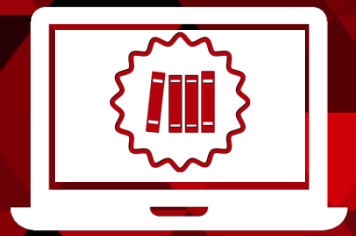
BIT CAMP 104



자바를 이용한 전자 도서관 시스템

2조 강진광 김준수 김태웅 박주원 아윤근 이아림

INDEX

1 시나리오

2 요구명세서

3 Class 다이어그램

4 유스케이스 다이어그램

5 유스 케이스 명세서

6 구현 & 사용기술

시나리오

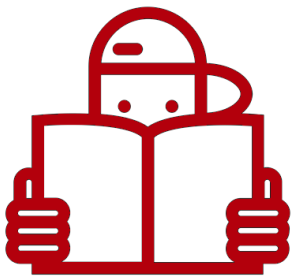
전자 도서관은 전자 책을 기반으로, 도서관은 책을 빌려주고, 사용자는 책을 빌려 받아 읽을 수 있는 시스템이다.

도서목록은 회원 비회원 구별 없이 검색이 가능하고, 인기순 검색과 책제목 검색이 있다.
관리자는 책을 추가하고 삭제가 가능하며 회원목록과 현재 대여된 도서 목록을 볼 수 있다.
로그인한 회원이라면 언제든지 전자도서관을 통해 책을 대여받아 이용할 수 있다.

책은 한 권당 한 명에게만 대여 가능하다.
대여된 책은 일정기간이 지나면 자동 반납된다.

요구명세서

User



: 도서관 이용자 요구명세서

- 이용자는 회원등록 상관없이 도서목록을 볼 수 있다.
  이용자는 회원가입 후 로그인 한다.
  로그인한 사용자는 여러 권의 도서를 대여할 수 있다.
  로그인한 사용자는 대여한 도서를 읽을 수 있다.
  로그인한 사용자는 자신이 대여한 도서 목록을 볼 수 있다.
  사용자는 전체 도서 목록을 검색할 수 있다.
  사용자는 책 제목으로 검색할 수 있고,
  인기순 도서 목록을 볼 수 있다.

요구명세서

Manager

: 도서관 담당자 요구명세서

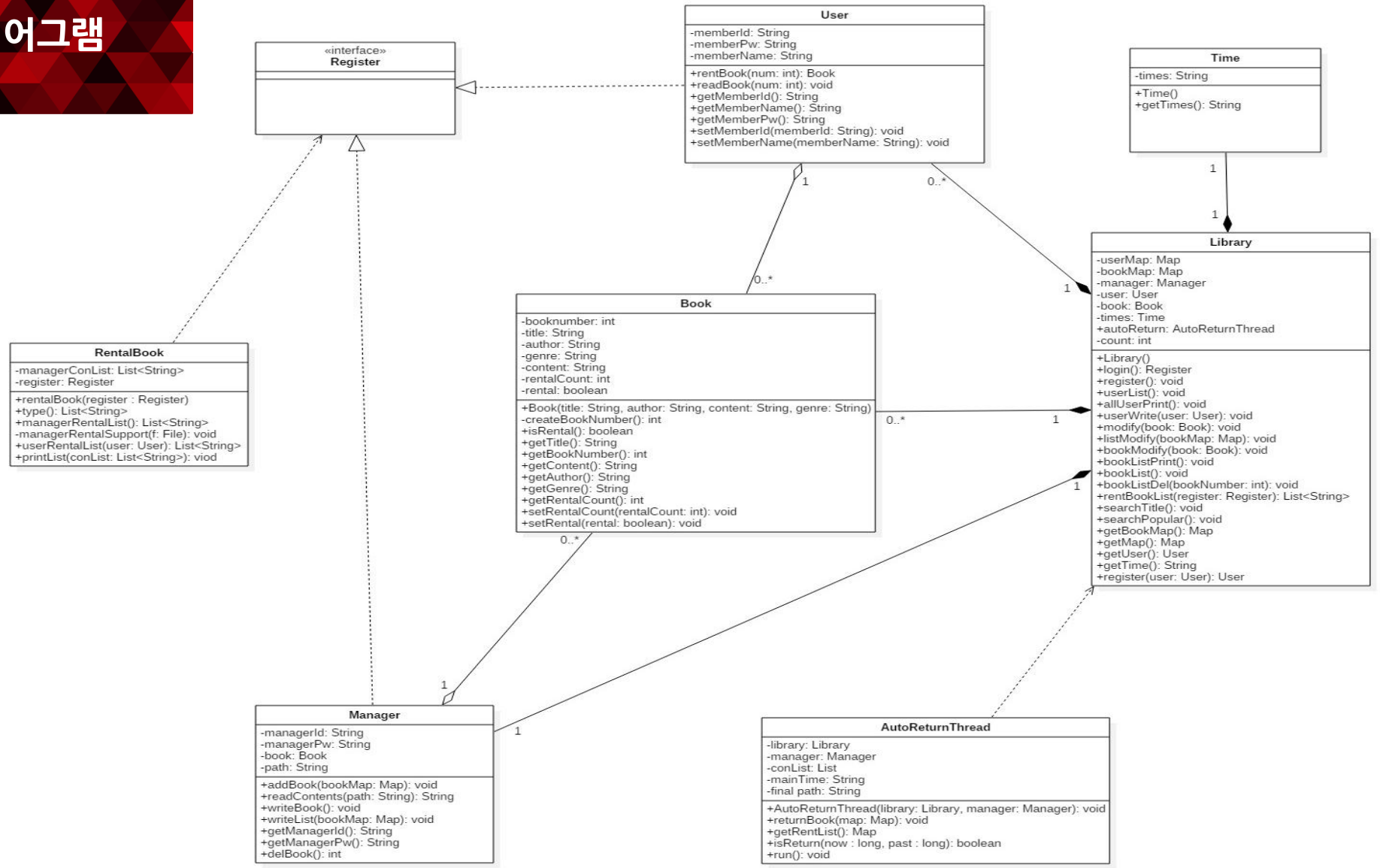
- 관리자는 새로운 도서를 추가할 수 있다.
  관리자는 기존의 도서를 삭제할 수 있다.
  관리자는 회원 목록을 볼 수 있다.
  관리자는 도서 목록을 검색할 수 있다.
  관리자는 책 제목으로 검색할 수 있고,
  인기순 도서 목록을 볼 수 있다.
  관리자는 현재 대여된 상태인 도서 목록을 볼 수 있다.

Library

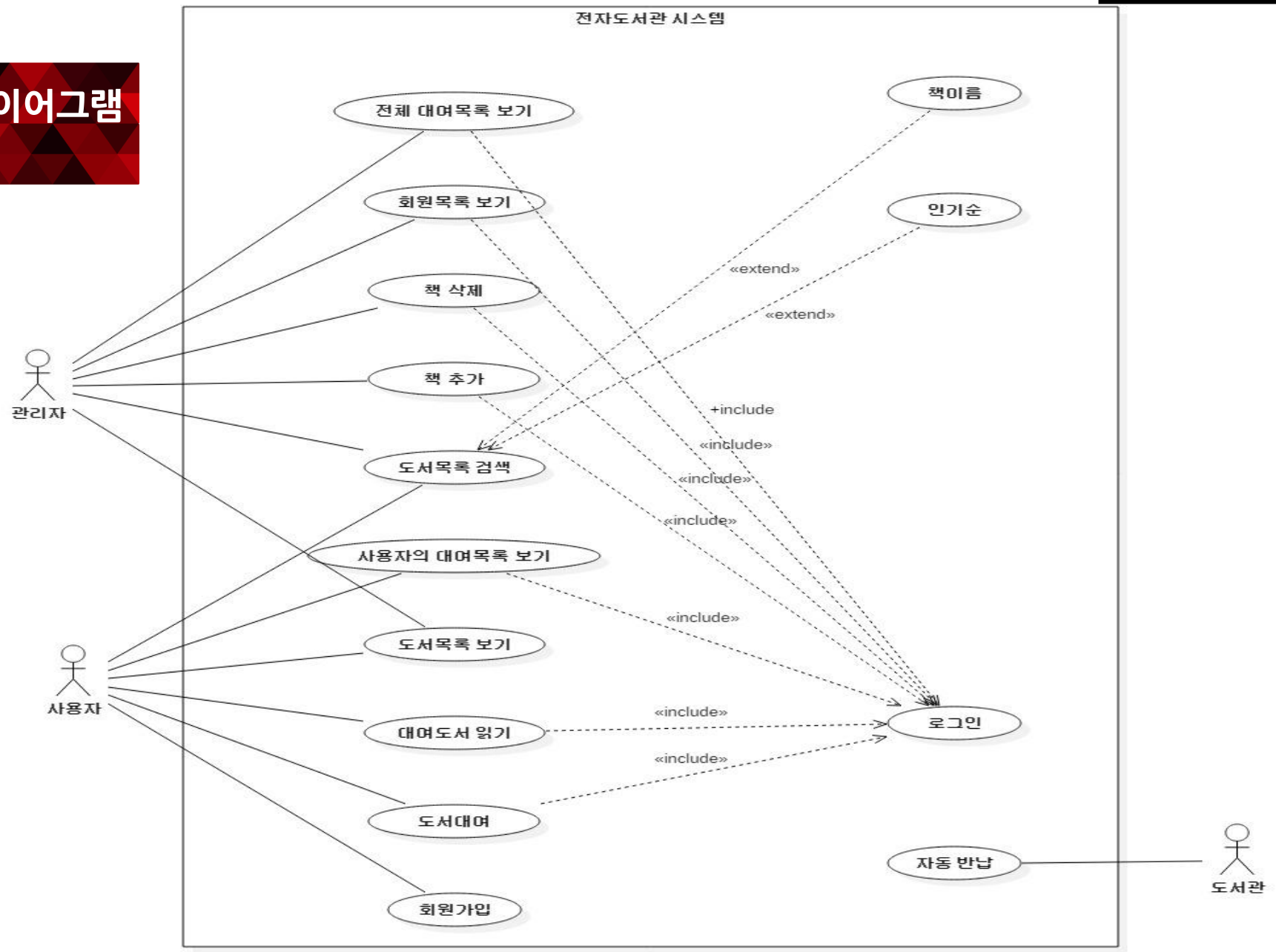
: 도서관 요구명세서

▶ 도서관은 대여한 도서는 기간이 지나면 자동 반납이 된다.

Class 다이어그램


«interface»
Register
User
-memberId: String
-memberPw: String
-memberName: String
+rentBook(num: int): Book
+readBook(num: int): void
+getMemberId(): String
+getMemberName(): String
+getMemberPw(): String
+setMemberId(memberId: String): void
+setMemberName(memberName: String): void
Time
-times: String
+Time()
+getTimes(): String
RentalBook
-managerConList: List<String>
-register: Register
+rentalBook(register : Register)
+type(): List<String>
+managerRentalList(): List<String>
-managerRentalSupport(f: File): void
+userRentalList(user: User): List<String>
+printList(conList: List<String>): viod
Book
-booknumber: int
-title: String
-author: String
-genre: String
-content: String
-rentalCount: int
-rental: boolean
+Book(title: String, author: String, content: String, genre: String)
-createBookNumber(): int
+isRental(): boolean
+getTitle(): String
+getBookNumber(): int
+getContent(): String
+getAuthor(): String
+getGenre(): String
+getRentalCount(): int
+setRentalCount(rentalCount: int): void
+setRental(rental: boolean): void
Library
-userMap: Map
-bookMap: Map
-manager: Manager
-user: User
-book: Book
-times: Time
+autoReturn: AutoReturnThread
-count: int
+Library()
+login(): Register
+register(): void
+userList(): void
+allUserPrint(): void
+userWrite(user: User): void
+modify(book: Book): void
+listModify(bookMap: Map): void
+bookModify(book: Book): void
+bookListPrint(): void
+bookList(): void
+bookListDel(bookNumber: int): void
+rentBookList(register: Register): List<String>
+searchTitle(): void
+searchPopular(): void
+getBookMap(): Map
+getMap(): Map
+getUser(): User
+getTime(): String
+register(user: User): User
Manager
-managerId: String
-managerPw: String
-book: Book
-path: String
+addBook(bookMap: Map): void
+readContents(path: String): String
+writeBook(): void
+writeList(bookMap: Map): void
+getManagerId(): String
+getManagerPw(): String
+delBook(): int
AutoReturnThread
-library: Library
-manager: Manager
-conList: List
-mainTime: String
-final path: String
+AutoReturnThread(library: Library, manager: Manager): void
+returnBook(map: Map): void
+getRentList(): Map
+isReturn(now : long, past : long): boolean
+run(): void
1
0..*

유스케이스 다이어그램


전자도서관 시스템
전체 대여목록 보기
회원목록 보기
책 삭제
책 추가
도서목록 검색
사용자의 대여목록 보기
도서목록 보기
대여도서 읽기
도서대여
회원가입
책이름
인기순
로그인
자동 반납
관리자
사용자
도서관
«extend»
«extend»
+include
«include»
«include»
«include»
«include»
«include»
«include»

유스 케이스 명세서

유스케이스명	도서 대여 및 읽기
액터명	도서관 이용자(회원)
개요	회원은 도서대여여부에 따라 대여할 수 있다
기본흐름	선행 조건) 회원은 로그인 된 상태여야 한다. 1. 회원은 책을 대여할 수 있다. 2. 회원은 대여한 책에 한하여 열람할 수 있다. (일정 기간)
대체흐름	1-1 다른 회원이 대여한 도서를 대출 신청할 경우 1. 대여 할 수 없는 도서라고 화면에 표시한다. 2-1 도서관에 존재하지 않는 도서를 검색했을 경우(제목검색) 1. 검색 결과 없다고 화면에 표시한다. 2-2 대여기간 외 열람을 원할 경우 1.자신의 폴더 안에 도서가 더 이상 없기 때문에 열람불가능

유스 케이스 명세서

유스케이스명	도서관 이용자의 대여목록 보기
액터명	도서관 이용자
개요	도서관 이용자는 자신이 대여한 도서의 목록을 볼 수 있다.
기본흐름	선행 조건) 도서관 이용자는 로그인 된 상태여야 한다. 1. 반납이 되지 않은 도서의 목록을 볼 수 있다.
대체흐름	1-1 대여한 도서가 없을 경우 1. 화면에 표시가 되지 않는다.

유스 케이스 명세서

유스케이스명	도서 추가 및 삭제
액터명	도서관 관리자
개요	관리자는 도서 추가와 삭제가 가능하다.
기본흐름	선행 조건) 관리자는 로그인 된 상태여야 한다. 1. 관리자는 도서코드,제목, 저자, 장르 항목에 맞추어 도서를 등록한다. 2. 관리자는 삭제하고자 하는 도서코드를 입력해 도서를 삭제한다.
대체흐름	1-1 도서 제목을 중복되게 등록할 경우 1. 도서 코드가 중복되지 않게 자동으로 생성되기 때문에 가능하다.

유스 케이스 명세서

유스케이스명	회원 목록 확인
액터명	도서관 관리자
개요	관리자는 회원목록을 확인할 수 있다
기본흐름	선행 조건) 관리자는 로그인 된 상태여야 한다. 1. 관리자는 회원들의 아이디, 비밀번호, 이름을 모두 확인할 수 있다.

유스 케이스 명세서

유스케이스명	전체 대여도서 목록확인
액터명	도서관 관리자
개요	관리자는 이용자가 대여한 도서의 전체 목록을 볼 수 있다.
기본흐름	선행 조건) 관리자는 로그인 된 상태여야 한다. 1. 관리자는 모든 이용자가 대여한 도서의 전체 목록을 본다.
대체흐름	1-1 이용자가 대여한 도서가 없을 경우 1. 화면에 표시가 되지 않는다.

유스 케이스 명세서

유스케이스명	도서목록 검색 및 보기
액터명	도서관 관리자 , 이용자
개요	관리자와 이용자는 도서 목록을 볼 수 있다. 관리자와 이용자는 도서 목록을 이름으로 검색하여 볼 수 있다. 관리자와 이용자는 도서 목록을 대여횟수 순으로 검색하여 볼 수 있다.
기본흐름	선행 조건) 프로그램이 실행 되어야 한다. 1. 모든 도서의 목록을 볼 수 있다. 2. 도서의 목록을 이름으로 볼 수 있다. 3. 모든 도서의 목록을 대여횟수 순으로 볼 수 있다.
대체흐름	1-1 도서가 없을 경우 1. 화면에 도서가 표시 되지 않는다. 2-1 검색한 이름이 포함된 책이 없을 경우 1. “검색결과 없습니다”가 출력 된다. 3-1 대여횟수가 동일 할 경우 1. 책 넘버 순으로 표시 된다.

유스 케이스 명세서

유스케이스명	자동 반납
액터명	도서관
개요	도서관 시스템은 대여 기간이 지난 도서를 자동으로 반납한다.
기본흐름	선행 조건) 도서관 시스템이 가동되고 있어야 한다. 1.대여한 도서의 대여 기한이 지나면 자동으로 반납한다.

구현 및 사용기술
파일IO 역/직렬화

```
날짜 : 2018-02-23
기능 : 책 파일 쓰기
작성자명 : 박주원
*/
public void writeBook() {

    String filename = path + book.getBookNumber() + ".txt";
    try {
        FileOutputStream fos = new FileOutputStream(filename);
        ObjectOutputStream out = new ObjectOutputStream(fos);

        out.writeObject(book);

        out.close();
        fos.close();

    } catch (IOException e) {
        e.printStackTrace();
    }

}
```

```
날짜 : 2018-02-22
기능 : 도서 번호를 파라미터로 받아와서 도서번호에 해당하는 도서를 대여하는 함수
작성자명 : 김준수
*/
public Book rentBook(int num, Time time) { // 도서 대여
    String filename = "C:\\Temp\\Library\\"; // 도서관 안에 도서 폴더 경로
    FileInputStream fr = null;
    ObjectInputStream ois = null;
    Book book = null; //txt파일에서 역직렬화된 객체를 담는 변수

    try {
        fr = new FileInputStream(filename +"Books\\"+ num + ".txt");
        ois = new ObjectInputStream(fr);

        book = (Book) ois.readObject();
        if (book.isRental()) {
            FileWriter fw = new FileWriter(filename + "Users\\" + memberId + "\\" + num + ".txt");
            BufferedWriter bw = new BufferedWriter(fw);

            bw.write(num + "@" + time.getTimes()[0] + "@" + time.getTimes()[1] + "@" + book.getTitle() + "@"
                    + book.getAuthor() + "@" + book.getGenre() + "@");
            bw.newLine();
            bw.write(book.getContent());

            book.setRental(false);
            book.setRentalCount(book.getRentalCount());

            bw.close();
            fw.close();
        } else {
            System.out.println("대여가 불가능 합니다");
        }
    } catch (Exception e) {
```

구현 및 사용기술
다중폴더 재귀함수

```
날짜 : 2018-02-23
기능 : Manager는 User의 모든 대여목록을 볼 수 있으므로
     재귀함수를 사용해서 모든 User의 폴더에 들어 갈 수 있도록 해주는
     함수이다.
작성자명 : 강진광
 */
private void managerRentalSupport(File f) {
    File[] arrFS = f.listFiles(); //파일의 주소값을 담아두는 배열
    for (int i = 0; i < arrFS.length; i++) {
        if (arrFS[i].isDirectory()) {
            managerRentalSupport(arrFS[i]); //다시 폴더에 들어가게 하는 재귀 함수
        } else {
            try {
                fr = new FileReader(arrFS[i].getPath());
                br = new BufferedReader(fr);

                String line = "";
                String line1 = "";
                for (int j = 0; (line = br.readLine()) != null; j++) {
                    line1 += line;
                }
                managerConList.add(String.join("@", f.getName(), line1));
                line1 = "";
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                try {
                    br.close();
                    fr.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
```

구현 및 사용기술
인터페이스 권한

```
public class Manager implements Register {

public class User implements Register, Serializable {
```

```
package kr.or.bit.library;
/**
클래스명 : Register
날짜 : 2018-02-20
작성자명 : 아윤근
*/
public interface Register {}
```

```
날짜 : 2018-02-23
기능 : RentalBook을 사용하는 사람이 Manager인지 User인지
     판별하여 함수를 다르게 실행 시킨다.
작성자명 : 강진광
 */
public List<String> type() {
    if(register instanceof Manager) {
        return this.managerRentalList();
    }else {
        return this.userRentalList((User)register);
    }
}
/**
```

구현 및 사용기술
이너클래스

```
날짜 : 2018-02-23
기능 : 책 인기순으로 검색하기
작성자명 : 이아림
 */
public void searchPopular() { // 인기순 검색
    // rentalCount
    System.out.println("인기순 검색");

    Set<Integer> set = bookMap.keySet();

    List<Book> bookList = new ArrayList<>();

    for (Integer i : set) {
        bookList.add(bookMap.get(i));
    }

    Collections.sort(bookList, new Comparator<Book>() {
        @Override
        public int compare(Book o1, Book o2) {

            if (o1.getRentalCount() > o2.getRentalCount()) {
                return -1;
            } else if (o1.getRentalCount() < o2.getRentalCount()) {
                return 1;
            } else {
                return 0;
            }
        }
    });
    listPrint(bookList);
}
```

구현 및 사용기술
자동반납 쓰레드

```
날짜 : 2018-02-24
기능 : 현재 대여 중인 책들의 리스트를 가져오고, 기준 대여 기간과 비교 후 반납될 책들의 리스트를 다시 반환하는 함수
작성자명 : 김태웅
 */
public Map<String, String> getRentList() {
    Map<String, String> returnList = new HashMap<>();    // 변환 getName
    conList = library.rentBookList(manager);              // 모든 대여된 책 List를 가져오기 위해 manager객체를 사용
    mainTime = library.getTime();                         // 현재 시간
    Iterator<String> it = conList.iterator();
    // write --> [0] 빌려간 놈, [1]도서코드, [2]대여 날짜, [3]시스템 시간, [4]도서명, [5]작가
    while (it.hasNext()) {
        try {
            String[] item = it.next().split("@");
            if(isReturn(Long.parseLong(mainTime[1]), Long.parseLong(item[3]))) {
                returnList.put(item[1], item[0]); // Map<도서코드, 빌려간놈>
            }
        }catch (Exception e) {
        }
    }
    return returnList; // 반납될 책들의 리스트
}
```

```
날짜 : 2018-02-24
기능 : 반납할 책인지 현재시간과 대여한 시간의 차를 이용해 true/false 리턴
작성자명 : 김태웅
 */
public boolean isReturn(long now, long past) {
    boolean isReturn = false;
    long term = (now - past) / 1000;     // 초단위로 환산
    long returnPeriod = 60;              // 반납기한: 초단위.
    if(term > returnPeriod)
        isReturn = true;

    return isReturn;
}
```

```
날짜 : 2018-02-24
기능 : getRentList()에서 찾은 map<Book code, id>를 파라미터로 가져와
      library의 booklist.txt, bookMap 최신화
작성자명 : 김태웅
 */
public void returnBook(Map<String, String> map) {          // 반납될 책들의 받아서~
    Map<Integer, Book> libBookMap= library.getBookMap();   // 도서관에 있는 전체 책 map 받고
    Set<String> bookcode = map.keySet();                   // Map<도서코드, 빌려간놈>
    Iterator<String> it = bookcode.iterator();
    while(it.hasNext()) {
        String code = it.next();
        File f = new File(String.join("\\", PATH, map.get(code), code) + ".txt");
        // join("\\", PATH, id, code) + ".txt"
        if(f.exists()) {
            f.delete();
            libBookMap.get(Integer.parseInt(code)).setRental(true);
            library.bookModify(libBookMap.get(Integer.parseInt(code)));
            System.out.println("Auto Book Return");
        }else {
            System.out.println("자동 반납할 책이 존재하지 않습니다.");
        }
    }
    library.listModify(libBookMap);
}
```

```
날짜 : 2018-02-24
기능 : Library System에서 background에서 자동 도서 반납 기능을 수행해줄 thread run 함수
작성자명 : 김태웅
 */
@Override
public void run() {
    while(true) {
        returnBook(getRentList());
        try {
            Thread.sleep(30000);                    // x / 1000초만큼 Auto-Return 반복
        } catch (InterruptedException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Thank you