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 № 2

«Разработка интерфейса пользователя»

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

Выполнил: Локтионов Т. И.

Факультет КТИ

Группа № 3311

Подпись преподавателя _____

Санкт-Петербург

2024 г

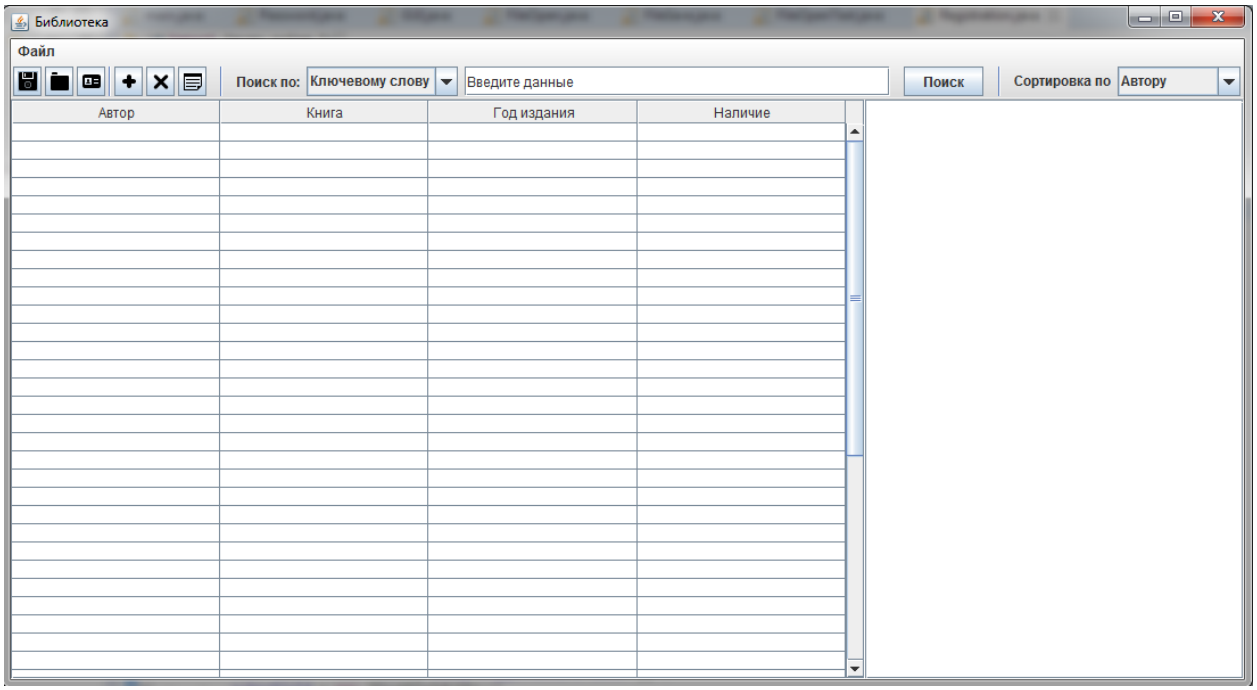
Цель работы:
знакомство с правилами построения экранной формы.

Описание задания:
1. Выбрать экранную форму из задания к курсовой работе.
2. Описать назначение экранной формы.
3. Задать вид экранной формы.
4. Создать приложение:
    4.1 Из библиотеки java.awt или javax.swing подобрать подходящий графический компонент.
    4.2 выбрать способ графической компоновки (BorderLayout && GridLayout)
    4.3 Построить экранную форму в соответствии с шаблоном

Описание экранной формы:

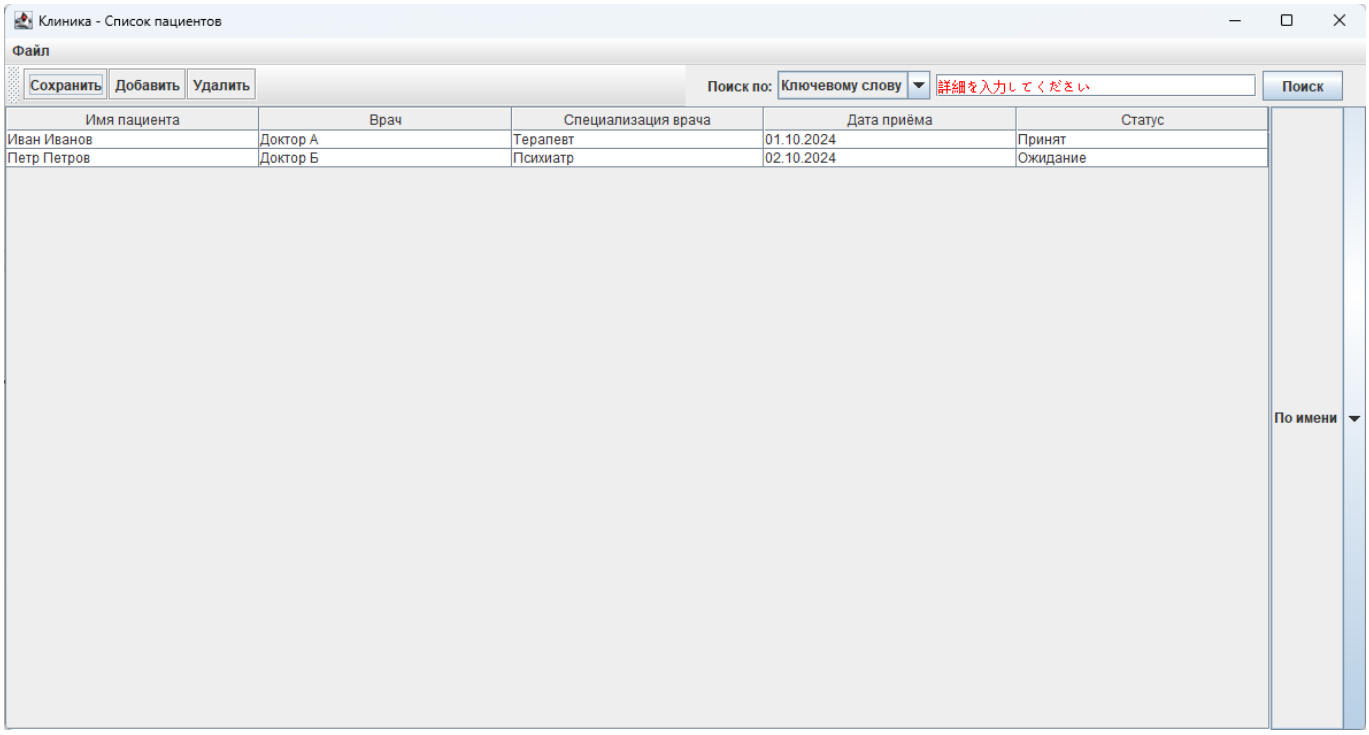
Экранная форма предназначена для отображения списка больных и врачей для администратора регистратуры поликлиники, она может менять свой размер на экране (начальный размер 800x600). Форма должна реализовывать следующие функции: загрузку списка пациентов, болезней, врачей, дат приема и состояния приема из файла и выгрузку этой информации в файл; редактирование списка, включая: добавление, удаление, корректировку информации; удобный поиск, по ключевым словам, и/или другими методами (имя пациента, дата и т.д.)

Макет (взят из приложенных к методичке файлов):

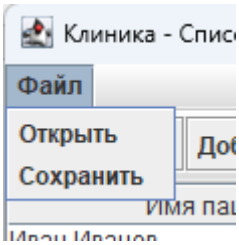


Работоспособность приложения:

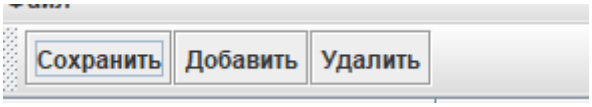
JFrame:



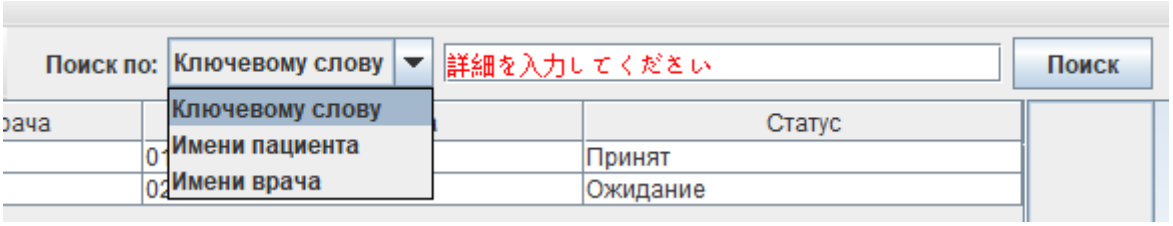
JMenuBar:



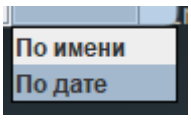
JToolBar:



JPanel && JTextField:



JComboBox:



Ссылки:

>> репозиторий: https://github.com/iconLti/LTprojects/tree/master/OOP/Java%20projects/Hospital-Lab02

>> видео отчет:

YouTube: https://youtu.be/jZ8cE-4RyuI

Google Disk: https://drive.google.com/file/d/1uPiuh6KwoJrLhBN9lzjDBScdnbxjzJlt/view?usp=sharing

Текст программы:

```
// ClinicApp.java

package main.clinicapp;

import javax.swing.SwingUtilities;

/**
 * Основной класс приложения, содержащий точку входа.
 * @author Tim Loktionov 3311
 * @version 1.00
 */
public class ClinicApp {
    /**
     * Главный метод запуска программы.
     * Вызывает метод создания и отображения интерфейса.
     *
     * @param args аргументы командной строки (не используются).
     */
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new GUI().buildAndShowGUI());
    }
}


// GUI.java

package main.clinicapp;

import javax.swing.*;
import java.awt.*;
import java.awt.event.FocusAdapter;
import java.awt.event.FocusEvent;
import javax.swing.table.DefaultTableModel;

/**
 * Основной класс, отвечающий за построение интерфейса приложения "Клиника".
 * Приложение предназначено для управления списком пациентов.
 * Включает добавление, удаление пациентов, сохранение данных и поиск по имени.
 *
 */
public class GUI {
    // Объявление компонентов
    private JFrame frame;
    private JMenuBar menuBar;
    private JMenu fileMenu;
```

```
    private JMenuItem openItem, saveItem;
    private JToolBar toolBar;
    private JButton saveButton, addButton, deleteButton;
    private JButton searchButton;
    private JComboBox<String> searchType;
    private JComboBox<String> sortType;
    private JTextField searchField;
    private JTable dataTable;
    private JScrollPane tableScrollPane;
    private DefaultTableModel tableModel;

    /**
     * Метод для построения и отображения графического интерфейса.
     * Создает основное окно приложения, меню, панель инструментов,
     * элементы для поиска и таблицу данных.
     */
    public void buildAndShowGUI() {
        frame = new JFrame("Клиника - Список пациентов");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(1200, 640);

        // Создание меню
        menuBar = new JMenuBar();
        fileMenu = new JMenu("Файл");
        openItem = new JMenuItem("Открыть");
        saveItem = new JMenuItem("Сохранить");
        fileMenu.add(openItem);
        fileMenu.add(saveItem);
        menuBar.add(fileMenu);
        frame.setJMenuBar(menuBar);

        // Панель инструментов
        toolBar = new JToolBar();
        saveButton = new JButton("Сохранить");
        addButton = new JButton("Добавить");
        deleteButton = new JButton("Удалить");
        toolBar.add(saveButton);
        toolBar.add(addButton);
        toolBar.add(deleteButton);


        // Панель поиска
        JPanel searchPanel = new JPanel();
        searchType = new JComboBox<>(new String[]{"Ключевому слову", "Имени
пациента", "Имени врача"});
        searchField = new JTextField(25);
        searchButton = new JButton("Поиск");
        // Текст подсказка
        String placeholder = "詳細を入力してください";
        searchField.setText(placeholder);
        searchField.setForeground(Color.RED); // цвет текста
        // Обработчик для изменения текста при "фокусе"
        searchField.addFocusListener(new FocusAdapter() {
            @Override
            public void focusGained(FocusEvent e) {
                if (searchField.getText().equals(placeholder)) {
                    searchField.setText("");
                    searchField.setForeground(Color.BLACK);
                }
            }
            @Override
            public void focusLost(FocusEvent e) {
                if (searchField.getText().isEmpty()) {
                    searchField.setForeground(Color.RED);
```

```
                searchField.setText(placeholder);
            }
        }
    });

    searchPanel.add(new JLabel("Поиск по:"));
    searchPanel.add(searchType);
    searchPanel.add(searchField);
    searchPanel.add(searchButton);

    // Контейнер для обеих частей
    JPanel topPanel = new JPanel(new GridLayout(1, 2)); // Одна строка, два
столбца
    topPanel.add(toolBar);
    topPanel.add(searchPanel);
    frame.add(topPanel, BorderLayout.NORTH);


    // Таблица с данными
    String[] columns = {"Имя пациента", "Врач", "Специализация врача", "Дата
приёма", "Статус"};
    tableModel = new DefaultTableModel(new Object[][]{
            {"Иван Иванов", "Доктор А", "Терапевт", "01.10.2024", "Принят"},
            {"Петр Петров", "Доктор Б", "Психиатр", "02.10.2024", "Ожидание"}
    }, columns);
    dataTable = new JTable(tableModel);
    tableScrollPane = new JScrollPane(dataTable);
    frame.add(tableScrollPane, BorderLayout.CENTER);

    // Сортировка
    sortType = new JComboBox<>(new String[]{"По имени", "По дате"});
    frame.add(sortType, BorderLayout.EAST);

    // Добавляем действия для кнопок
    addActionListeners();

    // Визуализация
    frame.setVisible(true);
}

/**
 * Метод для добавления обработчиков событий к кнопкам интерфейса.
 * Включает добавление, удаление пациентов, сохранение данных и поиск по имени
или врачу.
 */
private void addActionListeners() {
    // Добавление нового пациента
    addButton.addActionListener(e -> {
        String name = JOptionPane.showInputDialog("Введите имя пациента:");
        String doctor = JOptionPane.showInputDialog("Введите имя врача:");
        String date = JOptionPane.showInputDialog("Введите дату приёма:");
        String status = JOptionPane.showInputDialog("Введите статус:");
        if (name != null && doctor != null && date != null && status != null) {
            tableModel.addRow(new Object[]{name, doctor, date, status});
        }
    });

    // Удаление пациента
    deleteButton.addActionListener(e -> {
        int selectedRow = dataTable.getSelectedRow();
        if (selectedRow != -1) {
            tableModel.removeRow(selectedRow);
        } else {
```

```
                JOptionPane.showMessageDialog(frame, "Выберите пациента для
удаления.");
            }
        });

        // Сохранение данных (можно добавить логику сохранения в файл)
        saveButton.addActionListener(e -> JOptionPane.showMessageDialog(frame,
"Данные сохранены!"));

        // Поиск пациента или врача
        searchButton.addActionListener(e -> {
            String searchText = searchField.getText().toLowerCase();
            int searchColumn = searchType.getSelectedIndex() == 1 ? 0 : 1; // 0 - имя
пациента, 1 - имя врача

            boolean found = false;
            for (int i = 0; i < dataTable.getRowCount(); i++) {
                String value = dataTable.getValueAt(i,
searchColumn).toString().toLowerCase();
                if (value.contains(searchText)) {
                    dataTable.setRowSelectionInterval(i, i);
                    found = true;
                    break;
                }
            }
            if (!found) {
                JOptionPane.showMessageDialog(frame, "Ничего не найдено.");
            }
        });
    }
}
```