KoreaIT Spring04

가로선

# @ResponseBody

return 값을 view 형태로 인식하지 않고 콜백 메서드로 전달하기 위한 어노테이션

# 파일 업로드 필요한 라이브러리

pom.xml - fileupload, io

| *<!-- https://mvnrepository.com/artifact/commons-fileupload/commons-fileupload -->* <dependency>  <groupId>commons-fileupload</groupId>  <artifactId>commons-fileupload</artifactId>  <version>1.3.1</version> </dependency> |
| --- |

| *<!-- https://mvnrepository.com/artifact/commons-io/commons-io -->* <dependency>  <groupId>commons-io</groupId>  <artifactId>commons-io</artifactId>  <version>2.4</version> </dependency> |
| --- |

MultipartResolver Bean 생성

| *<!-- MultipartResolver 사용 -->*  <bean id="multipartResolver" class="org.springframework.web.multipart.commons.CommonsMultipartResolver">  <property name="defaultEncoding" value="UTF-8"/>  *<!-- 최대 업로드 용량 약 10mb -->*  <property name="maxUploadSize" value="1048576"/>  </bean> |
| --- |

<!-- Autowired 기능을 사용하기 싶다면 -->

<context:annotation-config/>

| @Controller public class FileUploadController {    @Autowired *//자동 주입 어노테이션*  *//servletContext, session, request 서블릿에서 지원하는 객체들.*  ServletContext application;    @Autowired  HttpServletRequest request;    @Autowired  HttpSession session;      static final String VIEW\_PATH = "/WEB-INF/views/";    @RequestMapping(value= {"/", "insert\_form.do"})  public String insert\_form() {    return VIEW\_PATH + "insert\_form.jsp";  }    @RequestMapping("upload.do")  public String upload(PhotoVO vo) {    *//ServletContext application application.getRealPath(... 를 가져왔었는데* *편하게 사용하기 위해 Autowired 사용*    String webPath = "/resources/upload";    String savePath = application.getRealPath(webPath);    *//String savePath = "c:/myupload";*      MultipartFile photo = vo.getPhoto();  String filename = "no\_file";    if(!photo.isEmpty()) {  *//업도르될 실제 파일명*  System.out.println("photo들어옴");  filename = photo.getOriginalFilename();    File saveFile = new File(savePath, filename);  *//savePath경로까지 가서 filename 이라고 하는게 있나 봐줄래?*    if(!saveFile.exists()) {  System.out.println("파일이없다");  saveFile.mkdirs();  } else {  *//currentTimeMillis 1970년 부터 2022년 현재까지의 시간을 1000분의 1초로 저장*  long time = System.currentTimeMillis();  filename = String.format("%d\_%s", time, filename);  saveFile = new File(savePath, filename);  }    try { *//실제로 파일을 업로드 하는 코드*  photo.transferTo(saveFile);    } catch (Exception e) {  *// TODO Auto-generated catch block*  e.printStackTrace();  }  }    vo.setFilename(filename);  request.setAttribute("vo", vo);    return VIEW\_PATH + "input\_result.jsp";  } } |
| --- |