KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

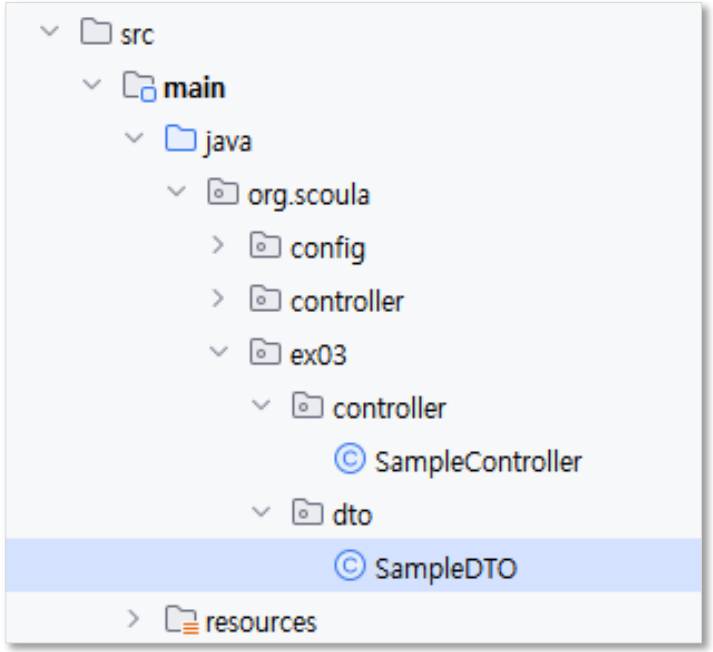
스프링 MVC의 Controller (심화 1)

[KB] IT's Your Life

KB 국민은행

스프링 MVC의 Controller

- **SampleDTO 클래스를 다음과 같이 생성하세요.**
  - @Data 설정
  - 멤버 변수
    - private String name;
    - private int age;



스프링 MVC의 Controller

SampleDTO.java

```
package org.scoula.ex03.dto;

import lombok.Data;

@Data
public class SampleDTO {
    private String name;
    private int age;
}
```

스프링 MVC의 Controller

SampleController에 다음 작업을 진행하세요.

- /sample/ex01의 GET 요청에 호출되는 ex01() 메서드 추가
  - SampleDTO 객체를 매개변수로 주입받음
  - SampleDTO의 내용을 로그로 출력
  - ex01 뷰이름 리턴

- 다음과 같이 요청을 보냈을 때 로그를 확인하세요.
  - http://localhost:8080/sample/ex01
  - http://localhost:8080/sample/ex01?name=AAA&age=10

스프링 MVC의 Controller

SampleController.java

```
...
public class SampleController {
        ...

        @GetMapping("/ex01")
        public String ex01(SampleDTO dto) {
                log.info("" + dto);

                return "ex01";
        }

}
```

스프링 MVC의 Controller

확인

http://localhost:8080/sample/ex01

INFO : org.scoula.ex03.controller.SampleController - SampleDTO(name=null, age=0)

http://localhost:8080/sample/ex01?name=AAA&age=10

INFO : org.scoula.ex03.controller.SampleController - SampleDTO(name=AAA, age=10)

스프링 MVC의 Controller

SampleController에 다음 작업을 진행하세요.

- /sample/ex02의 GET 요청에 호출되는 ex02() 메서드 추가
  - 쿼리 파라미터의 값을 받는 문자열 name과 정수 age를 매개변수 주입 지정
  - name과 age의 값을 로그로 출력
  - ex02 뷰이름 리턴

- **다음과 같이 요청을 보냈을 때 로그를 확인하세요.**
  - http://localhost:8080/sample/ex02?name=AAA&age=10
  - http://localhost:8080/sample/ex02?age=10
  - http://localhost:8080/sample/ex02?name=AAA

스프링 MVC의 Controller

SampleController.java

```
...
public class SampleController {
        ...

        @GetMapping("/ex02")
        public String ex02(
                        @RequestParam("name") String name,
                        @RequestParam("age") int age) {
                log.info("name: " + name);
                log.info("age: " + age);

                return "ex02";
        }
}
```

http://localhost:8080/sample/ex02?name=AAA&age=10

INFO : org.scoula.ex03.controller.SampleController - name: AAA
INFO : org.scoula.ex03.controller.SampleController - age: 10

스프링 MVC의 Controller

SampleController에 다음 작업을 진행하세요.

- /sample/ex02List의 GET 요청에 호출되는 ex02List() 메서드 추가
  - ids라는 쿼리 파라미터가 여러 개 전달된 경우를 처리하는 ids 매개변수 주입을 지정하세요.
  - 매개변수 타입을 ArrayList<String>로 처리한 경우와 String[]로 처리한 경우 모두 확인
  - ids의 값을 로그로 출력
  - ex02List 뷰이름 리턴

- **다음과 같이 요청을 보냈을 때 로그를 확인하세요.**
  - http://localhost:8080/sample/ex02List?ids=111&ids=222&ids=333

스프링 MVC의 Controller

SampleController.java

```
...
public class SampleController {
        ...


        @GetMapping("/ex02List")
        public String ex02List(@RequestParam("ids") ArrayList<String> ids) {
                log.info("ids: " + ids);

                return "ex02List";
        }
}
```

http://localhost:8080/sample/ex02List?ids=111&ids=222&ids=333

INFO : org.scoula.ex03.controller.SampleController - ids: [111, 222, 333]

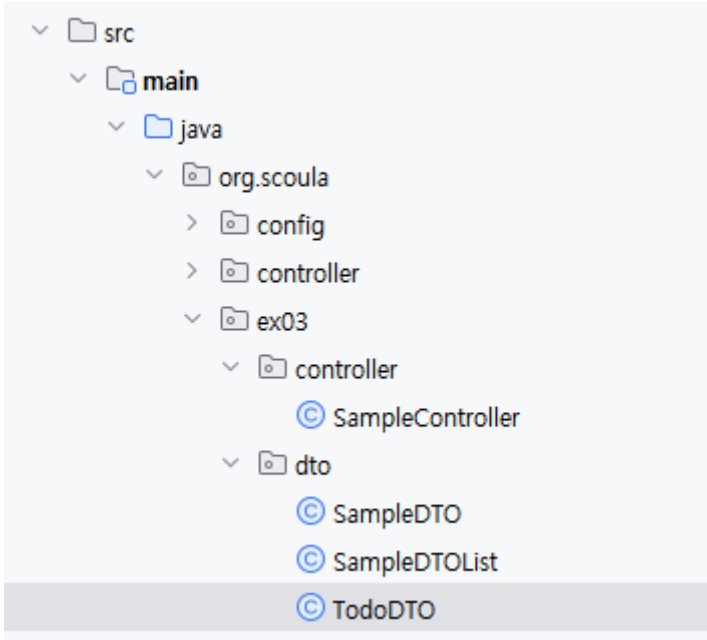
스프링 MVC의 Controller

SampleController.java

```
…
public class SampleController {
        …


        @GetMapping("/ex02List")
        public String ex02List(@RequestParam("ids") String[] ids) {
                log.info("array ids: " + Arrays.toString(ids));

                return "ex02List";
        }
}
```

스프링 MVC의 Controller

- **다음과 같이 TodoDTO를 작성하세요.**
  - @Data 설정
  - 멤버 변수
    - private String title;
    - private Date dueDate;
  - dueDate에 "yyyy-MM-dd" 포맷팅을 지정



TodoDTO.java

```
package org.scoula.ex03.dto;

import java.util.Date;

import org.springframework.format.annotation.DateTimeFormat;

import lombok.Data;

@Data
public class TodoDTO {
		private String title;

		@DateTimeFormat(pattern="yyyy-MM-dd")
		private Date dueDate;
}
```

스프링 MVC의 Controller

SampleController에 다음 작업을 진행하세요.

- /sample/ex03 GET 요청에 호출되는 ex03() 메서드 추가
  - TodoDTO로 매개변수 주입 지정
  - TodoDTO의 값을 로그로 출력
  - ex03 뷰이름 리턴

- **다음과 같이 요청을 보냈을 때 로그를 확인하세요.**
  - http://localhost:8080/sample/ex03?title=test&dueDate=2023-01-01

스프링 MVC의 Controller

org.scoula.ex03.controller.SampleController.java

```
…
public class SampleController {
        …

        @GetMapping("/ex03")
        public String ex03(TodoDTO todo) {
                log.info("todo: " + todo);
                return "ex03";
        }
}
```

http://localhost:8080/sample/ex03?title=test&dueDate=2023-01-01

INFO : org.scoula.ex03.controller.SampleController - todo: TodoDTO(title=test, dueDate=Sun Jan 01 00:00:00 KST 2023)

스프링 MVC의 Controller

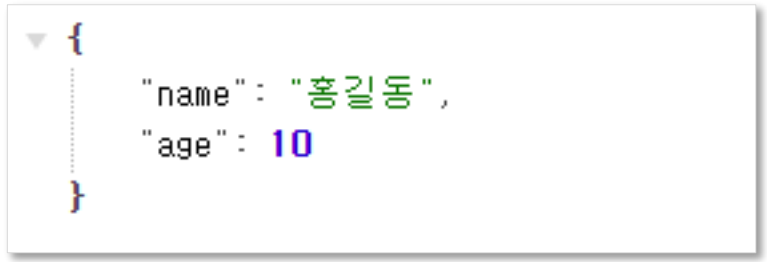
- **SampleController에 다음을 처리하는 ex07() 메서드를 추가하세요.**
  - 요청 URL: http://localhost:8080/sample/ex07
  - SampleDTO의 인스턴스 구성(name, age 설정)
  - 이 객체를 json으로 응답

스프링 MVC의 Controller

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



스프링 MVC의 Controller

- **SampleController에 다음을 처리하는 ex08() 메서드를 추가하세요.**
  - 요청 URL: http://localhost:8080/sample/ex08
  - ResponseEntity를 이용하여 응답
    - 응답 코드: 200
    - 헤더: Content-Type을 json으로 직접 설정
    - body: {name: "홍길동"} json 문자열

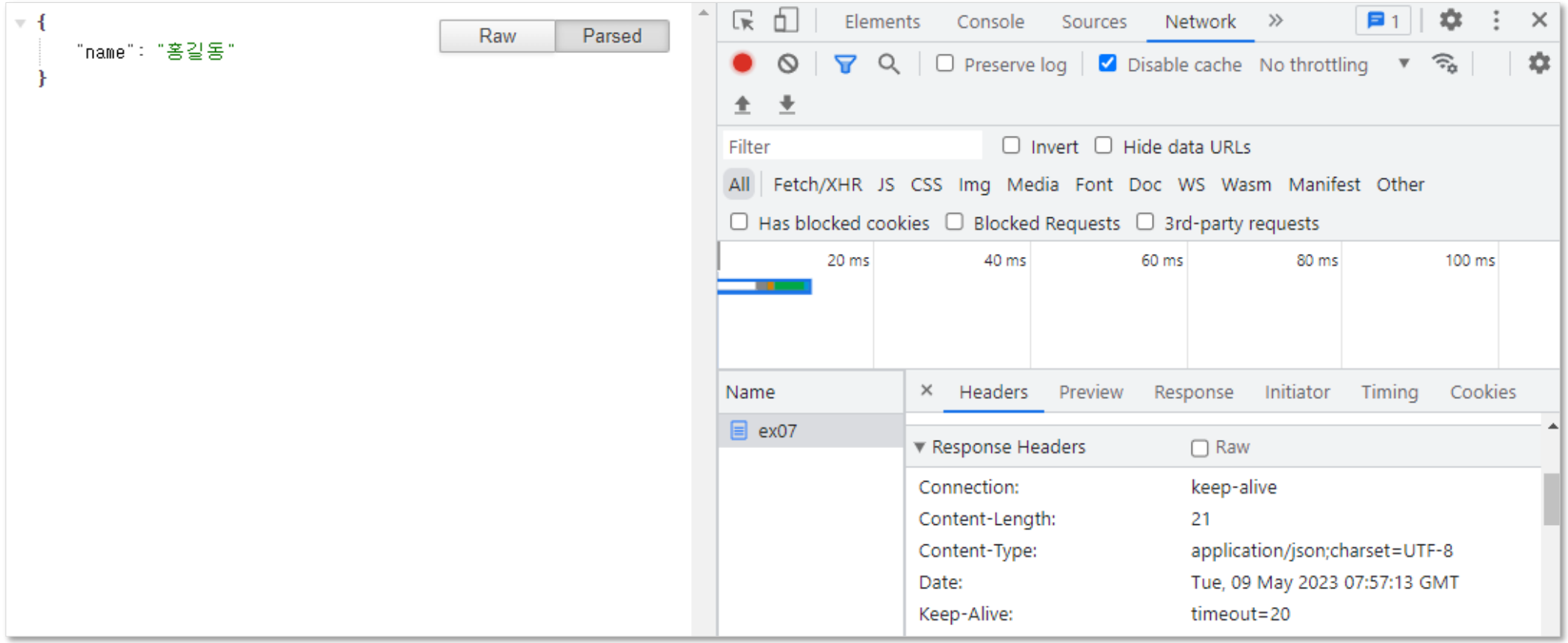
스프링 MVC의 Controller

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);
        }
}
```

스프링 MVC의 Controller

http://localhost:8080/sample/ex08



스프링 MVC의 Controller

- **스프링 MVC에서 예외 처리를 일괄 처리하는 Advice 클래스와 뷰를 작성하세요.**
  - 패키지: org.scoula.exception
  - 클래스: CommonExceptionAdvice
  - error 페이지 뷰: error_page

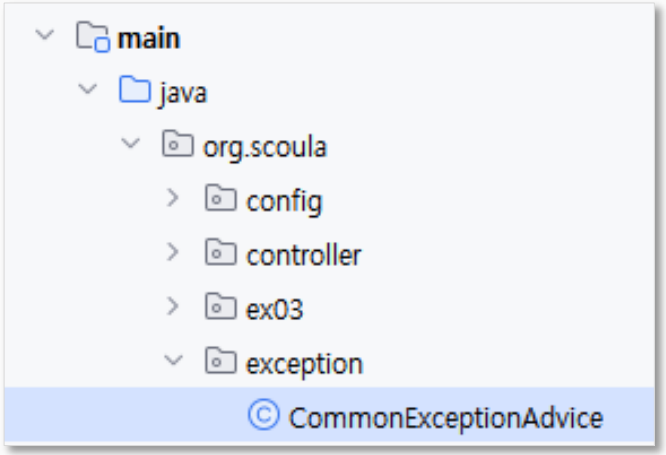
스프링 MVC의 Controller

- **@ControllerAdvice**
  - HTTP 상태코드 500 Internal Serveer Error에 대응하기 위한 기법
  - AOP(Aspect-Oriented-Programming)을 이용

  - 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";
        }

}
```



스프링 MVC의 Controller

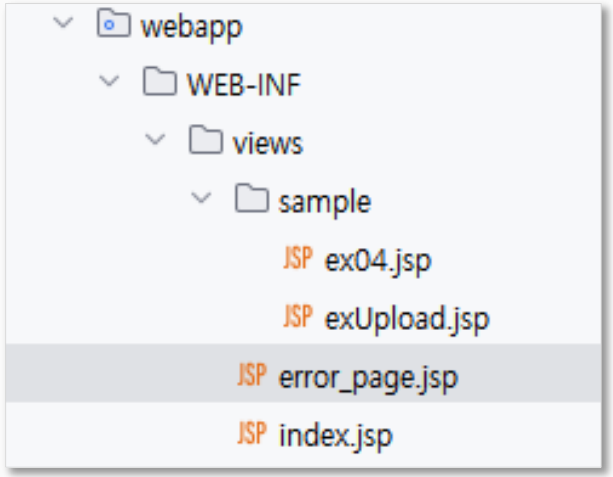
ServletConfig.java

```
package org.scoula.config;

...

@EnableWebMvc
@ComponentScan(basePackages = {
        "org.scoula.controller",
                        "org.scoula.exception",
        "org.scoula.ex03.controller"
}
public class ServletConfig implements WebMvcConfigurer{
   ...
}
```

스프링 MVC의 Controller

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>

  <ul>
   <c:forEach items="${exception.getStackTrace() }" var="stack">
     <li><c:out value="${stack}"></c:out></li>
   </c:forEach>
  </ul>

</body>
</html>
```



스프링 MVC의 Controller

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.invokeHandlerM
- org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.handleInternal(
- org.springframework.web.servlet.mvc.method.AbstractHandlerMethodAdapter.handle(AbstractHandlerMet
- org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:991)
- org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:925)

스프링 MVC의 Controller

- 404 에러 페이지를 처리하는 코드를 추가하세요.

스프링 MVC의 Controller

WebConfig.java

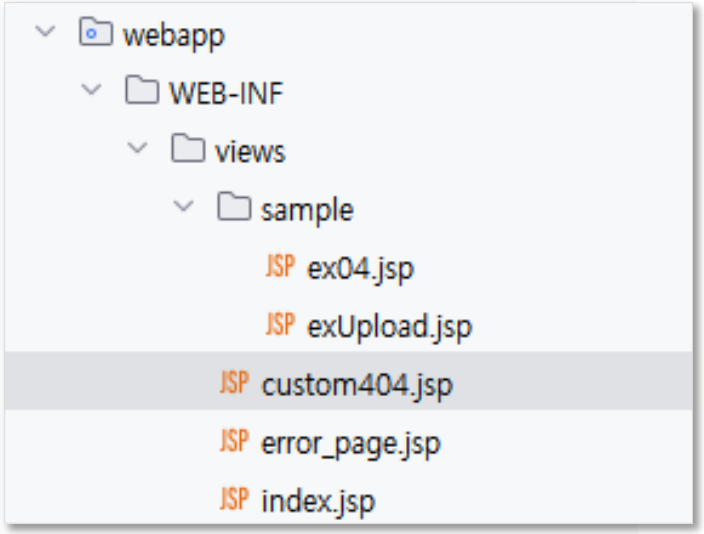
```
package org.scoula.config;
…

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) {

                    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 URL은 존재하지 않습니다.</h1>
</body>
</html>
```



KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

스프링 MVC의 Controller (심화 2)

[KB] IT's Your Life

KB 국민은행

스프링 MVC의 Controller

- **Servlet 3.0의 파일 업로드를 위한 설정을 추가하세요.**
  - 업로드 디렉토리: c:₩upload
  - 파일 하나당 업로드 가능한 최대 크기: 10M
  - 요청당 업로드 가능한 최대 크기: 20M
  - 업로드 시 메모리 파일로 처리할 최대 크기: 5M

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;
        }
}
```

스프링 MVC의 Controller

WebConfig.java

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

@Slf4j
@Configuration
public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer {
    final String LOCATION = "c:/upload";
    final long MAX_FILE_SIZE = 1024 * 1024 * 10L;
    final long MAX_REQUEST_SIZE =  1024 * 1024 * 20L;
    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)
                FILE_SIZE_THRESHOLD                 // 메모리 파일의 최대 크기(이보다 작으면 실제 메모리에서만 작업)
            );
        registration.setMultipartConfig(multipartConfig);
    }
}
```

스프링 MVC의 Controller

- **SampleController에 다음을 처리하는 exUpload() 메서드와 뷰를 추가하세요.**
  - 요청 url: /sample/exUpload
  - 요청 메서드 : GET
  - 뷰: sample/exUpload
    - 파일을 최대 5개 업로드할 수 있는 form 작성
    - action은 /sample/exUploadPost

- **업로드된 파일을 처리하는 exUploadPost() 메서드를 추가하세요.**
  - 파일의 원본 파일명과 파일 크기 출력

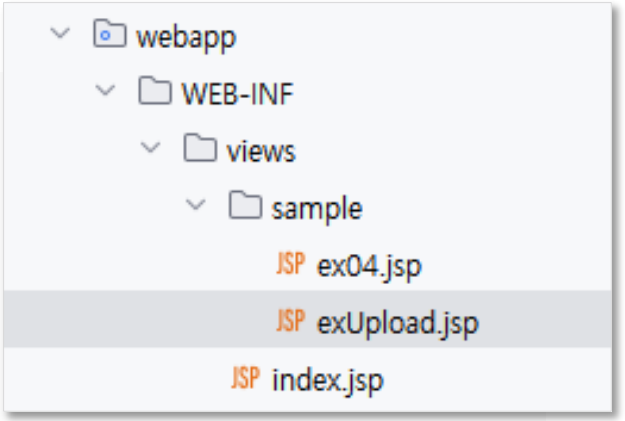
스프링 MVC의 Controller

SampleController.java

```
...
public class SampleController {
   ...

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

}
```

스프링 MVC의 Controller

views/sample/exUpload.jsp

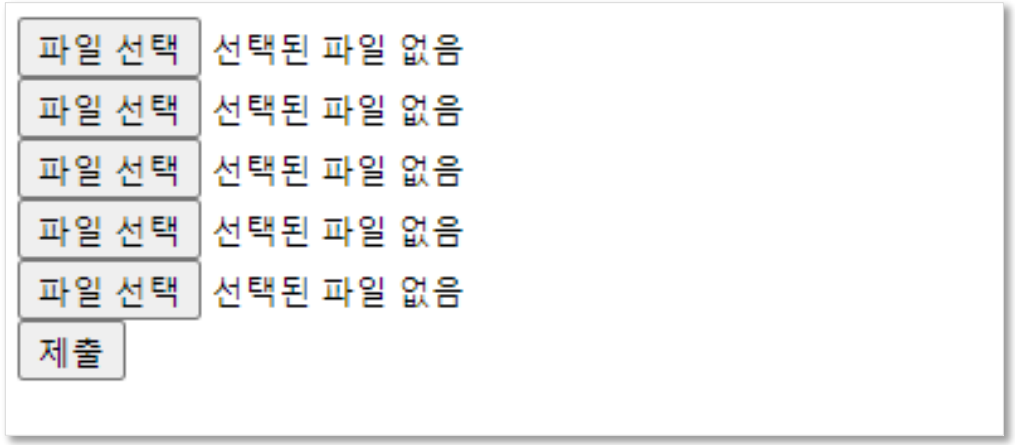



```
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Insert title here</title>
  </head>
  <body>
    <form action="/sample/exUploadPost" method="post" enctype="multipart/form-data" >
      <div>
        <input type="file" name="files" />
      </div>
      <div>
        <input type="file" name="files" />
      </div>
      <div>
        <input type="file" name="files" />
      </div>
      <div>
        <input type="file" name="files" />
      </div>
      <div>
        <input type="file" name="files" />
      </div>
```

스프링 MVC의 Controller

views/sample/exUpload.jsp

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



스프링 MVC의 Controller

SampleController.java

```
...
public class SampleController {
   ...

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

                for(MultipartFile file : files) {
                        log.info("-------------------------------------");
                        log.info("name:" + file.getOriginalFilename());
                        log.info("size:" + file.getSize());
                }

        }
}
```

스프링 MVC의 Controller