Звіт

Лабораторна работа 8. Основи введення/виведення Java SE

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

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: Момот Роман Євгенійович, КІТ119-а, варіант №14.

2. ОПИС ПРОГРАМИ

2.1. Засоби ООП: клас, метод класу, поле класу.

2.2. Ієрархія та структура класів: один публічний клас Main, публічний клас Event, у полях якого є час початку події, тривалість, адреса події, імена людей, гетери, сетери, конструктор класу та метод виведення даних класу. Також є клас EventList, у полях якого є масив елементів класу Event, розмір масиву, гетери та сетери поля розміру, методи додавання та видалення елементів.

2.3. Важливі фрагменти програми:

```
public class EventList {

    private int size = 0;

    Event[] array = new Event[size];



    public int getSize() {

        return size;

    }
```

```
public void setSize(int size) {
	this.size = size;
}

public void addEl(Event event)
{
	Event[] newArray = new Event[size+1];
	for (int i = 0; i < size; i++) {
		newArray[i] = array[i];
	}

	newArray[size] = event;
	size++;
	array = newArray;
}
public void deleteEl(int position)
{
	if(size != 0)
	{
		Event[] newArray = new Event[size-1];
		for (int i = 0; i < position-1; i++) {
			newArray[i] = array[i];
		}
		for (int i = position-1, j = position; j < size; i++, j++) {
			newArray[i] = array[j];
		}

		size--;
		array = newArray;
```

```
                }
                else
                {
                        System.out.println("Array is empty");
                }
        }
}


public static void main(String[] args) {
                EventList array = new EventList();
                String[] listOfPeople = {"Дмитрий Иванов", "Александр Гекторов",
"Иван Романов"};
                GregorianCalendar date1 = new GregorianCalendar(2017, 5, 28);
                date1.set(Calendar.HOUR_OF_DAY, 18);
                date1.set(Calendar.MINUTE, 0);
                date1.set(Calendar.SECOND, 10);
                Event event1 = new Event(date1,180,"Проспект Льва Ландау
87",listOfPeople);
                array.addEl(event1);


                String[] listOfPeople2 = {"Махатма Ганди", "Иисак Ньютон",
"Джордж Буш Младший"};
                date1 = new GregorianCalendar(2002, 1, 1);
                date1.set(Calendar.HOUR_OF_DAY, 9);
                date1.set(Calendar.MINUTE, 30);
                date1.set(Calendar.SECOND, 00);
                event1 = new Event(date1,45,"Площадь
Конституции",listOfPeople2);
                array.addEl(event1);
```

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

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 choise: ");

	choise = scan.nextInt();

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

	case 2:
```

```
System.out.print("\nEnter number of participants: ");
int value = scan.nextInt();
if(value < 1)
{
	System.out.println("Error. Wrong list size.");
	break;
}

String[] list = new String[value];
System.out.println("Enter list of names:");
scan.nextLine();
for (int i = 0; i < value; i++) {
	System.out.print(i+1 + ". ");
	list[i] = scan.nextLine();
}

GregorianCalendar date = new GregorianCalendar();
System.out.print("Enter event year: ");
value = scan.nextInt();
date.set(Calendar.YEAR, value);
System.out.print("Enter event month: ");
value = scan.nextInt();
date.set(Calendar.MONTH, value);
System.out.print("Enter event day: ");
value = scan.nextInt();
date.set(Calendar.DAY_OF_MONTH, value);
System.out.print("Enter event hour: ");
value = scan.nextInt();
date.set(Calendar.HOUR_OF_DAY, value);
```

```
            System.out.print("Enter event minute: ");
            value = scan.nextInt();
            date.set(Calendar.MINUTE, value);

            System.out.print("Enter event address: ");
            scan.nextLine();
            String temp = scan.nextLine();
            System.out.print("Enter event length: ");
            value = scan.nextInt();
            System.out.println("\nEvent added.\n");

            Event newEvent = new Event(date,value,temp,list);
            array.addEl(newEvent);

            break;

        case 3:
            System.out.println();
            for (int i = 0; i < array.getSize(); i++) {
                System.out.println(i+1 + ") ");
                array.array[i].outputData();
                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.deleteEl(position);
                System.out.println("\nElement deleted.\n");

                break;

            case 4:
                String address = new File("").getAbsolutePath();   //адрес начальной директории
                File folder = new File(address);                   //создание файла
                File[] arrayFiles = folder.listFiles();    //список файлов в текущений директории
                String filename;
        //название файла для записи
                String currentDirectory = address;                 //адрес текущей директории
                String highestDir = folder.getName();          //название максимально допустимой высокой аудитории

                boolean stop2 = false;     //выход из цикла выбора директории
                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 choise: ");
```

```
                    choise2 = scan.nextInt();

                    switch(choise2)
                    {
                    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();    //список файлов в текущений директории

                        break;

                    case 3:
                        boolean choise3 = false;

                        while(!choise3)
                        {
                            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();
        //список файлов в текущений директории
                        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();    //адрес
начальной директории
                    folder = new File(address);
        //создание файла
                    arrayFiles = folder.listFiles();                    //список
файлов в текущений директории
                    currentDirectory = address;
        //адрес текущей директории
                    highestDir = folder.getName();
        //название максимально допустимой высокой аудитории


                    stop2 = false;        //выход из цикла выбора
директории
                    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 serialization");

        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();    //список
файлов в текущений директории

                    break;

                case 3:
                    boolean choise3 = false;

                    while(!choise3)
                    {
                        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();
        //список файлов в текущений директории
                                            choise3 = 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 = (Event[])decoder.readObject();
                            decoder.close();
                            array.setSize(array.array.length);
                        } catch (Exception e) {
                            //System.out.println(e);
                            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;
                    }
                }
            }
```

Висновки

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи з Java SE.

Програма протестована, виконується без помилок.