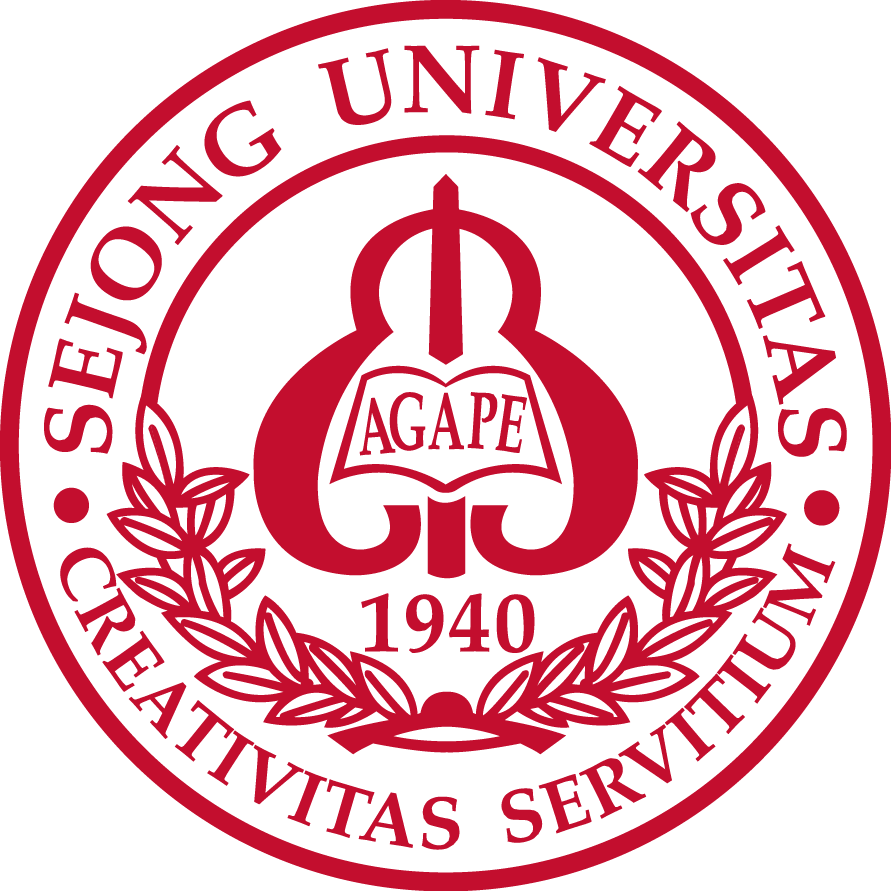
실력만은 20학번

문제해결기법 최종보고서

03조



|  |  |  |
| --- | --- | --- |
| 팀 장 |  |  |
|  | 최인호 | 16011039 |
| 팀 원 |  |  |
|  | 안건우 | 10611029 |
|  | 박기춘 | 16011035 |
|  | 김성수 | 16011021 |

목차

|  |  |  |
| --- | --- | --- |
| 프로젝트 소개 |  | 1 |
|  |
| 팀원 별 리펙토링 내용 |  |  |
| * 최인호 |  | 3 |
|  |
| * 안건우 |  | 8 |
|  |
| * 박기춘 |  | 15 |
|  |
| * 김성수 |  | 21 |
|  |

## 개인별 리펙토링 세부 내용

1. 담당 파트

2. 일정

3. 리펙토링 내용

4. 개인소감

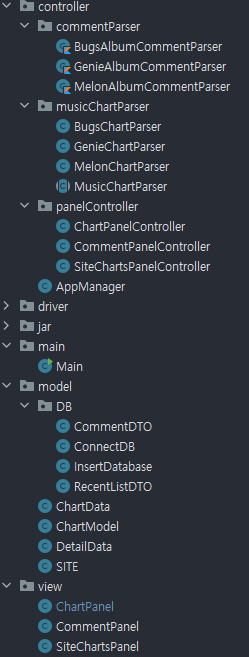
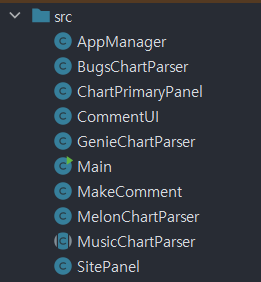
# 1. 프로젝트 소개

## 프로젝트 의의

각 사이트의 TOP 100 차트 및 앨범 별 댓글을 보여주는 프로그램

⁕각 사이트 : 멜론, 벅스, 지니

오리지널 프로젝트의 문제점



< MVC 분리 이전 이후 클래스 변화 >

DB 부재 -> DB 생성 및 작성

\*.txt -> DB

MVC 패턴 분리 필요성 ->Model / View / Controller 분리

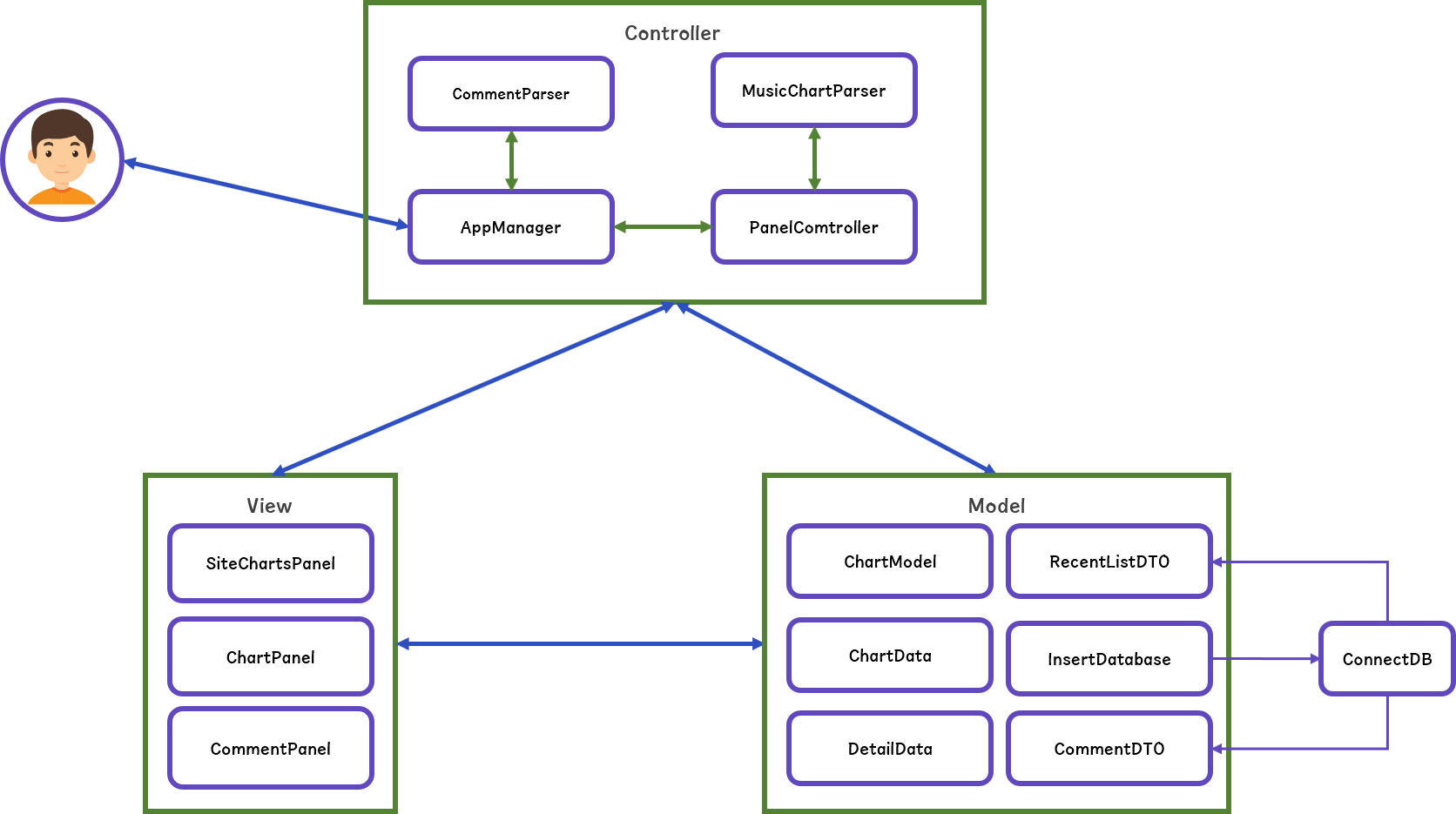
## 사용된 프레임워크 및 라이브러리

### 에돌이의 얕고 넓은 샘 :: jsoup - 자바를 위한 BeautifulSoup (HTML parser)웹 크롤러

### 아마존 웹 서비스 - 위키백과, 우리 모두의 백과사전DB



## 리펙토링 이후 프로젝트 구조도

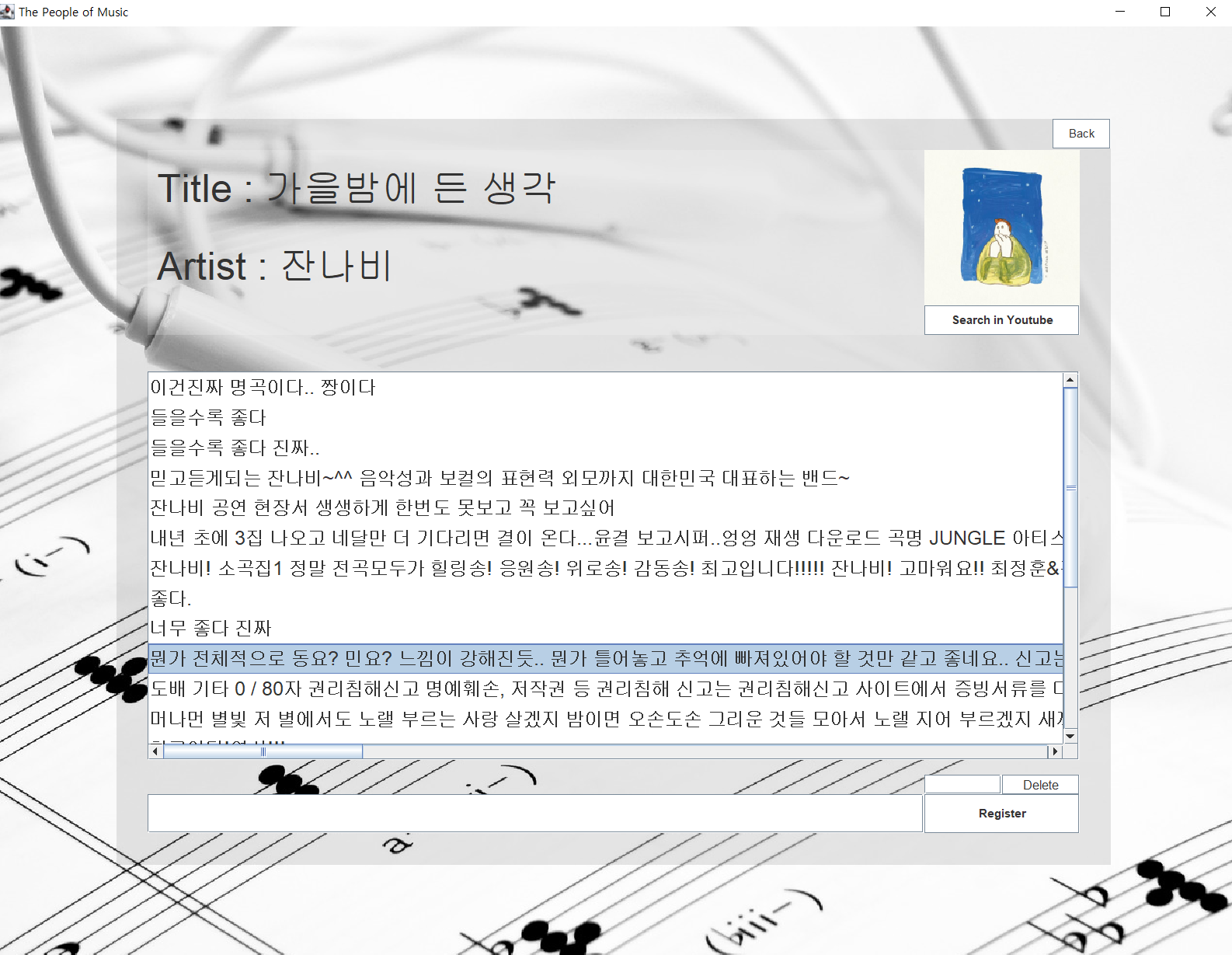


< 리팩토링 이후 전체 구조도 >

## 프로그램 작동 사진 및 영상

웹 크롤링 영상 [[ 링크 ]](https://sejonguniversity-my.sharepoint.com/:v:/g/personal/eknow25_sju_ac_kr/EYdgKPDokwVHmQksBK7xpr0BD2Ag_Waa0ftfCwAe8OezXA)

< 프로젝트 시연 사진 >



# 최인호 리펙토링 내용

## 담당 파트

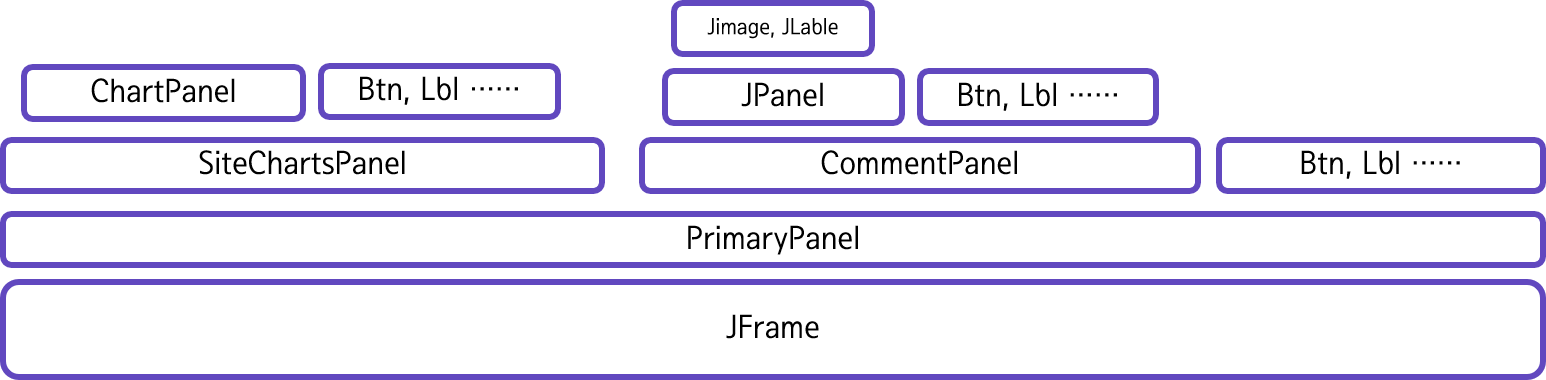
* View
* CommentParser

## 일정

|  |  |
| --- | --- |
| 9 주차 | MVC패턴 분리View 리펙토링 |
| 10 주차 |
| 11 주차 | 프로젝트 전체 조율 및 역할 분배 |
| 12 주차 | CommentParser 생성 |
| 13 주차 | CommentParser 리펙토링 |
| 14 주차 | CodeReview 피드백 반영 및 프로젝트 전체 점검 |

## View 리펙토링

구조도 개선



각자 선언되어 있던 Panel 클래스들을 JFrame 위에 PrimaryPanel 위로 올린 후 적층 적인 구조로 리펙토링을 진행했습니다.

## 1. ChartPanel

### 1. SitePanel -> ChartPanel 클래스 명 변경

Site 라는 이중적인 표현 대신에 정확히 Chart를 띄워주는 클래스로 ChartPanel로 클래스명 변경

### 2. constructor 내 메소드 분리

constructor에 너무 많은 변수들이 선언되고 사용되고 있기에 메소드 분리를 진행했습니다.

### 3. Inner Class분리

차트 모델이 되는 이내 클래스를 Model 파트로 클래스 분리를 진행했습니다.

### 4. Switch문 제거

switch 문을 static final 선언으로 전환하여서 한 눈에 보기 쉽게 리펙토링을 진행했습니다.

### 5. 메소드 명 변경



filter 라는 메소드 명은 어떤 것을 구체적으로 filter 해주는지 명확하지 않기에 메소드 명을 변경

#### 6. 계층적 구조 변경

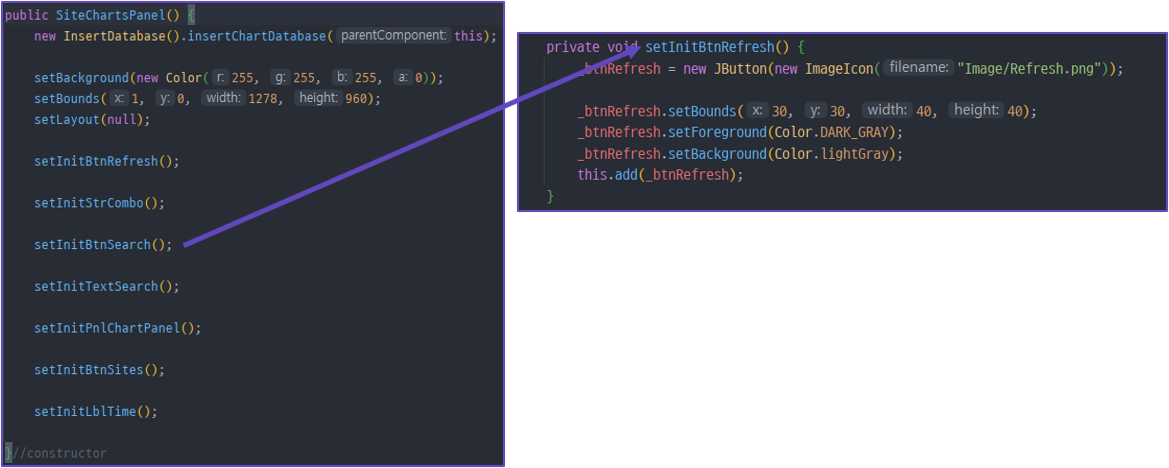
TableChart위에 TableModel이 있듯 TableChart를 정의하는 메소드에서 TableModel를  
선언하는 메소드를 호출하게끔 리펙토링을 했습니다.

## 2. SiteChartsPanel

### 1. ChartPrimaryPanel -> SiteChartsPanel 클래스명 변경

PrimaryPanel은 이미 존재하는 JPanel이기에 Chart를 모아둔 클래스이기에 클래스명 변경

### 2. Constructor 내 메소드 분리



constructor에 너무 많은 변수들이 선언되고 사용되고 있기에 메소드 분리를 진행했습니다.

## 3. CommentPanel

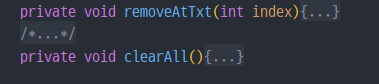
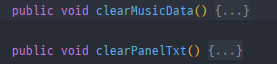
### 1. CommentUI -> CommentPanel

CommentUI 보다는 통일성 있는 클래스명 선언을 위해 CommentPanel로 클래스명 변경

### 2. 계층적 구조 변경

JPanel위에 올라가서 사용되는 것들은 해당 JPanel선언 하는 메소드에서 호출하도록 리펙토링 진행

### 3. 메소드 명 변경



어떤 txt를 제거하는지 어떤 것을 지우는지 명확하지 않은 메소드는 메소드 명 변경을 진행했습니다.

# 4. CommentParser

## 1. 기능 추가 이유

초기 프로젝트 진행 시 댓글을 크롤링하여서 보여주고 싶었으나 어떻게 가져와야 할 지 몰라서 포기했지만, 본 프로젝트 의미에 맞는 기능이기에 추가했습니다.

## 2. 사용된 라이브러리 및 언어

사용 언어 : 코틀린

사용된 프레임 워크 : Selinium ( 자바스크립트 실행 이후 댓글을 띄워주기에 Selinium을 사용했습니다.)

사용한 라이브러리 : Jsoup

## 3. 리펙토링 내용

### 1. 공통 변수 제거

초기에는 각 클래스마다 크롬드라이버를 구현해서 생성했지만 크롬드라이버를 구동 시키는 작업은 무겁기에 리펙토링을 진행하면서 공통적인 크롬드라이버를 선언부에서 파라미터로 전달하는 형식으로 리펙토링을 진행했습니다.

### 2. 메소드 분리



각 차트 앨범 사이트를 구분하는 ID를 Set로 반환하는 메소드를 분리를 리펙토링하면서 같은 앨범을 파싱하는 일을 사전에 방지했습니다.

Jsoup에서 얻어온 html파일을 DB에 넣을 수 있게끔 필터링하는 메소드를 분리했습니다.

## 개인 소감

전에는 자신의 코드를 작성하기만 했지 되돌아 본적은 없었습니다. 이번 수업 덕분에 리펙토링을 접할 수 있어서 새로웠습니다. 코드의 작성에 답은 없지만 옳은 길은 있다는 것도 조금은 깨달은 것 같습니다. 특히 이번 프로젝트에서 리펙토링뿐만 아니라 기능추가를 함께 했습니다. 기존 프로젝트의 기능 추가와 리펙토링 후 기능추가는 달랐습니다. 기존에는 기능을 구현해 놓고 그 기능을 추가하는데 상당한 에러들이 많이 발생하였었습니다. 변수가 맞지 않거나, 기능을 추가하기 위해 다른 클래스 곳곳에 코드를 추가하니 코드가 어지러웠지만, 이번에 리펙토링 후 기능을 추가하니 어디에 코드가 있어야 하는지 알기에 기존 코드에 추가하기가 깔끔했고, 새로운 코드 작성하면서도 클린 코드를 작성하도록 노력을 하게 된다는 점이 달랐습니다. 이번 수업을 통해서 리펙토링의 이유와 중요성을 알 수 있어서 좋았습니다.

# 안건우 리펙토링 내용

## **담당 파트**

AppManager

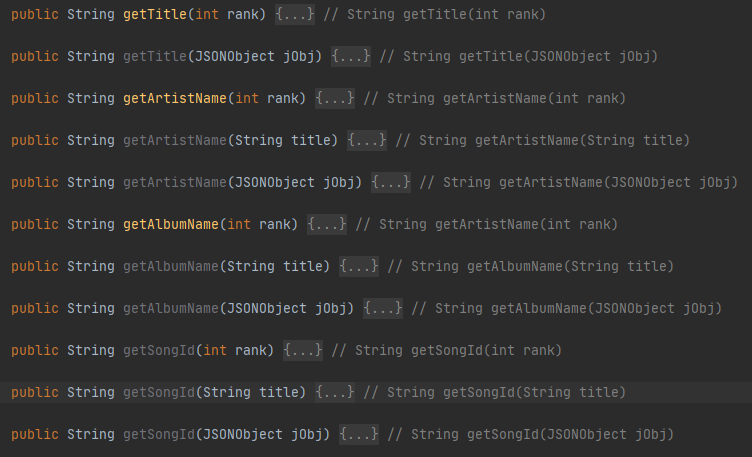
MusicChartParser

ChartData, DetailData

## 일정

|  |  |
| --- | --- |
| 9 주차 | AppManager 리펙토링 및 MVC 분리 |
| 10 주차 |
| 11 주차 | 카멜 표기법 |
| 12 주차 | MusicChartParser 리펙토링 |
| 13 주차 | ChartData, DetailData 리펙토링 |
| 14 주차 | CodeReview 피드백 반영 |

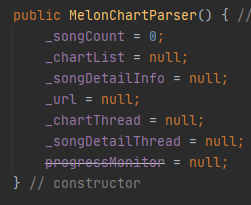
## AppManager 리펙토링



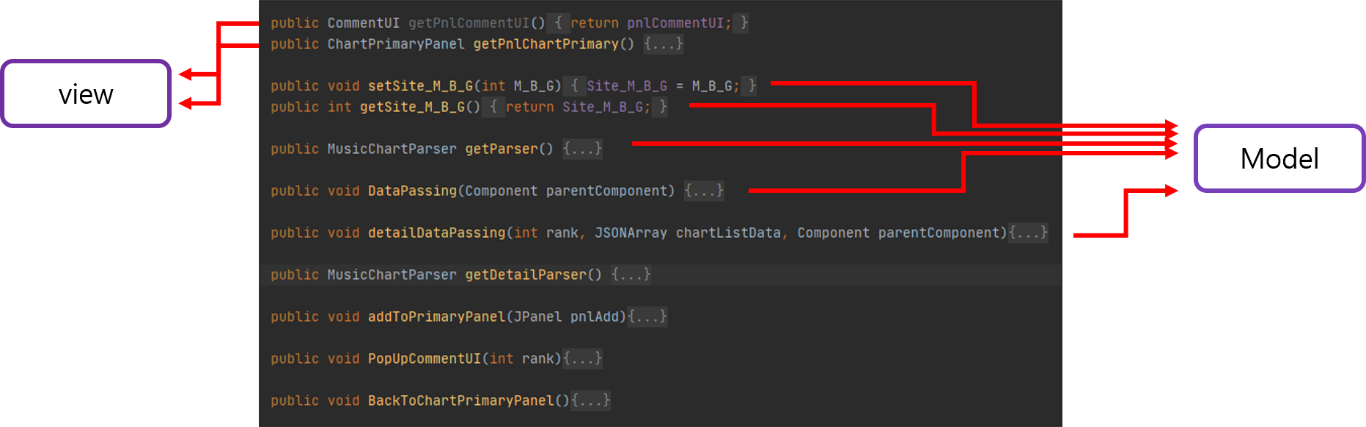
<메서드 이동 전 AppManager>

### AppManager의 메서드가 MVC의 기능을 전부 수행하여 메서드 이동 진행

AppManager의 메서드들을 각각의 MVC Class로 이동하였다.



<카멜 표기법 적용>



<AppManager 메서드 이동>

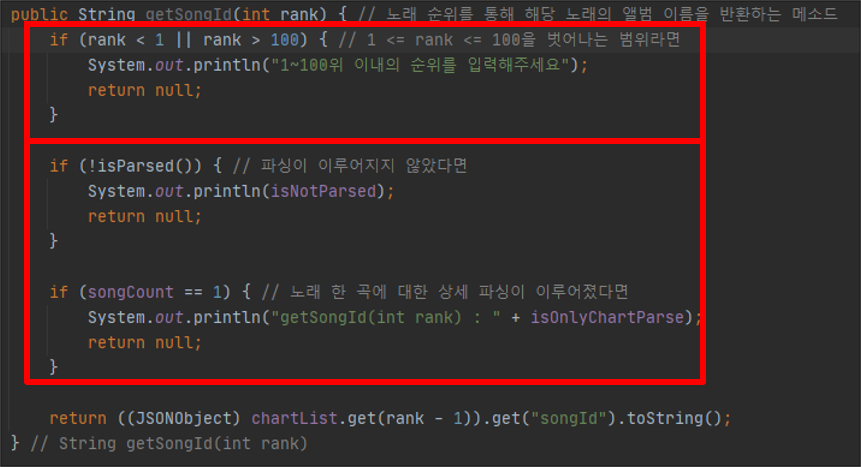
### 표기법

표기법을 카멜 표기법으로 통일하였다.

## MusicChartParser 리펙토링 (피드백 반영)

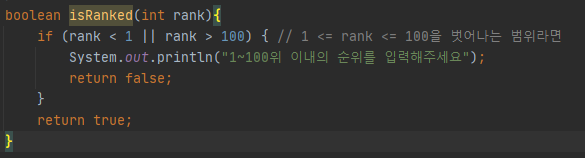
MusicChartParser에는 여러 메서드 안에 중복된 코드가 존재한다.

중복된 코드를 메서드로 추출하였다.

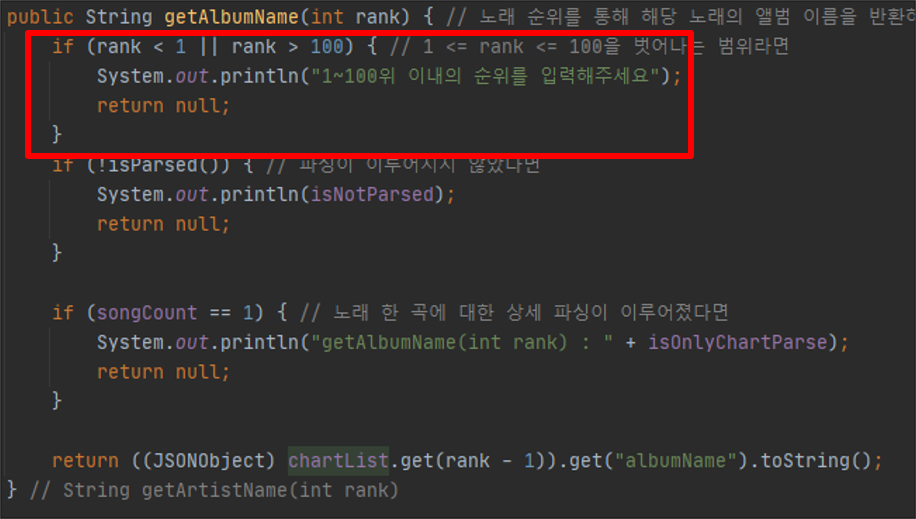


<매개변수가 rank인 메서드>

공통된 rank 값을 검사하는 코드를 rank를 매개변수로 받는 메서드로 추출하였다.

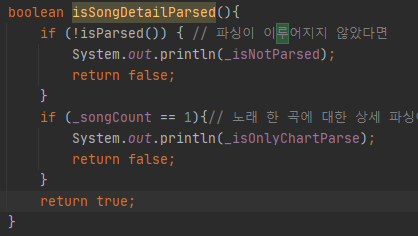


<Rank 값 검사 코드의 메서드 추출>

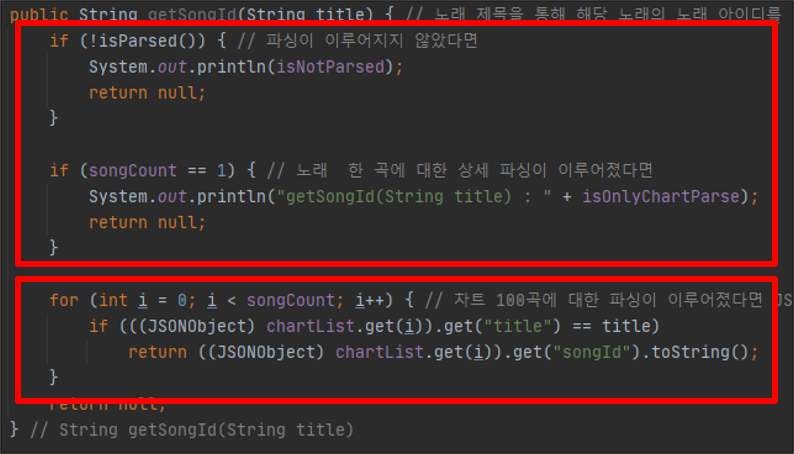


<매개변수가 rank인 메서드>

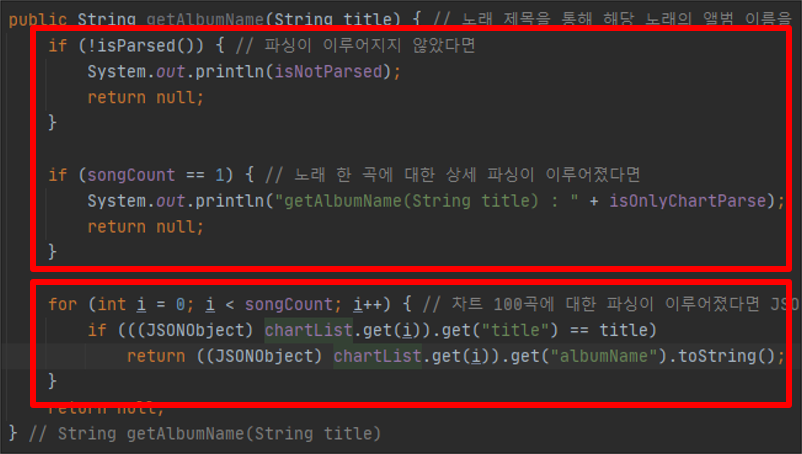
Parsing 여부를 검사하는 코드를 메서드로 추출하였다.



<Parsing 여부를 검사하는 코드의 메서드 추출>

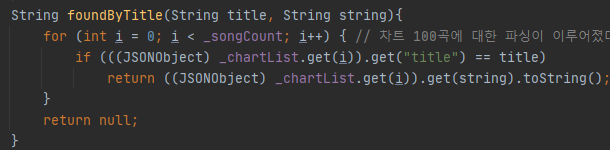


<매개변수가 title인 메서드>



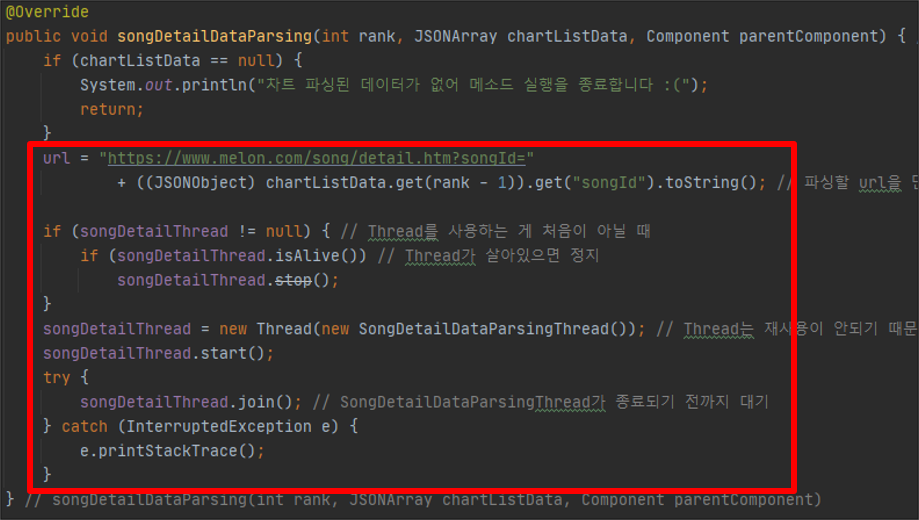
<매개변수가 title인 메서드>

title과 일치하는 JSONObject를 찾는 코드를 title을 매개변수로 받는 메서드로 추출하였다.



<title과 일치하는 JSONObject를 찾는 코드의 메서드 추출>

Parsing 코드를 메서드로 추출하였다.

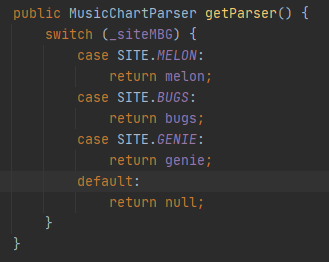


<rank와 JSONArray를 매개변수로 하는 메서드>



<JSONObject를 매개변수로 하는 메서드>

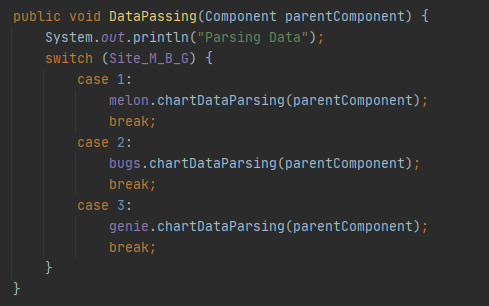
## ChartData, DetailData 리펙토링



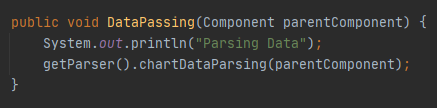
<분류를 위한 switch 문>

ChartData와 DetailData 클래스는 분류 부호 \_siteMBG를 사용한다.

분류를 위한 switch 문 하나만 남긴 후 나머지 switch 문을 수정하였다.



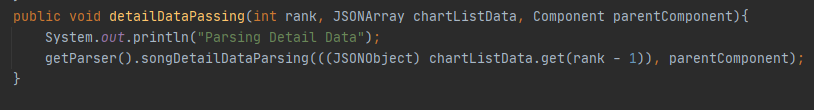
<switch 문 리펙토링 전 DataPassing 메서드>



<switch 문 리펙토링 후 DataPassing 메서드>



<switch 문 리펙토링 전 detailDataPassing 메서드>



<Switch 문 리펙토링 후 detailDataPassing 메서드>

## 개인소감

메서드 추출, 메서드 이동 등 여러 리펙토링 기법을 통해 효율적이고 보기 좋게 코딩할 수 있는 법을 알게 되었고 앞으로도 중간중간 리펙토링을 해야 함을 느꼈다.

# 박기춘 리펙토링 내용

## 담당 파트

전반적인 DB관련 model파트 및 관련된 컨트롤러의 일부를 담당하였습니다.

## 일정

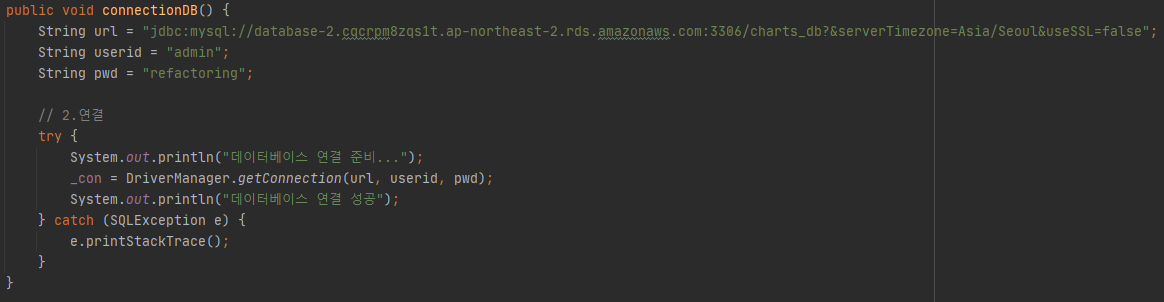
|  |  |
| --- | --- |
| 9 주차 | 로컬 DB에서 AWS DB로 연결 |
| 10 주차 | 코멘트 리펙토링 – 메서드 분리, 추출, 변수 |
| 11 주차 |
| 12 주차 | DAO, DTO 분리 및 데이터베이스 리펙토링 |
| 13 주차 | 최근 본 목록 기능 추가 및 리펙토링 |
| 14 주차 | 코드 리뷰 피드백 반영 및 미흡한 부분 추가 수정 |

## Data Base

### localhost에 연결된 DB를 AWS를 이용하여 외부에서 별도의 환경설정 없이 접근이 가능하도록 했습니다.

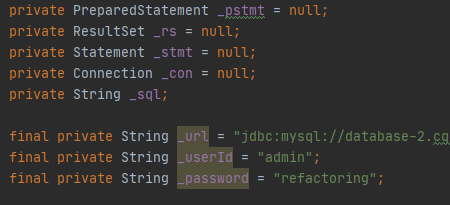


<변경 전 - localhost에 연결된 데이터베이스>



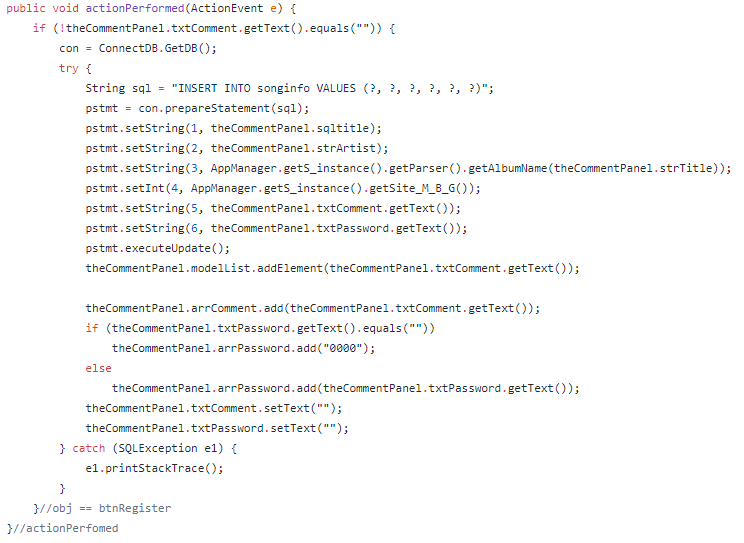
<변경 후 - AWS에 연결된 데이터베이스>

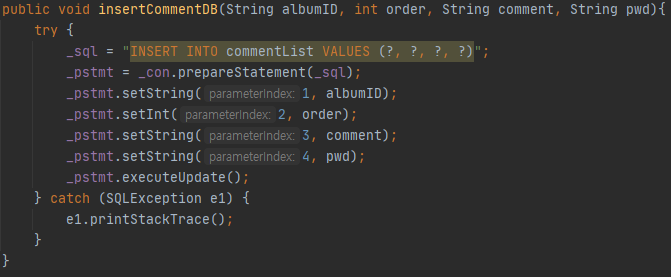
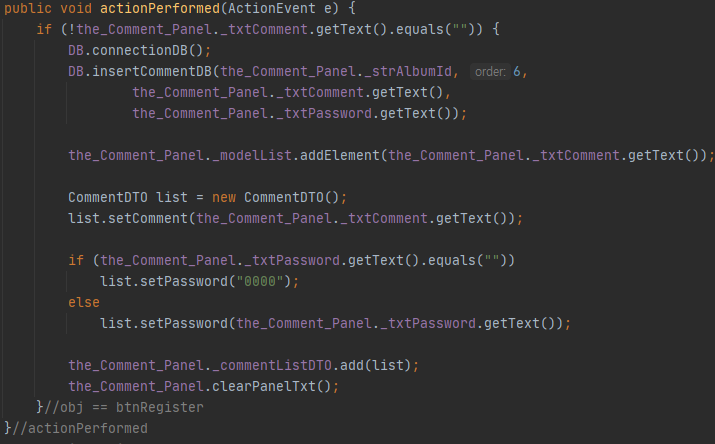
### 표기법을 카멜 표기법으로 통일하였다.



<카멜 표기법 적용>

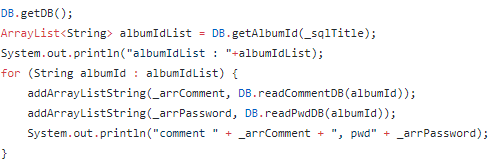
### 하나의 메서드 안에서 여러가지 기능을 처리하였다.



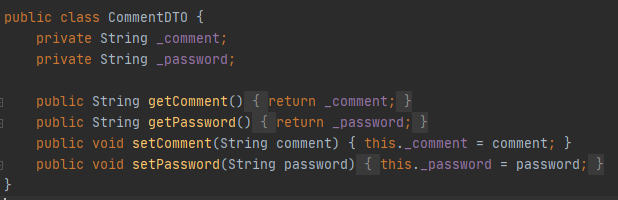
<변경 전 – 여러 기능이 같이 있는 하나의 메서드>

<변경 후 – 메서드 추출 및 클래스 분리>

### DAO, DTO분리를 통한 반복 사용되는 코드의 제거

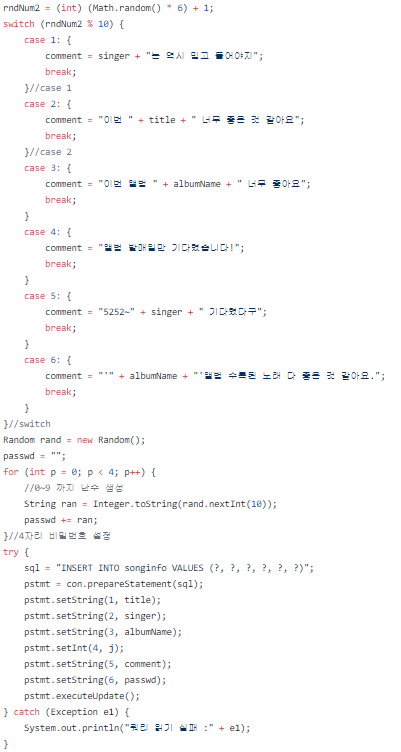
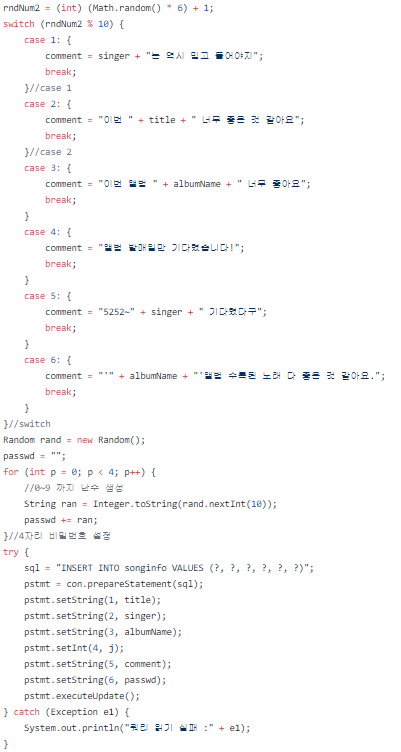


<변경 전 – ArrayList의 반복적인 사용>

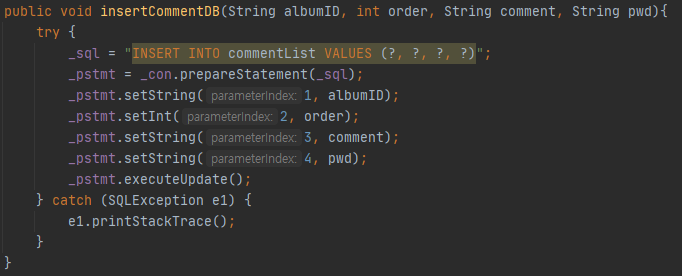
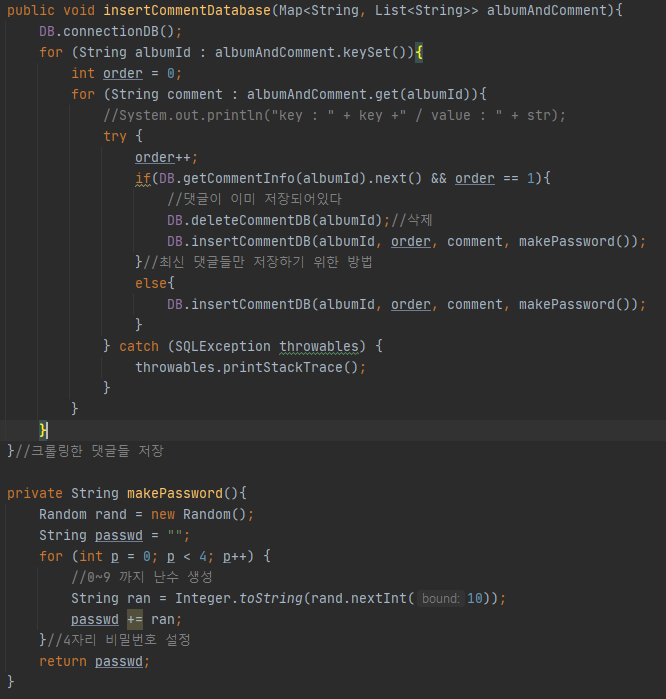
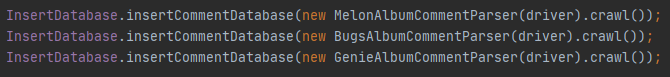


<변경 후 – 하나의 클래스 생성 후 DTO, DAO 분리>

### 랜덤으로 생성한 댓글의 저장에서 실제 크롤링한 댓글의 저장으로 변경하였다.



<변경 전 – 랜덤으로 만들어진 댓글을 DB에 저장>



<변경 후 – 크롤링한 실제 댓글을 DB에 저장하는 메서드를 만들고 분리>

## 개인소감

리펙토링 기법을 통해 길어졌던 코드를 간단하게 만들면서 예전에 이 프로젝트를 진행할 때 얼마나 코드가 비효율적이었는지 알게 되었다. 그리고 새로운 기능을 만들고 리펙토링하면서 설계의 중요성을 다시 한번 깨닫게 되는 계기가 되었다.

# 김성수 리펙토링 내용

## 담당 파트

MVC Controller 분리

최근 본 음악 목록 -> YouTube 음악 검색

## 13주차 까지의 일정

|  |  |
| --- | --- |
| 9 주차 | MVC 분리 마무리 |
| 10 주차 | 스위치문 정리 |
| 11 주차 | 메서드 정리 |
| 12 주차 | 유튜브 음악 검색 기능추가 |
| 13 주차 | 추가 기능 마무리 |
| 14 주차 | 코드리뷰 피드백 |

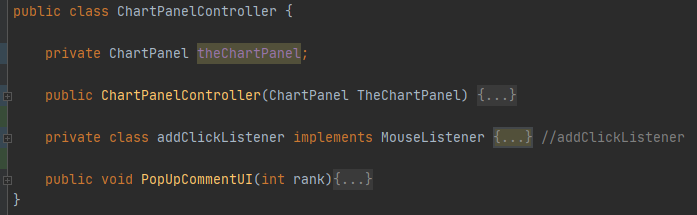
## MVC 분리 및 코드 리펙토링

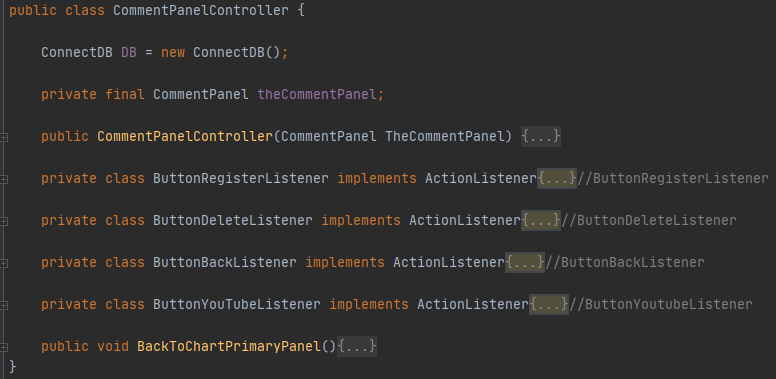
### MVC 모델 Controller 분리

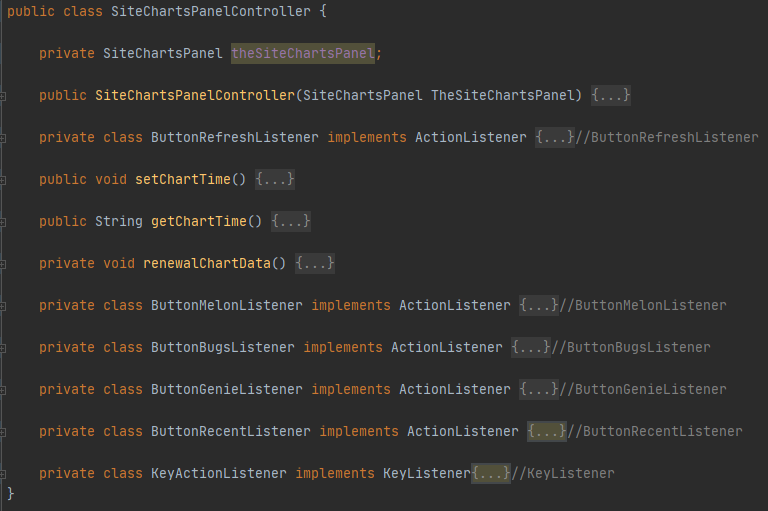


기존 코드들이 MVC 분리 없이 한 클래스 내부에 전부 존재하였음

분리 후 컨트롤러를 패키지로 묶고 리펙토링 중 생성된 Parser들 또한 컨트롤러로 이동





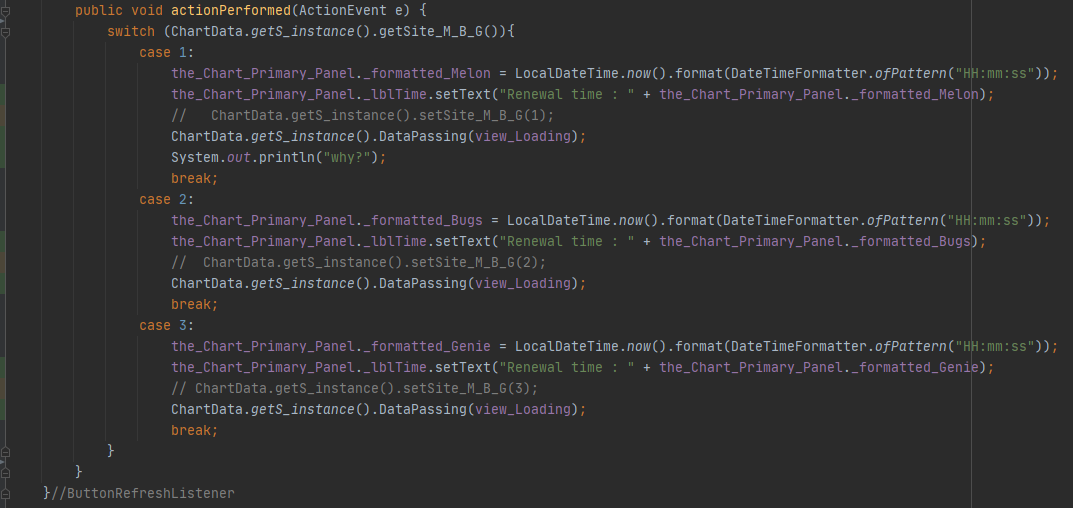


컨트롤러 클래스들은 상속된 패널의 컨트롤러 기능과 컨트롤러 기능에 포함된 메소드로 구성

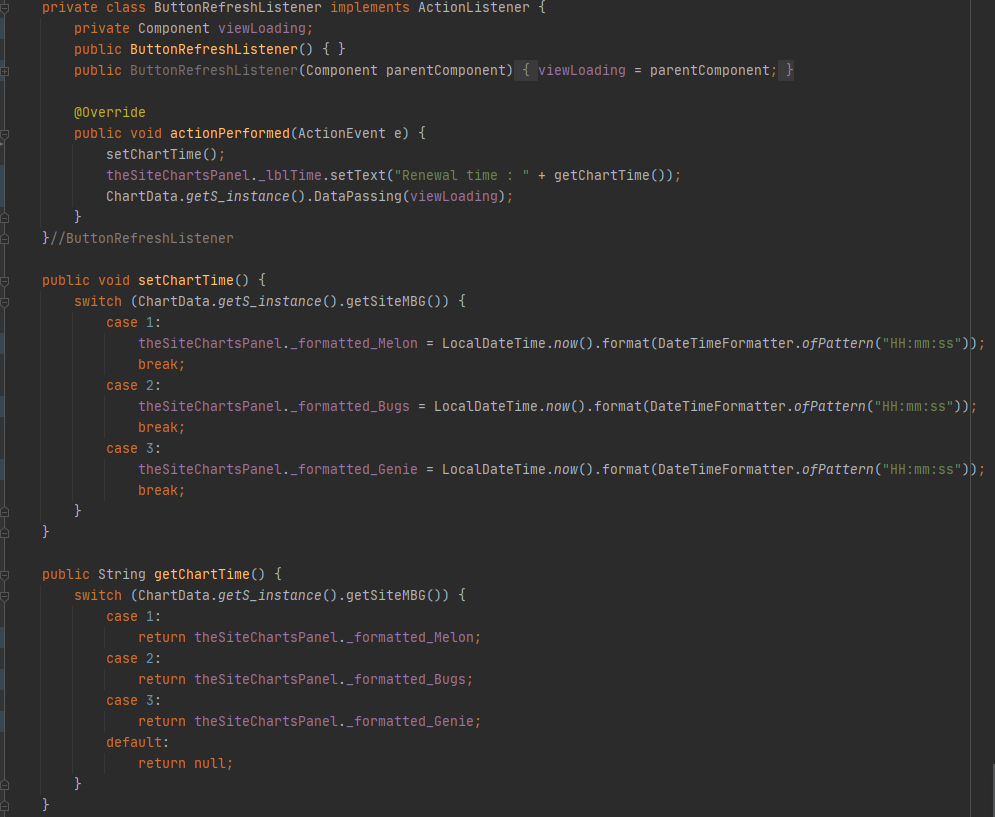
# 코드 리펙토링

### 스위치문 정리

변경 전



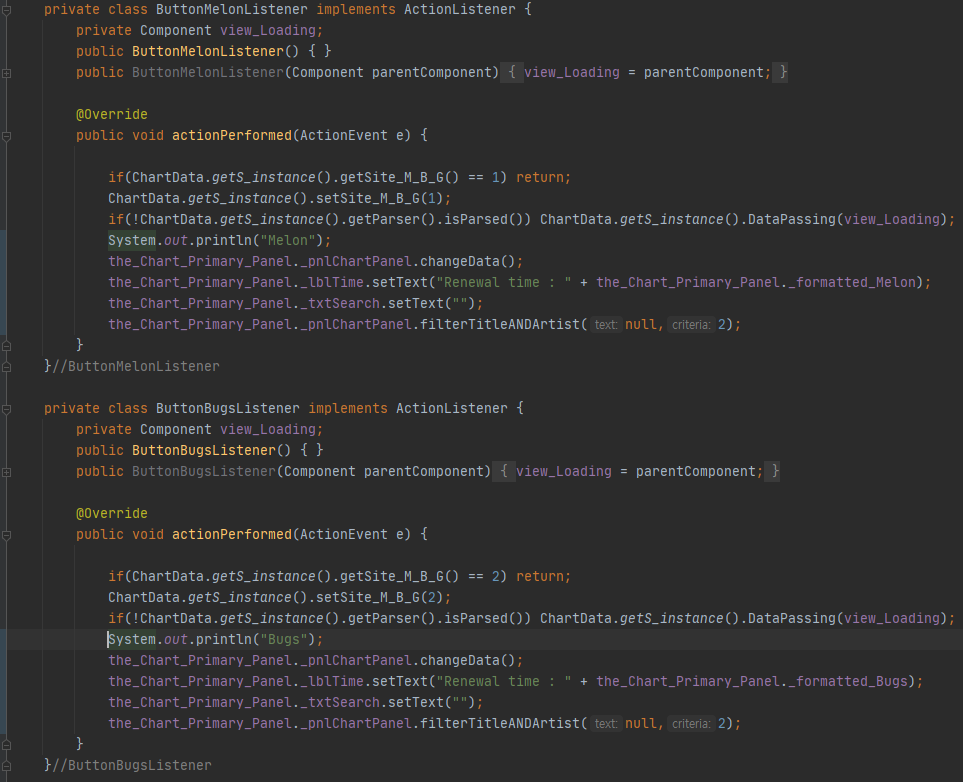
변경 후



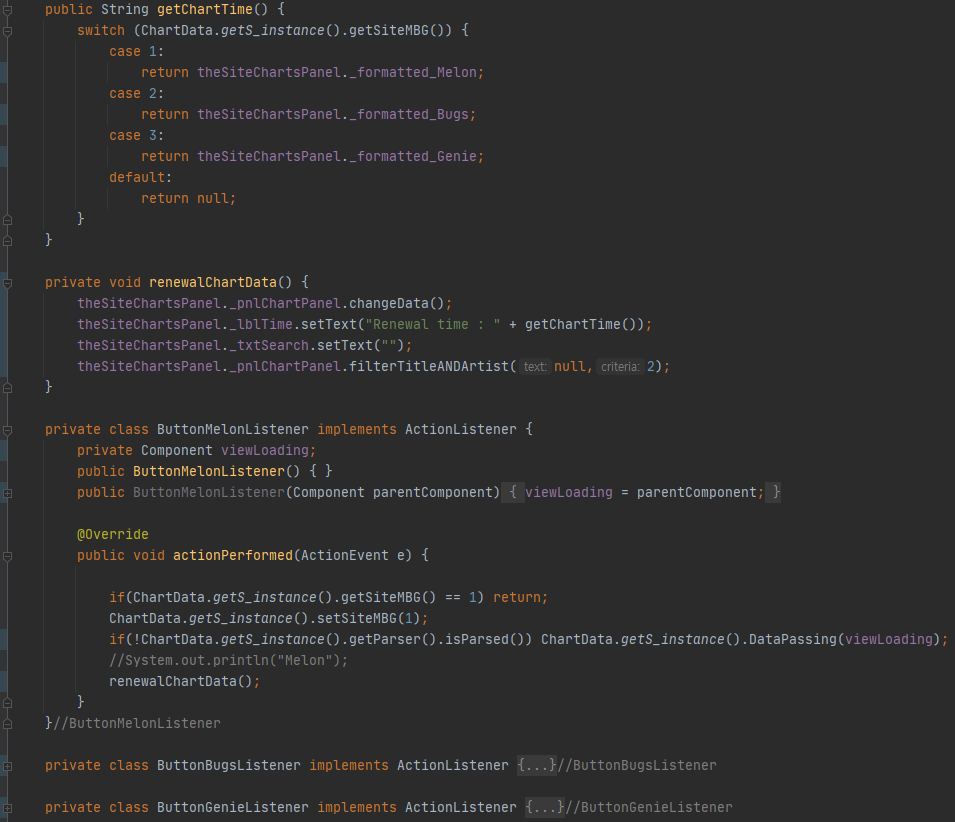
스위치문 내부에서 변수만 다른 채 공통적으로 사용되는 코드를 스위치 외부로 분리함

### 유사 메소드 정리

변경 전



변경 후



메소드에서 공통적으로 사용하는 코드를 분리했습니다.

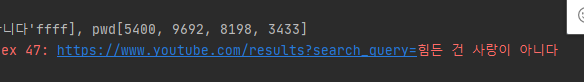
스위치문에서도 사용되는 변수를 따로 분리한 메소드를 사용했습니다.

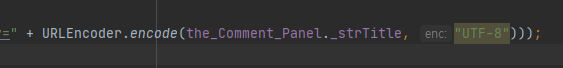
## 기능 추가 및 개선

애플리케이션 내부 곡 정보에서 “유튜브에서 검색하기” 기능을 통해 웹 팝업으로 검색 결과를 띄우는 기능을 추가



곡 이름에 공백 기호 및 한글 존재할 경우 인코딩 오류 발생





기존 스트링 전체를 주소창에 입력하는 방식에서 UTF-8 형식으로 인코딩하여 입력하는 방식으로 변경함

## 개인 소감

MVC 분리에 대해서 더 잘 알게 되었고 세 부분을 분리하는 법에 대해 알 수 있었다. 리펙토링 시 좀 더 효율적으로 리펙토링을 하는 방법에 대해 찾아볼 수 있게 되었고. 추가 기능을 통해 인코딩의 필요성을 알게 되었다.