

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

✓ views/index.jsp

```
<%@ 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>
                                                                  // View의 이름
                                                return "index";
<body>
<h1>${name} 환영합니다.</h1>
</body>
</html>
```

홍길동 환영합니다.

@ModelAttribute

- DTO 쿼리 파리미터는 자동으로 뷰로 저달됨
- 기본 자료형 쿼리 파라미터는 뷰로 전달되지 않음
- o @ModelAttribute("파라미터명")
 - 쿼리 파라미터를 request 스코프에 저장
 - 파라미터명이 스코프의 키가 됨

☑ SampleController.java

✓ 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

SAMPLE DTO SampleDTO(name=aaa, age=11)

PAGE

→ page는 뷰에 전달되지 않음

SampleController.java

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

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

Controller 메서드의 리턴 타입

- String
 - jsp 뷰의 경로/이름으로 해석
- o void
 - 호출한 URL과 동인 이름의 jsp로 해석
- o VO, DTO 타입
 - JSON 타입의 데이터로 변환해서 브라우저로 응답
- o ResponseEntity 타입
 - Http 헤더 정보와 내용을 가공하여 직접 브라우저로 응답
- Model, ModelAndView
 - Model로 데이터를 변환하거나 뷰이름을 같이 지정
- HttpHeaders
 - 응답에 내용 없이 Http 헤더 메시지만 전달

void 타입

```
@GetMapping("/ex05")
public void ex05() {
  log.info("/ex05......");
}
```

HTTP 상태 404 – 찾을 수 없음

타입 상태 보고

메시지 /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
- o 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");
                                  // forward
    return "index";
```

RedirectAttributes

o Servlet에서 redirect 방식

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

o 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
```

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

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

SampleController.java

```
@Controller
public class SampleController {
           @GetMapping("/ex07")
           public @ResponseBody SampleDTO ex07() {
                       log.info("/ex07.....");
                       SampleDTO dto = new SampleDTO();
                       dto.setAge(10);
                       dto.setName("홍길동");
                       return dto;
```

http://localhost:8080/sample/ex07

```
"name": "홍길동",
"age": 10
```

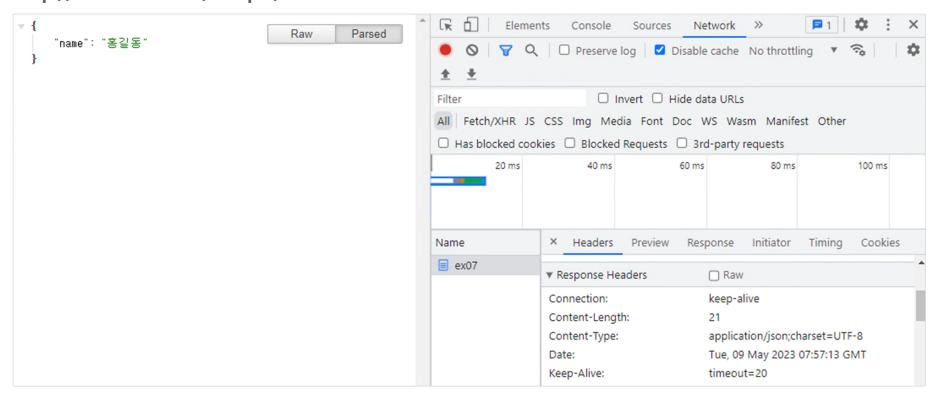
☑ ResponseEntity 타입

- 브라우저로 직접 응답하는 경우
- 응답 헤더 설정
- 응답 바디 설정

SampleController.java

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

http://localhost:8080/sample/ex08



파일 업로드

☑ 파일 업로드 방법

- o Servlet 3.0 기능 이용
 - 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

ServletConfig.java

```
@EnableWebMvc
@ComponentScan(basePackages = { "org.scoula.controller" })
public class ServletConfig implements WebMvcConfigurer{
                      Servlet 3.0 파일 업로드 사용시 - MultipartResolver 빈 등록
           @Bean
           public MultipartResolver multipartResolver() {
                      StandardServletMultipartResolver resolver
                                  = new StandardServletMultipartResolver();
                      return resolver;
```

WebConfig.java

```
package org.scoula.config;
@Slf4i
@Configuration
public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer {
 final String LOCATION = "c:/upload";
 final long MAX FILE SIZE = 1024 * 1024 * 10L; // 10M
                                                                  -1:크기 제한 없음
 final long MAX_REQUEST_SIZE = 1024 * 1024 * 20L; // 20M
 final int FILE SIZE THRESHOLD = 1024 * 1024 * 5;
                                                    // 5M
 @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);
```

파일 업로드

SampleController.java

webapp

views/sample/exUpload.jsp

```
✓ □ WEB-INF

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

∨ □ views

<!DOCTYPE html>
<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.jsp
 <body>
  <form action="/sample/exUploadPost" method="post" enctype="multipart/form-data" accept-charset="UTF-8" >
   <div>
    <input type="file" name="files" />
   </div>
   <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
```

multipart encoding

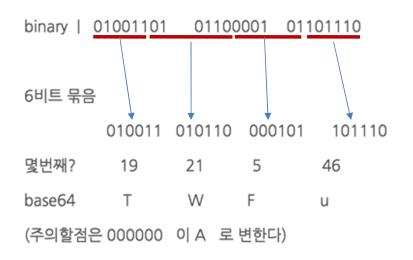
Content-Type: multipart/form-data; boundary=frame <<body>> --frame Content-Type: image/jpeg Content-Length: 33377 [JPEG data] base64 encoding --frame Content-Type: image/jpeg Content-Length: 33377 [JPEG data]

base64 인코딩

- o 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

o base64 색인표

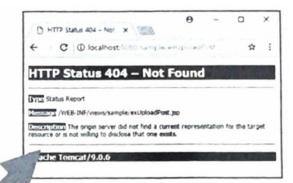
Value	Char	Value	Char	Value	Char	Value	Char
0	Α	16	Q	32	g	48	w
1	В	17	R	33	h	49	x
2	С	18	S	34	i	50	У
3	D	19	Т	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	٧	37	-1	53	1
6	G	22	W	38	m	54	2
7	Н	23	X	39	n	55	3
8	I	24	Υ	40	0	56	4
9	1	25	Z	41	p	57	5
10	K	26	а	42	q	58	6
11	L	27	b	43	r	59	7
12	М	28	С	44	s	60	8
13	N	29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



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

SampleController.java

```
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());
```



```
INFO: org.zerock.controlle Control
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:232402
INFO: org.zerock.controller.SampleController - name:20171211_154611.jpg
INFO: org.zerock.controller.SampleController - size:413005
INFO: org.zerock.controller.SampleController - name:2027711.jpg
INFO: org.zerock.controller.SampleController - name:2027711.jpg
INFO: org.zerock.controller.SampleController - name:2027711.jpg
INFO: org.zerock.controller.SampleController - name:2027712.jpg
INFO: org.zerock.controller.SampleController - name:2027712.jpg
INFO: org.zerock.controller.SampleController - name:2027712.jpg
INFO: org.zerock.controller.SampleController - size:304061
```

✓ 스프링 MVC의 예외 처리

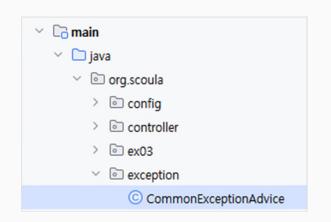
- @ExceptionHandler와 @ControllerAdvice를 이용한 처리
- @ResponseEntity를 이용하는 예외 메시지 구성

@ControllerAdvice

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

org.scoula.exception.CommonExceptionAdvice.java

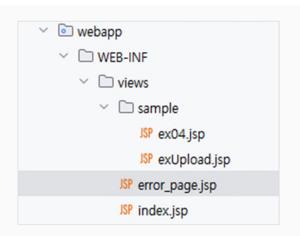
```
package org.scoula.exception;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;
@ControllerAdvice
@Log4j2
public class CommonExceptionAdvice {
            @ExceptionHandler(Exception.class)
            public String except(Exception ex, Model model) {
                        log.error("Exception ......" + ex.getMessage());
                        model.addAttribute("exception", ex);
                        log.error(model);
                        return "error page";
```



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

- 404 에러는 서버에서 Exception을 발생시키지 않음
 - 예외 클래스: NoHandlerFoundException
- 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");

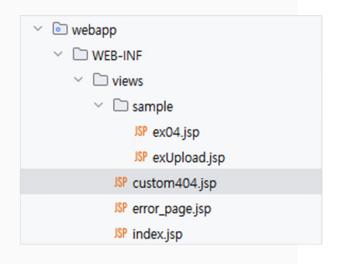
    // 파일 업로드 설정
...
}
```

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)
            public String handle404(NoHandlerFoundException ex) {
    log.error(ex);
    model.addAttribute("uri", request.getRequestURI());
    return "custom404";
```

views/custom404.jsp

```
<!DOCTYPE html>
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>해당 URL(${uri})은 존재하지 않습니다.</h1>
</body>
</html>
```



☑ 404 에러 페이지

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

해당 URL(/sample/badurl)은 존재하지 않습니다.