Music Playlist Manager

Project Report

Course Name:
Data Structures And Object Oriented Programming
Section. 00001

Student Name:
Kexian Li

Table of Contents

PROJECT DESCRIPTION
Music Playlist Manager

Scenario

This project is a music playlist manager. It allows users to explore, filter, and organize a large collection of songs sourced from a CSV file. The user can search songs by artist, genre, or release date, sort songs by title or popularity (number of plays), premium users have the added ability to create playlists and load playlists

Design Paradigm

This project demonstrates:

- Class hierarchy using inheritance
- An interface with an abstract method
- Polymorphism with method overriding and overloading
- Use of collections such as ArrayList, Map, and Set
- File I/O (reading and writing from CSV)
- Stream for sorting and filtering
- Exception handling
- Test-Driven Development (JUnit tests)
- Git for version control

Expected Output

- Load and parse `songs.csv` to display song data
- Filter and view songs by genre, artist, or release year
- Find out the top 10 artists/songs (by popularity) per year
- Search songs by name/artist name
- Sort songs by title or number of plays
- Create and manage playlists (for Premium users)
- Favorite songs/artists (For premium users)
- Show statistics like most played artist and average song duration
- Play ads for guest users when playing songs.

Class Hierarchies

Hierarchy 1: Songs

- `Song` (base class)

  - `PopSong`

- `RockSong`
- `CountrySong`

Hierarchy 2: Users

- `User` (base class)

- `GuestUser`
- `PremiumUser`

Interface

Interface: `Playable`

- Abstract method: `void play()`
- Implemented by: `Song` and `Ad`

Purpose: Defines a common behavior for anything that can be "played"

Runtime Polymorphism

- Method `play()` overridden in each subclass of `Song`, and `ad` (Example: when playing a pop song it can print out: “playing a pop hit”, but the message will be different for different genres)
- Method `toString()` overridden in `Song` and `User`
- Method overloading (addSong(Song), addSong(String title, String artist, …))
-

TextIO

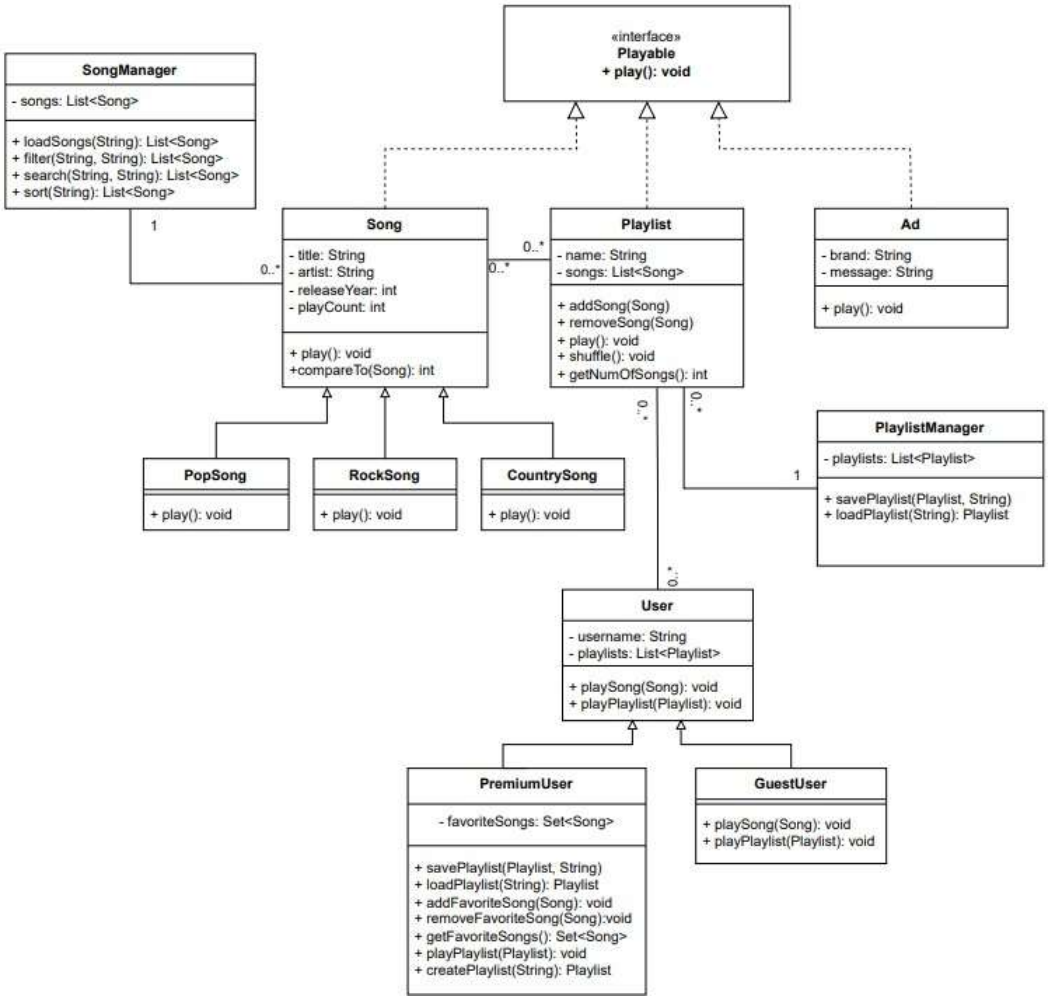
- `SongManager` reads songs from `songs.csv` and sort/filters the songs.
- `PlaylistManager` saves user playlists to a text or CSV file

Comparable and Comparator

- `Song` implements `Comparable<Song>` to sort by title
- Custom comparators:

- `CompareByReleaseDate`
- `CompareByPlayCount`

UML Class Diagram


«interface»
Playable
+ play(): void
SongManager
- songs: List<Song>
+ loadSongs(String): List<Song>
+ filter(String, String): List<Song>
+ search(String, String): List<Song>
+ sort(String): List<Song>
Song
- title: String
- artist: String
- releaseYear: int
- playCount: int
+ play(): void
+compareTo(Song): int
Playlist
- name: String
- songs: List<Song>
+ addSong(Song)
+ removeSong(Song)
+ play(): void
+ shuffle(): void
+ getNumOfSongs(): int
Ad
- brand: String
- message: String
+ play(): void
PopSong
+ play(): void
RockSong
+ play(): void
CountrySong
+ play(): void
PlaylistManager
- playlists: List<Playlist>
+ savePlaylist(Playlist, String)
+ loadPlaylist(String): Playlist
User
- username: String
- playlists: List<Playlist>
+ playSong(Song): void
+ playPlaylist(Playlist): void
PremiumUser
- favoriteSongs: Set<Song>
+ savePlaylist(Playlist, String)
+ loadPlaylist(String): Playlist
+ addFavoriteSong(Song): void
+ removeFavoriteSong(Song):void
+ getFavoriteSongs(): Set<Song>
+ playPlaylist(Playlist): void
+ createPlaylist(String): Playlist
GuestUser
+ playSong(Song): void
+ playPlaylist(Playlist): void
1
0..*
0..*
0..*
0..*
0..*
0..*
1

Program Features

1. Class Hierarchy

   Song -> PopSong, RockSong, CountrySong

   Each subclass overrides play()

```
Song pop = new PopSong( title: "pop",  artist: "popArtist",  releaseYear: 2025,  playCount: 0);
Song rock = new RockSong( title: "rock",  artist: "rockArtist",  releaseYear: 2025,  playCount: 0);
Song country = new CountrySong( title: "country",  artist: "countryArtist",  releaseYear: 2025,  playCount: 0);

User user = new PremiumUser( username: "TEST");
user.playSong(pop);
user.playSong(rock);
user.playSong(country);
```

```
Now playing a pop hit: "pop" by popArtist! Enjoy~
Rocking out to "rock" by rockArtist!Turn it up to 11!
Playing a heartfelt country tune: "country" by countryArtist. Yeehaw!
```

   User -> PremiumUser, GuestUser

```
User premium = new PremiumUser( username: "Premium");
User guest = new GuestUser( username: "Guest");
premium.playSong(pop);
premium.playSong(rock);
premium.playSong(country);
System.out.println("--------------GuestUser-----------------");
guest.playSong(pop);
guest.playSong(rock);
guest.playSong(country); //plays ad before the third song for guest users
```

```
Now playing a pop hit: "pop" by popArtist! Enjoy~
Rocking out to "rock" by rockArtist!Turn it up to 11!
Playing a heartfelt country tune: "country" by countryArtist. Yeehaw!
--------------GuestUser-----------------
Now playing a pop hit: "pop" by popArtist! Enjoy~
Rocking out to "rock" by rockArtist!Turn it up to 11!
Ad from Null: No ads found.
Playing a heartfelt country tune: "country" by countryArtist. Yeehaw!
```

2. Playlist creation and TextIO

```
List<Song> songs = new ArrayList<>(List.of(pop, rock, country));
PlaylistManager.createPlaylist( name: "TestPlaylist", songs);
guest.playPlaylist(PlaylistManager.findPlaylistByName("TestPlaylist"));
```

```
Now playing a pop hit: "pop" by popArtist! Enjoy~
Rocking out to "rock" by rockArtist!Turn it up to 11!
Ad from Null: No ads found.
Playing a heartfelt country tune: "country" by countryArtist. Yeehaw!
```

   Saving playlist to csv file

```
PlaylistManager.savePlaylistToFile(testPLaylist);
```

Song.java | Main.java | TestPlaylist.csv

```
pop,popArtist,2025,1,pop
rock,rockArtist,2025,1,rock
country,countryArtist,2025,1,country
```

3. Filtering & Searching

   Filter by different attributes of song:

```
Blinding Lights,The Weeknd,Pop,2020,3500000000
Shape of You,Ed Sheeran,Pop,2017,3800000000
Bohemian Rhapsody,Queen,Rock,1975,1800000000
Smells Like Teen Spirit,Nirvana,Rock,1991,1500000000
Take Me Home Country Roads,John Denver,Country,1971,1200000000
Friends in Low Places,Garth Brooks,Country,1990,900000000
Rolling in the Deep,Adele,Pop,2011,2000000000
Hotel California,Eagles,Rock,1976,1700000000
Jolene,Dolly Parton,Country,1973,800000000
Levitating,Dua Lipa,Pop,2020,2200000000
```

```
SongManager.loadFromCSV( file: "src/main/resources/songs.csv");
System.out.println(SongManager.filterByArtist( artistName: "The Weeknd"));
```

   Result:

```
[Title: Blinding Lights', Artist: The Weeknd', Release Year: 2020, Play Count: 3500000000, Genre: Pop]
```

   By Release Year:

```
SongManager.loadFromCSV( file: "src/main/resources/songs.csv");
System.out.println(SongManager.filterByReleaseYear(2020));
```

```
[Title: Blinding Lights', Artist: The Weeknd', Release Year: 2020, Play Count: 3500000000, Genre: Pop, Title: Levitating', Artist: Dua Lipa', Release Year: 2020, Play Count: 2200000000, Genre: Pop]
```

   Searching:

```
SongManager.loadFromCSV( file: "src/main/resources/songs.csv");
System.out.println(SongManager.search( searching: "levita"));
System.out.println(SongManager.search( searching: "sheeRAn"));
```

```
[Title: Levitating', Artist: Dua Lipa', Release Year: 2020, Play Count: 2200000000, Genre: Pop]
[Title: Shape of You', Artist: Ed Sheeran', Release Year: 2017, Play Count: 3800000000, Genre: Pop]
```

4. Other similar features include, loading, saving ads from/to CSV, Filtering by playcount/ release year range.
5. Getting top artists/songs.

```
SongManager.loadFromCSV( file: "src/main/resources/songs.csv");
System.out.println(SongManager.getTopArtists());
System.out.println(SongManager.getTopPlayed());
```

```
[Ed Sheeran, The Weeknd, Dua Lipa, Adele, Queen, Eagles, Nirvana, John Denver, Garth Brooks, Dolly Parton]
[Title: Shape of You', Artist: Ed Sheeran', Release Year: 2017, Play Count: 3800000000, Genre: Pop, Title: Blinding Lights', Artist: The Weeknd', Release Year: 2
```

6. Playcount increases with each play

```
Song song = SongManager.getSongs().get(1);
premium.playSong(song);
System.out.println(song.getPlayCount());
premium.playSong(song);
System.out.println(song.getPlayCount());
```

```
Now playing a pop hit: "Shape of You" by Ed Sheeran! Enjoy~
3800000001
Now playing a pop hit: "Shape of You" by Ed Sheeran! Enjoy~
3800000002
```

7. Create and manage playlist

```
premium.createPlaylist( name: "NewPlaylist");
premium.addSongToPlaylist(PlaylistManager.findPlaylistByName("NewPlaylist"), pop);
System.out.println("Number of songs in playlist: " + PlaylistManager.findPlaylistByName("NewPlaylist").getNumOfSongs());
```

```
Number of songs in playlist: 1
```

8. Song sorting

```
System.out.println(SongManager.getSongs());
System.out.println(SongManager.sortByReleaseYearDescending());
```

```
[Title: Blinding Lights', Artist: The Weeknd', Release Year: 2020, Play Count: 3500000000
, Genre: Pop, Title: Shape of You', Artist: Ed Sheeran', Release Year: 2017, Play Count: 3800000000
, Genre: Pop, Title: Bohemian Rhapsody', Artist: Queen', Release Year: 1975, Play Count: 1800000000
, Genre: Rock, Title: Smells Like Teen Spirit', Artist: Nirvana', Release Year: 1991, Play Count: 1500000000
, Genre: Rock, Title: Take Me Home Country Roads', Artist: John Denver', Release Year: 1971, Play Count: 1200000000
, Genre: Country, Title: Friends in Low Places', Artist: Garth Brooks', Release Year: 1990, Play Count: 900000000
, Genre: Country, Title: Rolling in the Deep', Artist: Adele', Release Year: 2011, Play Count: 2000000000
, Genre: Pop, Title: Hotel California', Artist: Eagles', Release Year: 1976, Play Count: 1700000000
, Genre: Rock, Title: Jolene', Artist: Dolly Parton', Release Year: 1973, Play Count: 800000000
, Genre: Country, Title: Levitating', Artist: Dua Lipa', Release Year: 2020, Play Count: 2200000000
, Genre: Pop]
[Title: Blinding Lights', Artist: The Weeknd', Release Year: 2020, Play Count: 3500000000
, Genre: Pop, Title: Levitating', Artist: Dua Lipa', Release Year: 2020, Play Count: 2200000000
, Genre: Pop, Title: Shape of You', Artist: Ed Sheeran', Release Year: 2017, Play Count: 3800000000
, Genre: Pop, Title: Rolling in the Deep', Artist: Adele', Release Year: 2011, Play Count: 2000000000
, Genre: Pop, Title: Smells Like Teen Spirit', Artist: Nirvana', Release Year: 1991, Play Count: 1500000000
, Genre: Rock, Title: Friends in Low Places', Artist: Garth Brooks', Release Year: 1990, Play Count: 900000000
, Genre: Country, Title: Hotel California', Artist: Eagles', Release Year: 1976, Play Count: 1700000000
, Genre: Rock, Title: Bohemian Rhapsody', Artist: Queen', Release Year: 1975, Play Count: 1800000000
, Genre: Rock, Title: Jolene', Artist: Dolly Parton', Release Year: 1973, Play Count: 800000000
, Genre: Country, Title: Take Me Home Country Roads', Artist: John Denver', Release Year: 1971, Play Count: 1200000000
, Genre: Country]
```

9. Other similar, or simple features such as shuffle play, getting number of songs in a playlist, searching for a playlist by its name, managing favorite songs, etc.

Challenges

I came across many challenges during the development of this project:

1. I originally designed playCount to be an integer, but then during the development, I realized that the numbers can be too large for int, and decided to switch to long.
2. Understanding which class should have access to which methods of other classes to ensure that PremiumUser's features are only accessible by premiumUser was a challenge I faced
3. Using the proper data structure for different purposes was a challenge, such as, I originally decided to use ArrayList, then later switch to Set to avoid duplication.
4. Understanding TextIO was a major challenge I faced, trying to read, write, and manipulate data from csv files efficiently.
5. Trying to be efficient with the usage of different methods was challenging, as well as being efficient in writing my own user defined methods. For example, at one point, I wrote methods in different Classes that did the same thing.
6. Exception handling, especially when it comes to TextIO, was challenging for me to understand fully.
7. I decided to use subclasses to differentiate different music genres, which caused problems later on when trying to develop methods to write Songs to a csv file, as well as reading and creating Song objects from a csv file.
8. I decided to not implement features relating to song duration, mostly due to not finding the feature useful enough.

Learning Outcomes

Gained practical experience with using lambdas & streams.

Increased understanding of encapsulation.

Getting used to using Junit Tests.

Creating helper methods instead of writing everything within the same method

Gained practical experience on debugging.

Learned how to properly create an UML Class Diagram.
Learned how to use Git Repository for version control.