

Звіт

Лабораторна работа 8.

Основи введення/виведення Java SE

Мета роботи:

Оволодіння навичками управління введенням/виведенням даних з використанням класів платформи Java SE.

1. ВИМОГИ

- 1) Забезпечити можливість збереження і відновлення масива об'єктів рішення завдання лабораторної роботи №7.
- 2) Забороняється використання стандартного протокола серіалізації.
- 3) Продемонструвати використання моделі Long Term Persistence.
- 4) Забезпечити діалог з користувачем у вигляді простого текстового меню.
- 5) При збереженні та відновленні даних забезпечити діалоговий режим вибору директорії з відображенням вмісту і можливістю переміщення по підкаталогах.

1.1. Розробник: Татаренко Андрій Геннадійович, КІТ-119а, варіант №20.

2. ОПИС ПРОГРАМИ

2.1. Засоби ООП: класи, методи класу, domain-об'єкти.

2.2. Ієрархія та структура класів: публічний клас Main, клас-контейнер, та клас Book.

2.3. Важливі фрагменти програми:

```
public static void main(String... str)
{
    List array = new List();
    String[] Authors1 = {"Sposobin I.V."};
    String[] Genre1 = {"Education"};
    Book book1 = new Book(9785811425396L, "Elementari Music Theory", Authors1, "Planet of
Music", Genre1, 2018);
    array.AddObject(book1);

    String[] Authors2 = {"Sapkovskii A."};
    String[] Genre2 = {"Romance", "Epic Fantasy"};
    Book book2 = new Book(5792100810L, "Witcher The Last Wish", Authors2, "superNOWA",
Genre2, 1993);
    array.AddObject(book2);

    boolean stop = false;
    Scanner scan = new Scanner(System.in);
    int choice;

    while(!stop)
    {
        System.out.println("What to do?");
        System.out.println("1. Output data");
        System.out.println("2. Add element");
        System.out.println("3. Delete element");
        System.out.println("4. Serialize data");
        System.out.println("5. Deserialize data");
        System.out.println("6. End program");
        System.out.println("=====");
        System.out.print("Your choice: ");

        choice = scan.nextInt();

        switch (choice) {
            case 1:
                System.out.println();
                for (int i = 0; i < array.GetSize(); i++) {
                    System.out.println(i+1 + " ");
                    array.array[i].Output();
                    System.out.println();
                }
                break;

            case 2:
                System.out.print("Enter ISBN: ");
                long ISBN = scan.nextInt();
                scan.nextLine();
                System.out.print("Enter a name: ");
                String Name = scan.nextLine();
                System.out.print("Enter a publisher: ");
                String Publish = scan.nextLine();
                System.out.print("Enter a date: ");
                int Date = scan.nextInt();
                System.out.print("Enter count of authors: ");
                int value = scan.nextInt();
```

```

if(value < 1)
{
    System.out.println("Error. Wrong list size.");
    break;
}
System.out.print("Enter authors name:");
String[] listA = new String[value];
scan.nextLine();
for (int i = 0; i < value; i++) {
    listA[i] = scan.nextLine();
}
System.out.print("Enter count of genres: ");
value = scan.nextInt();
if(value < 1)
{
    System.out.println("Error. Wrong list size.");
    break;
}
System.out.print("Enter ganres: ");
String[] listG = new String[value];
scan.nextLine();
for (int i = 0; i < value; i++) {
    listG[i] = scan.nextLine();
}

System.out.println("\nBook added.\n");

Book newBook = new Book(ISBN, Name, listA, Publish, listG, Date);
array.AddObject(newBook);

break;

```

case 3:

```

System.out.println();
for (int i = 0; i < array.GetSize(); i++) {
    System.out.println(i+1 + " ");
    array.array[i].Output();
    System.out.println();
}

System.out.print("Enter the number of element: ");
int position = scan.nextInt();
if(position > array.GetSize() || position < 1)
{
    System.out.println("Error.Wrong ID.");
    break;
}
array.DeleteObject(position);
System.out.println("\nElement deleted.\n");

break;

```

case 4:

```

String address = new File("").getAbsolutePath();
File folder = new File(address);
File[] arrayFiles = folder.listFiles();
String filename;
String currentDirectory = address;
String highestDir = folder.getName();

boolean stop2 = false;
int index = 0;
int choise2 = 0;

```

```

        System.out.print("\nEnter XML file name: ");
        scan.nextLine();
        filename = scan.nextLine();

        if (filename.indexOf(".xml") == -1) {
            filename += ".xml";
        }

        while(!stop2)
        {
            index = 0;

            System.out.println("\nCurrent path: " + currentDirectory);
            System.out.println("Current XML file name: " + filename);
            System.out.println("\nFiles and directories in current path:");
            for (index = 0; index < arrayFiles.length; index++) {
                System.out.println(index+1 + ". " +
arrayFiles[index].toString().substring(currentDirectory.length()+1));
            }

            System.out.println();
            System.out.println("What to do?");
            System.out.println("1. Write XML file in current directory");
            System.out.println("2. Go up one level folder");
            System.out.println("3. Enter the folder");
            System.out.println("4. Change the XML file name");
            System.out.println("5. Leave the serialization");

            System.out.println("=====");
            System.out.print("Your choice: ");
            choice2 = scan.nextInt();

            switch(choice2)
            {
            case 1:
                stop2 = true;
                break;

            case 2:
                if(folder.getName().equals(highestDir))
                {
                    System.out.print("\nYou can't go up one level
folder.");
                    break;
                }
                currentDirectory = currentDirectory.substring(0,
currentDirectory.indexOf(folder.getName())-1);
                folder = new File(currentDirectory);
                arrayFiles = folder.listFiles();

                break;

            case 3:
                boolean choice3 = false;

                while(!choice3)
                {
                    System.out.print("\nChoose the number of directory:

");
                    index = scan.nextInt();
                    if(index < 1 || index > arrayFiles.length ||
!arrayFiles[index-1].isDirectory())

```

```

        {
            System.out.println("That's not a directory.

Try another.");
        }
        else
        {
            currentDirectory = arrayFiles[index-

1].toString();

            System.out.println("New current directory: "

+ currentDirectory);

            folder = new File(currentDirectory);
            arrayFiles = folder.listFiles();
            choise3 = true;
        }
    }
    break;

case 4:
    System.out.print("\nEnter XML file name: ");
    scan.nextLine();
    filename = scan.nextLine();

    if (filename.indexOf(".xml") == -1) {
        filename += ".xml";
    }
    break;

case 5:
    System.out.println("Leaving the serialization section");
    break;

default:
    System.out.println("Error. The wrong command. Try again");
    break;

}

}
address = currentDirectory;
System.out.println("\nFile will be written in current directory: " + address);
System.out.println("XML file name: " + filename);
folder = new File(address);
File realFile = new File(folder,filename);
try {
    XMLEncoder encoder = new XMLEncoder(new
BufferedOutputStream(new FileOutputStream(realFile)));
    encoder.writeObject(array.array);
    encoder.close();
} catch (Exception e) {
    System.out.println(e);
    break;
}
System.out.println("Serialization successful.\n");

break;

case 5:
    address = new File("").getAbsolutePath();
    folder = new File(address);
    arrayFiles = folder.listFiles();
    currentDirectory = address;
    highestDir = folder.getName();

```

```

stop2 = false;
index = 0;
choise2 = 0;

while(!stop2)
{
    index = 0;

    System.out.println("\nCurrent path: " + currentDirectory);
    System.out.println("Files and directories in current path:");
    for (index = 0; index < arrayFiles.length; index++) {
        System.out.println(index+1 + ". " +
arrayFiles[index].toString().substring(currentDirectory.length()+1));
    }

    System.out.println();
    System.out.println("What to do?");
    System.out.println("1. Read XML file in current directory");
    System.out.println("2. Go up one level folder");
    System.out.println("3. Enter the folder");
    System.out.println("4. Leave the deserialization");

    System.out.println("=====");
    System.out.print("Your choise: ");
    choise2 = scan.nextInt();

    switch(choise2)
    {
    case 1:
        System.out.print("\nEnter the id of file: ");
        index = scan.nextInt();
        if(arrayFiles[index-1].getName().indexOf(".xml")==-1 ||
arrayFiles[index-1].isDirectory())
        {
            System.out.println("That's not an .XML file.");
            break;
        }

        stop2 = true;
        break;

    case 2:
        if(folder.getName().equals(highestDir))
        {
            System.out.println("You can't go up one level
folder.");
            break;
        }
        currentDirectory = currentDirectory.substring(0,
currentDirectory.indexOf(folder.getName())-1);
        folder = new File(currentDirectory);
        arrayFiles = folder.listFiles();

        break;

    case 3:
        boolean choise3 = false;

        while(!choise3)
        {
            System.out.print("\nChoose the number of directory:

");
            index = scan.nextInt();

```

```

!arrayFiles[index-1].isDirectory()

Try another.");

1].toString();
+ currentDirectory);

if(index < 1 || index > arrayFiles.length ||
{
    System.out.println("That's not a directory.
}
else
{
    currentDirectory = arrayFiles[index-
    System.out.println("New current directory: "
    folder = new File(currentDirectory);
    arrayFiles = folder.listFiles();
    chose3 = true;
}
}
break;

case 4:
    System.out.println("Leaving the serialization section");
    stop2 = true;
    break;

default:
    System.out.println("Error. The wrong command. Try again");
    break;

}

}
address = currentDirectory;
System.out.println("XML file address: " + address + "\\" + arrayFiles[index-
1].getName());

address = address + "\\" + arrayFiles[index-1].getName();
folder = new File(address);
try {
    XMLDecoder decoder = new XMLDecoder(new
BufferedInputStream(new FileInputStream(folder)));
    array.array = (Book[])decoder.readObject();
    decoder.close();
    array.SetSize(array.array.length);
} catch (Exception e) {
    System.out.println();
    break;
}
System.out.println("Deserialization successful.\n");

break;

case 6:
    System.out.println("\nTerminating the program");
    stop = true;
    break;

default:
    System.out.println("Error. Wrong command. Try again.");
    break;

}

}

scan.close(); }

```

Результат виконання програми

```
3. Delete element
4. Serialize data
5. Deserialize data
6. End program
=====
Your choise: 1

1)
ISBN: 9785811425396
Название: Elementari Music Theory
Автор: Sposobin I.V.,
Издание: Planet of Music
Жанр: Education,
Год публикации: 2018

2)
ISBN: 5792100810
Название: Witcher The Last Wish
Автор: Sapkovskii A.,
Издание: superNOWA
Жанр: Romance, Epic Fantasy,
Год публикации: 1993

What to do?
1. Output data
2. Add element
3. Delete element
4. Serialize data
5. Deserialize data
6. End program
=====
Your choise:
```

Висновки

При виконанні даної лабораторної роботи було набуто навички сереалізації domain-об'єктів. Розробили меню.

Програма протестована, виконується без помилок.