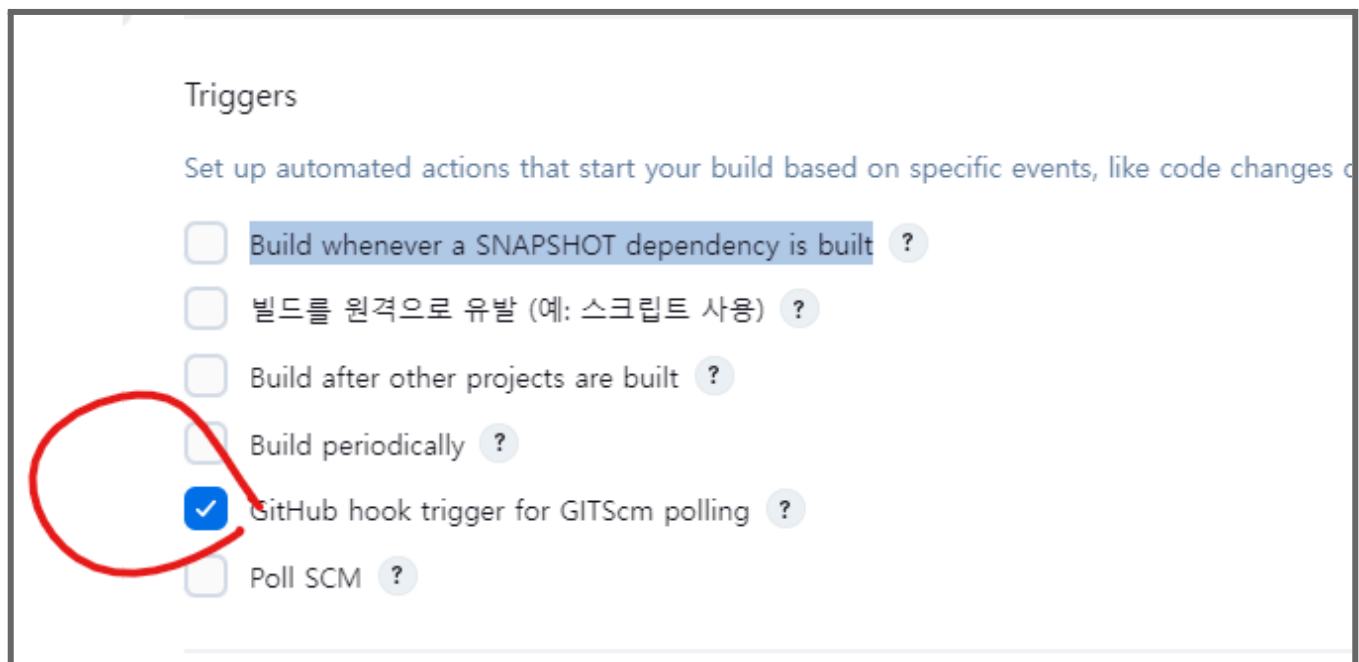
Spring Maven 프로젝트 배포 연습!!





Git push 하면 자동 빌드 시스템 구축

젠킨스 - 해당 프로젝트 - 구성

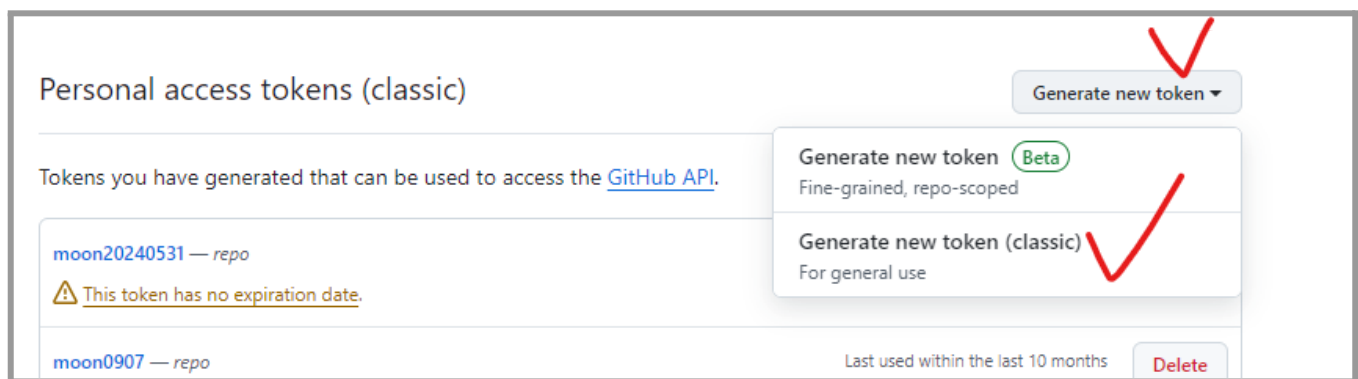


github - 배포프로젝트 -settings - Webhooks - Add webhook버튼

Payload URL **http://jenkins ip:port/github-webhook/**
Content type **application/json** 선택

Git Token 얻기

프로필 -Settings - Developer settins(맨아래쪽) - Personal access Tokens - Tokens(classic) -



Note

token20250305

What's this token for?

Expiration *



No expiration ▾

The token will never expire!

GitHub strongly recommends that you set an expiration date for your token to help keep your information secure.

[Learn more](#)

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

- | | |
|---|--------------------------------------|
| <input checked="" type="checkbox"/> repo | Full control of private repositories |
| <input checked="" type="checkbox"/> repo:status | Access commit status |
| <input checked="" type="checkbox"/> repo_deployment | Access deployment status |
| <input checked="" type="checkbox"/> public_repo | Access public repositories |
| <input checked="" type="checkbox"/> repo:invite | Access repository invitations |
| <input checked="" type="checkbox"/> security_events | Read and write security events |