

8. Основи введення/виведення Java SE

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

1 ВИМОГИ

1.1 Розробник

Інформація про розробника:

- Кононенко Дмитро Олексійович
- НТУ “ХПІ” 1.KIT102.8a
- Варіант 8

1.2 Загальне завдання

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.

1.3 Задача

Варіант 8. Автостанція. Запис в розкладі: номер рейсу; час відправлення; дні тижня; кількість вільних місць; маршрут - необмежений набір значень у вигляді “назва станції, час прибуття”.

2 ОПИС ПРОГРАМИ

2.1 Засоби ООП

У даній програмі присутні об'єктно-орієнтовані методи:

Інкапсуляція – захист даних від неправомірного користування.

2.2 Ієрархія та структура даних

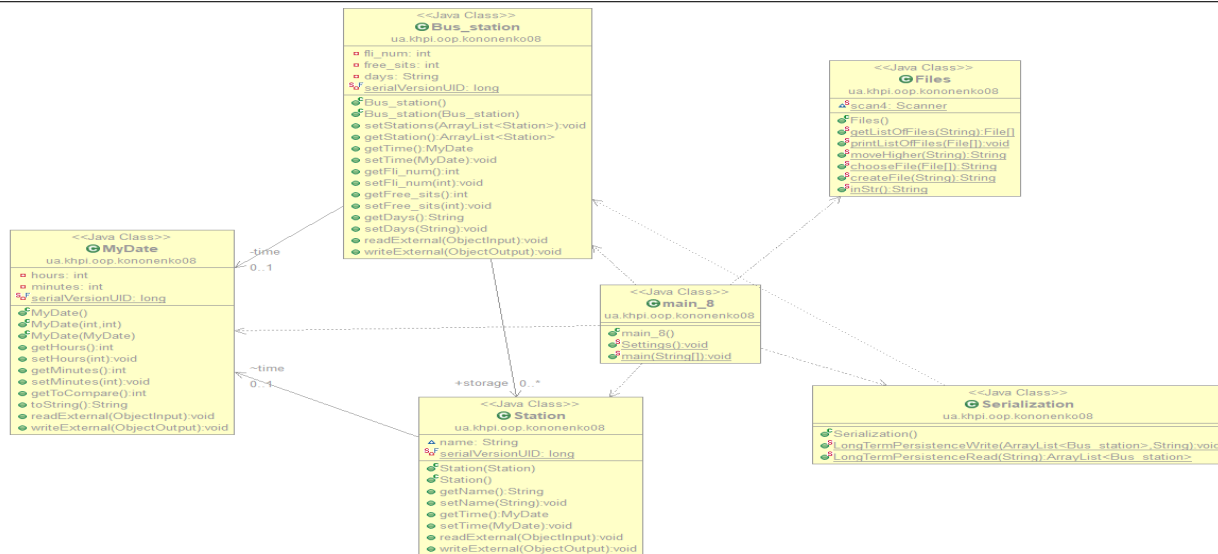


Рисунок 1 – Діаграма класів

2.3 Важливі фрагменти програми

```
78● @SuppressWarnings("unchecked")
79 @Override
80 public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {
81     // TODO Auto-generated method stub
82     fli_num = in.readInt();
83     free_sits = in.readInt();
84     days = (String) in.readObject();
85     time = ((MyDate) in.readObject());
86     storage = ((ArrayList<Station>) in.readObject());
87 }
88
89● @Override
90 public void writeExternal(ObjectOutput out) throws IOException {
91
92     out.writeObject(getFli_num());
93     out.writeObject(getFree_sits());
94     out.writeObject(getTime());
95     out.writeObject(getDays());
96     out.writeObject(getStation());
97 }
98
99 }
100
```

Рисунок 2 – Реалізація нестандартного протоколу серіалізації та десеріалізації

```
12 public class Serialization {
13
14● public static void LongTermPersistenceWrite(ArrayList<Bus_station> object, String path) throws FileNotFoundException {
15
16     XMLEncoder encoder = new XMLEncoder(
17         new BufferedOutputStream(
18             new FileOutputStream(path)));
19
20     encoder.writeObject(object);
21     encoder.close();
22 }
23
24● @SuppressWarnings("unchecked")
25 public static ArrayList<Bus_station> LongTermPersistenceRead(String path) throws FileNotFoundException {
26     XMLDecoder decoder = new XMLDecoder(
27         new BufferedInputStream(
28             new FileInputStream(path)));
29
30     ArrayList<Bus_station> object = (ArrayList<Bus_station>) decoder.readObject();
31     decoder.close();
32     return object;
33 }
34
35 }
36
37
```

Рисунок 3 – Реалізація моделі Long Term Persistence

3 ВАРІАНТИ ВИКОРИСТАННЯ

Програма дозволяє створювати об'єкти – “записи в розкладі”, що заносяться у запис каталогу, тобто створюється масив об'єктів. Користувач може додавати об'єкти до масиву, видаляти елементи вибірково, а також очистити увесь масив одним викликом відповідної кнопки меню. Також присутня можливість серіалізувати/десеріалізувати об'єкти з файлу.

```
0 - Exit
1 - Insert data
2 - Show data
3 - Add entry to schedule
4 - delete selected
5 - clear array
6 - save
7 - download
8 - files
Select:
2
Number of passage: 34
Number of sits: 20
Input working days: Monday
Time of departure: 12:30

Station and time:
Киев 14:15
Харьков 12:30
```

Рисунок 4 – Результати виведення масиву об'єктів

```

<?xml version="1.0" encoding="UTF-8"?>
- <java class="java.beans.XMLDecoder" version="1.8.0_181">
- <object class="java.util.ArrayList">
- <void method="add">
- <object class="ua.khpi.oop.kononenko08.Bus_station" id="Bus_station0">
- <void property="days">
  <string>Monday</string>
</void>
- <void property="fli_num">
  <int>34</int>
</void>
+ <void property="free_sits">
- <void property="time">
- <void property="hours">
  <int>12</int>
</void>
- <void property="minutes">
  <int>30</int>
</void>
</void>
</object>
</void>
</object>
- <void class="ua.khpi.oop.kononenko08.Station" id="Station0">
- <void property="time">
- <void property="hours">
  <int>12</int>
</void>
- <void property="minutes">
  <int>30</int>
</void>
</void>
- <void property="name">
  <string>Харьков</string>
</void>
</void>
- <void class="ua.khpi.oop.kononenko08.Station" id="Station1">
- <void property="time">
- <void property="hours">
  <int>14</int>
</void>
- <void property="minutes">
  <int>15</int>
</void>

```

Рисунок 5 – Зміст файлу *Test.xml*

ВИСНОВКИ

В даній лабораторній роботі розробив та реалізував класи та методи відповідно прикладної галузі, реалізував управління масивом domain-об'єктів, а також забезпечив та продемонстрував коректне відображення кирилиці.