도서관리 프로그램

1. 프로젝트의 목표

최대한 오류 없이 돌아가는 기본적인 도서관리 프로그램.

도서예약 등을 추가로 구현하여 약간의 차별성.

.

1. 목적
2. 인터페이스와 세터함수, xml을 이용하여 최대한 di가 유지되는 프로그램.
3. Mysql을 이용한 데이터 관리 프로그램
4. JSP를 이용한 화면구성

->관련 데이터는 되도록 json을 활용

1. 간단한 설명
2. 기존에 구성해둔 데이터 설계를 활용하여 MYSQL에 DB를 구성한다.
3. 이때에 member, book 클래스는 각각 setter함수를 쓰며 interface 를 이용하여 di를 유지 시켜 둔다.
4. 이후 DAO안의 INSERT(정보 삽입), SELECT(해당객체 찾기), UPDATE(동기화)

함수들을 이용하여 mysql과 연동하여 sql쿼리를 이용하여 값을 가져온다..

1. 각각 member객체를 관리하는 memberDAO book을 관리하는 bookDAO,

Borrowtable과 Reservetable를 관리해주는 serviceDAO 세개로 구성한다

1. WEBCONTROLLER를 이용하여 JSP파일의 이동 및 파라미터 전달을 컨트롤 한다.
2. 구현 함수들

Book {

Long bnum;

String bname;

String publisher;

String author;

**기본적인 book의 형태이다.**

**이름, 출판사, 지은이, 예약자 목록, 현재 빌릴 수 있는 지의 상태**

**모든 인자들은 set 과 get으로 관리한다.**

**SQL관리시 접근에 용이한 BNUM을 따로 선언.**

Member {

**private** Long mnum;

**private** String id;

**private** String password;

**private** **boolean** ban =**true**;

**역시 멤버의 기본형태로**

**Id, 비밀번호, 빌린 책 목록, 현재 빌린 책의 개수, 제제상태 를 보인다**

**Get,set함수를 이용하여 인자를 바꾼다.**

**이곳에도 MNUM을 따로 두어 SQL 이용시에 활용한다.**

1. DAO (MEMBER , BOOK)

**this**.jdbcTemplate = **new** JdbcTemplate(dataSource);

우선 JDBC에 기존에 설정해둔 DATASOURCE를 이용하여 연결해준다.

1. **public** List<Book> selectAll() {  
   처음으로 구현한 내부함수는 SELECTALL이다. 이는

select \* from Book

을 이용하여 SQL로 모든 BOOK(MEMBER)값을 읽어와 LIST형식으로 저장한다.

1. selectByBname(String Bname) {

두번째로는 책의 이름을 이용하여 해당 책의 객체를 찾는 함수이다.

select \* from Book where Bname = ?

를 사용하여 책(멤버) 테이블에서 원하는 이름을 가진 객체를 찾아낸다.

1. insert(Book book)

책의 객체를 입력으로 받아서 DB에 넣어주는 함수이다.

insert into Book (Bname, Author,Publisher) " + " values (?, ?, ?)

이와 같은 SQL문을 이용하여 넣어준다.

1. update(Book book)

위에서 선언한 객체의 정보를 바꾸는데사용한다

update Member set Bname = ?, Author = ?, Publisher = ? " + " where Bname = ?

1. ServiceDAO)
2. **this**.jdbcTemplate = **new** JdbcTemplate(dataSource);

서비스 DAO 역시 DB에 연결해준다.

1. Binsert(Long Mnum, Long Bnum)
2. Rinsert(Long Mnum, Long Bnum)
3. Bdelete(Long Mnum, Long Bnum)
4. Rdelete(Long Mnum, Long Bnum)

책을 빌리거나 반납,예약할 때에 해당 정보들을 넣거나 삭제하는 함수들이다.

1. List<Book> BselectByMnum2(Long Mnum)

나머지 함수들은 위 DAO들과 대다수 비슷하다.

그러나 SQL사용시 BNUM, MNUM과 같은 외래키를 이용하여

BORROWTABLE이나 RESERVETABLE에 접근을 해야 때가 있는데

select \* from book where Bnum IN (select bt2.Bnum from BorrowTable as bt2 where Mnum = ?

이처럼 Mnum을 이용하여 borrowtable에서 값을 찾아와 book테이블에서 객체를 찾는 행위를 해줄 수 도 있다

1. addBook(String Bname,String Publisher, String Author)

insert(book)

책 객체를 DB에 추가하는 함수.

책이름 출판사 작가를 입력으로 받아 추가한다.

addMember(String ID,String Password)

역시 멤버의 아이디와 비밀번호를 입력받아 추가한다.

1. borrowBook(String ID, String Password, String Bname)

책을 빌리는 함수로

serviceDAO.BcountByMnum(member.getMnum()) < 3

빌린 책이 3권이 넘었는지 체크,

serviceDAO.BcountByBnum(book.getBnum()) == 0

해당 책이 대출중인지 체크,

serviceDAO.RcountByBnum(book.getBnum()) > 0

예약자가 있는지 체크,

serviceDAO.RselectByBnum(book.getBnum()).get(0).getMnum().equals(member.getMnum())

예약자의 첫번째가 고객인지 확인후 책의 대출 여부를 결정한다.

1. findBook(String bookname)
2. findMember(String ID)

책의 이름과 고객의 아이디를 가지고 DB에서 객체를 가져온다.

1. reserveBook(String ID, String Bname)

예약서비스는 따로 체크는

serviceDAO.BcountByBnum(book.getBnum())>0

대출중인 여부만 확인 후,

Rinsert(member.getMnum(), book.getBnum())

INSERT 함수를 이용하여 RESERVETABLE에 추가하여 준다.

1. returnBook(String ID, String Bname)

현재 대출중인지를 확인

serviceDAO.BRcount(member.getMnum(), book.getBnum()) > 0

이후 반납을 한다.

serviceDAO.Bdelete(member.getMnum(), book.getBnum())

1. bookDAO.selectAll();
2. memberDAO.selectAll();

각각 모든 객체를 보여주는 함수이다.

실행 조건 체크

1. 대출 이후 다른 고객이 예약을 신청 O
2. 대출 이후 다른 고객이 같은 책을 다시 대출 X
3. 예약자 순번에 따른 대출 가능여부 x
4. 예약자가 두 명 이상일 때 순서에 따른 대출여부
5. 대출 권수가 3권이 넘을 때. 대출여부

해당 사례들로 오류들을 체크한다.

또한 결과값이 잘못 입력 될 경우 null 값을 주어 빈페이지를 보이게 한다.