Ubuntu 사용자 계정에 Tomcat 설치

#01. 기존의 Tomcat 삭제하기

만약 이전 포스팅을 통해서 Tomcat을 샌드박스 형태로 설치한 상황이라면 아래의 명령어를 순차적으로 수행해서 제거합니다.

```
# 부팅시 자동 시작 해제
$ sudo systemctl disable tomcat10

# 서비스 중지
$ sudo systemctl stop tomcat10

# Tomcat10 제거
$ sudo purge -y tomcat10
```

#02. Tomcat10 설치하기

1. Tomcat 내려받기

https://tomcat.apache.org/download-10.cgi에서 Tomcat의 **tar.gz** 형식의 다운로드 URL을 확인합니다.

> https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.44/bin/apache-tomcat-10.1.44.tar.gz

여기서는 사용자 홈 디렉토리에서 진행합니다.

```
$ pwd
/home/leekh
```

리눅스 터미널에 톰켓을 운영할 사용자 계정으로 접속 후 아래의 명령으로 다운로드 받습니다.

```
$ wget https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.44/bin/apache-tomcat-10.1.44.tar.gz
```

2. Tomcat 압축 해제

```
$ tar -xzvf apache-tomcat-10.1.44.tar.gz
```

압축이 해제된 폴더를 확인합니다.

```
$ ls -l
합계 12345
# ...
-rw-r--r--  1 leekh leekh 12345678  3월  3 12:34 apache-tomcat-10.1.44.tar.gz
drwxr-xr-x  9 leekh leekh     4096  3월  3 12:35 apache-tomcat-10.1.44
# ...
```

3. Tomcat 서비스 등록하기

서비스 설정 파일 생성

`systemctl` 명령으로 실행할 수 있는 서비스 설정 파일을 직접 생성해야 합니다.

```
# sudo vi /etc/systemd/system/생성할서비스이름.service
$ sudo vi /etc/systemd/system/tomcat10-leekh.service
```

VI 에디터가 열리면 다음의 내용을 입력합니다.

```
[Unit]
Description={간단한 설명글}
After=network.target

[Service]
Type=forking

User={사용자계정}
Group={사용자그룹, 일반적으로 계정명과 동일}

Environment=CATALINA_HOME={TOMCAT설치경로}
Environment=CATALINA_BASE={TOMCAT설치경로}

ExecStart={TOMCAT설치경로}/bin/startup.sh
ExecStop={TOMCAT설치경로}/bin/shutdown.sh

Restart=on-failure
RestartSec=10

[Install]
WantedBy=multi-user.target
```

여기서는 사용자 계정명이 `leekh`이므로 아래와 같이 설정했습니다.

```
[Unit]
Description=Tomcat for user leekh
After=network.target

[Service]
Type=forking

User=leekh
Group=leekh

Environment=CATALINA_HOME=/home/leekh/apache-tomcat-10.1.44
Environment=CATALINA_BASE=/home/leekh/apache-tomcat-10.1.44

ExecStart=/home/leekh/apache-tomcat-10.1.44/bin/startup.sh
ExecStop=/home/leekh/apache-tomcat-10.1.44/bin/shutdown.sh

Restart=on-failure
RestartSec=10
```

```
[Install]
WantedBy=multi-user.target
```

서비스 등록하기

```
# 서비스 설정파일 갱신
$ sudo systemctl daemon-reload

# 서비스 부팅시 자동 실행 등록
$ sudo systemctl enable tomcat10-leekh

# 서비스 가동하기
$ sudo systemctl start tomcat10-leekh

# 서비스 동작 확인
$ systemctl status tomcat-leekh
```

이 단계까지 진행하면 리눅스 외부에서 `http://리눅스아이피:8080`으로 접속했을 때 Tomcat의 Welcom 페이지가 표시됩니다. 단, 접속을 위해 8080번 포트의 방화벽을 열어줘야 합니다.

```
# 8080번 방화벽 해제
$ sudo ufw allow 8080/tcp

# 방화벽 갱신
$ sudo ufw reload

# 방화벽 상태 확인
$ sudo ufw status
```

#03. SpringBoot App을 Tomcat에 배포하기

1. 업로드, 로그 디렉토리 생성

파일이 업로드 될 폴더와 로그 파일이 저장될 폴더를 생성합니다.

Tomcat이 현재 사용자 권한으로 실행되기 때문에 이미 소유자 권한을 갖고 있으므로 추가적인 퍼미션 처리는 필요 없습니다.

```
$ mkdir spring-upload
$ mkdir spring-log
```

2. SpringBoot 코드 수정

build.gradle

```
plugins {
    // ...
    id 'war'    // <--- 추가 (위치 상관 없음)
    // ...
```

```
}

// ...

dependencies {
    // ...

    // WAR 배포를 위한 의존성 (위치 상관 없음)
    providedRuntime 'org.springframework.boot:spring-boot-starter-tomcat'

    // ...
}

// 추가
war {
    archiveFileName =
"${archiveBaseName.get()}-${archiveVersion.get()}.${archiveExtension.get()}"
}

// ...
```

메인클래스 (OOOOApplication.java)

`SpringBootServletInitializer` 클래스에 대한 상속을 추가하고 `configure` 메서드를 재정의 합니다.

```
package kr.hossam.myshop;

import org.springframework.boot.SpringApplication;
import
org.springframework.boot.actuate.autoconfigure.wavefront.WavefrontProperties.Application;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.scheduling.annotation.EnableScheduling;

// import 구문 추가
import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;
import org.springframework.boot.builder.SpringApplicationBuilder;

@EnableScheduling
@SpringBootApplication
// SpringBootServletInitializer 클래스 상속 추가
public class MyshopApplication extends SpringBootServletInitializer {

    public static void main(String[] args) {
        SpringApplication.run(MyshopApplication.class, args);
    }

    // WAR 배포를 위한 설정을 수행하는 메서드 재정의
    @Override
    protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
        // return application.sources(클래스이름.class);
        return application.sources(MyshopApplication.class);
    }
}
```

logback-spring.xml

로그 파일이 저장되는 경로를 앞서 생성한 디렉토리로 지정합니다.

```
<!-- ... 생략 ... -->

<!-- log file path -->
<property name="LOG_PATH" value="/home/leekh/spring-log" />

<!-- ... 생략 ... -->
```

application.properties

파일이 업로드 될 디렉토리의 경로를 앞서 생성한 디렉토리로 지정합니다.

이 때 데이터베이스 접속 정보가 개발장비와 Linux서버가 서로 다르다면 Linux 서버에 맞춰 수정합니다.

```
spring.datasource.url=jdbc:log4jdbc:mariadb://127.0.0.1:3306/myschool?characterEncoding=UTF8
spring.datasource.driver-class-name=net.sf.log4jdbc.sql.jdbcapi.DriverSpy
spring.datasource.username=myschool
spring.datasource.password=1234

# ... 생략 ...

upload.dir=/home/leekh/study-springboot
```

FileHelper.java

이 파일은 제가 진행하는 수업의 예제 코드에 해당하는 내용입니다.

SpringBoot를 Tomcat에 배포할 경우 jar 파일의 이름으로 중간 경로가 생기는데 이를 ContextPath라고 합니다.

업로드 된 파일의 URL을 생성할 때 ContextPath를 덧붙여 주는 처리를 추가합니다.

```
package kr.hossam.myshop.helpers;

// ... 생략 ...

    // [추가] Request 객체 주입을 위한 import문
import jakarta.servlet.http.HttpServletRequest;

@Slf4j
@Component
@RequiredArgsConstructor
public class FileHelper {

    // [추가] Request 객체 주입
    private final HttpServletRequest request;

    // ... 생략 ...

    public String getFileUrl(String filePath) throws Exception {
        if (filePath == null || filePath.isEmpty()) {
            return null;
        }
```

```
        // 파일이 존재하는지 확인
        File file = new File(uploadDir, filePath);
        if (!file.exists()) {
            return null;
        }

        // [수정] 파일 URL 생성 --> 사이트 URL에 ContextPath 추가
        String fileUrl = String.format("%s%s%s", request.getContextPath(), uploadUrl,
filePath);
        return fileUrl;
    }
}
```

3. SpringBoot 빌드하기

프로젝트 루트 디렉토리 위치에서 터미널을 열고 아래의 명령을 수행합니다.

> 개발PC에서 수행하세요.

```
$ gradlew bootWar
```

프로젝트 폴더내의 `/build/libs` 디렉토리 안에 `*.war`파일이 생성됩니다. 이 파일의 이름을 적절히 수정합니다.

톰캣에 업로드 했을 때 `http://서버주소:포트번호/war파일이름` 으로 URL이 생성됩니다.

4. war 파일 업로드

FTP를 통해 리눅스에 접속합니다. 톰캣이 설치된 사용자 계정으로 접속하면 Tomcat의 디렉토리가 보입니다.

톰캣 디렉토리 내의 `webapps` 폴더 안에 war 파일을 업로드 하고 톰캣을 재시작 합니다.

```
$ sudo systemctl restart tomcat10-leekh
```

이제 웹 브라우저로 접속하여 결과를 확인합니다.

5. 수정된 war 파일을 재배포하기

war 파일을 Tomcat의 webapps 디렉토리에 업로드하고 Tomcat을 재가동하면 war파일의 압축이 해제됩니다.

FTP로 war파일과 압축이 해제된 폴더를 삭제한 후, 수정된 war파일을 다시 업로드하고 Tomcat을 재가동하면 재배포가 완료됩니다.