KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

# 개발을 위한 준비 (심화 1)

## [KB] IT's Your Life

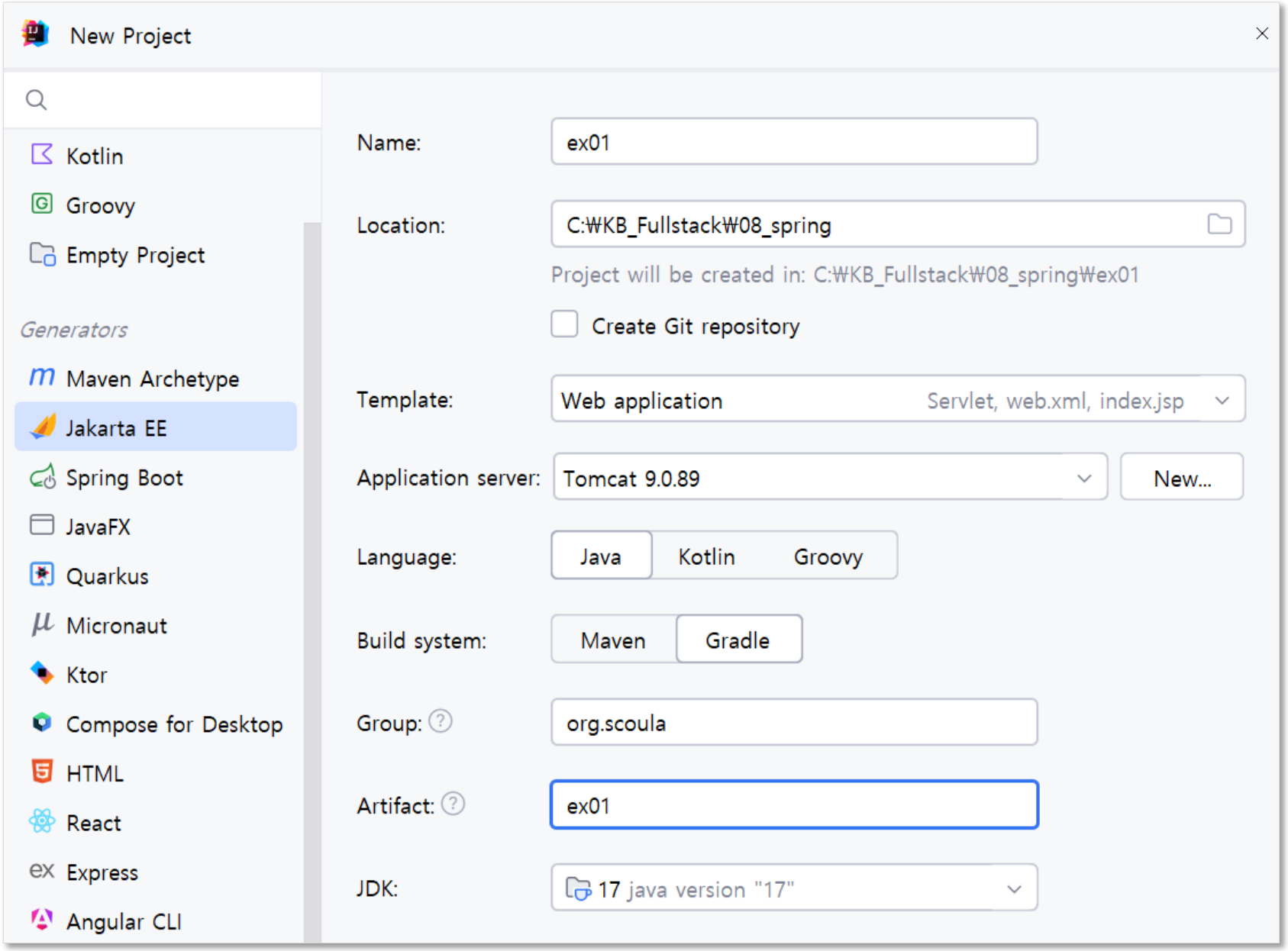
KB 국민은행

## Spring Legacy 프로젝트의 기본 틀을 만드세요.

- 프로젝트명: ex01
- group: org.scoula
- Artifact Id: ex01

- 작성 파일
  - build.gradle
  - resources/log4j2.xml
  - org.scoula.config
    - RootConfig.java
    - ServletConfig.java
    - WebConfig.java
  - org.scoula.controller
    - HomeController
  - WEB-INF/views/index.jsp
- 삭제 파일
  - 기존 index.jsp, web.xml 삭제

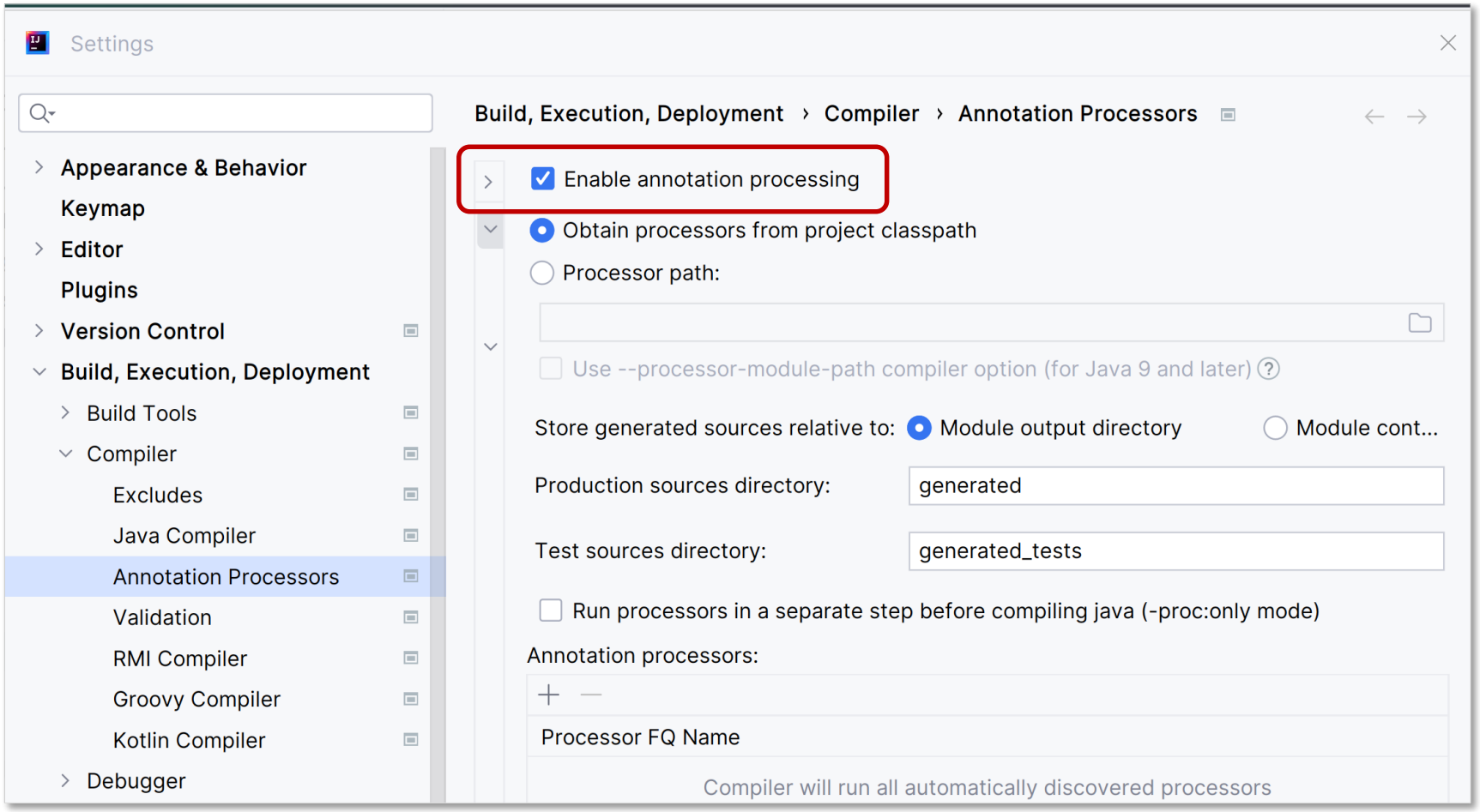
- 톰캣 설정
  - context path: /


New Project
Kotlin
Groovy
Empty Project
Generators
Maven Archetype
Jakarta EE
Spring Boot
JavaFX
Quarkus
Micronaut
Ktor
Compose for Desktop
HTML
React
Express
Angular CLI
Name:
ex01
Location:
C:₩KB_Fullstack₩08_spring
Project will be created in: C:₩KB_Fullstack₩08_spring₩ex01
Create Git repository
Template:
Web application
Servlet, web.xml, index.jsp
Application server:
Tomcat 9.0.89
New...
Language:
Java
Kotlin
Groovy
Build system:
Maven
Gradle
Group:
org.scoula
Artifact:
ex01
JDK:
17 java version "17"

- **Version: Jakarta EE 8**
  - Servlet 4.0.1

## Lombok 설정

- File > Setting... > Build, Exeuction, Deployment > Compiler > Annotation Processors
  - Enable annotation processing 체크 → 프로젝트 생성시 마다 해줘야 함



## build.gradle

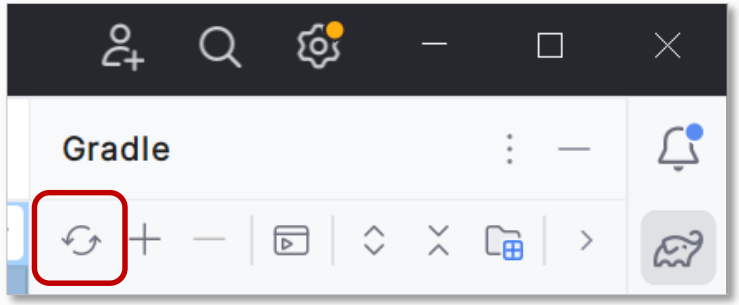
```
plugins {
  id 'java'
  id 'war'
}

group 'org.scoula'
version '1.0-SNAPSHOT'

repositories {
  mavenCentral()
}

ext {
  junitVersion = '5.9.2'
  springVersion = '5.3.37'
  lombokVersion = '1.18.30‘
  log4jVersion = '2.22.1'
}

sourceCompatibility = '1.17'
targetCompatibility = '1.17'

tasks.withType(JavaCompile) {
  options.encoding = 'UTF-8'
}
```

## build.gradle

```
dependencies {
  // 스프링
  implementation ("org.springframework:spring-context:${springVersion}")
          { exclude group: 'commons-logging', module: 'commons-logging' }
  implementation "org.springframework:spring-webmvc:${springVersion}"
  implementation 'javax.inject:javax.inject:1'

  // AOP
  implementation 'org.aspectj:aspectjrt:1.9.20'
  implementation 'org.aspectj:aspectjweaver:1.9.20'

  // JSP, SERVLET, JSTL
  implementation('javax.servlet:javax.servlet-api:4.0.1')
  compileOnly 'javax.servlet.jsp:jsp-api:2.1'
  implementation 'javax.servlet:jstl:1.2'

  // Logging
    implementation "org.apache.logging.log4j:log4j-api:${log4jVersion}"
    implementation "org.apache.logging.log4j:log4j-core:${log4jVersion}"
    implementation "org.apache.logging.log4j:log4j-slf4j-impl:${log4jVersion}"

  // xml내 한글 처리
  implementation 'xerces:xercesImpl:2.12.2'
```

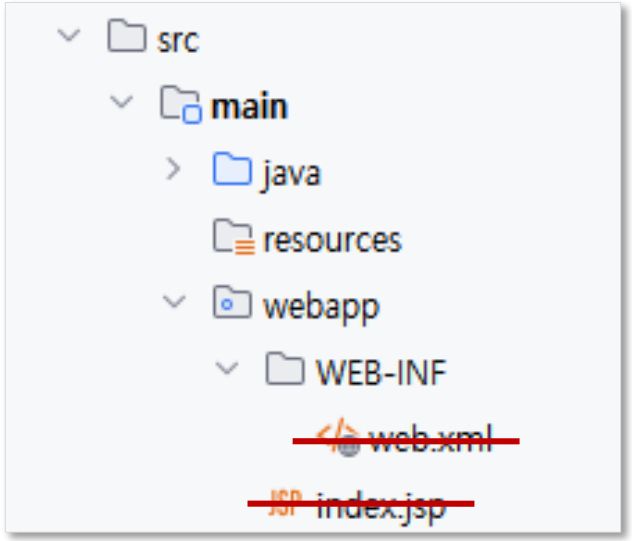
## build.gradle

```
  // Lombok
  compileOnly "org.projectlombok:lombok:${lombokVersion}"
  annotationProcessor "org.projectlombok:lombok:${lombokVersion}"

  // Jackson - Json 처리
  implementation 'com.fasterxml.jackson.core:jackson-databind:2.9.4'

  // 테스트
  testImplementation "org.springframework:spring-test:${springVersion}"
  testCompileOnly"org.projectlombok:lombok:${lombokVersion}"
  testAnnotationProcessor "org.projectlombok:lombok:${lombokVersion}"

  testImplementation("org.junit.jupiter:junit-jupiter-api:${junitVersion}")
  testRuntimeOnly("org.junit.jupiter:junit-jupiter-engine:${junitVersion}")
}


test {
  useJUnitPlatform()
}
```

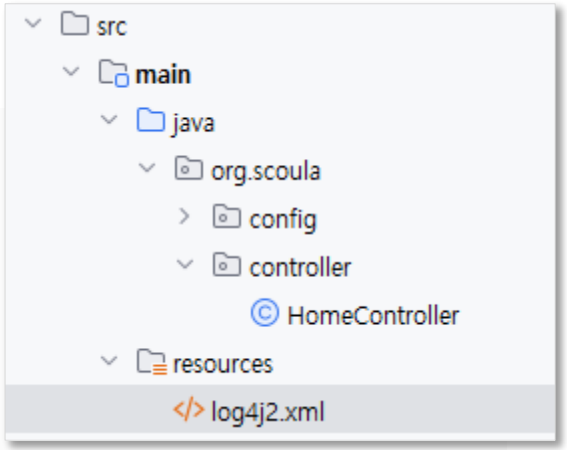


## 디렉토리 구조

- java
  - Java 클래스 배치
- resources
  - Java 클래스를 제외한 모든 파일(설정, xml 등)
- webapp
  - Web URL로 접근하는 root 디렉토리
  - WEB-INF
    - 웹 설정, 스프링 View 배치 디렉토리
    - index.jsp, web.xml 삭제



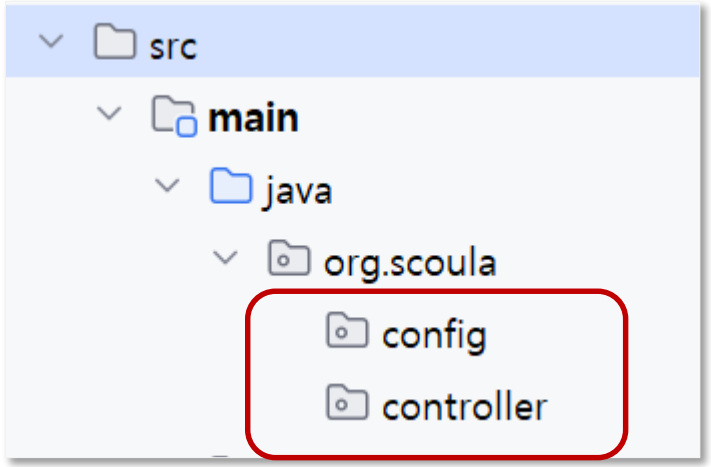
## resources/log4j2.xml




```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>

    <!-- Appender, Layout 설정 -->
    <Appenders>
        <Console name="console" target="SYSTEM_OUT">
            <PatternLayout pattern=" %-5level %c(%M:%L) - %m%n"/>
        </Console>
    </Appenders>

    <!-- Logger 설정 -->
    <Loggers>
        <Root level="INFO">
            <AppenderRef ref="console"/>
        </Root>
        <Logger name="org.scoula" level="INFO" additivity="false" >
            <AppenderRef ref="console"/>
        </Logger>
        <Logger name="org.springframework" level="INFO" additivity="false">
            <AppenderRef ref="console"/>
        </Logger>
    </Loggers>

</Configuration>
```

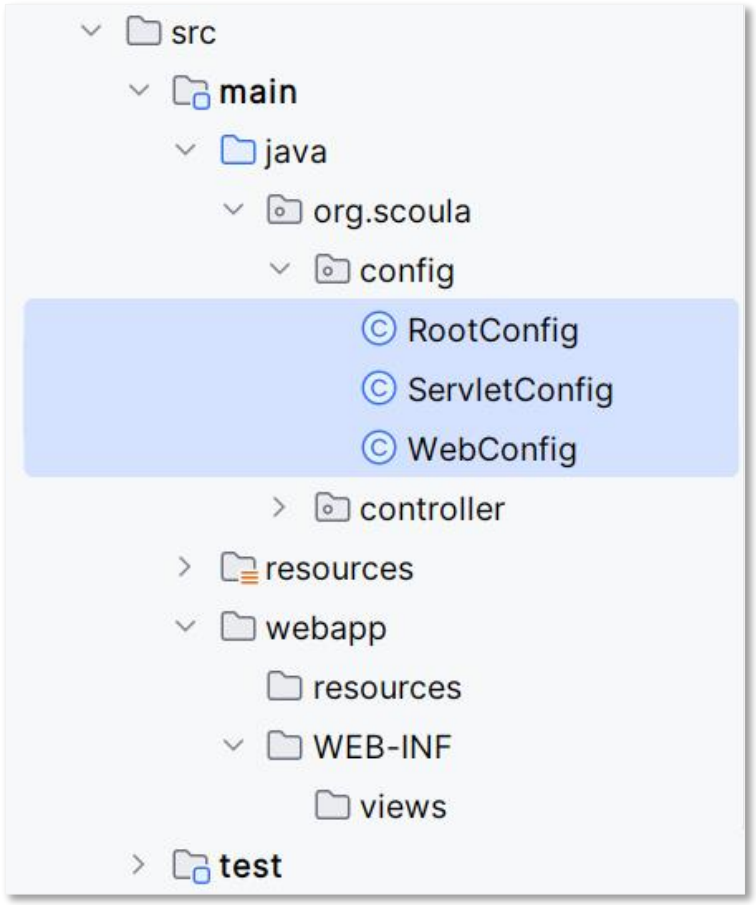
## 패키지 생성

- src/main/java에 패키지 생성
  - org.scoula
    - config : 스프링 설정 패키지
    - controller : 기본 컨트롤러 패키지



## 설정 파일 만들기

- org.scoula.config
  - RootConfig.java
  - ServletConfig.java
  - WebConfig.java



## RootConfig.java

```
package org.scoula.config;

import org.springframework.context.annotation.Configuration;

@Configuration
public class RootConfig {

}
```

## ServletConfig.java

```
package org.scoula.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
import org.springframework.web.servlet.config.annotation.ViewResolverRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
import org.springframework.web.servlet.view.InternalResourceViewResolver;
import org.springframework.web.servlet.view.JstlView;

@EnableWebMvc
@ComponentScan(basePackages = {"org.scoula.controller"})                    // Spring MVC용 컴포넌트 등록을 위한 스캔 패키지
public class ServletConfig implements WebMvcConfigurer {

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {
        registry
            .addResourceHandler("/resources/**")      // url이 /resources/로 시작하는 모든 경로
            .addResourceLocations("/resources/");     // webapp/resources/경로로 매핑
    }
```

## ServletConfig.java

```
    // jsp view resolver 설정
    @Override
    public void configureViewResolvers(ViewResolverRegistry registry) {
        InternalResourceViewResolver bean = new InternalResourceViewResolver();

        bean.setViewClass(JstlView.class);
        bean.setPrefix("/WEB-INF/views/");
        bean.setSuffix(".jsp");

        registry.viewResolver(bean);
    }

}
```

## WebConfig.java

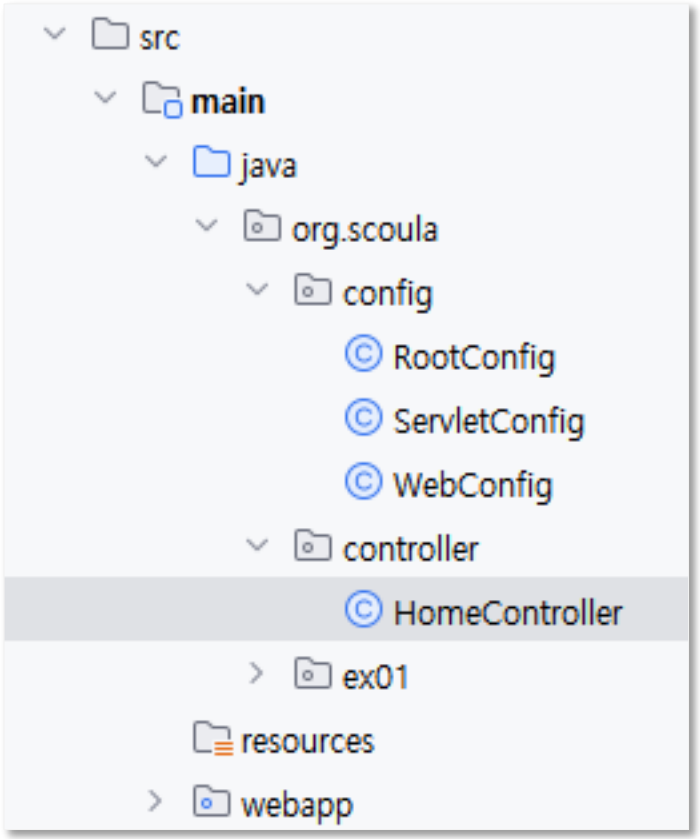
```
package org.scoula.config;

import org.springframework.web.filter.CharacterEncodingFilter;
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

import javax.servlet.Filter;

public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        return new Class[] { RootConfig.class };
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { ServletConfig.class };
    }

    // 스프링의 FrontController인 DispatcherServlet이 담당할 Url 매핑 패턴, / : 모든 요청에 대해 매핑
    @Override
    protected String[] getServletMappings() {
        return new String[] { "/" };
    }
```

## WebConfig.java

```
    // POST body 문자 인코딩 필터 설정 - UTF-8 설정
    protected Filter[] getServletFilters() {
        CharacterEncodingFilter characterEncodingFilter = new CharacterEncodingFilter();

        characterEncodingFilter.setEncoding("UTF-8");
        characterEncodingFilter.setForceEncoding(true);

        return new Filter[] {characterEncodingFilter};
    }

}
```

## 컨트롤러 만들기

- org.scoula.controller
  - HomeController.java



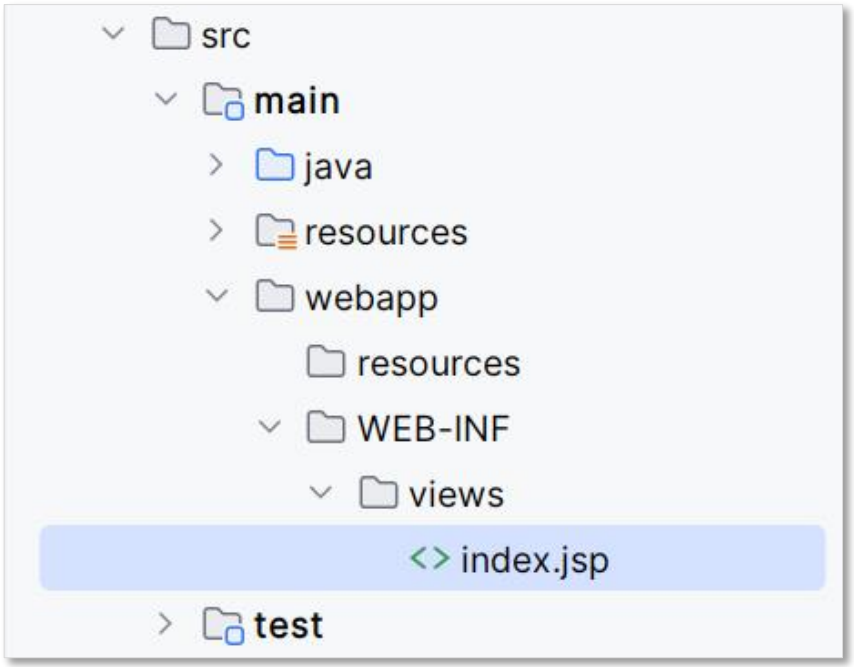
## controller.HomeController.java

```
package org.scoula.controller;

import lombok.extern.log4j.Log4j2;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
@Log4j2
public class HomeController {

    @GetMapping("/")
    public String home() {
        log.info("================> HomController /");
        return "index";                    // View의 이름
    }

}
```

## Jsp 파일 만들기

- src/main/webapp
  - WEB-INF/views/index.jsp

```
@Override
public void configureViewResolvers(ViewResolverRegistry registry) {
  InternalResourceViewResolver bean = new InternalResourceViewResolver();

  bean.setViewClass(JstlView.class);
  bean.setPrefix("/WEB-INF/views/");
  bean.setSuffix(".jsp");

  registry.viewResolver(bean);
}
```

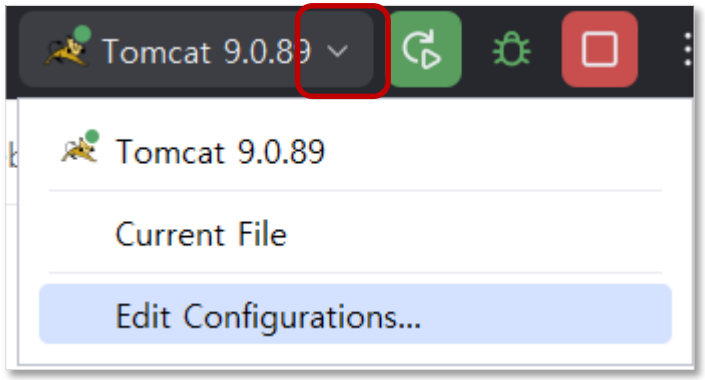



- controller에서 index라는 view 이름을 리턴한 경우
  /WEB-INF/views/ + view 이름 + .jsp
  → /WEB-INF/views/index.jsp

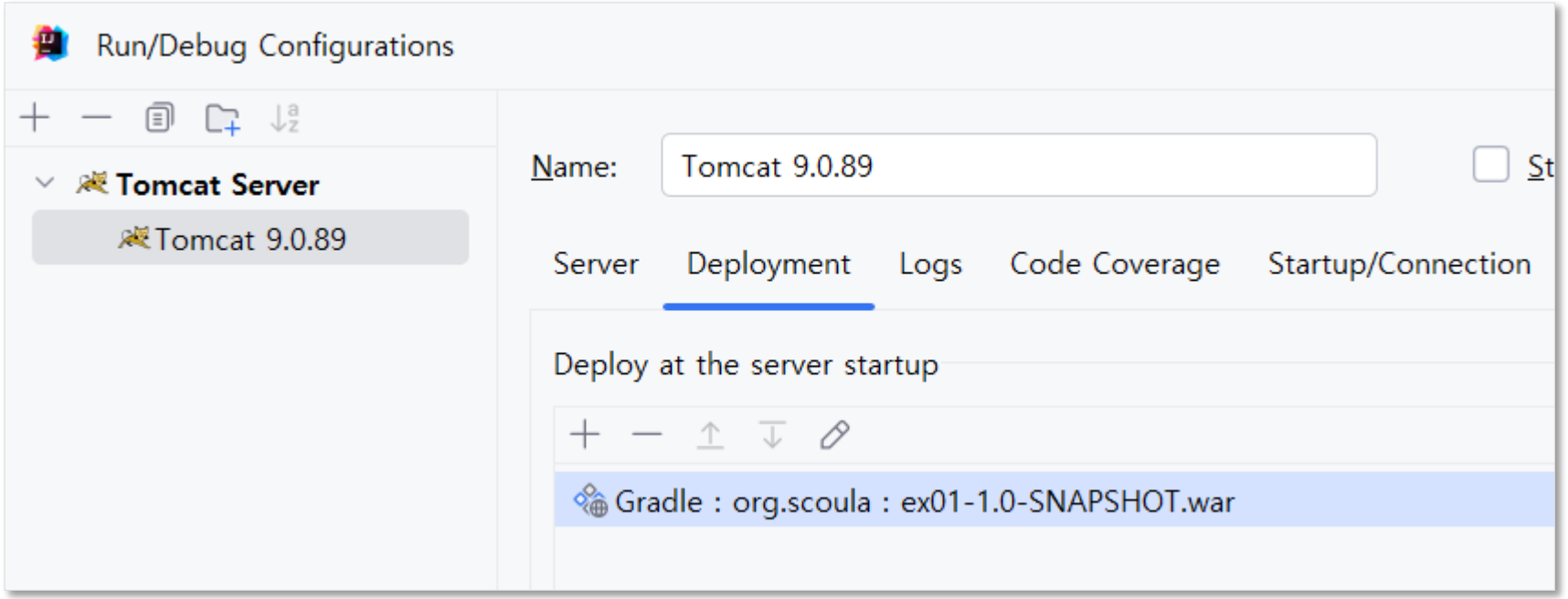
## views/index.jsp

```
<!DOCTYPE html>
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
            <h1>환영합니다.</h1>
</body>
</html>
```

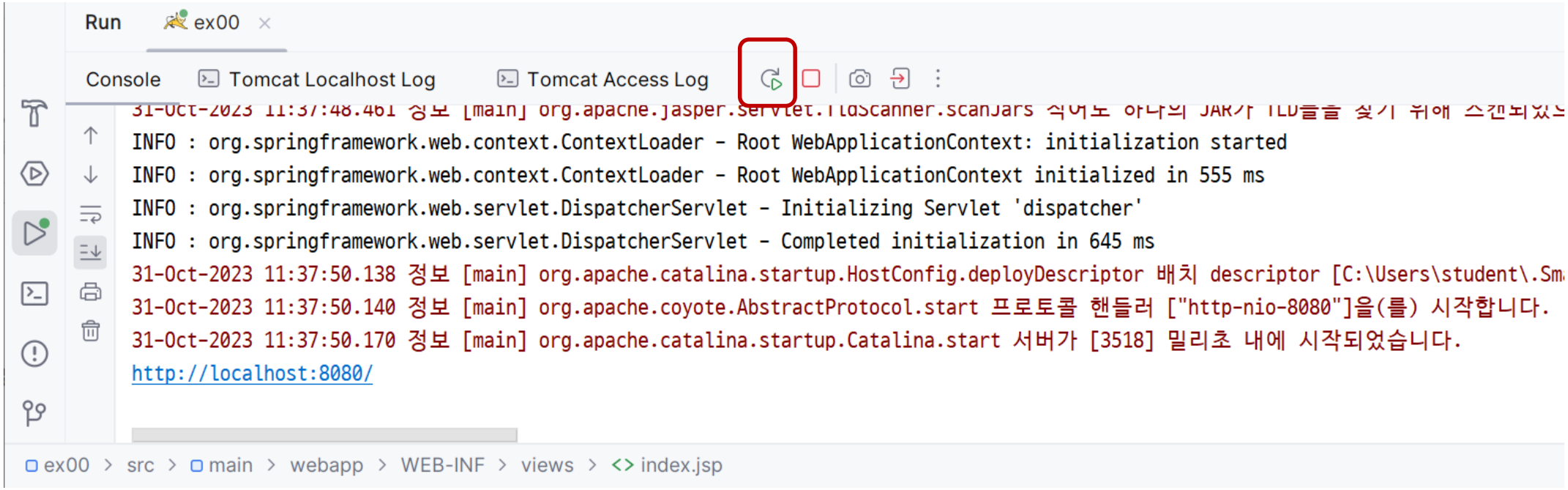
## 톰캣 설정




- Application context: /

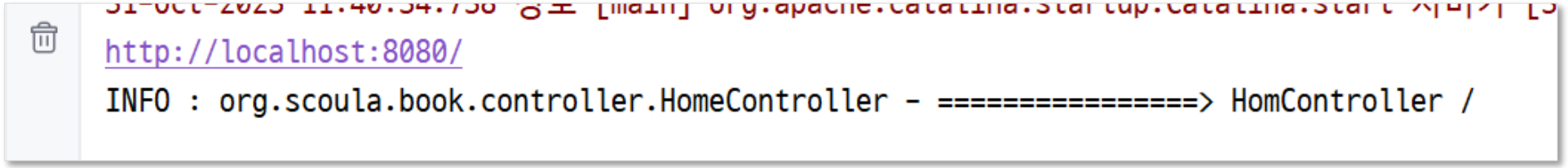


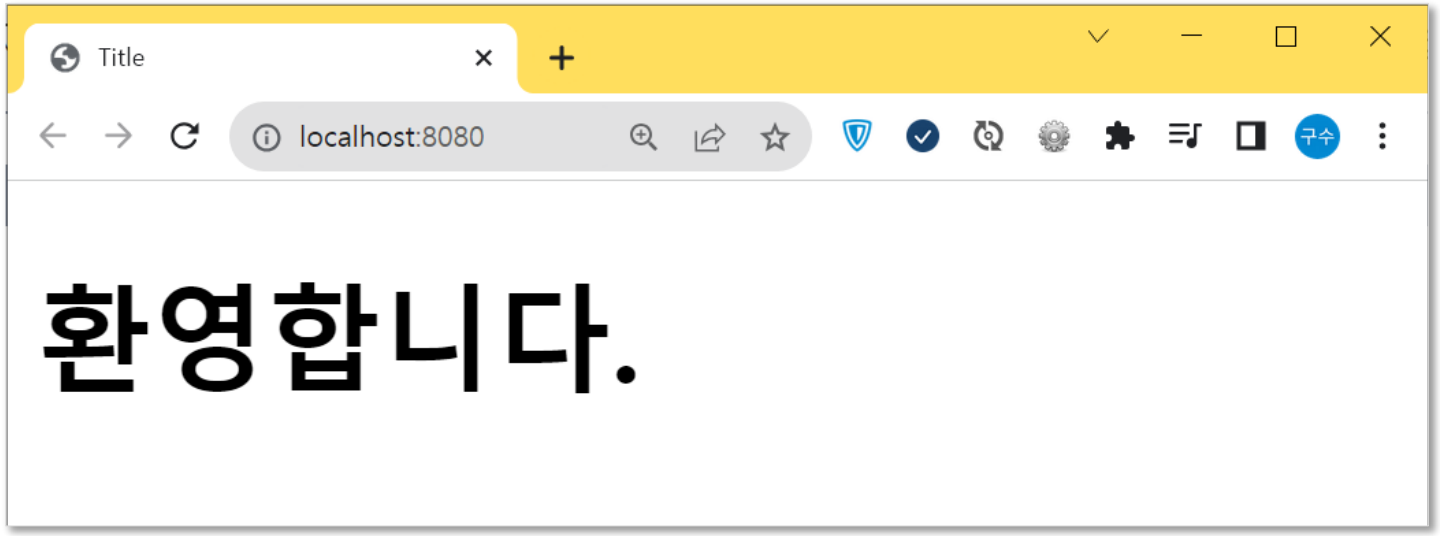
## 애플리케이션 실행


Run
ex00
Console
Tomcat Localhost Log
Tomcat Access Log
INFO : org.springframework.web.context.ContextLoader - Root WebApplicationContext: initialization started
INFO : org.springframework.web.context.ContextLoader - Root WebApplicationContext initialized in 555 ms
INFO : org.springframework.web.servlet.DispatcherServlet - Initializing Servlet 'dispatcher'
INFO : org.springframework.web.servlet.DispatcherServlet - Completed initialization in 645 ms
31-Oct-2023 11:37:50.138 정보 [main] org.apache.catalina.startup.HostConfig.deployDescriptor 배치 descriptor [C:\Users\student\.Sm
31-Oct-2023 11:37:50.140 정보 [main] org.apache.coyote.AbstractProtocol.start 프로토콜 핸들러 ["http-nio-8080"]을(를) 시작합니다.
31-Oct-2023 11:37:50.170 정보 [main] org.apache.catalina.startup.Catalina.start 서버가 [3518] 밀리초 내에 시작되었습니다.
http://localhost:8080/
ex00 > src > main > webapp > WEB-INF > views > index.jsp

## 확인

- http://localhost:8080/



KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

# 개발을 위한 준비 (심화 2)

## [KB] IT's Your Life

KB 국민은행

# 개발을 위한 준비

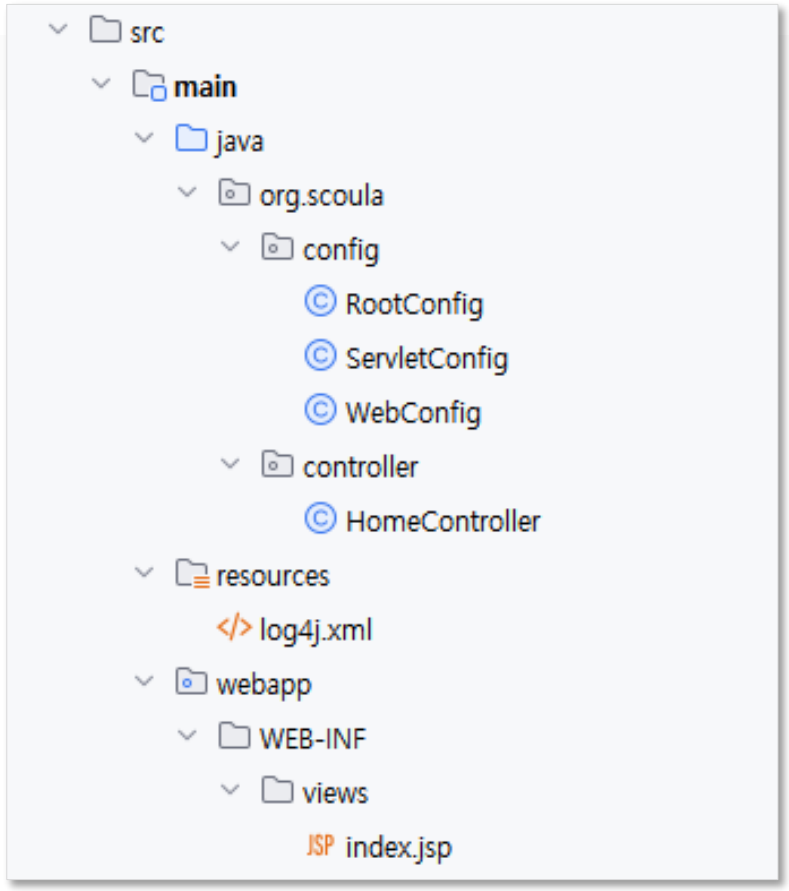
- **ex01 프로젝트를 복사하여 SpringLegacy 프로젝트를 만들고, 이 프로젝트를 프로젝트 템플릿으로 저장하세요.**
  - settings.gradle에서 프로젝트명 변경
  - Tomcat 설정
    - Artifact Id 조정
  - 템플릿 이름: SpringLegacy

## 프로젝트 템플릿 만들기

- 탐색기에서 ex01 프로젝트 폴더를 SpringLegacy이름으로 복사 후 SpringLegacy 프로젝트 열기
- SpringLegacy/settings.gradle

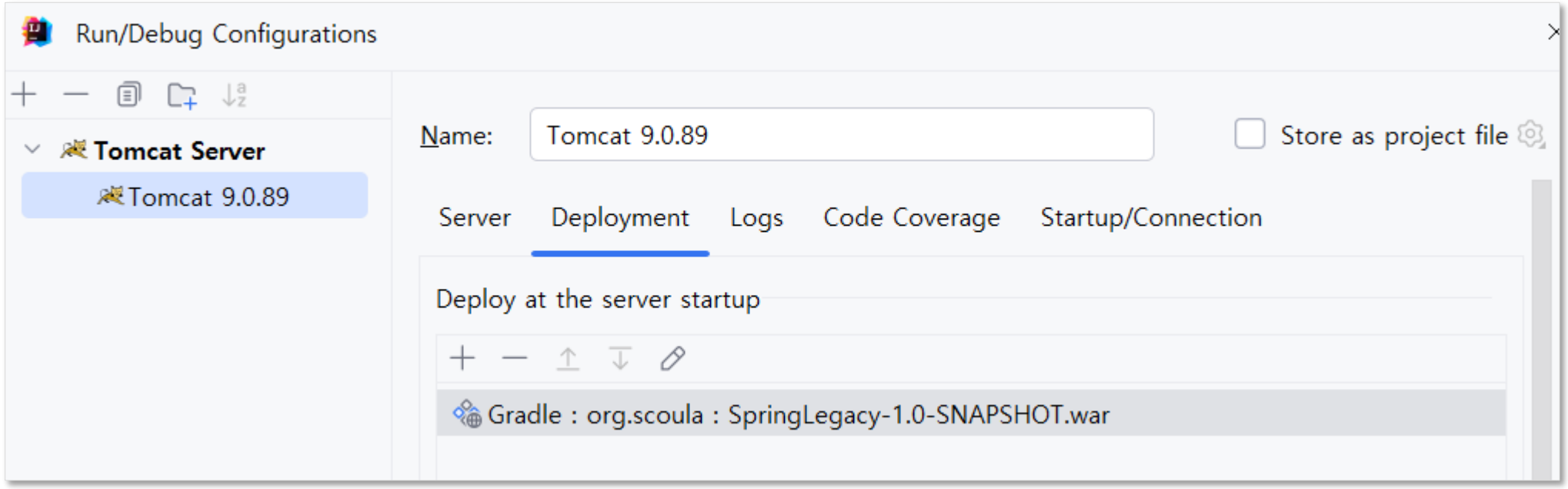
```
rootProject.name = "SpringLegacy"
```

  - 수정 후 Sync



## 톰캣 설정 수정 → Artifact 변경

```
rootProject.name = "SpringLegacy"
```



## 프로젝트 템플릿 만들기

- File > New Projects Setup > Save Project as Template...