Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Институт информационных технологий и управления в технических системах

Кафедра ИС

ОТЧЕТ

по лабораторной работе №4

ИССЛЕДОВАНИЕ ВОЗМОЖНОСТЕЙ РАЗРАБОТКИ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА В JAVA ПРИЛОЖЕНИЯХ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ SWING

Выполнил:

ст. гр. ИС/б-21-о

Куркчи А. Э.

Проверил:

Кузнецов А. С.

Севастополь

2016

1. ЦЕЛЬ РАБОТЫ

В ходе выполнения данной лабораторной работы необходимо ознакомиться с особенностями инструментария библиотеки SWING для создания графического интерфейса приложений на языке Java и приобрести практические навыки создания Java-программ с графическим интерфейсом, позволяющим пользователю осуществлять взаимодействие с программой: задавать исходные данные, просматривать результаты работы программы в удобном виде.

1. ПОСТАНОВКА ЗАДАЧИ

Необходимо создать Java приложение с графическим интерфейсом пользователя, реализующее добавление, редактирование, сортировку и удаление данных заданного по варианту типа информации T(см. табл. 4.1). Данные отображать в виде таблицы. Реализовать поля ввода для добавления и редактирования новых записей. Предусмотреть возможность загрузки информации из текстового файла и сохранения в текстовый файл.

При написании программы следует учесть следующие требования и рекомендации:

1. Создать публичный класс, представляющий заданный по варианту задания (см. табл. 4.1) тип информации (т.е. строку таблицы).
2. Создать модель данных таблицы. Для этого создать класс, расширяющий абстрактный класс AbstractTableModel. Создать в нем объект коллекции типа T, соответствующий варианту задания.

Переопределить методы:

* *public Class<?> getColumnClass(int columnIndex)*
* *public int getColumnCount()*
* *public String getColumnName(int columnIndex)*
* *public Object getValueAt(int rowIndex, int columnIndex)*
* *public boolean isCellEditable(int rowIndex, int columnIndex)*
* *public void setValueAt(Object value, int rowIndex, int columnIndex)*

Определить методы:

* *public void addRow(<объект>) – добавления элемента (строки)*
* *public void deleteRow(String Поле\_1) – удаления элемента по значению поля 1*
* *public void updateRow(int row, <объект>) – изменения элемента заданной строки)*

1. Для реализации окна приложения реализовать дочерний класс JFrame.
2. Для представления таблицы с данными использовать компонент класса JTable, разместив его в контейнере JScrollPane (для возможности добавления полос прокрутки).
3. Поля ввода для добавления и редактирования данных реализовать текстовыми компонентами JTextField. Каждое поле снабдить подписью при помощи компонентов JLabel.
4. Для выполнения действий открытия файла, добавления, изменения, удаления записи, сортировки и сохранения файла реализовать соответствующие кнопки, с использованием компонентов JButton и добавлением ActionListener. Реализовать загрузку записи в поля для редактирования при щелчке по строке таблицы.
5. Для выбора файла при открытии и сохранении использовать компонент JFileChooser.
6. Удаление и сортировка элементов должно проходить по ключевому полю P (см. табл. 4.1). Направление сортировки U(см. табл. 4.1).

|  |  |  |  |
| --- | --- | --- | --- |
| A | 2 | Возрастание | ArrayList |

Вариант № 6

A: Компакт диск (Название альбома, Исполнитель, Количество треков, Длительность звучания);

1. ТЕКСТ ПРОГРАММЫ

Файл Main.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab04**;**

**import** javax**.**swing**.\*;**

public class Main **{**

public static void main**(**String**[]** args**)** **{**

**try** **{**

UIManager**.**setLookAndFeel**(**UIManager**.**getSystemLookAndFeelClassName**());**

**}** **catch** **(**ClassNotFoundException **|** UnsupportedLookAndFeelException **|** IllegalAccessException **|** InstantiationException e**)** **{**

\_log**(**"Unexpected UIManager exception, L&F was not set\n"**);**

e**.**printStackTrace**();**

**}**

MainFrame mf **=** **new** MainFrame**();**

mf**.**setVisible**(true);**

**}**

private static void \_log**(**Object**...** args**)** **{**

StringBuilder sb **=** **new** StringBuilder**();**

**for(**Object str **:** args**)** **{**

sb**.**append**(**str**);**

**}**

System**.**out**.**print**(**sb**.**toString**());**

**}**

**}**

Файл CD.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab04**;**

**import** java**.**io**.**DataInputStream**;**

**import** java**.**io**.**DataOutputStream**;**

**import** java**.**io**.**IOException**;**

**import** java**.**util**.**Comparator**;**

public class CD **{**

protected String title**;**

protected String author**;**

protected int trackCount**;**

protected int duration**;**

public CD**(**String title**,** String author**,** int trackCount**,** int duration**)** **{**

setTitle**(**title**);**

setAuthor**(**author**);**

setTrackCount**(**trackCount**);**

setDuration**(**duration**);**

**}**

public String getTitle**()** **{**

**return** title**;**

**}**

public String getAuthor**()** **{**

**return** author**;**

**}**

public int getTrackCount**()** **{**

**return** trackCount**;**

**}**

public int getDuration**()** **{**

**return** duration**;**

**}**

public void setTitle**(**String title**)** **{**

**this.**title **=** title**;**

**}**

public void setAuthor**(**String author**)** **{**

**this.**author **=** author**;**

**}**

public void setTrackCount**(**int trackCount**)** **{**

**this.**trackCount **=** trackCount**;**

**}**

public void setDuration**(**int duration**)** **{**

**this.**duration **=** duration**;**

**}**

public void copyFrom**(**CD cd**)** **{**

setTitle**(**cd**.**getTitle**());**

setAuthor**(**cd**.**getAuthor**());**

setTrackCount**(**cd**.**getTrackCount**());**

setDuration**(**cd**.**getDuration**());**

**}**

public void write**(**DataOutputStream dos**)** **throws** IOException **{**

dos**.**writeUTF**(**getTitle**());**

dos**.**writeUTF**(**getAuthor**());**

dos**.**writeInt**(**getTrackCount**());**

dos**.**writeInt**(**getDuration**());**

**}**

public static CD read**(**DataInputStream dis**)** **throws** IOException **{**

**return** **new** CD**(**dis**.**readUTF**(),** dis**.**readUTF**(),** dis**.**readInt**(),** dis**.**readInt**());**

**}**

public static class Comp **implements** Comparator**<**CD**>** **{**

protected final int field**;**

public Comp**(**int field**)** **{**

**this.**field **=** field**;**

**}**

@Override

public int compare**(**CD cd1**,** CD cd2**)** **{**

**switch(**field**)** **{**

**case** 0**:**

**return** cd1**.**getTitle**().**compareTo**(**cd2**.**getTitle**());**

**case** 1**:**

**return** cd1**.**getAuthor**().**compareTo**(**cd2**.**getAuthor**());**

**case** 2**:**

**return** cd1**.**getTrackCount**()** **-** cd2**.**getTrackCount**();**

**case** 3**:**

**return** cd1**.**getDuration**()** **-** cd2**.**getDuration**();**

**}**

**return** 0**;**

**}**

**}**

**}**

Файл CDTableModel.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab04**;**

**import** javax**.**swing**.**table**.**AbstractTableModel**;**

**import** java**.**io**.\*;**

**import** java**.**util**.\*;**

public class CDTableModel **extends** AbstractTableModel **{**

protected static CDTableModel instance **=** **null;**

protected final List**<**CD**>** list**;**

protected int sortField **=** **-**1**;**

protected int sortOrder **=** **-**1**;**

protected final String**[]** names **=** **{**"Title"**,** "Author"**,** "Track count"**,** "Duration"**};**

public static CDTableModel getInstance**()** **{**

**return** instance **==** **null** **?** **(**instance **=** **new** CDTableModel**())** **:** instance**;**

**}**

protected CDTableModel**()** **{**

list **=** **new** ArrayList**<>();**

**}**

public void read**(**File file**)** **{**

sortField **=** **-**1**;**

list**.**clear**();**

**try(**DataInputStream dis **=** **new** DataInputStream**(new** FileInputStream**(**file**)))** **{**

int count **=** dis**.**readInt**();**

**for(**int i**=**0**;**i**<**count**;**i**++)** **{**

list**.**add**(**CD**.**read**(**dis**));**

**}**

**}** **catch** **(**FileNotFoundException e**)** **{**

System**.**out**.**println**(**"Input file not found"**);**

e**.**printStackTrace**();**

**}** **catch** **(**IOException e**)** **{**

e**.**printStackTrace**();**

**}**

fireTableDataChanged**();**

fireTableStructureChanged**();**

**}**

public void write**(**File file**)** **{**

**try(**DataOutputStream dos **=** **new** DataOutputStream**(new** FileOutputStream**(**file**)))** **{**

dos**.**writeInt**(**list**.**size**());**

**for(**CD cd **:** list**)** **{**

cd**.**write**(**dos**);**

**}**

**}** **catch** **(**FileNotFoundException e**)** **{**

System**.**out**.**println**(**"Output file not found"**);**

e**.**printStackTrace**();**

**}** **catch** **(**IOException e**)** **{**

e**.**printStackTrace**();**

**}**

**}**

public void sort**(**int j**)** **{**

Comparator**<**CD**>** comp **=** **new** CD**.**Comp**(**j**);**

**if(**sortField **==** j**)** **{**

sortOrder **=** 1 **-** sortOrder**;**

**}** **else** **{**

sortField **=** j**;**

sortOrder **=** 0**;**

**}**

**if(**sortOrder **==** 1**)** **{**

comp **=** comp**.**reversed**();**

**}**

Collections**.**sort**(**list**,**comp**);**

fireTableDataChanged**();**

fireTableStructureChanged**();**

**}**

public void addRow**(**CD cd**)** **{**

list**.**add**(**cd**);**

sortField **=** **-**1**;**

fireTableDataChanged**();**

fireTableStructureChanged**();**

**}**

public void deleteRow**(**int i**)** **{**

list**.**remove**(**i**);**

fireTableDataChanged**();**

**}**

public long deleteRow**(**String author**)** **{**

long count **=** list**.**stream**().**filter**(**cd **->** cd**.**getAuthor**().**equalsIgnoreCase**(**author**)).**count**();**

**if(**count **>** 0**)** **{**

list**.**removeIf**(**cd **->** cd**.**getAuthor**().**equals**(**author**));**

fireTableDataChanged**();**

**}**

**return** count**;**

**}**

public void updateRow**(**int i**,** CD cd**)** **{**

list**.**set**(**i**,**cd**);**

sortField **=** **-**1**;**

fireTableDataChanged**();**

fireTableStructureChanged**();**

**}**

@Override

public void setValueAt**(**Object value**,** int i**,** int j**)** **{**

CD cd **=** list**.**get**(**i**);**

**switch(**j**)** **{**

**case** 0**:**

cd**.**setTitle**(**String**.**valueOf**(**value**));**

**break;**

**case** 1**:**

cd**.**setAuthor**(**String**.**valueOf**(**value**));**

**break;**

**case** 2**:**

cd**.**setTrackCount**(**Integer**.**valueOf**(**String**.**valueOf**(**value**)));**

**break;**

**case** 3**:**

cd**.**setDuration**(**Integer**.**valueOf**(**String**.**valueOf**(**value**)));**

**break;**

**}**

**}**

@Override

public boolean isCellEditable**(**int i**,** int j**)** **{**

**return** **true;**

**}**

@Override

public String getColumnName**(**int j**)** **{**

**if(**sortField **==** j**)** **{**

**return** names**[**j**].**concat**(**" "**).**concat**(**sortOrder **==** 1 **?** "▲" **:** "▼"**);**

**}** **else** **{**

**return** names**[**j**];**

**}**

**}**

@Override

public Class**<?>** getColumnClass**(**int j**)** **{**

**switch(**j**)** **{**

**case** 0**:**

**return** String**.**class**;**

**case** 1**:**

**return** String**.**class**;**

**case** 2**:**

**return** Integer**.**class**;**

**case** 3**:**

**return** Integer**.**class**;**

**}**

**return** **null;**

**}**

@Override

public int getRowCount**()** **{**

**return** list**.**size**();**

**}**

@Override

public int getColumnCount**()** **{**

**return** 4**;**

**}**

@Override

public Object getValueAt**(**int i**,** int j**)** **{**

**if(**i **>=** list**.**size**())** **{**

**return** **null;**

**}**

CD cd **=** list**.**get**(**i**);**

**switch(**j**)** **{**

**case** 0**:**

**return** cd**.**getTitle**();**

**case** 1**:**

**return** cd**.**getAuthor**();**

**case** 2**:**

**return** cd**.**getTrackCount**();**

**case** 3**:**

**return** cd**.**getDuration**();**

**}**

**return** **null;**

**}**

**}**

Файл MainFrame.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab04**;**

**import** javax**.**swing**.\*;**

**import** java**.**awt**.**event**.**KeyAdapter**;**

**import** java**.**awt**.**event**.**KeyEvent**;**

**import** java**.**awt**.**event**.**MouseAdapter**;**

**import** java**.**awt**.**event**.**MouseEvent**;**

**import** java**.**io**.**File**;**

public class MainFrame **extends** JFrame **{**

private JPanel rootPanel**;**

private JButton addB**;**

private JButton deleteB**;**

private JButton loadB**;**

private JButton saveB**;**

private JScrollPane scrollPane**;**

private JTable dataTable**;**

private AddFrame addFrame**;**

private DeleteFrame deleteFrame**;**

public MainFrame**()** **{**

**super(**"SevSU IS-21 semester.04 Java Lab.04"**);**

setDefaultCloseOperation**(**EXIT\_ON\_CLOSE**);**

setContentPane**(**rootPanel**);**

setLocationByPlatform**(true);**

addFrame **=** **new** AddFrame**();**

deleteFrame **=** **new** DeleteFrame**(this);**

dataTable**.**setModel**(**CDTableModel**.**getInstance**());**

dataTable**.**setSelectionMode**(**ListSelectionModel**.**SINGLE\_SELECTION**);**

dataTable**.**getTableHeader**().**setReorderingAllowed**(false);**

addB**.**addActionListener**(**actionEvent **->** **{**

**if(!**addFrame**.**isVisible**())** **{**

addFrame**.**setLocationRelativeTo**(this);**

addFrame**.**clear**();**

addFrame**.**setVisible**(true);**

**}**

**});**

deleteB**.**addActionListener**(**actionEvent **->** **{**

**if(!**deleteFrame**.**isVisible**())** **{**

deleteFrame**.**setLocationRelativeTo**(this);**

deleteFrame**.**clear**();**

deleteFrame**.**setVisible**(true);**

**}**

**});**

loadB**.**addActionListener**(**actionEvent **->** **{**

JFileChooser fileChooser **=** **new** JFileChooser**();**

fileChooser**.**setCurrentDirectory**(new** File**(**System**.**getProperty**(**"user.dir"**)));**

int res **=** fileChooser**.**showDialog**(this,** "Load from file"**);**

**if(**res **==** JFileChooser**.**APPROVE\_OPTION**)** **{**

CDTableModel**.**getInstance**().**read**(**fileChooser**.**getSelectedFile**());**

**}**

**});**

saveB**.**addActionListener**(**actionEvent **->** **{**

JFileChooser fileChooser **=** **new** JFileChooser**();**

fileChooser**.**setCurrentDirectory**(new** File**(**System**.**getProperty**(**"user.dir"**)));**

int res **=** fileChooser**.**showDialog**(this,** "Save to file"**);**

**if(**res **==** JFileChooser**.**APPROVE\_OPTION**)** **{**

CDTableModel**.**getInstance**().**write**(**fileChooser**.**getSelectedFile**());**

**}**

**});**

dataTable**.**getTableHeader**().**addMouseListener**(new** MouseAdapter**()** **{**

@Override

public void mouseClicked**(**MouseEvent mouseEvent**)** **{**

CDTableModel**.**getInstance**().**sort**(**dataTable**.**columnAtPoint**(**mouseEvent**.**getPoint**()));**

**}**

**});**

dataTable**.**addKeyListener**(new** KeyAdapter**()** **{**

@Override

public void keyTyped**(**KeyEvent keyEvent**)** **{**

**if((**keyEvent**.**getKeyCode**()** **==** KeyEvent**.**VK\_DELETE**)** **&&** **(**dataTable**.**getSelectedRow**()** **>=** 0**))** **{**

CDTableModel**.**getInstance**().**deleteRow**(**dataTable**.**getSelectedRow**());**

**}**

**}**

**});**

pack**();**

**}**

public int getSelected**()** **{**

**return** dataTable**.**getSelectedRow**();**

**}**

**}**

Файл AddFrame.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab04**;**

**import** javax**.**swing**.\*;**

public class AddFrame **extends** JFrame **{**

private JTextField titleF**;**

private JTextField authorF**;**

private JSpinner trackCountF**;**

private JSpinner durationF**;**

private JButton addB**;**

private JButton cancelB**;**

private JPanel rootPanel**;**

public AddFrame**()** **{**

**super(**"Add form | SevSU IS-21 semester.04 Java Lab.04"**);**

setDefaultCloseOperation**(**DISPOSE\_ON\_CLOSE**);**

setContentPane**(**rootPanel**);**

setLocationByPlatform**(true);**

trackCountF**.**setModel**(new** SpinnerNumberModel**(**1**,**1**,**100**,**1**));**

durationF**.**setModel**(new** SpinnerNumberModel**(**0**,**0**,**Integer**.**MAX\_VALUE**,**1**));**

addB**.**addActionListener**(**actionEvent **->** **{**

**if(**titleF**.**getText**().**isEmpty**())** **{**

JOptionPane**.**showMessageDialog**(this,** "Title can not be empty"**,** "Error"**,** JOptionPane**.**ERROR\_MESSAGE**);**

**return;**

**}**

**if(**authorF**.**getText**().**isEmpty**())** **{**

JOptionPane**.**showMessageDialog**(this,** "Author can not be empty"**,** "Error"**,** JOptionPane**.**ERROR\_MESSAGE**);**

**return;**

**}**

CDTableModel**.**getInstance**().**addRow**(new** CD**(**titleF**.**getText**(),** authorF**.**getText**(),** **(**int**)** trackCountF**.**getValue**(),** **(**int**)** durationF**.**getValue**()));**

JOptionPane**.**showMessageDialog**(this,** "Element added"**,** "Success"**,** JOptionPane**.**INFORMATION\_MESSAGE**);**

setVisible**(false);**

**});**

cancelB**.**addActionListener**(**actionEvent **->** **{**

setVisible**(false);**

**});**

pack**();**

**}**

public void clear**()** **{**

titleF**.**setText**(**""**);**

authorF**.**setText**(**""**);**

trackCountF**.**setValue**(**1**);**

durationF**.**setValue**(**0**);**

**}**

**}**

Файл DeleteFrame.java

package ru**.**justnero**.**study**.**sevsu**.**java**.**lab04**;**

**import** javax**.**swing**.\*;**

public class DeleteFrame **extends** JFrame **{**

private JPanel rootPanel**;**

private JTextField authorF**;**

private JCheckBox bySelectionF**;**

private JButton deleteB**;**

private JButton cancelB**;**

public DeleteFrame**(**MainFrame mf**)** **{**

**super(**"Delete form | SevSU IS-21 semester.04 Java Lab.04"**);**

setDefaultCloseOperation**(**DISPOSE\_ON\_CLOSE**);**

setContentPane**(**rootPanel**);**

setLocationByPlatform**(true);**

deleteB**.**addActionListener**(**actionEvent **->** **{**

**if(**bySelectionF**.**isSelected**())** **{**

**if(**mf**.**getSelected**()** **>=** 0**)** **{**

CDTableModel**.**getInstance**().**deleteRow**(**mf**.**getSelected**());**

JOptionPane**.**showMessageDialog**(this,** "Deleted 1 element"**,** "Success"**,** JOptionPane**.**INFORMATION\_MESSAGE**);**

setVisible**(false);**

**}** **else** **{**

JOptionPane**.**showMessageDialog**(this,** "No element selected"**,** "Error"**,** JOptionPane**.**ERROR\_MESSAGE**);**

**}**

**}** **else** **{**

**if(!**authorF**.**getText**().**isEmpty**())** **{**

long count **=** CDTableModel**.**getInstance**().**deleteRow**(**authorF**.**getText**());**

**if(**count **>** 0**)** **{**

JOptionPane**.**showMessageDialog**(this,** "Deleted "**+**String**.**valueOf**(**count**)+**" element(s)"**,** "Success"**,** JOptionPane**.**INFORMATION\_MESSAGE**);**

setVisible**(false);**

**}** **else** **{**

JOptionPane**.**showMessageDialog**(this,** "No elements found"**,** "Error"**,** JOptionPane**.**ERROR\_MESSAGE**);**

**}**

**}**

**}**

**});**

cancelB**.**addActionListener**(**actionEvent **->** **{**

setVisible**(false);**

**});**

bySelectionF**.**addActionListener**(**actionEvent **->** **{**

authorF**.**setEnabled**(!**bySelectionF**.**isSelected**());**

**});**

pack**();**

**}**

public void clear**()** **{**

authorF**.**setText**(**""**);**

**}**

**}**

1. ИНТЕРФЕЙС ПРОГРАММЫ

Графический интерфейс программы разбит на 3 окна: основное окно отображения, окно добавления и окно удаления. Окна представлены на рисунке 1.

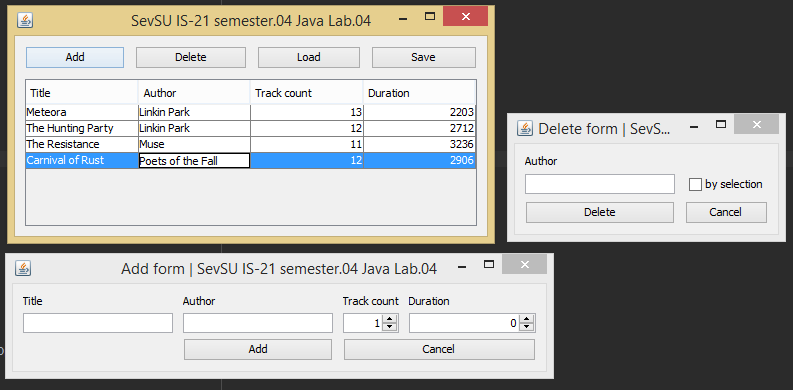


Рисунок 1 – Графический интерфейс программы

Вывод

В ходе выполнения данной лабораторной работы изучены особенности инструментария библиотеки SWING для создания графического интерфейса приложений на языке Java и приобретены практические навыки создания Java-программ с графическим интерфейсом, позволяющим пользователю осуществлять взаимодействие с программой: задавать исходные данные, просматривать результаты работы программы в удобном виде.