실습 가이드 · Hands-on

스프링부트 배포 실습

가상머신에서 빌드부터 실행·방화벽 설정까지

리눅스 가상머신에 **SSH로 접속한 상태**를 가정합니다. 소스를 받아 직접 빌드하고, 백그라운드로 띄운 뒤, 방화벽을 열어 외부에서 접속할 수 있게 만드는 전 과정을 명령어 단위로 정리했습니다. 배포 대상은 화면(타임리프 템플릿)만 응답하는 스프링부트 앱 `linuxsprov` 입니다(DB 연동 없음).

전체 5단계

1 레포지토리 `git clone` → **2** `application.properties` 작성(vi) → **3** `gradlew` 실행권한 부여 →

4 빌드 & 백그라운드 실행 → **5** 방화벽 허용(ufw)

1 소스 받기부터 빌드·실행까지

STEP 1 레포지토리 git clone

산출물(.jar)을 직접 전송하는 대신, 서버(가상머신)에서 소스를 받아 그 자리에서 빌드하는 방식입니다.

```
$ git clone <레포지토리 주소>
$ cd linuxsprov
```

STEP 2 application.properties 작성 (vi)

앱의 동작 설정을 작성합니다. DB는 아직 없으므로, **화면(템플릿)을 그려 응답하는 데 필요한 설정만** 넣습니다.

```
$ vi src/main/resources/application.properties
```

application.properties

```
server.port=800
spring.thymeleaf.prefix=classpath:/templates/
spring.thymeleaf.suffix=.html
spring.thymeleaf.cache=false
spring.thymeleaf.check-template-location=true
spring.thymeleaf.mode=HTML
```

설정	의미
`server.port=800`	앱이 800번 포트에서 동작(기본 8080 대신). 뒤 5단계에서 방화벽도 800을 열어 줌
`thymeleaf.prefix / suffix`	타임리프 템플릿 위치(`templates/`)와 확장자(`.html`) → `templates/이름.html` 을 화면으로 렌더링
`thymeleaf.cache=false`	템플릿 캐시 끄기 — 수정 시 즉시 반영(개발 편의)
`check-template-location=true`	템플릿 폴더가 실제로 있는지 시작 시 확인
`thymeleaf.mode=HTML`	HTML 모드로 처리

> 타임리프(Thymeleaf)는 서버가 HTML 페이지를 그려서 돌려주는 템플릿 엔진입니다. **주의:** 800처럼 1024 미만 포트는 리눅스에서 관리자 권한이 필요해, 실행 시 `sudo` (또는 systemd의 `User=root`)가 쓰입니다.

STEP 3 gradlew 실행권한 부여

빌드 스크립트 `gradlew` 에 실행 권한(`x`)을 줍니다.

```
$ sudo chmod +x gradlew
```

STEP 4 빌드 진행 & 백그라운드 실행

`clean`(이전 산출물 제거)과 `bootJar`(실행 가능한 JAR 생성)를 함께 실행합니다.

```
$ ./gradlew clean bootJar
```

빌드가 끝나면 산출물이 다음 경로에 생깁니다. 그 폴더로 이동해 백그라운드로 실행합니다.

```
# 산출물 경로: /opt/linuxsprov/build/libs/linuxsprov-0.0.1-SNAPSHOT.jar
$ cd /opt/linuxsprov/build/libs
$ sudo nohup java -jar linuxsprov-0.0.1-SNAPSHOT.jar > error.log 2>&1 &
```

명령 조각	의미
`sudo`	800번(1024 미만) 포트를 쓰기 위한 관리자 권한
`nohup … &`	터미널을 닫아도 종료되지 않게 + 백그라운드 실행
`> error.log 2>&1`	표준출력과 표준에러를 모두 `error.log` 파일로 모음

⚠️ 백그라운드 실행 시 알아둘 점

nohup으로 띄운 프로세스는 **서버를 재부팅하면 자동으로 다시 뜨지 않습니다.** 재부팅한 경우 위 명령으로 다시 실행해야 합니다. 또한 다시 실행할 때는 **기존 프로세스를 먼저 종료**해야 800 포트 충돌이 없습니다(예: `ps -ef | grep java` 로 PID 확인 후 종료). 실행 로그는 같은 폴더의 `error.log` 에서 확인할 수 있습니다(`tail -f error.log`).

2 방화벽 허용 (ufw)

STEP 5 800번 포트 열기

앱이 800번 포트에서 돌아도, 방화벽이 그 포트를 막고 있으면 외부에서 접속되지 않습니다. 그래서 포트를 열어 줍니다.

```
$ sudo ufw allow 800/tcp
$ sudo ufw status
```

🔥 ufw 란? (Uncomplicated Firewall)

우분투에 기본 포함된 **방화벽 관리 도구**입니다. 리눅스 본래의 복잡한 방화벽(iptables)을 누구나 쉽게 다루도록 감싼 것이라 이름이 "Uncomplicated(복잡하지 않은)"입니다. "어떤 포트·서비스를 들여보낼지(allow) / 막을지(deny)"를 간단한 명령으로 설정합니다. 꼭 필요한 포트만 열고 나머지는 닫아 두는 것이 보안의 기본입니다.

명령	설명
`sudo ufw allow 800/tcp`	800번 포트의 TCP 접속을 **허용**(우리 앱 포트를 외부에 개방)
`sudo ufw status`	현재 방화벽 규칙·상태 **확인** (800/tcp 가 ALLOW로 보이면 정상)
`sudo ufw enable / disable`	(참고) 방화벽 자체를 켜기 / 끄기
`sudo ufw delete allow 800/tcp`	(참고) 앞서 추가한 허용 규칙을 삭제

참고: SSH 접속(22번)을 쓰는 중이라면 방화벽을 켜기 전에 `sudo ufw allow 22/tcp` 도 함께 열어 둬야 원격 접속이 끊기지 않습니다. 클라우드 환경이라면 ufw 외에 **클라우드 보안그룹**에서도 같은 포트를 열어야 할 수 있습니다.