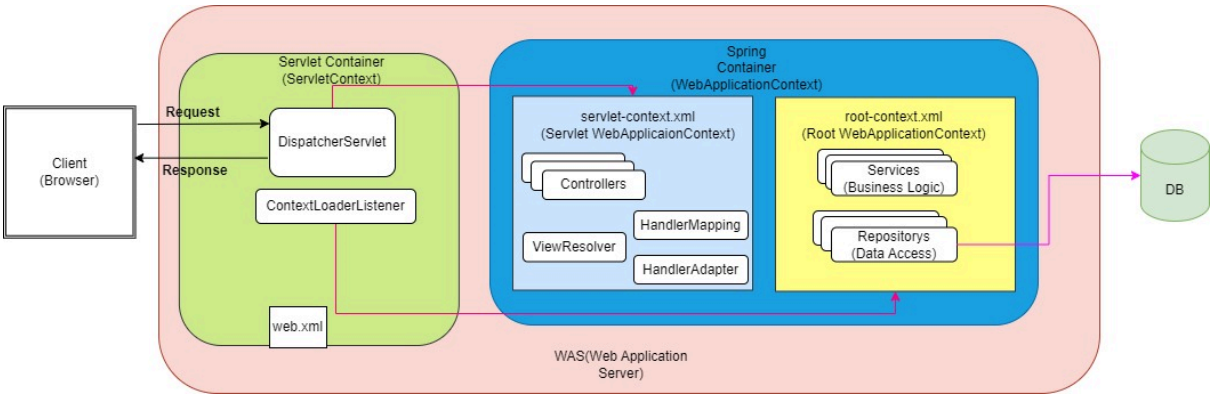
SPRING 구조 및 동작 순서

1. Spring 실행 순서



1. 웹 어플리케이션 실행 시 WAS(Tomcat)이 web.xml 파일 loading

2. web.xml의 ContextLoaderListener 생성 (Java Class)

1. 서블릿 컨테이너가 파일을 읽고 구동되면
2. ContextLoaderListener 가 메모리에 생성
3. ContextLoaderListener 클래스는 ServletContextListener 인터페이스를 구현 → ApplicationContext를 생성

4. ContextLoaderListener 클래스는 Servlet의 생명 주기를 관리
5. Servlet이 사용되는 시점에 ApplicationContext 등록, Servlet이 종료되는 시점에 ApplicationContext 삭제

```
<!-- web.xml 파일 내용 중 -->
<!-- Creates the Spring Container shared by all Servlets and Filters -->
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

3. 생성된 ContextLoaderListener - root-context.xml 파일 loading

4. root-context.xml의 Spring Container가 구동

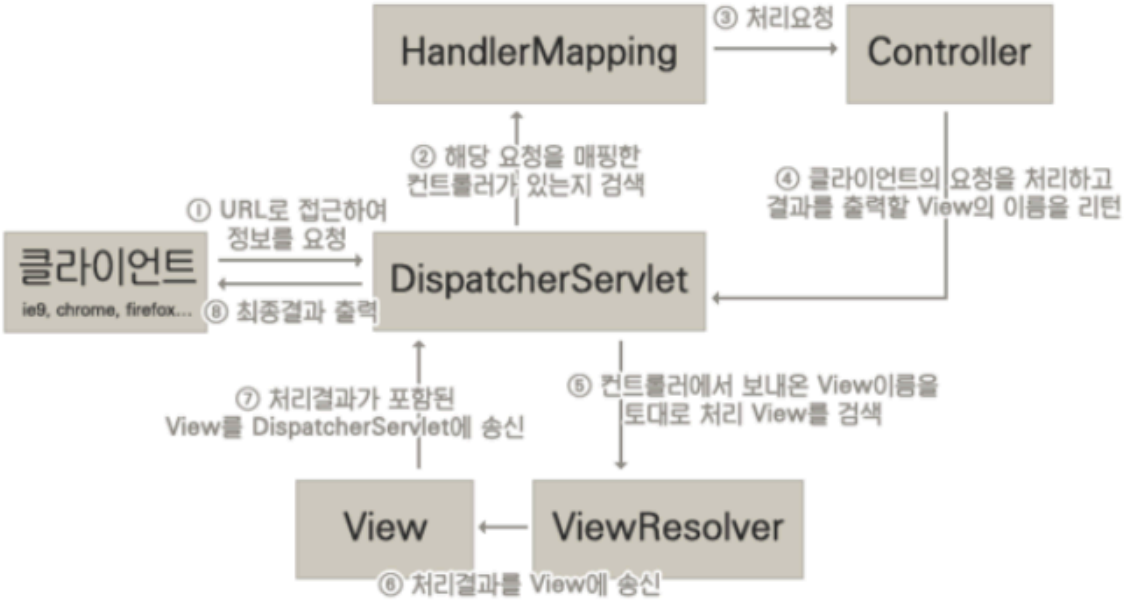
1. root-context.xml 는 주로 공동 bean을 설정 (Service, DBSource, repository 등)

5. DispatcherServlet 생성 - servlet-context.xml 파일 loading

1. SpringContainer가 Controller 객체를 메모리에 생성
2. DispatcherServlet은 FrontController 역할

```
<!-- web.xml 파일 내용 중 -->
<!-- Processes application requests -->
<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>
```

2. 구동되는 구조



1. 클라이언트가 어플리케이션에 접근하는 URL을 DispatcherServlet이 가로챔
2. RequestMappingHandlerMapping이 요청에 맞는 Controller를 찾음
3. Controller에서 로직 처리
4. Controller에서 view를 리턴하면 ViewResolver에서 view가 존재하는지 검색
5. DispatcherServlet은 ViewResolver를 통해 JSP파일의 이름과 경로를 리턴 받아 최종적으로 JSP를 실행