**Министерство образования и науки Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ бюджетное образовательное учреждение высшего профессионального образования**

**«Санкт-Петербургский государственный электротехнический университет**

**“ЛЭТИ” им.В.И.Ульянова (Ленина)»**

**КАФЕДРА МОЭВМ**

**ОТЧЕТ**

**по лабораторно-практической работе № 3**

**«Обработка событий»**

**по дисциплине «Объектно - ориентированное программирование на языке Java»**

Выполнил: Лебедев И.А.

Факультет: КТИ

Группа: №3312

Подпись преподавателя: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2024

**Содержание**

[Цель работы 3](#_Toc180177721)

[Описание действий, которые должны реализовать слушатели 3](#_Toc180177722)

[Скриншоты, иллюстрирующие работу слушателей 4](#_Toc180177723)

[Текст программы 7](#_Toc180177724)

[Приложение 10](#_Toc180177725)

# Цель работы

Знакомство со способами подключения слушателей со-бытий к графическим компонентам пользовательского интерфейса.

# Описание действий, которые должны реализовать слушатели

*1. Слушатель для кнопки "Добавить":* При нажатии на кнопку "Добавить" в таблицу добавляется новая запись. Данные новой записи: "Новая собака", "Неизвестная порода", "Новый владелец", "Новый судья", "Нет наград". После добавления выводится сообщение "Добавлена новая собака". Внутри программы устанавливается флаг unsavedChanges в значение true, что указывает на наличие несохраненных изменений.

*2. Слушатель для кнопки "Изменить":* При нажатии на кнопку "Изменить" обновляется информация о выбранной записи в таблице. После этого пользователю выводится сообщение "Изменена информация о собаке". Также устанавливается флаг unsavedChanges в значение true.

*3. Слушатель для кнопки "Удалить":* При нажатии на кнопку "Удалить" программа проверяет, выбрана ли строка в таблице. Если строка выбрана, пользователю предлагается подтвердить удаление выбранной записи. Если пользователь соглашается, строка удаляется из таблицы, и выводится сообщение "Запись удалена". Флаг unsavedChanges также устанавливается в true. Если строка не выбрана, выводится сообщение "Пожалуйста, выберите строку для удаления".

*4. Слушатель для кнопки "Сохранить":* При нажатии на кнопку "Сохранить" данные приложения сохраняются, и пользователю выводится сообщение "Данные сохранены". После сохранения флаг unsavedChanges сбрасывается в значение false.

*5. Слушатель для закрытия окна:* При попытке закрытия окна программа проверяет наличие несохраненных изменений. Если изменения были внесены, пользователю предлагается сохранить данные. Если пользователь соглашается на сохранение, данные сохраняются, и программа закрывается. В противном случае окно закрывается без сохранения. Если изменений не было, окно закрывается сразу без дополнительных сообщений.

# Скриншоты, иллюстрирующие работу слушателей

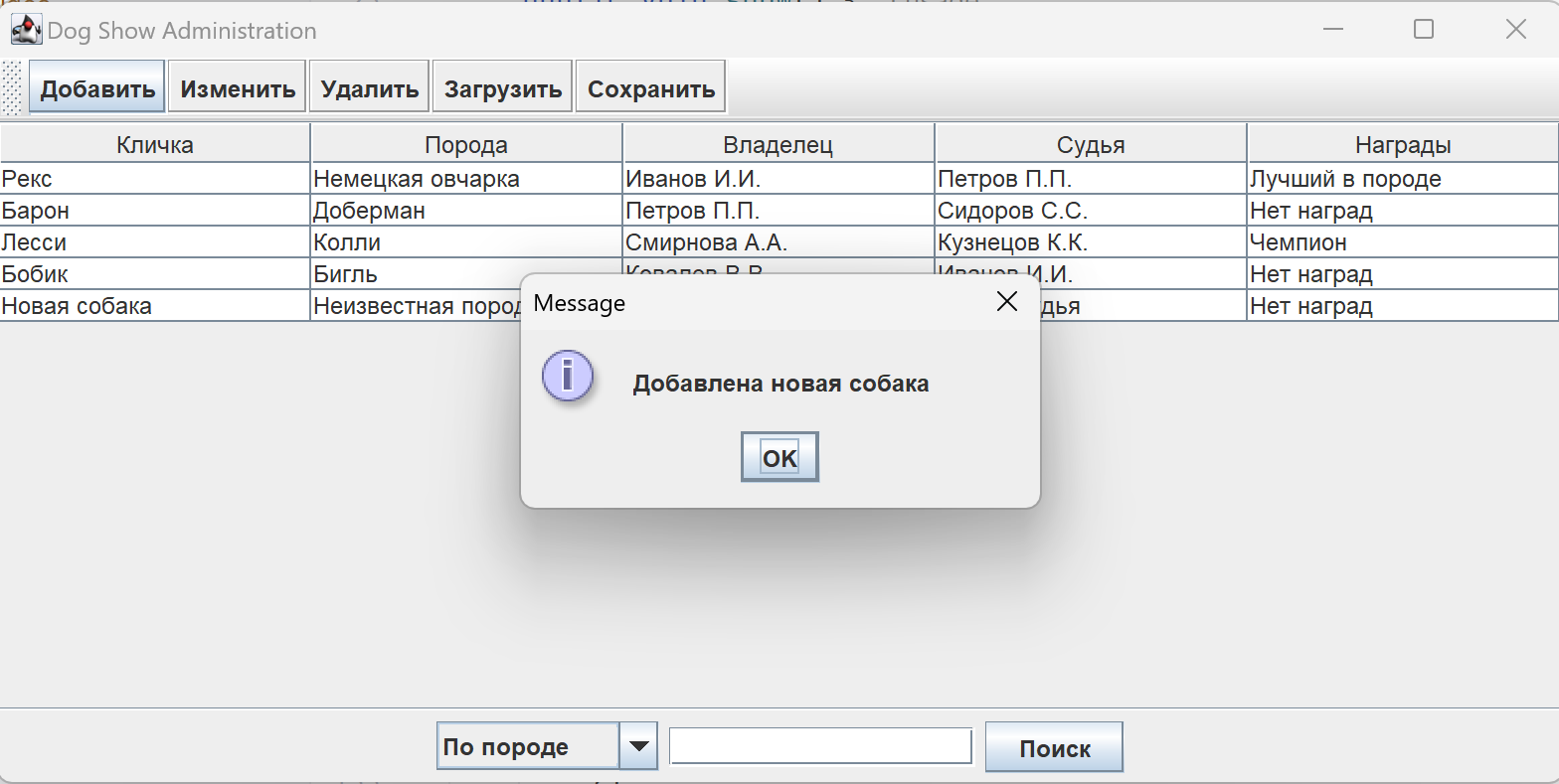


Рисунок 1 – Слушатель для кнопки "Добавить"

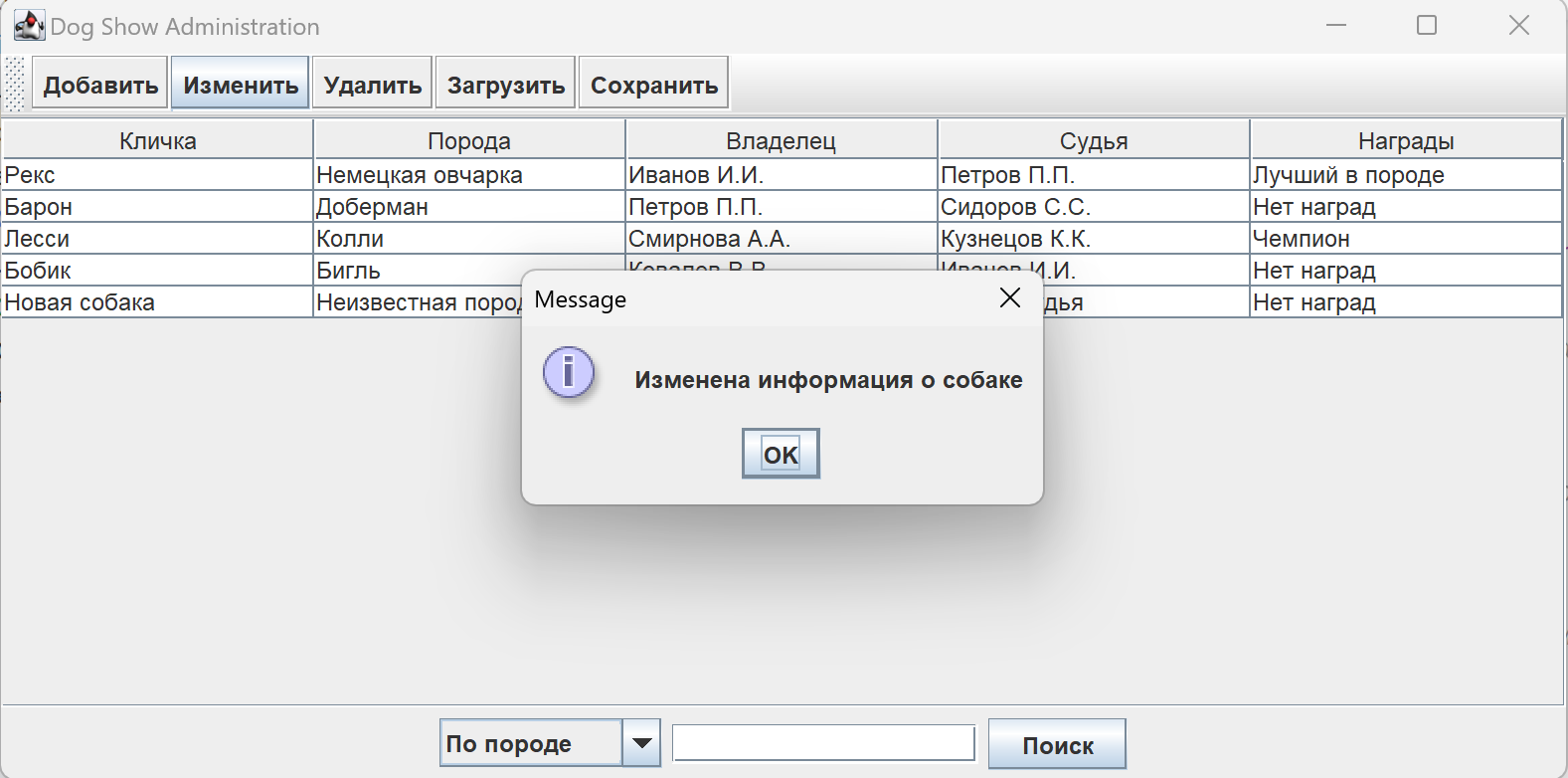


Рисунок 2 – Слушатель для кнопки "Изменить"

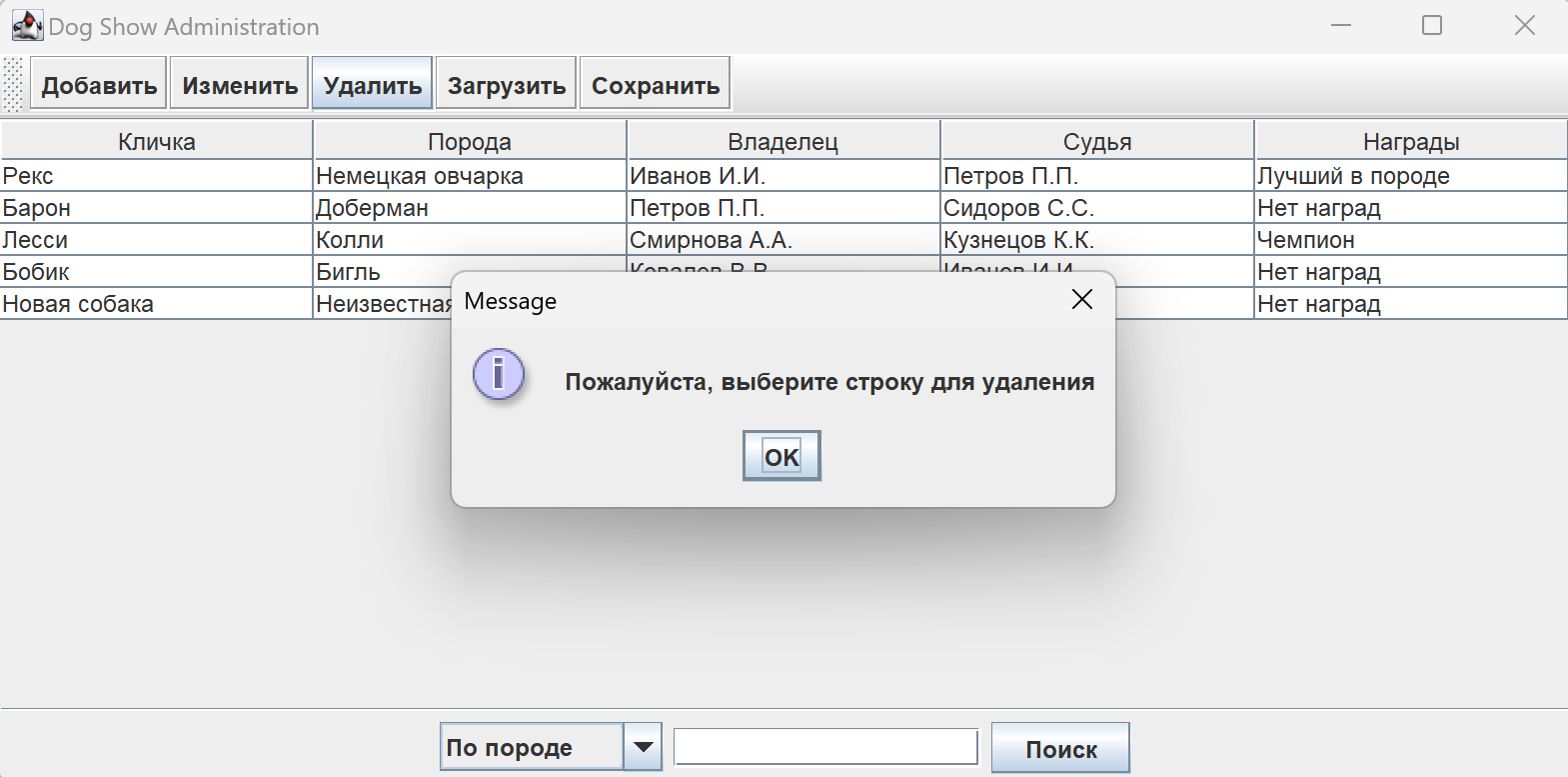


Рисунок 3 – Слушатель для кнопки "Удалить" без выбора строки

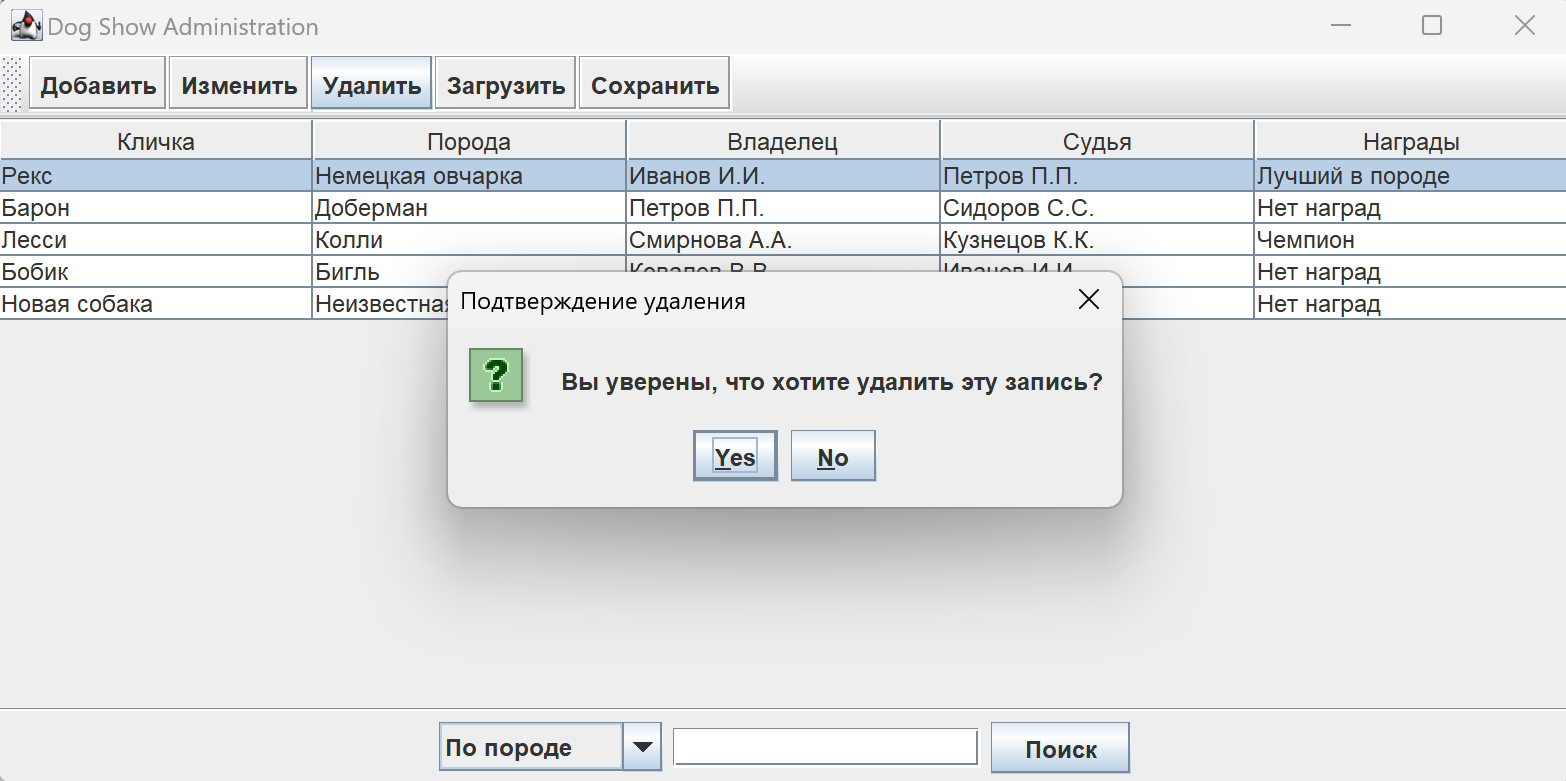


Рисунок 4 – Слушатель для кнопки "Удалить" с выбором строки

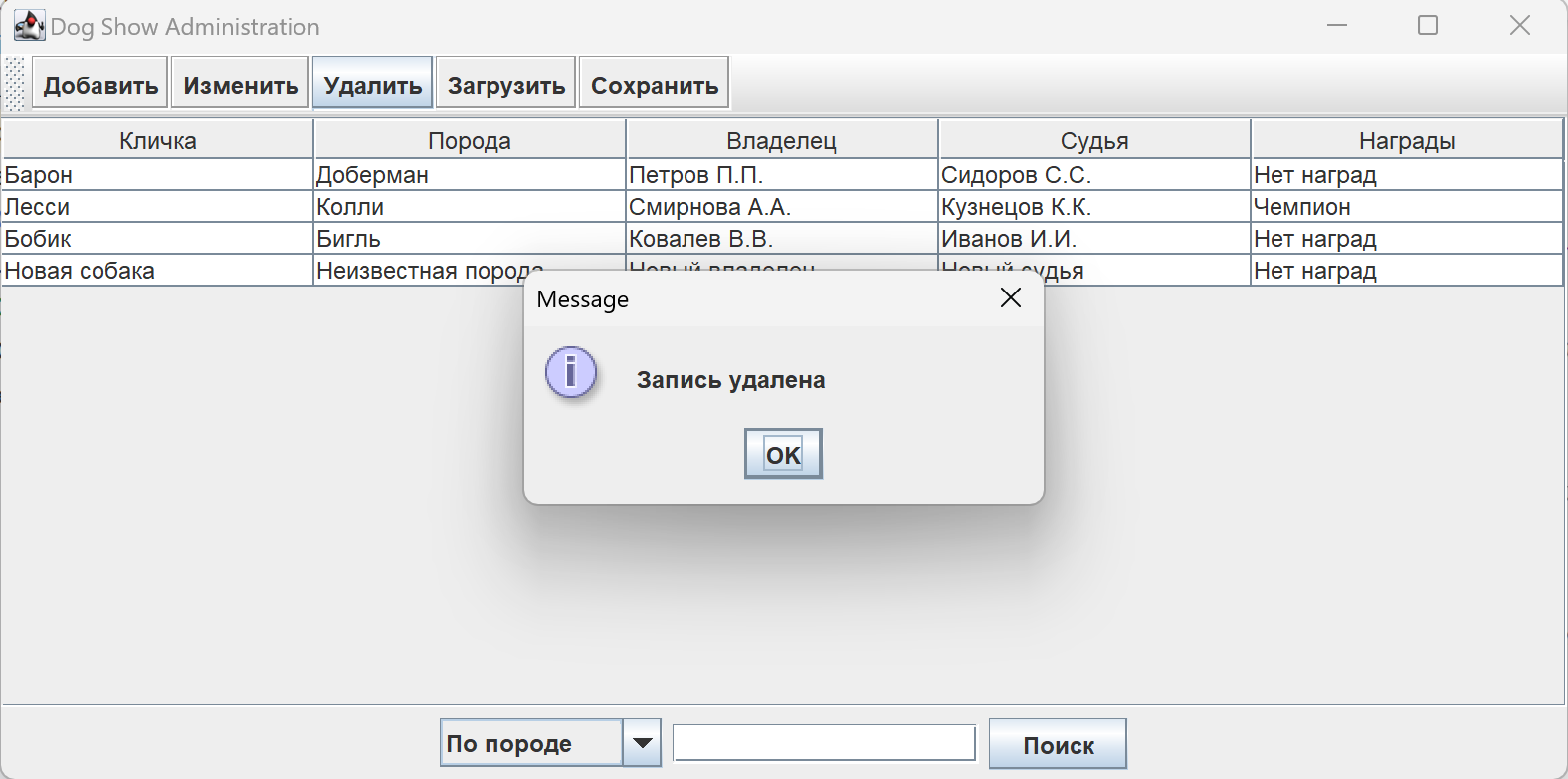


Рисунок 5 – Результат удаления

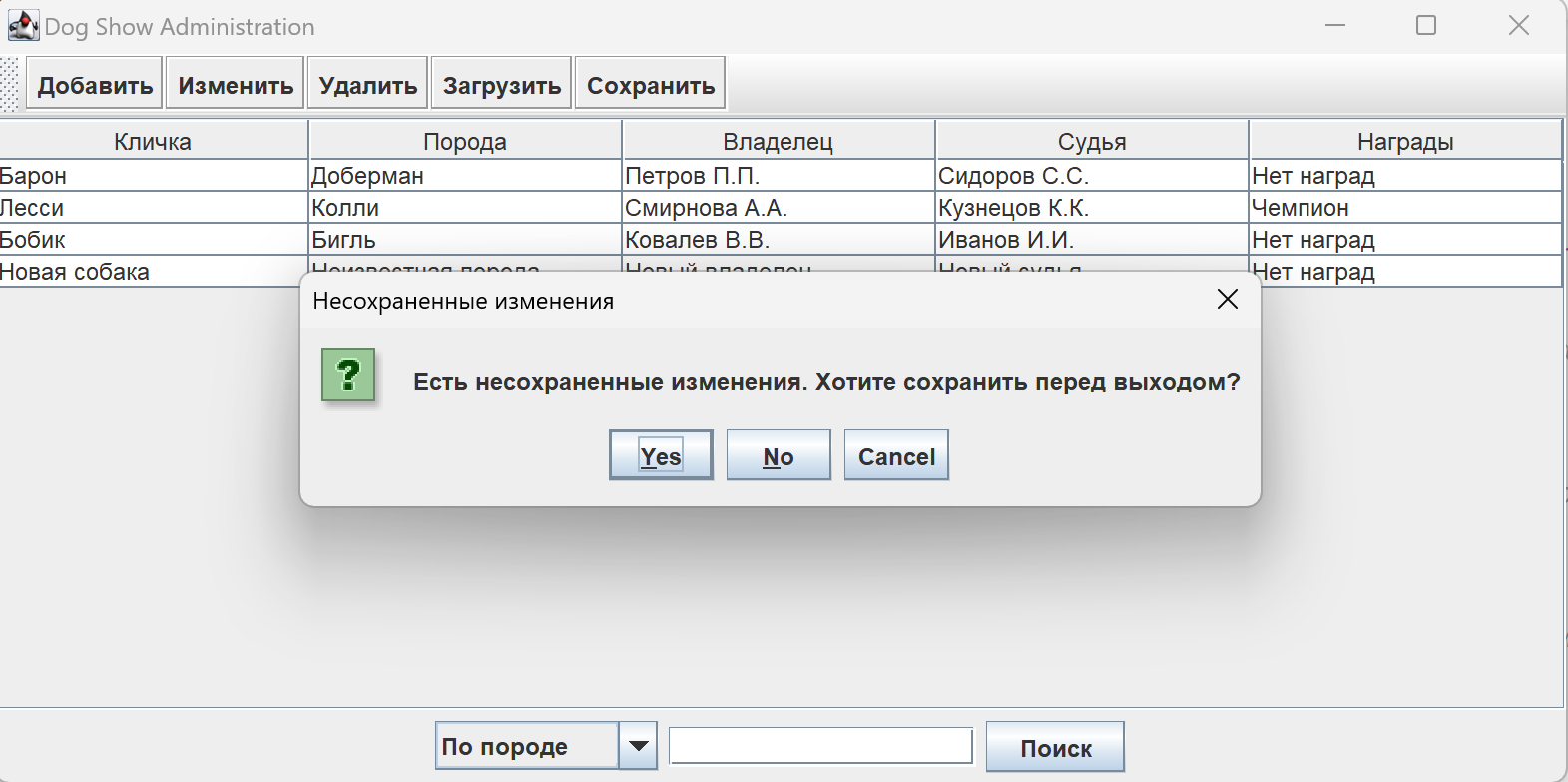


Рисунок 6 – Слушатель для закрытия окна

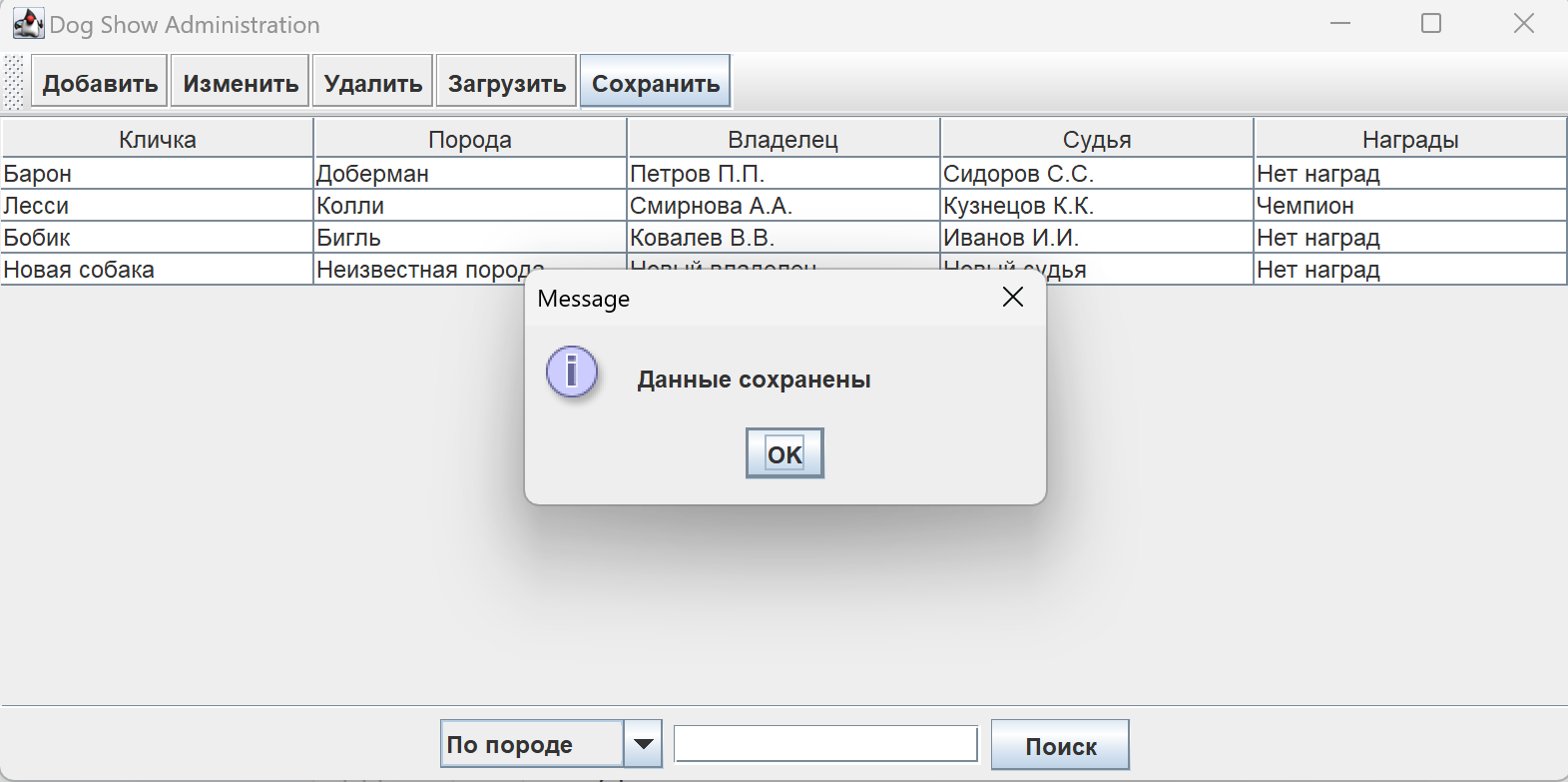


Рисунок 7 – Слушатель для кнопки сохранения и если при закрытии окна выбран вариант сохранить данные

# Текст программы

|  |
| --- |
| **import javax.swing.\*; import javax.swing.table.DefaultTableModel; import java.awt.\*; import java.awt.event.\*;  */\*\*  \* @author Лебедев Игнат 3312  \* @version 1.0  \*/* public class Main {  *// Объявление графических компонентов* private JFrame mainFrame;  private DefaultTableModel tableModel;  private JTable dataTable;  private JButton addDogButton, editDogButton, deleteDogButton, loadDogButton, saveDogButton;  private JTextField searchField;  private JComboBox<String> searchCriteriaComboBox;  private boolean unsavedChanges = false; *// Флаг для отслеживания несохраненных изменений   /\*\*  \* Метод для построения и визуализации экранной формы.  \*/* public void show() {  *// Создание основного окна* mainFrame = new JFrame("Dog Show Administration");  mainFrame.setSize(800, 400);  mainFrame.setLocation(100, 100);  mainFrame.setDefaultCloseOperation(JFrame.*DO\_NOTHING\_ON\_CLOSE*); *// Слушатель для закрытия окна добавлен ниже   // Создание основного контейнера с BorderLayout* JPanel mainPanel = new JPanel(new BorderLayout());   *// Создание кнопок для панели инструментов* addDogButton = new JButton("Добавить");  editDogButton = new JButton("Изменить");  deleteDogButton = new JButton("Удалить");  loadDogButton = new JButton("Загрузить");  saveDogButton = new JButton("Сохранить");   *// Создание панели инструментов и добавление кнопок* JToolBar toolBar = new JToolBar("Панель инструментов");  toolBar.add(addDogButton);  toolBar.add(editDogButton);  toolBar.add(deleteDogButton);  toolBar.add(loadDogButton);  toolBar.add(saveDogButton);   *// Добавление панели инструментов в верхнюю часть mainPanel* mainPanel.add(toolBar, BorderLayout.*NORTH*);   *// Создание таблицы с данными о собаках* String[] columns = {"Кличка", "Порода", "Владелец", "Судья", "Награды"};  String[][] data = {  {"Рекс", "Немецкая овчарка", "Иванов И.И.", "Петров П.П.", "Лучший в породе"},  {"Барон", "Доберман", "Петров П.П.", "Сидоров С.С.", "Нет наград"},  {"Лесси", "Колли", "Смирнова А.А.", "Кузнецов К.К.", "Чемпион"},  {"Бобик", "Бигль", "Ковалев В.В.", "Иванов И.И.", "Нет наград"}  };  tableModel = new DefaultTableModel(data, columns);  dataTable = new JTable(tableModel);  JScrollPane scrollPane = new JScrollPane(dataTable);   *// Добавление таблицы в центральную часть mainPanel* mainPanel.add(scrollPane, BorderLayout.*CENTER*);   *// Создание элементов для простого поиска* searchCriteriaComboBox = new JComboBox<>(new String[]{"По породе", "По владельцу", "По судье"});  searchField = new JTextField(15);  JButton searchButton = new JButton("Поиск");   *// Создание панели поиска и фильтрации* JPanel searchPanel = new JPanel();  searchPanel.add(searchCriteriaComboBox);  searchPanel.add(searchField);  searchPanel.add(searchButton);   *// Добавление панели поиска в нижнюю часть mainPanel* mainPanel.add(searchPanel, BorderLayout.*SOUTH*);   *// Добавление mainPanel в главное окно* mainFrame.add(mainPanel);   */\*\*  \* Слушатель для кнопки "Добавить".  \*/* addDogButton.addActionListener(new ActionListener() {  @Override  public void actionPerformed(ActionEvent e) {  *// Добавление новой записи в таблицу* tableModel.addRow(new Object[]{"Новая собака", "Неизвестная порода", "Новый владелец", "Новый судья", "Нет наград"});  unsavedChanges = true; *// Устанавливаем флаг при изменении данных* JOptionPane.*showMessageDialog*(mainFrame, "Добавлена новая собака");  }  });   */\*\*  \* Слушатель для кнопки "Изменить".  \*/* editDogButton.addActionListener(new ActionListener() {  @Override  public void actionPerformed(ActionEvent e) {  unsavedChanges = true; *// Устанавливаем флаг при изменении данных* JOptionPane.*showMessageDialog*(mainFrame, "Изменена информация о собаке");  }  });   */\*\*  \* Слушатель для кнопки "Удалить".  \*/* deleteDogButton.addActionListener(new ActionListener() {  @Override  public void actionPerformed(ActionEvent e) {  int selectedRow = dataTable.getSelectedRow(); *// Получение индекса выбранной строки* if (selectedRow != -1) {  int response = JOptionPane.*showConfirmDialog*(mainFrame, "Вы уверены, что хотите удалить эту запись?", "Подтверждение удаления", JOptionPane.*YES\_NO\_OPTION*);  if (response == JOptionPane.*YES\_OPTION*) {  tableModel.removeRow(selectedRow); *// Удаление строки* unsavedChanges = true; *// Устанавливаем флаг при изменении данных* JOptionPane.*showMessageDialog*(mainFrame, "Запись удалена");  }  } else {  JOptionPane.*showMessageDialog*(mainFrame, "Пожалуйста, выберите строку для удаления");  }  }  });   */\*\*  \* Слушатель для кнопки "Сохранить".  \*/* saveDogButton.addActionListener(new ActionListener() {  @Override  public void actionPerformed(ActionEvent e) {  saveDataToFile(); *// Сохранение данных* }  });   */\*\*  \* Слушатель для закрытия окна.  \*/* mainFrame.addWindowListener(new WindowAdapter() {  @Override  public void windowClosing(WindowEvent e) {  if (unsavedChanges) {  int response = JOptionPane.*showConfirmDialog*(mainFrame, "Есть несохраненные изменения. Хотите сохранить перед выходом?", "Несохраненные изменения", JOptionPane.*YES\_NO\_CANCEL\_OPTION*);  if (response == JOptionPane.*YES\_OPTION*) {  saveDataToFile(); *// Логика сохранения данных* mainFrame.dispose();  } else if (response == JOptionPane.*NO\_OPTION*) {  mainFrame.dispose(); *// Закрыть окно без сохранения* }  } else {  mainFrame.dispose(); *// Закрыть окно, если изменений нет* }  }  });   *// Отображение окна* mainFrame.setVisible(true);  }   */\*\*  \* Метод для сохранения данных в файл.  \*/* private void saveDataToFile() {  *// Логика сохранения данных в файл* JOptionPane.*showMessageDialog*(mainFrame, "Данные сохранены");  unsavedChanges = false; *// Сбрасываем флаг после сохранения* }   */\*\*  \* Главный метод для запуска приложения.  \* @param args аргументы командной строки  \*/* public static void main(String[] args) {  *// Создание и отображение формы* new Main().show();  } }** |

# Приложение

Ссылка на репозиторий: <https://github.com/TrueTalentless/OOP-LAB-03>