­­­МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» им.В.И.УЛЬЯНОВА (ЛЕНИНА)

Кафедра вычислительной техники

Отчет по лабораторной работе №2,3,4,5

по дисциплине «Объектно-ориентированное программирование»

Тема: «**Разработка интерфейса пользователя**»

|  |  |
| --- | --- |
| Студенты гр. 9310 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Брызгалова Е.А. |
|  |  |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_Павловский М.Г. |

Санкт-Петербург

2021

**Содержание**

|  |  |
| --- | --- |
| Цели работы | 3 |
| Описание экранной формы | 3 |
| Описание действий реализованными слушателями | 4 |
| Перечень ситуаций, которые контролируются с помощью исключений | 7 |
| Иллюстрация процесса загрузки данных из файла и сохранения данных в файл | 7 |
| Выводы | 9 |
| Список литературы | 9 |
| Документация JavaDoc | 10 |
| Код программы | 14 |

**Цели работы**

1. Знакомство с правилами построения экранной формы.
2. Обработка событий.
3. Обработка исключений.
4. Сохранение и загрузка данных из файла.

**Описание экранной формы**

Экранная форма (рис.1) состоит из трёх блоков: панель инструментов, таблица, панель для поиска.

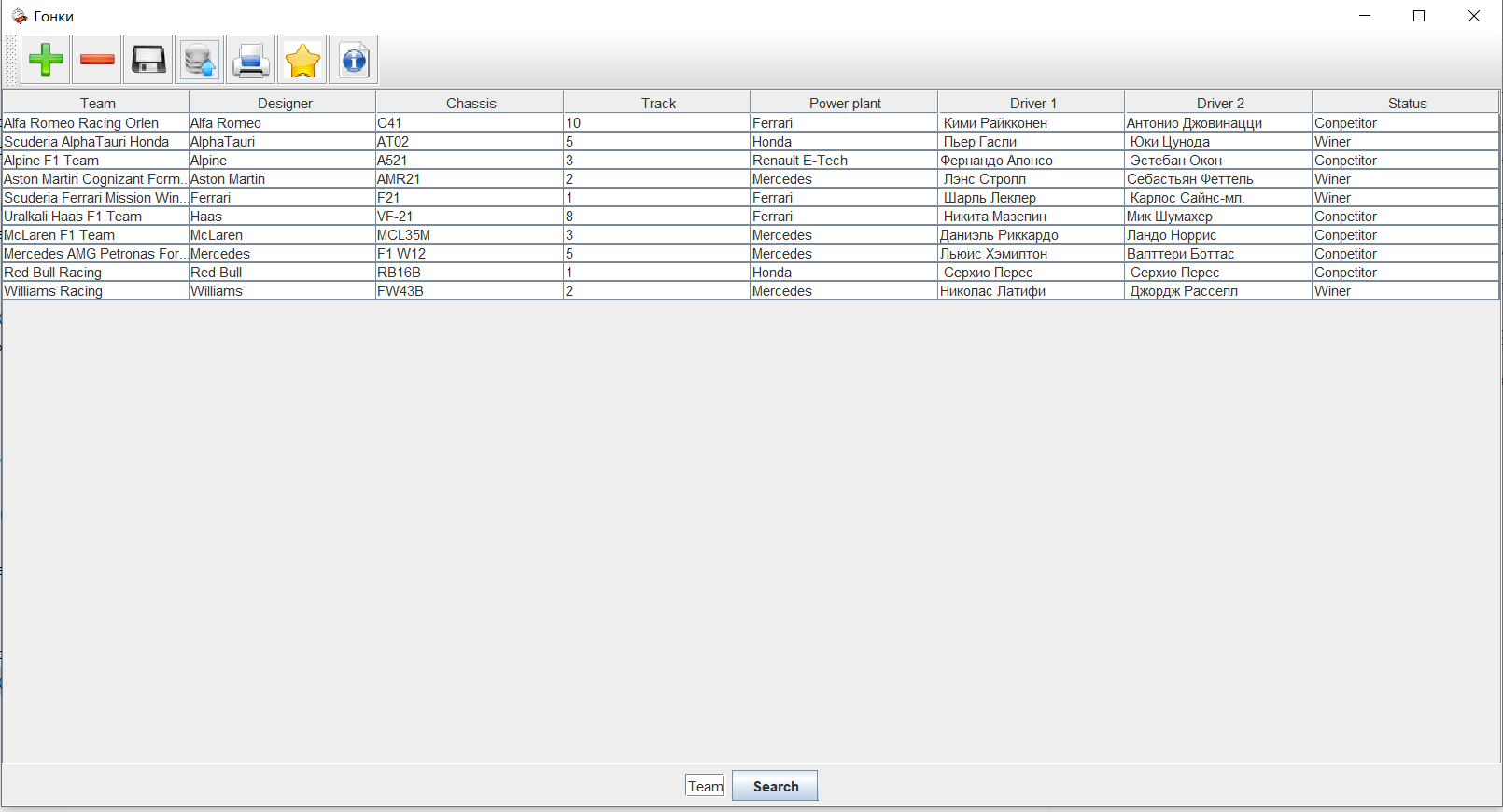


Рис.1 Экранная форма

Панель инструментов JToolBar (рис.2) размещена менеджером граничного размещения BorderLayout.NORTH в верху окна программы и состоит из 6 кнопок JButton («Сохранить», «Загрузка базы данных», «Добавить», «Удалить», «Печать», «Информация о программе»).

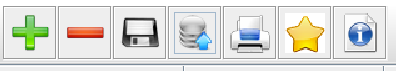


Рис.2 Панель инструментов

Таблица JTabel (рис.3) состоит из 9 столбцов («Id», «Страна», «Производитель», «Калибр», «Тип ствола», «Ёмкость магазина», «Вес», «Цена», «Наличие»). Таблица находится в центре экранной формы, размещена менеджером граничного размещения BorderLayout.CENTER.

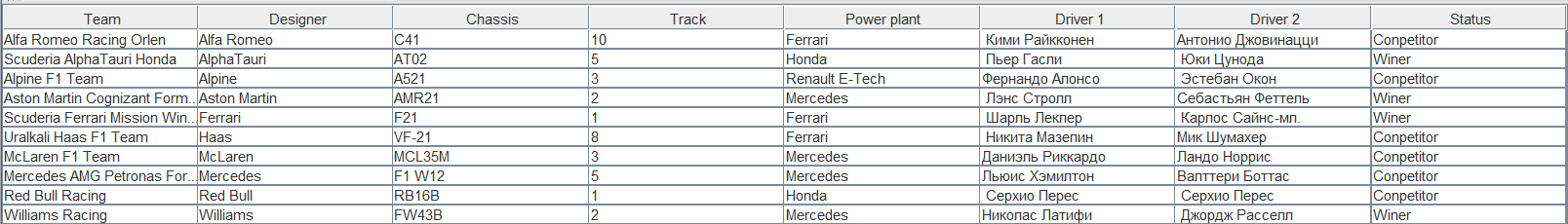


Рис.3 Таблица с данными

Панель поиска состоит из поля ввода JTextField и кнопки «Поиск JButton (Рис.4). Размещена менеджером граничного размещения BorderLayout.SOUTH.



Рис.4 Панель поиска

**Описание действий реализованными слушателями**

Источники событий save, load, add, delete, print, info, search, ok, cancel будут вызвать интерфейс слушателя ActionListener метода actionPerformed.

При нажатии кнопки save открывается диалоговое окно для ввода имени файла с сохранёнными данными (рис.5).

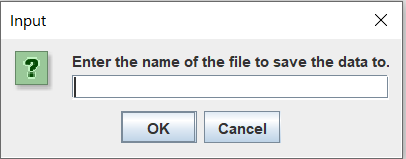


Рис.5 Окно для ввода имени файла

При нажатии кнопки load открывается диалоговое окно для ввода пути к файлу с подготовленными данными (рис.6).

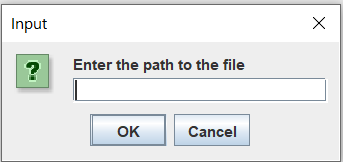


Рис.6 Окно для ввода пути к файлу

При нажатии кнопки add открывается диалоговое окно – форма, для заполнения данных новой позиции (рис.7).

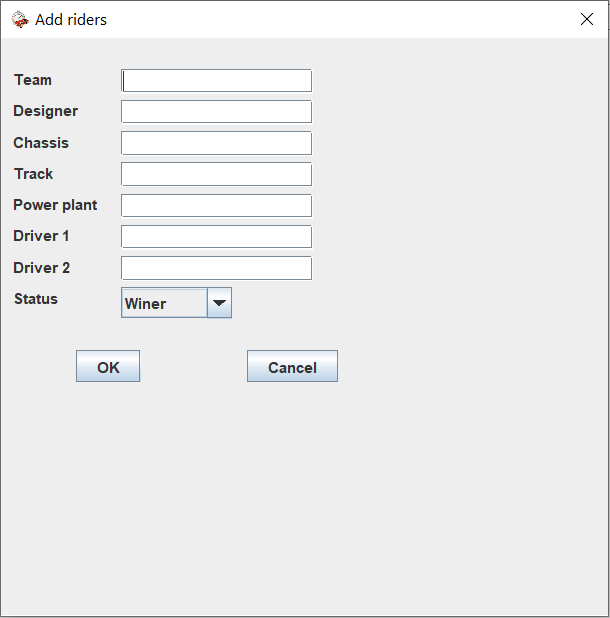


Рис.7 Форма для добавления новой позиции

При нажатии кнопки delete удаляется последняя позиция в таблице.

При нажатии кнопки print открывается окно с параметрами печати (рис.8).

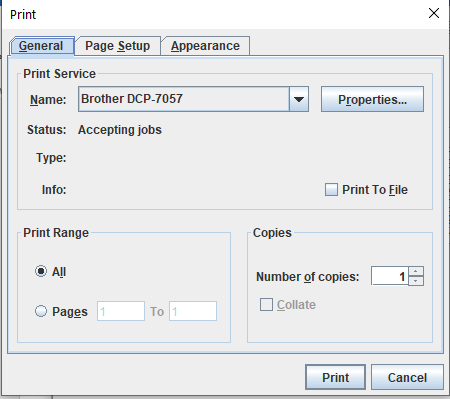


Рис.8 Меню печати

При нажатии кнопки info выводится краткая информация о программе (рис.9)

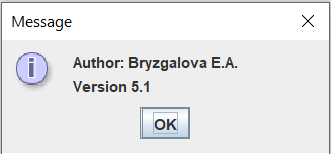


Рис.9 Краткая информация о программе

При вводе названия производителя в поле для ввода и нажатии кнопки search происходит вывод данных, найденных по введённому названию производителя (рис. 10).

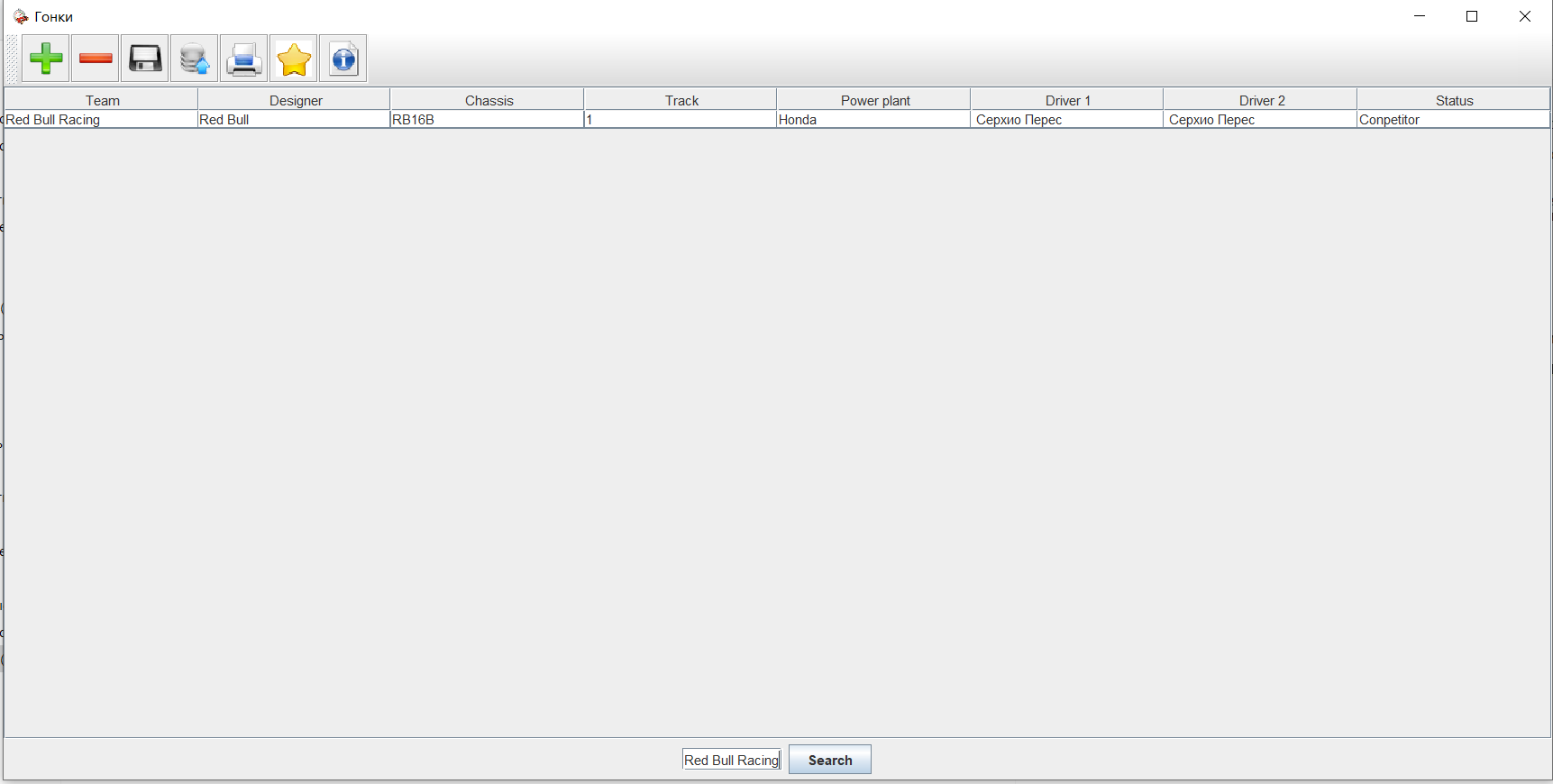


Рис. 10 Поиск

При нажатии кнопки «Звездочка» происходит вывод данных, содержащих в графе статус выражение “Winner”(рис. 11).

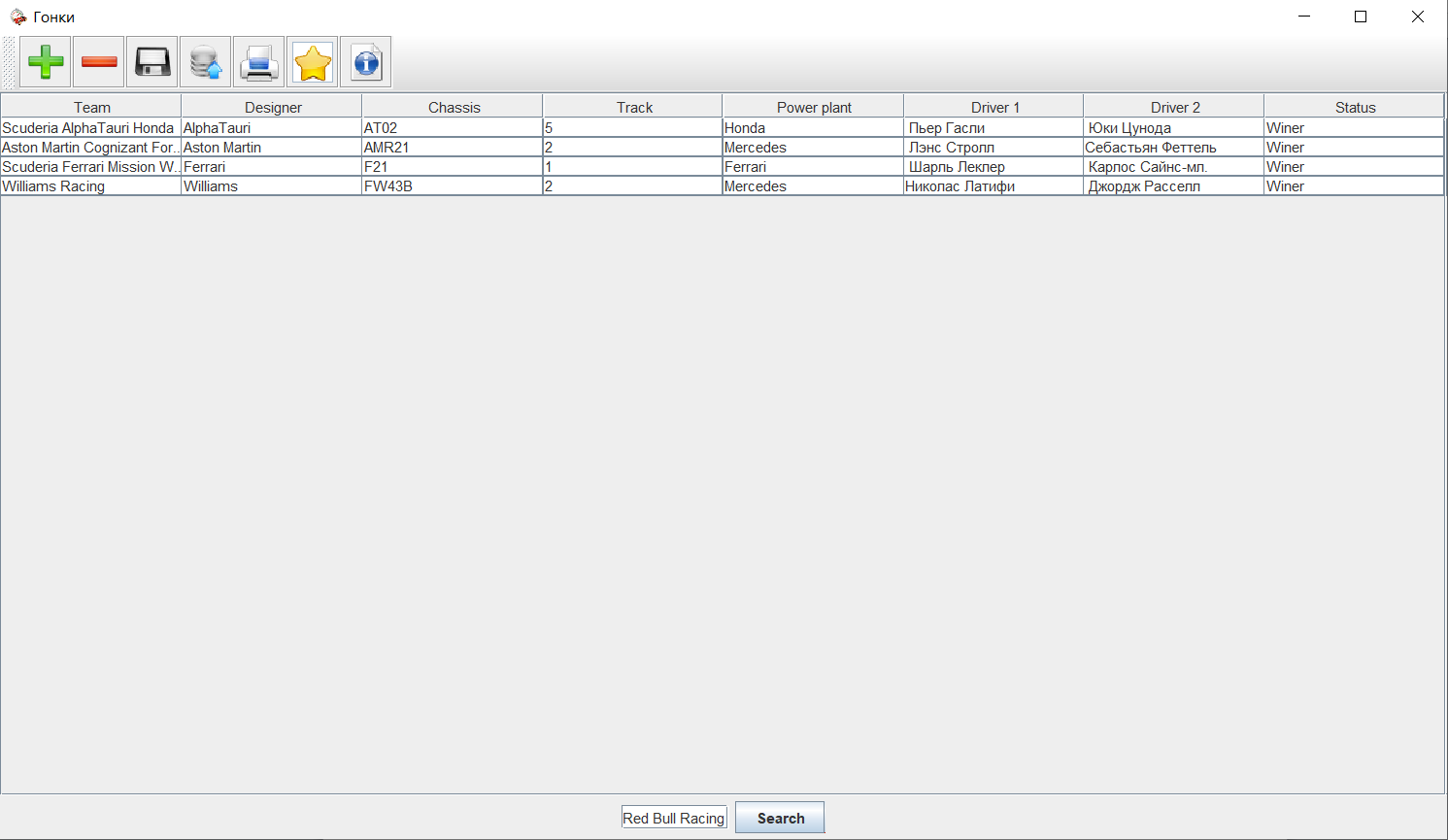


Рис. 11 Звездочка

**Перечень ситуаций, которые контролируются с помощью исключений**

* Если в поле для ввода пути файла, при загрузки данных из файла, срабатывает исключение IOException и выводится диалоговое окно с сообщением об ошибке (рис.12).

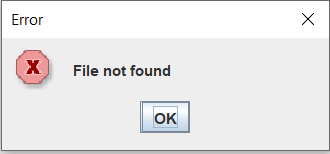


Рис.12

* Если пользователь пытается удалить несуществующую позицию, срабатывает исключение IndexOutOfBoundsException и выводится диалоговое окно с сообщением об ошибке (рис.13).

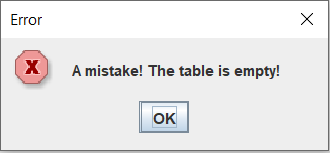
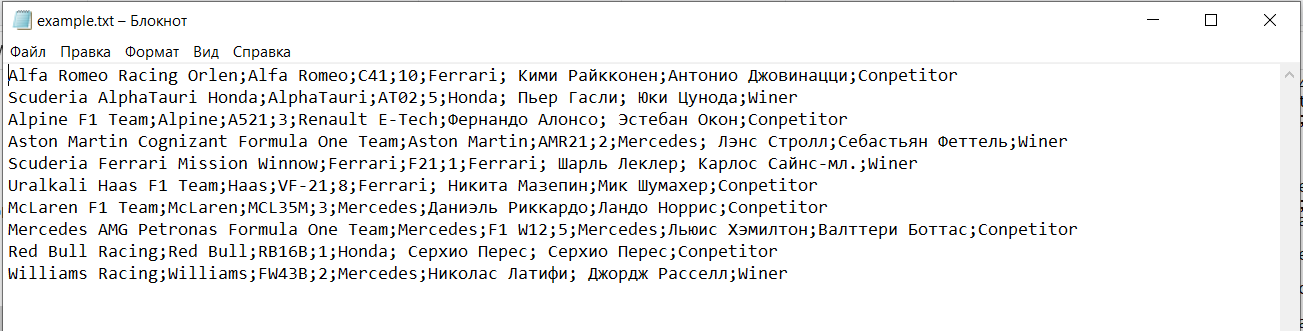


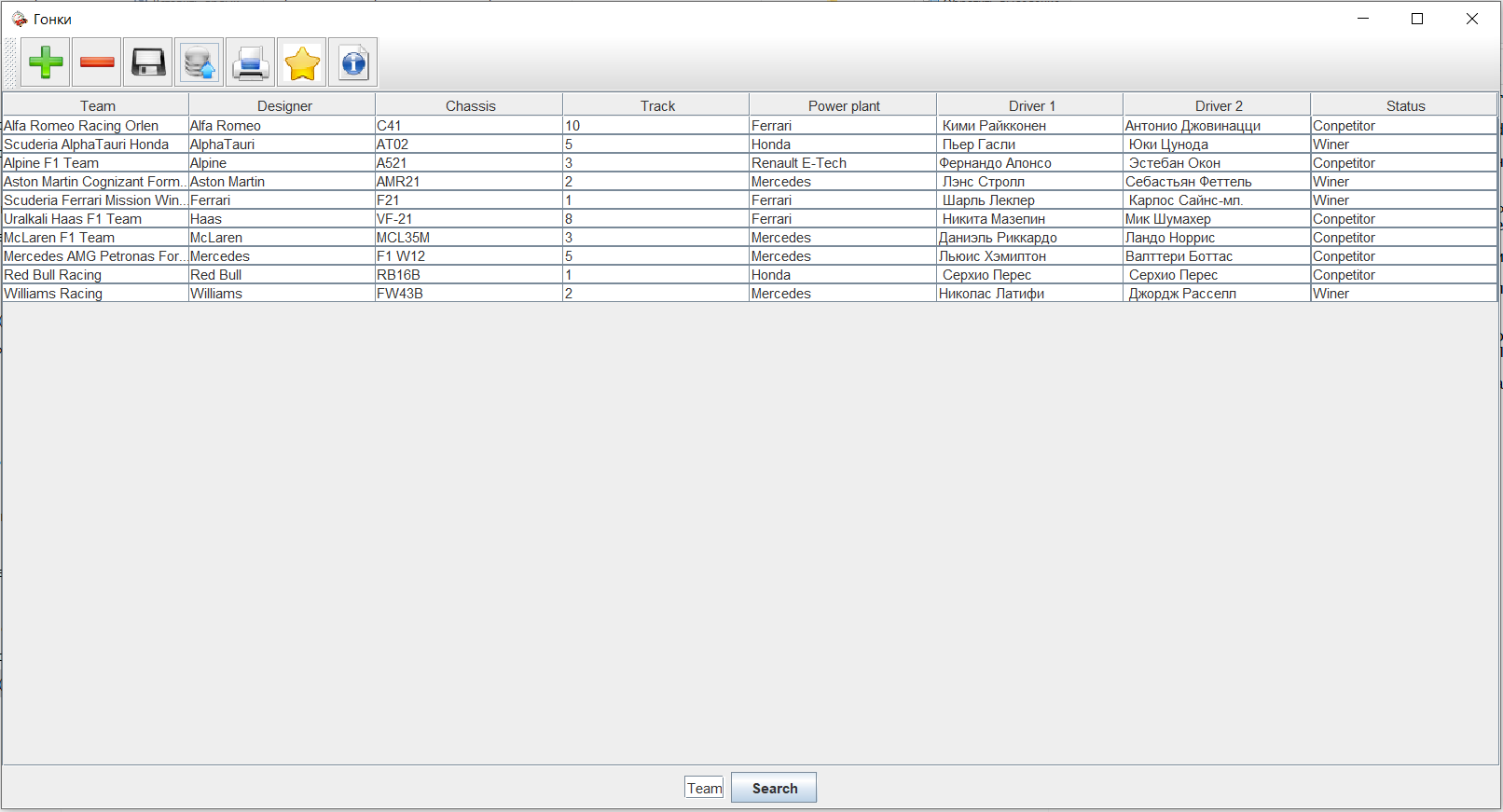
Рис.13

**Иллюстрация процесса загрузки данных из файла и сохранения данных в файл**

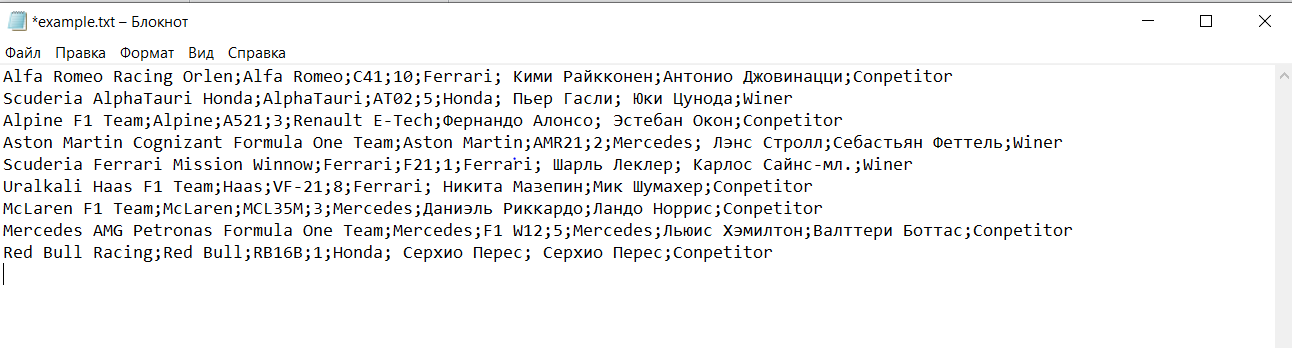
Загрузка из файла example.txt



После загрузки данных



После удаления последней строки и сохранения изменений



**Выводы**

В ходе выполнения лабораторных работ был создан интерфейс пользователя. Также были получены практические данные по созданию и разметки графических интерфейсов, загрузке данных из файла и сохранение данных в файл, обработке событий и исключений.

**Список использованной литературы**

Объектно-ориентированное программирование на языке Java: Методические указания к лабораторным работам / Сост.: С. А. Беляев, М. Г. Павловский, Г. В. Разумовский; Под общ. ред. Г. В. Разумовского. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2013. 63 с.

**Документация JavaDoc**

**Package** [edu.java.lab1](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\package-summary.html)

# Class First\_lab

java.lang.Object

edu.java.lab1.First\_lab

public class **First\_lab**

extends java.lang.Object

**Version:**

5.1

**Author:**

Bryzgalova Ekaterina 9307

## *Constructor Summary*

| **Constructors** | |
| --- | --- |
| **Constructor** | **Description** |
| [**First\_lab**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\First_lab.html#%3Cinit%3E())() |  |

## *Method Summary*

All MethodsStatic MethodsInstance MethodsConcrete Methods

| **Modifier and Type** | **Method** | **Description** |
| --- | --- | --- |
| static void | [**main**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\First_lab.html#main(java.lang.String%5B%5D))​(java.lang.String[] args) |  |
| void | [**show**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\First_lab.html#show())() |  |

### Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## *Constructor Details*

### First\_lab

public First\_lab()

## *Method Details*

### show

public void show()

### main

public static void main​(java.lang.String[] args)

**Package** [edu.java.lab1](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\package-summary.html)

# Class Data

java.lang.Object

edu.java.lab1.Data

public class **Data**

extends java.lang.Object

Class for work to data

## *Constructor Summary*

| **Constructors** | |
| --- | --- |
| **Constructor** | **Description** |
| [**Data**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\Data.html#%3Cinit%3E())() |  |

## *Method Summary*

All MethodsInstance MethodsConcrete Methods

| **Modifier and Type** | **Method** | **Description** |
| --- | --- | --- |
| void | [**Add**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\Data.html#Add(javax.swing.table.DefaultTableModel,javax.swing.JFrame))​(javax.swing.table.DefaultTableModel model, javax.swing.JFrame frame) | Method for add new position in table |
| void | [**Delete**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\Data.html#Delete(javax.swing.table.DefaultTableModel,javax.swing.JFrame))​(javax.swing.table.DefaultTableModel model, javax.swing.JFrame frame) | method for delete position |
| void | [**LoadFile**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\Data.html#LoadFile(javax.swing.JFrame,java.util.Vector))​(javax.swing.JFrame frame, java.util.Vector<java.util.Vector<java.lang.String>> data) | method for load data from file |
| void | [**SaveFile**](file:///C:\Eclips\work_space\First_lab\doc\edu\java\lab1\Data.html#SaveFile(javax.swing.table.DefaultTableModel))​(javax.swing.table.DefaultTableModel model) | method for save data in file |

### Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## *Constructor Details*

### Data

public Data()

## *Method Details*

### Add

* + public void Add​(javax.swing.table.DefaultTableModel model,

javax.swing.JFrame frame)

Method for add new position in table

**Parameters:**

model - - model of table

frame - - object type of window

### Delete

* + public void Delete​(javax.swing.table.DefaultTableModel model,

javax.swing.JFrame frame)

method for delete position

**Parameters:**

model - - model of table

frame - - object type of window

### LoadFile

* + public void LoadFile​(javax.swing.JFrame frame,

java.util.Vector<java.util.Vector<java.lang.String>> data)

method for load data from file

**Parameters:**

frame - - object type of window

data - - data table

### SaveFile

public void SaveFile​(javax.swing.table.DefaultTableModel model)

method for save data in file

**Parameters:**

model - - model of table

**Код программы**

**Класс Data**

package edu.java.lab1;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import java.nio.file.Files;

import java.nio.file.Paths;

import java.util.Vector;

import javax.swing.JOptionPane;

import static javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE;

/\*\*

\* Class for work to data

\*/

public class Data {

/\*\*

\* Method for add new position in table

\* @param model - model of table

\* @param frame - object type of window

\*/

public void Add(DefaultTableModel model,JFrame frame) {

JDialog window = new JDialog(frame,"Add riders");

window.setModal(true);

window.setSize(500,500);

window.setLocation(300,300);

Container contentPane = window.getContentPane();

SpringLayout layout = new SpringLayout();

contentPane.setLayout(layout);

Component Team = new JLabel("Team");

JTextField tTeam = new JTextField(15);

contentPane.add(Team);

contentPane.add(tTeam);

layout.putConstraint(SpringLayout.WEST , Team, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Team, 25,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, tTeam, 25,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , tTeam, 55,

SpringLayout.EAST , Team);

Component Designer = new JLabel("Designer");

JTextField tDesigner = new JTextField(15);

layout.putConstraint(SpringLayout.WEST , Designer, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Designer, 50,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, tDesigner, 50,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , tDesigner, 34,

SpringLayout.EAST , Designer);

contentPane.add(Designer);

contentPane.add(tDesigner);

Component Chassis = new JLabel("Chassis");

JTextField tChassis = new JTextField(15);

layout.putConstraint(SpringLayout.WEST , Chassis, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Chassis, 75,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, tChassis, 75,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , tChassis, 41,

SpringLayout.EAST ,Chassis);

contentPane.add(Chassis);

contentPane.add(tChassis);

Component Track = new JLabel("Track");

JTextField tTrack = new JTextField(15);

layout.putConstraint(SpringLayout.WEST , Track, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Track, 100,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, tTrack, 100,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , tTrack, 54,

SpringLayout.EAST , Track );

contentPane.add(Track);

contentPane.add(tTrack);

Component Power\_plant = new JLabel("Power plant");

JTextField tPower\_plant = new JTextField(15);

layout.putConstraint(SpringLayout.WEST , Power\_plant, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Power\_plant, 125,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, tPower\_plant, 125,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , tPower\_plant, 19,

SpringLayout.EAST , Power\_plant );

contentPane.add(Power\_plant);

contentPane.add(tPower\_plant);

Component Driver\_1 = new JLabel("Driver 1");

JTextField tDriver\_1 = new JTextField(15);

layout.putConstraint(SpringLayout.WEST , Driver\_1, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Driver\_1, 150,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, tDriver\_1, 150,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , tDriver\_1, 40,

SpringLayout.EAST ,Driver\_1);

contentPane.add(Driver\_1);

contentPane.add(tDriver\_1);

Component Driver\_2 = new JLabel("Driver 2");

JTextField tDriver\_2 = new JTextField(15);

layout.putConstraint(SpringLayout.WEST , Driver\_2, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Driver\_2, 175,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, tDriver\_2, 175,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , tDriver\_2, 40,

SpringLayout.EAST , Driver\_2);

contentPane.add(Driver\_2);

contentPane.add( tDriver\_2 );

Component Status = new JLabel("Status");

String S[] = {"Winer","Conpetitor"};

JComboBox comboAvailability = new JComboBox<>(S);

layout.putConstraint(SpringLayout.WEST ,Status, 10,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, Status, 200,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, comboAvailability, 200,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , comboAvailability, 50,

SpringLayout.EAST , Status);

contentPane.add(Status);

contentPane.add(comboAvailability);

JButton ok = new JButton("OK");

JButton cancel = new JButton("Cancel");

layout.putConstraint(SpringLayout.WEST , ok, 60,

SpringLayout.WEST , contentPane);

layout.putConstraint(SpringLayout.NORTH, ok, 250,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.NORTH, cancel, 250,

SpringLayout.NORTH, contentPane);

layout.putConstraint(SpringLayout.WEST , cancel, 85,

SpringLayout.EAST , ok );

contentPane.add(ok);

contentPane.add(cancel);

ok.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

Vector <String> s = new Vector<>();

s.add(tTeam.getText());

s.add(tDesigner.getText());

s.add(tChassis.getText());

s.add(tTrack.getText());

s.add(tPower\_plant.getText());

s.add(tDriver\_1.getText());

s.add(tDriver\_2.getText());

s.add(S[comboAvailability.getSelectedIndex()]);

int count = 0;

for(String s1:s) {

if((s1.length() > 0) && (!s1.equals(" "))) {

count++;

}

}

if(count == 8) {

model.addRow(s);

window.dispose();

}

else {

JOptionPane.showMessageDialog (frame,"Not all fields are filled in!");

}

}

});

cancel.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int n = JOptionPane.showConfirmDialog(window,"Are you sure you want to close the window?","Message",JOptionPane.YES\_NO\_OPTION);

if(n == JOptionPane.YES\_OPTION) {

window.dispose();

}

}

});

window.setDefaultCloseOperation(DISPOSE\_ON\_CLOSE);

window.setVisible(true);

}

/\*\*

\* method for delete position

\* @param model - model of table

\* @param frame - object type of window

\*/

public void Delete(DefaultTableModel model, JFrame frame) {

try {

model.removeRow(model.getRowCount()-1);

}

catch (IndexOutOfBoundsException ev) {

JOptionPane.showMessageDialog (frame,"A mistake! The table is empty!","Error", JOptionPane.ERROR\_MESSAGE);

}

}

/\*\*

\* method for load data from file

\* @param frame - object type of window

\* @param data - data table

\*/

public void LoadFile(JFrame frame,Vector <Vector<String>> data) {

String filePath = JOptionPane.showInputDialog("Enter the path to the file");

try (BufferedReader reader = Files.newBufferedReader(Paths.get(filePath))) {

data.clear();

String line;

String s[];

while ((line = reader.readLine()) != null) {

s = line.split(";");

Vector <String> row = new Vector<>();

for(int i = 0;i < s.length;i++) {

row.add(s[i]);

}

data.add(row);

}

} catch (IOException ev) {

JOptionPane.showMessageDialog(frame,"File not found","Error", JOptionPane.ERROR\_MESSAGE);

}

}

/\*\*

\* method for save data in file

\* @param model - model of table

\*/

public void SaveFile(DefaultTableModel model) {

String fileName = JOptionPane.showInputDialog("Enter the name of the file to save the data to.");

try {

BufferedWriter writer = new BufferedWriter(new FileWriter(fileName));

for(int i = 0;i < model.getRowCount();i++) {

for(int j = 0;j < model.getColumnCount();j++) {

if(j != model.getColumnCount()-1) {

writer.write(model.getValueAt(i,j) + ";");

}

else {

writer.write((String)model.getValueAt(i,j));

}

}

writer.write("\n");

}

writer.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Класс Fist\_lab**

package edu.java.lab1;

// Подключение графических библиотек

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Vector;

import javax.swing.JOptionPane;

import java.awt.print.PrinterException;

import java.text.MessageFormat;

import javax.swing.table.TableModel;

import javax.swing.table.TableRowSorter;

/\*\*

\* @author Bryzgalova Ekaterina 9307

\* @version 5.1

\*/

public class First\_lab {

// Объявления графических компонентов

private JFrame bookList;

private DefaultTableModel model;

private JButton delete;

private JButton load;

private JButton save;

private JButton add;

private JButton print;

private JButton info;

private JButton winners;

private JToolBar toolBar;

private JTable books;

private JTextField Name;

private JButton filter;

//Создание окна

public void show() {

// Создание окна

bookList = new JFrame("Гонки");

bookList.setSize(1300, 700);

bookList.setLocation(100, 100);

ImageIcon icon = new ImageIcon("./pictures\_OOP/car1.png");

bookList.setIconImage(icon.getImage());

bookList.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// Создание кнопок и прикрепление иконок

add = new JButton(new ImageIcon("./pictures\_OOP/add.png"));

save = new JButton(new ImageIcon("./pictures\_OOP/save.png"));

load = new JButton(new ImageIcon("./pictures\_OOP/load.png"));

print = new JButton(new ImageIcon("./pictures\_OOP/print.png"));

delete = new JButton(new ImageIcon("./pictures\_OOP/delete.png"));

info = new JButton(new ImageIcon("./pictures\_OOP/info.png"));

winners=new JButton(new ImageIcon("./pictures\_OOP/rating.png"));

filter = new JButton("Search");

// Настройка подсказок для кнопок

add.setToolTipText("Edd position");

load.setToolTipText("Download the list of riders");

save.setToolTipText("Save the list of riders");

print.setToolTipText("Print");

winners.setToolTipText("List of winners");

delete.setToolTipText("Delete a position");

info.setToolTipText("Program information");

// Добавление кнопок на панель инструментов

toolBar = new JToolBar("Toolbar");

toolBar.add(add);

toolBar.add(delete);

toolBar.add(save);

toolBar.add(load);

toolBar.add(print);

toolBar.add(winners);

toolBar.add(info);

// Размещение панели инструментов

bookList.setLayout(new BorderLayout());

bookList.add(toolBar, BorderLayout.NORTH);

// Создание таблицы с данными

final String [] column = {"Team", "Designer", "Chassis","Track","Power plant","Driver 1", "Driver 2","Status"};

Vector<String> columns = new Vector<>();

for(int i = 0;i < column.length;i++) {

columns.add(column[i]);

}

Vector <Vector<String>> data = new Vector<>();

model= new DefaultTableModel(data, columns);

books = new JTable(model);

final TableRowSorter <TableModel> sorter = new TableRowSorter<>(model);

books.setRowSorter(sorter);

JScrollPane scroll = new JScrollPane(books);

scroll = new JScrollPane(books);

Data d = new Data();

// Обработка нажатия кнопки загрузки базы данных

load.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

d.LoadFile(bookList,data);

model.setDataVector(data,columns);

}

});

// Обработка нажатия кнопки поиск

filter.addActionListener (new ActionListener()

{

public void actionPerformed (ActionEvent event)

{

String text =Name.getText();

if (text.length() == 0) {

sorter.setRowFilter(null);

} else {

sorter.setRowFilter(RowFilter.regexFilter(text));

}

}

});

winners.addActionListener (new ActionListener()

{

int f=1;

public void actionPerformed (ActionEvent event)

{

if(f==1) {

sorter.setRowFilter(RowFilter.regexFilter("Winer"));

f=0;

}

else {

sorter.setRowFilter(null);

f=1;

}

}

});

// Обработка нажатия кнопки информация о программе

info.addActionListener (new ActionListener()

{

public void actionPerformed (ActionEvent event)

{

JOptionPane.showMessageDialog (bookList, "Author: Bryzgalova E.A.\nVersion 5.1");

}

});

// Обработка нажатия кнопки добавления новой позиции

add.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

d.Add(model,bookList);

}

});

// Обработка нажатия кнопки удаление

delete.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

d.Delete(model,bookList);

}

});

//Обработка нажатия кнопки сохранения

save.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

d.SaveFile(model);

}

});

print.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

try {

MessageFormat headerFormat = new MessageFormat("Page {0}");

MessageFormat footerFormat = new MessageFormat("- {0} -");

books.print(JTable.PrintMode.FIT\_WIDTH, headerFormat, footerFormat);

} catch (PrinterException pe) {

System.err.println("Error printing: " + pe.getMessage());

}

}

});

/////////////////////////////////////////////////////////////////////////

// Размещение таблицы с данными

bookList.add(scroll, BorderLayout.CENTER);

// Подготовка компонентов поиска

Name = new JTextField("Team");

// Добавление компонентов на панель

JPanel filterPanel = new JPanel();

filterPanel.add(Name);

filterPanel.add(filter);

// Размещение панели поиска внизу окна

bookList.add(filterPanel, BorderLayout.SOUTH);

books.setEnabled(false);

// Визуализация экранной формы

bookList.setVisible(true);

}

public static void main(String[] args) {

// Создание и отображение экранной формы

new First\_lab().show();

}

}