나만의 단어 암기 프로그램 만들기

2018320242 이준수

Ⅰ. 개요

ⅰ. 내가 겪고 있는 문제 상황

나에게는 심도 있게 배우고 싶은 언어가 두 개 있다. 일본어와 영어이다. 이 언어들을 배우면서 가장 크게 다가오는 문제점은 어휘의 방대함이었다. 독해력이나 문법이나 회화나 결국 폭 넓은 어휘 지식 위에서 말할 수 있는 내용들이기 때문이다. 문제는 어휘를 늘리는 가장 좋은 방법은 단순히 반복해서 보는 것 뿐이라는 사실이었다. 내 일과의 대부분이 컴퓨터를 보는 시간으로 구성되어 있고 단어장을 보는 시간은 없거나 극히 일부라는 것을 보면 단어를 외우는 것은 거의 불가능에 가깝다는 생각이 문득 들었다. 그래서 컴퓨터를 통해 단어를 편리하게 암기할 수 있는 프로그램을 필요성을 느끼게 되었다.

여기에서 나는 또 문제에 당착했다. Anki와 같은 선구자들이 있지만 이들은 내가 생각하는 것과는 다르게 작동한다는 것이다. 알고 있는 단어를 맞춰보는 데에 있어서 다양한 선택지를 제공하는 것이 아닌 단순히 '보고 넘어간다'의 수준에 그치는 것이 가장 큰 문제이며, 독자적인 단어장 파일 형식을 사용해 생각보다 '나에게 맞는 단어장'을 만드는 것이 어렵다는 문제 또한 발견했다(온라인과 모바일 지원이 핵심적인 프로그램이지만 논외로 하자). 그리고, 일본어 한자 등을 띄워주는 방식이 내가 생각하는 방식과는 아주 다른 방향성을 가지고 있기 때문에 이 프로그램의 사용이 조금 꺼려졌다.

ⅱ. 문제 상황을 해결할 수 있는 방법

1. Anki의 대체제를 찾는다.

대부분의 프로그램들이 Anki와 비슷한 문제점을 지니고 있다는 점(내가 사용하고 싶은 방식과 꽤 동떨어져 있다)과 암기 팩 등을 만드는 커뮤니티들이 대개 영어권이며 Anki를 위주로 돌아간다는 현실을 생각해볼 때 그다지 좋은 선택지가 되지 못한다는 결론에 도달했다.

2. Anki의 클론을 만든다.

필요한 부분 만을 고쳐서 만들 수 있겠지만 프로그램의 구조를 이해하는 데에 걸리는 시간과 효율이 좋지 못하고 프로그램의 근본적인 부분을 바꾸기 힘들 것이라는 생각을 했다.

3. 별개의 프로그램을 구현한다.

이 경우 내 입맛에 맞는 프로그램을 만들 수 있지만 이 프로그램에 넣기 위한 DB를 내가 만들어

야 한다는 가장 큰 치명적인 단점이 있다.

iii. 여러 가지 해결 방법 중 내가 선택한 방법

- Anki의 클론을 만든다

DB는 공부를 하면서 채워 넣는다는 생각으로 만들다 보면 내가 사용하기 위한 목적으로는 훌륭하게 작동할 것이라는 생각을 했고, 다른 사람이 사용할 시 추가적으로 DB를 구축하기만 하면 된다.

iv. 예상되는 모습 및 기대하는 효과

가장 처음 프로그램을 키면 다양한 단어장 파일 중 원하는 것을 선택할 수 있는 리스트를 띄우도록 할 것이다. 이 리스트에는 단어를 본 수, 사용자가 외웠다고 생각되는 단어의 수, 마지막으로 단어장을 본 날짜 들을 추가적으로 볼 수 있다. 또한 이 리스트에서 검색/정렬을 통해 단어장을 더 빠르게 찾을 수 있게 할 것이며, 일반적으로 많이 쓰이는 txt/csv 형식의 단어장을 추가적으로 불러올 수 있는 방법을 만들 것이다.

리스트를 선택한 뒤로는 단어를 처음 보기 위한 버튼과 지금까지 외운 단어를 테스트하는 버튼 두 가지를 띄운다. 단어를 처음 보기 위한 버튼을 누르면 지금까지 열람한 적 없는 단어들을 뜻과 예문 등과 함께 띄워주며 사용자가 이 단어를 잘 알고 있는지 여부를 3단계 정도로 나누어 묻는다. 이것은 사용자가 새로운 단어를 손쉽게 볼 수 있게 함과 동시에, 추후 단어 테스트를 할 때 단어가 나오는 빈도를 결정하기 위함이다.

단어를 테스트 하기 위한 버튼을 누르면 단어를 알고 있는 정도에 따라 보정을 받고, 단어의 뜻과 단어를 매칭하거나 그 뜻을 묻는 문제 등을 랜덤하게 출력한다. 단어를 맞춘 여부에 따라서 이 단어를 아는 정도의 척도가 바뀌게 되고, 이 과정이 순환되며 최종적으로는 사용자가 단어에 익숙해질 수 있도록 한다.

이 프로그램을 사용함으로써 사용자가 원하는 단어를 더 많이, 더 빠르게 숙지함으로써 학습 효과를 증대화하고, 언어에 더 익숙해질 수 있는 기회를 마련할 수 있을 것이다.

Ⅱ. 프로젝트 계획

ⅰ. 주차별 작업 계획(계획)

1주차(11/2~11/8)	- txt/csv 파일 형식 구상 DB의 역할을 할 txt와 csv 파일의 형식을 구상하여, 이 파일을 어떻게 불러올지를 결정한다. 이 파일들에는 index, 단어, 그 뜻, 이외로 필요한 내용 등이 들어간다.

	- setting/진행 상황을 저장할 파일 형식 구성 각각의 DB 역할을 할 txt와 csv 파일에 대해, 해당 파일들의 현재 상태와 진행 상황을 기록할 파일 형식을 계획한다. 이 파일에는 DB 파일의 경로, 현재 확인한 단어, 어떤 단어를 알고 있는 정도 등의 정보가 담길 것이다. - 파일 파서와 랜덤 단어 추출 구현 txt/csv 파일과 현재 상황을 저장하는 파일들의 파서를 구현해 정보를 읽어올 수 있도록 하고, 읽어온 정보를 바탕으로 랜덤으로 단어를 추출하는 기능을 만든다. - 단어 분류 기능 단어를 외운 정도에 따라 그 index를 설정 파일에 저장한다.
2주차(11/9~11/15)	- swing을 이용한 프로그램 시작 화면 만들기 단어장 DB의 리스트를 설정 파일을 읽어옴으로써 list로써 띄우고, search, sort 등의 기능을 통해 찾기에 용이하도록 한다. - swing GUI와 단어 랜덤으로 띄우기 기능의 결합 및 단어 난이도 선택 1주차에서 만든 단어 선택 기능을 이용하여 GUI에 단어의 뜻을 부착, 사용자가 단어의 난이도를 선택하여 설정에 저장할 수 있도록 한다. - 단어 퀴즈/문제 구상에 따른 GUI 틀 제작 효율적인 단어 퀴즈의 형식을 구상하여 그에 따른 GUI를 제작한다. 문제가 주어지는 부분과 답을 선택하는 버튼이 주가 될 것으로 생각된다.
3주차(11/16~11/22)	- 단순 랜덤 선택 방식에서 단어를 아는 정도에 따른 선택 구현 다양한 수학적인 통계 방식 중, 적절한 것을 골라 알고리즘을 구상하고 그 경향에 따라 단어를 랜덤으로 선택하도록 메소드를 수정한다. - 단어 퀴즈 GUI 내부 기능의 일차적인 구현 #1 2주차에서 구현한 GUI 틀을 기반으로 단어 퀴즈를 구현하고, 그 결과에 따라 단어의 난이도를 재배치하는 기능을 넣는다.

	- GUI들의 유기적인 연결 초기 선택, 단어 퀴즈, 단어 보기 기능을 유기적으로 하나의 GUI로 묶어 프로그램의 프로토타입을 만든다.
4주차(11/23~11/29)	- 단어 퀴즈 구현 #2 다양한 타입의 단어 퀴즈를 생성하여 랜덤으로 선택하도록 하여 퀴즈 유형의 다양성을 높이고, 단어를 아는 정도의 척도를 3단계의 이산적으로 할 것인지 연속적으로 할 것인 것인지를 프로그램의 방향성에 따라 결정한다. - 단어 보기 화면의 기능 추가 Google 등에서 단어를 검색한 결과를 브라우저를 띄워 확인할 수 있게 하거나, 단어 북마크 기능 등을 추가하여 사용자가 더 자주 보고 싶거나 추후에 볼 단어를 따로 관리할 수 있도록 한다. - 단어 퀴즈 통계 추가 #1 단어 퀴즈를 맞추거나 틀린 횟수, 매일 공부한 횟수를 집계하여 사용자가 공부 추이를 볼 수 있도록 한다.
5주차(11/30~12/6)	- 단어 퀴즈 통계 추가 #2 공부 추이 정보를 기반으로, 다양한 그래프 및 기준 척도를 제공하여 사용자가 원하는 방식으로 공부 추이를 볼 수 있도록 한다. - GUI 개선 및 수정 GUI를 사용하며 불편했던 부분을 수정하고, 더 직관적이고 사용성이 좋도록 한다. - 코드 리팩토링 #1 코드/클래스/메소드 간 상호작용이 원활하지 못하거나 직관적이지 못한 부분을 수정하여 코드의 흐름 및 클래스 관계를 더 윤활하게 만든다.
6주차(12/7~12/13)	- 코드 리팩토링 #2 추가적으로 리팩트링을 실행하고, 프로그램의 속도를 더 개선할 수 있는 방향으로 코드를 수정한다. - 버그 수정 프로그램을 사용하며 경험한 버그를 수정한다.

ii. 주차별 작업 계획(실제 진행)

1주차(11/2~11/8)	- txt/csv 파일 형식 구상 DB에 넣을 데이터를 초기에 저장할 수 있도록 역할을 할 txt와 csv 파일의 형식을 구상하여, 이 파일을 어떻게 불러올지를 결정한다. 이 파일들에는 단어, 그 뜻이 공백, 개행으로 구분되어 들어간다. - 단어장 정보/setting/진행 상황을 저장할 DB 형식 구성 단어 정보를 담고 있는 단어장, 현재 상태와 진행 상황을 기록할 DB 형식을 계획한다. DB에는 단어 정보, 현재 확인한 단어, 어떤 단어를 알고 있는 정도 등의 정보가 담길 것이다. - txt/csv 파일 파서 구현 txt/csv 파일의 파서를 구현해 형식에 맞게 정보를 읽어 DB에 저장할 수 있도록 하고, 읽어온 정보들을 초기화한다. - 원시적 DB 구현 DB에서 단순히 랜덤 단어, 단어장 목록 등을 추출해오고 txt/csv 파일을 파싱한 결과를 DB에 저장하기 위한 기능을 구현한다.
2주차(11/9~11/15)	- swing을 이용한 프로그램 시작 화면 만들기 단어장 DB의 리스트를 설정 파일을 읽어옴으로써 list로써 띄우고, 단어장을 선택할 시 해당하는 Menu GUI로 이동하도록 한다. - Menu GUI 구성 추후 Quiz, Learn GUI와 결합될 수 있도록 단순한 기능만을 구현하고, 메인 화면 GUI와 결합시켜 작동을 확인한다(메인 화면에서 선택한 단어장을 정상적으로 불러오는지 확인한다). - 캡슐화를 위한 DB 메소드 추가 Menu GUI, 메인 화면 GUI를 사용하며 DB 클래스에서 단어장 목록 불러오기 등 필요한 정보들을 제공하는 메소드를 추가한다.
3주차(11/16~11/22)	- 단순 랜덤 선택 방식 구현 DB 클래스에서 단어장의 단어를 랜덤으로 추출할 수 있는 기능을 구현하고, 테스트한다.

	- Learn GUI 기능의 구현 #1 Learn GUI 틀을 구현하고, 그 결과에 따라 단어의 난이도를 재배치하는 기능을 넣는다. - DB 메소드 추가/수정 단어를 랜덤으로 선택하는 기능 이외에 단어장에 배우지 않은 단어가 남았는지를 확인하는 메소드, 단어장을 추가하는 메소드, 단어를 넣는 메소드를 구현하고, txt/csv 파일을 파싱해 넣는 메소드의 로직을 수정한다.
4주차(11/23~11/29)	- Learn GUI 기능의 구현 #2 확인하지 않은 단어를 랜덤으로 추출하고, 단어를 아는 정도의 척도를 3단계로 나누어 사용자가 선택한 결과에 따라 그 값을 변경한다. - DB 메소드 추가/수정 일부 DB 메소드를 수정/제거하고, 단어를 선택하는 메소드를 개선하고 단어를 아는 정도를 나타내는 척도를 수정하는 메소드를 추가한다. - 코드 리팩토링 #1 DB 메소드의 로직을 수정하여 오류를 수정한다.
5주차(11/30~12/6)	- Learn GUI 기능의 구현 #3 배워야 할 단어가 남지 않았을 때, 경고 메시지를 띄워 사용자가 선택하도록 하는 기능을 추가한다. Learn GUI를 더 다듬고 개선한다. - Quiz GUI 프로토타입 작성 단어장에서 단어를 무작위로 불러와 그 단어와 뜻이 정상적으로 디스플레이 되는지 확인한다. - DB 메소드 추가/수정 단어장에서 단어를 랜덤으로 가져오는 메소드와 단어장의 모든 단어의 아는 정도를 초기화하는 메소드를 만든다. - 코드 리팩토링 #2 모든 DB 메소드에서 connection을 받아오는 부분의 로직을 수정한다.
6주차(12/7~12/13)	- 단어장 선택 화면 탐색/삭제 기능 추가

	단어장을 탐색할 수 있는 기능을 추가하고, 선택한 단어장을 삭제할 수 있는 기능을 추가한다. - 단순한 학습 통계 화면 구현 현재 배운 단어와 남은 단어의 수를 차트에 띄워 보여주는 화면을 추가한다. - Quiz GUI 완성 Quiz 패널들을 완성해 Quiz GUI에 붙여 출력하도록 한다. - Layout 변경 모든 Frame의 Layout을 더 유동적인 배치를 지원하는 GridBagLayout으로 수정한다. - DB 메소드 추가 위 기능들을 지원하기 위해 DB에서 필요한 메소드를 구현한다. - GUI 디자인 변경 GUI들의 margin, 색과 글씨들의 색을 설정해 가독성을 높인다.

Ⅲ. 프로젝트 결과

ⅰ. 계획서 대비 추가/수정된 부분

- txt/csv 파일을 기반으로 데이터를 저장하고 관리하는 것이 아닌, DB를 이용하여 데이터를 저장 및 관리한다.

- 날짜에 따른 학습 진전률 대신, 전체 단어 대비 학습률을 디스플레이 한다.

- 단어의 난이도에 따라 나오는 빈도가 조정되는 것이 아닌, 단순 랜덤으로 단어를 추출하고, 단어의 난이도를 다른 목적으로 사용하였다.

ⅱ. 깃허브 링크 주소

https://github.com/MariAli-Lover/MemWord

ⅲ. 동작 데모 영상 링크 주소

https://youtu.be/UrXxuG3shZs

Ⅳ. 회고

ⅰ. 프로젝트를 진행하며 느낀 점

이번 프로젝트는 뚜렷한 목표를 가지고 진행하는 것 중에서는 처음으로 경험하게 된 프로젝트였다. 예상외로 프로젝트를 대부분 처음 설계한대로 따라갈 수 있었기에 다행이라고 느꼈다. 하지만 구현, 로직의 한계로 인해 중간중간 기능 명세를 바꾸고, 몇 개의 기능은 어쩔 수 없이 드롭 해야 했기 때문에 큰 아쉬움이 느껴졌다. 그러나 이것은 역으로 말하면 프로젝트를 구현하는 데에 있어 처음의 계획에서, 어떤 기능을 선택하고, 어떤 기능을 후순위로 하거나 드롭 하는 것이 프로젝트의 명세 및 기능을 맞추는 데에 있어서 중요하다는 것 또한 깨달을 수 있는 뜻 깊은 시간이었다.

아마도 이 프로그램을 작성하며 가장 즐거웠던 부분은 DB 클래스를 작성하는 것이었을 것이다. DB 클래스를 작성하며, DB에서 처리하는 부분을 프로그램의 다른 부분이 관여할 필요 없이 전부 캡슐화 시키는 작업은 일종의 희열감까지 느껴졌다. 또, 3-4주차 즈음 발생한 DB busy 현상은 고치기 위해 DB의 로직을 거의 대부분 뜯어 고치는 강경책을 취했고, 혹여나 작동하지 않고 프로그램이 망가질까 노심초사 했으나, 내가 생각한 대로 프로그램의 오류를 고칠 수 있었기에 더더욱 기억에 남는 것 같다. 추후 타 프로젝트를 진행하면서도, 이 프로젝트에서 경험한 DB 관리 기법은 큰 도움이 될 수 있을 것이라고 생각한다.

이 프로그램을 작성하며 가장 어려웠던 부분은 Frame 사이를 이동시키는 방법을 생각하는 것과, 프로그램의 GUI를 전체적으로 다듬는 것이었다. 자바 Swing의 GridBagLayout은 사용법을 이해하는 것도 비교적 난해한 편이었으나, 타 레이아웃에 비해서 많은 수고를 요했기에 더 기억에 남는다. 프로그램을 기동하고, GUI 및 레이아웃을 비교적 미세하게 조정하고, 다시 프로그램을 기동하는 과정을 수 없이 반복해서 완성한 GUI이기에 많은 힘이 들었고, 비록 투박할지언정 많은 애착이 가는 부분이다.

ⅱ. 향후 프로젝트 진행 시 고려해야 할 사항

- 만약 학습일자 트래킹을 추가해야 한다면, DB에 해당하는 사전 이름을 이용해 또 다른 테이블을 만들어 일시를 기준으로 정보를 저장한다(DB에서 일시 기능을 지원하는지 확인 필요).

- 학습해야 할 단어를 JLabel에 딱 맞게 출력해주는 기능을 추가한다.

- 실질적으로 단어를 알고 있는 정도에 따라서 단어를 biased하게 출력할 수 있는 방법이 무엇인지 생각해본다.

Ⅴ. 참고한 자료

1. Anki(단어 암기 프로그램)

https://github.com/dae/anki