TUGAS W04: PRAKTIKUM REFACTORING

MANAJEMEN KONFIGURASI DAN EVOLUSI PERANGKAT LUNAK

Disusun Oleh :

Lintang Suminar Tyas Wening – 2211104009

Dosen Pengampu :

Yudha Islami Sulistya, S.Kom., M.Cs

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

DESKRIPSI TUGAS

Berdasarkan class Song.Java yang tercantum pada assignment ini lakukan hal-hal berikut ini:

1. Identifikasi bad smell yang ada pada class tersebut!

2. Lakukan refactoring terhadap class Song.Java sesuai dengan bad smell yang dilakukan, silahkan buat method dan class baru jika dibutuhkan!

Laporkan bad smell yang ditemukan dan refactoring yang dilakukan dalam sebuah file pdf dan link github

Jawab :

1. Berdasarkan kode Song.java, beberapa bad smell yang ditemukan ialah :
    - Large Class : Pada class Song.java terlalu banyak atribut yang mencakup informasi tentang lagu, album, dan artis. Sehingga harus dipecah menjadi beberapa bagian kelas kecil yang lebih spesifik

```java
package Assignment;


public class Song {

    private String id;
    private String title;
    private String releaseYear;
    private String musicFileURL;
    private int genre;

    private String albumName;
    private String albumCoverURL;

    private String artistName;
    private String artistAlias;
    private String artistImageURI;

    public Song(String id, String title, String releaseYear, String musicFileURL) {
        this.id = id;
        this.title = title;
        this.releaseYear = releaseYear;
        this.musicFileURL = musicFileURL;
    }

    public void setAlbum(String albumName, String albumCoverURL) {
        this.albumName = albumName;
        this.albumCoverURL = albumCoverURL;
    }

    public void setArtist(String artistName, String artistAlias, String artistImageURL) {
        this.artistName = artistName;
        this.artistAlias = artistAlias;
        this.artistImageURI = artistImageURI;
    }

    /**
     * Set the genre of this song
     *
     * 0 - undefined
     * 1 - pop
     * 2 - rock
     * 3 - hip hop
     * 4 - RnB
     * 5 - jazz
     * 6 - instrumentals
     * 7 - clowncore
     *
     * @param genre
     */
    public void setGenre(int genre) {
        this.genre = genre;
    }

    /**
     * Print info of the song based on desired detail level
     *
     * 0 - song info only
     * 1 - song info and artist info
     * 2 - song info and album info
     * 3 - song, artist, and album info
     *
     * @param genre
     */
    public void printInfo(int detailLevel) {
        if (detailLevel == 0) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
        }else if(detailLevel == 1) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
            if (!artistName.equals("")) {
                System.out.println("artist name: " + artistName);
            }
            if (!artistAlias.equals("")) {
                System.out.println("artist also known as: " + artistAlias);
            }
        }else if (detailLevel == 2) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
            if (!albumName.equals("")) {
                System.out.println("album title: " + albumName);
            }
        }else if (detailLevel == 3) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
            if (!artistName.equals("")) {
                System.out.println("artist name: " + artistName);
            }
            if (!artistAlias.equals("")) {
                System.out.println("artist also known as: " + artistAlias);
            }
            if (!albumName.equals("")) {
                System.out.println("album title: " + albumName);
            }
        }
    }
}
```

- Primitive Obsession (Menggunakan Genre sebagai int) : Menggunakan integer untuk genre tidak aman dikarenakan genre ditunjukan sebagai int dengan nilai tetap (angka 0-7). Hal ini menyebabkan kode sulit dibaca dan rawan kesalahan sehingga lebih baik menggunakan enum

```
10        private int genre;
11
```

```
/**
 * Set the genre of this song
 *
 * 0 = undefined
 * 1 = pop
 * 2 = rock
 * 3 = hip hop
 * 4 = RnB
 * 5 = jazz
 * 6 = instrumentals
 * 7 = clowncore
 *
 * @param genre
 */
public void setGenre(int genre) {
    this.genre = genre;
}
```

- Long Method : Pada fungsi printinfo(int detailLevel) memiliki banyak kondisi if-else yang dapat menyebabkan code duplication sehingga sulit dibaca dan dipahami. Untuk itu diharapkan dapat dipecah menjadi beberapa method

```
public void printInfo(int detailLevel) {
        if (detailLevel == 0) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
        }else if(detailLevel == 1) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
            if (!artistName.equals("")) {
                System.out.println("artist name: " + artistName);
            }
            if (!artistAlias.equals("")) {
                System.out.println("artist also known as: " + artistAlias);
            }
        }else if (detailLevel == 2) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
            if (!albumName.equals("")) {
                System.out.println("album title: " + albumName);
            }
        }else if (detailLevel == 3) {
            System.out.println("song title: " + title);
            System.out.println("release year: " + releaseYear);
            if (genre > 0) {
                System.out.println("genre: " + genre);
            }
            if (!artistName.equals("")) {
                System.out.println("artist name: " + artistName);
            }
            if (!artistAlias.equals("")) {
                System.out.println("artist also known as: " + artistAlias);
            }
            if (!albumName.equals("")) {
                System.out.println("album title: " + albumName);
            }
        }
```

- Long Parameter List : Class song terdapat parameter yang panjang membuat kode sulit dipahami saat memanggil konstruktor, sehingga yang harus dilakukan yaitu membuat kelas terpisah seperti artist dan album

```
public class Song {

    private String id;
    private String title;
    private String releaseYear;
    private String musicFileURL;
    private int genre;

    private String albumName;
    private String albumCoverURL;

    private String artistName;
    private String artistAlias;
    private String artistImageURL;
```

- Data Clump : Pada class Song terdapat atribut yang sering digunakan atau muncul bersama seperti Atribut artistName, artistAlias, dan artistImageURL begitu pula albumName dan albumCoverURL. Untuk itu sebaiknya lebih baik dipisahkan dalam class mereka masing-masing

2. Refactoring pada class Song :
   - ❖ Membuat kelas terpisah untuk Artis dan Album agar lebih terstruktur
   - ❖ Menggunakan enum untuk Genre agar lebih mudah dipahami dan tidak menggunakan angka tanpa makna yg jelas (magic numbers)
   - ❖ Refactor method printInfo agar lebih modular

   Implementasi refactoring :

   - Album.java

```java
package sesudah_refactoring;

public class Album {
    private String name;
    private String coverURL;

    public Album(String name, String coverURL) {
        this.name = name;
        this.coverURL = coverURL;
    }

    public void printInfo() {
        System.out.println("Album Title: " + name);
    }

}
```

   - Artist.java

```java
package sesudah_refactoring;

public class Artist {
    private String name;
    private String alias;
    private String imageURL;

    public Artist(String name, String alias, String imageURL) {
        this.name = name;
        this.alias = alias;
        this.imageURL = imageURL;
    }

    public void printInfo() {
        System.out.println("Artist Name: " + name);
        if (!alias.isEmpty()) {
            System.out.println("Also Known As: " + alias);
        }
    }

}
```

   - Genre.java

```java
package sesudah_refactoring;

public enum Genre {
    UNDEFINED, KPOP, POP, ROCK, HIP_HOP, RNB, JAZZ, INSTRUMENTALS, CLOWNCORE;
}
```

- Song.java

```java
package sesudah_refactoring;

public class Song {
    private String id;
    private String title;
    private String releaseYear;
    private String musicFileURL;
    private Genre genre;
    private Album album;
    private Artist artist;

    public Song(String id, String title, String releaseYear, String musicFileURL) {
        this.id = id;
        this.title = title;
        this.releaseYear = releaseYear;
        this.musicFileURL = musicFileURL;
        this.genre = Genre.UNDEFINED;
    }

    public void setAlbum(Album album) {
        this.album = album;
    }

    public void setArtist(Artist artist) {
        this.artist = artist;
    }

    public void setGenre(Genre genre) {
        this.genre = genre;
    }

    public void printInfo(int detailLevel) {
        System.out.println("Song Title: " + title);
        System.out.println("Release Year: " + releaseYear);
        if (genre != Genre.UNDEFINED) {
            System.out.println("Genre: " + genre);
        }

        if (detailLevel >= 1 && artist != null) {
            artist.printInfo();
        }
        if (detailLevel >= 2 && album != null) {
            album.printInfo();
        }
    }
}
```

- Main.java (Untuk menjalankan program dan memastikan semua class dapat dihubungkan tanpa erorr)

```java
package sesudah_refactoring;

public class Main {
    public static void main(String[] args) {
        // Membuat objek Artist
        Artist artist = new Artist("Seventeen", "Sebong",
                "https://www.allkpop.com/upload/2023/09/content/191059/web_data/allkpop_1695136312_untitled-1.jpg");

        // Membuat objek Album
        Album album = new Album("Face The Sun",
                "https://is2-ssl.mzstatic.com/image/thumb/Music122/v4/3a/a8/58/3aa8588c-0af9-a0f0-08b5-9d1baf0f5736/196922060847_Cover.jpg/1200x1200bf-60.jpg");

        // Membuat objek Song
        Song song = new Song("001", "HOT", "2022", "https://youtu.be/gRnuFC4Ualw?feature=shared");

        // Menghubungkan Song dengan Artist dan Album
        song.setArtist(artist);
        song.setAlbum(album);
        song.setGenre(Genre.KPOP);

        // Menampilkan informasi lagu dengan detail lengkap
        System.out.println("=== Detail Level 3 (Full Info) ===");
        song.printInfo(3);

        System.out.println("\n=== Detail Level 1 (Song + Artist) ===");
        song.printInfo(1);

        System.out.println("\n=== Detail Level 2 (Song + Album) ===");
        song.printInfo(2);

        System.out.println("\n=== Detail Level 0 (Song Only) ===");
        song.printInfo(0);
    }

}
```

Hasil Output :

```
PS D:\SEMESTER 6\mkepl\Praktikum_Refactoring>  d:; cd 'd:\SEMESTER 6\
STER 6\mkepl\Praktikum_Refactoring\bin' 'sesudah_refactoring.Main'
=== Detail Level 3 (Full Info) ===
Song Title: HOT
Release Year: 2022
Genre: KPOP
Artist Name: Seventeen
Also Known As: Sebong
Album Title: Face The Sun

=== Detail Level 1 (Song + Artist) ===
Song Title: HOT
Release Year: 2022
Genre: KPOP
Artist Name: Seventeen
Also Known As: Sebong

=== Detail Level 2 (Song + Album) ===
Song Title: HOT
Release Year: 2022
Genre: KPOP
Artist Name: Seventeen
Also Known As: Sebong
Album Title: Face The Sun

=== Detail Level 0 (Song Only) ===
Song Title: HOT
Release Year: 2022
Genre: KPOP
PS D:\SEMESTER 6\mkepl\Praktikum_Refactoring>
```

Link Github : https://github.com/LintangSuminar02/MKEPL-TEORI_LINTANG-SUMINAR-TYAS-WENING_2211104009_SE0601