# Subject.java

**package** com.fsd.corejava.entities;

**import** java.io.Serializable;

**import** java.util.Set;

/\*\*

\* **@author** Mayukh

\*

\*/

**public** **class** Subject **implements** Serializable {

**private** **static** **final** **long** ***serialVersionId*** = 1L;

**private** **long** subjectId;

**private** String subTitle;

**private** **int** durationInHours;

**private** Set<Book> references;

**public** Subject() {

};

**public** Subject(**long** subjectId, String subTitle, **int** durationInHours, Set<Book> references) {

**super**();

**this**.subjectId = subjectId;

**this**.subTitle = subTitle;

**this**.durationInHours = durationInHours;

**this**.references = references;

}

/\*\*

\* **@return** the subjectId

\*/

**public** **long** getSubjectId() {

**return** subjectId;

}

/\*\*

\* **@param** subjectId the subjectId to set

\*/

**public** **void** setSubjectId(**long** subjectId) {

**this**.subjectId = subjectId;

}

/\*\*

\* **@return** the subTitle

\*/

**public** String getSubTitle() {

**return** subTitle;

}

/\*\*

\* **@param** subTitle the subTitle to set

\*/

**public** **void** setSubTitle(String subTitle) {

**this**.subTitle = subTitle;

}

/\*\*

\* **@return** the durationInHours

\*/

**public** **int** getDurationInHours() {

**return** durationInHours;

}

/\*\*

\* **@param** durationInHours the durationInHours to set

\*/

**public** **void** setDurationInHours(**int** durationInHours) {

**this**.durationInHours = durationInHours;

}

/\*\*

\* **@return** the references

\*/

**public** Set<Book> getReferences() {

**return** references;

}

/\*\*

\* **@param** references the references to set

\*/

**public** **void** setReferences(Set<Book> references) {

**this**.references = references;

}

/\*

\* (non-Javadoc)

\*

\* @see java.lang.Object#toString()

\*/

@Override

**public** String toString() {

**return** "Subject [subjectId=" + subjectId + ", subTitle=" + subTitle + ", durationInHours=" + durationInHours

+ ", references=" + references + "]";

}

}

# Book.java

package com.fsd.corejava.entities;

import java.io.Serializable;

import java.time.LocalDate;

/\*\*

@author Mayukh

\*

\*/

public class Book implements Serializable {

private static final long serialVersionId = 1L;

private long bookId;

private String title;

private double price;

private Integer volume;

private LocalDate publishDate;

public Book() {

};

public Book(long bookId, String title, double price, Integer volume, LocalDate publishDate) {

super();

this.bookId = bookId;

this.title = title;

this.price = price;

this.volume = volume;

this.publishDate = publishDate;

}

/\*\*

@return the bookId

\*/

public long getBookId() {

return bookId;

}

/\*\*

@param bookId the bookId to set

\*/

public void setBookId(long bookId) {

this.bookId = bookId;

}

/\*\*

@return the title

\*/

public String getTitle() {

return title;

}

/\*\*

@param title the title to set

\*/

public void setTitle(String title) {

this.title = title;

}

/\*\*

@return the price

\*/

public double getPrice() {

return price;

}

/\*\*

@param price the price to set

\*/

public void setPrice(double price) {

this.price = price;

}

/\*\*

@return the volume

\*/

public Integer getVolume() {

return volume;

}

/\*\*

@param volume the volume to set

\*/

public void setVolume(Integer volume) {

this.volume = volume;

}

/\*\*

@return the publishDate

\*/

public LocalDate getPublishDate() {

return publishDate;

}

/\*\*

@param publishDate the publishDate to set

\*/

public void setPublishDate(LocalDate publishDate) {

this.publishDate = publishDate;

}

/\*

(non-Javadoc)

\*

@see java.lang.Object#toString()

\*/

@Override

public String toString() {

return "Book [bookId=" + bookId + ", title=" + title + ", price=" + price + ", volume=" + volume

+ ", publishDate=" + publishDate + "]";

}

}

# FileOperations.java

package com.fsd.corejava.main;

/\*\*

@author Mayukh

\*

\*/

import java.io.BufferedReader;

import java.io.EOFException;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.time.LocalDate;

import java.util.ArrayList;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import com.fsd.corejava.entities.Book;

import com.fsd.corejava.entities.Subject;

public class FileOperations {

static FileOutputStream fo = null;

static ObjectOutputStream osw = null;

static {

try {

fo = new FileOutputStream(new File("MyFileNew.txt"));

osw = new ObjectOutputStream(fo);

} catch (Exception e) {

e.printStackTrace();

}

}

public static void main(String[] args) throws IOException {

char select;

do {

Boolean isBookType = Boolean.FALSE;

System.out.println("please select menu items::::::::::::::::");

System.out.println("a.Add a Subject");

System.out.println("b.Add a Book");

System.out.println("c.Delete a Subject");

System.out.println("d.Delete a book");

System.out.println("e.Search for a book");

System.out.println("f.Search for a subject");

System.out.println("g.Exit");

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

select = br.readLine().charAt(0);

switch (select) {

case 'a':

System.out.println("Add a Subject");

System.out.println("Enter Subject Id: ");

Subject subObj = new Subject();

long subjectId = Long.parseLong(br.readLine());

subObj.setSubjectId(subjectId);

System.out.println("Enter Subject Title: ");

String subjectTitle = br.readLine();

subObj.setSubTitle(subjectTitle);

System.out.println("Enter Subject durationInHours: ");

String durationInHours = br.readLine();

subObj.setDurationInHours(Integer.parseInt(durationInHours));

Set<Book> refSet = new HashSet<>();

Book b1 = new Book(1L, "Java", 100.0, 300, LocalDate.now());

refSet.add(b1);

subObj.setReferences(refSet);

fileWriter(subObj, isBookType);

break;

case 'b':

System.out.println("Add a Book");

System.out.println("Enter Book Id: ");

Book bookObj = new Book();

long bookId = Long.parseLong(br.readLine());

bookObj.setBookId(bookId);

System.out.println("Enter Book Title: ");

String bookTitle = br.readLine();

bookObj.setTitle(bookTitle);

System.out.println("Enter price: ");

double price = Double.parseDouble(br.readLine());

bookObj.setPrice(price);

System.out.println("Enter volume: ");

int volume = br.read();

bookObj.setVolume(volume);

LocalDate publishDate = LocalDate.now();

bookObj.setPublishDate(publishDate);

isBookType = Boolean.TRUE;

fileWriter(bookObj, isBookType);

break;

case 'c':

System.out.println("Delete a Subject");

System.out.println("Enter Subject Id to delete: ");

long subId = Long.parseLong(br.readLine());

removeFileObj(subId, isBookType);

break;

case 'd':

System.out.println("Delete a Book");

System.out.println("Enter Book Id to delete: ");

long bId = Long.parseLong(br.readLine());

isBookType = Boolean.TRUE;

removeFileObj(bId, isBookType);

case 'e':

System.out.println("<---Search for a Book--->");

System.out.println("Enter Book Id to Search: ");

long bookIdSearch = Long.parseLong(br.readLine());

isBookType = Boolean.TRUE;

fileReader(bookIdSearch, isBookType);

break;

case 'f':

System.out.println("<---Search for a Subject--->");

System.out.println("Enter Subject Id to Search: ");

long SubIdSearch = Long.parseLong(br.readLine());

fileReader(SubIdSearch, isBookType);

break;

}

} while (select != 'g');

}

// Write Object to File method

public static void fileWriter(Object obj, Boolean isBookType) {

try {

// Write Object to file

osw.writeObject(obj);

if (isBookType) {

System.out.println("Book has been added successfully");

} else {

System.out.println("Subject has been added successfully");

}

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

// Read Object to File method

public static void fileReader(long searchId, Boolean isBookType) {

try {

FileInputStream fi = new FileInputStream(new File("MyFileNew.txt"));

ObjectInputStream oi = new ObjectInputStream(fi);

List objList = new ArrayList();

// Read Object from file

while (true) {

try {

objList.add(oi.readObject());

} catch (EOFException e) {

// TODO: handle exception

break;

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

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

List<Subject> subjectList = new ArrayList<>();

for (Object obj1 : objList) {

try {

Book bookObj = (Book) obj1;

bookList.add(bookObj);

} catch (Exception e) {

Subject subObj = (Subject) obj1;

subjectList.add(subObj);

// TODO: handle exception

}

}

System.out.println("objList: " + objList);

System.out.println("BookList: " + bookList);

System.out.println("SubjectList: " + subjectList);

Object searchItem = null;

if (isBookType) {

for (Book bObj : bookList) {

if (bObj.getBookId() == searchId) {

searchItem = bObj;

System.out.println("Book Details: " + searchItem);

break;

}

}

if (searchItem == null) {

System.out.println("No Data found...");

}

} else {

for (Subject sObj : subjectList) {

if (sObj.getSubjectId() == searchId) {

searchItem = sObj;

System.out.println("Subject Details: " + searchItem);

break;

}

}

if (searchItem == null) {

System.out.println("No Data found...");

}

}

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

// Remove a object from File

public static void removeFileObj(long objectId, Boolean isBookType) {

try {

FileInputStream fi = new FileInputStream(new File("MyFileNew.txt"));

ObjectInputStream oi = new ObjectInputStream(fi);

List objList = new ArrayList();

// Read Object from file

while (true) {

try {

objList.add(oi.readObject());

} catch (EOFException e) {

break;

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

// Create 2 different lists based on object type

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

List<Subject> subjectList = new ArrayList<>();

for (Object obj1 : objList) {

try {

Book bookObj = (Book) obj1;

bookList.add(bookObj);

} catch (Exception e) {

Subject subObj = (Subject) obj1;

subjectList.add(subObj);

}

}

Object removeObj = null;

if (isBookType) {

for (Book bObj : bookList) {

if (bObj.getBookId() == objectId) {

System.out.println("Book Details: " + bObj);

removeObj = bObj;

break;

}

}

} else {

for (Subject sObj : subjectList) {

if (sObj.getSubjectId() == objectId) {

System.out.println("Subject Details: " + sObj);

removeObj = sObj;

break;

}

}

}

if (removeObj == null) {

System.out.println("No match found..");

} else {

objList.remove(removeObj);

}

System.out.println("Updated objList: " + objList);

//Writing the updated list to file

FileOutputStream fo = new FileOutputStream(new File("MyFileNew.txt"));

ObjectOutputStream os = new ObjectOutputStream(fo);

for (Object obj : objList) {

os.writeObject(obj);

}

if (isBookType && removeObj != null) {

System.out.println("Book has been removed Successfully");

} else if (isBookType && removeObj != null) {

System.out.println("Subject has been removed Successfully");

}

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

}

# Output Screenshot:



















