**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра вычислительной техники**

отчет

**по лабораторной работе №2**

**по дисциплине «Объектно-ориентированное программирование»**

Тема: «**Разработка интерфейса пользователя**»

|  |  |  |
| --- | --- | --- |
| Студент гр. 4311 |  | Левицкий М. Ю. |
| Преподаватель |  | Лазарев Э.А. |

Санкт-Петербург

2025

# ЦЕЛЬ

Знакомство с правилами построения экранной формы.

# ЗАДАНИЕ

1. Выбрать тему для формы (например, форма заказа, регистрации пользователя, поиска по каталогу).

2. Описать её логику: что вводит пользователь, что видит в результате, какие кнопки доступны.

3. Нарисовать макет (прототип) этой формы.

4. Сопоставить каждому элементу макета конкретный Java-компонент (``JTextField``, `JButton` и т.д.).

5. Выбрать способ расположения компонентов (менеджер компоновки).

6. Создать класс и объявить в нём переменные для всех компонентов.

7. Написать метод, который создаёт и собирает форму из этих компонентов, с подробными комментариями.

8. Запустить программу и убедиться, что форма отображается.

9. Сгенерировать HTML-документацию из комментариев с помощью Javadoc.

**ОПИСАНИЕ НАЗНАЧЕНИЯ ЭКРАННОЙ ФОРМЫ**

1.Вывод информации о клиентах, их предпочитаемых газетах, адресов клиентов и телефонов  
2. Редактирование/удаление/добавление информации о клиентах

3. Поиск клиентов по базе

# КОНТРОЛЬНЫЙ ПРИМЕР

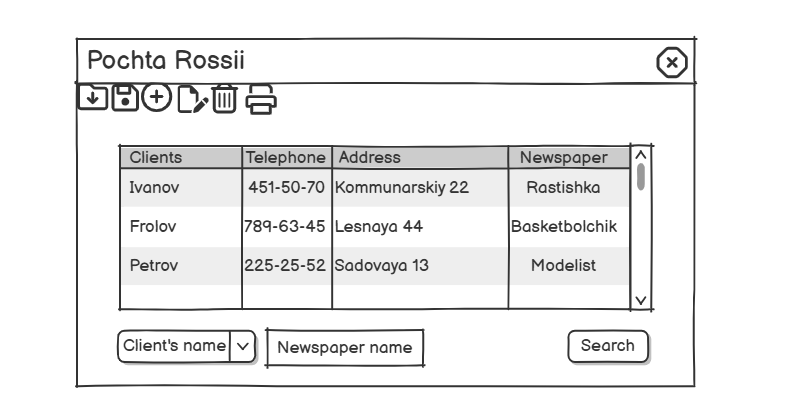
Описание проверки работоспособности приложения:

При запуске программы открывается окно, в котором отображается графический интерфейс нашего задания.

|  |
| --- |
|  |

**Скриншот для контрольного примера**

**МАКЕТ ЭКРАННОЙ ФОРМЫ**

****

# ТЕКСТ ДОКУМЕНТАЦИИ

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

# Код программы

|  |
| --- |
| **package** Lab2;  **import** javax.swing.\*;  **import** java.awt.\*;  **import** javax.swing.table.DefaultTableModel;  /\*\*  \* Главный класс приложения "Почта России" для управления клиентами.  \* Содержит графический интерфейс для отображения и управления списком клиентов.  \*  \* **@author** Mikhail  \* **@version** 1.0  \*/  **public** **class** Prog {    /\*\*  \* Создает и отображает главное окно приложения с интерфейсом управления клиентами.  \* Окно содержит:  \* - Таблицу с данными клиентов  \* - Панель инструментов с кнопками действий  \* - Панель ввода данных клиента  \* - Панель поиска клиентов  \*  \* Размер окна: 800x600 пикселей  \* Расположение: центр экрана  \*/    **public** **void** show() {  JFrame mainFrame = **new** JFrame("Pochta Rossii");  mainFrame.setSize(800, 600);  mainFrame.setLocationRelativeTo(**null**);  mainFrame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  mainFrame.setLayout(**new** BorderLayout());    // Создание модели и таблицы  String[] columns = {"Клиент", "Телефон", "Адрес", "Газета"};  Object[][] data = {  {"Ivanov", "451-50-70", "Kommunorský 22", "Rastishka"},  {"Frolov", "789-63-45", "Lesnaya 44", "Basketbolchik"},  {"Petrov", "225-25-52", "Sodovaya 13", "Modelist"}  };    DefaultTableModel model = **new** DefaultTableModel(data, columns) {  @Override  **public** **boolean** isCellEditable(**int** row, **int** column) {  **return** **false**;  }  };    JTable clientsTable = **new** JTable(model);  JScrollPane scroll = **new** JScrollPane(clientsTable);    // Создание панели инструментов  JToolBar toolBar = **new** JToolBar();  toolBar.setFloatable(**false**);    // Создание кнопок  JButton addButton = createButton("ADD.png", "Добавить");  JButton editButton = createButton("EDIT.png", "Редактировать");  JButton deleteButton = createButton("Recycle.jpg", "Удалить");  JButton searchButton = createButton("SEARCH.png", "Поиск");    toolBar.add(addButton);  toolBar.add(editButton);  toolBar.add(deleteButton);  toolBar.addSeparator();    JComboBox<String> surnameComboBox = **new** JComboBox<>();  updateSurnameComboBox(model, surnameComboBox);    toolBar.add(**new** JLabel("Фамилия:"));  toolBar.add(surnameComboBox);    // Создание панелей  JPanel inputPanel = createInputPanel();  JPanel searchPanel = createSearchPanel(searchButton);    // Компоновка  mainFrame.add(toolBar, BorderLayout.***NORTH***);  mainFrame.add(scroll, BorderLayout.***CENTER***);    JPanel southPanel = **new** JPanel(**new** BorderLayout());  southPanel.add(inputPanel, BorderLayout.***NORTH***);  southPanel.add(searchPanel, BorderLayout.***SOUTH***);  mainFrame.add(southPanel, BorderLayout.***SOUTH***);    mainFrame.setVisible(**true**);  }    /\*\*  \* Создает кнопку с изображением из папки images и текстом подсказки.  \* Если изображение не найдено, создает текстовую кнопку.  \*  \* **@param** iconName имя файла изображения (например, "ADD.png")  \* **@param** tooltip текст подсказки, отображаемый при наведении  \* **@return** созданная кнопка JButton  \*/    **private** JButton createButton(String iconName, String tooltip) {  **try** {  JButton button = **new** JButton(**new** ImageIcon("./images/" + iconName));  button.setToolTipText(tooltip);  **return** button;  } **catch** (Exception e) {  **return** **new** JButton(tooltip);  }  }    /\*\*  \* Создает панель для ввода данных клиента с полями:  \* - Имя клиента  \* - Телефон  \* - Адрес  \* - Газета  \*  \* Панель имеет заголовок "Данные клиента" и использует GridLayout (2x4).  \*  \* **@return** созданная панель JPanel  \*/    **private** JPanel createInputPanel() {  JPanel panel = **new** JPanel(**new** GridLayout(2, 4, 5, 5));  panel.setBorder(BorderFactory.*createTitledBorder*("Данные клиента"));    String[] labels = {"Имя клиента:", "Телефон:", "Адрес:", "Газета:"};  **for** (String label : labels) {  panel.add(**new** JLabel(label));  panel.add(**new** JTextField());  }    **return** panel;  }    /\*\*  \* Создает панель для поиска клиентов с текстовым полем и кнопкой поиска.  \* Панель имеет заголовок "Поиск клиента" и использует BorderLayout.  \*  \* **@param** searchButton кнопка поиска для добавления на панель  \* **@return** созданная панель JPanel  \*/    **private** JPanel createSearchPanel(JButton searchButton) {  JPanel panel = **new** JPanel(**new** BorderLayout(5, 5));  panel.setBorder(BorderFactory.*createTitledBorder*("Поиск клиента"));    JTextField searchField = **new** JTextField();  panel.add(**new** JLabel("Поиск:"), BorderLayout.***WEST***);  panel.add(searchField, BorderLayout.***CENTER***);  panel.add(searchButton, BorderLayout.***EAST***);    **return** panel;  }    /\*\*  \* Обновляет содержимое комбо-бокса фамилиями из таблицы клиентов.  \* Сохраняет текущее выделение, если это возможно.  \*  \* **@param** model модель таблицы с данными клиентов  \* **@param** comboBox комбо-бокс для обновления  \*/  **private** **void** updateSurnameComboBox(DefaultTableModel model, JComboBox<String> comboBox) {  String selected = (String) comboBox.getSelectedItem();  comboBox.removeAllItems();    **for** (**int** i = 0; i < model.getRowCount(); i++) {  comboBox.addItem((String) model.getValueAt(i, 0));  }    **if** (selected != **null** && comboBox.getItemCount() > 0) {  comboBox.setSelectedItem(selected);  }  }  /\*\*  \* Главный метод, запускающий приложение.  \* Создает и отображает графический интерфейс в потоке обработки событий Swing.  \*  \* **@param** args аргументы командной строки (не используются)  \*/  **public** **static** **void** main(String[] args) {  SwingUtilities.*invokeLater*(() -> **new** Prog().show());  }  } |

# ВЫВОДЫ

Я получил навыки работы в работе с приложением Balsamaiq, придумал макет моего интерфейса,реализовал интерфейс с помощью библиотек Java сгенерировал документацию Javadoc.