**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Вычислительной техники**

отчет

**по лабораторной работе № 3**

**по дисциплине «Технология объектно-ориентированного программирования и проектирования»**

Тема: Обработка событий

|  |  |
| --- | --- |
| Студент гр. 4312 | Устинов В. А. |
| Преподаватель | Лазарев Э. А. |

Санкт-Петербург

2025

**Цель работы**

Знакомство со способами подключения слушателей событий к графическим компонентам пользовательского интерфейса.

**Описание действий слушателей**

В коде слушатели объявляются в двух местах в цикле создания кнопок и в функции поиска. Объявление выглядит следующим образом *buttons[i].addActionListener(e -> handleButtonClick(buttonIndex*)) к каждой кнопке добавляется слушатель, в котором происходим обработка события и вызов функции, которая выполняет операцию соответствующую индексу кнопки.

* Add (buttonIndex = 3):

- Создает экземпляр AddElementWindow

- Показывает диалоговое окно для ввода данных о собаке

- Добавляет новую запись в таблицу и в список dogs

* Delete (buttonIndex = 4):

- Создает экземпляр DeleteElementWindow

- Показывает диалог для выбора записи к удалению

- Удаляет выбранную запись из таблицы и списка dogs

* Edit (buttonIndex = 5):

- Создает экземпляр EditElementWindow

- Показывает диалог для редактирования выбранной записи

- Обновляет данные в таблице и списке dogs

* Print (buttonIndex = 6):

- Создает экземпляр PrinterWindow

- Показывает данные в формате, пригодном для печати

* Exit (buttonIndex = 7):

- Показывает диалог подтверждения выхода

- При подтверждении - завершает работу приложения

* Search (buttonIndex = 8):

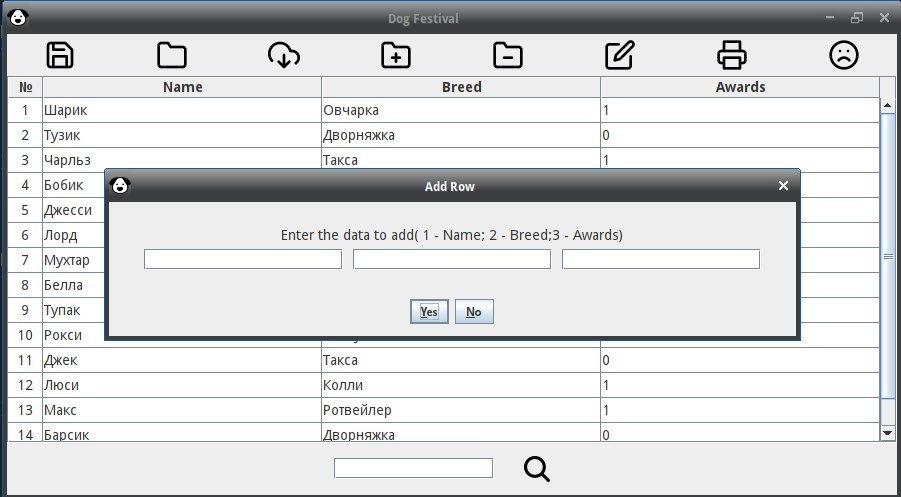
- Получает текст из searchTextField

- Фильтрует данные в таблице по введенному тексту

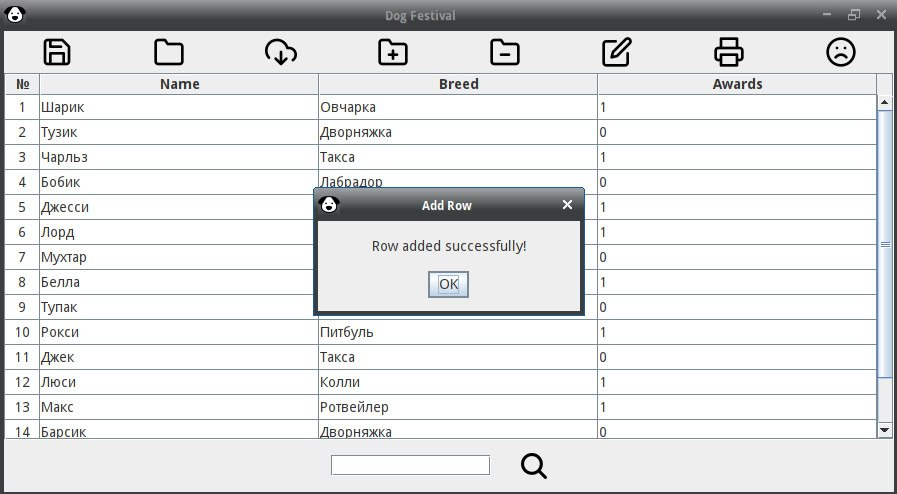
- Показывает только совпадающие строки

- Восстанавливает полную таблицу при пустом поиске

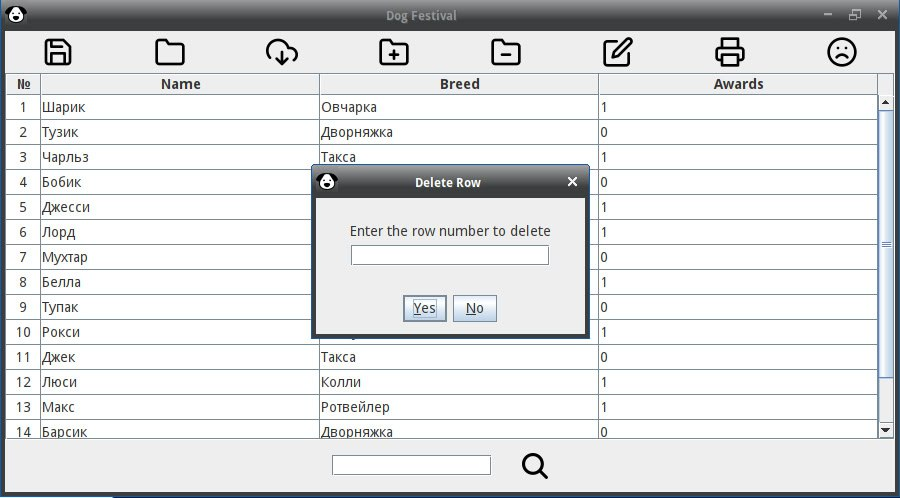
**Скриншоты, иллюстрирующие работу слушателей**

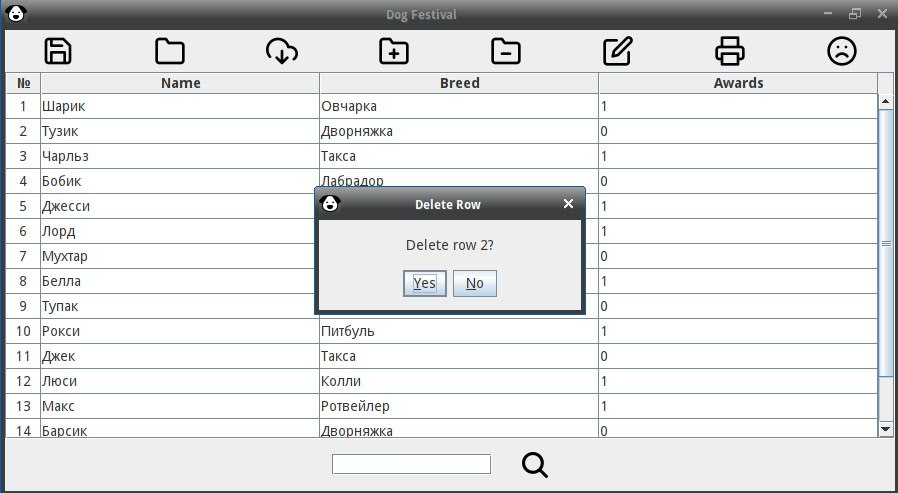
1. Add

*Рис. 1 Ввод данных о собаке для добавления*

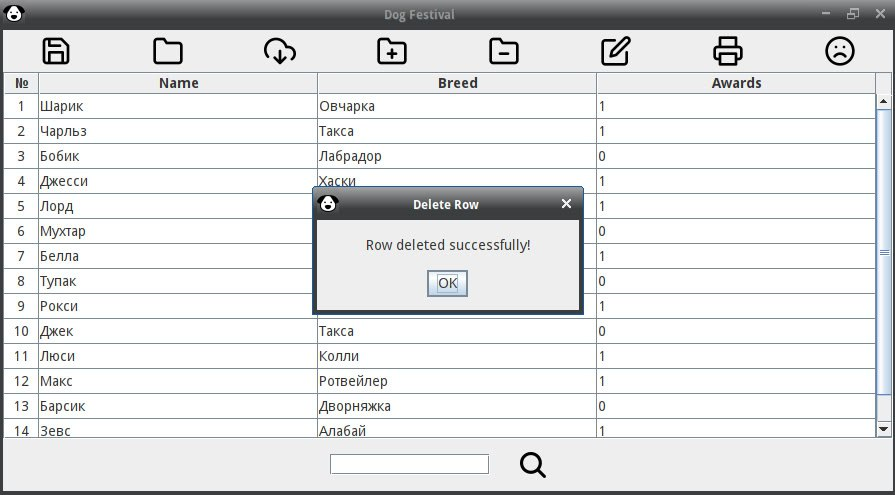


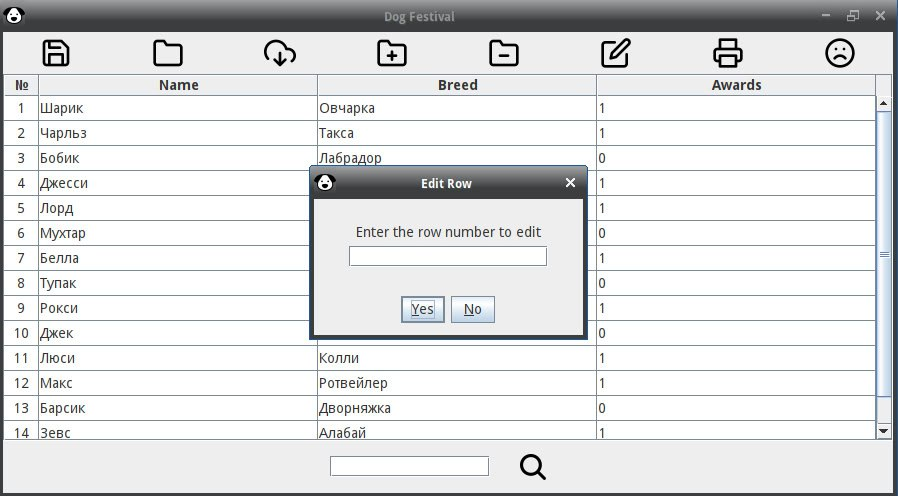
*Рис. 2 Уведомление о добавлении данных*

1. Delete

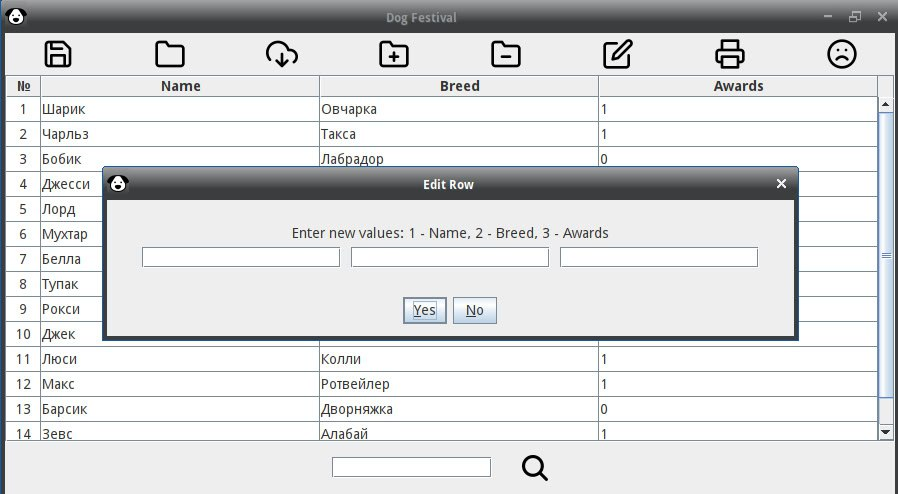
*Рис. 3 Ввод номера строки для удаления*

*Рис. 4 Подтверждение удаления*

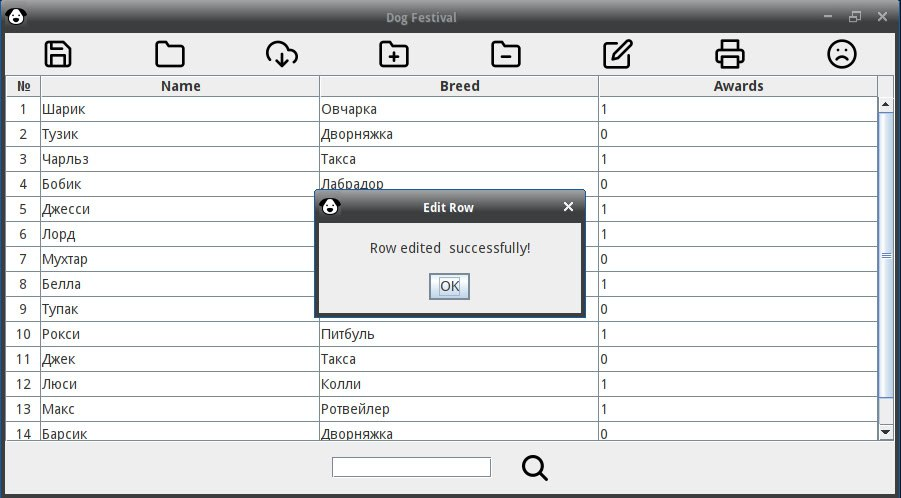
*Рис. 5 Подтверждение удаления*

1. Edit

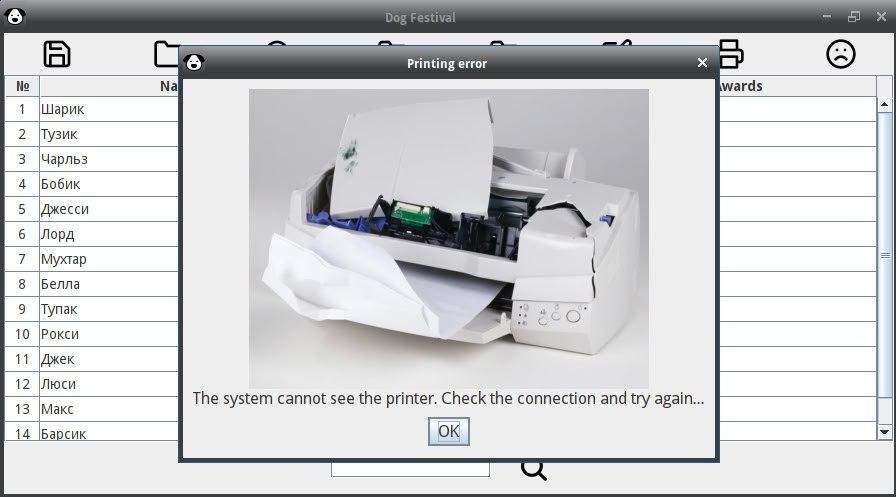
*Рис. 6 Ввод номера строки для редактирования*



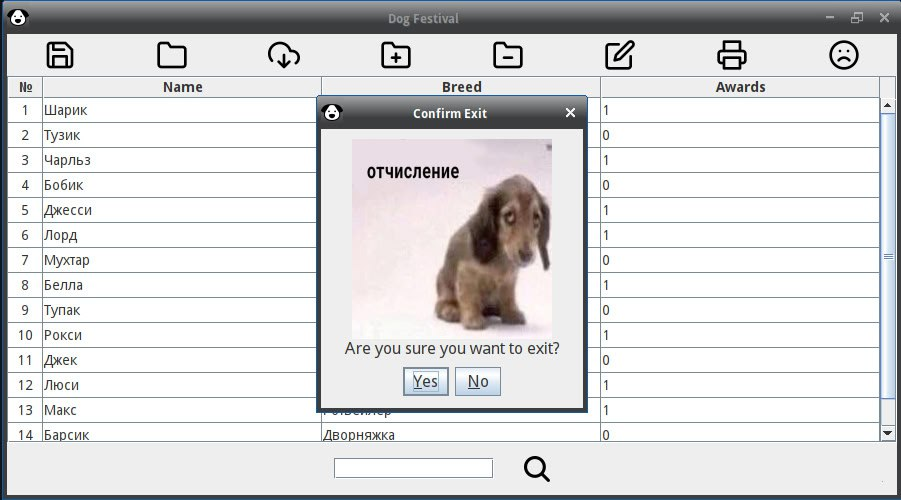
*Рис. 7 Ввод новых данных в выбранную строку*



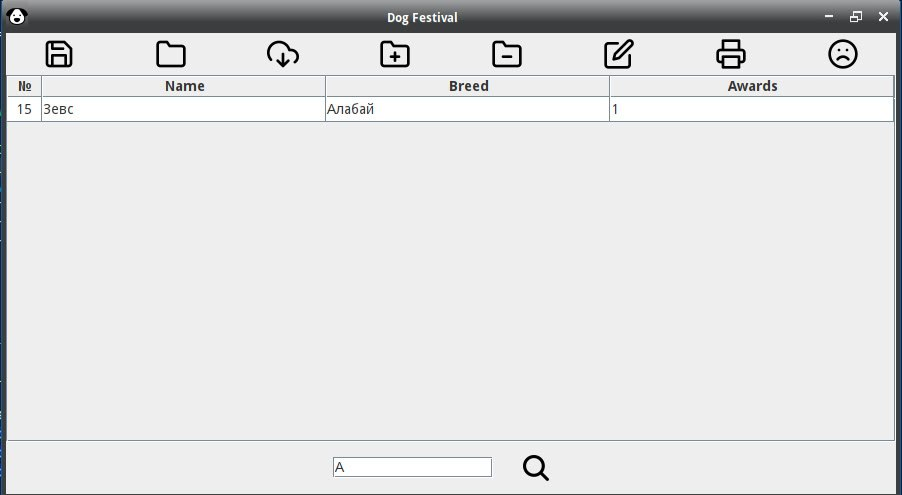
*Рис. 8 Подтверждение удаления*

1. Print

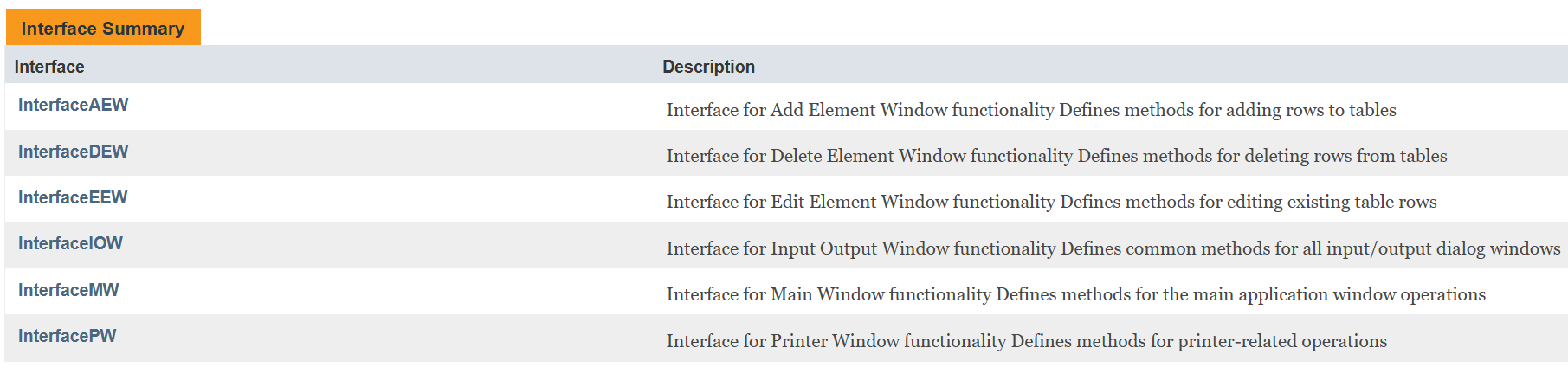
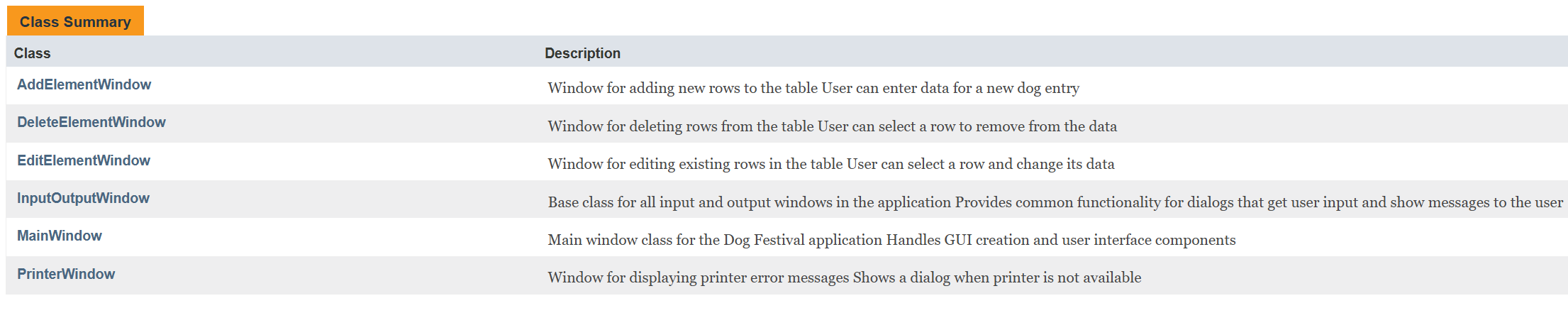
*Рис. 9 Принтер не подключен*

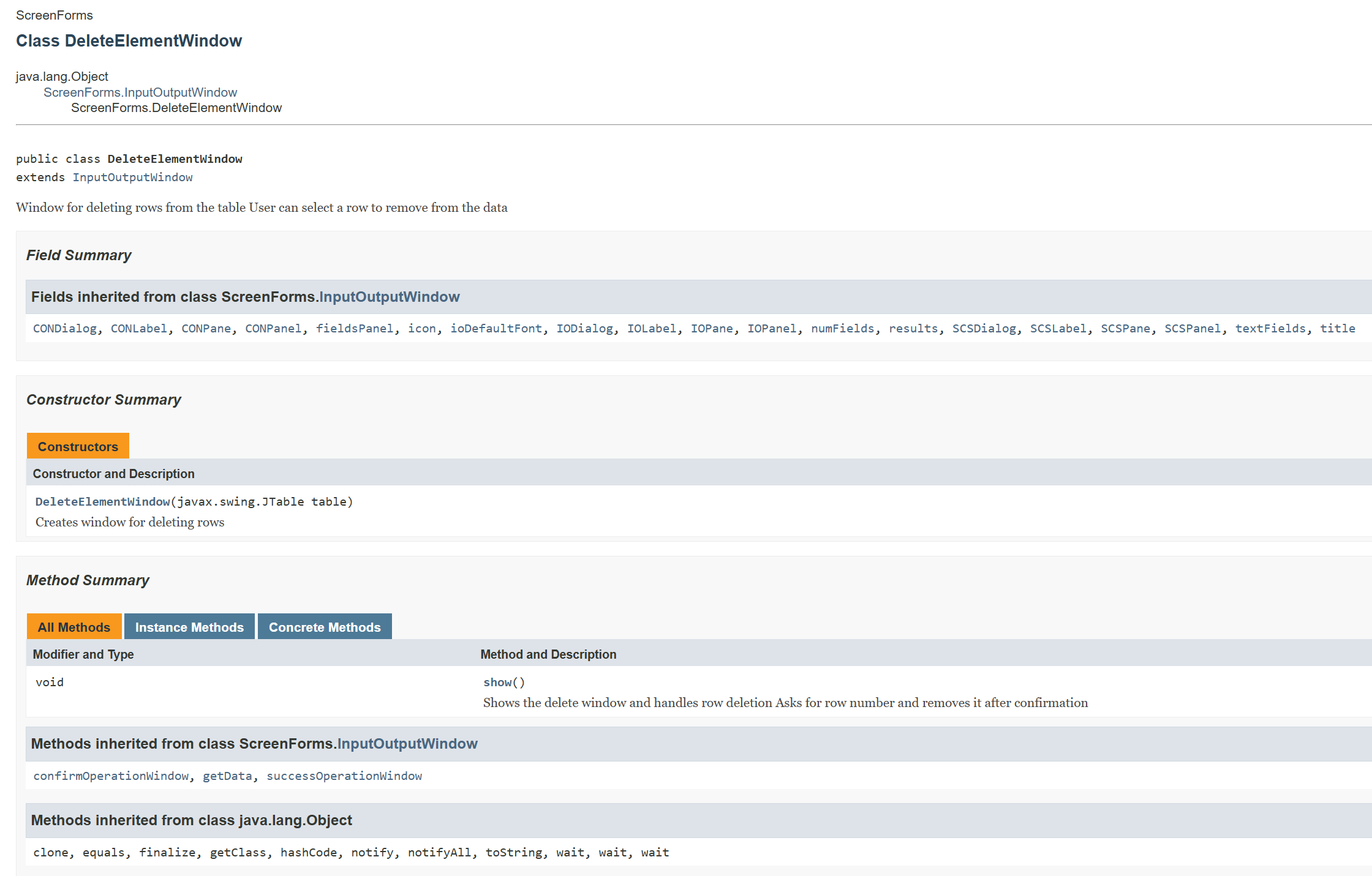
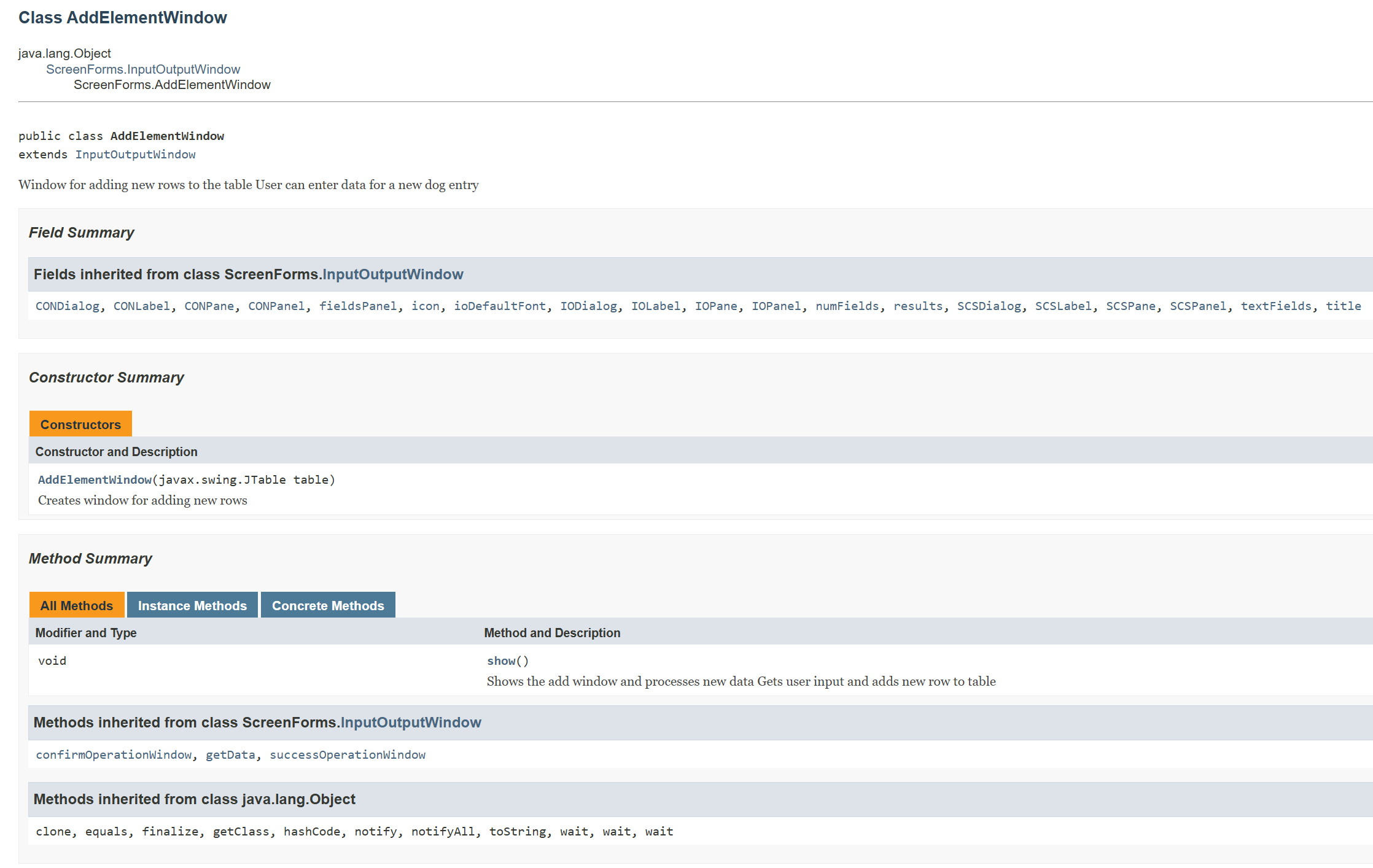
1. Exit

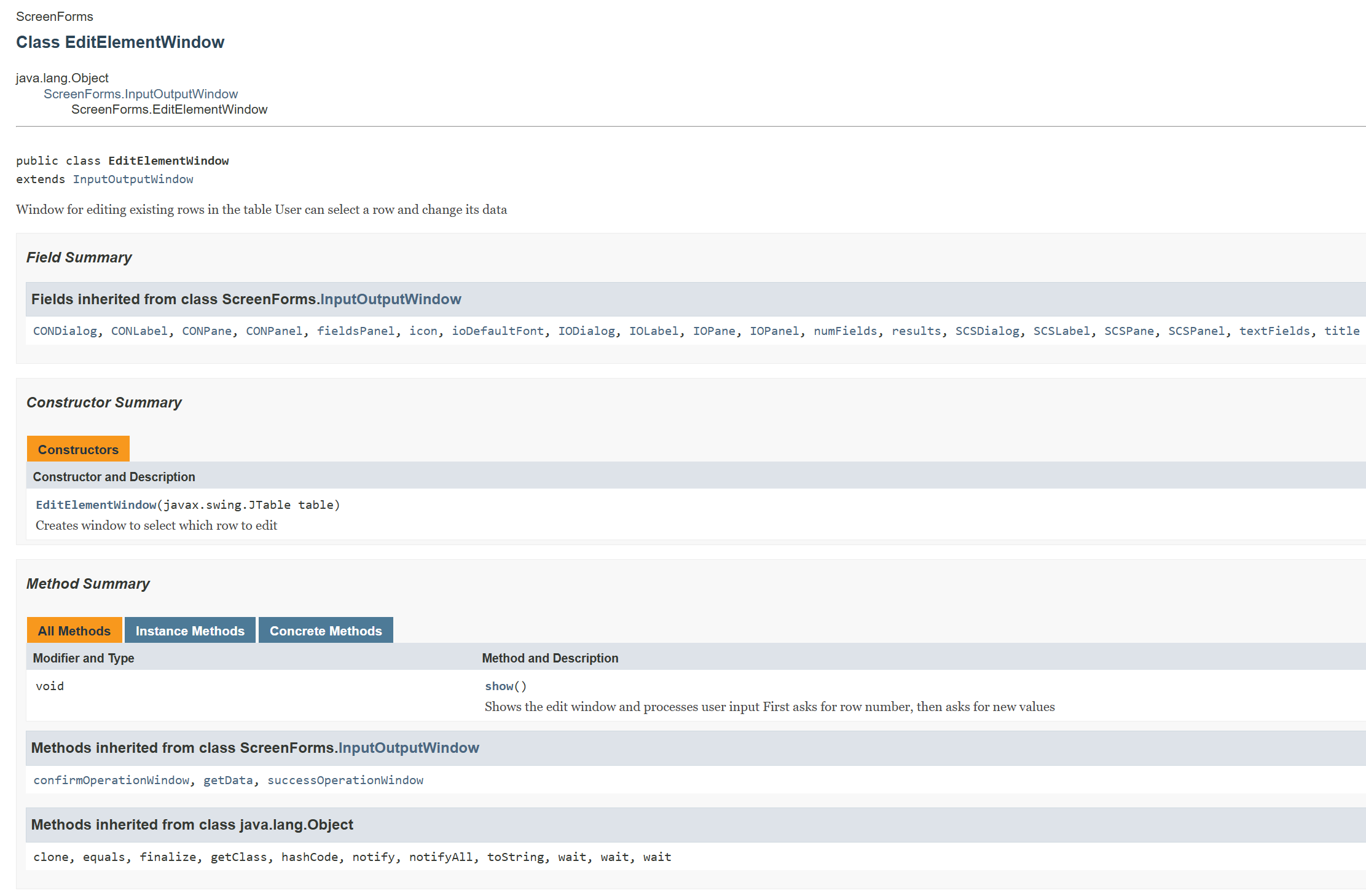
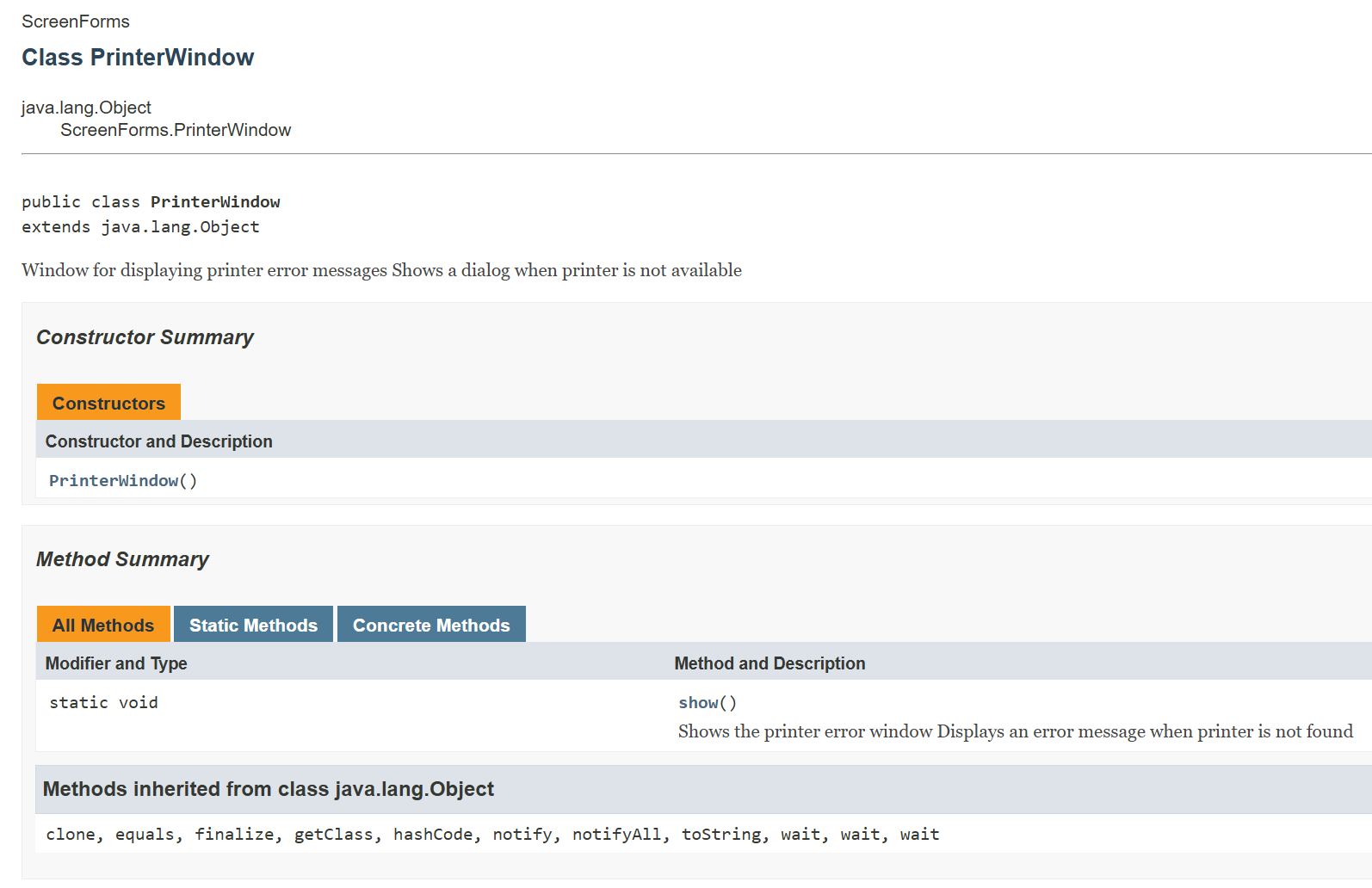
*Рис. 10 Подтверждение выхода*

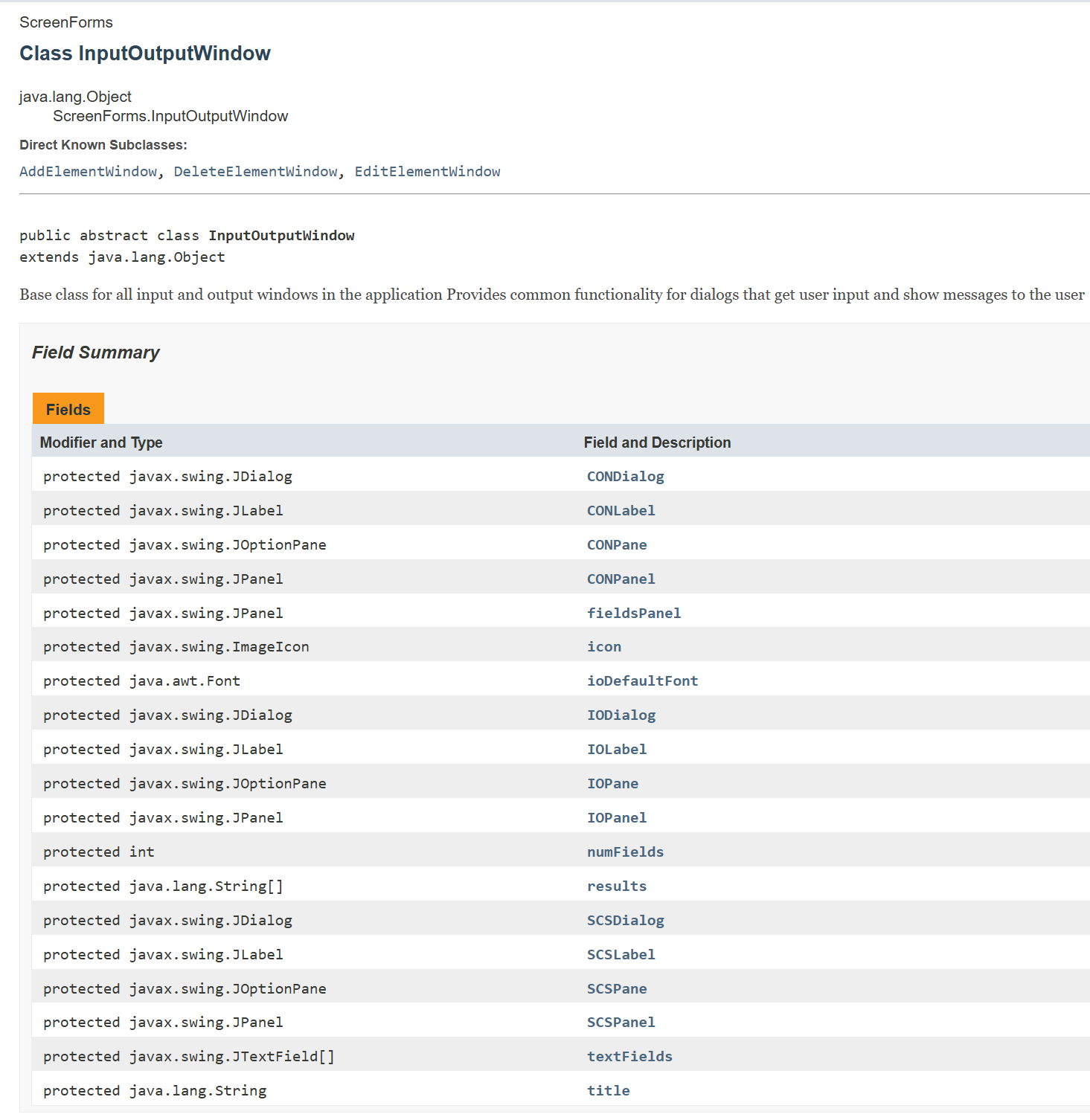
1. Search

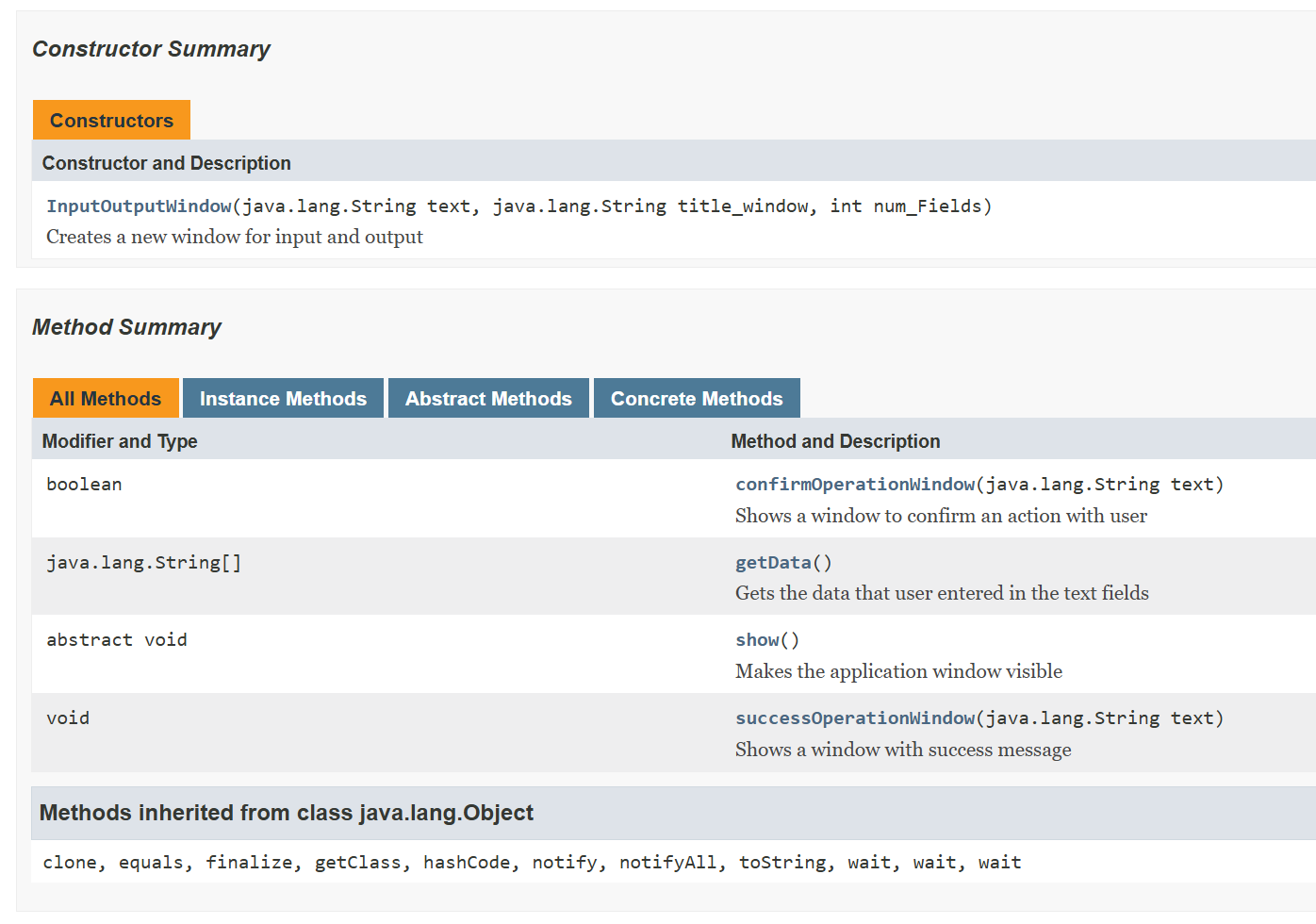
*Рис. 11 Поиск собак*

**Документация Javadoc**









**Исходные тексты слушателей**  
private void initButtonsPanel()  
    {  
     buttonsPanel = new JPanel();*//init buttons panel*  
        buttonsPanel.setLayout(new GridLayout(1, 8, 10, 10));*//set 1 row, 8 cols, 10 width, 10 heigth*  
  
        buttons = new JButton[9];*//init buttons array*  
        for(int i = 0; i < 9; i++)  
        {  
            ImageIcon imageForButton = new ImageIcon(imagePaths[i]);*//init image for icon*  
            Image scaledImage = imageForButton.getImage().getScaledInstance(32, 32, Image.SCALE\_SMOOTH);*//scale without loss of quality and fixed size of 32 by 32 px*  
            ImageIcon buttonIcon = new ImageIcon(scaledImage);*//init icon for button*  
            final int buttonIndex = i; *// need final for lambda function*  
  
            buttons[i] = new JButton(buttonIcon);*//set icon*  
            buttons[i].setToolTipText(tooltips[i]);*//set button tooltips*  
            buttons[i].setBorderPainted(false);*//remove the frame*  
            buttons[i].setContentAreaFilled(false);*//set transparent background*  
            buttons[i].setFocusPainted(false);*//Remove backlight when focusing*  
            buttons[i].addActionListener(e -> handleButtonClick(buttonIndex));*//add click handler calls lambda that calls click logic*  
            if (i < 8) {  
             buttonsPanel.add(buttons[i]);*//add button into buttonsPanel*  
            }  
        }  
    }

private void initSearchPanel()  
    {  
     inputPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 10, 5));*//init text panel for search*  
     searchTextField = new JTextField(12);  
        searchTextField.setFont(defaultFont);  
        buttons[8].addActionListener(e -> handleButtonClick(8));  
        inputPanel.add(searchTextField);  
        inputPanel.add(buttons[8]);  
    }

*/\*\*  
     \* Handles all button clicks in the app  
     \*/*  
    private static void handleButtonClick(int buttonIndex)  
    {  
        if (buttonIndex < tooltips.length)  
        {  
            switch(buttonIndex)  
            {  
             case 3:  
                 try {  
                     AddElementWindow addElem = new AddElementWindow(dogsTable);  
                     addElem.show();  
                 } catch (IOException e) {  
                     e.printStackTrace();  
                 }  
                 break;  
             case 4:  
                 try {  
                     DeleteElementWindow deleteElem = new DeleteElementWindow(dogsTable);  
                     deleteElem.show();  
                 } catch (IOException e) {  
                     e.printStackTrace();  
                 }  
                 break;  
  
             case 5:  
                 try {  
                     EditElementWindow editElem = new EditElementWindow(dogsTable);  
                     editElem.show();  
                 } catch (IOException e) {  
                     e.printStackTrace();  
                 }  
                 break;  
                 case 6:  
                     try {  
                         PrinterWindow.show();  
                     } catch (IOException e) {  
                     }  
                     break;  
                 case 7:  
                     exitApplication();  
                     break;  
                 case 8:  
                  String text = searchTextField.getText();  
                  performSearch(text);  
                  break;  
  
            }  
        }  
    }