하나씩 쉽게 따라 해보는 IBM WebSphere Application Server(WAS) v7 – 10

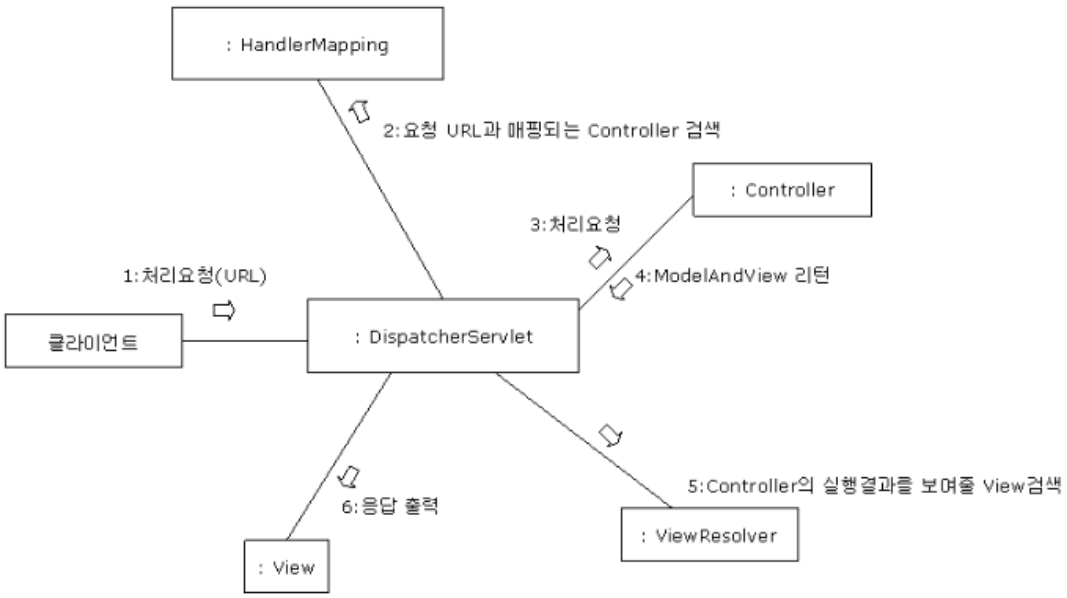
이정운 (juwlee@kr.ibm.com)

하나씩 쉽게 따라 해보는 IBM WAS v7 시리즈 1부는 이미 끝났지만 그 번외편으로 열 번째 이야기를 시작하려고 합니다. 사실 하나씩 쉽게 따라 해보는 IBM WAS v7 시리즈는 지난 아홉번째 편으로 끝을 냈지만, 요즘 Trend 이며 제 관심대상중의 하나가 바로 Open Framework 이라 간단하게 어떻게 사용할 수 있는지를 추가할려고 합니다. 앞에서 이야기 했듯 이 열 번째 시리즈는 번외편이니 반드시 따라하실 필요는 없습니다.

이번 편에서 다룰 것은 오픈 프레임웍 중에서도 Spring 2.5 의 Spring MVC(Model, View & Control) 기능을 WAS 에서 적용하는 것 입니다. Spring 은 그동안의 표준처럼 사용되던 J2EE 가 가지고 있는 여러 단점을 어떻게 하면 해결 할 수 있을까 고민한 끝에 나오기 시작했던 여러 오픈 프레임웍의 일종입니다. 또 다른, 대표적인 프레임웍에는 Struts 가 있죠. J2EE 가 가지고 있는 가장 큰 문제점인 무겁고 Container 의존적인 점을 해결하고자 IoC (Inversion of Control) 개념을 접목하여 Container 가 아닌 사용자에게 주체권을 가지고 오게 하고 이를 프레임웍 내에서 DI(Dependancy Ingection) 와 AOP(Aspect of Programming) 등의 개념을 이용해서 해결해주게 끔 하죠. 다시말해서 트랜잭션, 보안, DB 연결등을 Container 가 아니라 프레임웍 단으로 내려서 처리합니다. 이렇게 함으로써 가장 큰 장점은 EJB 를 사용할 필요없이 POJO 를 이용해서 Business logic 을 처리할 수 있습니다. 위에서 이미 설명한 것처럼, EJB 에서 가졌던 Container 가 하는 역할을 프레임웍이 대신처리할 수 있기 때문이죠

뭐, 이런 프레임웍에 관련된 내용은 책이나 인터넷에 많이 있고 이 강좌는 WAS 를 위한 것이므로 이 부분은 그만 각설하고 저희가 해볼 Spring MVC 에 대해서만 살펴보도록 하겠습니다.

[스프링 MVC의 처리 흐름]

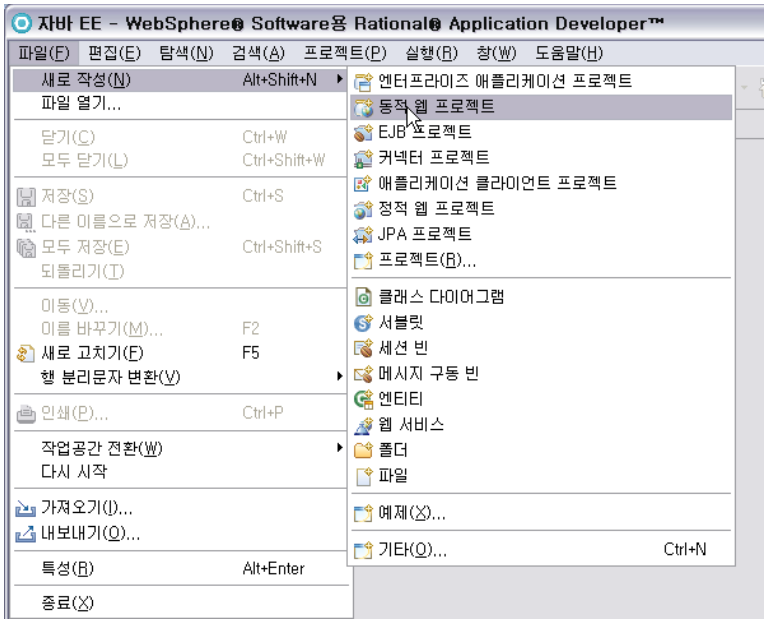


(그림 참조 : http://gomphoto.tistory.com/163)

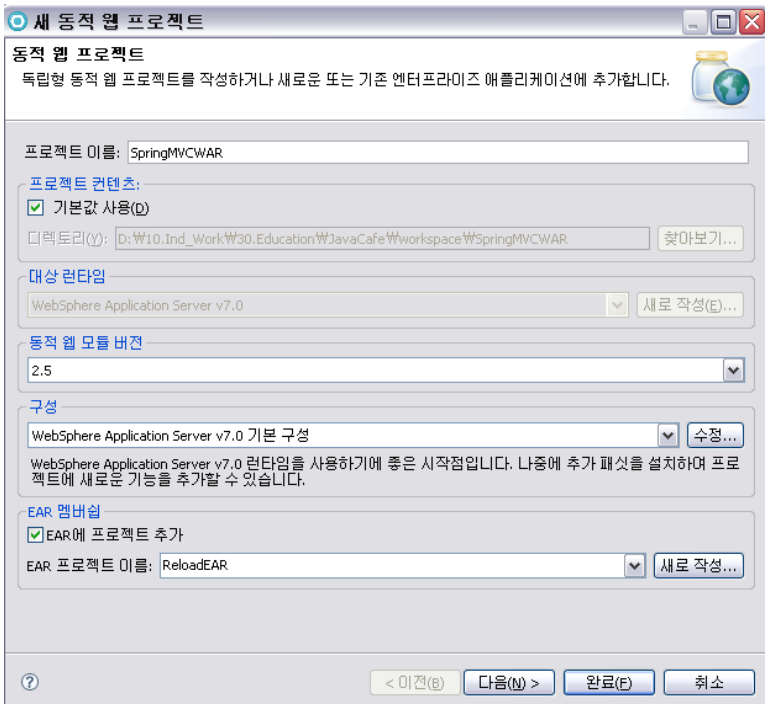
Spring MVC 는 Façade pattern 방식을 사용하여 DispactcherServlet 이 앞단에 위치하여 모든 request 를 받고 이를 Handler와 Controller 를 거쳐서 그 결과를 View 로 클라이언트에게 반환합니다. Spring MVC 의 구조 방식은 이미 알고 계신다는 전제를 걸고하는 WAS 강좌인 만큼 이를 실질적으로 WAS 에 배치하여 운영하는 테스트를 한번 해 보도록 하겠습니다.

Part 1. RAD v7.5 를 이용한 Spring MVC 어플리케이션 개발

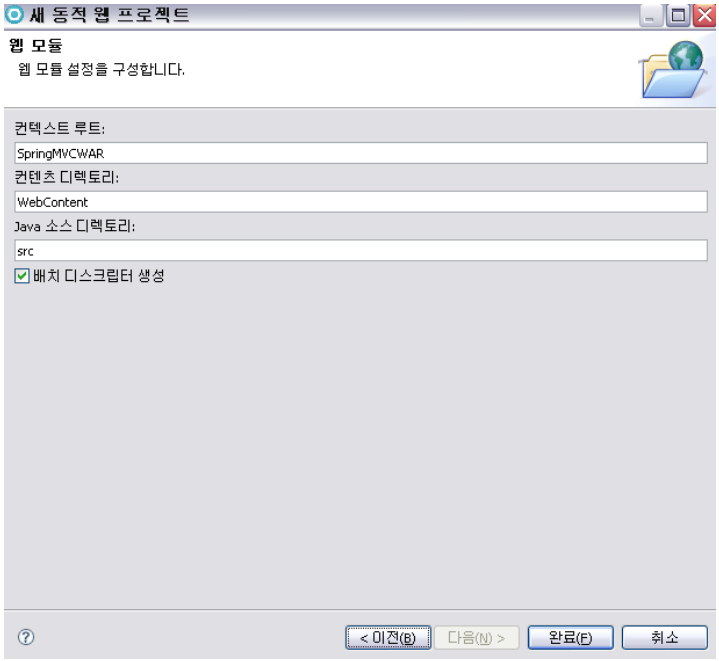
RAD v7.5 툴에 대해서는 이미 지난 강좌들에서 자세히 설명했으니 RAD v7.5 툴을 실행합니다. RAD v7.5 를 실행했으면 개발을 진행하기 위해서 우선 새로운 프로젝트를 만들어야 합니다. SPRING MVC 는 웹프로젝트 이므로 파일 > 새로작성 > 동적 웹프로젝트를 선택하여 개발을 진행할 동적 웹프로젝트를 만듭니다.



아시겠지만 동적 웹 프로젝트는 웹 환경에서 많이 사용되는 JSP 와 Servlet 을 돌릴 수 있는 어플리케이션 작업공간을 만들 수 있는 프로젝트 이며 WAR 형태의 패키지로 만들 수 있습니다. 동적 웹 프로젝트 마법사가 나오면 사용하고자 하는 프로젝트 이름을 넣어주고 EAR 프로젝트에 추가 부분을 disable 해줍니다. 이번 강좌에서는 WAR 만 만들 것 이므로 굳이 EAR에 프로젝트를 추가할 필요가 없습니다. 입력이 되었으면 다음을 누릅니다.

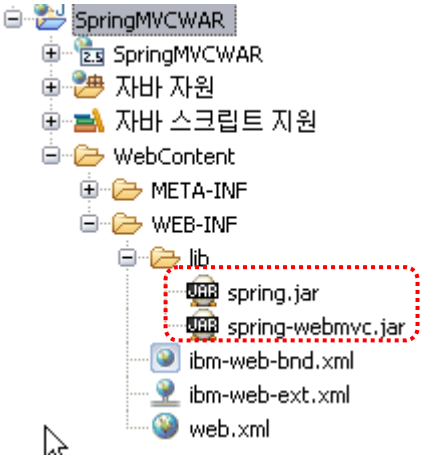


컨텍스트 루트(Context root) 와 웹 모듈 설정을 확인한 후 완료를 누릅니다.

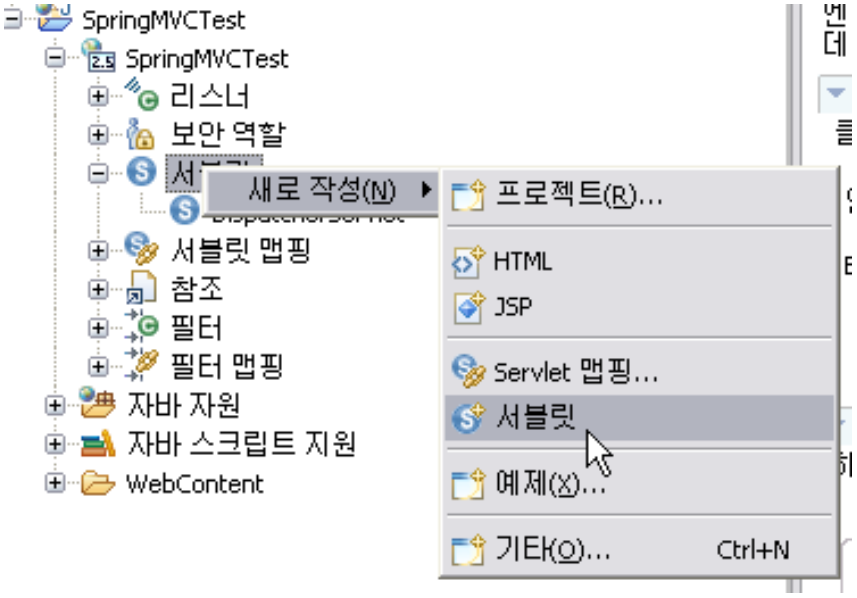


완료를 누르게 되면 기존에 주어진 이름으로 동적 웹 프로젝트가 생성된 것을 RAD 화면에서 확인할 수 있습니다. 이제 동적 웹 프로젝트가 만들어졌으면 작업공간과 환경은 만들어졌고 실제적으로 SpringMVC 동작을 위하여 환경 설정을 먼저 하도록 하겠습니다. SpringMVC 는 기본적으로 Spring 에서 제공하는 jar 파일에 동작을 위한 모든 소스가 들어간 Framework 의 개념이기 때문에 해당 환경을 설정하기 위해서는 해당 파일을 동적 웹 프로젝트가 인식하게 참조해주어야 합니다. 따라서 Spring Framework 에서 제공하는 spring.jar 파일과 spring-webmvc.jar 파일을 WebContent/Web-Inf/lib 밑에 복사해 둡니다. (해당 폴더에 jar 파일을 올려 놓으면 동적 웹프로젝트는 자동으로 인식할 수 있습니다. 여기서는 단순한 SpringMVC 를 하기 위해서 필요한 클래스만 올려놓은 것이라 두개만 올려놓은 것이고 향후 Spring 기능을 더 활용하고 프로젝트를 확장한다면 다른 jar 파일들도 올려놓으셔야 합니다.)

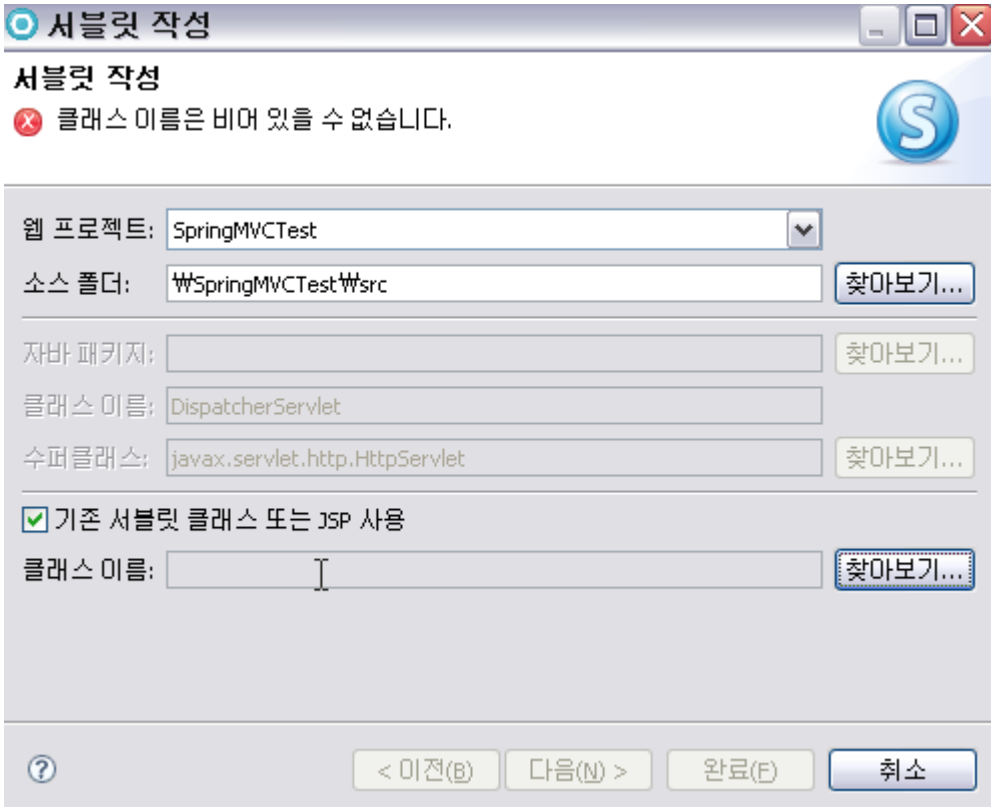
여기서는 단순하게 Spring Framework 를 돌리기 위해서 동적 웹프로젝트의 lib 폴더 밑에 두지만 필요하시면 전체 어플리케이션 아키텍처와 Scope 을 고려하여 EAR 의 Utility Jar 로 불러오시거나 WAS 의 Shared Library 로 설정할 수도 있습니다.



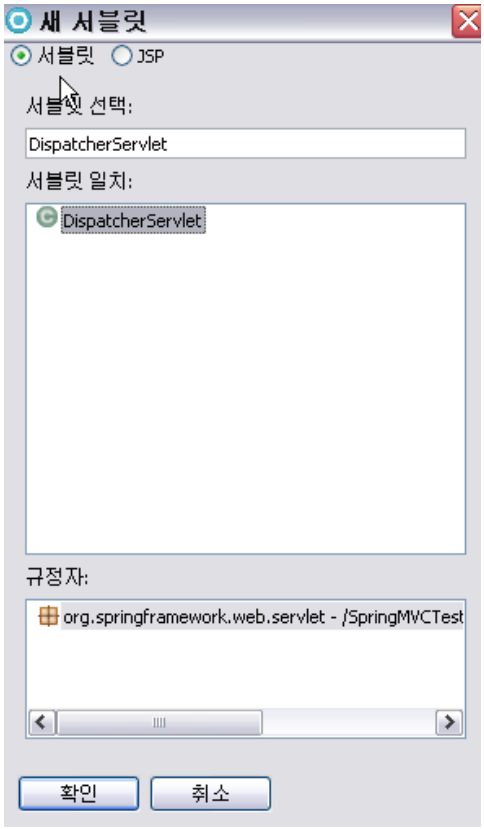
Spring framework 를 사용하기 위한 Jar 파일을 참조한 후에는 SpringMVC 에서 지정한 데로 DispatcherServlet 을 반드시 만들어야 합니다. 만들어 둔 동적 웹프로젝트에서 서블릿에서 마우스 우 클릭하여 새로작성 > 서블릿을 선택합니다.



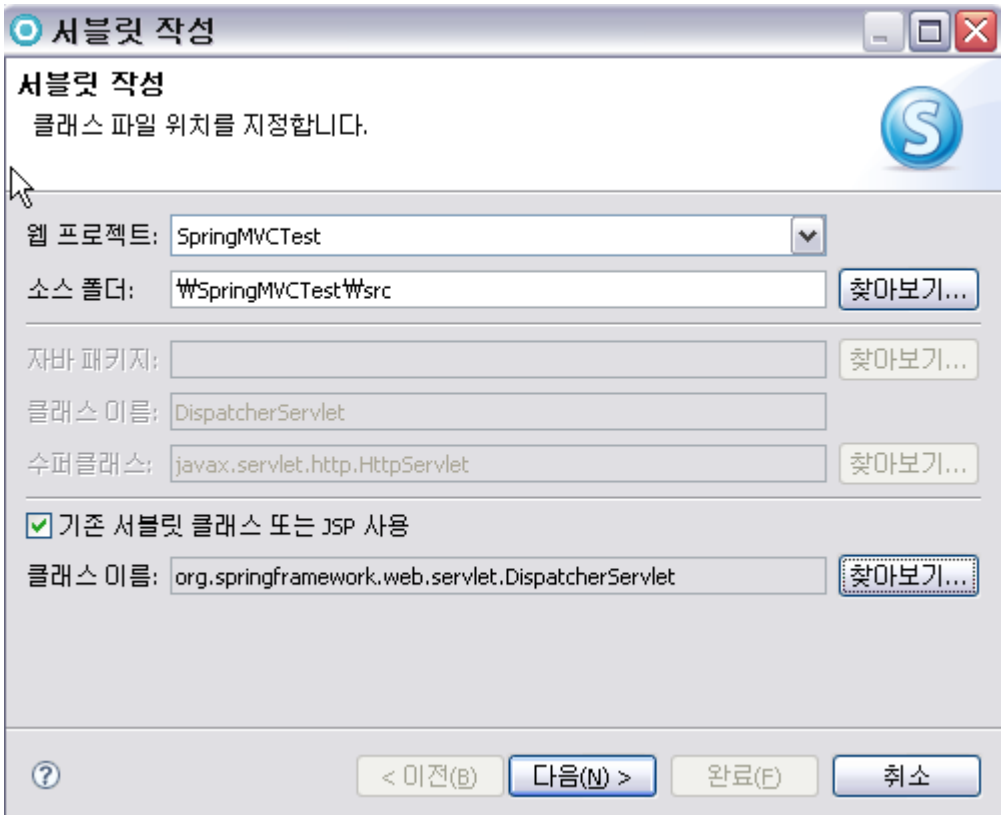
서블릿 작성 마법사가 나타나면 기존 서블릿 클래스 또는 JSP 사용 옵션을 클릭하고 찾아보기 버튼을 선택합니다.



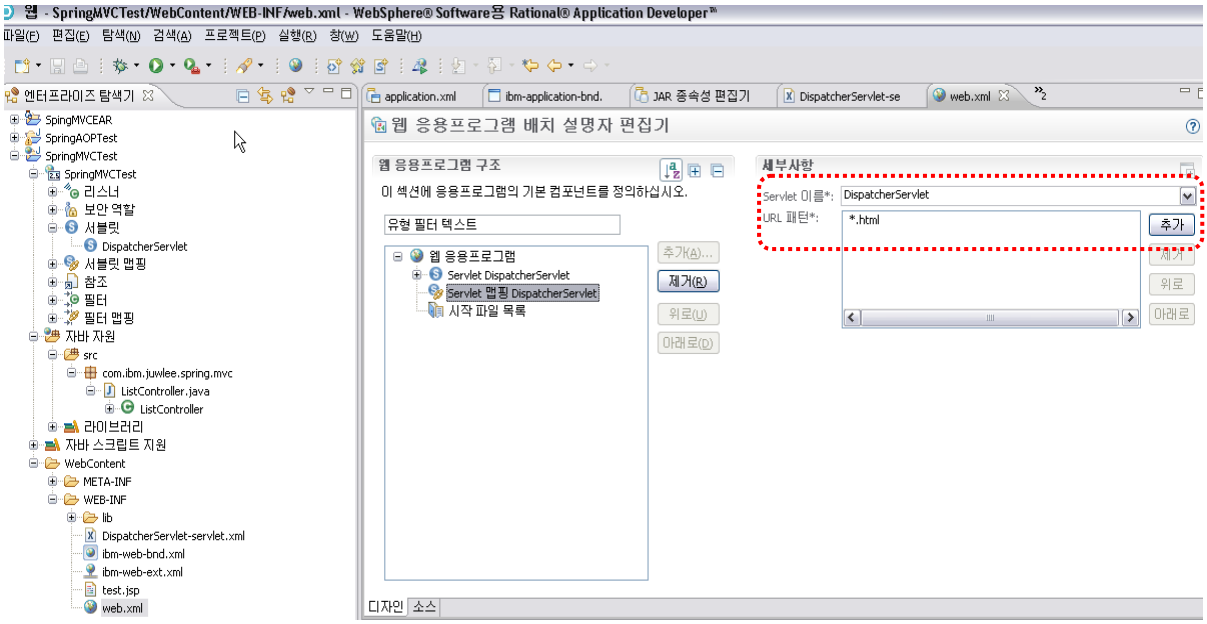
이후, Spring 에서 제공하는 DispatcherServlet 을 선택하고 확인합니다.



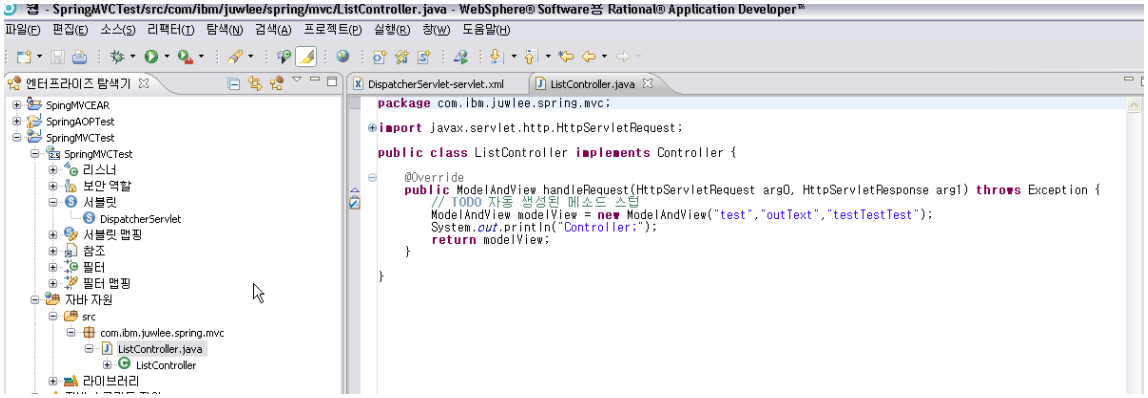
DispatcherServlet 을 확인했으면 다음을 클릭하고 완료하여 서블릿 작성을 마무리 합니다.



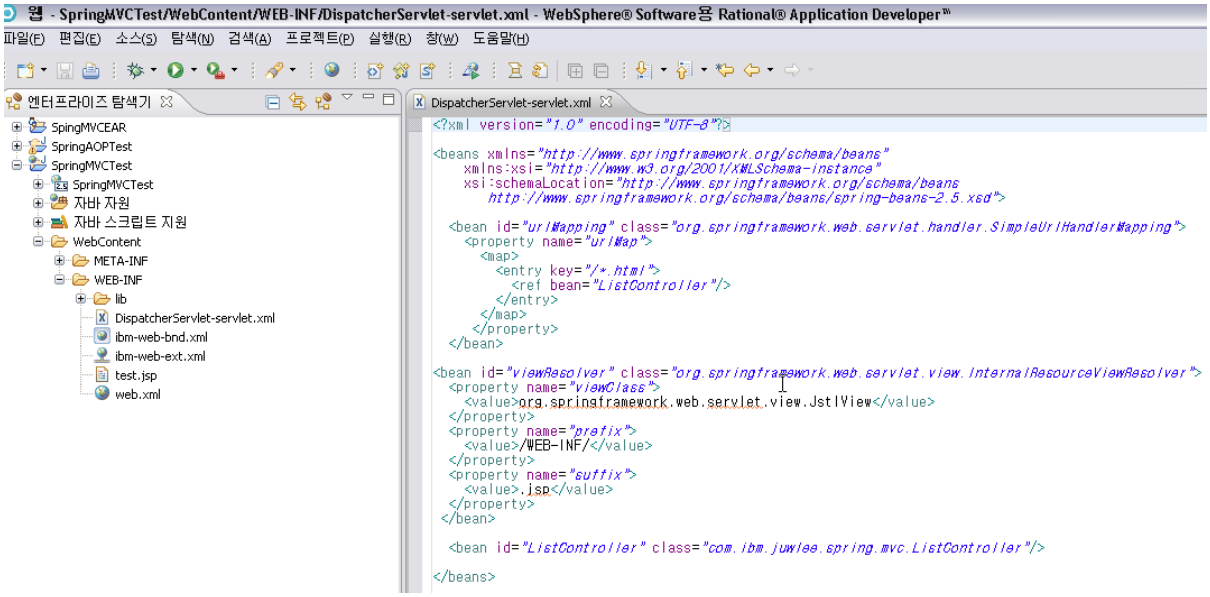
서블릿을 작성했으면 동적 웹프로젝트의 web.xml 파일을 클릭하여 설정을 확인합니다. 이후 DispatcherServlet 의 서블릿 맵핑을 클릭하여 어떤 URL 패턴을 기준으로 SpringMVC 를 수행할지 지정합니다. (하단의 예제에서는 *.html 을 선택하여 html 확장자로 들어오는 것은 반드시 SpringMVC 로 구동되게 설정하였습니다.)



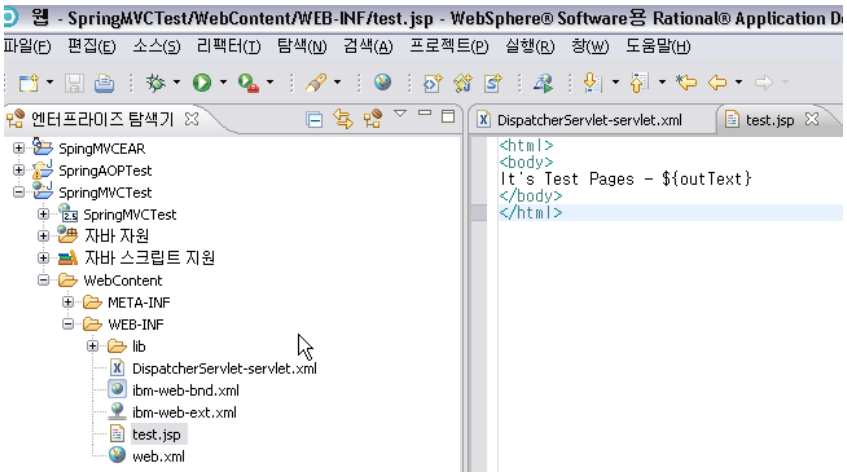
위와 같이 서블릿 설정이 완료되었으면 SpringMVC 에서 사용될 Controller 를 작성합니다.



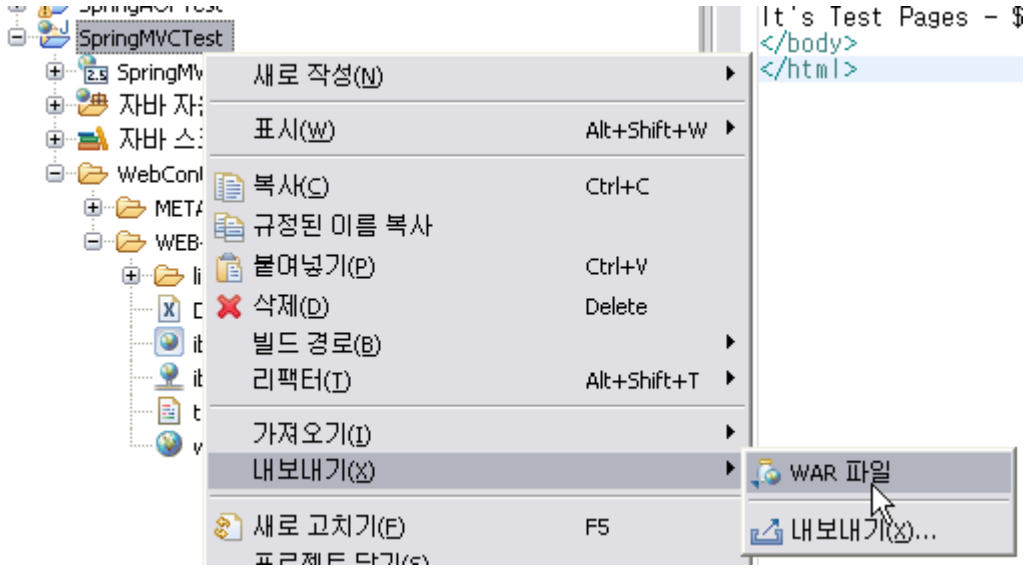
또한, Spring framework 에서 명시된 데로 DispatcherServlet-servlet.xml 설정 파일을 생성하여 SpringMVC 에서 사용될 bean 정의와 handller, controller, viewresolver 등의 mapping 을 설정합니다. (이전에도 설명드렸지만 이 부분은 Spring 개발 가이드를 참조하시는 것이 더 이해하기 편하실 수 있습니다. 지금 강좌는 SpringMVC 를 아신다는 관점에서 WebSphere 강좌입니다.)



마지막으로 결과값을 받아올 View 페이지인 간단한 jsp 페이지를 하나 만듭니다.



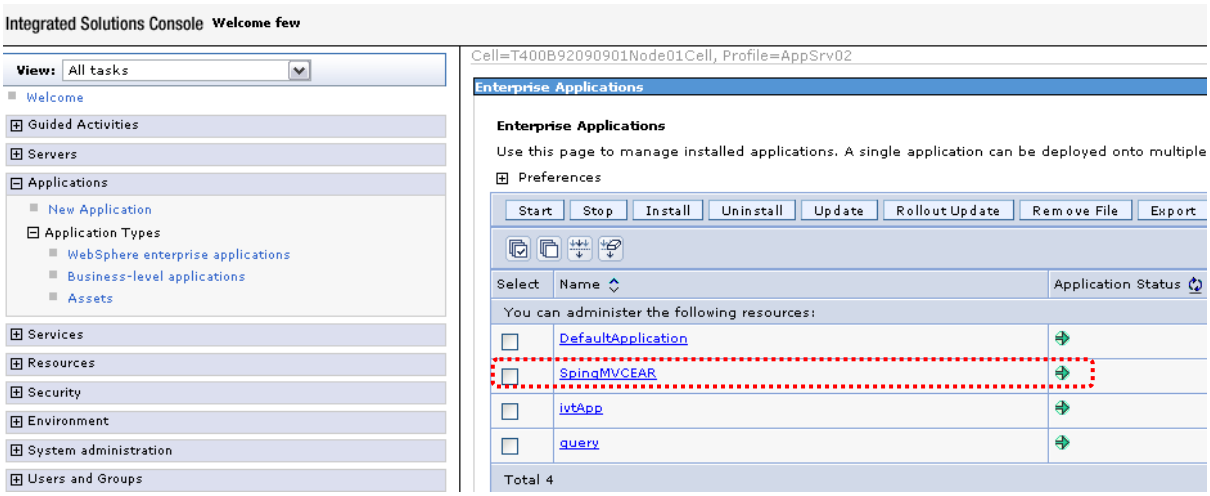
개발이 완료되었으면 내보내기 > WAR 를 선택하여 동적 웹프로젝트를 WAR 파일로 반출합니다.



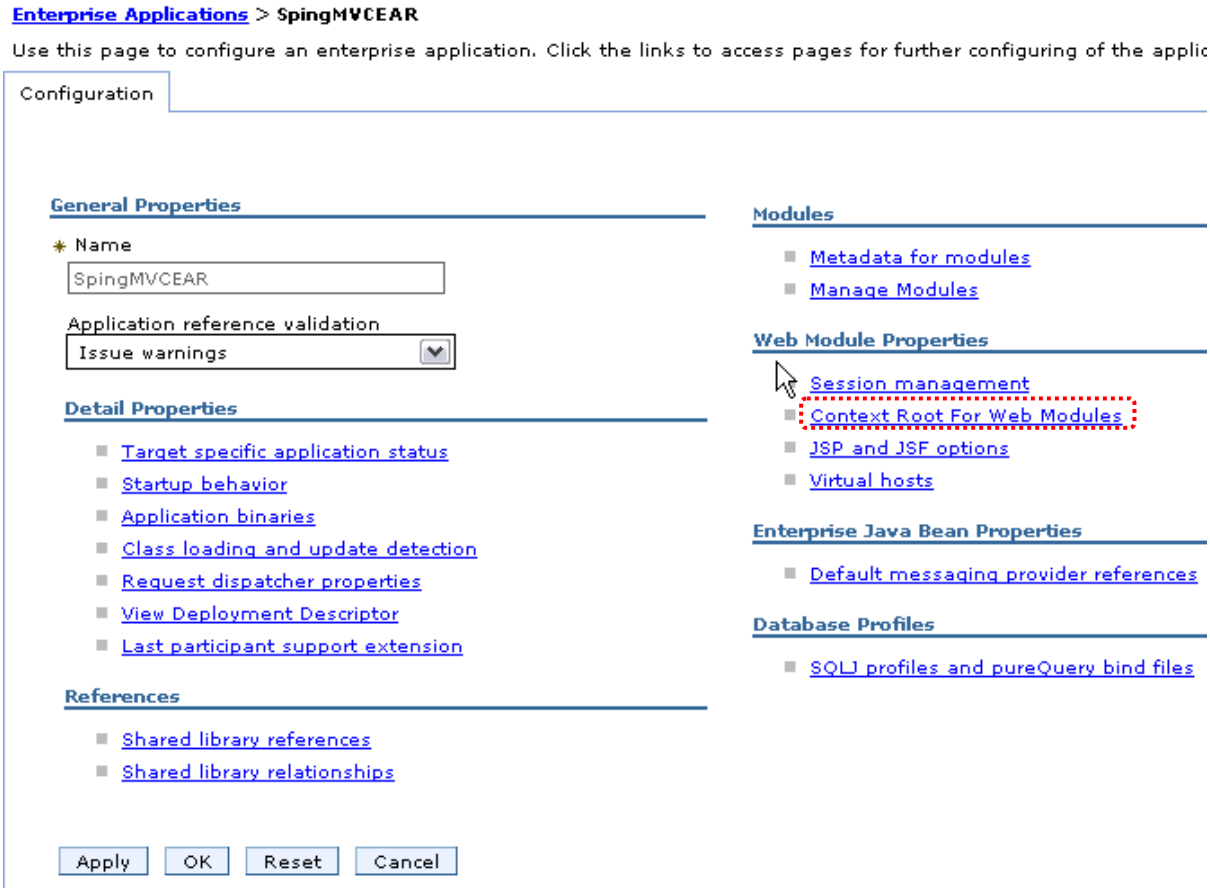
이렇게 반출된 WAR 파일을 WAS Admin console 을 이용하여 일반 어플리케이션처럼 배포하면 준비가 다 끝납니다. (어플리케이션 배포는 이전 강좌를 참고하시기 바라겠습니다.)

Part 2. Spring MVC 어플리케이션 테스트

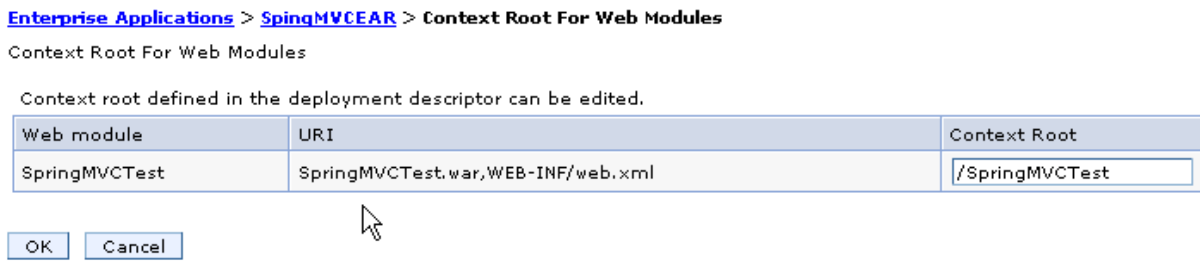
Spring framework 을 이용해 동적 웹프로젝트를 만들어서 WAS 에 배포한 후 어플리케이션이 올라왔는지 확인합니다.



어플리케이션이 정상적으로 올라왔으면 어플리케이션을 클릭하여 설정으로 들어갑니다.



설정에서 이 어플리케이션을 호출하기 위한 Context Root 를 확인합니다.



확인한 Context Root 를 이용해서 배포한 동적 웹어플리케이션을 브라우저를 이용하여 실행한 후 결과를 확인합니다.

결과가 정상적으로 test.jsp 에서 지정한데로 나온다면 SpringMVC 테스트는 성공하신 것 입니다. 즉, SpringMVC 에 의해서 URL mapping 에 지정된데로 *.html 이 DispatcherServlet 으로 가고 여기서 다시 handler 와 controller 를 탄 후 view resolver 를 통해서 물리적인 jsp 파일에 결과값이 mapping 되어서 실제적으로 위와 같은 결과를 확인할 수 있습니다.

참고 1) IBM RAD v7.5 Information Center
http://publib.boulder.ibm.com/infocenter/radhelp/v7r5/index.jsp

참고 2) The Spring series, Part 1: Spring 프레임웍 소개
http://www.ibm.com/developerworks/kr/library/wa-spring1/

참고 3) IBM WebSphere Developer Technical Journal: IBM WebSphere Application Server용 Spring 애플리케이션 개발 -- Part 1 (한글)
http://www.ibm.com/developerworks/kr/library/0603_patil/0603_patil1.html