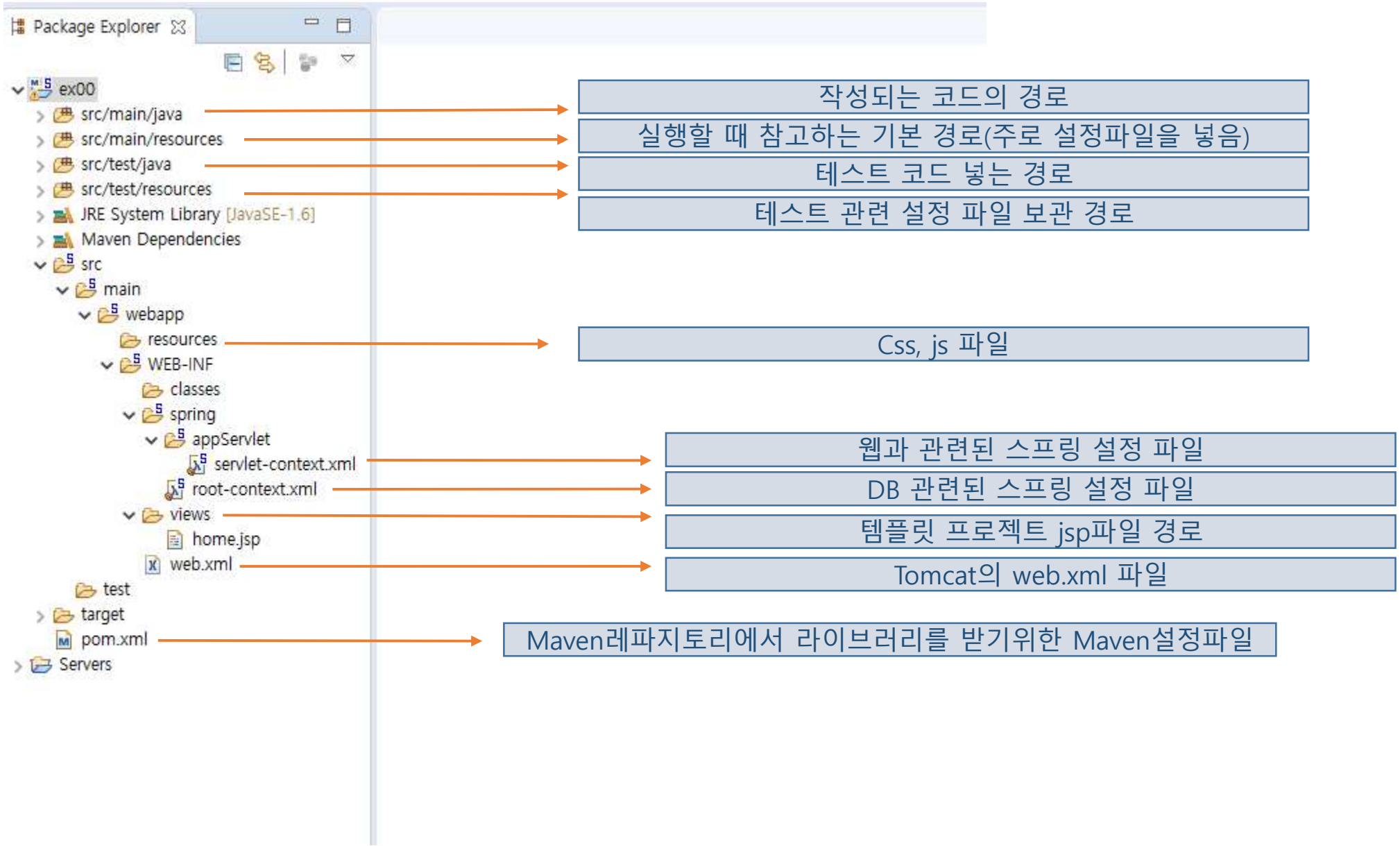
Spring Framework

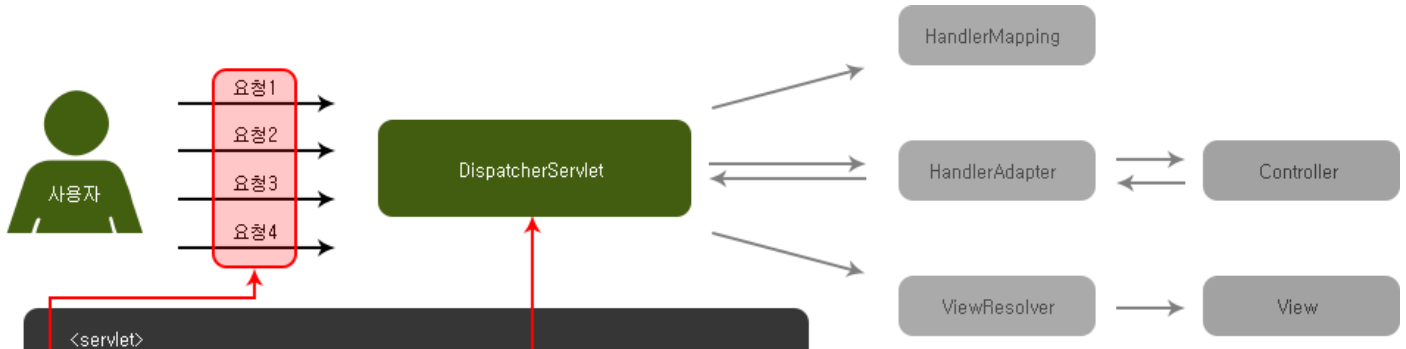
스프링 MVC웹서비스

1. 프로젝트 전체구조
2. web.xml
3. DispatcherServlet
4. servlet-context.xml
5. Controller 컨트롤러
6. View 뷰

1 : 프로젝트 전체구조


Package Explorer
ex00
src/main/java
src/main/resources
src/test/java
src/test/resources
JRE System Library [JavaSE-1.6]
Maven Dependencies
src
main
webapp
resources
WEB-INF
classes
spring
appServlet
servlet-context.xml
root-context.xml
views
home.jsp
web.xml
test
target
pom.xml
Servers
작성되는 코드의 경로
실행할 때 참고하는 기본 경로(주로 설정파일을 넣음)
테스트 코드 넣는 경로
테스트 관련 설정 파일 보관 경로
Css, js 파일
웹과 관련된 스프링 설정 파일
DB 관련된 스프링 설정 파일
템플릿 프로젝트 jsp파일 경로
Tomcat의 web.xml 파일
Maven레파지토리에서 라이브러리를 받기위한 Maven설정파일

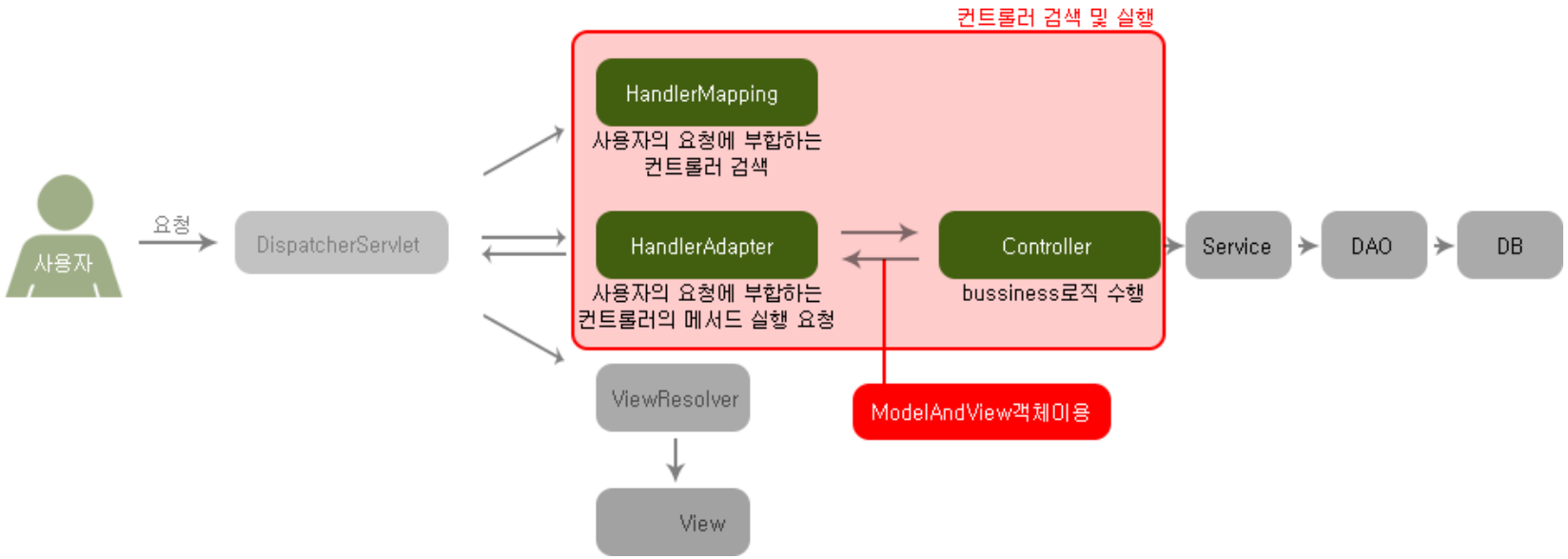
2 :web.xml




```
<servlet>
    <servlet-name>appServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
    <servlet-name>appServlet</servlet-name>
    <url-pattern>/</url-pattern>
</servlet-mapping>
```

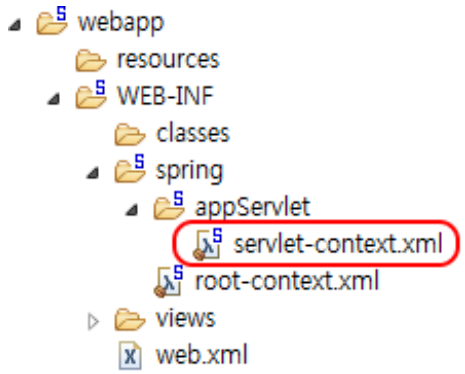
웹 어플리케이션에서 최초 사용자의 요청이 발생하면 가장먼저 DispatcherServlet이 사용자의 요청을 받는다고 하였다. 따라서 개발자는 DispatcherServlet을 서블릿으로 등록해주는 과정을 설정해 주어야 한다. 그리고 사용자의 모든 요청을 받기 위해서 서블릿 맵핑 경로는 '/'로 설정한다.

3 : DispatcherServlet



사용자의 모든 요청을 DispatcherServlet이 받은 후 HandlerMapping 객체에 Controller 객체 검색을 요청한다. 그러면 HandlerMapping 객체는 프로젝트에 존재하는 모든 Controller 객체를 검색한다. HandlerMapping 객체가 Controller 객체를 검색해서 DispatcherServlet 객체에 알려주면 DispatcherServlet 객체는 다시 HandlerAdapter 객체에 사용자의 요청에 부합하는 메소드 검색을 요청한다. 그러면 HandlerAdapter 객체는 사용자의 요청 에 부합하는 메소드를 찾아서 해당 Controller 객체의 메소드를 실행한다. Controller 객체의 메소드가 실행된 후 Controller 객체는 HandlerAdapter 객체에 ModelAndView 객체를 반환하는데 ModelAndView 객체에는 사용자 응답에 필요한 데이터정보와 뷰정보(JSP파일)가 담겨있다. 다음으로 HandlerAdapter 객체는 ModelAndView 객체를 다시 DispatcherServlet 객체에 반환한다.

4 : servlet-context.xml



프로젝트 구조

```
<servlet>
    <servlet-name>appServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
```

web.xml

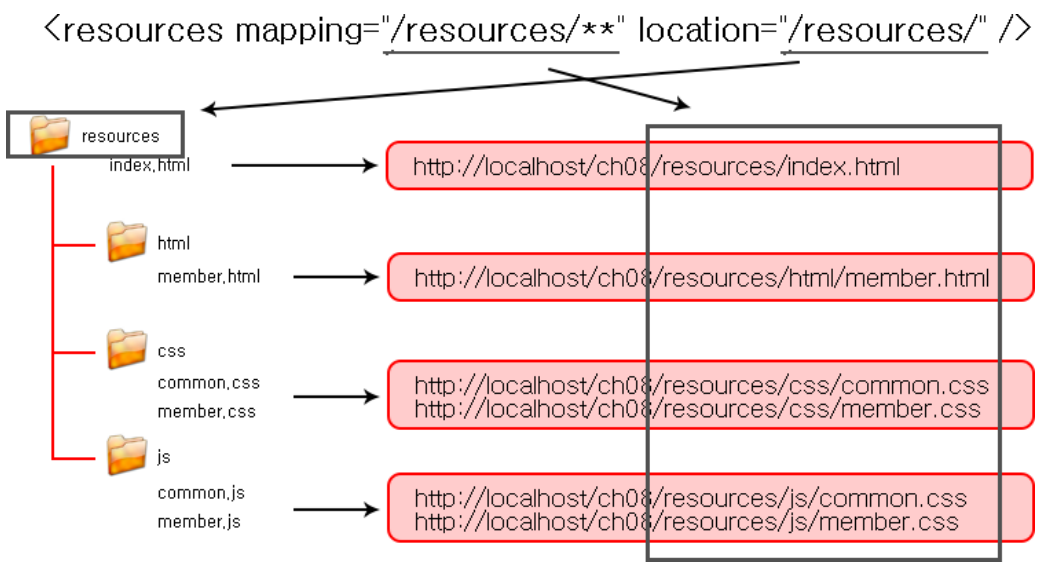
앞에서 DispatcherServlet를 서블릿으로 등록하는 과정을 살펴봤다. 서블릿으로 등록 될 때 contextConfigLocation이름으로 초기화 파라미터를 servlet-context.xml로 지정하고 있는데 이때 지정된 servlet-context.xml파일이 스프링 설정의 역할을 하는 파일이다.

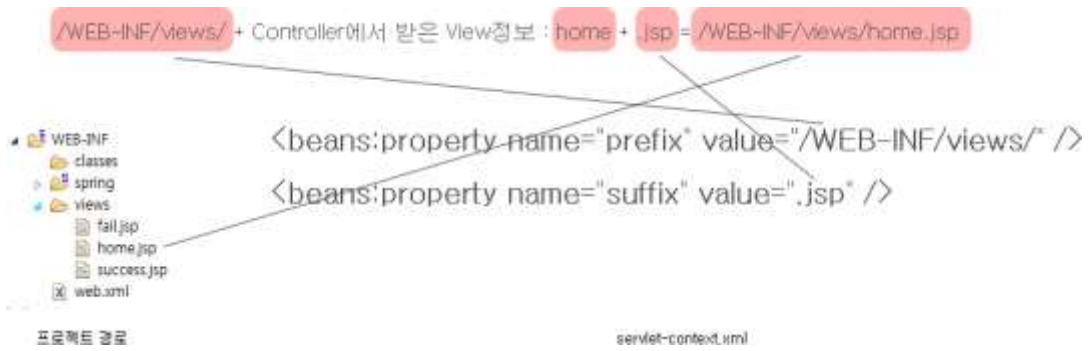
4 : servlet-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:be
ans="http://www.springframework.org/schema/beans" xmlns:contex
t="http://www.springframework.org/schema/context"
xsi:schemaLocation="http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd
http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd http://www.spri
ngframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">

<!-- DispatcherServlet Context: defines this servlet's request-processing infrastructure -->

<!-- Enables the Spring MVC @Controller programming model -->
<annotation-driven />

<!-- Handles HTTP GET requests for /resources/** by efficiently serving up static resources in the ${webappRoot}/resources directory -->
<resources mapping="/resources/**" location="/resources/" />

<!-- Resolves views selected for rendering by @Controllers to .jsp resources in the /WEB-INF/views directory -->
<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>

<context:component-scan base-package="com.bs.lec14" />

</beans:beans>
```

스프링 설정 파일은 클래스로부터 객체(빈:bean)를 생성하고 조립하는 역할을 한다고 학습했다. servlet-context.xml에서도 마찬가지로 프로젝트에 필요한 객체(빈:bean)를 생성하고 조립한다.

4 :servlet-context.xml

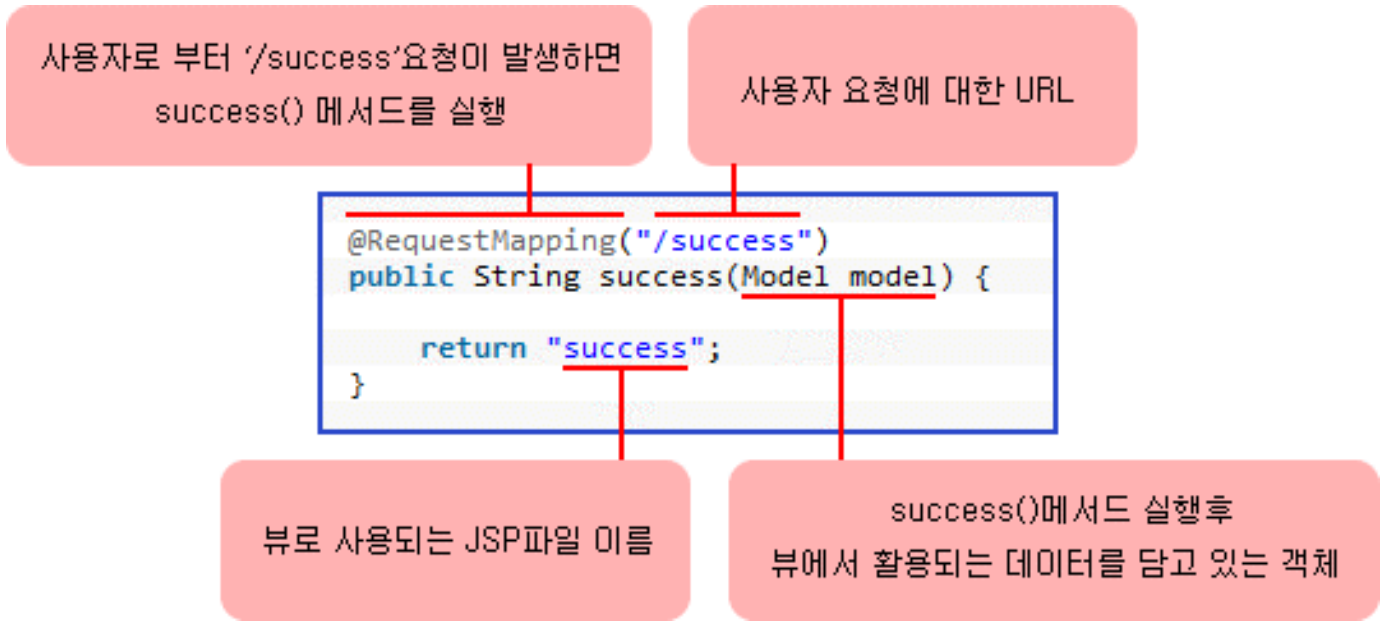




5 :Controller(컨트롤러)


```
@Controller
public class HomeController {

  @RequestMapping(value = "/", method = RequestMethod.GET)
  public String home(Locale locale, Model model) {

    Date date = new Date();
    DateFormat dateFormat = DateFormat.getDateTimeInstance(DateFormat.LONG, DateFormat.LONG, locale);

    String formattedDate = dateFormat.format(date);

    model.addAttribute("serverTime", formattedDate );

    return "home";
  }

}
```

5 : Controller(컨트롤러)

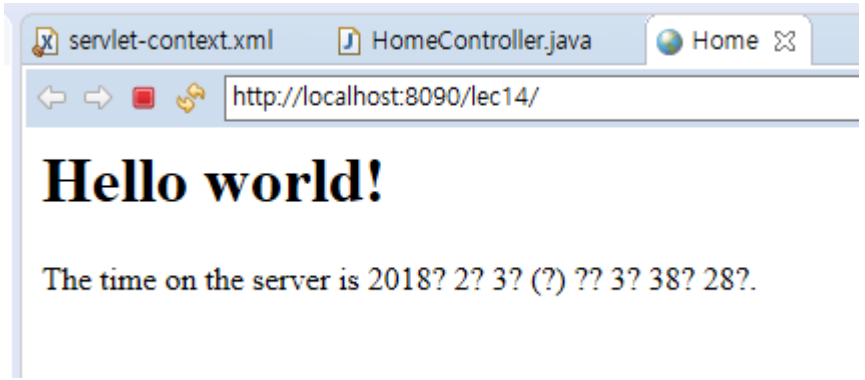


6 :View(뷰)

메서드 반환값 → `return "home";`

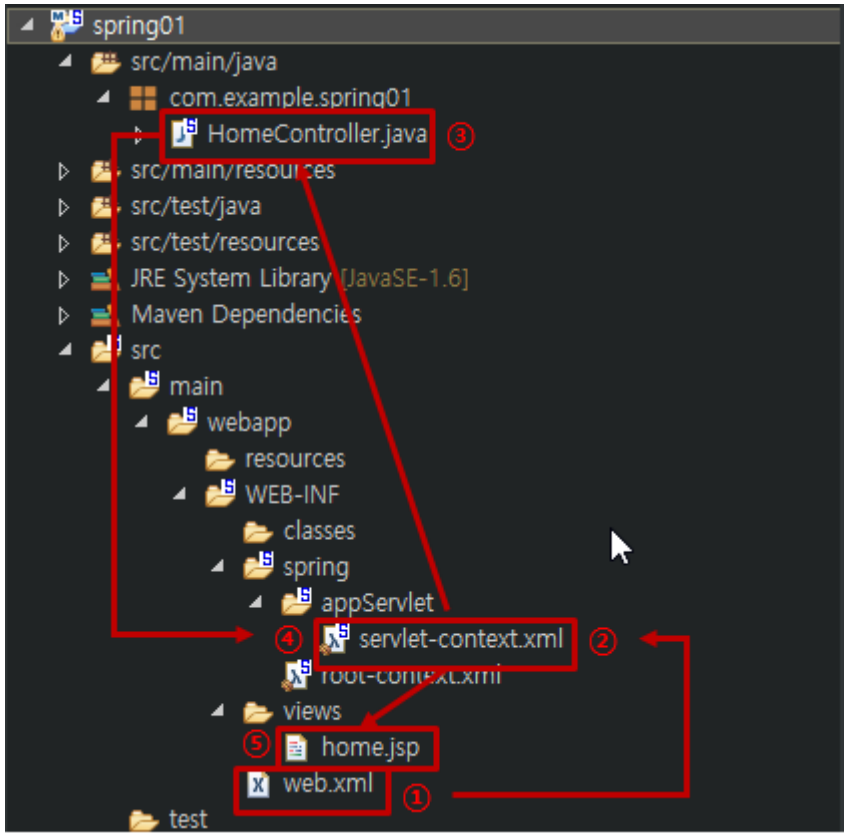
JSP 파일 → `/WEB-INF/views/home.jsp`

사용자 응답 브라우저 →



클라이언트 요청 정보(URL맵핑값)에 해당하는 JSP파일 실행

1. home.xml의 구동 과정



1. 클라이언트 요청
2. web.xml에서 dispatcherServlet이 요청 핸들링

3. servlet-context.xml이 요청에 대한 컨트롤러 검색
   (HandlerMapping으로 Controller검색)

4. 컨트롤러 요청 처리 후, home을 리턴
5. view resolver가 받은 home을 찾아서 처리