

СОЗДАНИЕ НАСТОЛЬНОГО ПРИЛОЖЕНИЯ “МЕДИАТЕКА” ДЛЯ ХРАНЕНИЯ ФИЛЬМОВ И СЕРИАЛОВ

КУРСОВАЯ РАБОТА ПО ДИСЦИПЛИНЕ “ОБЪЕКТНО-
ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ”

Выполнила: Каиркенова Адина
(Группа: 6В06112)

НАО “ЕВРАЗИЙСКИЙ
НАЦИОНАЛЬНЫЙ
УНИВЕРСИТЕТ ИМЕНИ
Л.Н.ГУМИЛЕВА”

ВВЕДЕНИЕ И АКТУАЛЬНОСТЬ

Современный пользователь взаимодействует с большим количеством контента: фильмы, сериалы, подборки платформ. Без организации коллекции возникает путаница.

Приложение «Медиатека» решает проблему структурирования и локального хранения.

Основные преимущества:

- автономность, отсутствие зависимости от интернета;
- простое добавление и удаление медиа-объектов;
- сохранение всех данных локально;
- интуитивный интерфейс.

— ЦЕЛИ И ЗАДАЧИ

1

Цель работы:

Разработать настольное приложение для хранения, поиска и управления коллекцией медиа.

2

Задачи:

создать объектно-ориентированную архитектуру;
разработать модели объектов (фильм, сериал);
реализовать графический интерфейс (Swing);
обеспечить ввод и валидацию данных;
внедрить функции сохранения/загрузки;
реализовать удаление и поиск элементов;
оформить приложение в единой визуальной теме.

ИСПОЛЬЗУЕМЫЕ ТЕХНОЛОГИИ

ДЛЯ ПРИЛОЖЕНИЯ БЫЛИ ВЫБРАНЫ ТЕХНОЛОГИИ,
ОБЕСПЕЧИВАЮЩИЕ ПРОСТОТУ И АВТОНОМНОСТЬ:

Java SE 21 - объектно-ориентированная основа проекта.
Swing / AWT - построение настольного GUI.

Java Collections Framework (List) - хранение объектов.
Serializable + ObjectOutputStream - файловое хранение

BorderLayout,
GridLayout, FlowLayout — организация интерфейса

АРХИТЕКТУРА

Приложение разделено на ключевые модули:

model - модели данных

Medialtem (абстракция)

Movie

Series

storage - слой хранения

MediaLibrary - загрузка и сохранение данных

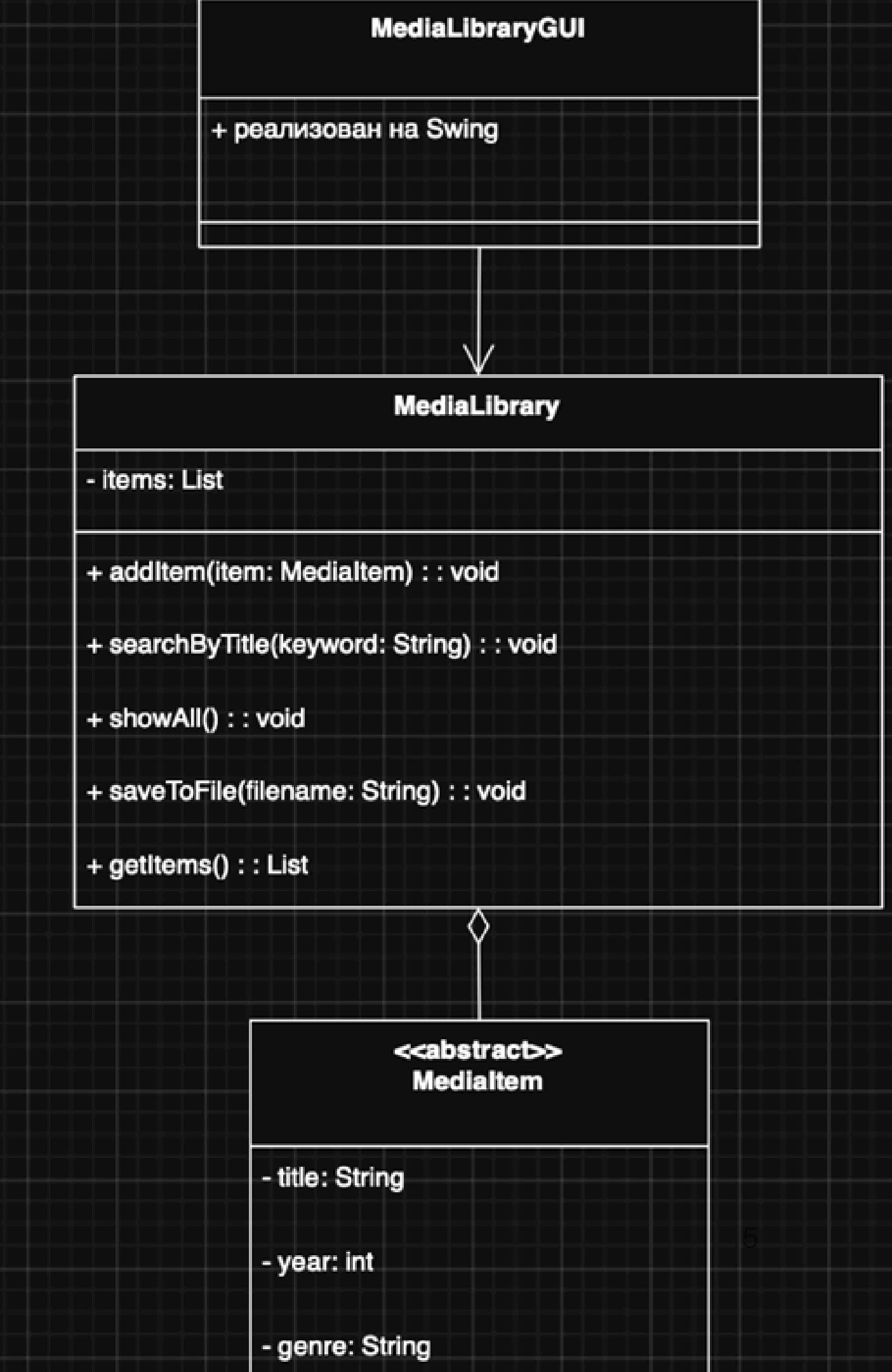
ui -графический интерфейс

MediaLibraryGUI - логика интерфейса и обработка

событий

Логика взаимодействия:

GUI → MediaLibrary → Medialtem / Movie / Series →
(сохранение в файл)



ПРИНЦИПЫ ООП

1. Инкапсуляция
private поля моделей
доступ через public getters
защита внутреннего состояния объектов
изоляция логики хранения в MediaLibrary

```
public abstract class MediaItem implements Serializable {  
    2 usages  
    private String title;  
    2 usages  
    private int year;  
    2 usages  
    private String genre;  
  
    3 usages  
    public MediaItem(String title, int year, String genre) {  
        this.title = title;  
        this.year = year;  
        this.genre = genre;  
    }  
  
    5 usages  
    public String getTitle() { return title; }  
    2 usages  
    public int getYear() { return year; }  
    2 usages  
    public String getGenre() { return genre; }
```

2. Наследование

Movie и Series наследуют
MediaItem

переопределяют метод getInfo()

```
public class Movie extends MediaItem {  
    2 usages
```

```
    4 usages  
    @Override  
    public String getInfo() {  
        return "Фильм: " + getTitle() + " (" + getYear() + "), жанр: " + getGenre() +  
               ", режиссёр: " + director;  
    }
```

— ПРИНЦИПЫ ООП

3.Полиморфизм
единый список List<MediaItem>
метод getInfo() вызывается одинаково, но выводит разные
данные

GUI не зависит от конкретного типа объекта

```
private void showAll() {  
    outputArea.setText("");  
    for (MediaItem item : library.getItems()) {  
        outputArea.append(item.getInfo());  
    }  
}
```

класс получает «способность» быть
сериализуемым через реализацию
интерфейса

```
public class MediaLibrary implements Serializable {
```

4.Абстракция
общий класс MediaItem скрывает
детали
GUI работает с абстракцией, а не с
конкретными реализациями

```
public abstract String getInfo();
```

ГРАФИЧЕСКИЙ ИНТЕРФЕЙС (UI)

UI реализован на Swing:

панель ввода новых объектов (GridLayout);

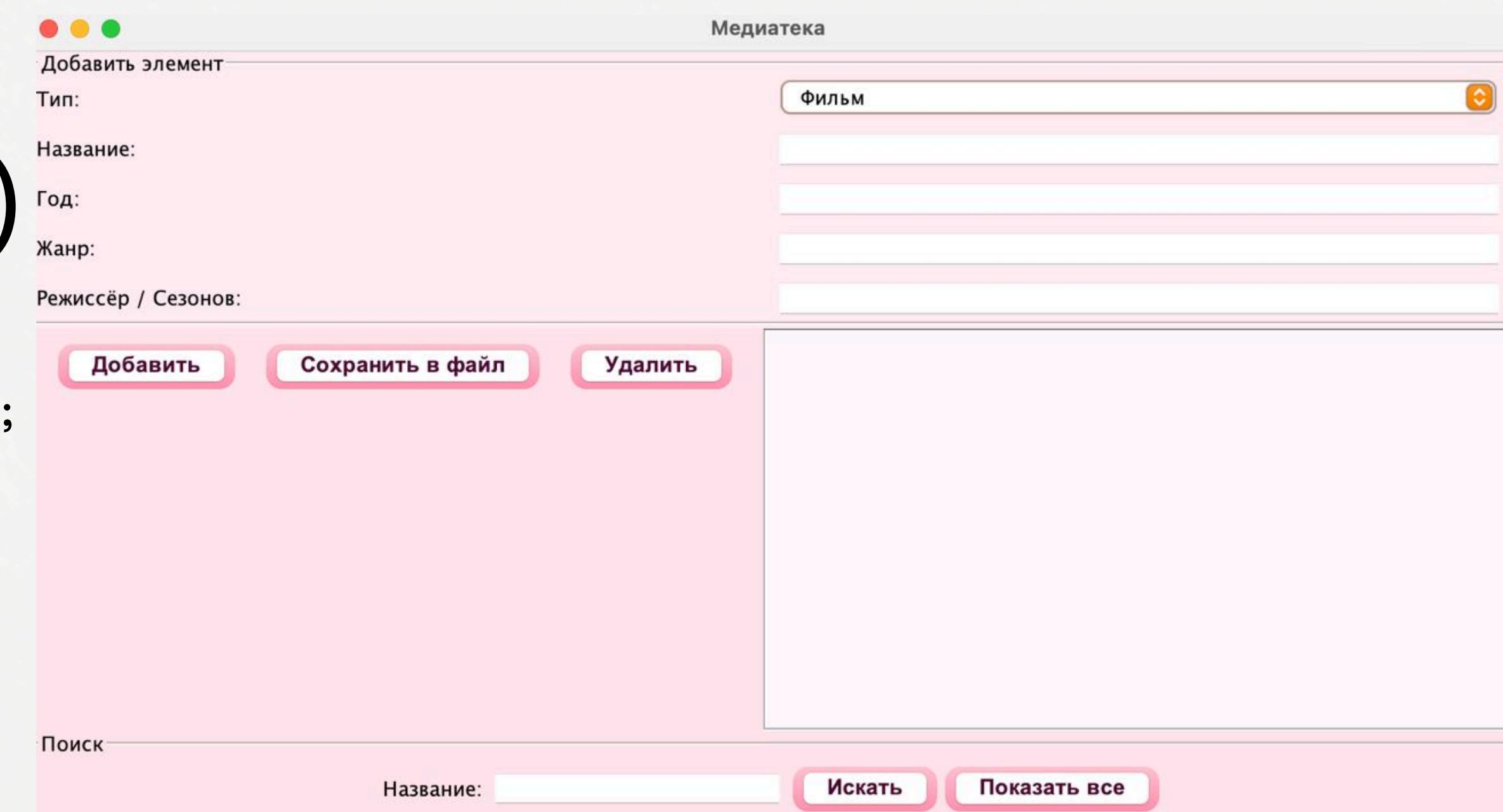
панель кнопок (FlowLayout) - красиво

расположены по центру;

зона вывода медиа (JTextArea +

JScrollPane);

панель поиска (BorderLayout.SOUTH).



ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ

Приложение позволяет:

- добавлять фильмы и сериалы;
- вводить название, год, жанр, режиссёра/сезоны;
- искать элементы по названию;
- выводить всю медиатеку;
- удалять элементы;
- сохранять и загружать данные;
- автоматически сохранять при выходе.

```
 JButton addButton = createCuteButton( text: "Добавить");
 addButton.addActionListener(this::handleAdd);

 JButton saveButton = createCuteButton( text: "Сохранить в файл");
 saveButton.addActionListener(e -> library.saveToFile( filename: "catalog.dat"));

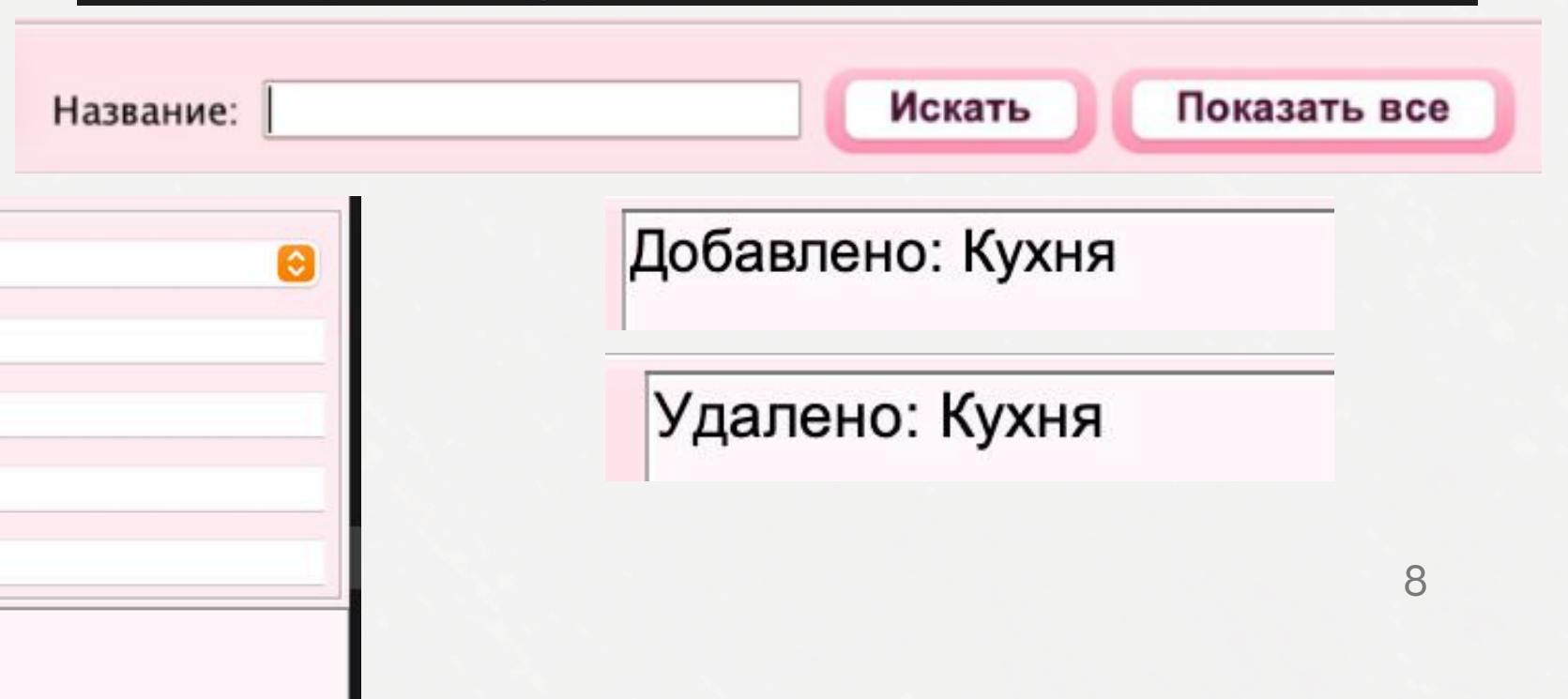
 JButton deleteButton = createCuteButton( text: "Удалить");
 deleteButton.addActionListener(e -> handleDelete());

 JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, hgap: 20, vgap: 10));
 buttonPanel.setBackground(new Color( r: 255, g: 220, b: 230));
 buttonPanel.add(addButton);
 buttonPanel.add(saveButton);
 buttonPanel.add(deleteButton);

 // Панель поиска
 JPanel searchPanel = new JPanel();
 searchPanel.setBorder(BorderFactory.createTitledBorder("Поиск"));
 searchPanel.setBackground(new Color( r: 255, g: 220, b: 230));
 searchPanel.add(new JLabel( text: "Название:"));
 searchPanel.add(searchField);

 JButton searchButton = createCuteButton( text: "Искать");
 searchButton.addActionListener(this::handleSearch);
 searchPanel.add(searchButton);

 JButton showAllButton = createCuteButton( text: "Показать все");
 showAllButton.addActionListener(e -> showAll());
 searchPanel.add(showAllButton);
```



ПРИМЕР КОДА ОБРАБОТКИ СОБЫТИЙ

Взаимодействие с GUI
построено через ActionListener:

```
private void handleAdd(ActionEvent e) {
    try {
        String title = titleField.getText();
        int year = Integer.parseInt(yearField.getText());
        String genre = genreField.getText();

        if (typeBox.getSelectedItem().equals("Фильм")) {
            String director = extraField.getText();
            library.addItem(new Movie(title, year, genre, director));
        } else {
            int seasons = Integer.parseInt(extraField.getText());
            library.addItem(new Series(title, year, genre, seasons));
        }

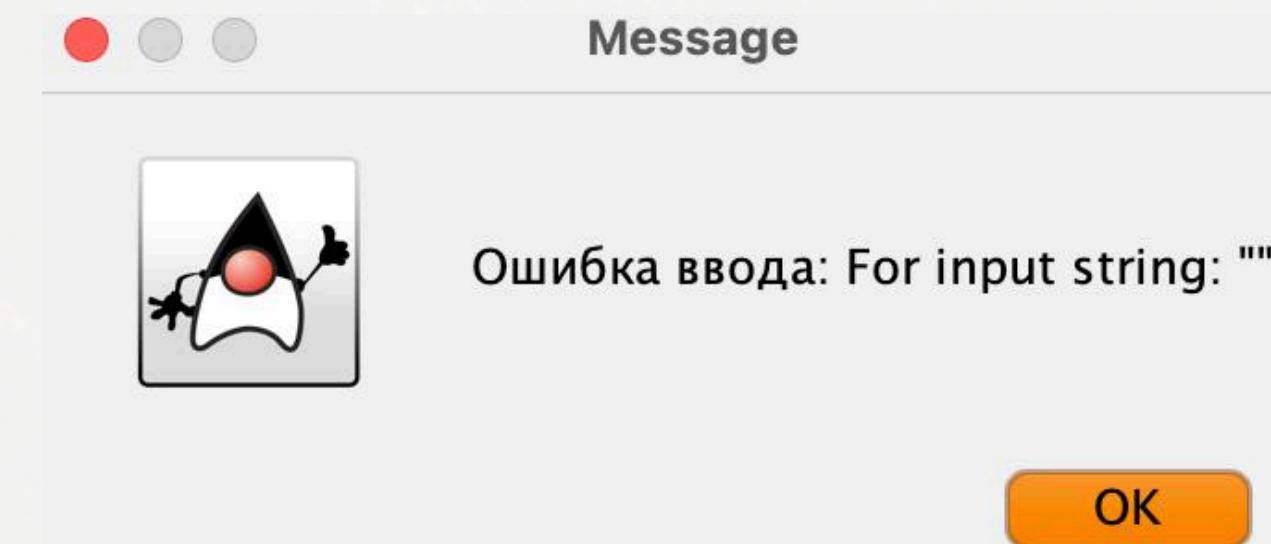
        clearInputs();
        outputArea.setText("Добавлено: " + title);

    } catch (Exception ex) {
        JOptionPane.showMessageDialog( parentComponent: this, message: "Ошибка ввода: " + ex.getMessage());
    }
}
```

```
JButton addButton = createCuteButton( text: "Добавить");
addButton.addActionListener(this::handleAdd);

JButton saveButton = createCuteButton( text: "Сохранить в файл");
saveButton.addActionListener(e -> library.saveToFile( filename: "catalog.dat"));

JButton deleteButton = createCuteButton( text: "Удалить");
deleteButton.addActionListener(e -> handleDelete());
```



ПРИМЕР МОДЕЛИ

```
public class Movie extends MediaItem {  
    2 usages  
    private String director;  
  
    2 usages  
    public Movie(String title, int year, String genre, String director) {  
        super(title, year, genre);  
        this.director = director;  
    }  
  
    4 usages  
    @Override  
    public String getInfo() {  
        return "Фильм: " + getTitle() + " (" + getYear() + "), жанр: " + getGenre() +  
               ", режиссёр: " + director;  
    }  
}
```

Класс Movie демонстрирует:
наследование поля
конструктор
полиморфизм через getInfo()

РЕЗУЛЬТАТЫ И ВЫВОДЫ

В РЕЗУЛЬТАТЕ РАБОТЫ СОЗДАНО НАСТОЛЬНОЕ ПРИЛОЖЕНИЕ:

- РЕАЛИЗОВАНЫ КЛЮЧЕВЫЕ ПРИНЦИПЫ ООП;
- ПОСТРОЕНА МНОГОСЛОЙНАЯ АРХИТЕКТУРА;
- РАЗРАБОТАН УДОБНЫЙ ГРАФИЧЕСКИЙ ИНТЕРФЕЙС;
- ОБЕСПЕЧЕНО ЛОКАЛЬНОЕ ХРАНЕНИЕ ДАННЫХ;
- ДОБАВЛЕНЫ ФУНКЦИИ ПОИСКА, ВЫВОДА, УДАЛЕНИЯ;
- ВЫПОЛНЕНА СТИЛИЗАЦИЯ ИНТЕРФЕЙСА.

Фильм: Властелин колец (2003), жанр: фентези, режиссёр: не знаю
Сериал: Острые козырьки (2015), жанр: Драма/Криминал, сезонов: 5
Сериал: Бумажный дом (2017), жанр: Криминал/Драма/Детектив, сезо
Фильм: Как приручить дракона (2024), жанр: Фэнтези/Приключения, ре
Фильм: Властелин колец (2005), жанр: Фэнтези/Приключения/, режисс
Фильм: Зеленая миля (1999), жанр: Фэнтезийная драма, режиссёр: Ф
Фильм: Пункт назначения (2008), жанр: Хоррор, режиссёр: Джейфрик
Сериал: Ходячие мертвецы (2010), жанр: Драма/Ужасы, сезонов: 6
Сериал: Игра престолов (2011), жанр: Фэнтези, сезонов: 8

```
switch (choice) {
    case "1" -> {
        System.out.print("Название: ");
        String title = sc.nextLine();
        System.out.print("Год: ");
        int year = Integer.parseInt(sc.nextLine());
        System.out.print("Жанр: ");
        String genre = sc.nextLine();
        System.out.print("Режиссёр: ");
        String director = sc.nextLine();
        library.addItem(new Movie(title, year, genre, director));
    }
}
```

СПАСИБО

ЗА

ВНИМАНИЕ