Minescu Andrei 1110EB

Tema 2: Aplicatie pentru gestiunea eficienta a colectiilor de carti ale unei biblioteci

Design Class Diagram

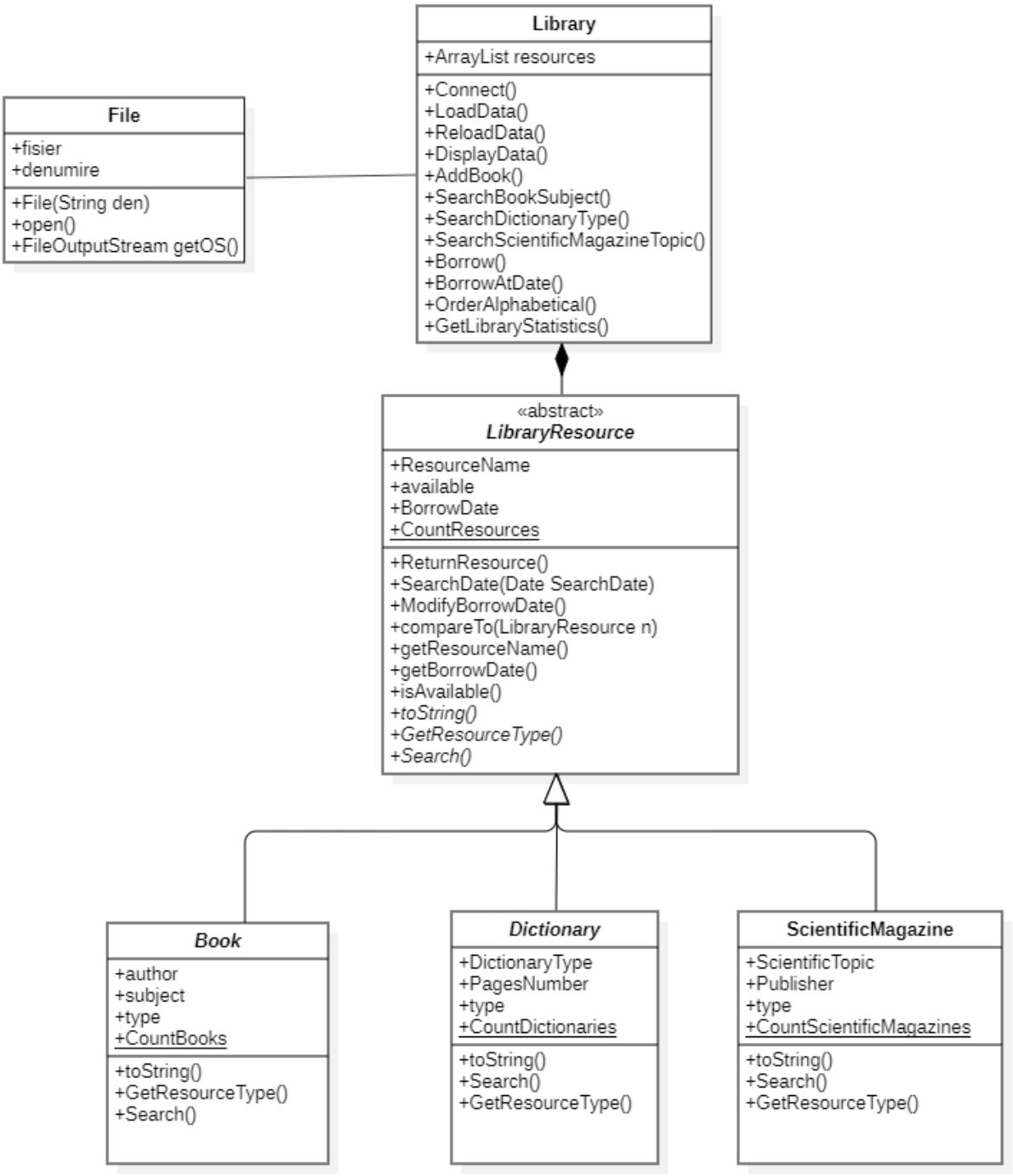

Fig 1: Schema Design Class Diagram

Clasa LibraryResource:

- Este o clasa abstracta ce continue metodele abstracte: toString(), GetResourceType(), Search()
- Constructorul va initializa variabilele ce reprezinta date commune tuturor resurselor din biblioteca indifferent de tip:
  - ResourceName - reprezeinta numele resursei din biblioteca
  - Available - reprezinta starea resursei din punct de vedere a disponibilitatii pentru a fi imprumutata (1-disponibil pentru imprumut / 0-indisponibil pentru imprumut, deja imprumutata)
  - BorrowDate – reprezinta data la care resursa a fost imprumutata

- Metodele acestei clase de baza sunt pe langa cei 2 constructori,
  - getResourceName: Getter pentru a returna numele resursei
  - getBorrowDate: Getter pentru a returna data la care a fost imprumutata resursa
  - isAvailable: Getter pentru a returna starea de disponibilitate a resursei
  - SearchDate(Date SearchDate): Va returna daca obiectul are sau nu data de imprumut cautat
  - compareTo: clasa implementeaza interfata Comparable pentru a sortar obiectele în ordine alfabetica

```
public abstract class LibraryResource implements Comparable<LibraryResource> {
    //implements Comparable<LibraryResource>

    protected enum ResourceType{BOOK, DICTIONARY, MAGAZINE};

    private String ResourceName;
    private boolean available;
    java.sql.Date BorrowDate;

    static int CountResources = 0;

    public LibraryResource(String ResourceName, boolean available, java.sql.Date BorrowDate) {
        this.ResourceName = ResourceName;
        this.available = available;
        this.BorrowDate = BorrowDate;
        CountResources++;
    }
```

Fig 2: Atribute si constructor pentru clasa LibraryResource

```
public abstract String toString();

public abstract String GetResourceType();

public abstract String Search();
```

Fig 3: Metodele abstracte din clasa LibraryResource

Clasa Book:

- Mosteneste clasa astracta LibraryResource
- Pe langa atributele mostenite, aceasta clasa contine si atributele proprii:
  - Author - autorul cartii
  - Subject – subiectul general al cartii
  - Type – Reprezinta tipul resursei si este de tip Enum si va fi de tipul BOOK
  - Variabila statica CountBooks va contabiliza numarul obiectelor de tip Book

- Metodele implementate in cadrul acestei clase sunt:
  - toString: afiseaza in consola obiectul curent de tip Clasa
  - GetResourceType: returneza tipul ca tip String
  - Search: returneaza subiectul general al cartii

```
public final class Book extends LibraryResource{

    private String author;
    private String subject;
    private ResourceType type;
    static int CountBooks = 0;

    public Book()
    {
        super();
        this.type = ResourceType.BOOK;
        this.author = null;
        this.subject = null;
        CountBooks++;
    }
```

```
public Book(String ResourceName, boolean available, java.sql.Date BorrowDate, String author, String subject) {
    super(ResourceName, available, BorrowDate);
    this.author = author;
    this.subject = subject;
    this.type = ResourceType.BOOK;
    CountBooks++;
}

public String toString(){
    return String.format("%-30s | %10s | %5s | %10s | %-25s | %15s", this.getResourceName(), type, isAvailable(), getBorrowDate(), "Autor: "+author, "Subiect: "+subject);
}

public String GetResourceType() { return ("Book"); }

public String Search() { return this.subject; }
```

Fig 4: Clasa derivata Book

Clasa Dictionary

- La fel ca la clasa Book, variabilele proprii sunt:
  - DictionaryType – tipul dictionarului
  - PagesNumber – numarul de pagini
  - type – Reprezinta tipul resursei si este de tip Enum si va fi de tipul DICTIONARY
  - Variabila statica CountDictionaries va contabiliza numarul obiectelor de tip Book
- Metodele implementate in cadrul acestei clase sunt:
  - toString: afiseaza in consola obiectul curent de tip Clasa
  - GetResourceType: returneza tipul ca tip String
  - Search: returneaza subiectul general al cartii

```
public final class Dictionary extends LibraryResource{

    private String DictionaryType;
    private int PagesNumber;
    private ResourceType type;

    static int CountDictionaries = 0;

    public Dictionary()
    {
        super();
        this.type = ResourceType.DICTIONARY;
        this.DictionaryType = null;
        this.PagesNumber = 0;
        CountDictionaries++;
    }
```

Fig 5: Clasa derivata Dictionary (Variabile private, statice si constructorul implicit)

Clasa Library

- Atributul principal al acestei clase este resources de tip ArrayList
- Pentru utilizarea unui driver-ului JDBC in metodele clasei se vor utiliza atributele url, con si instr
- Metodele implementate in cadrul acestei clase sunt:

  - Connect(): conectare la baza de date
  - LoadData(): Folosind instructiunea SQL SELECT va insera in cadrul resources toate datele din baza de date
  - DisplayData(): Va afisa folosind metoda abstracta toString toate obiectele din resources
  - AddBook(): Utilizand instructiunea SQL INSERT va adauga o noua inregistrare de tip Book in baza de date
  - SearchBookSubject(): Afiseaza cartile dintr-un anumir domeniu introdus de la tastatura
  - SearchDictionaryType(): Afiseaza dictionarele de un anumit tip introdus de la tastatura
  - SearchScientificMagazineTopic(): Afiseaza revistele stiintifice cu un anumit subiect introdus de la tastatura
  - Borrow(): Modifica in baza de date disponibiltatea unei carti de la disponibil la indisponibil(imprumutat)
  - GetLibraryStatistics(): Va afisa intr-un fisier text statsitcii despre numarul de resurse din fiecare tip

```
ublic class Library {

    private ArrayList<LibraryResource> resources;

    Library() { resources = new ArrayList<LibraryResource>(); }

    private String url = "jdbc:mysql://localhost:3306/library";
    public Connection con;
    public Statement instr;
```

```
public void LoadData() throws SQLException{
    String sql = "SELECT * FROM library";
    ResultSet rs = instr.executeQuery(sql);
    while (rs.next()) {
        String ResourceType = rs.getString( columnLabel: "ResourceType");

        if (ResourceType.equals("Book")) {
            String ResourceName = rs.getString( columnLabel: "Name");
            boolean available = (Boolean) rs.getObject( columnLabel: "available");
            java.sql.Date BorrowDate = rs.getDate( columnLabel: "BorrowDate");
            String author = rs.getString( columnLabel: "Author");
            String subject = rs.getString( columnLabel: "Subject");
            LibraryResource.ResourceType type = LibraryResource.ResourceType.BOOK;

            Book CurrentBook = new Book(ResourceName, available, BorrowDate, author, subject);
            resources.add(CurrentBook);
        }
```

Bibliografie

[1] https://www.geeksforgeeks.org

[2] Matt Weisfeld - The object-oriented thought process, Addison-Wesley Professional, 2008

[3] https://www.softwaretestinghelp.com/array-of-objects-in-java/

[4] https://www.tutorialspoint.com/java/java_object_classes.htm

[5] https://howtodoinjava.com/java/oops/object-oriented-programming/