



섹션 7. 생애 최초 배포하기

이번 섹션의 목표

43강. EC2에 접속해 리눅스 명령어 다뤄보기

44강. 배포를 위한 프로그램 설치하기

45강. 빌드와 실행, 그리고 접속

46강. 종료되지 않는 실행

47강. 가비아를 이용한 도메인 구입, DNS 적용

48강. Section 7 정리

이번 섹션의 목표

1. EC2에 접속하는 방법을 알아보고, EC2에 접속해 리눅스 명령어를 다뤄본다.
2. 개발한 서버의 배포를 위해 환경 세팅을 리눅스에서 진행하고, 실제 배포를 진행한다.
3. foreground와 background의 차이를 이해하고 background 서버를 제어한다.
4. 도메인 이름을 사용해 사용자가 IP 대신 이름으로 접속할 수 있도록 한다.

43강. EC2에 접속해 리눅스 명령어 다뤄보기

이번 시간에는 지난 시간에 우리가 빌렸던 컴퓨터에 접속해, 여러 명령어를 다뤄보자. 우리가 가장 익숙한 환경은 마우스를 활용하는 Windows 환경이지만, 대부분의 서버는 Linux 환경에 배포되며 Linux에서는 마우스를 사용하지 않는다!!! 대신 명령어를 통해 폴더를 만들거나, 파일을 만들고 옮기거나 다양한 동작을 수행할 수 있다.

자 이제 우리가 빌린 컴퓨터에 접속해 볼 예정인데, EC2에 접속할 때는 2가지 방법이 있다. 우리가 다운로드 받은 키 페어 (pem키)를 활용해 접속하는 방법과 AWS 콘솔을 활용해 접속하는 방법이다.

먼저 첫 번째 방법부터 살펴보자. EC2에 접속하기 위해서는 3가지 준비물이 필요하다.

1. 우리가 접속하려 하는 EC2의 IP 주소
2. 이전 시간에 다운로드 받았던 키 페어 (pem키)

3. 접속하기 위한 프로그램

- a. Windows - git CLI (git bash)
- b. Mac - terminal

이 3가지이다.

가장 먼저 접속하려 하는 EC2의 IP 주소는 AWS 인스턴스 대시보드에서 특정 인스턴스(= 컴퓨터)를 클릭하면 아래에 정보가 나오게 된다.

The screenshot shows the AWS Management Console interface for an EC2 instance. At the top, there's a search bar and a filter set to '인스턴스 상태 = running'. Below this is a table with columns: Name, 인스턴스 ID, 인스턴스 상태, 인스턴스 유형, and others. The instance 'Library App Server' with ID 'i-01a01f13449d94919' is listed with a status of '실행 중' (Running). Below the table, the details for this instance are shown. A red arrow points to the '퍼블릭 IPv4 주소' (Public IPv4 address) field, which displays '43.201.84.217'.

Name	인스턴스 ID	인스턴스 상태	인스턴스 유형
Library App Server	i-01a01f13449d94919	실행 중	t2.micro

인스턴스: i-01a01f13449d94919(Library App Server)

세부 정보 | 보안 | 네트워킹 | 스토리지 | 상태 검사 | 모니터링 | 태그

▼ 인스턴스 요약 정보

인스턴스 ID	퍼블릭 IPv4 주소
i-01a01f13449d94919 (Library App Server)	43.201.84.217 개방 주소법

다음으로 키 페어는 잘 보이는 곳에 위치시키자. 그런 다음, git CLI 혹은 Terminal을 열어 천천히 다음 명령어를 타이핑한다.

```
ssh -i
```

여기까지 타이핑했다면, 키페어를 드래그해서 명령창으로 가져오자! 그럼 다음과 같이 키페어의 전체 경로가 입력된다.

```
ssh -i 경로1/경로2/키페어이름.pem
```

그 다음 이제 ec2-user@IP 주소를 입력해 주자. 모두 입력하면 다음과 같다.

```
ssh -i 경로1/경로2/키페어이름.pem ec2-user@43.201.84.217
```

이제 엔터를 누르면, 접속을 원하는지 다시 한번 물어보고, yes를 입력하면 접속이 된다.

```
The authenticity of host '43.201.84.217 (43.201.84.217)' can't be established.  
ED25519 key fingerprint is SHA256:w5t4x9QoAjBX0Bptk1WpsN9zBNKPmc4Bic0SLL/BcMo.  
This key is not known by any other names  
Are you sure you want to continue connecting (yes/no/[fingerprint])? █
```

처음으로 위의 명령어를 타이핑했다면, 아래와 같은 에러가 나올 수 있다.

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  
@      WARNING: UNPROTECTED PRIVATE KEY FILE!      @  
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  
Permissions 0777 for '/Users/lannstark/Desktop/studying-developer.pem' are too open.  
It is required that your private key files are NOT accessible by others.  
This private key will be ignored.  
Load key "/Users/lannstark/Desktop/studying-developer.pem": bad permissions  
ec2-user@43.201.84.217: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
```

이 에러는 키페어 파일의 접근 권한을 보강하라는 에러이다.

```
chmod 400 경로1/경로2/키페어이름.pem
```

그럼 위와 같은 명령어를 사용해 주자. 이 명령어를 통해 키페어 파일의 접근 권한을 보강할 수 있다.

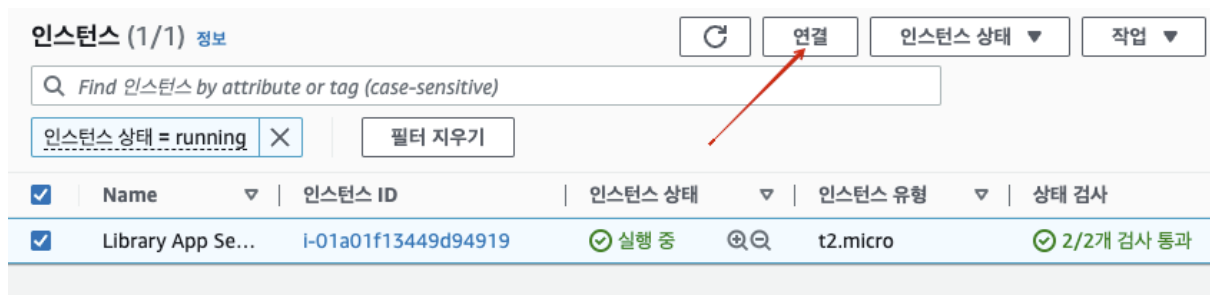
자 이제 다시 한번 접속 명령어를 사용해 주자.

```
ssh -i 경로1/경로2/키페어이름.pem ec2-user@43.201.84.217
```

접속에 성공하면, 이렇게 EC2 Amazon Linux 2 AMI라고 나오며 리눅스 환경에 들어간다. 검정 배경의 흰 글씨일 뿐이지만, 우리가 원격 컴퓨터에 접근하고 있는 것이다!

```
--|  __|_ )  
_! (  /  Amazon Linux 2 AMI  
---|\___|  
  
https://aws.amazon.com/amazon-linux-2/  
24 package(s) needed for security, out of 33 available  
Run "sudo yum update" to apply all updates.  
[ec2-user@ip-172-31-4-234 ~]$
```

두 번째 방법도 알아보자! AWS 콘솔 환경에서 우리가 빌린 컴퓨터를 선택하고, '연결'을 누른다.



그런 다음 별다른 설정을 하지 않고 인스턴스에 연결 > 연결을 한 번 더 누른다!

EC2 > 인스턴스 > i-01a01f13449d94919 > 인스턴스에 연결

인스턴스에 연결 정보

다음 옵션 중 하나를 사용하여 인스턴스 i-01a01f13449d94919 (Library App Server)에 연결

EC2 인스턴스 연결

Session Manager

SSH 클라이언트

EC2 직렬 콘솔

인스턴스 ID

i-01a01f13449d94919 (Library App Server)

퍼블릭 IP 주소

43.201.84.217

사용자 이름

ec2-user

사용자 지정 사용자 이름을 사용하여 연결하거나 인스턴스 시작에 사용한 AMI의 기본 사용자 이름 ec2-user를(를) 사용합니다.

참고: 대부분의 경우 추정된 사용자 이름은 정확합니다. 하지만 AMI 사용 지침을 읽고 AMI 소유자가 기본 AMI 사용자 이름을 변경했는지 확인하십시오.

취소

연결

그러면 화면이 이동하며 우리가 빌린 컴퓨터로 바로 접속된다!

```
 _ _ | ( _ _ ) Amazon Linux 2 AMI
 _ _ | \ _ _ | _ _ |

https://aws.amazon.com/amazon-linux-2/
24 package(s) needed for security, out of 33 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-4-234 ~]$
```

자 이제 리눅스 컴퓨터에 접근하였으니, 가장 중요한 5가지 명령어를 알아보자.

첫 번째 명령어는 폴더를 만드는 명령어 make directory의 약자 `mkdir` 이다.

```
# mkdir 폴더이름
mkdir folder1
```

위 명령어를 타이핑하면 아직 눈에 보이지는 않지만, 폴더가 생성되었다!! 두 번째 명령어로 폴더를 확인해 보자.

두 번째 명령어는 현재 위치에서 폴더나 파일을 확인하는 list segments의 약자 `ls` 이다.

```
ls
```

`ls` 를 타이핑 하니 우리가 만들었던 folder1을 확인할 수 있다! 🍀

만약 폴더의 자세한 정보를 확인하고 싶다면 `ls -l` 을 사용할 수 있다! 이때 `ls` 를 명령어 `-l` 을 l 옵션이라고 부른다.

```
ls -l
drwxrwxr-x 2 ec2-user ec2-user 6 Dec  5 06:00 folder1
```

이때 각 항목의 의미는 다음과 같다.

- `drwxrwxr-x` : 이 폴더에 접근할 수 있는 권한을 의미한다. 앞에 'd'가 폴더라는 의미이고 그다음은 3글자씩 끊어서 봐야 한다. 이때 r은 읽을 수 있는 권한, w는 쓸 수 있는 권한, x는 실행할 수 있는 권한을 의미한다. `rwX / rwx / r-x`로 봤을 때 첫 번째 3개는 이 폴더의 주인의 권한을 표시했고, 다음 3개는 이 폴더의 소유 그룹의 권한, 마지막 3개는 아무나 접근했을 때의 권한을 의미한다.
 - 우리가 아까 사용했던 `chmod` 명령어가 바로 접근 권한을 변경하는 명령어이다.
- `2` : 이 폴더에 걸려 있는 바로 가기 개수를 의미한다.
- `ec2-user` : 이 폴더의 주인 이름이다.
- `ec2-user` : 이 폴더의 소유 그룹 이름이다.
- `6` : 파일의 크기를 의미한다. 보통 byte 단위이다.
- `Dec 5 06:00` : 파일의 최종 변경 시각이다.
- `folder1` : 파일 이름을 의미한다.

매우 좋다~! 🍀 지금은 어렵게 느껴질 수 있지만,

이제 폴더를 2개 더 만들어보자.

```
mkdir folder2
mkdir folder3
```

이 3개의 폴더 중 folder2 안으로 들어갈 것이다. 폴더에 들어가는 linux 명령어는 change directory의 약자 `cd` 이다.

```
# folder2 안으로 들어간다.  
cd folder2
```

자 위의 명령어를 입력하면 folder2 안으로 들어왔는데, 크게 달라진 것은 없어 보인다. 이럴 때 나의 현재 위치를 확인하기 위해 print working directory의 약자 `pwd` 를 타이핑할 수 있다.

```
pwd
```

그러면 현재 나의 위치가 나오고, folder2 안에 있는 것을 확인할 수 있다!! 🎉

다음으로 다시 folder2 바깥으로 나가보자. 폴더를 이동하는 명령어는 `cd` 인데 현재 위치에서 상위 폴더로 가려면 `cd ..` 를 타이핑해 주어야 한다. 여기서 `..` 은 상위 폴더를 의미한다.

```
cd ..
```

그리고 다시 `pwd`와 `ls`를 타이핑해 보면, 원래 있던 홈 디렉토리로 잘 이동한 것을 확인할 수 있다.

오늘 살펴볼 마지막 명령어는 `rmdir` 이다. 이 명령어는 비어 있는 디렉토리를 제거할 수 있도록 해주는 명령어이다. folder3을 제거하기 위해서는

```
rmdir folder3
```

을 타이핑하면 된다. 위 명령어를 입력하고 `ls` 를 타이핑하면, folder1과 folder2만 남아 있는 것을 확인할 수 있다.

지금까지 살펴보았던 명령어를 요약하면 다음과 같다.

- `mkdir` : 폴더를 만드는 명령어

- `rmdir` : 폴더를 제거하는 명령어
- `cd` : 현재 위치를 변경하는 명령어
 - `cd ..` : 상위 폴더로 이동한다.
- `pwd` : 현재 위치를 확인하는 명령어
- `ls` : 현재 위치에 있는 폴더와 파일을 확인하는 명령어
 - `ls -l` : 조금 더 자세한 정보를 표시할 수 있다.

매우 좋다~ 🍌 이제 기본적인 리눅스 명령어를 살펴보았으니, 다음 시간에는 우리의 도서 관리 애플리케이션 배포를 위한 프로그램들을 설치해 보자.

44강. 배포를 위한 프로그램 설치하기

이번 시간에는 리눅스에서 배포를 위해 필요한 프로그램을 설치할 예정이다.

설치해야 하는 3가지 프로그램 목록은 다음과 같다.

- 코드를 가져오기 위한 git
- 우리가 만든 서버를 구동할 java
- 데이터베이스의 역할을 할 mysql

자 지난 시간에 했던 것처럼 EC2에 접속하자.

가장 먼저 타이핑할 명령어는,

```
sudo yum update
```

이다. `sudo` 라는 가장 앞에 있는 구문은 관리자의 권한으로 실행한다는 의미이고, `yum update` 에서 yum은 리눅스의 패키지 관리 프로그램이다. gradle과 비슷한 역할을 하고 있다고 생각하면 된다. `yum update` 를 하게 되면 현재 설치되어 있는 여러 프로그램들을 자동으로 최신화해준다.

매우 좋다~! 🍌 이제 git을 설치하자!

```
sudo yum install git
```


아까와 비슷하게 yum과 관리자 권한을 이용해 git을 설치해 준다.

업데이트와 비슷하게 중간에 yes / no를 물어보는데 이렇게 물어보는 것이 번거롭다면,

```
sudo yum install git -y
```

처럼 끝에 `-y`를 붙여줌으로써 yes를 입력한 것과 같은 효과를 낼 수 있다.

잘 설치되었는지 아래 명령어를 통해 확인해 주자!

```
git
```

만약 설치되었다면, git의 명령어들 목록이 나오게 된다.

다음은 java 설치이다. 우리가 사용한 java버전은 11버전이기에 11버전을 설치해 주면 된다.

```
sudo yum install java-11-amazon-corretto -y
```

크게 어려울 것은 없다! 설치가 다 되었다면,

```
java -version
```

명령어를 통해 잘 설치되었는지 확인할 수 있다.

마지막으로는 mysql을 설치해야 한다.

```
# mysql 8.0을 설치할 수 있도록 설정한다.  
wget https://dev.mysql.com/get/mysql80-community-release-el7-5.noarch.rpm  
sudo rpm -ivh mysql80-community-release-el7-5.noarch.rpm  
  
# mysql을 설치한다.  
sudo yum install mysql-community-server -y
```

mysql이 잘 설치되었다면, 다음과 같은 명령어로 확인해 보자!

```
sudo systemctl status mysqld
```

출력 결과

```
• mysqld.service - MySQL Server
  Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
  Active: inactive (dead)
  Docs: man:mysqld(8)
        http://dev.mysql.com/doc/refman/en/using-systemd.html
```

현재 **Active: inactive** 라고 나온다면 mysql의 설치는 잘 되었고 활성화되지 않은 상태라는 의미이다.

```
sudo systemctl restart mysqld
```

를 이용해 활성화해 주자!!

다음으로 우리가 설치한 mysql에 접근할 건데,

```
sudo cat /var/log/mysqld.log | grep "A temporary password"
```

출력 결과

```
2022-12-12T12:50:04.931422Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: pEnqZLA7qc.J
```

위의 명령어를 사용해 현재 설정되어 있는 임시 비밀번호를 알아오자. 다음으로

```
mysql -u root -p
```

라는 명령어를 타이핑하고, 비밀번호에 아까 확인했던 임시 비밀번호는 입력 혹은 복사 붙여넣기 해주자.

매우 좋다~!!! 😊 우리는 서버 배포에 필요한 모든 프로그램을 설치하였다!

mysql 같은 경우에는 데이터베이스와 테이블도 필요하므로 바로 설정해 주도록 하자.

```
create database library;

create table user(
  id bigint auto_increment,
  name varchar(25),
  age int,
```

```

    primary key (id)
);

create table book(
    id bigint auto_increment,
    name varchar(255),
    primary key (id)
);

create table user_loan_history (
    id bigint auto_increment,
    user_id bigint,
    book_name varchar(255),
    is_return tinyint(1),
    primary key (id)
);

```

자 다음으로 한 가지 작업을 더 해주어야 한다! 바로 1) 임시 비밀번호를 다른 비밀번호로 변경하고, 2) 우리의 application.yml에 그 비밀번호를 설정해 주는 일이다!!

먼저 현재 root 계정의 임시 비밀번호를 변경해 보자.

```

ALTER user 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '변경하고 싶은 비밀번호';

```

이때 비밀번호는 8자리 이상, 대문자, 소문자, 숫자, 특수문자를 포함해야 한다. **Abcd1234!** 로 변경해 보겠다.

```

ALTER user 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'Abcd1234!';

```

다음으로는 우리가 EC2에 설치한 mysql에 spring이 접속할 수 있도록 방금 확인한 기본 비밀번호를 **application.yml**에 설정해 주자!

그래야 우리가 살펴보았던 것처럼 우리 컴퓨터에서 **local** profile로 실행될 때는 H2 DB를, EC2에서 **dev** profile로 실행될 때는 EC2에 있는 mysql을 정상적으로 쓸 수 있을 것이다!



```

spring:
  config:
    activate:
      on-profile: dev
  datasource:
    url: "jdbc:mysql://localhost/library"

```

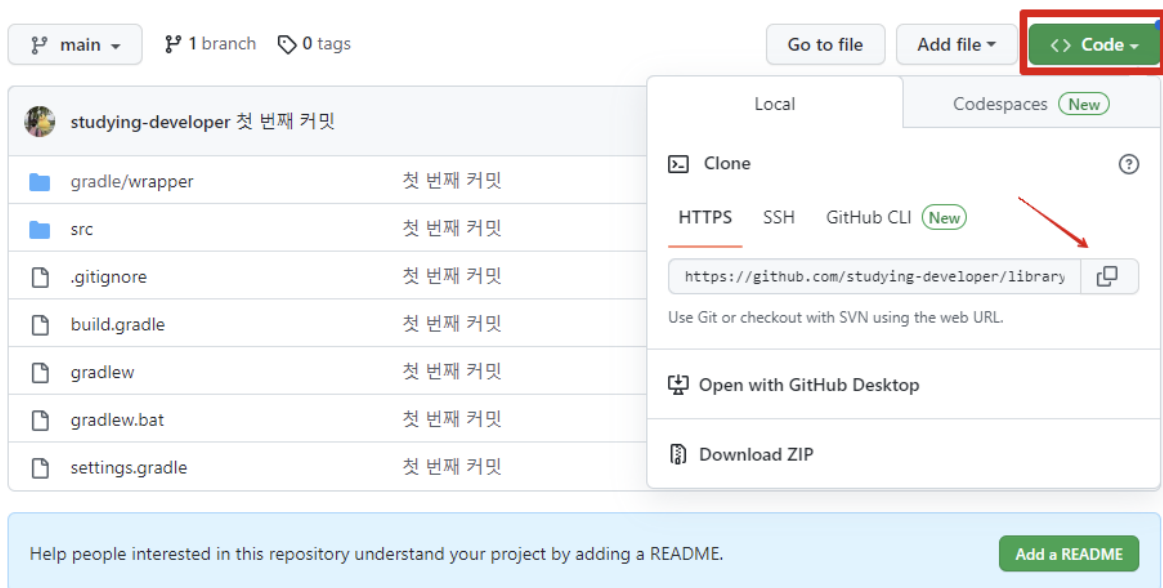
```
username: "root"
password: "Abcd1234!" ### <-- 여기에 변경한 비밀번호를 채워주자!
driver-class-name: com.mysql.cj.jdbc.Driver
```

이렇게 기본 비밀번호를 입력했다면, 다시 한번 git을 이용해 원격 저장소에 최신 코드를 반영해 주자! 😊

이제, 다음 시간에는 git을 이용해 우리의 코드를 가져와 빌드와 실행을 거쳐 도서 관리 애플리케이션에 접속해 볼 것이다!! 🔥

45강. 빌드와 실행, 그리고 접속

이번 시간에는 드디어 EC2에서 우리의 서버를 실행시키고, 우리가 만든 서버에 접속해 다양한 기능을 사용해 볼 예정이다. 가장 먼저 지난 Section에서 살펴보았던 것처럼 코드를 가져와야 한다. 이제 git을 사용할 차례이고, 이때 github 저장소 사이트에 들어가 `code` 클릭 > 주소가 필요하다.



```
git clone [github 저장소 주소]
git clone https://github.com/studying-developer/library-app.git
```

`git clone` 명령어는 주소로 주어진 저장소에 있는 코드들을 현재 위치로 복사하는 명령어이다.

위 명령어를 타이핑하고, `ls` 명령어를 쳐보면 `library-app` 폴더가 생긴 것을 확인할 수 있다.

매우 좋다~! 👍 이제 `library-app` 폴더에 들어가 우리가 개발한 프로젝트를 빌드하고 실행해 볼 것이다.

자 그전에!! 현재 우리가 사용하고 있는 AWS EC2 무료 컴퓨터는 RAM이 1GB로 무척 작아 빌드나 실행을 진행하다 컴퓨터가 멈출 수 있다..! 😭 따라서 해결책으로 다음 명령어들을 사용해서 부족한 메모리 대신 Disk를 일부 사용할 수 있도록 처리해야 한다.

어떤 프로그램이건, 프로그램이 실행되면 기본적으로 메모리를 사용하는데 이때 메모리가 부족하면 디스크를 대신 사용하게 해주는 설정이다!

```
# swap 메모리를 할당한다 (128M * 16 = 2GB)
sudo dd if=/dev/zero of=/swapfile bs=128M count=16

# 스왑 파일에 대한 권한 업데이트
sudo chmod 600 /swapfile

# swap 영역 설정
sudo mkswap /swapfile

# swap 파일을 사용할 수 있도록 만든다.
sudo swapon /swapfile

# swap 성공 확인
sudo swapon -s
```

매우 좋다~! 👍 이제 본격적으로 우리의 프로젝트를 빌드하고 실행시키도록 하자!

빌드를 위해서는 `gradle` 명령어를 사용해야 한다. 다음 두 명령어를 차례로 입력해주자.

```
chmod +x ./gradlew # ./gradlew를 사용하기 위해 실행할 수 있도록 설정한다.

# gradle을 이용해 우리의 프로젝트를 빌드한다. 이때 테스트를 돌리지 않는다.
./gradlew build -x test
```

일반적으로는 `./gradlew build` 명령어를 사용하기도 하는데, 이 명령어는 테스트 단계를 포함해 프로젝트를 빌드하는 명령어이다. 현재 우리는 1) 테스트 코드를 작성하지는 않았고 2) 무료 서버이다 보니 테스트까지 돌릴 수 있는 사양이 되지 않아 테스트를 제외하고 빌드하는 명령어를 수행했다.

빌드 과정이 모두 끝이 나면, `ls` 명령어를 타이핑했을 때 `build` 폴더가 새로 생긴 것을 확인할 수 있다.

```
# ./gradlew build 전의 폴더 구조
build.gradle
gradle
gradlew
gradlew.bat
settings.gradle
src

# ./gradlew build 후의 폴더 구조
build # build 폴더가 추가되었다!
build.gradle
gradle
gradlew
gradlew.bat
settings.gradle
src
```

이 `build/libs/` 안에 우리가 개발한 소스코드를 뭉쳐 놓은 아티팩트, jar 파일이 존재하게 된다!! 👍

매우 좋다~!! 😊 이제 빌드된 jar 파일을 이용해 서버를 실행시켜보자.

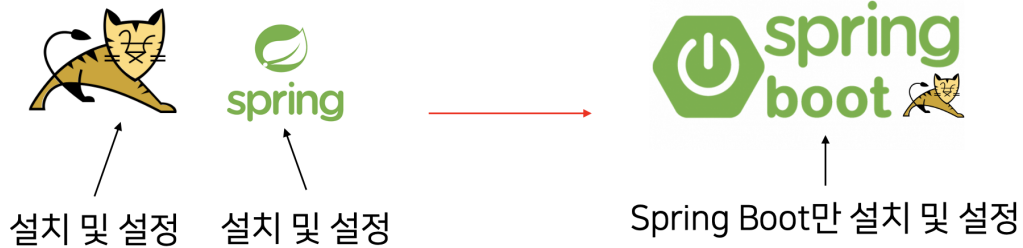
jar 파일을 실행시켜줄 때 <38강. profile과 H2 DB>에서 다루었던 것처럼 dev profile을 이용해 실행시켜 주어야 한다. 명령어는 다음과 같다.

```
java -jar build/libs/library-app-0.0.1-SNAPSHOT.jar --spring.profiles.active=dev
```

너무 간단하다! 소스코드를 뭉쳐주는 명령어를 한 번 쳐주고~ 뭉쳐진 jar 파일을 실행시키는 명령어를 한 번 쳐주니~ 이렇게 서버가 바로 동작하게 되었다.

이것이 바로 <1강. 스프링 프로젝트를 시작하는 두 번째 방법>에서 언급했던 Spring Boot 부터는 톰캣이 내장되어 가능한 일이다.

톰캣은 웹 애플리케이션 서버(Web Application Server, WAS)의 한 종류로, 요청이 들어오면 그 요청을 약속된 형식에 맞추어 스프링에 전달해 주는 역할을 수행한다. 예전에는 스프링과 톰캣이 분리되어 있었기 때문에, 스프링을 배포하려고 하면 톰캣도 다운로드 받아 설정해 주고 스프링에도 추가적인 설정을 해주어야 했다. 하지만 지금은 ‘톰캣이 내장된 덕분에’ 이렇게 jar 파일만 실행시키면 서버가 동작하는 것이다!



매우 좋다~!! 이제 EC2의 IP를 이용해 웹 UI에 접근해 보도록 하자.

인스턴스 (1/1) 정보

Find 인스턴스 by attribute or tag (case-sensitive)

인스턴스 상태 = running 필터 지우기

<input checked="" type="checkbox"/>	Name	인스턴스 ID	인스턴스 상태	인스턴스 유형	상
<input checked="" type="checkbox"/>	Library App Server	i-01a01f13449d94919	실행 중	t2.micro	

인스턴스: i-01a01f13449d94919(Library App Server)

세부 정보 보안 네트워킹 스토리지 상태 검사 **모니터링** 태그

▼ 인스턴스 요약 정보

인스턴스 ID i-01a01f13449d94919 (Library App Server)	퍼블릭 IPv4 주소 43.201.84.217 개방 주소법
---	---

크롬이나 다른 인터넷 브라우저를 열어 `http://IP주소:8080/v1/index.html` 을 입력하면 된다.
예를 들어, `http://43.201.84.217:8080/v1/index.html` 이다.

그럼 우리가 개발한 <도서 관리 애플리케이션>이 보이고, 모든 기능이 정상적으로 동작하는 것을 알 수 있다!!

자 이제 현재 동작하고 있는 서버를 종료시켜보자. 서버를 종료시키기 위해서는 `ctrl + c` 를 활용할 수 있다. 이 `ctrl + c` 는 꼭 스프링 서버가 아니라 무언가를 중단하는 신호이다.

그리고 현재 빌드 되어 있는 결과물을 제거하기 위한 명령어는 `./gradlew clean` 이다. 혹시나 코드의 변경이 있어 빌드를 다시 하고 싶다면 `./gradlew clean` 을 활용 해보도록 하자!

너무 좋다~!! 😊 자 그런데 사실 지금 방식에는 한 가지 문제가 존재한다. 바로 우리가 EC2와의 접속을 끊으면 서버도 함께 종료된다는 점이다. 다음 시간에는 이 문제를 해결해 보도록 하자! 🔥

46강. 종료되지 않는 실행

이번 시간에는 우리가 AWS EC2 접속을 종료하더라도 서버가 계속해서 실행되고 있을 수 있게 처리해 보도록 하자. 그러기 위해서 우리는 'foreground'와 'background'에 대해서 간단히 짚고 넘어가야 한다! 이들은 굉장히 쉬운 개념이다!

- foreground
 - 우리가 보고 있는 프로그램을 의미한다. 예를 들어 문서를 보고 있다면 PDF 프로그램이 foreground 프로그램이고, 인프런 강의를 보고 있다면, 브라우저가 foreground 프로그램이다.
- background
 - 우리가 보고 있지 않은데 실행 중인 프로그램을 의미한다. 대표적으로 컴퓨터에서 조용히 돌아가고 있는 백신을 background 프로그램이라고 할 수 있다. 또, 음악을 틀어두고 게임을 하고 있다면 이 경우 음악 재생 프로그램이 background 프로그램이 된다.

좋다~! 우리가 EC2 접속을 종료할 때 서버가 같이 종료되었던 이유는 서버가 foreground로 동작했기 때문이다. 그러니 background로 동작할 수 있게끔 명령어를 추가해 주자!

우리가 사용했던 명령어인

```
java -jar build/libs/library-app-0.0.1-SNAPSHOT.jar --spring.profiles.active=dev
```

앞뒤로 `nohup [명령어] &` 를 붙이면 된다.

```
nohup java -jar build/libs/library-app-0.0.1-SNAPSHOT.jar --spring.profiles.active=dev  
&
```

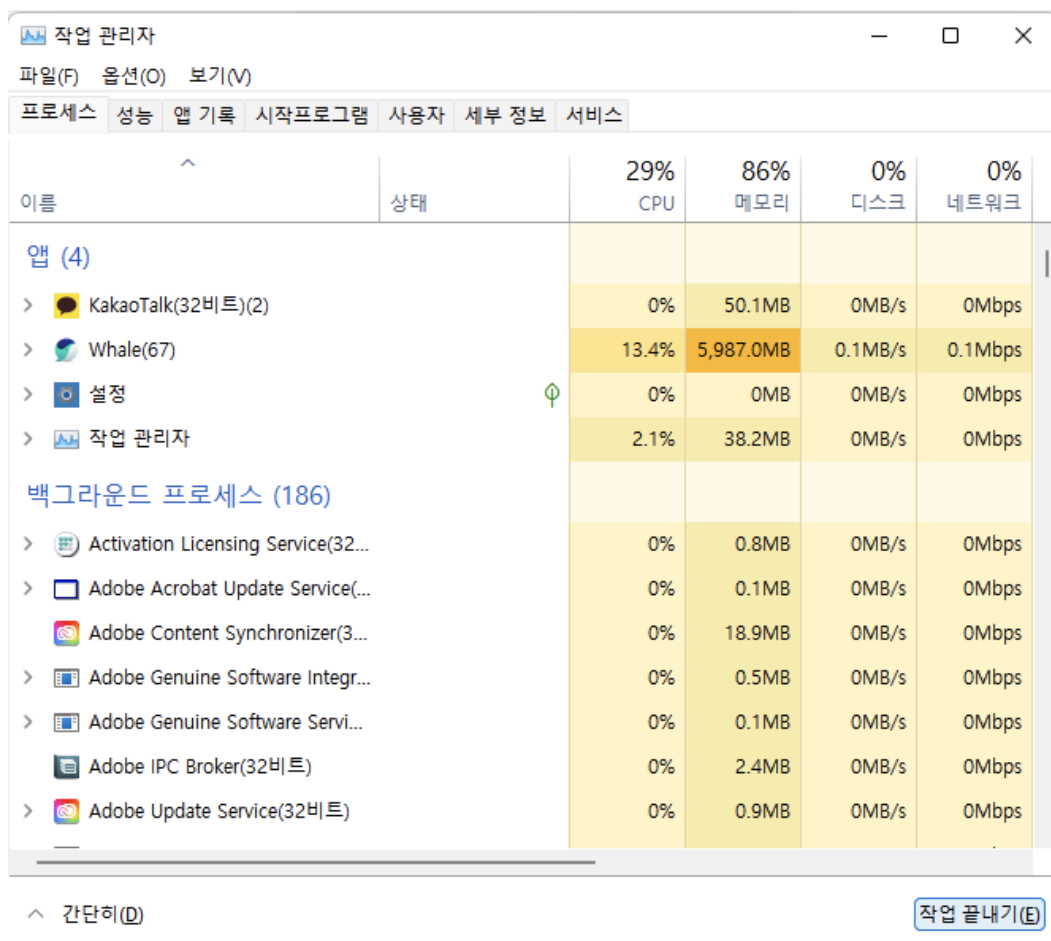

그러면, `nohup.out` 이란 파일이 생기며 서버 프로그램이 background로 실행되고, 이 파일 안에는 서버의 로그가 남게 된다!! 👍

이때 우리가 clone 받은 폴더에 `nohup.out` 파일을 남기면, 서버 관련 파일과 로그 관련 파일이 섞이므로, 상위 폴더로 올라가 실행하도록 하자.

진작 생긴 `nohup.out` 파일을 제거하는 명령어는 다음과 같다.

```
rm nohup.out # nohup.out 파일을 제거한다.
```

또한 현재 도서관리 애플리케이션이 background로 여전히 돌고 있기 때문에 해당 프로그램을 종료해 주어야 한다. 그렇다면 background로 동작하고 있는 서버를 종료하려면 어떻게 해야 할까?! **Windows의 작업관리자**와 같은 방법을 사용해야 한다! 아래 이미지를 한 번쯤 본 적이 있을 것이다.



리눅스의 작업 관리자 명령어는 `ps aux` 이다. 그리고 우리의 서버는 java로 실행되고 있으므로, 여기서 java 관련 프로그램만 보고 싶다면,

```
ps aux | grep java
```

를 타이핑하면 된다! 그렇게 되면 다음과 같이 실행 프로그램 목록이 나오는데, 프로그램마다 고유한 번호가 있다. 이 번호를 기억하고

```
kill -9 번호
```

라고 입력하게 되면, background에서 돌고 있던 서버 프로그램이 종료되게 된다.

좋다~ 상위 폴더로 올라가면 우리의 서버 프로그램 실행 경로만 수정해 주면 된다!

```
nohup java -jar library-app/build/libs/library-app-0.0.1-SNAPSHOT.jar --spring.profiles.active=dev &
```

`nohup.out` 파일을 확인하는 방법은 아래 2가지가 존재한다.

1. 파일에 직접 들어가 내용물을 읽는 방법
2. 파일에 들어가지 않고, 현재 접속한 터미널 상에서 내용물을 확인하는 방법 (일종의 미리보기)

파일에 직접 가는 명령어는 리눅스 상의 편집기인 `vim`을 활용하는 방법으로,

```
vi nohup.out
```

을 사용하면 된다.

만약 `vim` 접속한 상황에서 나오고 싶다면, ESC를 누른 후, `:q`를 타이핑해 빠져나오면 된다.

현재 접속한 터미널에서 내용물을 확인하는 명령어는 다음 2가지가 주로 사용된다.

```
cat nohup.out  
  
tail -f nohup.out
```

`cat` 명령어는 파일에 있는 내용물을 모두 출력하는 명령어이다. 때문에 파일 내용물의 양이 많지 않고 실시간 업데이트가 잘 되지 않는 파일을 확인할 때 주로 사용한다.

`tail` 명령어는 현재 파일의 끝부분을 출력하는 명령어로, `-f` 옵션을 사용할 경우 파일에 내용물이 추가되면 추가된 내용물도 지속적으로 보여줄 수 있다!!

`nohup.out` 과 같은 로그 파일의 경우에는 최근 로그를 보려면 `tail -f` 명령어를, 이전 로그를 보려면 `vi` 명령어를 추천한다!

매우 좋다~! 😊 이제 우리가 EC2 접속을 종료하더라도 서버가 정상적으로 동작하게 된다. 지난 시간처럼 한 번 확인해 보도록 하자!! 👍

이번 시간에는 EC2 접속을 종료할 때도 서버가 실행될 수 있게 하는 방법과 다양한 리눅스 명령어를 추가적으로 알아보았다. 다음 시간에는 우리 만의 도메인을 구매해 IP주소가 아닌, 도메인 이름으로 서버에 접속하는 방법을 알아보자! 😊

47강. 가비아를 이용한 도메인 구입, DNS 적용

이번 시간에는 <가비아>라는 사이트를 이용해 도메인을 구입하고 우리가 배포해 놓은 서버에 연결해 보자! 도메인은 저렴하면 500원짜리 1,000원짜리도 있어서, 실습하시기에 부담이 없으시면 따라해보시는 것도 좋고, 돈을 아끼고 싶으시다면 어떻게 진행되는지 알아만 두신 후, 나중에 꼭 필요할 때 사용해 보시는 것도 좋다.

AWS와 마찬가지로, 먼저 <가비아>에 회원가입을 해야 한다.

- 가비아 : <https://www.gabia.com/>

회원가입

로그인 정보

아이디	6~16자 / 영문 소문자, 숫자 사용 가능	
비밀번호 ⓘ	8~16자 / 문자, 숫자, 특수 문자 모두 혼용	
비밀번호 확인	비밀번호를 다시 입력해 주세요	

회원 정보

회원 유형	<div>개인</div> <div>기업</div>
-------	-----------------------------

가입 약관 동의

- ☐ 모든 약관에 동의합니다.
- ☐ 가비아이용 약관에 동의합니다. (필수) [상세보기 >](#)
- ☐ 개인정보 수집 및 이용에 동의합니다. (필수) [상세보기 >](#)
- ☐ 마케팅 활용 및 광고성 정보 수신에 동의합니다. (선택) [상세보기 >](#)
- 가비아와 가비아CNS에 관한 수신 동의 항목입니다.

• 14세 미만은 회원 가입이 제한됩니다.

가입하기

회원가입은 크게 어려운 게 없으니, 진행하면 된다.

회원가입이 끝났다면, 로그인을 하고 메인 화면에서 구매하고 싶은 도메인 이름을 검색하면 된다. 이 이름이 바로 naver.com / google.com 과 같은 이름이 될 것이다!

추천 도메인	이벤트 도메인	KR 도메인	국가 도메인	브랜드 도메인	도메인 장바구니
<input type="checkbox"/> 등록 가능					<div>0개 등록 선택 ✕</div> <div>신청하기</div> <div>견적서 출력</div>
studying-developer.com		EVENT 15,000원 / 22,000원	선택		
studying-developer.co.kr 대한민국		EVENT 15,000원 / 20,000원	선택		
studying-developer.kr 대한민국		EVENT 15,000원 / 20,000원	선택		
studying-developer.shop 🔍		이미 등록된 도메인			
studying-developer.net		18,000원 / 22,000원	선택		
studying-developer.site		EVENT 1,900원 / 49,000원	선택		
studying-developer.org		22,000원	선택		

도메인 검색하면, 금액이 포함된 목록이 나오게 되고 현재 할인 중인 도메인을 선택하면 1년간은 할인된 금액으로 사용할 수 있다. 잘 찾으면 500원이나 1000원짜리 도메인도 있다.

마음에 드는 도메인을 찾았다면, **선택** 을 누른 후 **신청하기** 를 누르면 된다.

그럼 여러 가지 정보를 입력하게 되는데, 이때 2가지 정도만 주의하면 된다.



아래 이미지는 예시로 실제 제가 구매한 도메인은 studying-developer.shop입니다!

강의에서 사용했던 도메인 및 IP는 강의 현재 시점으로 모두 정리되어 있습니다



서비스 신청



1. 신청 정보 입력



2. 서비스 비용 결제



신청 정보

도메인	등록 비용 X 기간: 3년 ▼
studying-developer.site	31,666원 x 3년 ▼ = 95,000원 ✕

서비스 관리 정보

네임서버 목록

네임서버 설정 

☒ 가비아 네임서버 사용

☐ 타사 네임서버 사용 

부가서비스 추가

안전 잠금 신청
(무료)

☐ 안전 잠금 서비스 신청

동시 신청 가능 서비스

동시 신청 서비스

☐ 웹호스팅

☐ 컨테이너호스팅

☐ 워드프레스호스팅

☒ 신청 안 함

1. 실습의 목적이라면 등록 기간이 기본 3년으로 되어 있으니 이를 1년으로 변경해 주자. 1년이 지난 후부터는 할인된 가격으로 적용되지 않아 비용이 많이 들어간다.
2. 서비스 관리 정보 → 네임서버 설정 부분에서는 ‘가비아 네임서버 사용’을 선택하자. 기본적으로 선택되어 있어 딱히 변경할 내용은 없다.


이렇게 결제까지 하게 되면 한 5분 ~ 10분 정도 있다가 [메인 페이지 > My 가비아](#) 를 들어가 보자!


이때 다음과 같이 “도메인 1건”이 나온다면 성공이다!!! 😊


최태현님

My 정보 관리

12월 결제 예상 금액

0원 

결제 수단 등록하기 

1% 더 할인 받으려면?
정기결제 신청하기 

결제 관련 바로가기

미결제 주문서

0개 발급

결제하지 않은 주문서 및 가상 계좌

세금 계산서

0개 발급 가능

이번 달 기준 발급 가능한 세금 계산서

카드

0개 생성

구매할 서비스 목록이 담긴 주문서

예치금


0원

현금처럼 사용 가능한 예치금

이용 중인 서비스

대시보드 튜토리얼보기 >

현재 이용 중인 서비스를 확인하세요.



도메인

1건 >

IT환경의 최적화된 가비아 서비스를 이용해보세요.

호스팅

홈페이지

쇼핑몰

클라우드

하이웍스

IDC

보안

전체보기

이용 중인 서비스 관리 버튼입니다.

DNS 관리툴

소유권 이전

담당자 설정

도메인 통합 관리툴

• 관리 정보 입력 필요 서비스

0건 >

• 12월 연장 필요 서비스

0건 >

이 도메인 1건을 클릭해 들어가면, [관리](#) 라는 버튼이 보인다.

☐ 도메인 **studying-developer.shop** 2022-08-16 ~ 2023-08-16 (D-237) 연장 > 49,000원/년 **관리**

< 1 >

관리 버튼을 누르고 왼쪽 탭에서 **DNS 정보** 탭을 누르면 다음과 같은 화면이 보인다.

DNS 정보

DNS 정보는 도메인에 연결된 서비스를 확인할 수 있습니다.

DNS 설정, 도메인 연결, 포워딩, 파킹 서비스 등의 신청 및 설정은 'DNS 관리'에서 가능합니다.

DNS 관리

		상세 ▾		20개 ▾
번호	도메인명 ▾ 전체 ▾	DNS 정보	연결 서비스	만기일 ▾
1	studying-developer.shop	설정된 DNS 레코드 정보가 없습니다.		2023-08-16

엑셀 다운로드

바로 여기서 우리는 구매한 도메인과 우리가 빌린 EC2 서버를 연결할 수 있다!! 🍀

DNS 관리를 눌러 **설정** > **레코드 수정** > **레코드 추가**를 눌러주고

DNS 설정

DNS 설정	도메인 연결	포워딩	웹 파킹	모바일 파킹
<input type="checkbox"/>	도메인명 ^	DNS 정보		
<input type="checkbox"/>	studying-developer.shop	설정된 DNS 레코드 정보가 없습니다.		

설정

DNS 관리

studying-developer.shop

• 레코드 개수 : 0개 • 최근 업데이트 : - • 네임서버 : ns.gabia.co.kr

DNS 설정

레코드 수정

DNS 레코드 수정

타입 ▾ ⓘ	호스트	값/위치	TTL	우선 순위	서비스 ▾	상태
A ▾	www	43.201.84.217	600 ▾		DNS 설정	확인 삭제

+ 레코드 추가

저장 취소

- 타입 : A
- 호스트 : www
 - 나중에 www.studying-developer.shop 으로 접속할 수 있게 된다. www 대신 다른 이름을 적어도 좋다!
- 값/위치 : EC2에서 빌린 인스턴스의 IP

를 설정해 주도록 하자!!

그리고 **확인** > **저장** 을 누르면, 도메인 네임이 등록된다!! 🙌

매우 좋다~~ 다음으로 해주어야 할 것은, AWS EC2에서 8080 포트로 접근할 수 있도록 security group (방화벽) 설정을 해주어야 한다!

AWS EC2 대시보드에 들어가 인스턴스 선택 > 보안 탭 > 보안 그룹 클릭 > 인바운드 규칙 편집 > 규칙 추가를 해주자!

인스턴스 (1/1) 정보

인스턴스를 속성 또는 (case-sensitive) 태그로 찾기

인스턴스 상태 = running 필터 지우기

이름	인스턴스 ID	인스턴스 상태	인스턴스 유형	상태 검사	경보 상태	가용 영역	퍼블릭 IPv4 DNS
Library App Se...	i-01a01f13449d94919	실행 중	t2.micro	2/2개 검사 통과	경보 없음	ap-northeast-2a	ec2-43-201-84

인스턴스: i-01a01f13449d94919(Library App Server)


세부 정보 **보안** 네트워킹 스토리지 상태 검사 모니터링 태그

▼ 보안 세부 정보

IAM 역할
-

소유자 ID
344400803440

시작 시간
Thu Oct 27 202

보안 그룹
sg-02c127cc51f1fa4b0 (launch-wizard-2) 

▼ 인바운드 규칙

필터 규칙

이름	보안 그룹 규칙 ID	포트 범위	프로토콜	원본	보안 그룹
-	sgr-098dc6c3e60cf687d	22	TCP	0.0.0.0/0	launch-wizard-2


인바운드 규칙 **아우트바운드 규칙** 태그

이제 Reachability Analyzer를 사용하여 네트워크 연결을 확인할 수 있습니다. Reachability Analyzer 실행

인바운드 규칙 (1/1)

보안 그룹 규칙 필터

이름	보안 그룹 규칙 ID	IP 버전	유형	프로토콜	포트 범위	소스	설명
-	sgr-098dc6c3e60cf687d	IPv4	SSH	TCP	22	0.0.0.0/0	-

태그 관리 인바운드 규칙 편집 

인바운드 규칙 정보

보안 그룹 규칙 ID	유형	정보	프로토콜	정보	포트 범위	정보	소스	정보
sgr-098dc6c3e60cf687d	SSH		TCP		22		사용자 지정	Q
-	사용자 지정 TCP		TCP		8080		Anywhere-I...	Q

규칙 추가

8080포트로 아무나 들어올 수 있게 하거나, 현재 나의 ip를 설정하면 된다!!

자 이제 모든 준비는 끝이 났다! 원래는 `http://43.201.84.217:8080/v1/index.html` 로 접속했지만, 우리는 이제 DNS 설정을 했기 때문에 `43.201.84.217` 대신 `www.studying-developer.shop` (예시)을 사용할 수 있다!

`http://www.studying-developer.shop:8080/v1/index.html` (예시)로 접속해 보자!!

매우 좋다~!! 이제 우리는 도서관리 애플리케이션 서버 개발부터 배포까지 완전히 마무리하였다. 다음 시간에는 이번 Section을 정리해 보자!! 🍀 😊

48강. Section 7 정리

이번 Section에서는 지금까지 우리가 열심히 개발했던 서버를 드디어 배포해 보았다.

🍀 🎉 이 과정에서 다음과 같은 내용들을 다룰 수 있었다.

1. EC2에 접속하는 방법을 알아보고, EC2에 접속해 리눅스 명령어를 다뤄본다.
2. 개발한 서버의 배포를 위해 환경 세팅을 리눅스에서 진행하고, 실제 배포를 진행한다.
3. foreground와 background의 차이를 이해하고 background 서버를 제어한다.
4. 도메인 이름을 사용해 사용자가 IP 대신 이름으로 접속할 수 있도록 한다.

추가적으로 이번 Section에서 나왔던 리눅스 명령어, Gradle 명령어들을 간단히 정리해 보자.

- `mkdir` : 폴더를 만든다.
- `ls` : 현재 위치에서 폴더나 파일을 확인한다.
 - `ls -l` : 조금 더 자세한 정보를 확인할 수 있다.
- `cd` : 폴더 안으로 들어간다.
 - `cd ..` : 상위 폴더로 올라간다.
- `pwd` : 현재 위치를 확인한다.

- `rmdir` : 비어 있는 폴더(디렉토리)를 제거한다.
- `sudo yum update` : 관리자의 권한으로 설치되어 있는 여러 프로그램을 최신화한다.
- `sudo yum install 프로그램 이름` : 관리자의 권한으로 프로그램을 설치한다.
- `sudo systemctl status 프로그램` : 프로그램의 상태를 확인한다.
- `sudo systemctl restart 프로그램` : 프로그램을 재시작한다.
- `chmod` : 파일이나 폴더의 권한을 변경한다.
- `ctrl + c` : foreground로 실행 중인 프로그램을 중단하는 신호이다.
- `nohup [명령어] &` : 명령어를 background로 실행시킨다.
- `rm` : 파일을 제거한다.
- `vi` : 리눅스 편집기인 vim을 이용해 파일을 연다.
- `cat` : 파일에 있는 내용물을 모두 출력한다.
- `tail` : 현재 파일의 끝부분을 출력한다.
 - `tail -f` : 현재 파일의 끝부분을 실시간으로 출력한다.
- `ps aux` : 현재 실행 중인 프로그램 목록을 확인한다.
 - `ps aux | grep java` : java가 들어가는 프로그램 목록을 확인한다.
- `kill -9 프로그램 번호` : 해당 프로그램을 종료시킨다.
- `./gradlew build` : 프로젝트를 빌드한다.
- `./gradlew build -x test` : 프로젝트를 빌드하는데, 테스트는 생략한다.
- `./gradlew clean` : 현재 빌드된 결과물을 제거한다.

다음 Section에서는 Spring과 Spring Boot는 어떻게 다른지, build.gradle에 있는 코드의 의미는 무엇인지 등을 알아보도록 하자! 🔥🏃