

2025년 상반기 K-디지털 트레이닝

# 스프링 MVC의 Controller

[KB] IT's Your Life



- Controller 메서드의 매개변수로 Model 타입
  - 컨트롤러에서 생성된 데이터를 JSP에 전달
    - request 스코프에 속성으로 저장
    - jsp로 forward
  - Servlet에서 모델2 방식으로 데이터를 전달하는 방식

```
request.setAttribute("serverTime", new java.util.Date());
```

```
RequestDispatcher dispatcher = request.getRequestDispatcher("/WEB-
INF/jsp/home.jsp");
```

dispatcher.forward(request, response);

#### HomeController.java

```
package org.scoula.controller;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
@Controller
                                                                                      BL의 결과가 model매개변수를
통해서 전달됨
                                               모델 Dependency 요구됨
@Slf4j
public class HomeController {
                                               스코프에 담을 맵이라고 생각.
 @GetMapping("/")
 public String home(Model model) {
                                         setAttribute대신에 addAttribute 키 밸류 쌍
   model.addAttribute("name", "홍길동"); ~
                              // View의 이름
   return "index";
```

## views/index.jsp

```
request scope
에 해당됨.
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="ko">
                                           @GetMapping("/")
<head>
                                           public String home(Model model) {
 <meta charset="UTF-8">
                                              model.addAttribute("name", "홍길동");
 <title>Title</title>
</head>
                                              return "index";
                                                                 // View의 이름
<body>
<h1>${name} 환영합니다.</h1>
</body>
</html>
                                                            .전에 배운 EL 표기
```

# 홍길동 환영합니다.

#### @ModelAttribute

- ○ DTO 쿼리 파리미터는 자동으로 뷰<u>로</u> 저달됨
- - @ModelAttribute("파라미터명")
  - 이때 이 어노테이션을 사용 쿼리 파라미터를 request 스<u>코프에 저장</u>
  - 파라미터명이 스코프의 키가 됨

#### ✓ views/sample/ex04.jsp

```
<!DOCTYPE html>
                                                                                                  <%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

∨ □ WEB-INF

<html>
<head>

∨ □ views

<meta charset="UTF-8">

✓ □ sample

<title>Insert title here</title>
                                                                                                               JSP ex04.jsp
</head>
<body>
                                                                                                            JSP index.jsp
           <h2>SAMPLE DTO ${sampleDTO}</h2>
           <h2>PAGE ${page}</h2>
</body>
</html>
```

http://localhost:8080/sample/ex04?name=aaa&age=11&page=9

INFO org.scoula.ex03.controller.SampleController(ex04:84) - dto: SampleDTO(name=aaa, age=11) INFO org.scoula.ex03.controller.SampleController(ex04:85) - page: 9

어떻게 해야할까?

#### SampleController.java

```
...
public class SampleController {
...
@GetMapping("/ex04")
public String ex04(SampleDTO dto, @ModelAttribute("page") int page) {
    log.info("dto: " + dto);
    log.info("page: " + page);
    return "sample/ex04";
}

olicited The page of the
```

http://localhost:8080/sample/ex04?name=aaa&age=11&page=9

```
SAMPLE DTO SampleDTO(name=aaa, age=11)
PAGE 9
```

Controller 메서드의 리턴 타입 메서드의 리턴의 의미.

뷰의 이름 으로

rest통신 할때많이 쓰임 결과가 JSON응답

jsp 뷰의 경로/이름으로 해석 ✔

젤 많이 쓰이는 2가지

oid 컨텍스트 경로는 빼고 ✔ 호출한 URL과 동인 이름의 jsp로 해석 ✔

#### VO, DTO 타입

JSON 타입의 데이터로 변환해서 브라우저로 응답

#### o ResponseEntity 탁입

Http 헤더 정보와 내용을 가공하여 직접 브라우저로 응답

Model, ModelAndView모델 데이터와 뷰 정보를 같이 리턴하는 특수한 경우.

Model로 데이터를 변환하거나 뷰이름을 같이 지정

#### HttpHeaders

응답에 내용 없이 Http 헤더 메시지만 전달

특수한경우

#### void 타입

```
@GetMapping("/ex05") url이 이류의 이름이 됨. 컨텍스트 경로 빼고. public void ex05() { log.info("/ex05......"); }
```

#### HTTP 상태 404 – 찾을 수 없음

타입 상태 보고

비시지 /WEB-INF/views/sample/ex05.jsp

설명 Origin 서버가 대상 리소스를 위한 현재의 representation을 찾지 못했거나, 그것이 존재하는지를 밝히려 하지 않습니다.

VMware to Runtime 9.0.33.A.RELEASE

#### → 요청 url을 jsp의 경로로 해석

/sample/ex05  $\rightarrow$  /WEB-INF/views/sample/ex05.jsp

#### String 타입

- o forward: 포워드 방식으로 처리하는 경우
  - "뷰 이름" 문자열 리턴
- redirect: 리다이렉트 방식으로 처리하는 경우 ✓
  - "redirect:요청url" 문자열 리턴



### HomeController.java

```
@Controller
@Slf4j
public class HomeController {
  @GetMapping("/")
  public String home(Model model) {
  model.addAttribute("name", "홍길동");
    return "index";
```

#### RedirectAttributes

Servlet에서 redirect 방식

```
response.sendRedirect("/sample/ex06?name=aaa&age=10");
```

#### Spring MVC 방식

// RedirectAttributes ra매개변수 지정 가정 // addFlashAttribute(이름, 값) 메서드로 지정

```
ra.addAttribute("name", "AAA");
ra.addAttribute("age", 10);
```

return "redirect:/sample/ex06"; // 뷰 이름이 아닌 요청 경로를 제시 🗸

재요청시 리퀘스트 스코프 생명주기 끝남. 전에 스코프에 담긴거 유지가 안됨.

첫요청에대한 응답에 첫요청에 대한 결과를 쿼리 스트링에 담아서 보낼 수 있고

또는

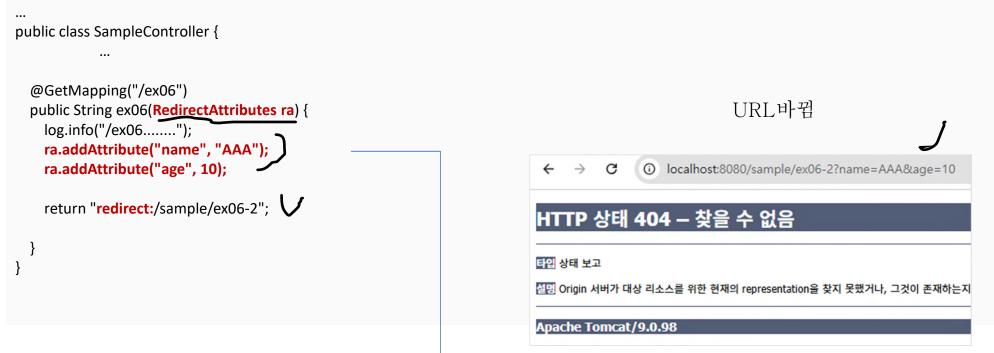
간단히 세션스코프에 임시적으로 넣어서

스프링에서는 이를 지워함

RedirectAttributes라는 객체를 매개변수로 DI요청하고 addAttribute메서드 사용.

세션 에 저장은 addFlashAttribute메서드 사용

#### SampleController.java



- O http://localhost:8080/sample/ex06 요청
- → http://localhost:8080/sample/ex06-2?name=AAA&age=10 로 리다이렉트 됨

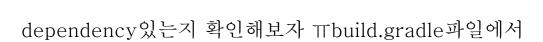
#### ☑ 객체 타입

- JSON 타입 응답하는 경우 사용
- o jackson-databind 라이브러리 필요

JSON문자열 <-> 자바객체 파싱이 필요함 기본 자바에는 변환해주는게 없다.

jackson 혹은 gson 라이브러리가 필요함

implementation 'com.fasterxml.jackson.core:jackson-databind:2.9.4'



#### SampleController.java

```
@Controller
                                                   JSON응답을 내보내겠다는 어노테이션
public class SampleController {
                                                     response바디파트에 들어갈 객체다
라는 의미!!!
         @GetMapping("/ex07")
         public @ResponseBody SampleDTO ex07() {
                   log.info("/ex07.....");
                   'SampleDTO dto = new SampleDTO();
                   dto.setAge(10);
                   dto.setName("홍길동");
                                                BL결과를 리턴
                   return dto;
```

#### http://localhost:8080/sample/ex07

```
"name": "홍길동",
"age": 10
```

응답으로 JSON문자열이 왔다. 자바객체를 반환했는데

☑ ResponseEntity 타입

응답 구성할때 직접적으로 핸들링할 수 있다

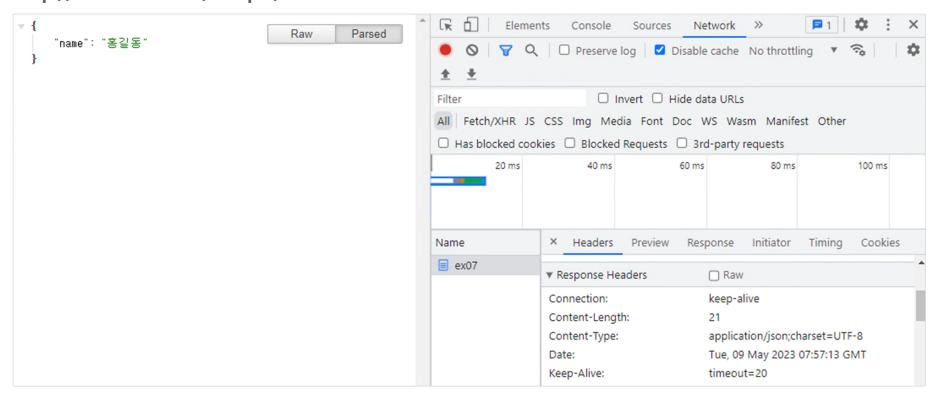
○ 브라우저로 직접 응답하는 경우

○ 응답 헤더 설정 ○ 으다 바디 서저

#### SampleController.java

```
import org.springframework.http.HttpHeaders;
 @Controller
                                                         body 부분 객체타입
 public class SampleController {
            @GetMapping("/ex08")
             public ResponseEntity<String> ex08() {
                        log.info("/ex08.....");
                        // {"name": "홍길동"}
                        String msg = "{\"name\": \"홍길동\"}";
헤더 커스터마이징 (HttpHeaders header = new HttpHeaders(); header.add("Content-Type", "application/json;charset=UTF-8");
                        return new ResponseEntity<>(msg, header, HttpStatus.OK);
                                                    바디 헤더 상태코드 다 지정.
```

#### http://localhost:8080/sample/ex08



#### 3 <mark>파일 업로드</mark>

#### 💟 파일 업로드 방법

- o Servlet 3.0 기능 이용 자체 기능 사용
  - multipart 설정
    - location: 업로드 처리 디렉토리 경로
    - maxFileSize: 업로드 가능한 파일 하나의 최대 크기
    - maxRequestSize: 업로드 가능한 전체 최대 크기(여러 파일 업로드 하는 경우)
    - fileSizeThreshold: 메모리 파일의 최대 크기(이보다 작으면 실제 메모리에서만 작업)

#### ○ 업로드 디렉토리 준비

■ c:\upload 업로드했을때 여기다가 저장하겠다

#### 파일 업로드

### ServletConfig.java

## WebConfig.java

```
package org.scoula.config;
       @Slf4i
       @Configuration
       public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer {
       r final String LOCATION = "c:/upload";
         final long MAX_FILE_SIZE = 1024 * 1024 * 10L; // 10M 한파일마다
제약조건final long MAX_REQUEST_SIZE = 1024 * 1024 * 20L; // 20M 한번요청시 -1:크기 제한 없음
                                                          // 5M 이크기보다 작으면 메모리써.
이보다 크면 파일에 임시파일로
       final int FILE SIZE THRESHOLD = 1024 * 1024 * 5;
         @Override
         protected void customizeRegistration(ServletRegistration.Dynamic registration) {
           MultipartConfigElement multipartConfig = new MultipartConfigElement(
                  LOCATION, // 업로드 처리 디렉토리 경로
                  MAX FILE SIZE, // 업로드 가능한 파일 하나의 최대 크기
                  MAX REQUEST SIZE, // 업로드 가능한 전체 최대 크기(여러 파일 업로드 하는 경우)
                                                // 메모리 파일의 최대 크기(이보다 작으면 실제 메모리에서만 작업)
                  FILE SIZE THRESHOLD
           registration.setMultipartConfig(multipartConfig);
```

#### 서울 답도그

## SampleController.java

form을 통해서 업로드하니까 폼으로 가는 url

## views/sample/exUpload.jsp

```
webapp
                                                                                                   WEB-INF
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
                                                                                                     views
<html>
                                                                                                        sample
 <head>
                                                                                                             JSP ex04.jsp
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
                                                                                                             JSP exUpload.jsp
 <title>Insert title here</title>
 </head>
                                                                                                          JSP index.isp
 <body>
  <form action="/sample/exUploadPost" method="post" enctype="multipart/form-data" accept-charset="UTF-8" >
  <div>
   <input type="file" name="files" />
  </div>
                                                           multi part
  <div>
                                                           클라이언트가 서버로 데이터를 보낼때
어떤 인코딩을 쓸건지는 3가지.
   <input type="file" name="files" />
  </div>
                                                           1 ison
   <div>
                                                           2 url encoding a=b& c=d& ...
   <input type="file" name="files" />
                                                           3 multipart
  </div>
                                                           바이러니 데이터를 전송할때 사용하는 인코딩 방법
   <div>
                                                           part란? input하나가 파트이다.
   <input type="file" name="files" />
  </div>
                                                           part는 자체적으로 헤더와 바디를 가진다.
파트마다 바디를가지고 그 바디를 설명하는 헤더가 있다.
   <div>
   <input type="file" name="files" />
  </div>
```

### 3 파일 업로드

## views/sample/exUpload.jsp

```
<div>
   <input type="submit" />
  </div>
 </form>
</body>
</html>
```

#### http://localhost:8080/sample/exUpload

```
파일 선택 선택된 파일 없음
제출
```

#### 3 파일 업로드

#### multipart encoding

기존 헤더 정보가 있고

Content-Type: multipart/form-data; boundary=frame 파트를 나눌때 "frame"이라는 문자열을 써서 경계를 나누겠다 기존 헤더 <<br/>body>> --frame Content-Type: image/jpeg 헤더 파트 Content-Length: 33377 구분 개행 http는 텍스트 기반 프로토콜이니까 64개의 글자로 표현된다. 바이너리 데이터가 바디 [JPEG data] 64가지 정보로 나누어 인코딩 base64 encoding --frame Content-Type: image/jpeg 파트 Content-Length: 33377 [JPEG data]

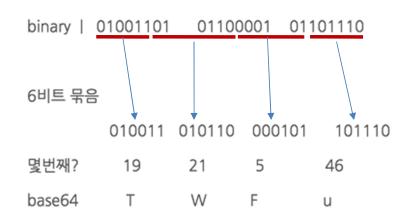
#### base64 인코딩

- Binary Data를 Text로 바꾸는 Encoding(binary-to-text encoding schemes)
- Binary Data를 Character set에 영향을 받지 않는 공통 ASCII 영역의 문자로만 이루어진 문자열로 바꾸는 **Encoding**

워본 바이너리 데이터

#### base64 색인표

Value Char		Value Char		Value Char		Value Char	
0	А	16	Q	32	g	48	w
1	В	17	R	33	h	49	x
2	С	18	S	34	i	50	У
3	D	19	Т	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	٧	37	-1	53	1
6	G	22	W	38	m	54	2
7	Н	23	X	39	n	55	3
8	I	24	Υ	40	0	56	4
9	1	25	Z	41	p	57	5
10	K	26	а	42	q	58	6
11	L	27	b	43	r	59	7
12	М	28	С	44	s	60	8
13	N	29	d	45	t	61	9
14	0	30	e	46	u	62	
15	Р	31	f	47	v	63	1



(주의할점은 000000 이 A 로 변한다)

base64 인코딩을 하면 크기가 원본보다 33% 커짐(3byte → 4byte)

#### 파일 업로드

## SampleController.java

```
파일 업로드 부분만

...
public class SampleController {

...

@PostMapping("/exUploadPost")
public void exUploadPost(ArrayList<MultipartFile> files) {

for(MultipartFile file: files) {

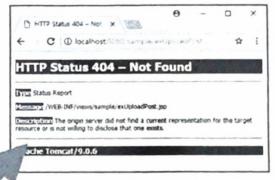
log.info("-------");

log.info("name:" + file.getOriginalFilename()); // 윈도우 OS: 한글 파일명인 경우 깨짐

log.info("size:" + file.getSize());

     오늘은 업로드만 시키는 동작까지만

}
```



뷰는 없으니까!

```
INFO: org.zerock.controller.SampleController - name:161falac3a8684a.jpg
INFO: org.zerock.controller.SampleController - size:1018843
INFO: org.zerock.controller.SampleController - name:888627.jpg
INFO: org.zerock.controller.SampleController - name:888627.jpg
INFO: org.zerock.controller.SampleController - size:332402
INFO: org.zerock.controller.SampleController - name:20171211_154611.jpg
INFO: org.zerock.controller.SampleController - size:413005
INFO: org.zerock.controller.SampleController - size:413005
INFO: org.zerock.controller.SampleController - name:구항가제1.jpg
INFO: org.zerock.controller.SampleController - size:121493
INFO: org.zerock.controller.SampleController - name:구항가제2.jpg
INFO: org.zerock.controller.SampleController - name:구항가제2.jpg
INFO: org.zerock.controller.SampleController - name:구항가제2.jpg
INFO: org.zerock.controller.SampleController - name:구항가제2.jpg
INFO: org.zerock.controller.SampleController - size:304061
```

로그 잘찍히는지

#### 스프링 MVC의 예외 처리

- @ExceptionHandler와 @ControllerAdvice를 이용한 처리
- o @ResponseEntity를 이용하는 예외 메시지 구성

에러에는 크게 2가지가 있다. 404 에러, 500에러 가장 흔하게 발생하는 에러들이다. 매번 에러 코드를 직접 하는것보다는 한곳에 모아서 중앙집중식으로 관리하는게 편한다

500에러 처리는 @EceptionHAndler을 써서 특정 예ㅎ외를 감지할 수 있다. @ControllerAdvice AOP를 통해 처리.

#### @ControllerAdvice

- o HTTP 상태코드 500 Internal Serveer Error에 대응하기 위한 기법
- o AOP(Aspect-Oriented-Programming)을 이용
- org.scoula.exception.CommonExceptionAdvice.java

관점 지향 프로그래밍. 스프링의 대표기능중 하나 BL을 어떤 관점에서 실행하고 싶으냐. 성능 모니터링 관점에서 실행하고 싶다. 시간측정이 필요하겠죠. 이번에는 보안관점에서 실행하고 싶다. 인증했는지에 대한 검증이 필요함 예외를 처리하는 관점에서 실행하고 싶다. trycatch문 필요.

이때 BL은 고정되어있는 관점에따라서 전처리 후처리가 맞게 바뀌어야 한다.

이를 잘 하게 하는 프로그래밍이 AOP

## org.scoula.exception.CommonExceptionAdvice.java

```
package org.scoula.exception;
                                                                                         main
                                                                                          iava
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;

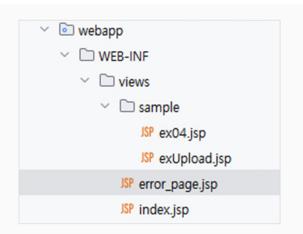
✓ org.scoula

@ControllerAdvice AOP
                                                                                               > onfig
                                                                                               > ontroller
@Log4j2
                                                                                               > @ ex03
public class CommonExceptionAdvice { 에러가 발생했을때 포워딩되서 오는 컨트롤러
                                                                                               exception
          @ExceptionHandler(Exception.class) 예외발생시 나한테로 와 public String except(Exception ex, Model model) { 컨트롤러 실행하다가,,
                                                                                                   © CommonExceptionAdvice
                     log.error("Exception ......" + ex.getMessage());
                      model.addAttribute("exception", ex);
                                                                                           DI요구.
                     log.error(model);
                     return "error_page";
                   리턴하는 의미도 컨트롤러 다른 메서드와 같음
뷰의 이름임
```

## ServletConfig.java

#### views/error\_page.jsp

```
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ page session="false" import="java.util.*"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>
<h4><c:out value="${exception.getMessage()}"></c:out></h4>
<c:forEach items="${exception.getStackTrace() }" var="stack">
  <c:out value="${stack}"></c:out>
 </c:forEach>
</body>
</html>
```



#### ☑ 확인

- http://localhost:8080/sample/ex04?name=aaa&age=11
  - page 파리미터 누락

#### No primary or default constructor found for int

- org.springframework.web.method.annotation.ModelAttributeMethodProcessor.createAttribute(ModelAttrib
- org.springframework.web.servlet.mvc.method.annotation.ServletModelAttributeMethodProcessor.createAtt
- org.springframework.web.method.annotation.ModelAttributeMethodProcessor.resolveArgument(ModelAtt
- org.springframework.web.method.support.HandlerMethodArgumentResolverComposite.resolveArgument()
- org.springframework.web.method.support.InvocableHandlerMethod.getMethodArgumentValues(Invocable
- org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerM
- org.springframework.web.servlet.mvc.method.annotation.ServletInvocableHandlerMethod.invokeAndHand
- org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandlerN
- org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.handleInternal(F
- org.springframework.web.servlet.mvc.method.AbstractHandlerMethodAdapter.handle(AbstractHandlerMethodAdapter.handle)
- org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:991)
- org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:925)

#### ◎ 404 에러 페이지 🍑

컨트롤러까지 가지 않았으니

- 404 에러는 서버에서 Exception을 발생시키지 않음
  - 예외 클래스: NoHandlerFoundException

디폴트는 예외가 발생하지 않음

예외가 발새하도록 설정 필요

- 404에러를 Exception으로 처리하려면 설정 필요
  - Advice에서 NoHandlerFoundException 예외 처리

#### WebConfig.java

```
package org.scoula.config;
...

public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer{
...

@Override
protected void customizeRegistration(ServletRegistration.Dynamic registration) {
    registration.setInitParameter("throwExceptionIfNoHandlerFound", "true");
    // 파일 업로드 설정
...

에외가 발생하게끔 지정
}
}
```

#### CommonExceptionAdvice.java

```
package org.scoula.exception;
public class CommonExceptionAdvice {
  @ExceptionHandler(Exception.class)
 public String except(Exception ex, Model model) {
   log.error("Exception ......" + ex.getMessage());
   model.addAttribute("exception", ex);
   log.error(model);
   return "error page";
          @ExceptionHandler(NoHandlerFoundException.class)
          @ResponseStatus(HttpStatus.NOT_FOUND) 404로 응답 보내고
          public String handle404(NoHandlerFoundException ex) {
   log.error(ex);
   model.addAttribute("uri", request.getRequestURI());실제 요청 객체를 uri이름으로 저장함. 리퀘스트 스코프에
   return "custom404",♥ 뷰 이름 custom404.jsp
```

#### ✓ views/custom404.jsp

```
<!DOCTYPE html>
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
                                                                                                     webapp
<html>

✓ □ WEB-INF

<head>

∨ □ views

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>

✓ □ sample

</head>
                                                                                                                   JSP ex04.jsp
<body>
                                                                                                                   JSP exUpload.jsp
<h1>해당 URL(${uri})은 존재하지 않습니다.</h1>
                                                                                                                JSP custom404.jsp
</body>
</html>
                                                                                                                JSP error_page.jsp
                                                                                                                JSP index.jsp
```

500 404 처리하는 코드도 템플릿에 추가하면 좋겠다!!

또한 멀티파트 처리하는 코드도 템플릿처리!

☑ 404 에러 페이지

http://localhost:8080/sample/badurl?page=1

해당 URL(/sample/badurl)은 존재하지 않습니다.