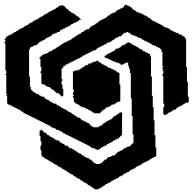


ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ

ФАКУЛТЕТ КОМПЮТЪРНИ СИСТЕМИ И ТЕХНОЛОГИИ

Документация за проект по Софтуерни шаблони

Задание №1: Да се създаде програма за управление на библиотека от медия (книги, филми, музика и д.р.) с различни полета данни. Да се предоставят функционалности за добавяне, премахване, филтриране по жанр, и поставяне на оценка рейтинг. Демонстрирайте работа с Iterator pattern и Adapter pattern.

Допълнителен софтуерен шаблон: Observer

Информация за проекта: Десктоп приложение с графичен интерфейс (JavaFX) и конзолен демо режим. Използва се Java 17 като основен език за програмиране, Maven за управление на зависимости и билд процеса, Gson за устойчивост (запис и зареждане на данни в JSON файл), JUnit за юнит тестове и IntelliJ IDEA като среда за разработка. Проектът е структуриран модулно, с отделни пакети за домейн модели, дизайн шаблони, графичен интерфейс и помощни папки. Той поддържа полиморфизъм за различни типове медии, автоматично уведомяване при промени (чрез Observer) и безопасно итератиране (чрез Iterator). За постоянство данните се записват в "library.json" автоматично при всяка промяна и се зареждат при стартиране. Проектът е тестван с юнит тестове за ключови функционалности и шаблони, и е предназначен за демонстрация на дизайн шаблони в реално приложение.

Изисквания за стартиране на проекта:

- Инсталиран Java JDK 17 или по-висока версия (препоръчително OpenJDK или Oracle JDK).

- Инсталиран Maven (или използвайте вградената версия в IntelliJ IDEA).

- За графичен интерфейс: JavaFX SDK (включен чрез Maven зависимости, но уверете се, че е зареден).

- Отваряне на папката в IntelliJ IDEA Community Edition или друга IDE с поддръжка на Maven.

- За GUI: Изпълнете команда mvn clean javafx:run в терминала на IDE или от Maven панела (Lifecycle > clean, после Plugins > javafx > javafx:run) или Изпълнете main метода в класа MediaLibraryApp.java директно от IDE (right-click > Run).

- За конзолен демо: Изпълнете main метода в класа MediaLibrary.java директно от IDE (right-click > Run).

- За тестове: Изпълнете mvn test от терминала или от Maven панела (Lifecycle > test).

- Ако има грешки при стартиране, проверете PATH към JDK и reload на Maven проекта (right-click pom.xml > Maven > Reload Project).

Файлова структура:

- src/main/java/com.student.media_library – основен корен за код:

  - model – пакет с домейн класове (Media.java – интерфейс, Book.java, Film.java, Music.java – конкретни имплементации за медии типове).

  - patterns – пакет с класове за дизайн шаблони (Observable.java и Observer.java – интерфейси за Observer pattern, LibraryDisplayObserver.java – конзолен обзървър, GenreIterator.java – итератор за филтриране, LegacyBook.java – несъвместим клас, BookAdapter.java – адаптер).

  - gui – пакет с GUI класове (MediaLibraryApp.java – основен GUI клас с таблица и бутони, AddMediaDialog.java – диалог за добавяне на медия).

  - util – пакет с помощни класове (PersistenceUtil.java – статични методи за JSON персистентност).

- src/test/java/com.student.media_library – тестове (MediaLibraryTest.java – юнит тестове за функционалност и шаблони).

- library.json – файл за персистентност (автоматично генериран, съдържа JSON масив с медии обекти).

- pom.xml – Maven конфигурационен файл (зависимости: Gson, JavaFX, JUnit; плъгини за компилация и JavaFX run).

- README.md – кратък файл с инструкции и описание (опционален, съдържа преглед на проекта).

- docs – папка за документация (uml.mermaid – Mermaid код за UML, uml.png – генерирана диаграма, documentation.pdf – този документ).

Описание на класовете и шаблоните:

- Model пакет: Този пакет съдържа интерфейса Media, който дефинира общи методи за всички медии типове (getTitle, getGenre, getRating, getYear, setRating, getDescription). Конкретните класове Book, Film и Music имплементират интерфейса с тип-специфични полета (напр. Book има author и pageCount, Film – director и durationMinutes). getDescription връща форматиран стринг с всички детайли, включително рейтинг за визуализация.

- Patterns пакет: Съдържа интерфейси Observable и Observer за Observer pattern (уведомяване при промени). LibraryDisplayObserver – конкретен обзървър за конзола, който печата актуалното състояние. GenreIterator – имплементация на Iterator за филтриране по жанр, прескачайки несъвпадащи. LegacyBook – симулира стар клас с несъвместими методи, BookAdapter – адаптира го към Media чрез мапинг на полета и методи.

- MediaLibrary клас: Основният клас за управление, имплементира Observable. Съдържа списък с медии, методи за добавяне, премахване, задаване на рейтинг (с уведомяване на observers и запис в JSON). filterByGenre връща Iterable с GenreIterator за безопасно итериране. Конструкторът зарежда от JSON.

- GUI пакет: MediaLibraryApp – основен клас за GUI, имплементира Observer за автоматично обновяване на таблицата при промени. Създава таблица с колони за медии и бутони за действия. AddMediaDialog – диалог за добавяне, с ChoiceBox за тип и динамични полета (напр. за Music – допълнително поле за trackLength).

- Util пакет: PersistenceUtil – статични методи за save (сериализация на списъка в JSON) и load (парсинг на JSON и създаване на обекти според полета, напр. ако има "author", създава Book).

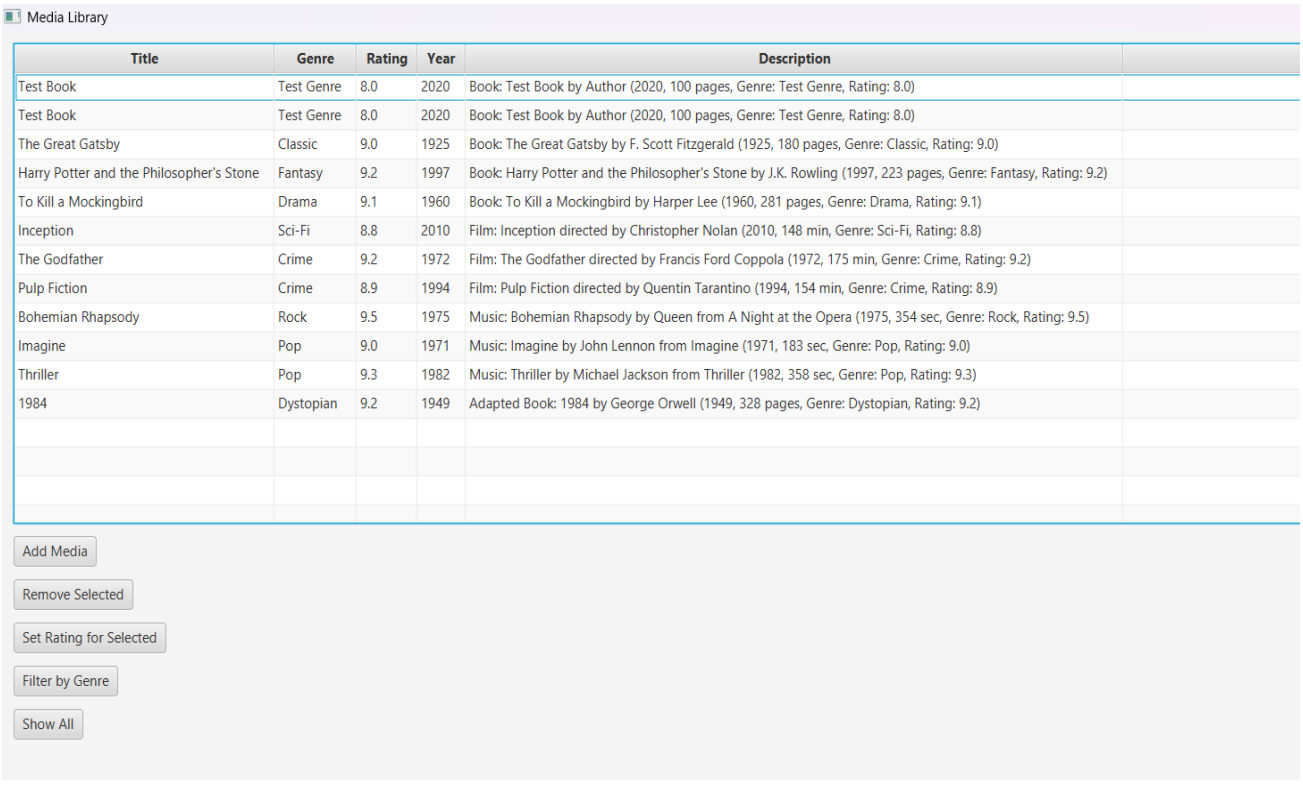
Описание на шаблоните:

- Iterator pattern: Използван за филтриране на медии по жанр без излагане на вътрешния списък. GenreIterator имплементира hasNext и next, прескачайки несъвпадащи елементи. Това осигурява encapsulation и лесно итератиране в GUI филтъра или конзола.

- Adapter pattern: Адаптира LegacyBook (с различни полета като bookTitle) към Media интерфейс. BookAdapter обвива LegacyBook и мапира методи (напр. getTitle връща getBookTitle). Демонстрира интеграция на стари класове в библиотеката, като се добавят уеднаквени в GUI или конзола.

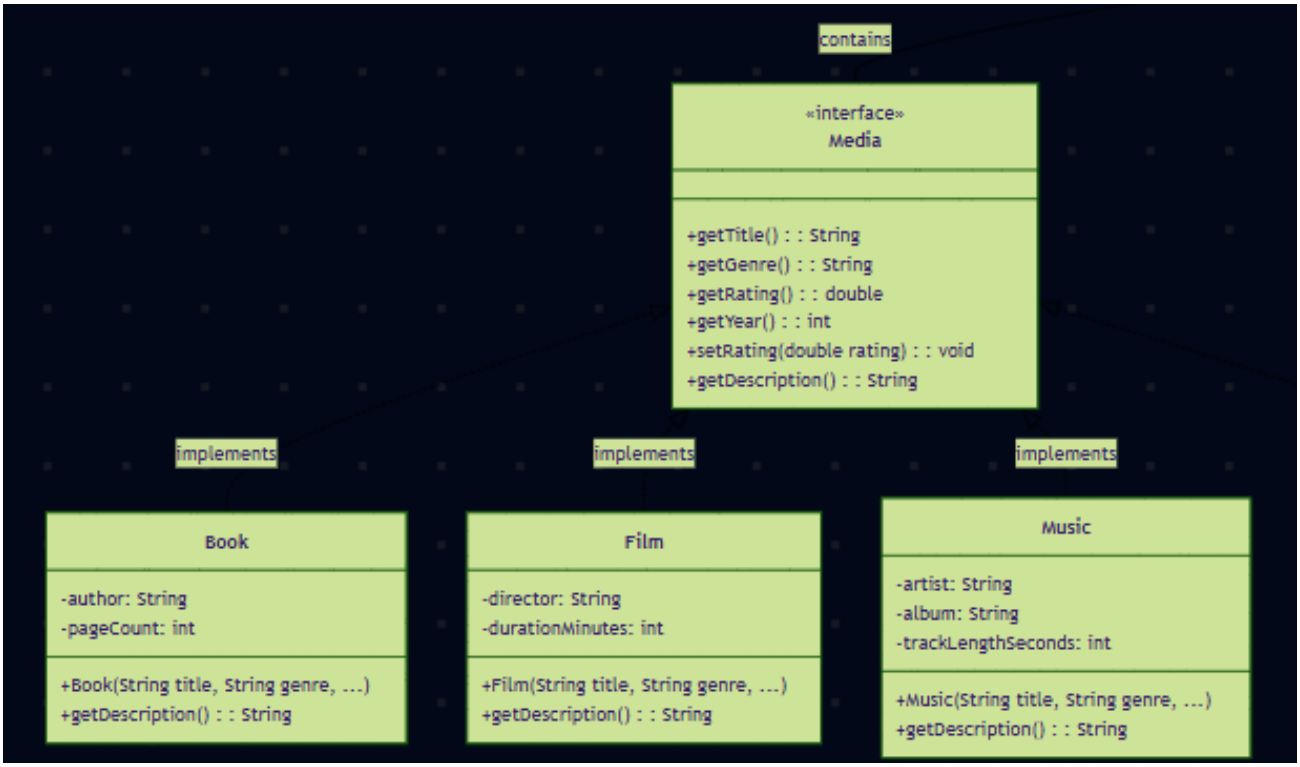
- Observer pattern (допълнителен): MediaLibrary уведомява observers при всяка промяна (add/remove/setRating). LibraryDisplayObserver печата в конзола, MediaLibraryApp обновява таблицата.

Графичен интерфейс:

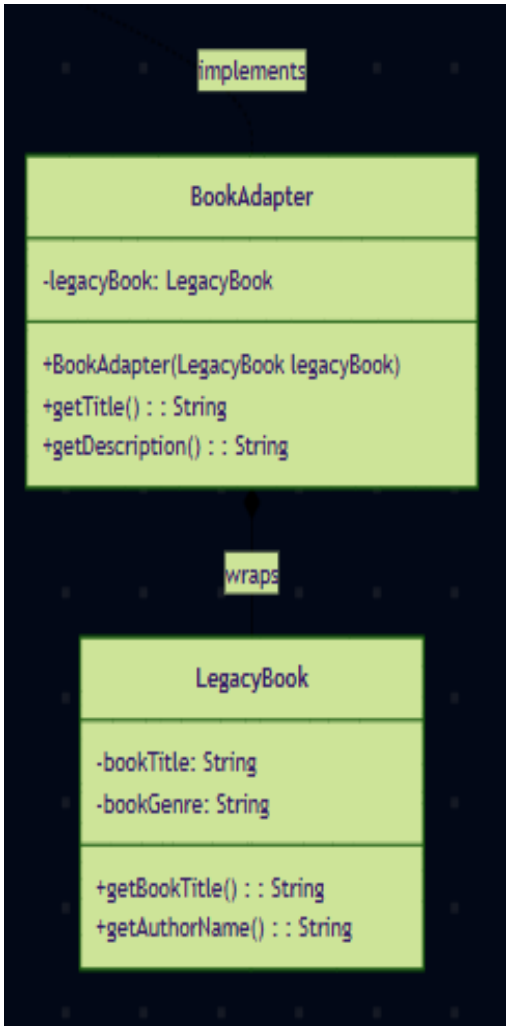


UML диаграма:

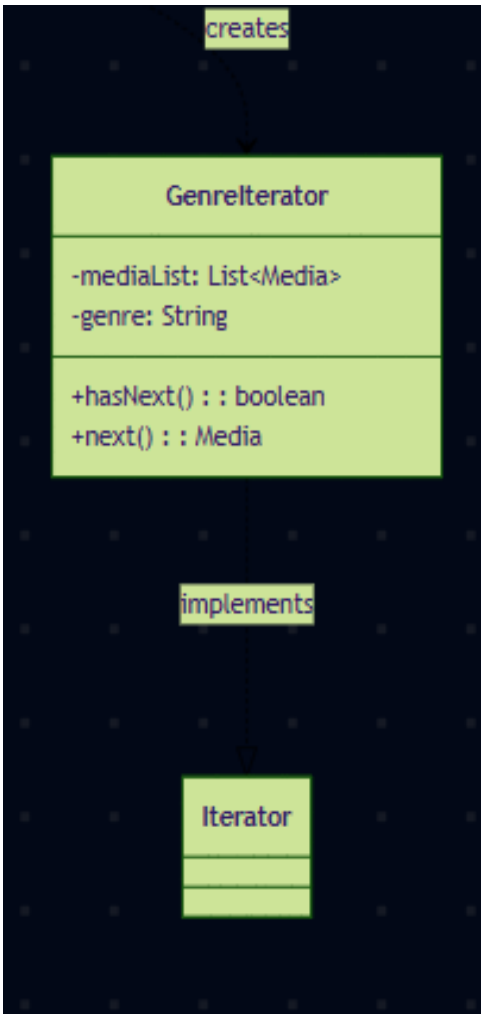
Основната медийна йерархия



Adapter Шаблон



Iterator Шаблон



Observer шаблон и цялостната система MediaLibrary