9일차(Mybatis-글쓰기,글상세보기,글수정,글삭제,글조회하기구현)

SqlMapConfig.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
 "HTTP://mybatis.org/dtd/mybatis-3-config.dtd">

 <configuration>
    <typeAliases>
        <typeAlias type = "lee.BoardCommand" alias="board" />
    </typeAliases>

    <mappers>
        <mapper resource="lee/Board.xml" />
    </mappers>
 </configuration>
```

Board.xml

```
<select id="list" resultType="board">
    select * from springboard order by num desc
</select>
```

- 위 SqlMapConfig.xml의 board와 같다

---

1. 글쓰기 구현

1. BoardDAO 인터페이스에서 공통 추상 메서드 선언

```
public interface BoardDAO {

    //1. 글목록 보기
    public List list() throws DataAccessException; //스프링 예외처리 클래스

    //2-1) 최대값 구하기-> select max(num) from springboard
    public int getNewNum() throws DataAccessException;

    //2-2. 글쓰기
    public void write(BoardCommand data) throws DataAccessException;

}
```

2 . SqlMapBoardDao 클래스 -> 메서드 작성

- BoardDAO 인터페이스를 상속받음
  BoardDAO.java 메서드 작성 시 SqlMapBoardDao도 메서드 작성해야한다
- 형식) sqlSession객체명.selectOne(실행시킬 sql구분의 id값, 매개변수명)
- 형식) sqlSession객체명.selectList(실행시킬 sql구문의 id값, 매개변수명)~ 레코드 여러개

```
public class SqlMapBoardDao extends SqlSessionDaoSupport implements BoardDAO {

    @Override
    public List list() throws DataAccessException {
        return getSqlSession().selectList("list");
    }

    @Override
    public int getNewNum() throws DataAccessException {
        return (Integer)getSqlSession().selectOne("getNewNum");
    }

    @Override
    public void write(BoardCommand data) throws DataAccessException {
        getSqlSession().insert("write", data);
    }

}
```

3 . Board.xml

- Board.xml의 기본적인 속성

```
<insert>, <update>, <delete>, <select>
-----------------------------------------
1.id(필수)->sql문장에 종류와 상관없이 필수로 작성(구분인자값)
              ->관례(메서드명)->list()

2.parameterType(선택)->SQL구문중에서 매개변수를 입력을 받아서
                                    처리해주는 경우의 SQL구문에 필요
                                  ->where조건식
                                  ->입력을 받는 자료형을 쓰게 되어있다.
                      parameterType="java.lang.String" or"String"
                                                "java.lang.Integer" or "in
t"

 select * from springboard where num=#{매개변수명}  ?대신에
 insert into springboard values(#{매개변수},,,,)

3.resultType(선택)->sql구문을 사용해서 반환값이 있는 경우
 대부분->select num from springboard (insert,update,delete  X)

  resultType="java.lang.Integer" or "int"
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="board">

<!-- springboard테이블에서 sql작업 (1.글목록보기) -->

<select id="list" resultType="board">
    select * from springboard order by num desc
</select>

 <!-- 2.최대값 구하기 -->
 <select id="getNewnum" resultType="int">
    select max(num) from springboard
 </select>

 <!-- 3.글쓰기 -->
 <insert id="write" parameterType="board">
    insert into springboard(num, author, title, content, writeday)
    values(#{num}, #{author}, #{title}, #{content}, now() )
 </insert>
 </mapper>
```

4. 각각의 ActionController가 필요

WriteActionController.java

- 1.request(요청객체) 2.response(응답)
  3.입력받은값을 저장한 객체
  4.BindException->사용자로부터 값을 입력시 에러발생->처리해주는 객체
- 변경 전) response.sendRedirect("list.jsp");
  변경 후) redirect:/요청명령어

```
//사용자로부터 값을 주로 입력을 받아서 처리해주는 컨트럴러
public class WriteActionController extends AbstractCommandController {

    BoardDAO dao; //BoardDAO dao=new BoardDAO();
    //commandClass->사용자로부터 입력받은 값만 따로 저장해주는 메서드
    //<property name="commandClass"  value="lee.BoardCommand" />

    public void setDao(BoardDAO dao) {
        this.dao = dao;
        System.out.println("setDao()호출됨(dao)=>"+dao);
    }

    @Override
    protected ModelAndView handle(HttpServletRequest request, HttpServletRespon
se response, Object command, BindException error) throws Exception {

        request.setCharacterEncoding("utf-8");
        BoardCommand data=(BoardCommand)command;

        int newNum = dao.getNewNum()+1;
        data.setNum(newNum);

        dao.write(data);//dao.write(data);

        return new ModelAndView("redirect:/list.do");
    }
}
```

글 검색하기

1 . BoardDAO

```
//6. 글 검색하기(String searchName, String searchValue)->hasMap처리
    public List search(BoardCommand data)throws DataAccessException;
```

2. Board.xmd

```
<!-- 8. 글 조회하기(글검색분야, 글검색어) -->
<select id="search" resultType="board" parameterType="board">
 select * from springboard where #{searchName} = like '%#{searchValue}%'
 order by num desc
</select>
```

3. SqlMapBoardDao.java

```
@Override
public List search(BoardCommand data) throws DataAccessException {
  // TODO Auto-generated method stub
  return getSqlSession().selectList("search",data);
}
```

4. SearchActionController.java

```
//Controller를 상속받는 이유->요청을 받아서 처리해주는 역할
public class SearchActionController implements Controller {

    BoardDAO dao; //BoardDAO dao=new BoardDAO();

    public void setDao(BoardDAO dao) {
        this.dao = dao;
        System.out.println("setDao()호출됨(dao)=>"+dao);
    }

    @Override
    public ModelAndView handleRequest(HttpServletRequest request,
                                      HttpServletResponse response) thro
ws Exception {
        // TODO Auto-generated method stub
        System.out.println("SearchActionController 실행됨!");
        String searchName=request.getParameter("searchName");
        String searchValue=request.getParameter("searchValue");

        //ArrayList list=dao.search(searchName,searchValue);
        BoardCommand data = new BoardCommand();
        data.setSearchName(searchName);
        data.setSearchValue(searchValue);
        List list = dao.search(data);


        ModelAndView mav=new ModelAndView();
        mav.setViewName("list");//list.jsp
        //request.setAttribute("list",list);
        mav.addObject("list",list); //${list}
        return mav;
    }
}
```

매개변수 잘 못 받은 에러

```
Exception

org.springframework.web.util.NestedServletException: Request processing faile
d; nested exception is org.springframework.jdbc.InvalidResultSetAccessExceptio
n:
### Error querying database.  Cause: java.sql.SQLException: 부적합한 열 인덱스
### The error may exist in lee/Board.xml
### The error may involve board.search-Inline
### The error occurred while setting parameters
### SQL: select * from springboard where ? like '%?%'    order by num desc
### Cause: java.sql.SQLException: 부적합한 열 인덱스
; invalid ResultSet access for SQL []; nested exception is java.sql.SQLExceptio
n: 부적합한 열 인덱스
    org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkSe
rvlet.java:659)
    org.springframework.web.servlet.FrameworkServlet.doGet(FrameworkServlet.jav
a:552)
    javax.servlet.http.HttpServlet.service(HttpServlet.java:635)
    javax.servlet.http.HttpServlet.service(HttpServlet.java:742)
    org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
Root Cause

org.springframework.jdbc.InvalidResultSetAccessException:
### Error querying database.  Cause: java.sql.SQLException: 부적합한 열 인덱스
### The error may exist in lee/Board.xml
### The error may involve board.search-Inline
### The error occurred while setting parameters
   =>파라미터값을 세팅하는 동안에 에러가 발생
### SQL: select * from springboard where ? like '%?%'    order by num desc
### Cause: java.sql.SQLException: 부적합한 열 인덱스
; invalid ResultSet access for SQL []; nested exception is java.sql.SQLExceptio
n: 부적합한 열 인덱스
    org.springframework.jdbc.support.SQLErrorCodeSQLExceptionTranslator.doTrans
late(SQLErrorCodeSQLExceptionTranslator.java:237)
```

Mybatis에서 매개변수를 받을 경우 주의할 점 ★★★

- #{매개변수}

1. 값을 입력(글쓰기, 글수정) => #{매개변수}(=멤버변수)
매개변수가 순수 값을 입력받는 값이 아닌 필드명으로 값을 입력받거나 특수기호가 포함되는 값을 입력 받은 경우 ->
#{매개변수} 사용 불가

2. 매개변수 받는 위치 => 필드명, 특수기호가 포함되는 경우에는
#{매개변수} => **${매개변수}**

Board.xml

- like연산자의 **%%** 쓸 경우

```
<!-- 8. 글 조회하기(글검색분야, 글검색어) -->
<select id="search" resultType="board" parameterType="board">
 select * from springboard where #{searchName} = like '%${searchValue}%'
 order by num desc
</select>
```

SpringAnno 프로젝트

index.jsp

```
<%
 // response.sendRedirect("http://localhost:8090/SpringMybatis/list.do");
  response.sendRedirect(request.getContextPath()+"/list.do");
%>
```

board-servlet.xml

변경 전

```
<!--1.글목록보기  -->
<bean name="/list.do"  class="lee.ListActionController">
    <property name="dao">
         <ref bean="boardDAO" />
    </property>
</bean>
```

1. 변경 후

```
<bean name="/list.do" class="lee.ListActionController" />
```

```
property태그를 생략->Setter Method설정X ->Setter Method호출
                                  sqlSessionFactoryBean 객체생성X->NullPointerExce
ption

java.lang.NullPointerException
 lee.ListActionController.handleRequest(ListActionController.java:28)
```

---

2. @Autowired, @Required 이그노얼 사용 하기 위해 추가

```
<!-- @Autowired,@Required  -->
<bean class="org.springframework.beans.factory.annotation.RequiredAnnotationBeanPostProcessor" />
<bean class="org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor" />
```

3. ListActionController.java

- 클래스를 POJO클래스로 변경해야한다
- 컨트롤러 역할을 하기 위해서는 Controller 인터페이스를 상속받거나 AbstractCommandController를 상속받아야 한다
  why? 요청을 받아서 처리해주는 메서드를 상속받기 위해서
  But! POJO클래스는
  1. **인터페이스 or 추상클래스를 상속받지 않아도 된다**
  => 독립적으로 요청을 받아서 처리해줄 수 있다.
  2.@Controller를 부여받게되면 자기자신이 @Controller 가 된다.
  3.요청을 받아서 처리해주는 메서드를 내마음대로 변경이 가능하다.(문법 제약에서 벗어난다)

ListActionController.java 변경 전

```java
package lee;

import java.util.ArrayList;
import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.mvc.Controller;

//Controller를 상속받는 이유->요청을 받아서 처리해주는 역할
public class ListActionController implements Controller {

    BoardDAO dao; //BoardDAO dao=new BoardDAO();

    public void setDao(BoardDAO dao) {
        this.dao = dao;
        System.out.println("setDao()호출됨(dao)=>"+dao);
    }

    @Override
    public ModelAndView handleRequest(HttpServletRequest request,
                                      HttpServletResponse response) thro
ws Exception {
        // TODO Auto-generated method stub
        System.out.println("ListActionController 실행됨!");
        //ArrayList list=dao.list();
        List list = dao.list();

        ModelAndView mav=new ModelAndView();
        mav.setViewName("list");//list.jsp
        //request.setAttribute("list",list);
        mav.addObject("list",list); //${list}
        return mav;
    }
}
```

변경 후

- @Controller :컨트롤러 역할
- @Required : 필수로 지정하게 해줌
- @Autowired : Setter, Getter 자동
- @RequestMapping("list.do") : 해당 메서드로 실행하게해주는것

```java
package lee;

import java.util.ArrayList;
import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Required;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.ModelAndView;
//import org.springframework.web.servlet.mvc.Controller;

//Controller를 상속받는 이유->요청을 받아서 처리해주는 역할

@Controller
public class ListActionController {

    BoardDAO dao; //BoardDAO dao=new BoardDAO();

    @Required
    @Autowired
    public void setDao(BoardDAO dao) {
        this.dao = dao;
        System.out.println("setDao()호출됨(dao)=>"+dao);
    }

    @RequestMapping("list.do")
    public ModelAndView handleRequest(HttpServletRequest request,
                                      HttpServletResponse response) thro
ws Exception {
        // TODO Auto-generated method stub
        System.out.println("ListActionController 실행됨!");
        //ArrayList list=dao.list();
        List list = dao.list();

        ModelAndView mav=new ModelAndView();
        mav.setViewName("list");//list.jsp
        //request.setAttribute("list",list);
        mav.addObject("list",list); //${list}
        return mav;
    }
}
```

글쓰기 어노테이션 하기

board-servlet.xml

변경 전

```
<!-- 5.글수정하기 -->
<bean name="/update.do"  class="lee.UpdateActionController">
  <property name="dao">
       <ref bean="boardDAO" />
  </property>
  <property name="commandClass"  value="lee.BoardCommand" />
</bean>
```

변경 후

```
@RequestParam("title")String title
= String title = request.getParameter("title");
```

```
package lee;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.validation.BindException;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.mvc.AbstractCommandController;

//사용자로부터 값을 주로 입력을 받아서 처리해주는 컨트럴러
//public class WriteActionController extends AbstractCommandController {
    public class WriteActionController{

    BoardDAO dao; //BoardDAO dao=new BoardDAO();
    //commandClass->사용자로부터 입력받은 값만 따로 저장해주는 메서드
    //<property name="commandClass"  value="lee.BoardCommand" />

    public void setDao(BoardDAO dao) {
        this.dao = dao;
        System.out.println("setDao()호출됨(dao)=>"+dao);
    }

    //1.request(요청객체) 2.response(응답) 3.입력받은값을 저장한 객체
    //4.BindException->사용자로부터 값을 입력시 에러발생->처리해주는 객체

    @RequestMapping("/write.do")
    protected ModelAndView test(@RequestParam("title")String title, @RequestPar
am("author")String author, @RequestParam("content")String content) throws Excep
tion {

/*      request.setCharacterEncoding("utf-8");
        BoardCommand data=(BoardCommand)command;*/

        int newNum = dao.getNewNum()+1;
        BoardCommand data = new BoardCommand();
        data.setNum(newNum);
        data.setTitle(title);
        data.setAuthor(author);
        data.setContent(content);

        dao.write(data);//dao.write(data);

        return new ModelAndView("redirect:/list.do");
    }

}
```

web.xml 에서 board로 서블릿 이름 정하면

board- servlet.xml을 찾아서 실행한다