

6장. 예외 처리 & 파일 업로드



exception & file



Spring Boot

■ 예외 처리

1. 예외의 개념

사용자가 시스템을 사용하다 보면 웹 페이지 URL 경로를 잘못 입력하거나 파라미터를 잘못 전달하여 문제가 발생하는 경우가 있다.

또는 서버에서 실행되는 프로그램에서 생각하지 못했던 문제가 발생할 수 도 있다.

이렇게 사용자의 부주의나 코드 자체의 오류로 인해 문제가 발생했을때, 발생한 문제를 적절하게 처리하지 않으면 사용자 브라우저에서 에러 메시지가 출력된다.

2. 스프링부트에서 예외를 처리하는 방법

- 1) @ControllerAdvice 어노테이션을 이용하여 모든 컨트롤러에서 발생하는 예외를 일괄적으로 처리한다. (전역 예외처리)
- 2) @ExceptionHandler 어노테이션을 이용하여 각 컨트롤러마다 발생하는 예외를 개별적으로 처리한다. (로컬 예외처리)



예외 처리

@ControllerAdvice

- 예외 처리와 원래의 컨트롤러가 혼합된 형태의 클래스가 작성되는 방식
- @ExceptionHandler는 해당 메서드가 들어가는 예외 타입을 처리한다

//코드 및 인증 오류 등 처리 - code 500 에러

@ExceptionHandler(Exception.class)

//페이지를 찾을 수 없음 - 404 에러

@ExceptionHandler(NoHandlerFoundException.class)



예외 처리

사용자 정의 예외

- BoardException 만들기

```
src/main/java
├── com.boot
│   ├── com.boot.controller
│   │   ├── BoardController.java
│   │   ├── ExceptionController.java
│   │   └── LoginController.java
│   ├── com.boot.domain
│   └── com.boot.exception
│       ├── BoardException.java
│       ├── BoardNotFoundException.java
│       └── GlobalExceptionHandler.java
└── com.boot.persistence
```

```
package com.boot.exception;

public class BoardException extends RuntimeException{

    private static final long serialVersionUID = 31L;

    public BoardException(String message) {
        super(message);
    }

}
```



예외 처리

사용자 정의 예외

- Board 엔티티가 없을 때 사용할 BoardNotFoundException 클래스

```
package com.boot.exception;

public class BoardNotFoundException extends BoardException{

    private static final long serialVersionUID = 41L;

    public BoardNotFoundException(String message) {
        super(message);
    }
}
```



예외 처리

예외 발생하기

- ## - index.html에 예외를 발생시킬 링크 추가

[illegible]

예외 처리

- ExceptionController 클래스 만들기

```
@Controller
public class ExceptionController {

    @GetMapping("/boardError")
    public String boardError() {
        throw new BoardNotFoundException("검색된 게시글이 없습니다.");
    }

    @GetMapping("/illegalArgumentError")
    public String illegalArgumentError() {
        throw new IllegalArgumentException("부적절한 인자가 전달되었습니다.");
    }

    @GetMapping("/sqlError")
    public String sqlError() throws SQLException{
        throw new SQLException("SQL 구문에 오류가 있습니다.");
    }
}
```



예외 처리

➤ 예외 처리하기

1. 예외 처리기 작성

```
@ControllerAdvice
public class GlobalExceptionHandler {

    @ExceptionHandler(BoardException.class)
    public String handleCustomException(BoardException exception, Model model) {
        model.addAttribute("exception", exception);
        return "/errors/boardError";
    }

    @ExceptionHandler(Exception.class)
    public String handleException(Exception exception, Model model) {
        model.addAttribute("exception", exception);
        return "/errors/globalError";
    }
}
```



예외 처리

➤ 예외 처리하기

2. 예외 전용 페이지

/templates/errors/boardError.html

```
<div id="container">
  <h2>Boardexception 발생!</h2>
  <a th:href="@{/}">메인 화면으로</a><hr>
  <table class="exception">
    <tr>
      <th th:text="${exception.message}"></th>
    </tr>
    <tr th:each="trace : ${exception.stackTrace}">
      <td th:text="${trace}">
    </tr>
  </table>
</div>
```

```
src/main/resources
├── static
├── templates
│   └── errors
│       ├── boardError.html
│       ├── globalError.html
│       ├── getBoard.html
│       ├── getBoardList.html
│       ├── hello.html
│       ├── index.html
│       ├── insertBoard.html
│       └── login.html
```



예외 처리

➤ 예외 처리하기

2. 예외 전용 페이지

/templates/errors/globalError.html

```
<div id="container">
  <h2>Exception 발생!</h2>
  <a th:href="@{/}">메인 화면으로</a><hr>
  <table class="exception">
    <tr>
      <th th:text="${exception.message}"></th>
    </tr>
    <tr th:each="trace : ${exception.stackTrace}">
      <td th:text="${trace}">
    </tr>
  </table>
</div>
```



예외 처리

➤ 예외 처리하기

게시판 프로그램입니다.



[글 목록](#) [로그인](#)

`BoardException` 발생 `IllegalArgumentException` 발생 `SQLException` 발생



예외 처리

➤ 예외 처리하기

Boardexception 발생!

[메인 화면으로](#)

검색된 게시글이 없습니다.

com.boot.controller.ExceptionController.boardError(ExceptionController.java:14)

java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)

java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:77)

Exception 발생!

[메인 화면으로](#)

부적절한 인자가 전달되었습니다.

com.boot.controller.ExceptionController.illegalArgumentError(ExceptionController.java:19)

java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)


java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:77)



예외 처리

➤ 예외 테스트하기

1) 게시물 번호가 없는 경우

 localhost:8181/getBoard?seq=2500

글 상세 보기

제목	첫 번째 게시물
작성자	테스터
	등록이 잘 되네요..

Exception 발생!

[메인 화면으로](#)

No value present
java.base/java.util.Optional.get(Optional.java:143)
com.boot.service.BoardServiceImpl.getBoard(BoardServiceImpl.java:32)



예외 처리

➤ 예외 테스트하기

2) 게시물 번호를 문자로 입력한 경우

localhost:8181/getBoard?seq=abc

글 상세 보기

제목	첫 번째 게시물
작성자	테스터
	등록이 잘 되네요 ..

Exception 발생!

[메인 화면으로](#)

Failed to convert value of type 'java.lang.String' to required type 'java.lang.Long'; nested exception is java.lang.NumberFormatException: "abc"

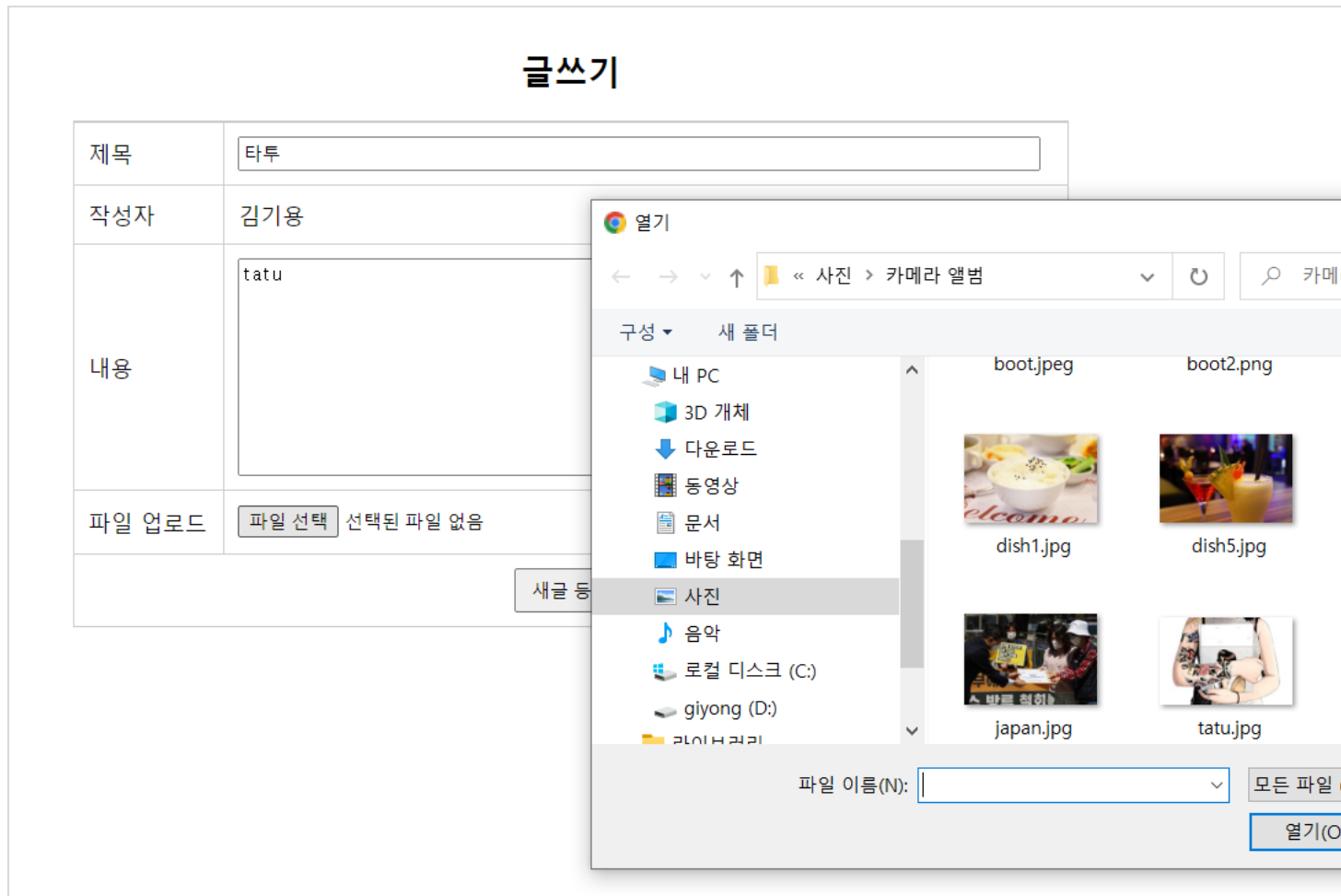
org.springframework.web.method.annotation.AbstractNamedValueMethodArgumentResolver.resolveArgument(AbstractNamedValueMethodArgumentResolver.java:112)

org.springframework.web.method.support.HandlerMethodArgumentResolverComposite.resolveArgument(HandlerMethodArgumentResolverComposite.java:121)



파일 업로드

➤ 파일 업로드



파일 업로드

➤ 파일 업로드 관련 설정

Spring boot에서는 multipartResolver 가 빈으로 등록되어 있어서 파일 용량 업로드 경로등을 설정해 주면 됨

```
# 파일 업로드
spring.servlet.multipart.location=c:/upload
spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=50MB
```



파일 업로드

➤ insertBoard.html 폼 속성 수정

파일 업로드 항목이 있을 경우는 반드시 form의 enctype 속성을 "**multipart/form-data**"로 추가 해야 한다.

```
<form th:action="@{insertBoard}" method="post" enctype="multipart/form-data">
  <table class="tbl_reg">
    <tr>
      <td>제목</td>
      <td><input type="text" name="title"></td>
    </tr>
    <tr>
      <td>작성자</td>
      <td>sec:authentication="principal.member.name"</td>
    </tr>
    <tr>
      <td>내용</td>
      <td>
        <textarea name="content" rows="10" cols="50"></textarea>
      </td>
    </tr>
    <tr>
      <td>파일 업로드</td>
      <td><input type="file" name="uploadFile"></td>
    </tr>
  </table>
</form>
```



파일 업로드

➤ FileDto 만들기

```
package com.boot.dto;

import lombok.Getter;

@Getter @Setter
public class FileDto {
    private String uuid;           //unique 한 파일 이름을 만들 필드
    private String fileName;       //파일 이름
    private String contentType;    //파일 유형

    public FileDto() {}

    public FileDto(String uuid, String fileName, String contentType) {
        this.uuid = uuid;
        this.fileName = fileName;
        this.contentType = contentType;
    }
}
```



파일 업로드

➤ BoardController 수정하기

```
@PostMapping("/insertBoard")
public String insertBoard(Board board, @RequestParam MultipartFile[] uploadFile,
    @AuthenticationPrincipal SecurityUser principal) throws IllegalStateException,
//파일 업로드
//MultipartFile 배열을 파라미터로 전달
for(MultipartFile file : uploadFile) {
    if(!file.isEmpty()) {
        FileDto dto = new FileDto(UUID.randomUUID().toString(),
            file.getOriginalFilename(), file.getContentType());

        File newFileName = new File(dto.getUuid() + "_" + dto.getFileName());
        //전달된 내용을 실제 물리적인 파일로 저장해 줌
        file.transferTo(newFileName);
    }
}
```



파일 업로드

➤ 업로드한 결과

