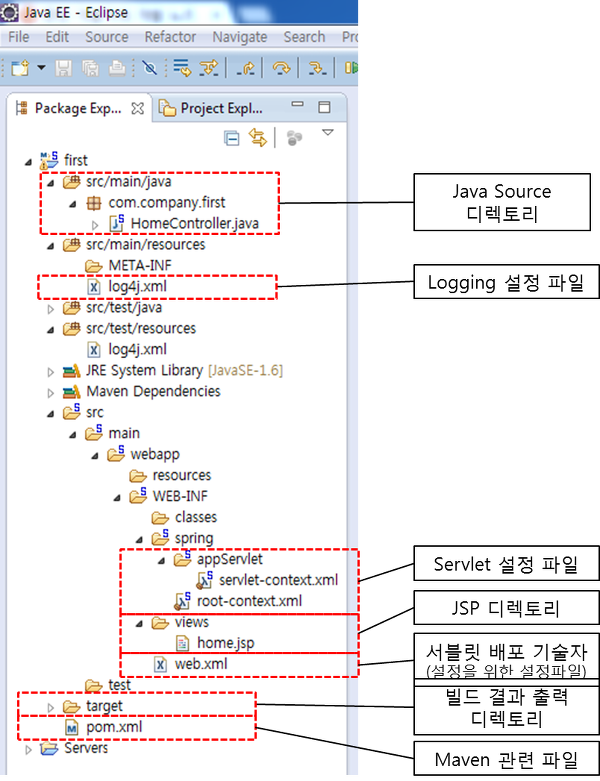
스프링(Spring) 개발 - (5) 스프링 기본 프로젝트 분석  
  
이번 글에서는 지난시간에 만들었던 스프링 프로젝트에 대해서 간단히 분석을 해보겠습니다.

----------------------------------------------------------------------------------------------------

**1. 폴더구조 파악**

우리가 만들었던 first 프로젝트는 다음과 같은 구조를 가지고 있다.



이 구조를 간단히 살펴보자.

1) src/main/java는 java 파일이 모여있는 디렉토리이다. 우리가 앞으로 만들 java 파일은 전부 이 디렉토리에 구성된다.

2) src/main/resources는 여기서는 사용하지 않지만, 추후 스프링 설정 파일이나 쿼리가 저장될 디렉토리이다.

3) src/test/ 관련 폴더는 test 관련 폴더인데, 우리는 TDD(Test Driven Development) 방법론이나 테스트코드를 따로 작성하는 방식은 아직까지는 사용하지 않기 때문에, 현재로써는 필요없는 폴더이다. 과감히 삭제.

4) 우리는 메이븐을 사용하는데, 메이븐의 기본 폴더는 src/main/webapp 폴더가 기본 폴더이다. webapp 폴더 밑에 모든 jsp 및 js 등의 파일이 포함된다.

5) servlet-context.xml, root-context.xml은 서블릿(Servlet)관련 설정파일이다.

**2. HomeController.java**

Controller는 웹 클라이언트에서 들어온 요청을 해당 비지니스 로직으로 분기시켜주고, 수행결과의 응답을 해주는 Dispatcher의 역할을 담당하는 클래스이다. 스프링의 MVC 구조는 다음 글에서 좀 더 자세하게 설명하기로 하고, 여기서는 간단히 소스를 살펴보기만 한다.

@RequestMapping(value = "/", method = RequestMethod.GET)

public String home(Locale locale, Model model) {

logger.info("Welcome home! The client locale is {}.", locale);

Date date = new Date();

DateFormat dateFormat = DateFormat.getDateTimeInstance(DateFormat.LONG, DateFormat.LONG, locale);

String formattedDate = dateFormat.format(date);

model.addAttribute("serverTime", formattedDate );

return "home";

}

먼저 @RequestMapping 이라는 부분이 웹 클라이언트 (jsp)에서 들어온 요청에 해당하는 비지니스 로직을 찾아주는 역할을 한다. 뒤에 method는 이 요청이 POST 인지, GET 방식인지를 말해주는데, 우리는 앞으로 거의 대부분을 POST로 보낼것이기 때문에, 이 부분은 지울 계획이다.

그 다음은 return "home"; 이라는 부분이다. 이 부분은 수행결과의 응답을 어디로 보낼지를 명시해준다. 나중에 서블릿(Servlet) 설정에서 다시 설명하겠지만, "home"이라는 것은 jsp 파일명을 의미한다.

서블릿 설정에서 자동으로 앞에 "/WEB-INF/views"를 붙여주고 (prefix),

뒤에 ".jsp"를 붙여주도록 되어있다.(suffix)

따라서 우리가 위에서 본 src/main/webapp/WEB-INF/views/home.jsp가 호출되게 되는 것이다.

세번째로, model.addAttribute("serverTime", formattedDate); 부분이다.

이는 비지니스 로직에서 수행한 결과를 화면으로 보내주기 위한 부분이다. serverTime이라는 이름으로 formattedDate를 전송함을 의미한다. 이를 사용하는 방법은 home.jsp에서 이야기 한다.

**3. home.jsp**

home.jsp는 다음과 같이 작성되어있다.

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<%@ page session="false" %>

**Hello world!**

The time on the server is ${serverTime}.

첫번째로 볼 부분은 The time on the server is ${serverTime}. 이라는 부분이다.

우리는 Controller에서 model.addAttribute("serverTime", formattedDate); 라는 방식으로 화면으로 결과값을 보내주었다고 했었다. 여기서 serverTime이라는 이름으로 보낸 부분이 ${serverTime}이라는 방식으로 사용됨을 볼 수 있다.

${} 는 EL(Expression Language)를 사용한 부분이다. EL에 대한 설명은 인터넷에서 쉽게 찾아볼 수 있는 관계로, 바로 넘어가도록 한다.

${serverTime}이라는 부분이 서버에서 넘어온 결과를 화면에 보여준다.

**3. web.xml**

web.xml을 위에서 서블릿 배포 기술자라고 했다. 영어로는 DD (Deploment Descriptor)라고 한다. web.xml은 WAS (Web Application Server)(여기서는 Tomcat)이 최초 구동될 때, WEB-INF 디렉토리에 존재하는 web.xml을 읽고, 그에 해당하는 웹 애플리케이션 설정을 구성한다. 다시 말해, 각종 설정을 위한 설정파일이라고 이야기 할 수 있다.

web.xml은 나중에 수정을 할 부분이 있는데, 그때 조금 더 자세히 설명하도록 한다.

**4. servlet-context.xml**

servlet-context는 서블릿 관련 설정이다. 우리가 여기서 주목해야하는 부분은

<beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<beans:property name="prefix" value="/WEB-INF/views/" />

<beans:property name="suffix" value=".jsp" />

</beans:bean>

↑ 이 부분이다.

Controller을 설명할 때, 서블릿 설정이 자동으로 prefix와 suffix를 붙인다고 해줬는데, 그 역할을 담당한다. 즉, 우리가 일일이 전체경로와 .jsp를 붙이지 않아도 되도록 도와준다.

그 다음은 <context:component-scan base-package="com.company.first" /> 이다.

이 부분은 스프링에서 사용하는 bean을 일일이 xml에 선언하지 않고도 필요한 것을 어노테이션(Annotation)을 자동으로 인식하게 하는 역할을 한다. 이는 나중에 다시 설명하도록 한다.

----------------------------------------------------------------------------------------------------

이번 글에서는 지난 시간에 만들었던 first 프로젝트를 간단히 살펴봤습니다.

물론 이 모든게 한번에 이해하기 힘들겁니다. 이번에는 그냥 간단히 보고 넘어가는 식으로, 그냥 이런게 있구나~ 라고 생각하시면 됩니다. 다음 글에서부터는 본격적으로 스프링을 사용한 프로젝트를 구성할 예정입니다. 상당히 많은 양이 예정되어 있는데, 여기서 나왔던 부분들이 하나씩 하나씩 좀 더 자세하게 설명됩니다.

출처: <https://addio3305.tistory.com/37?category=772645> [흔한 개발자의 개발 노트]