TUGAS WEEK 04
PRAKTIKUM REFACTORING

MANAJEMEN KONFIGURASI EVOLUSI DAN PERANGKAT LUNAK

Disusun Oleh:

Aorinka Anendya Chazanah / 2211104013

S1 SE-06-01

Dosen Pengampu:

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

TUGAS WEEK 04

A. Identifikasi Bad Smell dalam Class Song

1. **God Class**

   Song memiliki terlalu banyak tanggung jawab: menyimpan informasi lagu, album, dan artis sekaligus. Ini menyebabkan class menjadi besar dan sulit dikelola.

2. **Primitive Obsession**

   Genre disimpan sebagai int dengan nilai tetap (hardcoded), lebih baik menggunakan enum untuk meningkatkan keterbacaan kode.

3. **Long Method (printInfo)**

   printInfo(int detailLevel) memiliki banyak kondisi if-else yang membuatnya sulit dibaca dan dipelihara. Harus dipisah menjadi metode lebih kecil.

4. **Data Clump**

   Informasi album dan artis sering digunakan bersama-sama, lebih baik dipisah ke class Album dan Artist.

B. Langkah-langkah Refactoring

1. **Memisahkan Album dan Artist dari Song**

   Masalah:

   - Song menyimpan terlalu banyak informasi yang tidak berkaitan langsung, seperti detail album dan artis.
   - Hal ini menyebabkan Song memiliki terlalu banyak tanggung jawab (**God Class**).

   Solusi:

   - Membuat class Album untuk menyimpan informasi album.
   - Membuat class Artist untuk menyimpan informasi artis.

   Kode sebelum refactoring

```
private String albumName;
private String albumCoverURL;

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

Kode setelah refactoring

```java
// Class untuk Album
class Album {
    private String name;
    private String coverURL;

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

    public String getName() {
        return name;
    }

    public String getCoverURL() {
        return coverURL;
    }
}
```

```java
// Class untuk Artist
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 String getName() {
        return name;
    }

    public String getAlias() {
        return alias;
    }

    public String getImageURL() {
        return imageURL;
    }
}
```

2. **Menggunakan Enum untuk Genre**

Masalah:

- Genre direpresentasikan sebagai int dengan angka tetap (Primitive Obsession).
- Penggunaan angka untuk genre membuat kode sulit dipahami dan rawan kesalahan.

Solusi:

- Menggunakan enum untuk menggantikan int genre.

Kode sebelum refactoring

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

Kode setelah refactoring

```
// Enum untuk Genre
enum GenreType {
    UNDEFINED, POP, ROCK, HIPHOP, RNB, JAZZ, INSTRUMENTALS, CLOWNCORE;
}
```

Pada class song

```
// Class untuk Song
class Song {
    private String title;
    private int releaseYear;
    private GenreType genre;
    private Album album;
    private Artist artist;

    public Song(String title, int releaseYear, GenreType genre, Album album, Artist artist) {
        this.title = title;
        this.releaseYear = releaseYear;
        this.genre = genre != null ? genre : GenreType.UNDEFINED;
        this.album = album;
        this.artist = artist;
    }
```

3. **Refaktor printInfo menjadi Beberapa Metode Kecil**

Masalah:

- printInfo menggunakan banyak if-else, membuatnya panjang dan sulit dibaca (Long Method).
- Setiap bagian (lagu, artis, album) sebaiknya memiliki metode sendiri.

Solusi:

- Memisahkan pencetakan ke tiga metode kecil:
    - printBasicInfo()
    - printArtistInfo()
    - printAlbumInfo()

Kode sebelum refactoring

```
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);
        }
```

Kode setelah refactoring

```
    public void printInfo(int detailLevel) {
        printBasicInfo();
        if (detailLevel >= 1) {
            printArtistInfo();
        }
        if (detailLevel >= 2) {
            printAlbumInfo();
        }
    }

    private void printBasicInfo() {
        System.out.println("Song Title: " + title);
        System.out.println("Release Year: " + releaseYear);
        if (genre != GenreType.UNDEFINED) {
            System.out.println("Genre: " + genre);
        }
    }

    private void printArtistInfo() {
        if (artist != null) {
            System.out.println("Artist Name: " + artist.getName());
            if (!artist.getAlias().isEmpty()) {
                System.out.println("Also Known As: " + artist.getAlias());
            }
        }
    }

    private void printAlbumInfo() {
        if (album != null) {
            System.out.println("Album Title: " + album.getName());
        }
    }
}
```

C. Hasil Running Program setelah Refactoring

```
Listening on 55570
User program running
=== Detail Level 0 ===
Song Title: My Song
Release Year: 2022
Genre: POP

=== Detail Level 1 ===
Song Title: My Song
Release Year: 2022
Genre: POP
Artist Name: John Doe
Also Known As: JD

=== Detail Level 2 ===
Song Title: My Song
Release Year: 2022
Genre: POP
Artist Name: John Doe
Also Known As: JD
Album Title: Greatest Hits
```

D. Full Kode Song.java setelah Refactoring

```
// Enum untuk Genre
enum GenreType {
    UNDEFINED, POP, ROCK, HIPHOP, RNB, JAZZ, INSTRUMENTALS, CLOWNCORE;
}

// Class untuk Album
class Album {
    private String name;
    private String coverURL;

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

    public String getName() {
        return name;
    }

    public String getCoverURL() {
        return coverURL;
    }
}

// Class untuk Artist
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 String getName() {
        return name;
    }

    public String getAlias() {
        return alias;
    }

    public String getImageURL() {
        return imageURL;
    }
}

// Class untuk Song
class Song {
    private String title;
    private int releaseYear;
    private GenreType genre;
    private Album album;
    private Artist artist;

    public Song(String title, int releaseYear, GenreType genre, Album album,
Artist artist) {
        this.title = title;
        this.releaseYear = releaseYear;
        this.genre = genre != null ? genre : GenreType.UNDEFINED;
        this.album = album;
        this.artist = artist;
    }

    public void printInfo(int detailLevel) {
        printBasicInfo();
        if (detailLevel >= 1) {
            printArtistInfo();
        }
        if (detailLevel >= 2) {
            printAlbumInfo();
        }
    }

    private void printBasicInfo() {
        System.out.println("Song Title: " + title);
        System.out.println("Release Year: " + releaseYear);
        if (genre != GenreType.UNDEFINED) {
            System.out.println("Genre: " + genre);
        }
    }

    private void printArtistInfo() {
        if (artist != null) {
            System.out.println("Artist Name: " + artist.getName());
            if (!artist.getAlias().isEmpty()) {
                System.out.println("Also Known As: " + artist.getAlias());
```

```
            }
        }
    }

    private void printAlbumInfo() {
        if (album != null) {
            System.out.println("Album Title: " + album.getName());
        }
    }
}

// Main Class untuk Testing
public class Main {
    public static void main(String[] args) {
        // Membuat objek Artist dan Album
        Artist artist = new Artist("John Doe", "JD",
"https://image.url/john.jpg");
        Album album = new Album("Greatest Hits",
"https://image.url/album.jpg");

        // Membuat objek Song
        Song song = new Song("My Song", 2022, GenreType.POP, album, artist);

        // Menampilkan informasi lagu dengan berbagai detail level
        System.out.println("=== Detail Level 0 ===");
        song.printInfo(0);

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

        System.out.println("\n=== Detail Level 2 ===");
        song.printInfo(2);
    }
}
```