



서버 프로그램 구현

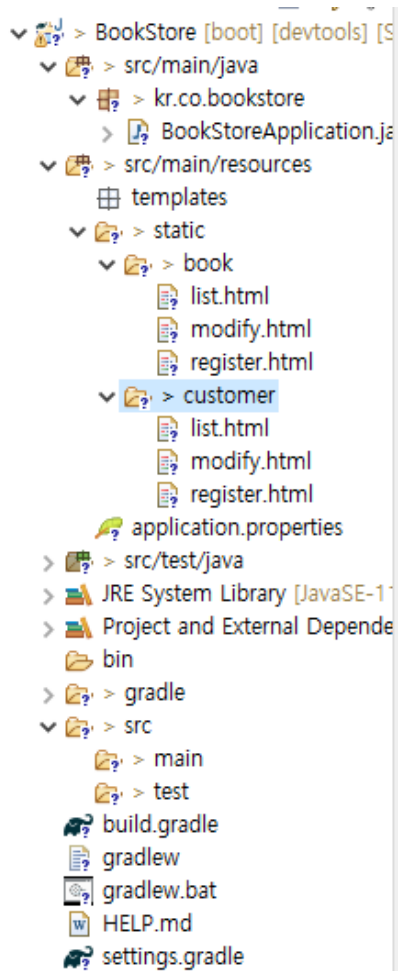
응용 프로그래밍 및 화면 구현

심규영

목차

- 1) 프로젝트 생성 및 구현
- 2) 화면 구현
- 3) 기능 구현
- 4) 실행

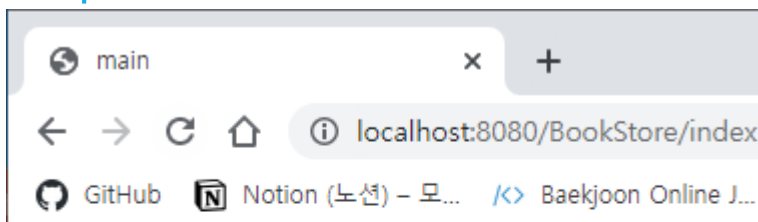
1) 프로젝트 생성 및 구현



- ProjectName : BookStore
- WAS : Tomcat 9(Spring boot app)
- DB : java2_bookstore
- Framework : Spring(SpringBoot), Mybatis
- Tools : Eclipse, Workbench

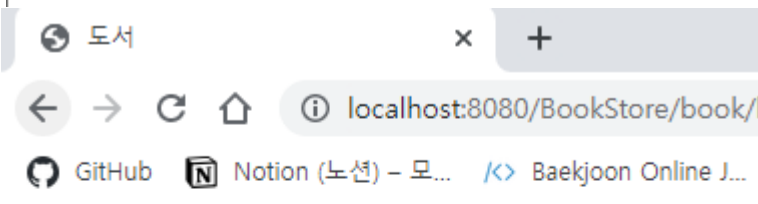
2) 화면 구현

- 메인(index)
- book(도서) 목록, 등록, 수정 화면 구현
- customer(고객) 목록, 등록, 수정 화면 구현



BookStore

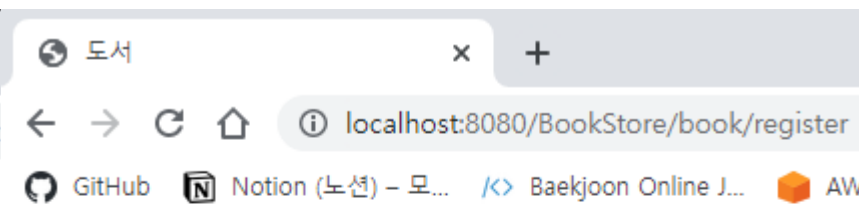
[도서목록](#) [고객목록](#)



도서목록

[처음으로 도서등록](#)

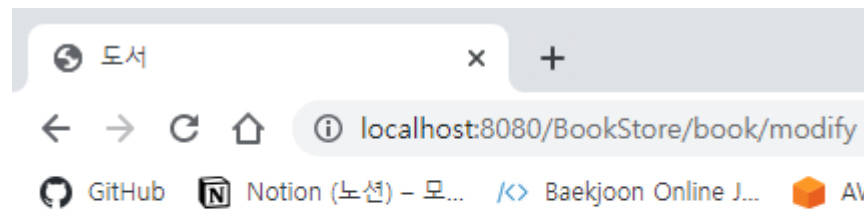
도서번호	도서명	출판사	가격	관리
1	축구의 역사	굿스포츠	7,000	수정 삭제



도서등록

[처음으로 도서목록](#)

도서명	<input type="text"/>
출판사	<input type="text"/>
가격	<input type="text"/>
<input type="button" value="등록"/>	



도서수정

[처음으로 도서목록](#)

도서번호	<input type="text"/>
도서명	<input type="text"/>
출판사	<input type="text"/>
가격	<input type="text"/>
<input type="button" value="수정"/>	

2) 화면 구현

고객 목록

[처음으로 고객등록](#)

고객번호	고객명	주소	휴대폰	관리
1	박지성	영국 맨체스터	010-1234-1001	수정 삭제

고객등록

[처음으로 고객목록](#)

고객명

주소

휴대폰

고객수정

[처음으로 고객목록](#)

고객번호

고객명

주소

휴대폰

3) 기능 구현

- 기본 설정
- VO 설정
- book.xml 설정

```
application.properties ×
1 server.servlet.context-path=/BookStore
2 server.port=8080
3 spring.thymeleaf.cache=false
4
5 # Mybatis 설정
6 spring.datasource.url=jdbc:mysql://127.0.0.1:3306/java2_bookstore
7 spring.datasource.username=root
8 spring.datasource.password=1234
9 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
10
11 #MyBatis Mapper 경로설정 -> Ch06Application 클래스 상단에 @MapperScan("kr
12 mybatis.mapper-locations=classpath:mappers/**/*.xml
13
```

```
@Getter
@Setter
@AllArgsConstructor
@NoArgsConstructor
@ToString
```

```
public class BookVO {
    private int bookId;
    private String bookName;
    private String publisher;
    private int price;
}
```

```
@Getter
@Setter
@AllArgsConstructor
@NoArgsConstructor
@ToString
```

```
public class CustomerVO {
    private int custId;
    private String name;
    private String address;
    private String phone;
}
```

```
book.xml ×
-//mybatis.org//DTD Mapper 3.0//EN (doctype)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper
3     PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4     "https://mybatis.org/dtd/mybatis-3-mapper.dtd">
5 <mapper namespace="book">
6     <insert id="insertBook">
7         INSERT INTO `book`(`bookName`,`publisher`,`price`) VALUES (#
8     </insert>
9     <select id="selectBook" resultType="kr.co.bookstore.VO.BookVO">
10         SELECT * FROM `book` WHERE `bookId`=#{bookId};
11     </select>
12     <select id="selectBooks" resultType="kr.co.bookstore.VO.BookVO">
13         SELECT * FROM `book`;
14     </select>
15     <update id="updateBook">
16         UPDATE `book` SET
17             `bookName`=#{bookName},
18             `publisher`=#{publisher},
19             `price`=#{price}
20         WHERE `bookId`=#{bookId};
21     </update>
22     <delete id="deleteBook">
23         DELETE FROM `book` WHERE `bookId`=#{bookId};
24     </delete>
25 </mapper>
```

3) 기능 구현

- customer.xml 설정
- BookService 설정

```
customer.xml ×
-//mybatis.org//DTD Mapper 3.0//EN (doctype)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper
3   PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4   "https://mybatis.org/dtd/mybatis-3-mapper.dtd">
5 <mapper namespace="customer">
6   <insert id="insertCustomer">
7     INSERT INTO `customer` (`name`,`address`,`phone`) VALUES (#{name},
8   </insert>
9   <select id="selectCustomer" resultType="kr.co.bookstore.VO.BookVO">
10     SELECT * FROM `customer` WHERE `custId`=#{custId};
11 </select>
12 <select id="selectCustomers" resultType="kr.co.bookstore.VO.BookVO">
13     SELECT * FROM `customer`;
14 </select>
15 <update id="updateCustomer">
16     UPDATE `customer` SET
17       `name`=#{name},
18       `address`=#{address},
19       `phone`=#{phone}
20     WHERE `custId`=#{custId};
21 </update>
22 <delete id="deleteCustomer">
23     DELETE FROM `customer` WHERE `custId`=#{custId};
24 </delete>
25 </mapper>
```

```
@Service
public class BookService {

    @Autowired
    private BookDAO dao;

    // service

    // create
    public void insertBook(BookVO vo) {
        dao.insertBook(vo);
    }

    // read
    public BookVO selectBook(int bookId) {
        return dao.selectBook(bookId);
    }
    public List<BookVO> selectBooks() {
        return dao.selectBooks();
    }

    // upload
    public void updateBook(BookVO vo) {
        dao.updateBook(vo);
    }

    // delete
    public void deleteBook(int bookId) {
        dao.deleteBook(bookId);
    }
}
```

3) 기능 구현

- BookDAO 설정
- BookController 설정

```
@Repository
public class BookDAO {

    @Autowired
    private SqlSessionTemplate mybatis;

    // create
    public void insertBook(BookVO vo) {
        mybatis.insert("book.insertBook", vo);
    }

    // read
    public BookVO selectBook(int bookId) {
        return mybatis.selectOne("book.selectBook", bookId);
    }
    public List<BookVO> selectBooks() {
        return mybatis.selectList("book.selectBooks");
    }

    // upload
    public void updateBook(BookVO vo) {
        mybatis.update("book.updateBook", vo);
    }

    // delete
    public void deleteBook(int bookId) {
        mybatis.delete("book.deleteBook", bookId);
    }
}
```

```
BookController.java X
13
14 @Controller
15 public class BookController {
16
17     @Autowired
18     private BookService service;
19
20     // list
21
22     @GetMapping("/book/list")
23     public String list(Model model) {
24         List<BookVO> books = service.selectBooks();
25         model.addAttribute("books", books);
26         return "/book/list";
27     }
28
29     // register
30
31     @GetMapping("/book/register")
32     public String register(Model model) {
33         BookVO book = new BookVO();
34         model.addAttribute("book", book);
35         return "/book/register";
36     }
37
38     @PostMapping("/book/register")
39     public String register(BookVO vo) {
40         service.insertBook(vo);
41         return "redirect:/book/list";
42     }
43
44     // modify
45
46     @GetMapping("/book/modify")
47     public String modify(int bookId, Model model) {
48         BookVO book = service.selectBook(bookId);
49         model.addAttribute("book", book);
50         return "/book/modify";
51     }
52
53     @PostMapping("/book/modify")
54     public String modify(BookVO vo) {
55         service.updateBook(vo);
56         return "redirect:/book/list";
57     }
58
59     // delete
60
61     @GetMapping("/book/delete")
62     public String delete(int bookId) {
63         service.deleteBook(bookId);
64         return "redirect:/book/list";
65     }
66 }
```


3) 기능 구현

- CustomerService 설정
- CustomerDAO 설정

```
@Service
public class CustomerService {
```

```
    @Autowired
    private CustomerDAO dao;
```

```
    // service
```

```
    // create
```

```
    public void insertCustomer(CustomerVO vo) {
        dao.insertCustomer(vo);
    }
```

```
    // read
```

```
    public CustomerVO selectCustomer(int custId) {
        return dao.selectCustomer(custId);
    }
```

```
    public List<CustomerVO> selectCustomers() {
        return dao.selectCustomers();
    }
```

```
    // update
```

```
    public void updateCustomer(CustomerVO vo) {
        dao.updateCustomer(vo);
    }
```

```
    // delete
```

```
    public void deleteCustomer(int custId) {
        dao.deleteCustomer(custId);
    }
}
```

```
@Repository
public class CustomerDAO {
```

```
    @Autowired
```

```
    private SqlSessionTemplate mybatis;
```

```
    // create
```

```
    public void insertCustomer(CustomerVO vo) {
        mybatis.insert("customer.insertCustomer", vo);
    }
```

```
    // read
```

```
    public CustomerVO selectCustomer(int custId) {
        return mybatis.selectOne("customer.selectCustomer", custId);
    }
```

```
    public List<CustomerVO> selectCustomers() {
        return mybatis.selectList("customer.selectCustomers");
    }
```

```
    // update
```

```
    public void updateCustomer(CustomerVO vo) {
        mybatis.update("customer.updateCustomer", vo);
    }
```

```
    // delete
```

```
    public void deleteCustomer(int custId) {
        mybatis.delete("customer.deleteCustomer", custId);
    }
}
```

3) 기능 구현

- CustomerController 설정
- Main페이지 설정

```
CustomerController.java X // modify
14 @Controller
15 public class CustomerController {
16
17     @Autowired
18     private CustomerService service;
19
20     // list
21
22     @GetMapping("/customer/list")
23     public String list(Model model) {
24         List<CustomerVO> customers = service.selectCustomers();
25         model.addAttribute("customers", customers);
26         return "/customer/list";
27     }
28
29     // register
30
31     @GetMapping("/customer/register")
32     public String register(Model model) {
33         CustomerVO customer = new CustomerVO();
34         model.addAttribute("customer", customer);
35         return "/customer/register";
36     }
37
38     @PostMapping("/customer/register")
39     public String register(CustomerVO vo) {
40         service.insertCustomer(vo);
41         return "redirect:/customer/list";
42     }
43
44     @GetMapping("/customer/modify")
45     public String modify(int custId, Model model) {
46         CustomerVO customer = service.selectCustomer(custId);
47         model.addAttribute("customer", customer);
48         return "/customer/modify";
49     }
50
51     @PostMapping("/customer/modify")
52     public String modify(CustomerVO vo) {
53         service.updateCustomer(vo);
54         return "redirect:/customer/list";
55     }
56
57     // delete
58     @GetMapping("/customer/delete")
59     public String delete(int custId) {
60         service.deleteCustomer(custId);
61         return "redirect:/customer/list";
62     }
63 }

@Controller
public class MainController {

    @GetMapping(value = {"/", "/index"})
    public String index() {
        return "/index";
    }
}
```

3) 기능 구현

- book/list
- book/modify

```
<body>
  <h1>도서목록</h1>

  <a th:href="@{../index}">처음으로</a>
  <a th:href="@{../register}">도서등록</a>

  <table border="1">
    <tr>
      <th>도서번호</th>
      <th>도서명</th>
      <th>출판사</th>
      <th>가격</th>
      <th>관리</th>
    </tr>
    <tr th:each="book:${books}">
      <td th:text="${book.bookId}">1</td>
      <td th:text="${book.bookName}">축구역사</td>
      <td th:text="${book.publisher}">굿스포츠</td>
      <td th:text="${book.price}">7,000</td>
      <td>
        <a th:href="@{../modify(bookId=${book.bookId})}">수정</a>
        <a th:href="@{../delete(bookId=${book.bookId})}">삭제</a>
      </td>
    </tr>
  </table>
</body>
```

```
<form th:action="@{../modify}" th:object="${book}" method="post">
  <table border="1">
    <tr>
      <td>도서번호</td>
      <td>
        <input type="number" th:field="*{bookId}" readonly="readonly">
      </td>
    </tr>
    <tr>
      <td>도서명</td>
      <td><input type="text" th:field="*{bookName}"></td>
    </tr>
    <tr>
      <td>출판사</td>
      <td><input type="text" th:field="*{publisher}"></td>
    </tr>
    <tr>
      <td>가격</td>
      <td><input type="number" th:field="*{price}"></td>
    </tr>
    <tr>
      <td colspan="2" align="right"><input type="submit" value="수정"></td>
    </tr>
  </table>
</form>
```

3) 기능 구현

- book/register
- customer/list

```
<form th:action="@{/book/register}" th:object="${book}" method="post">
  <table border="1">
    <tr>
      <td>도서명</td>
      <td><input type="text" th:field="*{bookName}"></td>
    </tr>
    <tr>
      <td>출판사</td>
      <td><input type="text" th:field="*{publisher}"></td>
    </tr>
    <tr>
      <td>가격</td>
      <td><input type="number" th:field="*{price}"></td>
    </tr>
    <tr>
      <td colspan="2" align="right"><input type="submit" value="등록"></td>
    </tr>
  </table>
</form>
```

```
<h1>고객목록</h1>

<a th:href="@{../index}">처음으로</a>
<a th:href="@{../register}">고객등록</a>

<table border="1">
  <tr>
    <th>고객번호</th>
    <th>고객명</th>
    <th>주소</th>
    <th>휴대폰</th>
    <th>관리</th>
  </tr>
  <tr th:each="customer:${customers}">
    <td th:text="${customer.custId}">1</td>
    <td th:text="${customer.name}">박지성</td>
    <td th:text="${customer.address}">영국 맨체스터</td>
    <td th:text="${customer.phone}">010-1234-1001</td>
    <td>
      <a th:href="@{../modify(custId=${customer.custId})}">수정</a>
      <a th:href="@{../delete(custId=${customer.custId})}">삭제</a>
    </td>
  </tr>
</table>
```

3) 기능 구현

- customer/modify
- customer/register

```
<form th:action="@{./modify}" th:object="${customer}" method="post">
  <table border="1">
    <tr>
      <td>고객번호</td>
      <td>
        <input type="text" th:field="*{custId}" readonly="readonly">
      </td>
    </tr>
    <tr>
      <td>고객명</td>
      <td><input type="text" th:field="*{name}"></td>
    </tr>
    <tr>
      <td>주소</td>
      <td><input type="text" th:field="*{address}"></td>
    </tr>
    <tr>
      <td>휴대폰</td>
      <td><input type="text" th:field="*{phone}"></td>
    </tr>
    <tr>
      <td colspan="2" align="right"><input type="submit" value="수정"></td>
    </tr>
  </table>
</form>
```

```
<form th:action="@{./register}" th:object="${customer}" method="post">
  <table border="1">
    <tr>
      <td>고객명</td>
      <td><input type="text" th:field="*{name}"></td>
    </tr>
    <tr>
      <td>주소</td>
      <td><input type="text" th:field="*{address}"></td>
    </tr>
    <tr>
      <td>휴대폰</td>
      <td><input type="text" th:field="*{phone}"></td>
    </tr>
    <tr>
      <td colspan="2" align="right"><input type="submit" value="등록"></td>
    </tr>
  </table>
</form>
```

4) 실행

도서목록

처음으로 도서등록

도서번호	도서명	출판사	가격	관리
1	축구의 역사	굿스포츠	7000	수정 삭제
2	축구하는 여자	나무수	13000	수정 삭제
3	축구의 이해	대한미디어	22000	수정 삭제
4	골프 바이블	대한미디어	35000	수정 삭제
5	피겨 교본	굿스포츠	8000	수정 삭제
6	역도 단계별기술	굿스포츠	6000	수정 삭제
7	야구의 추억	이상미디어	20000	수정 삭제
8	야구를 부탁해	이상미디어	13000	수정 삭제
9	올림픽 이야기	삼성당	7500	수정 삭제
10	Olympic Champions	Pearson	13000	수정 삭제
11	스포츠의학	한솔의학서적	0	수정 삭제
13	도서이름2	출판사2	20000	수정 삭제



도서등록

처음으로 도서등록

도서명	도서이름1
출판사	출판사1
가격	10000
<input type="button" value="등록"/>	



도서수정

처음으로 도서등록

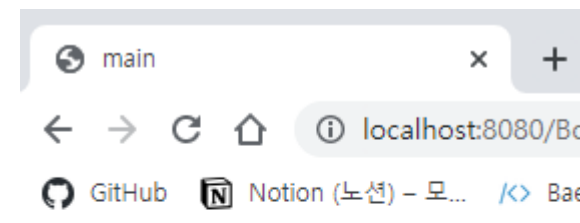
도서번호	15
도서명	도서이름3
출판사	출판사3
가격	15000
<input type="button" value="수정"/>	



11	스포츠의학	한솔의학서적	0	수정 삭제
13	도서이름2	출판사2	20000	수정 삭제
15	도서이름3	출판사3	15000	수정 삭제
10	Olympic Champions	Pearson	13000	수정 삭제
11	스포츠의학	한솔의학서적	0	수정 삭제
15	도서이름3	출판사3	15000	수정 삭제



- 메인 페이지
- 도서 목록, 등록, 수정, 삭제 기능 동작 화면



BookStore

[도서목록](#) [고객목록](#)

4) 실행

- 고객 목록, 등록, 수정, 삭제 기능 동작 화면

고객목록

처음으로 고객등록

고객번호	고객명	주소	휴대폰	관리
1	박지성	영국 맨체스타	000-5000-0001	수정 삭제
2	김연아	대한민국 서울	000-6000-0001	수정 삭제
3	장미란	대한민국 강원도	000-7000-0001	수정 삭제
4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
8	사람이름2	마산	010-2234-5678	수정 삭제

고객등록

처음으로 고객목록

고객명	사람이름1
주소	창원시
휴대폰	010-1234-5678
<input type="button" value="등록"/>	

4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
8	사람이름2	마산	010-2234-5678	수정 삭제
11	사람이름1	창원시	010-1234-5678	수정 삭제

고객수정

처음으로 고객목록

고객번호	11
고객명	사람이름3
주소	부산시
휴대폰	010-4321-8765
<input type="button" value="수정"/>	

4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
8	사람이름2	마산	010-2234-5678	수정 삭제
11	사람이름3	부산시	010-4321-8765	수정 삭제

3	장미란	대한민국 강원도	000-7000-0001	수정 삭제
4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
11	사람이름3	부산시	010-4321-8765	수정 삭제