Mybatis

프로젝트명 : B07bMybatisParameter

준비사항

- 기존 프로젝트인 B07aMybatis를 복사합니다.
- 프로젝트명 ⇒ 우클릭 ⇒ Properties ⇒ Web Project Settings 에서 Context root를 프로젝트명으로 변경합니다.
- settings.gradle 에서 rootProject.name 을 프로젝트명으로 변경합니다.
- Refresh Gradle Project 를 눌러 적용합니다.

기존 프로젝트의 구성을 그대로 유지한채 파라미터 부분만 수정해봅니다.

회원등록하기

컨트롤러 : com.edu.springboot.MainController.java

```
35      //회원등록
36      @RequestMapping(value="/regist.do", method=RequestMethod.GET)
37      public String member1() {
38          return "regist";
39      }
40      @RequestMapping(value="/regist.do", method=RequestMethod.POST)
41      public String member6(HttpServletRequest req) {
42          String id = req.getParameter("id");
43          String pass = req.getParameter("pass");
44          String name = req.getParameter("name");
45
46          int result = dao.insert(id, pass, name);
47          if(result==1) System.out.println("입력되었습니다.");
48          return "redirect:list.do";
49      }
```

인터페이스 : com.edu.springboot.jdbc.IMemberService.java

```
 9  @Mapper
10  public interface IMemberService {
11
12      public List<MemberDTO> select();
13      public int insert(String id, String pass, String name);
14      public MemberDTO selectOne(@Param("_id") String id);
15      public int update(Map<String, String> paramMap);
16      public int delete(String id);
17  }
```

매퍼 : resources/mybatis/mapper/MemberDAO.xml

```
14    <insert id="insert">
15        insert into member (id, pass, name) values
16            (#{param1}, #{param2}, #{param3})
17    </insert>
```

회원정보수정

컨트롤러 : com.edu.springboot.MainController.java

```
51    //회원수정
52    @RequestMapping(value="/edit.do", method=RequestMethod.GET)
53    public String member3(HttpServletRequest req, MemberDTO memberDTO,
54            Model model) {
55        memberDTO = dao.selectOne(req.getParameter("id"));
56        model.addAttribute("dto", memberDTO);
57        return "edit";
58    }
59    @RequestMapping(value="/edit.do", method=RequestMethod.POST)
60    public String member7(HttpServletRequest req) {
61        String id = req.getParameter("id");
62        String pass = req.getParameter("pass");
63        String name = req.getParameter("name");
64
65        Map<String, String> paramMap = new HashMap<String, String>();
66        paramMap.put("m_id", id);
67        paramMap.put("m_pass", pass);
68        paramMap.put("m_name", name);
69
70        int result = dao.update(paramMap);
71        if(result==1) System.out.println("수정되었습니다.");
72        return "redirect:list.do";
73    }
```

인터페이스 : com.edu.springboot.jdbc.IMemberService.java

```
 9 @Mapper
10 public interface IMemberService {
11
12     public List<MemberDTO> select();
13     public int insert(String id, String pass, String name);
14     public MemberDTO selectOne(@Param("_id") String id);
15     public int update(Map<String, String> paramMap);
16     public int delete(String id);
17 }
```

매퍼 : resources/mybatis/mapper/MemberDAO.xml

```
19      <select id="selectOne" resultType="com.edu.springboot.jdbc.MemberDTO">
20          select * from member where id=#{_id}
21      </select>
22
23      <update id="update" parameterType="java.util.HashMap">
24          update member set pass=#{m_pass}, name=#{m_name} where id=#{m_id}
25      </update>
26
```

여기까지 작성하세요.

회원정보삭제

컨트롤러 : com.edu.springboot.MainController.java

```
75      //회원삭제
76      @RequestMapping("/delete.do")
77      public String member4(HttpServletRequest req) {
78          String id = req.getParameter("id");
79          int result = dao.delete(id);
80          if(result==1) System.out.println("삭제되었습니다.");
81          return "redirect:list.do";
82      }
83  }
```

인터페이스 : com.edu.springboot.jdbc.IMemberService.java

```
 9  @Mapper
10  public interface IMemberService {
11
12      public List<MemberDTO> select();
13      public int insert(String id, String pass, String name);
14      public MemberDTO selectOne(@Param("_id") String id);
15      public int update(Map<String, String> paramMap);
16      public int delete(String id);
17  }
```

매퍼 : resources/mybatis/mapper/MemberDAO.xml

```
27      <delete id="delete">
28          delete from member where id=#{0}
29      </delete>
30
31  </mapper>
32
```

여기까지 작성하세요.

Mapper에서 제어문 사용하기

View : webapp/WEB-INF/views/list.jsp

검색폼 추가

```jsp
<body>
    <h2>회원리스트</h2>
    <form>
    <table>
    <tr>
        <td>
            <input type="checkbox" name="searchField" value="id" />아이디
            <input type="checkbox" name="searchField" value="name" />이름
            <input type="checkbox" name="searchField" value="pass" />패스워드
            <input type="text" name="searchKeyword" />
            <input type="submit" value="검색" />
        </td>
    </tr>
    </table>
    </form>
    <table border="1">
        <tr>
```

DTO : com.edu.springboot.jdbc.MemberDTO.java

DTO에 멤버변수 추가

```java
package com.edu.springboot.jdbc;

import java.util.List;

import lombok.Data;

@Data
public class MemberDTO {
    private String id;
    private String pass;
    private String name;
    private String regidate;

    private List<String> searchField;
    private String searchKeyword;
}
```

Controller : com.edu.springboot.MainController.java

컨트롤러에 매개변수 추가

```java
    //회원목록
    @RequestMapping("/list.do")
    public String member2(Model model, MemberDTO memberDTO) {

        System.out.println(memberDTO.getSearchField());
        System.out.println(memberDTO.getSearchKeyword());

        model.addAttribute("memberList", dao.select(memberDTO));
        return "list";
    }
```

Interface : com.edu.springboot.jdbc.IMemberService.java

인터페이스의 추상메서드 수정

```java
package com.edu.springboot.jdbc;

import java.util.List;

@Mapper
public interface IMemberService {

    //public List<MemberDTO> select();
    public List<MemberDTO> select(MemberDTO memberDTO);

    //등록처리 : request 내장객체를 통해 받은 파라미터를 전달한다.
    public int insert(String id, String pass, String name);
```

Mapper : resources/mybatis/mapper/MemberDAO.xml

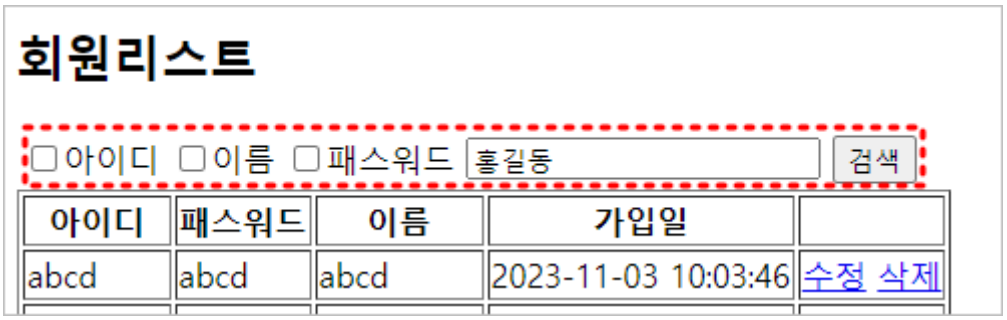
매퍼에 제어문 추가

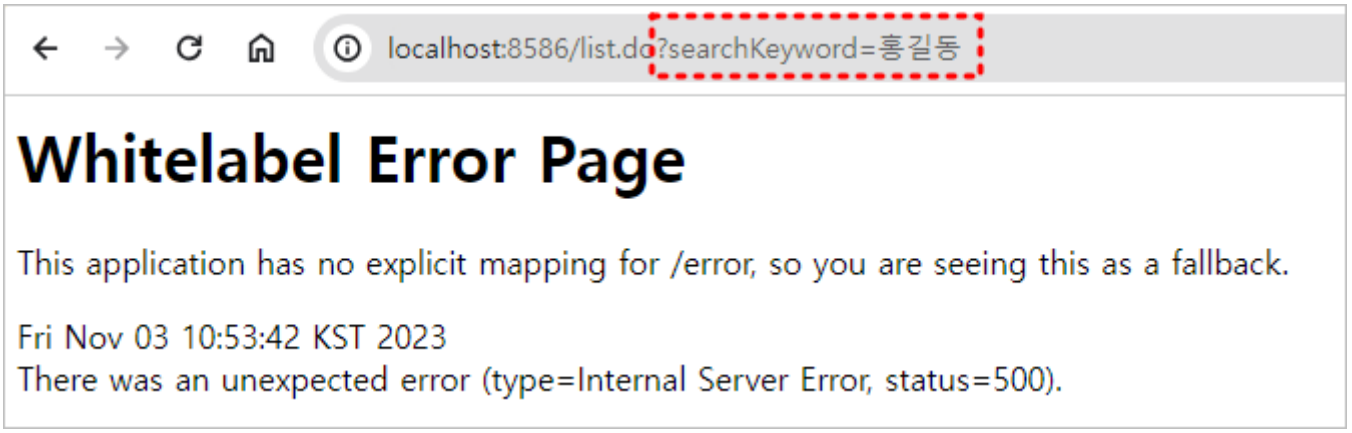
```xml
<mapper namespace="com.edu.springboot.jdbc.IMemberService">

    <!-- <select id="select" resultType="com.edu.springboot.jdbc.MemberDTO">
        SELECT * FROM member ORDER BY regidate DESC
    </select> -->
    <select id="select" parameterType="com.edu.springboot.jdbc.MemberDTO"
            resultType="com.edu.springboot.jdbc.MemberDTO">
        select * from member
        <if test="searchKeyword!=null and !searchKeyword.equals('')" >
            where
            <foreach collection="searchField" item="sfield" open="(" close=")"
                separator="or">
                ${sfield} like '%'||#{searchKeyword}||'%'
            </foreach>
        </if>
        order by regidate desc
    </select>
```

여기까지 작성하세요.

퀴즈]

체크항목이 선택되지 않은 상태에서 검색어를 입력 후 '검색' 버튼을 누르면 에러가 발생한다.





이 경우 Backend에서 체크항목에 대한 검증을 하는것보다는 Frontend에서 진행 후 경고창을 띄워주는것이 합리적이다.

따라서 하나 이상의 항목을 체크해야 검색할 수 있도록 Javascript를 추가하시오.