

Spring Framework

23. Spring MVC 어플리케이션 작성(3)

CONTENTS

1

사용자 수정화면 기능 구현

2

사용자 수정 및 삭제 기능 구현

3

Spring MVC의 예외 처리

학습 목표

- 사용자 수정화면 기능 구현에 대하여 이해할 수 있습니다.
- 사용자 수정 및 삭제 기능 구현에 대하여 이해할 수 있습니다.
- Spring MVC의 예외 처리에 대해 이해할 수 있습니다.

A person's hands are shown holding a black smartphone. The phone's screen is white and blank. The background is dark with out-of-focus circular light spots in shades of yellow, orange, and blue, creating a bokeh effect. A semi-transparent dark banner is at the bottom, containing a yellow decorative element and a title in Korean.

1. 사용자 수정화면 기능 구현

■ 사용자 정보 수정화면 : Controller와 JSP 구현 절차

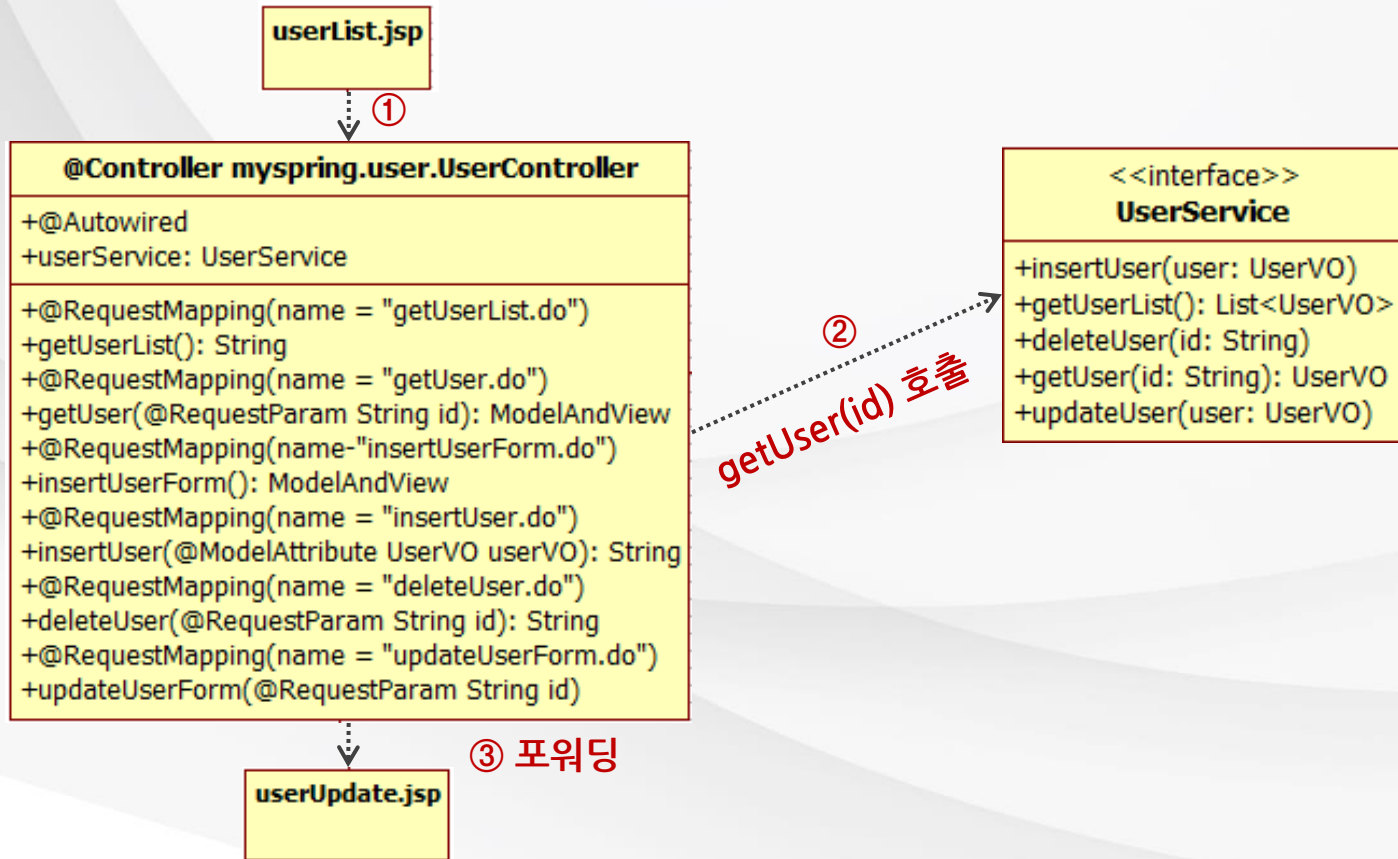
1 사용자 정보를 수정하는 화면을 포워딩 해주는 updateUserForm (@RequestParam String id) 메서드를 작성하고 @RequestMapping과 @RequestParam 어노테이션을 선언

2 userList.jsp 페이지를 수정

3 userUpdate.jsp 페이지에 View 영역의 코드를 작성

4 Browser 상에서 JSP를 실행

■ 사용자 정보 수정화면 : Controller와 JSP 호출 순서



■ 사용자 정보 수정화면 : Controller와 JSP 구현

1.userList.jsp

userList.jsp

```
<c:forEach var= "user" items= "${userList}">
  <tr>
    <td>
      <a href= "getUser.do?id=${user.userId}"> ${user.userId} </a>
    </td>
    <td> ${user.name} </td>
    <td> ${user.gender} </td>
    <td> ${user.city} </td>
    <td> <a href= "updateUserForm.do?id=${user.userId}"> 수정 </a> </td>
    <td> <a href= "deleteUser.do/${user.userId}"> 삭제 </a> </td>
  </tr>
</c:forEach>
```

■ 사용자 정보 수정화면 : Controller와 JSP 구현

2. UserController.java

```
*UserController.java

@RequestMapping("/updateUserForm.do")
public ModelAndView updateUserForm(@RequestParam String id) {
    UserVO user = userService.getUser(id);
    List<String> genderList = new ArrayList<String>();
    genderList.add("남");
    genderList.add("여");
    List<String> cityList = new ArrayList<String>();
    cityList.add("서울");
    cityList.add("부산");
    cityList.add("대구");
    cityList.add("제주");
    Map<String, Object> map = new HashMap<String, Object>();
    map.put("genderList", genderList);
    map.put("cityList", cityList);
    map.put("user", user);
    return new ModelAndView("userUpdate", "map", map);
}
```


■ 사용자 정보 수정화면 : Controller와 JSP 구현

3. updateUser.jsp

```
userUpdate.jsp
<h2 class="text-center">사용자 정보 수정</h2>
<tr>
  <td>거주지|:</td>
  <td>
    <select name="city">
      <c:forEach items=${map.cityList} var='cityName'>
        <c:choose>
          <c:when test='${cityName eq map.user.city}'>
            <option value='${cityName}' selected>${cityName}</option>
          </c:when>
          <c:otherwise>
            <option value='${cityName}'>${cityName}</option>
          </c:otherwise>
        </c:choose>
      </c:forEach>
    </select>
  </td>
</tr>
```

■ 사용자 정보 수정화면 : 결과 화면

사용자 목록

아이디	이름	성별	거주지		
jay	박정우	남	부산	수정	삭제
polar	연아	여	부산	수정	삭제
vega2k	박소울	여	제주	수정	삭제
사용자 등록					



사용자 정보 수정

아이디 :	vega2k
이름 :	<input type="text" value="박소울"/>
성별 :	<input type="radio"/> 남 <input checked="" type="radio"/> 여
거주지 :	<input type="text" value="제주"/>
<input type="button" value="수정"/>	

A person's hands are shown holding a black smartphone with a white screen. The person is wearing a dark jacket. The background is dark with out-of-focus, warm-toned circular lights (bokeh) in shades of yellow, orange, and blue. A semi-transparent dark blue banner is at the bottom, containing a yellow chevron icon and the title text.

2. 사용자 수정 및 삭제 기능 구현

■ 사용자 정보 수정 : Controller와 JSP 구현 절차

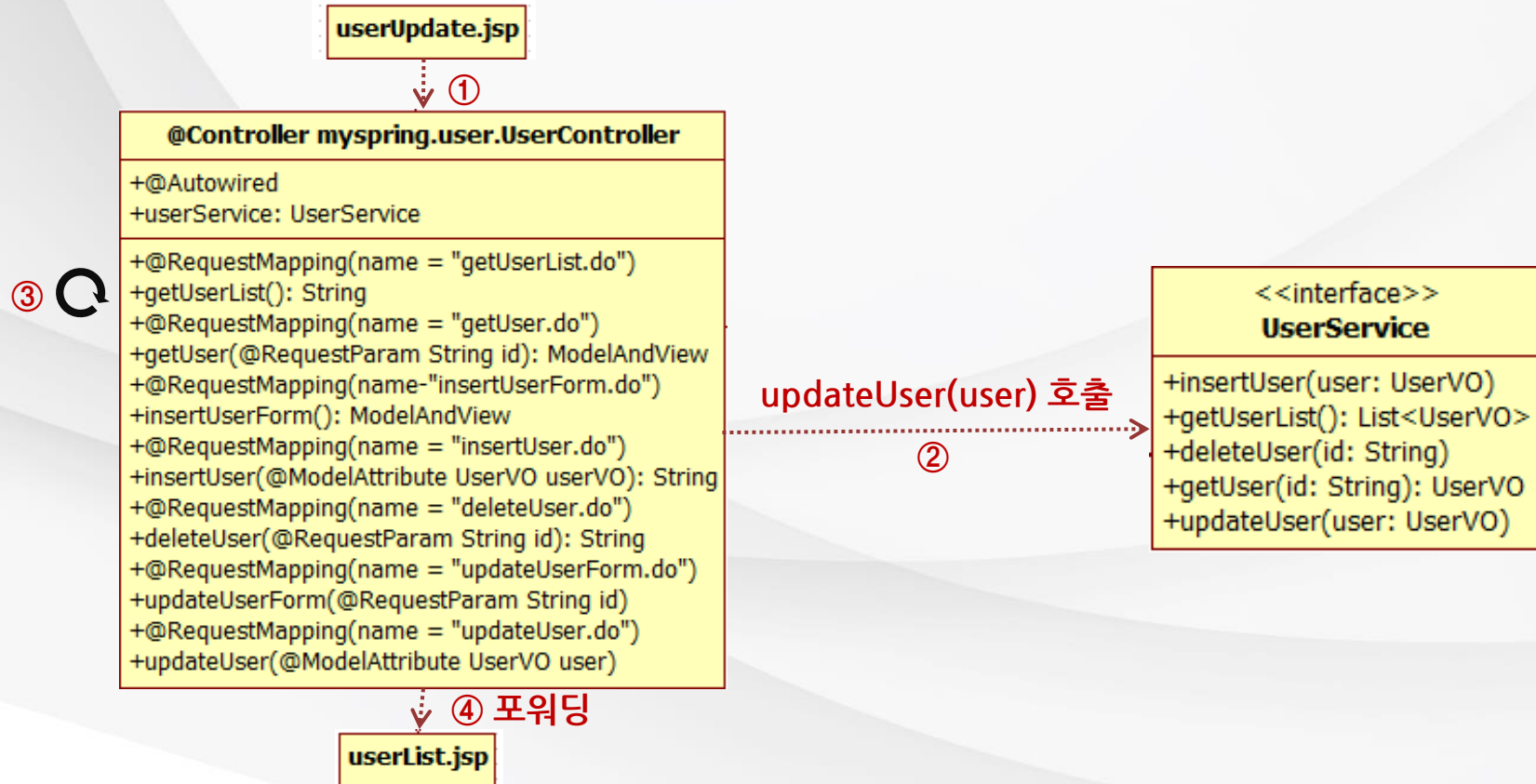
1 사용자 정보를 수정하는 화면을 포워딩 해주는 updateUser (@ModelAttribute UserVO user) 메서드를 작성하고 @RequestMapping과 @ModelAttribute 어노테이션을 선언

: 수정 후에 목록 조회가 redirect 되도록 하여, 수정된 사용자 정보를 확인할 수 있도록 해야 함

2 userUpdate.jsp 페이지에 View 영역의 코드를 작성

3 Browser 상에서 JSP를 실행

■ 사용자 정보 수정 : Controller와 JSP 호출 순서



■ 사용자 정보 수정 : Controller와 JSP 구현

1. updateUser.jsp

```
userUpdate.jsp
<h2 class="text-center">사용자 정보 수정</h2>
<form method="post" action="updateUser.do">
  <input type="hidden" name="userId" value="${map.user.userId}" />
  <table class="table table-bordered table table-hover">
    <tr>
      <td>아이디 :</td>
      <td>${map.user.userId}</td>
    </tr>
    <tr>
      <td>이름 :</td>
      <td><input type="text" name="name" value="${map.user.name}" />
    </td>
    </tr>
    .....
    <tr>
      <td colspan="2" class="text-center"><input type="submit" value="수정" /></td>
    </tr>
  </table>
</form>
```

■ 사용자 정보 수정 : Controller와 JSP 구현

2. UserController.java

```
UserController.java ✕  
  
@RequestMapping("/updateUser.do")  
public String updateUser(@ModelAttribute UserVO user) {  
    userService.updateUser(user);  
    return "redirect:/getUserList.do";  
}
```

2. 사용자 수정 및 삭제 기능 구현

■ 사용자 정보 수정 : 결과 화면

사용자 목록					
아이디	이름	성별	거주지		
jay	박정우	남	부산	수정	삭제
polar	연아	여	부산		
vega2k	박소울	여	제주		

사용자 정보 수정

아이디 :	vega2k
이름 :	<input type="text" value="박소울New"/>
성별 :	<input checked="" type="radio"/> 남 <input type="radio"/> 여
거주지 :	<input type="text" value="부산"/>
<input type="button" value="수정"/>	

사용자 목록			
아이디	이름	성별	거주지
jay	박정우	남	부산
polar	연아	여	부산
vega2k	박소울New	남	부산

■ 사용자 정보 삭제 : Controller와JSP 구현 절차

1

사용자 정보를 삭제하는 `deleteUser(@PathVariable String id)` 메서드를 작성하고 `@RequestMapping`과 `@PathVariable` 어노테이션을 선언

: 삭제 후에 목록 조회가 `redirect` 되도록 하여,
삭제된 사용자를 확인할 수 있도록 해야 함

2

`userList.jsp` 페이지를 수정

3

Browser 상에서 JSP를 실행

■ 사용자 정보 삭제 : Controller를 위한 핵심 어노테이션(Annotation)

구성요소	설명
@PathVariable	파라미터를 URL 형식으로 받을 수 있도록 해줌

```
userList.jsp
<c:forEach var="user" items="${userList}">
  <tr>
    <td>
      <a href="getUser.do?id=${user.userId}">${user.userId}</a>
    </td>
    <td>${user.name}</td>
    <td>${user.gender}</td>
    <td>${user.city}</td>
    <td>
      <a href="updateUserForm.do?id=${user.userId}">수정</a>
      <td><a href="deleteUser.do/${user.userId}">삭제</a></td>
    </tr>
</c:forEach>
```

```
UserController.java
package myspring.user.controller;

@Controller
public class UserController {
    @Autowired
    private UserService userService;

    @RequestMapping(value="/deleteUser.do/{id}")
    public String deleteUser(@PathVariable String id) {
        userService.deleteUser(id);
        return "redirect:/getUserList.do";
    }
}
```

■ @PathVariable 사용을 위한 DispatcherServlet의 url-pattern 변경

기
존

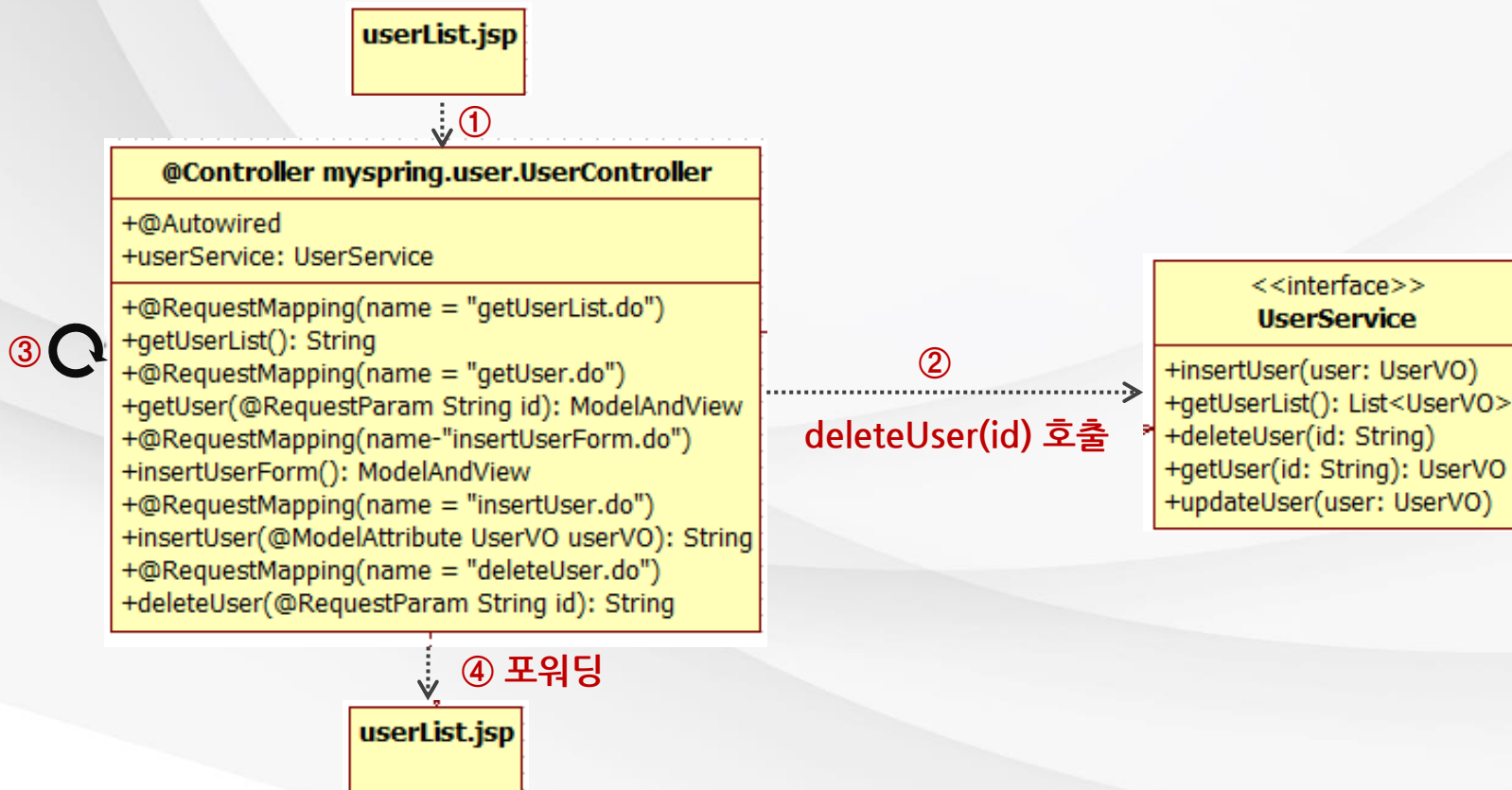
```
<servlet-mapping>  
  <servlet-name>springDispatcherServlet</servlet-name>  
  <url-pattern>*.do</url-pattern>  
</servlet-mapping>
```

변
경

```
<servlet-mapping>  
  <servlet-name>springDispatcherServlet</servlet-name>  
  <url-pattern>/</url-pattern>  
</servlet-mapping>
```



■ 사용자 정보 삭제 : Controller와 JSP 호출 순서



■ 사용자 정보 삭제 : Controller와 JSP 구현

1. userList.jsp

```
userList.jsp
<c:forEach var="user" items="${userList}">
  <tr>
    <td>
      <a href="getUser.do?id=${user.userId}">${user.userId}</a>
    </td>
    <td>${user.name}</td>
    <td>${user.gender}</td>
    <td>${user.city}</td>
    <td>
      <a href="updateUserForm.do?id=${user.userId}">수정</a>
    </td>
    <td><a href="deleteUser.do/${user.userId}">삭제</a></td>
  </tr>
</c:forEach>
```

■ 사용자 정보 삭제 : Controller와 JSP 구현

2. UserController.java

```
UserController.java ✕  
  
package myspring.user.controller;  
  
@Controller  
public class UserController {  
    @Autowired  
    private UserService userService;  
  
    @RequestMapping(value="/deleteUser.do/{id}")  
    public String deleteUser(@PathVariable String id) {  
        userService.deleteUser(id);  
        return "redirect:/getUserList.do";  
    }  
}
```

2. 사용자 수정 및 삭제 기능 구현

■ 사용자 정보 삭제 : 결과 화면

사용자 목록

아이디	이름	성별	거주지		
gildong	홍길동2	남2	경기2	수정	삭제
polar	연아2	여2	경기2	수정	삭제
vega2k	박소울	여	서울	수정	삭제



사용자 목록


아이디	이름	성별	거주지		
gildong	홍길동2	남2	경기2	수정	삭제
polar	연아2	여2	경기2	수정	삭제

A person's hands are shown holding a smartphone, with the screen glowing. The background is dark with out-of-focus, warm-toned bokeh lights. A semi-transparent dark banner is at the bottom, containing a yellow decorative element and the title text.

3. Spring MVC의 예외 처리

■ @ExceptionHandler 어노테이션(Annotation)의 사용

- ◉ 컨트롤러의 메서드에 @ExceptionHandler 어노테이션을 설정하여 컨트롤러의 메서드에서 예외가 발생했을 때 예외 처리를 할 수 있음
- ◉ 예외가 발생했을 때, 예외 Type과 Message를 보여주는 JSP 페이지(viewError.jsp)를 작성해야 함



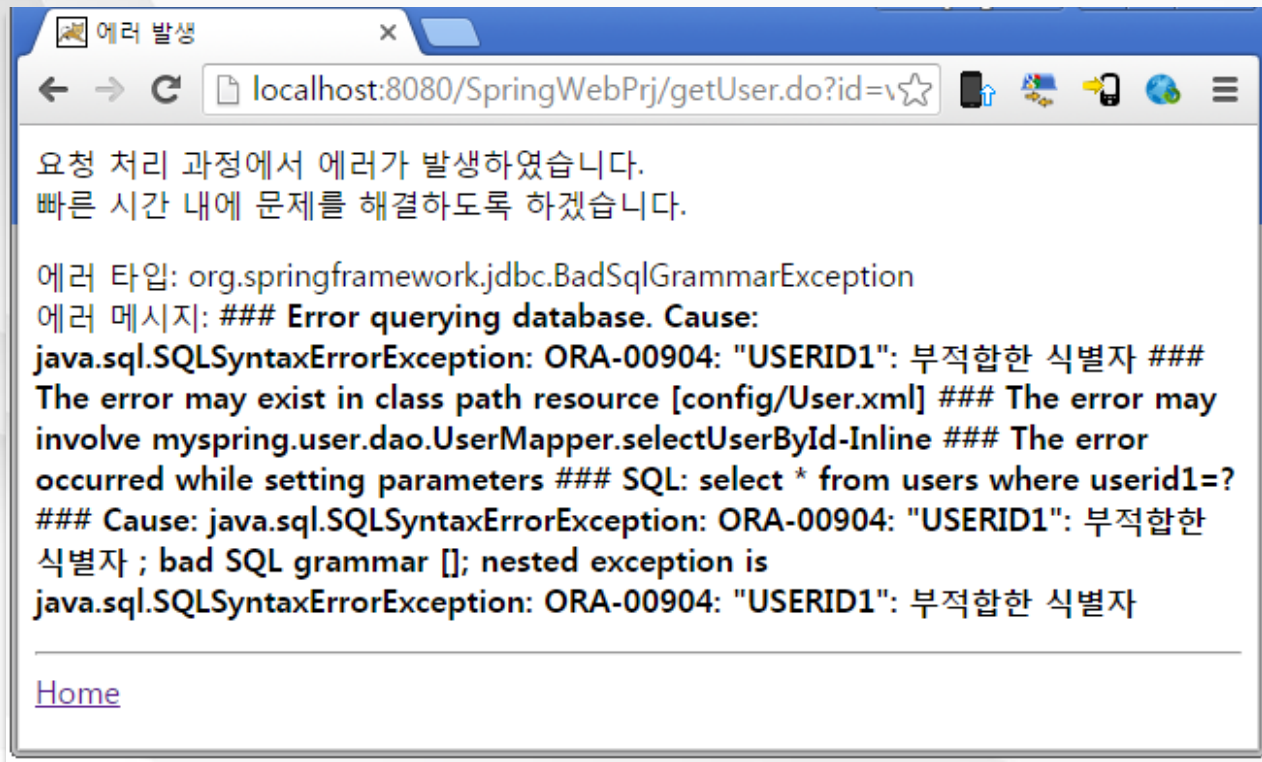
```
*UserController.java ✕  
  
@ExceptionHandler  
public String handleException(Exception e) {  
    return "viewError";  
}
```

■ Error Page 작성

viewError.jsp

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ page isErrorPage = "true" %>
<html>
<head><title>에러 발생</title></head>
<body>
  요청 처리 과정에서 에러가 발생하였습니다.<br>
  빠른 시간 내에 문제를 해결하도록 하겠습니다.
  <p>
    에러 타입: <%= exception.getClass().getName() %> <br>
    에러 메시지: <b><%= exception.getMessage() %></b>
  <hr>
  <a href= "${pageContext.request.contextPath}">Home</a>
</body>
</html>
```

■ Exception 발생 시 Error Page가 보여짐



지금까지 [Spring MVC 어플리케이션 작성(3)]에 대해서 살펴보았습니다.

사용자 수정화면 기능 구현

- UserService Bean을 호출하여 사용자 정보를 수정하기 전에 조회하는 기능 구현

사용자 수정 및 삭제 기능 구현

- UserService Bean을 호출하여 사용자 정보를 수정 및 삭제하는 기능 구현
- @ModelAttribute, @PathVariable의 어노테이션 사용

Spring MVC 의 예외 처리

- @ExceptionHandler 어노테이션의 사용
- Error Page 작성