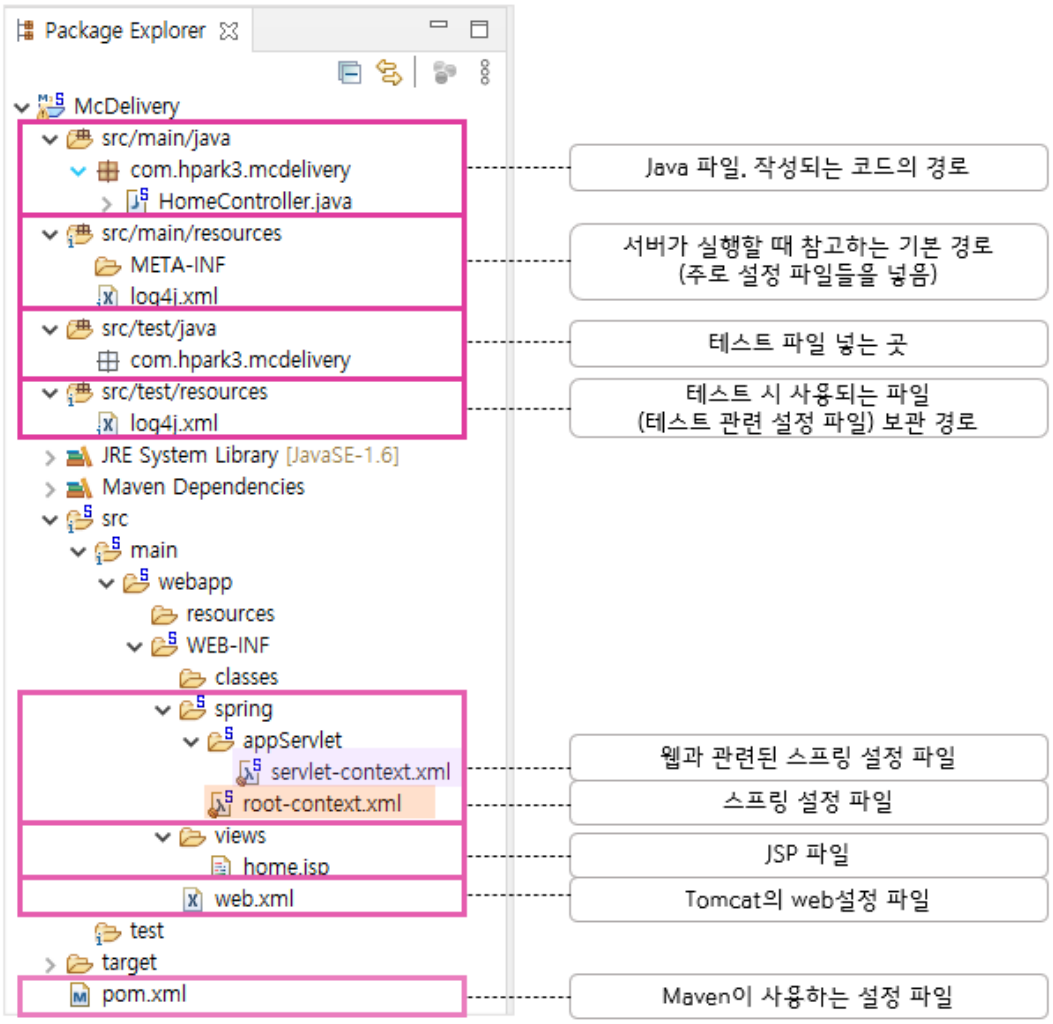
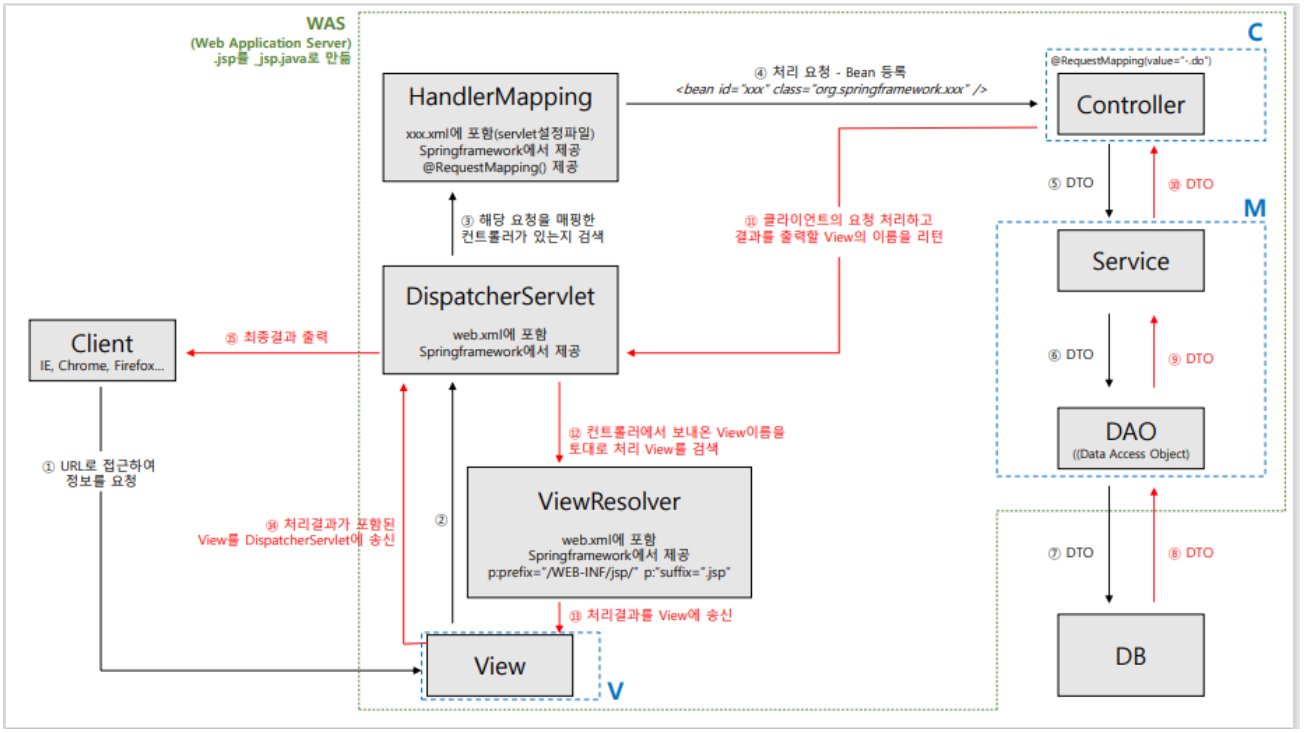
1. 스프링 MVC 프로젝트의 기본 구조



2. 스프링 프레임워크의 전체 동작 순서

Request -> **DispatcherServlet** (web.xml)-> **HandlerMapping** (servlet-context.xml) -> **Controller** [Controller -> Service -> DAO -> DB -> DAO -> Service -> Controller] -> **DispatcherServlet** -> **ViewResolver** -> **View** -> Response



출처: https://server-engineer.tistory.com/253

① 클라이언트가 Request 요청을 하면 **DispatcherServlet**이 요청을 가로챈다.

이때 DispatcherServlet이 모든 요청을 가로채는건 아니고 web.xml에 <url-pattern>에 등록된 내용만 가로챈다.

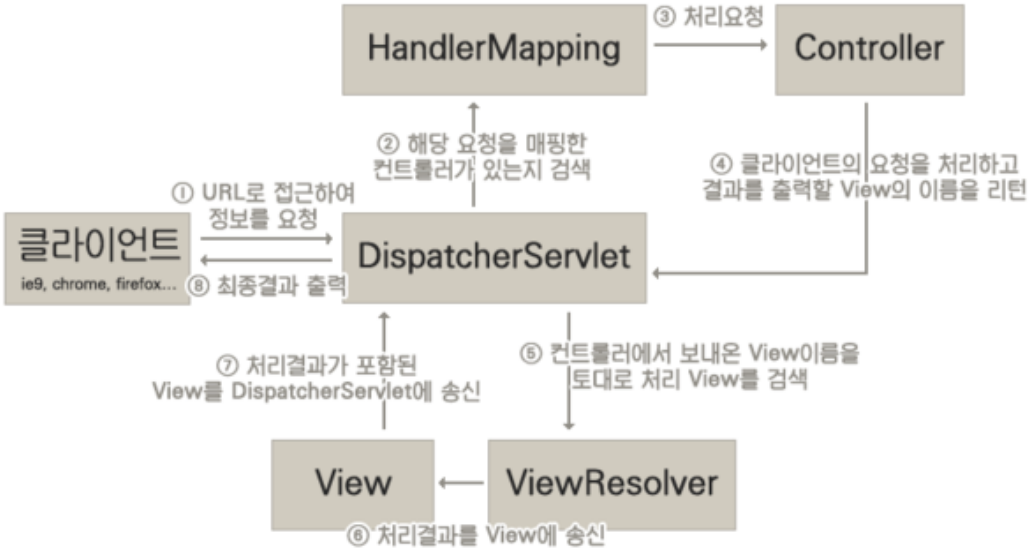
② DispatcherServlet이 가로챈 요청을 **HandlerMapping**에게 보내 해당 요청을 처리할 수 있는 **Controller**를 찾는다.

③ 실제 로직 처리 (Controller -> Service -> DAO -> DB -> DAO -> Service -> Controller)

④ 로직 처리 후, **DispatcherServelet**에 View이름을 리턴한다.

⑤ **ViewResolver**(p:prefix="/WEB-INF/jsp/" p:"suffix=".jsp")를 통해 결과를 출력할 **View** 화면을 검색한다.

⑥ 처리결과를 View에 송신하고, View화면을 최종 클라이언트에게 전송한다.



출처: http://egloos.zum.com/springmvc/v/504151

* DispatcherServlet (web.xml에 설정)

웹요청이 들어오면 **DispatcherServelet**객체가 요청을 어떤 컨트롤러에게 위임할 지 결정한다. 즉, **디스패처서블릿**이 모든 클라이언트의 요청을 받고, 세부경로를 각 컨트롤러에게 뿌리는 역할을 하기 때문에, web.xml에 모든 서블릿을 일일이 다 등록할 필요가 없어진다. (☞ 더 알아보기)

web.xml은 WAS (Web Application Server: 여기서는 Tomcat)가 최초 기동될 때, WEB-INF 디렉토리에 존재하는 web.xml을 읽고, 그에 해당하는 웹 애플리케이션 설정을 구성한다.

* HandlerMapping (servlet-context.xml에 설정)

핸들러매핑은, 웹/클라이언트의 요청과 해당요청을 처리하는 Controller의 매핑을 담당하는 인터페이스이다. (☞ 더 알아보기)

servlet-context.xml는 요청과 관련된 객체이다.

url과 관련된 **controller**나 **@(어노테이션), ViewResolver, Interceptor, MultipartResolver** 등의 설정을 해준다.

```
<!-- Controller의 메서드에서 반환하는 문자열 앞 뒤에 붙힐 경로 정보를 셋팅 -->
<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>

<!-- 스캔할 bean들이 모여있는 패키지를 지정 -->
<context:component-scan base-package="com.hpark3.mcdelivery" />
```

위와같이 서블릿에서, prefix(접두사)로 경로를 suffix(접미사)로 확장자를 붙여주기 때문에

컨트롤러에서 일일이 전체경로를 입력하거나 .jsp를 붙이지 않아도 되도록 도와준다.

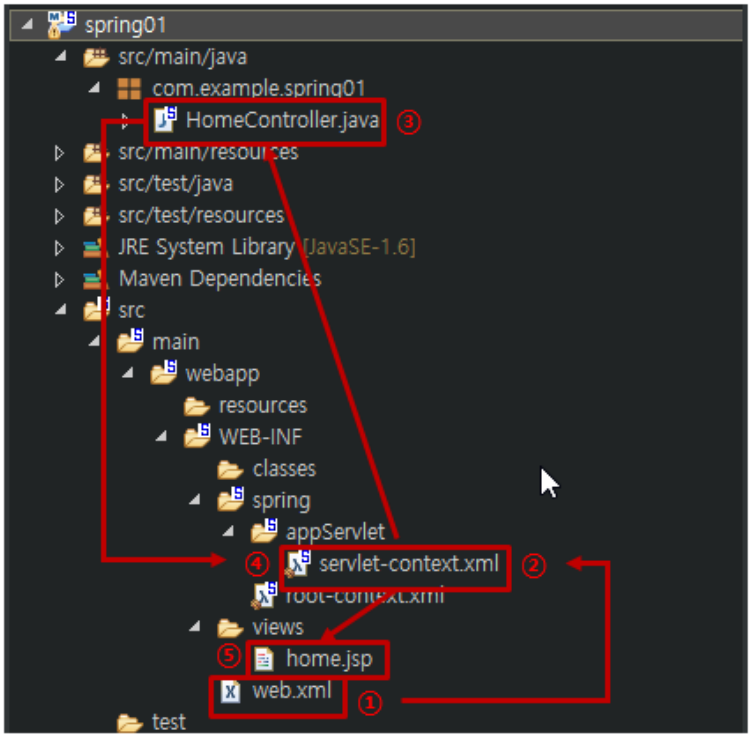
또한, 스캔할 bean들이 모여있는 패키지를 지정함으로써

스프링에서 사용하는 bean을 일일이 xml 에 선언하지 않고도 필요한 것을 어노테이션(@)으로 자동 인식하게 한다.

*** root-contex.xml**

view와 관련되지 않은 객체를 정의하고, 따라서 **Service, Repository(DAO), DB** 등 비즈니스 로직과 관련된 설정을 해준다.

스프링 프로젝트를 생성하고 바로 실행하면 브라우저에 home.jsp가 실행된다.
여기서 home.jsp가 구동되는 과정은 아래와 같다.



1. 클라이언트 요청(/, root 페이지 요청)
2. web.xml에서 dispatcherServlet가 클라이언트 요청을 핸들링
3. servlet-context.xml에서 해당 클래스의 웹요청을 처리하는 컨트롤러를 사용(HandlerMapping으로 Controller를 검색)
4. 해당 Controller가 요청을 처리후, home을 리턴
5. View에 출력

출처: https://doublesprogramming.tistory.com/84