프로젝트 보고서 - RMeal

2018320220 조재건

1. 프로젝트 개요

A. 문제상황

일주일에 4일 이상, 점심, 저녁을 학교 근처 식당에서 식사를 해결한다. 가본 식당은 많지만, 항상 무엇을 먹을지 고민에 빠지게 된다. 때로는 갈 만한 식당이 있어도 순간 생각나지 않아서 후회하는 경험이 많았다. 이러한 상황을 부르는 신조어로, '선택장애' 라는 말이 나타날 정도로 이는 많은 사람들이 겪는 사소한 고민이다. 이미 선택장애를 해소해주는, 식당을 골라주는 프로그램이 있지만, 지금은 가기 곤란하거나 가기 싫은 식당이 나오면 여러 번 더 돌려야 하는 단점이 있다.

B. 문제 상황을 해결할 수 있는 방법

공통적으로, 식당에 대한 데이터베이스를 직접 입력하고, 내가 가고 싶은 식당들을 데이터에 담는 것을 전제로 하였다. 또한, 그 데이터베이스들 중 무작위로 식사할 식당을 선택해주는 프로그램을 만들 것이다. 위에서 제시한 가기 곤란하거나 가기 싫은 식당이 나오는 것을 방지하기 위해 추가적인 기능을 도입하고자 한다. 이를 위한 아이디어가 두 가지가 있다.

i. 조건 별 검색 시스템

식당을 데이터베이스에 추가할 때, 식당에 대한 조건을 같이 입력 받는 것이다. 예를 들면, 어떤 일식집이 있다고 하자. 이 식당에 대해 가격대, 음식의 종류, 대략적인 위치(참살이길, 정대후문 등등), 식사 가능 인원(혼밥가능, 2인석, 4인석 등등), 영업시간, 예상 식사 시간과 같은 조건을 같이 입력하는 것이다. 이렇게 식당에 대한 데이터베이스를 충분히 저장한다. 이후 식당을 추천 받기 전에, 현재 나의 상황에 해당하는 조건을 입력한다. 3명의 친구와 식사를 하고, 저렴한 가격에, 양식, 중식, 술집은 빼고, 참살이길에 있는 식당에 들어가길 원한다면 조건을 체크하고 검색을 누르면 해당하는 식당 중에 무작위로 하나를 추천하는 방식이다.

ii. 가중치 시스템

위의 시스템을 기반으로 하여 추가로 구현이 가능하면 도전할 시스템이다. 각 조건들에 대해 가중치를 부여하여 무작위 추출에 관여하는 것이다. 예를 들어, 일식, 한식, 양식 중 어떤 것을 먹던 상관이 없지만, 그래도 양식보다는 일식이 먹고 싶다면, 검색 전에 일식의 가중치를 높게 부여하는 것이다. 이를 통해 사용

자가 더 먹고 싶은 음식을 먹을 수 있다.

C. 내가 선택한 방법

우선 첫번째 아이디어의 구현을 목표로 한다. 조건이 다양하기 때문에, 가게 클래스가 어떻게 조건들을 담을지, 어떻게 효율적으로 검색할지에 대한 구상이 필요하다. 기본적으로 가게 인스턴스에 대한 정보는 파일 입출력을 공부하여 파일에 저장하도록 하고, 프로그램이 종료된 후에도 데이터베이스가 유지되도록 한다. 프로그램은 스윙 컴포넌트를 활용하여, 컴퓨터 언어를 모르는 사람도 이용할 수 있도록 하고자 한다. 저장 및 검색 알고리즘을 우선적으로 구현하고, 여유가 남으면 가중치를 적용한 검색 알고리즘으로 확장할 예정이다.

D. 기대효과

"뭐 먹지"라는 불필요한 고민을 최대한 줄일 수 있을 것으로 보인다. 현재 나의 상황에 맞춘 검색이 가능하다는 점 덕분에 단순히 무작위 추출 프로그램이 아닌, 내가 실제로 사용할 수 있는 프로그램이 될 것이다.

2. 프로젝트 계획

A. 작업해야 할 내용

다양한 공부가 선행되어야 할 것으로 보인다. 우선적으로 예측되는 작업 내용은 아래와 같다.

i. 가게 특성 관련 조건 선택

가게에 대해 들어갈 조건에 대해 구상해야 한다. 조건을 사용자가 추가하도록 하는 것은 프로그램을 복잡하게 할 것이라고 생각하였다. 또한, 이후 검색을 할 때 조건이 너무 많아서 해당하는 식당이 나오지 않을 가능성이 있으므로, 꼭 필요하고 의미 있는 조건을 구상할 것이다.

ii. 가게 데이터베이스 - 파일 입출력

가게 데이터베이스를 저장하는 방식을 구상해야 한다. 프로그램이 종료되어도 데이터를 유지시켜야 하므로, 파일 입출력을 활용할 것이다. 따라서, 파일 입출력에 대한 학습이 선행되고 이를 계획하여야 한다.

iii. 검색 알고리즘 구상

구현에서 가장 난관으로 예상되는 부분이다. 먹고 싶은 음식이 굳이 일식 하나일 필요가 없고, 일식, 중식, 한식 중 무엇을 먹어도 상관없을 수 있다. 가격대 역시, 저렴해도, 비싸도 상관없을 수 있다. 따라서 때로는 조건이 수십가지가 포

함될 수도 있다. 이러한 상황에서 어떻게 효과적으로 가게들을 가려낼 것인지 고민할 필요가 있다.

iv. GUI 구상

GUI는 단순히 콘솔과 다르게 디자인을 필요로 한다. 어떻게 디자인해야 효율적으로 프로그램이 될 수 있을지 생각해야한다. 또한, 스윙 컴포넌트를 활용하기로 하였으므로, 이에 대한 학습이 선행되어야 계획할 수 있다.

v. 구현

vi. 디버깅

B. 주차별 작업 계획

i. 1주차 - 파일 입출력 및 스윙 컴포넌트 학습

ii. 2주차 – 프로그램 구상(조건, 알고리즘, 클래스 등)

iii. 3주차~ – 코딩, 이후 디버깅

iv. 여유가 생기면 기능을 확장한다.

C. 주차별 작업 결과

i. 1주차 - 프로그램 구상 및 Git 사용법 학습

ii. 2주차 - 스윙 컴포넌트 학습 및 프로그램 제작(version 0.1~version 0.8)

1. Version0.1
   -패키지 생성: Data관리 패키지 'Data', GUI관리 패키지 'GUI'
   -Data- OptionList, Restaurant 클래스 생성
   -GUI- MainGUI 클래스로 메인화면 구현

2. Version0.2
   -OptionList, Restaurant 구현
   _AddGUI로 식당 추가 화면 구현 시작
   -스윙 컴포넌트를 생성할 때 반복적인 작업들을 줄이기 위해 CreateComponent라는 컴포넌트 생성 Manager 클래스를 만들어 static 메소드로 컴포넌트 생성 과정을 구현함

3. Version0.3
   -AddGUI 구현 완성함
   -SearchGUI 구현 시작함
   -식당의 정보를 열람할 수 있는 RestaurantInfo 클래스 생성함.

4. Version0.4
   -SearchGUI 추가 구현함

5. Version0.5
   -X버튼 누를 시 종료할 지 물어보는 WindowListener를 구현함.
   -전체적인 GUI를 효율적으로 개선함.

6. Version0.6
   -식당의 옵션 정보를 수정할 수 있는 ModifyRestaurantInfoGUI클래스를 생성함
   -SearchGUI의 검색결과 리스트를 JList로 구현함.
   -JList에서 마우스 더블클릭으로 각 식당 정보를 열람할 수 있도록 MouseListener를 이용하여 구현함.

7. Version0.7
   -RestaurantInfoGUI 클래스 구현 시작함.
   -SearchGUI의 GUI를 일부 개선함.

8. Version0.8
   -GUI패키지의 이름을 Main으로 변경함
   -식당 추가, 검색, 삭제 등의 데이터 관리 기능을 각 GUI에서 분리하였고, Data 패키지의 RestaurantManager클래스를 생성하여 데이터 관리를 담당하는 기능들을 static 메소드로 구현하여 모아둠.
   -RestaurantInfoGUI 클래스 추가 구현함.

iii. 3주차 – 프로그램 제작(Version0.9~Version0.12)

1. Version0.9
   -Option 객체 삭제-> String으로 대체, 관련한 모든 코드 수정
   -SearchGUI와 RestaurantInfoGUI 클래스를 추가 구현함.

2. Version0.10
   -SearchGUI의 검색하기 기능 구현함.
   -검색 후에도 CheckBox의 상태를 유지시키기 위해 boolean배열을 이용하는 알고리즘을 구현함.

3. Version0.11
   -AddGUI의 코드를 활용하여 ModifyRestaurantInfoGUI클래스를 구현함. 식당 정보를 수정하면서 겹치는 데이터에 대한 예외처리를 구현함.

4. Version0.12

-현재 검색 알고리즘이 잘못된 것을 확인하고, 검색, 삭제, 수정 등의 RestaurantManager 내부 메소드들을 전부 수정함.
-ModifyRestaurantInfoGUI 클래스 내부 구현을 완성함.
-여기까지 식당 추가, 검색, 삭제, 수정 등의 기능이 전부 구현됨.

iv. 4추차 – 프로그램 기능 완성 및 추가 수정(Version 1.0~Version1.3)

1. Version1.0
-RestaurantInfoGUI 클래스의 코드를 일부 효율적으로 개선함.
-RecommendGUI 클래스의 식당 추천 기능을 구현함
-일차적으로 프로그램의 모든 기능을 완성함.

2. Version1.1
-ArrayList<Restaurant> 형태로 이용하던 식당 리스트를 RestaurantList라는 클래스를 만들어서 자료 구조를 개선함

3. Version1.2
-기존 GUI의 이동방식은 매번 JFrame을 끄고 다음 JFrame을 키는 방식이었으나, JFrame 하나만 유지하고 화면전환을 구현하기 위해 RMealMainFrame을 생성하고, 나머지 GUI클래스들을 JFrame 상속에서 JPanel 상속으로 바꾸어 GUI를 개선함.

4. Version1.3
-프로그램 종료 후에 다시 실행했을 때도, 기존 데이터를 유지시키기 위해 객체 직렬화를 이용하여 텍스트 파일에 저장하도록 구현하는 DataManager 클래스를 생성함. DataManager 클래스는 파일 입출력 및 객체 직렬화만 담당하는 static 메소드들을 담고 있음.

v. 5주차~ – 프로그램 개선 및 수정(Version 1.4~)

1. Version1.4
-프로그램을 exe 형식으로 변환하여 실행하였을 때, 기존에 나눔스퀘어 폰트는 폰트가 설치되지 않은 환경에서 기본글꼴로 나오는 상황을 확인하였음. 따라서 프로그램 내 모든 폰트를 함초롬돋움으로 변환함.

2. Version1.5
-각 JPanel 객체에서 MainFrame에 접근할 수 있도록 Frame의 Contentpane을 Container으로 만들어서 GUI 객체 생성시에 매개변수로 전달하였는데, getParent()메소드를 이용해서 접근할 수 있으므로 관련 코드를 일부 수정함.

3. Version1.6
   -1.5에서 수정중이었던 getParent()관련 코드의 나머지 부분을 수정하여 코드를 개선함.

4. Version1.7
   -식당의 정보를 수정하고 그 식당을 바로 삭제하면 레이아웃이 이상해지는 오류를 발견 후 해결함.

5. Version1.8
   -InteliJ에서 제시한 Waring들 약 200여개를 수정하여 더 효율적으로 개선함 (For문=>foreach문, Anonymous class->lambda expression)

3. 프로젝트 결과

A. 계획서 대비 변화 부분

i. 가중치 시스템 구현 취소

여유가 있으면 가중치 검색을 도입하려 했으나, 이를 위해서는 프로그램의 전반적인 데이터 구조 및 알고리즘을 모두 수정해야했기에, 남은 기간 내에 해결하지 못할 것으로 보아 취소하고, 더 효율적이고 가독성 있는 코드를 작성하기 위해 기존 코드를 꾸준히 수정하였다.

ii. 프로그램 제작과 스윙 컴포넌트 학습을 병행

기존 계획서에는 스윙 컴포넌트 학습을 먼저 하고 프로그램 제작을 하려했으나, 하나씩 기능을 사용하다가 프로그램 제작을 바로 하기로 결정함.

iii. 검색 조건 축소

효율적인 GUI 구성과 검색 알고리즘의 난이도가 지나치게 오르는 것을 막기 위해 영업시간, 식사 예상 시간의 조건을 제외하고 위치, 식사 타입, 가격대, 식사 가능 인원의 4가지 조건만 사용하였다.

B. 깃허브 링크 주소

https://github.com/pandracoon/Java-Project-RMeal

C. 동작 데모 영상 링크 주소

https://youtu.be/O4eDGuMeKN8

4. 회고

책에 나오는 예제가 아닌, 내가 직접 만들고 싶은 프로그램을 만드는 것은 이번이 처음이

었다. 되돌아 보았을 때, 그렇게 대단한 프로그램이라고 생각되지는 않지만, 처음이기에 의미가 크고, 많은 난관에 봉착했던 것 같다. 초기 계획단계에서의 어려운 점은 자바를 전부 다 배우지 않았기 때문에 난이도가 높다는 점이다. 내가 자바를 다 알지 못하기 때문에, 이것이 구현하기에 어려운지 쉬운지, 혹은 불가능한지 등을 파악하는 것이 쉽지 않았다. 그래서 구현할 수 있을지의 관점에서 이번 주제는 "검색 및 파일 관리"알고리즘만 잘 만들면 어렵지 않은 주제였기 때문에, 다른 어려운 구현에 신경 쓰지 않고 오로지 JAVA 자체에 집중할 수 있었다. 프로그램을 만들면서 동시에 공부를 하다 보니, 바로 며칠전에 짠 코드가 매우 비효율적이라는 것을 깨닫고 코드들을 전부 뒤엎는 경우가 허다했다. 또, 최대한 가독성이 좋은, 객체지향이라는 관점을 살려보고 싶었다. 파일을 어떻게 분리할 것인지, 인터페이스와 추상 클래스는 언제 사용하는 것인지, 클래스는 어떻게 분리할 것인지 등을 고민했다. 그 과정에서 JAVA와 객체지향이 무엇인지 더 깊이 있는 이해가 가능하게 되었다. 특히 인터페이스와 같은 상속 개념은 처음 배우고 예제를 풀 때까지도 잘 이해가 안되는 개념이었는데, 이번 프로젝트를 통해 인터페이스를 어떻게 적용할지 고민하는 과정 속에서(결국 필요 없다고 판단하여 사용하지는 않았으나) 정말 인터페이스가 무엇인지, 왜 중요한지 알게 되었다. 아쉬운 점도 많았다. 계속해서 배우면 배울수록 더 효율적인 코드를 발견해서 고쳤지만, 몇몇 건드리면 코드 전체를 새로 써야 하는 경우도 있었다. 시간 상의 문제로 고치지 못한 것이 아쉽고, 추가로 구현하기로 했던 기능들 역시 같은 이유로 구현하지 못해서 아쉬웠다. 다음에 개발을 이어간다면, 이 프로그램에 한해서는 이를 커뮤니티로 확장해서 다양한 사람들이 식당에 대한 정보를 추가해서 공유할 수 있도록 한다면 더 의미 있는 프로그램이 될 것 같다. 또한, 지도API를 통해 식당의 위치 또한 접근이 가능하면 좋고, 여기에 이 프로그램을 Application으로 개발하면 더 좋을 것이다. 지금은 컴퓨터에서만 실행이 가능하지만, 대학생이 아닌 사람들은 굳이 식사 전에 컴퓨터를 할 일이 없을 것이므로, 사실 이 프로그램은 다소 효율적이지 못한 프로그램이 된다. 이번 프로젝트를 통해 이제 무엇이든 도전하면 만들 수 있다는 자신감을 얻었고, 이후 다른 오픈소스나 시스템을 이용해서 더 의미 있는 프로그램을 만들기로 다짐하는 계기가 되었다.

5. 참고한 자료

https://github.com/pandracoon/Java-Project-RMeal.git

주로 수업 때 사용하는 책과 구글링을 통해 어려움을 해결했다.