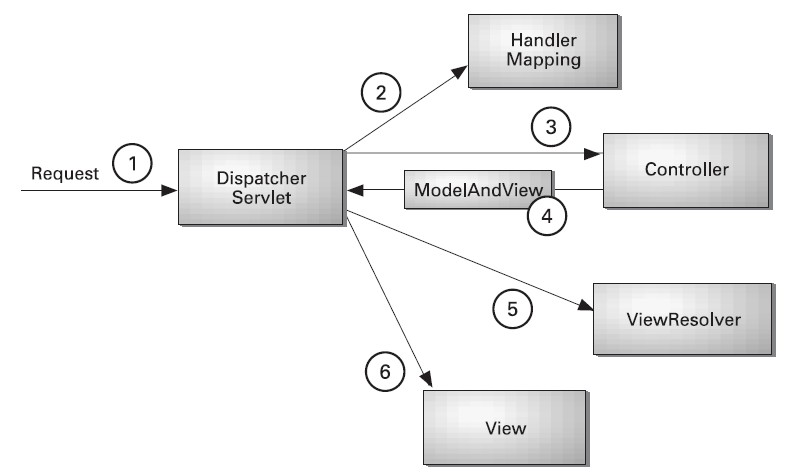


**spring MVC에서 하나의 요청에 대한 Life Cycle**

****

**Spring MVC내에서 처리되는 하나의 요청에 대한 Life Cycle과정은 위 그림과 같다.**

**클라이언트의 요청에 대한 최초 진입 지점은 DispatcherServlet이 담당하게 된다. 대부분의 MVC를 지원하는 프레임워크가 메인 Servlet을 가지는 것처럼 Spring MVC 또한 메인 Servlet이 최초 진입지점으로 다음의 작업을 처리하게 된다.**

**DispatcherServlet은 Spring Bean Definition에 설정되어 있는 Handler Mapping 정보를 참조하여 해당 요청을 처리하기 위한 Controller를 찾는다.**

**DispatcherServlet은 선택된 Controller를 호출하여 클라이언트가 요청한 작업을 처리한다.**

**Controller는 Business Layer와의 통신을 통하여 원하는 작업을 처리한 다음 요청에 대한 성공유무에 따라 ModelAndView 인스턴스를 반환한다. ModelAndView 클래스에는 UI Layer에서 사용할 Model데이터와 UI Layer로 사용할 View에 대한 정보가 포함되어 있다.**

**DispatcherServlet은 ModelAndView의 View의 이름이 논리적인 View 정보이면 ViewResolver를 참조하여 이 논리적인 View 정보를 실질적으로 처리해야할 View를 생성하게 된다.**

**DispatcherServlet은 ViewResolver를 통하여 전달된 View에게 ModelAndView를 전달하여 마지막으로 클라이언트에게 원하는 UI를 제공할 수 있도록 한다. 마지막으로 클라이언트에게 UI를 제공할 책임은 View 클래스가 담당하게 된다.**

/hello.mvc

요청

/service.mvc

xxxController

web.xml

/ ->

ModelAndView

DispatcherServlet

View 파일

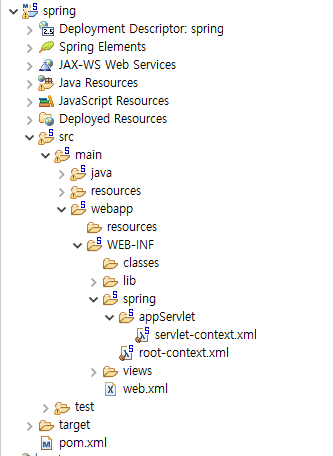
결과 리턴

servlet-context.xml

자바클래스

라이브러리

jsp

****

**web.xml : 전체 웹 컨텍스트 설정( / (모든 url) -->**DispatcherServlet

**servlet-context.xml: spring mvc 설정 (url mapping, view)**

**ioc/di**

**@Component , Repository, Service : 클래스 선언**

**@Autowired , Qualifier("vo") : 변수 , 메소드 선언**

**==> <context:component-scan base-package="패키지명"**

**aop**

**@Aspect : 클래스 선언**

**@Pointcut:메소드 위**

**@Before , After, AfterReturing, Around**

**==><aop:autoproxy />**

**mvc**

**@Controller**

**@RequestMapping**

**@RequestParam**

**@ModelAttribute**

**@RestController**

**@ResponseBody**

**==>**  <annotation-driven />

**@Controller**

**class A**

**@RequestMapping(name="/hello", method=RequestMethod.POST)**

**public ModelAndView handleRequest**

**(HttpSer...... reqeust, .....response)**

**@Controller**

**@RequestMapping("/hello")**

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

**@RequestMapping(value="/hello" , method=RequestMethod.POST)**

**프로젝트**

**게시판관리**

**게시물조회/저장/수정/삭제+**

**회원관리**

**회원조회/등록/수정/삭제+**

**상품관리**

**상품등록/수정/삭제/조회+...**

**"관리" – 조회 / 저장 / 수정 / 삭제 (기본)**

**crud : create (저장) / read(조회) / update / delete**

**board 테이블 – BoardVO – BoardDAO – BoardController : 4개 이상 RequestMapping 메소드**

**폼 / 폼 처리**

**member 테이블**

**product 테이블**

**- 컨트롤러 메소드 리턴타입**

**ModelAndView: 모델+뷰(명시또는url자동)**

**String : 뷰(명시)**

**void : 뷰(url 자동)**

**Map : 모델+ 뷰(url 자동)**

**- 컨트롤러 메소드 매개변수타입**

**1> HttpServletRequest 등 서블릿 api**

**2> String , int, double, boolean 타입 + 변수명(=요청파라미터명): 요청파라미터값 자동 저장**

**3> @RequestParam("i") String j , int, double, boolean 타입 + 변수명(=요청파라미터명): 요청파라미터값 자동 저장**

**4> 요청파라미터갯수 많으면 VO 객체 정의하고**

**@ModelAttribute("m") XXXVO v**

**4-1. 요청파라미터명과 xxxVO 멤버변수명 동일 자동 저장**

**4-2. XXXVO 객체는 자동 MODEL – JSP 전달**

**@Controller:클래스**

**@RequestMapping(value="", method=...):메소드**

**(또는 클래스)**

**컨트롤러 매핑된 메소드 매개변수**

**@RequestParam(value="", required= , defaultVaule="")**

**@ModelAttribute("") LoginVO (또는 메소드)**

**@RequestParam(value="id", required=false, defaultValue="guest")**

**String userid**

**@ModelAttribute("vo") BoardVO vo**

**- html태그**

**매핑**

**request.getContextPath() ---> "/mvc2"**

**<form action="/project/list"**

**<a href="url"**

**<script src="url"**

**- servlet, jsp, spring(웹서버)**

**@RequestMapping("/list")**