음원관리 프로그램 최종

2015103278 응용수학과 이동주

- 프로젝트의 주제

예전에는 mp3플레이어를 통해 음악을 다운받아서 들었다면, 이제는 음원 스트리밍사이트를 주로 사용한다. 이들이 제공하는 기능들이 어떤 것이 있는지 알아보고, 분석해 직접 음원관리 프로그램을 제작한다.

- 프로그램의 목표

수업시간에 배운 다양한 자료구조들을 통해 음원관리 프로그램을 직접 제작해본다. 또한 현재 음원관리 프로그램에 추가되었으면 하는 기능이 있다면 이를 추가한 음원관리 프로그램을 구현한다. 최종적으로 다양한 자료구조의 사용법을 익히고, 사용하는 프로그램을 분석해 발전시킬 수 있는 능력을 키운다.

1차 프로젝트에서 끝내지 못했던 관리자와 사용자 모드의 구분과 로그인 및 회원가입기능, 개인마다 구분된 사용자리스트를 구현한다.

- 프로그램 구현방안

1. 음원프로그램을 관리하기 위한 효율적인 자료구조 사용

1) 최적의 자료구조 사용 : 효율적인 관리를 위한 적절한 자료구조의 선택

SortedList, Circular Queue, Stack, LinkedList, DoublyLinkedList… etc

2) Music Library : 관리할 모든 곡에 대해 여러 가지의 조건검색이 가능하도록 구현

SongList, SingerList, GenreList, AlbumList, EntList, MasterList

3) 새롭게 추가한 기능 기능

로그인기능을 통한 관리자모드

- 로그인을 통해 관리자모드로 들어가서 MasterList의 수정 가능

사용자 각 개인용 로그인 및 회원가입

- 사용자도 로그인 및 회원가입을 통해 사용자 각 개인의 리스트 관리

2. 클래스의 기능들

1) MusicType : 곡의 상세정보를 출력하기 위한 클래스, Primary key로 m\_ID 사용

2) SimpleMusicType : 곡의 간단한 정보를 가지고 있는 클래스, Primary key로 m\_ID 사용

3) SongType : Primary key로 m\_ID을 사용

4) SingerType : Primary key로 m\_singername을 사용. 각 가수의 곡들을 리스트로 저장

5) GenreType : Primary key로 m\_genretype을 사용. 각 장르의 곡들을 리스트로 저장

6) MelodizerType : Primary key로 m\_melodizer를 사용. 각 작곡가의 곡들을 리스트로 저장

7) AlbumType : Primary key로 m\_albumname을 사용. 각 앨범의 곡들을 리스트로 저장

8) UserType : 사용자의 정보를 저장

9) SortedList : 배열로 정의되어있으며 동적으로 배열의 크기를 할당가능. 새로운 데이터를 추가할 때 자동적으로 정렬을 시켜준다

10) SortedLinkedList : LinkedList로 정의되어있으며 효율적인 메모리사용을 위해 사용. 마찬가지로 데이터를 추가할 때 자동적으로 정렬이 된다

11) Application : 구현한 함수들을 실행시킴

- Admin, User용을 함수에 따로 표시해 코드의 이해도를 높였다.

각 클래스의 포함관계

SongList->GenreList

SongList->AlbumList->SingerList->EntList