<SimpleMerge>

**Team Project Report**

**권순우 김남웅 박지호 유동제 이수호 정연지**

**CAUCSE**

**2017-6-9**

# 1. 개요

본 보고서는 중앙대학교 컴퓨터공학부 2017-1학기의 ICT소프트웨어 공학 프로젝트의 진행과 결과에 대한 보고를 위하여 작성되었습니다.

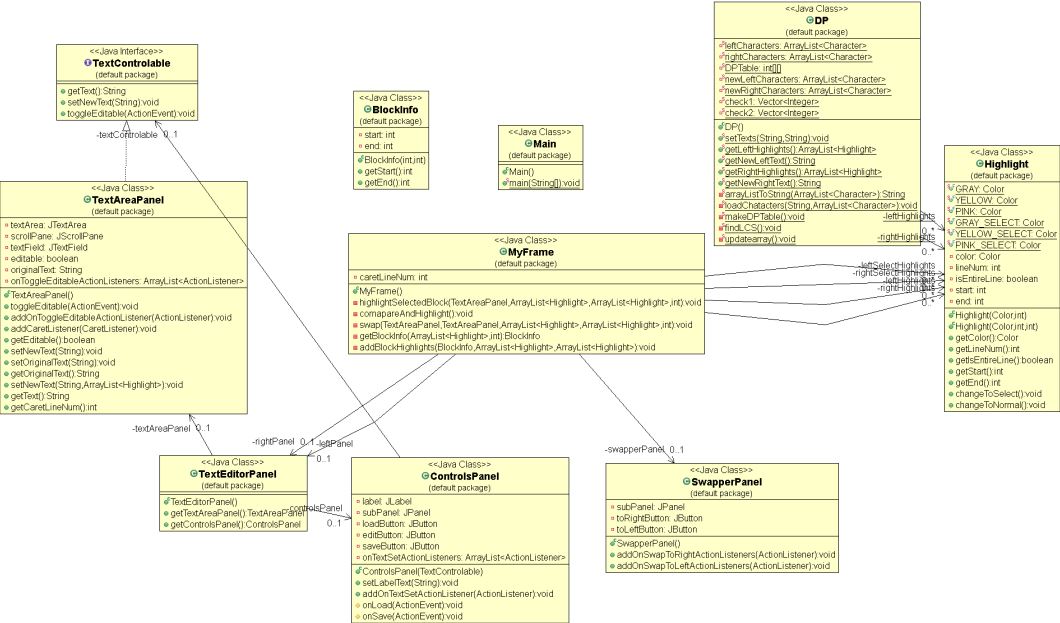
# 2. 시스템 모델

## 2.1 전제

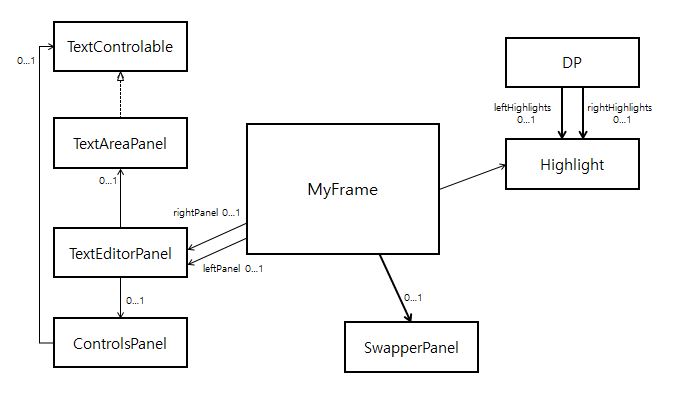
본 프로젝트에 제출된 프로그램의 개발에는 Java 프로그래밍 언어가 사용되었습니다. Junit과 EasyMock을 이용하여 Unit Test를 진행하였으며, MVC 모델을 활용하여 설계되었습니다.

## 2.2 MVC와 OOP (Class Diagram 첨부)

전체 Class Diagram



약소화



2.2.1 MVC

MVC 모델의 주요 목적은 조작과 출력을 데이터로부터 분리하는데 있습니다.

1. 모델: MyFrame 클래스가 모델의 역할을 맡게 됩니다. MyFrame은 컨트롤러에서 받은 사용자 입력에 따라 Highlight 클래스에서 필요한 정보를 요청하여 뷰에서 제시된 출력에 양쪽 패널의 각각 차이점 부분을 표시하는 정보를 넘겨줍니다. Highlight 클래스는 DP 클래스에 구현된 LCS (Longest Common Subsequence) 획득 알고리즘을 이용, 필요한 정보를 MyFrame으로 넘겨주게 됩니다. SwapperPanel에서 신호를 받을 경우, TextEditorPanel로 정보를 알려준 후, 각각 패널의 정보를 갱신, Highlight를 통해 새로운 n-LCS를 표시하라는 신호를 넘겨줍니다.
2. 컨트롤러: ControlsPanel 클래스와 SwapperPanel 클래스가 각각 다른 역할을 하게 됩니다. ControlsPanel은 GUI에 제시된 Load, Edit, Save 버튼에 대하여 사용자가 본 버튼을 사용하였는지의 여부를 확인합니다. Load와 Save의 경우, TextEditorPanel을 통하여 TextAreaPanel에서 각각의 기능을 사용할 수 있도록 요청하게 됩니다. Edit 버튼의 경우, TextControlable로 사용자 입력여부를 확인하여, TextAreaPanel의 Editable 상태를 true로 변경합니다. SwapperPanel 클래스는 Merge 기능을 하는 두 개의 버튼의 상태를 확인하여, MyFrame에 그 정보를 보내게 됩니다.
3. 뷰: TextAreaPanel은 불러와진 파일, 유저가 사용할 텍스트 편집 창 등을 TextEditorPanel, TextControlable에서 신호를 받아 상태를 변경하게 됩니다.

2.2.2 OOP Design Principles

본 프로젝트에서 가장 두드러지게 확인할 수 있는 설계 원칙은 Single-Responsibility Principle 입니다. 매우 명확한(Specific) 목적을 가진 프로그램의 설계가 목표였기에, 미래의 확장성에 대하여 비교적 자유로웠습니다.

Single-Responsibility Principle

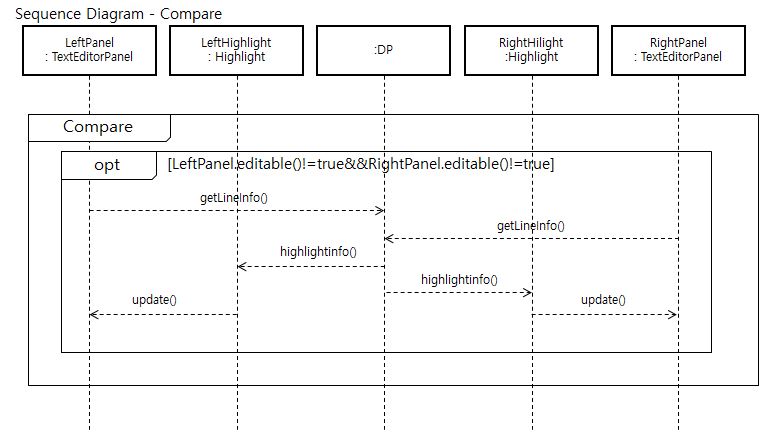
사용자의 입력을 받음을 확인하는 ControlsPanel과 SwapperPanel은 서로 각각의 역할을 가지고 있습니다. ControlsPanel의 경우 파일의 저장, 불러오기, 편집에만 관여하며, SwapperPanel은 두 n-LCS를 합치는 데에만 관여하게 됩니다. 또한 ControlsPanel에서 편집 버튼의 입력이 확인될 때, 자신이 직접 조작하는 것이 아닌, TextControlable에게 신호만 보내며, 그 클래스는 다시 TextAreaPanel의 상태를 변경하는 형태로 구현되었습니다.

추가로 하이라이트는 TextAreaPanel에서 일어나지만, 실제로 LCS를 계산하는 함수는 Highlight이므로 역할이 분리되어 있습니다.

## 2.3 주요 기능 작동 절차 (Sequence Diagram 첨부)

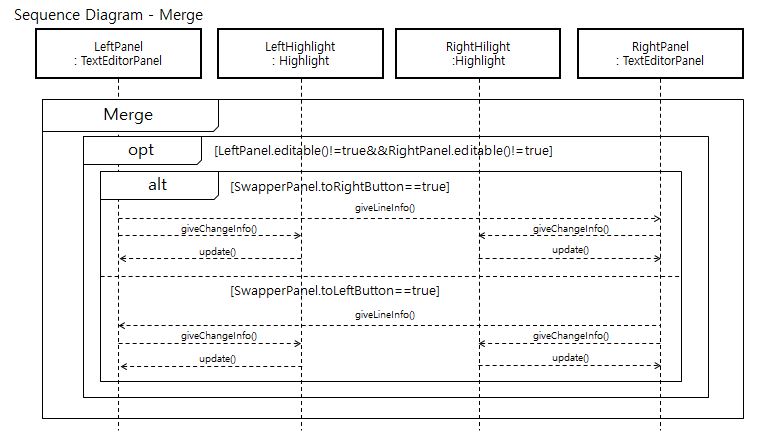
본 항목에서는 주요 기능이라고 할 수 있는 Compare와 Merge 기능이 어떠한 절차로 작동하는지를 Sequence Diagram을 통하여 설명합니다.

2.3.1 Compare



Compare 기능의 경우 양쪽 TextAreaPanel의 editable이 false 일 때(두 패널 모두 편집 불가능 상태일 때) 만 작동하게 됩니다. TextAreaPanel의 데이터 정보가 DP 클래스에 도착하면, DP 클래스 에서는 n-LCS를 찾아내어 각각의 Highlight으로 정보를 넘겨주게 됩니다. Highlight는 다시 그것을 각각의 TextAreaPanel에 표시하게 됩니다.

2.3.2 Merge

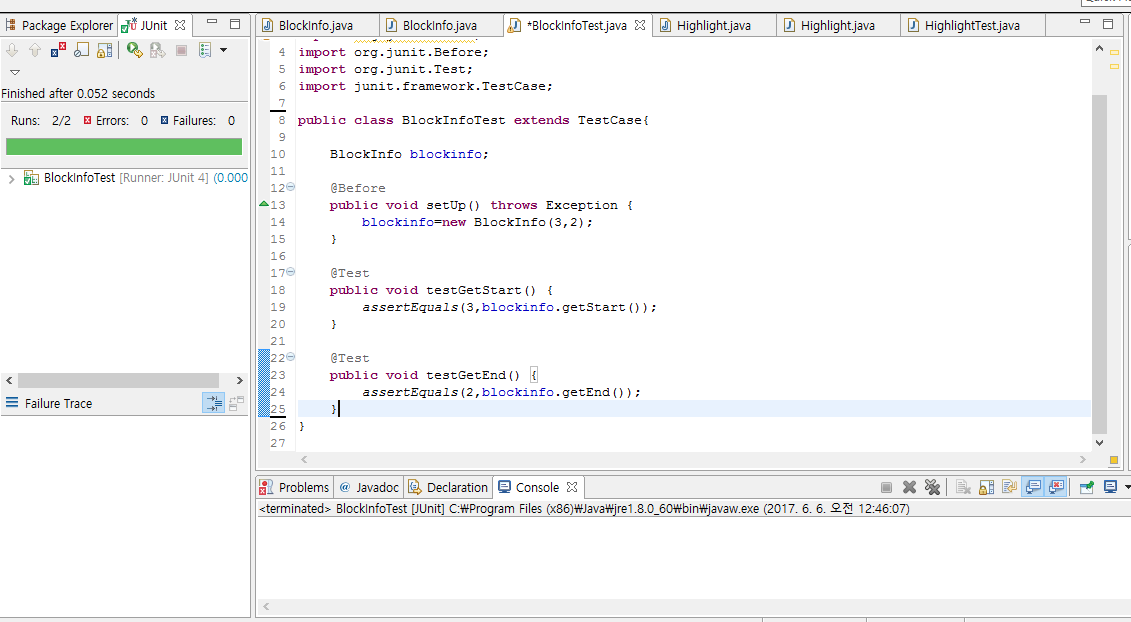


Merge 기능 역시 양쪽 TextAreaPanel의 editable이 false 일 때만 작동하게 됩니다. “>>” 버튼과 “<<” 중 어느 버튼이 눌렸느냐에 따라 alt fragment의 결정이 갈라서게 됩니다. 하지만 구조는 완전히 똑같습니다. 한쪽의 TextAreaPanel의 정보를 반대쪽으로 보내게 되면, 각각의 TextAreaPanel은 Highlight 클래스에 둘이 일치하게 되었다는 정보를 주게 되어, 각 패널의 표기 상태를 업데이트 합니다.

## 2.4 실험 결과

Unit Test를 위하여 Junit과 Easymock를 활용하였습니다. Easymock으로 Mock을 생성하여 DP 클래스가 정상적으로 작동한다고 가정하고, 거기서 도출되는 결과값을 각각 주요 함수들에 대입합니다. 테스트의 결과는 아래에 제시되었습니다. GUI 테스트는 기간 내에 완수하지 못하였습니다.

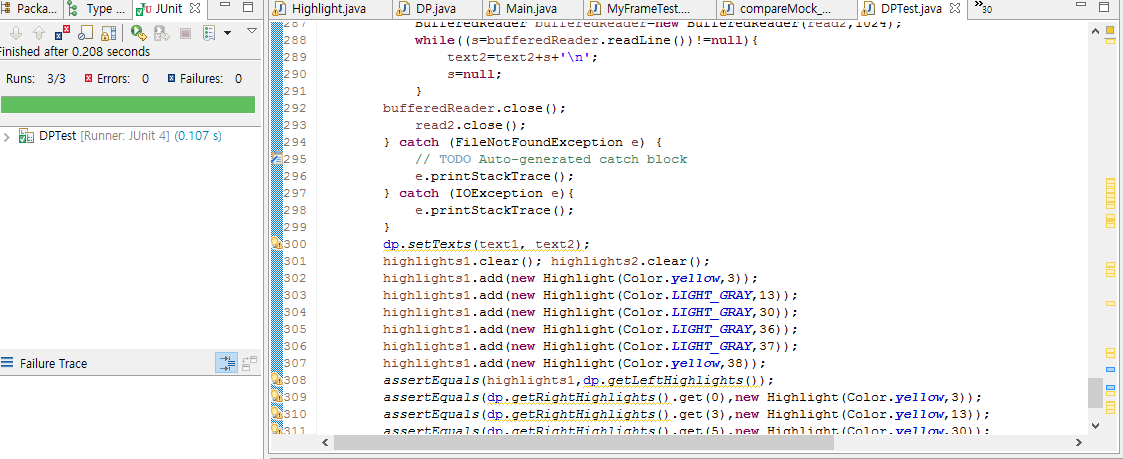
1. BlockInfo



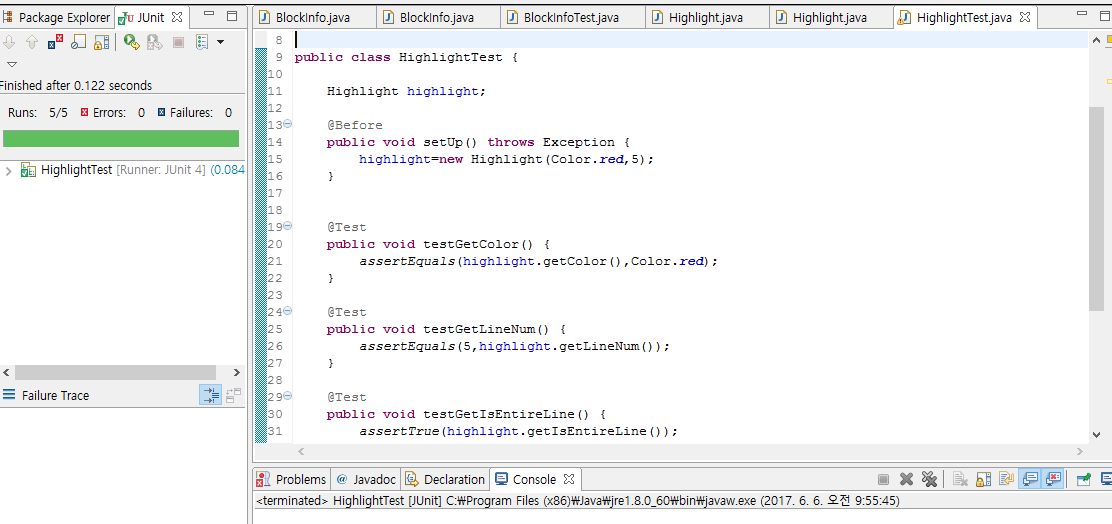
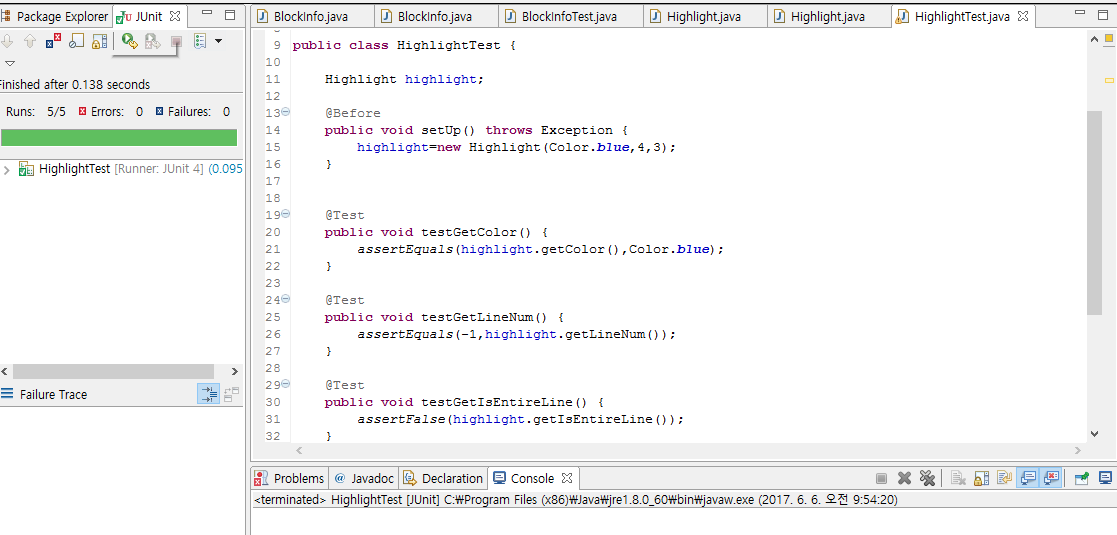
1. CompareAndHighlight



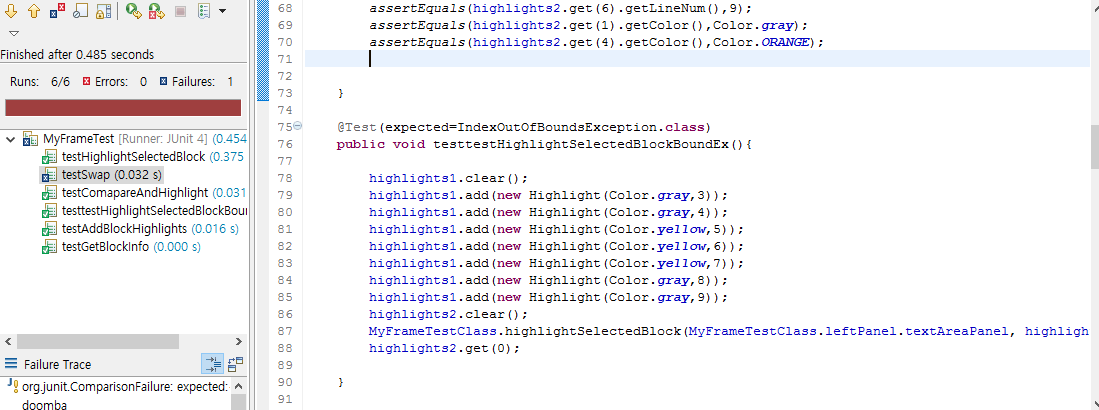
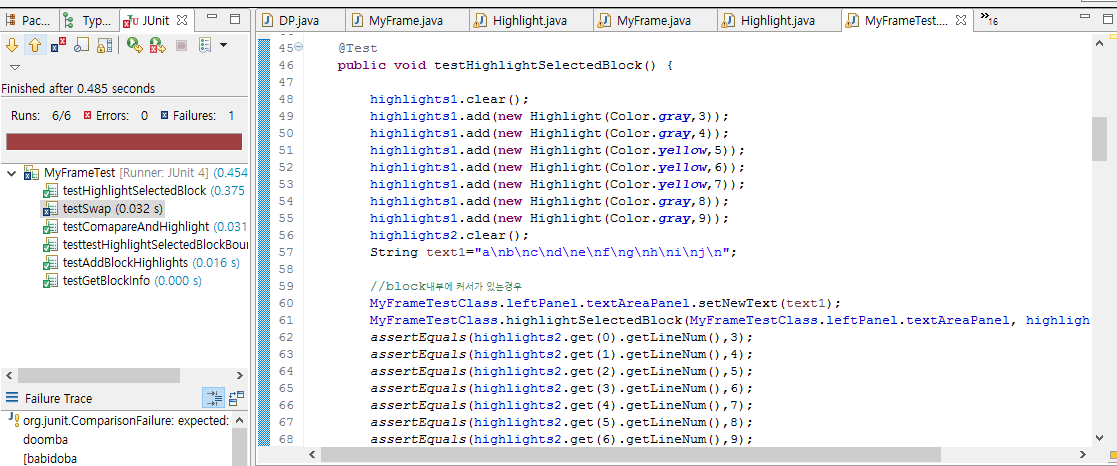
1. DP



1. Highlight



1. HighlightSelectedBlock



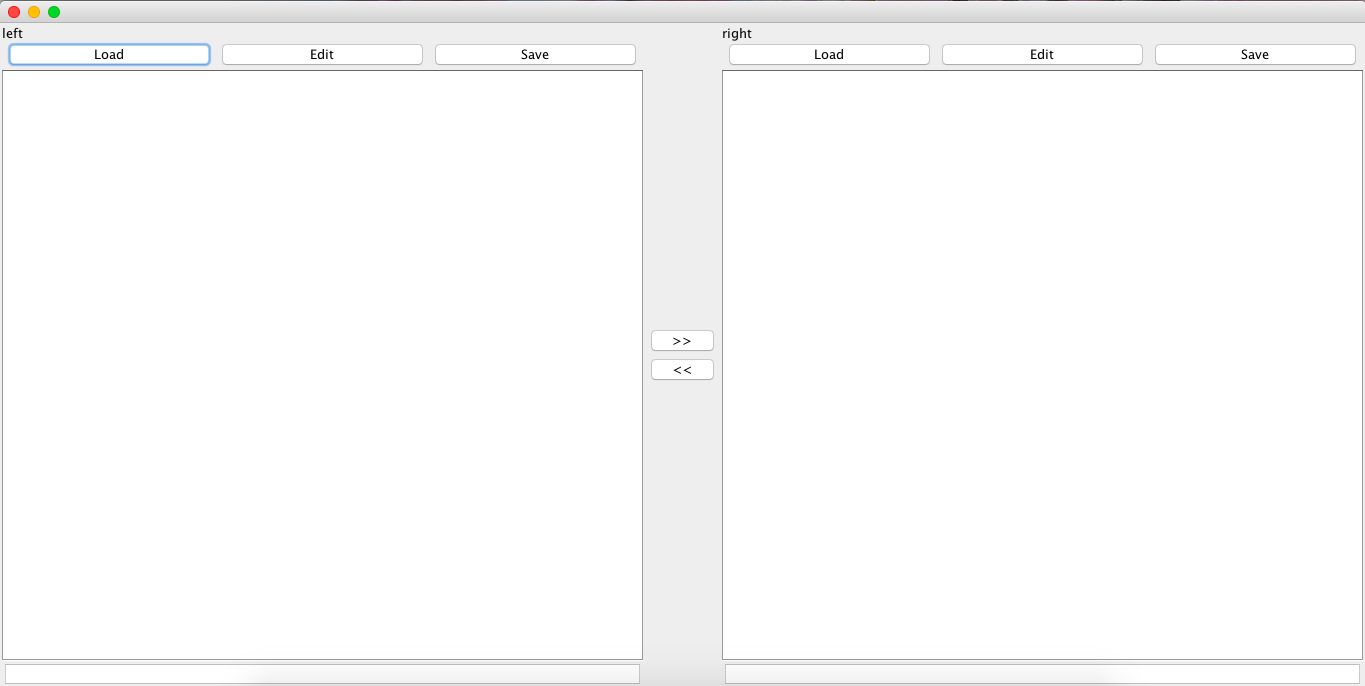
이 항목의 Failure는 테스트 당시에 다른 함수의 오류로 인해 발생한 것이며, 본 항목의 테스트에는 영향이 없습니다.

1. Swap

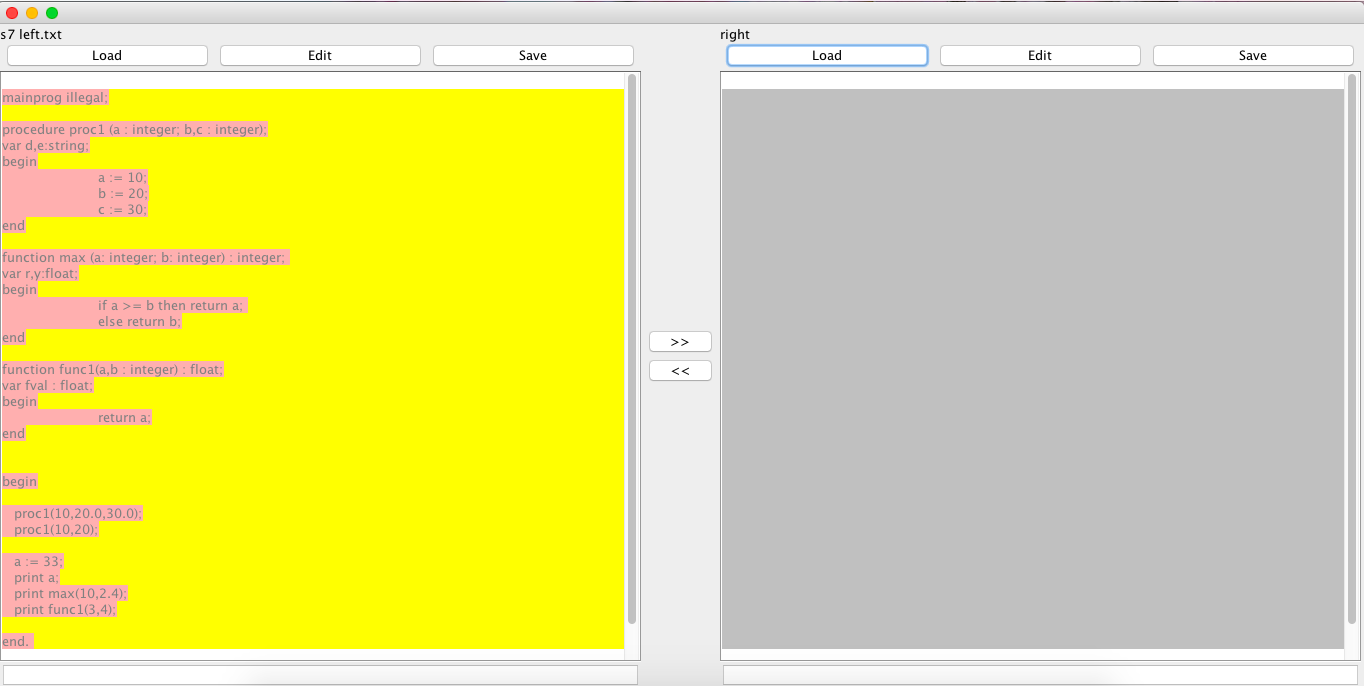


## 2.5 사용 예시

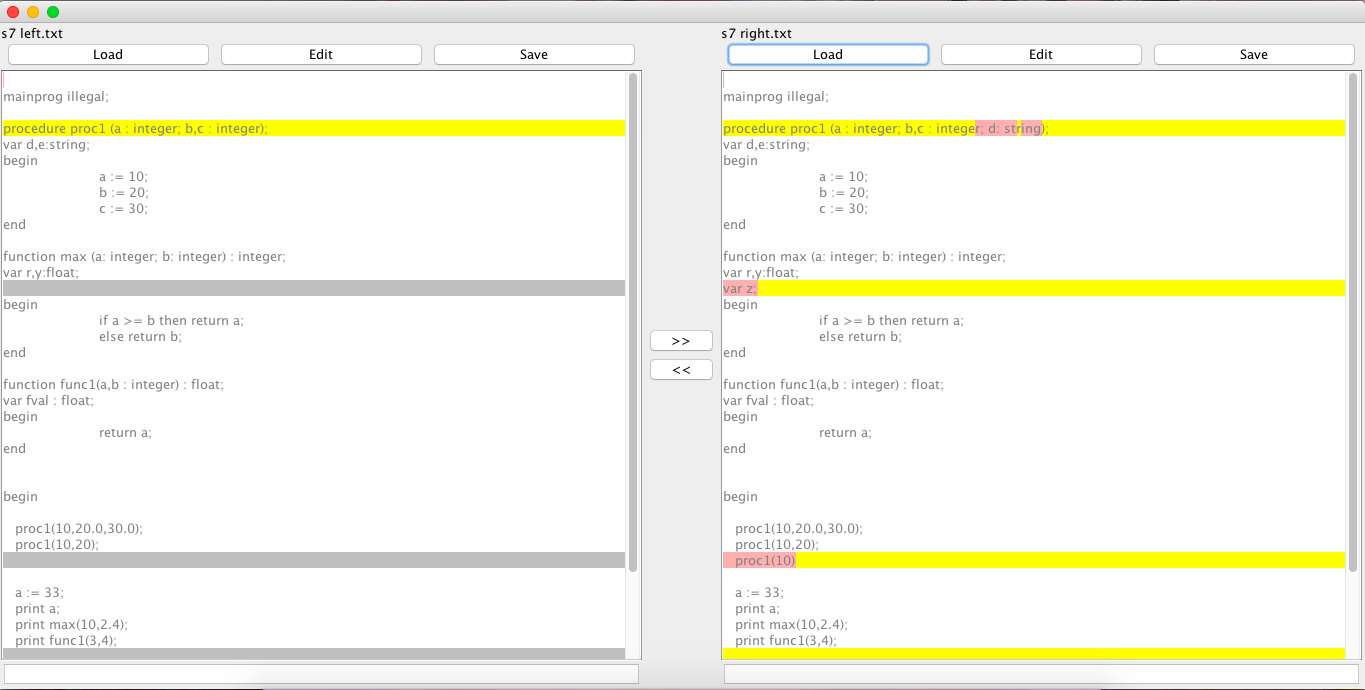
1) 실행 화면



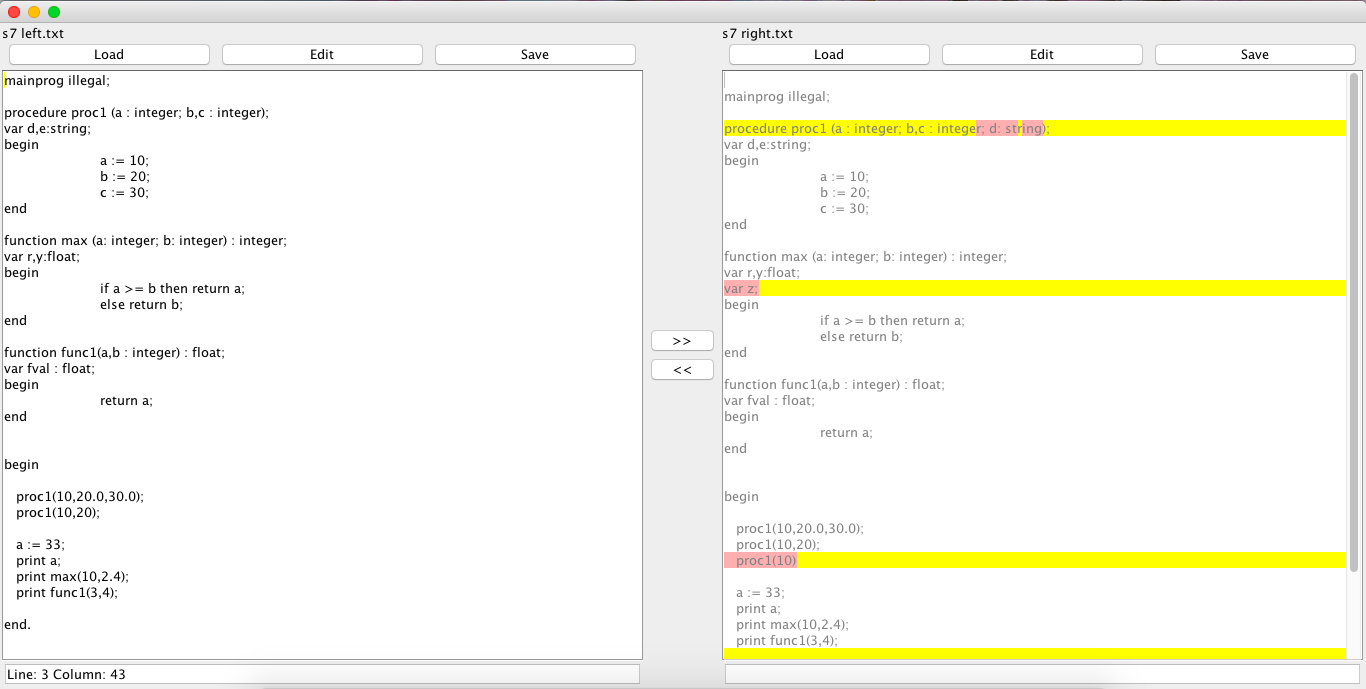
2) 왼쪽 패널 로드



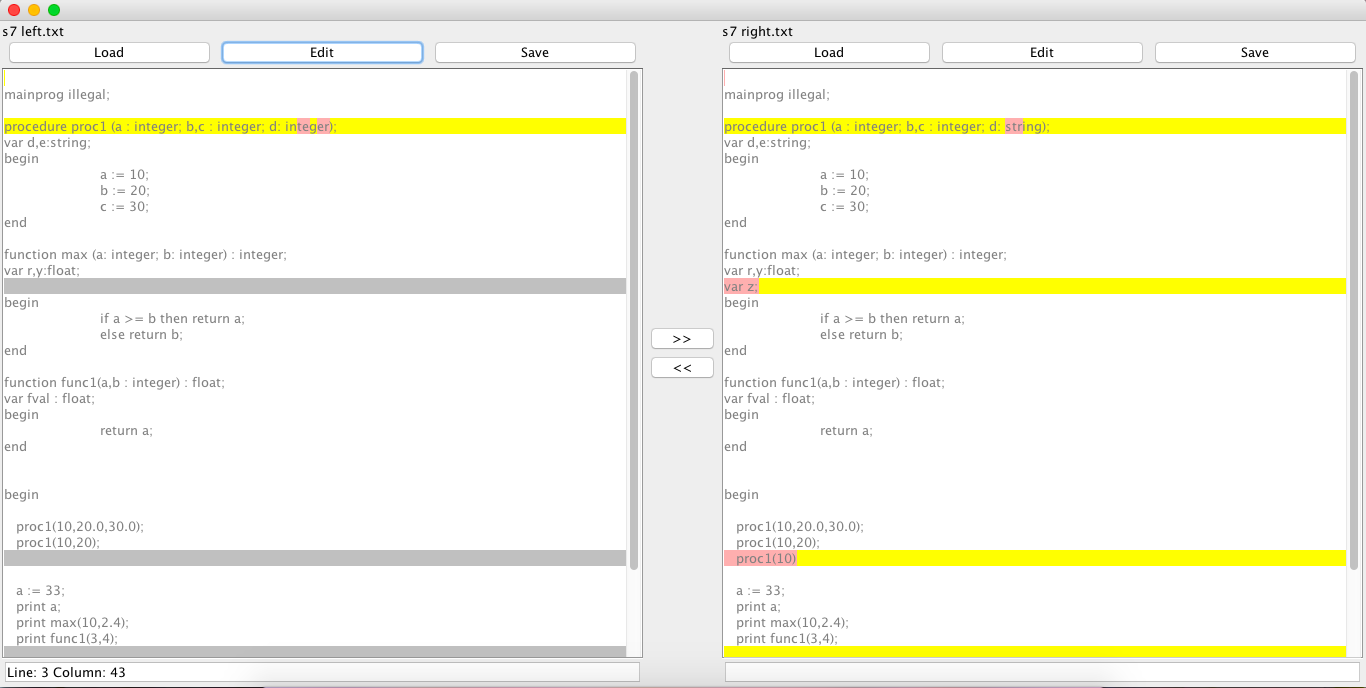
3) 양쪽 패널 로드



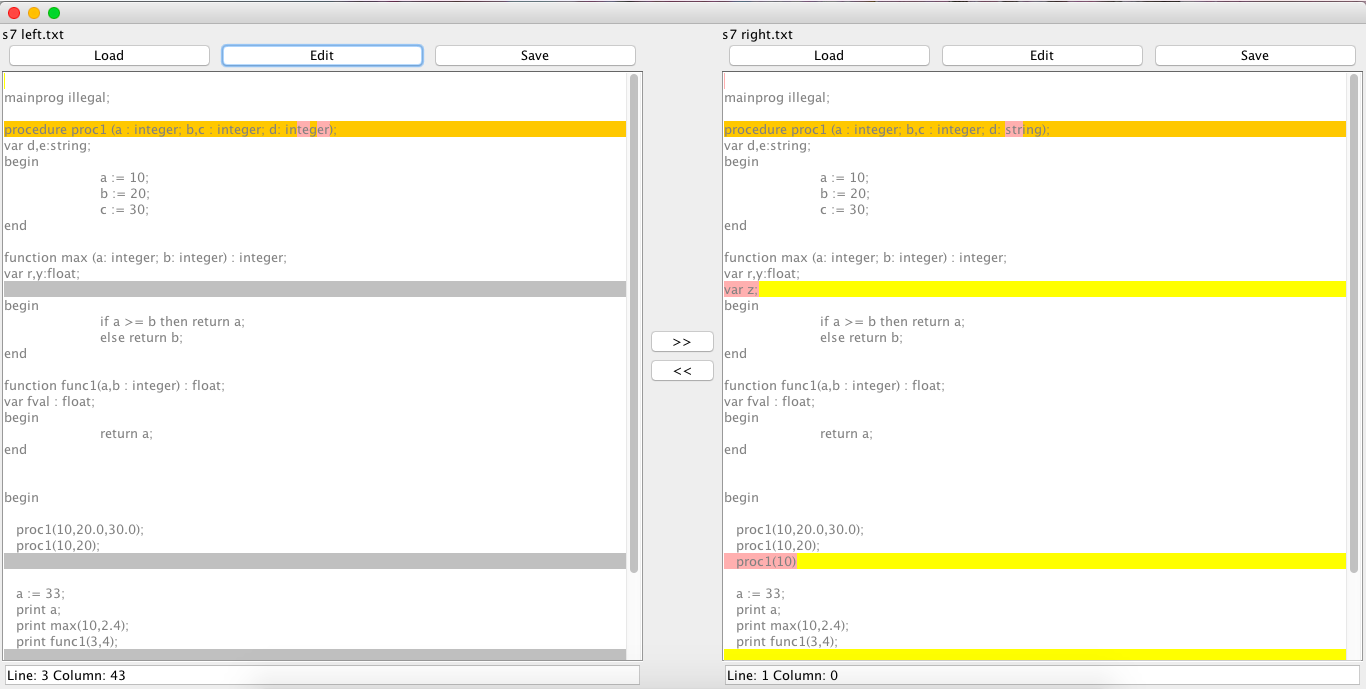
4) 왼쪽 패널 편집



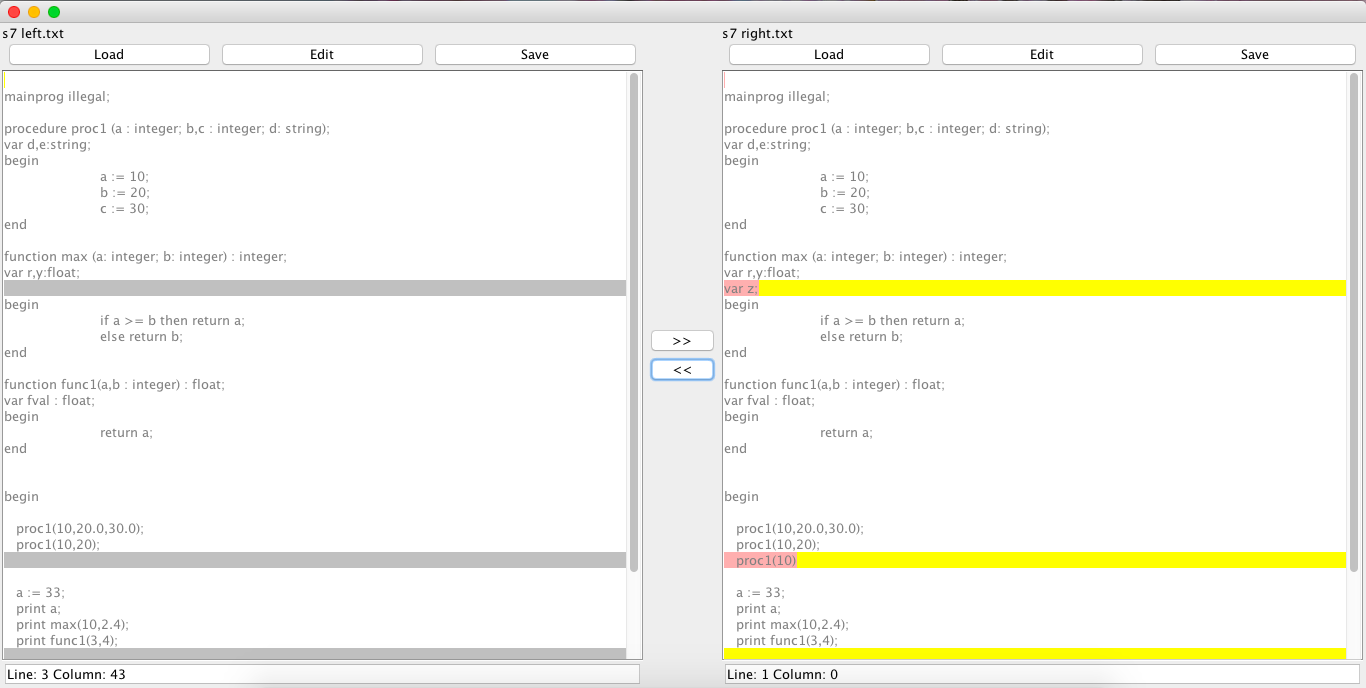
5) 편집 후 표기 변경



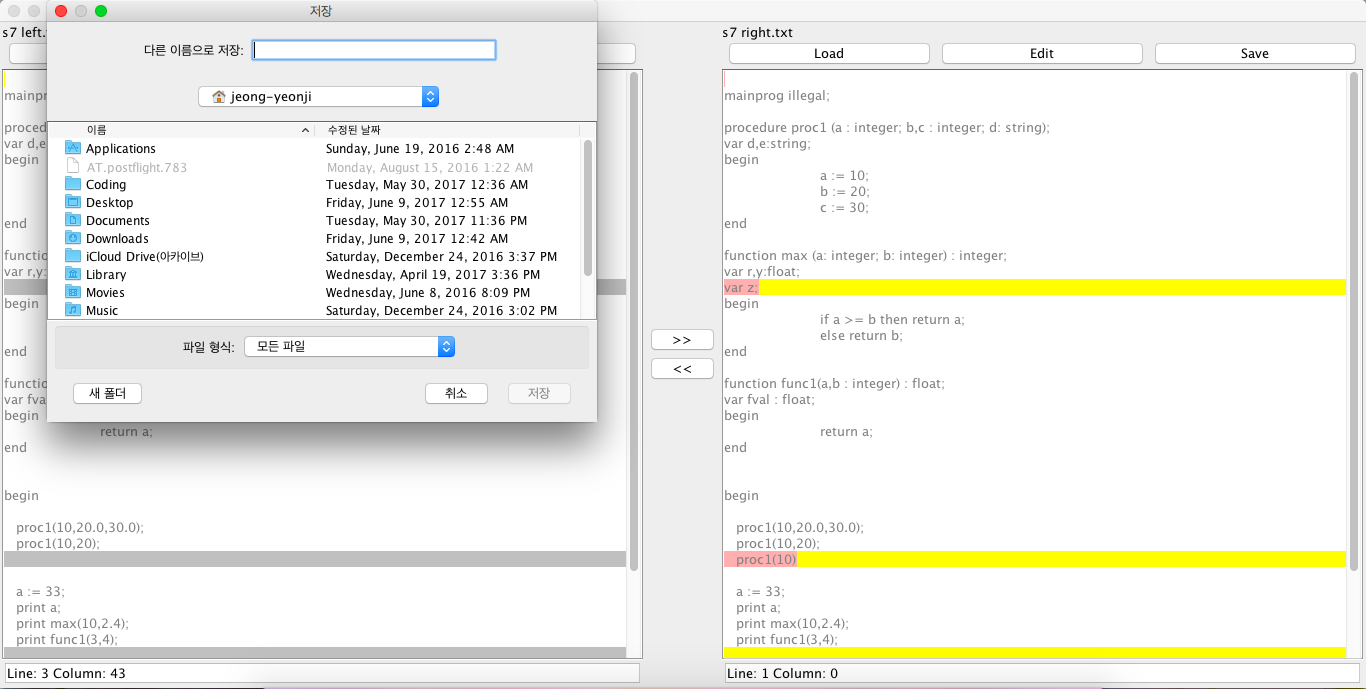
6) 부분 선택



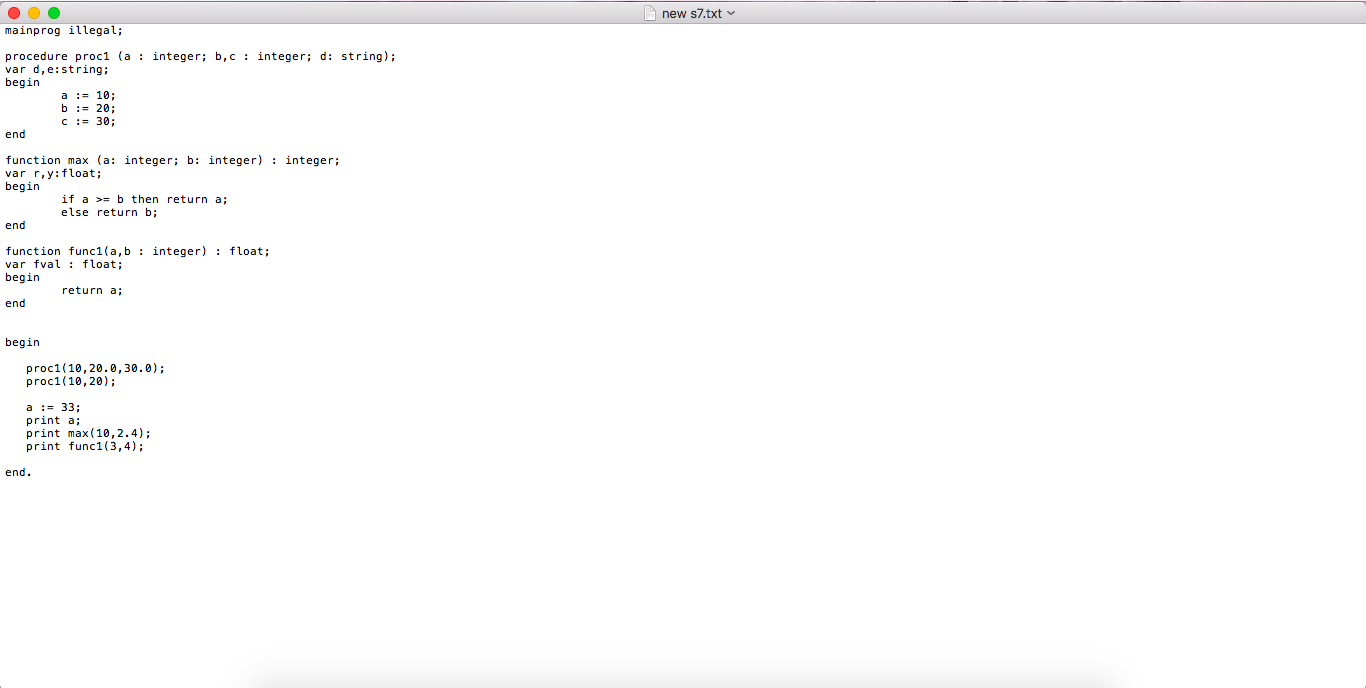
7) Merge



8) 저장



9) 저장된 파일 확인



## 2.6 결점

편집 기능에 약간의 기능장애가 발생합니다. 평범한 텍스트 에디터와는 다르게, 문자가 입력되면, 입력 커서가 입력된 문자의 앞쪽에 다시 배치되게 됩니다. 편집이 가능은 하나, 일반적인 사용자에게 익숙한 방식으로 되지 않습니다.

# 3. Project Management Report

## 3.1 개요

이번 프로젝트에서 저희는 github를 활용하기 위하여 노력하였습니다. 팀원 모두 github를 사용하는 경험이 없었기에, 활용하는데 어려움이 따랐습니다.

## 3.2 진행

<https://github.com/aquashdw/simpleMerge>

위 링크에서 이번 프로젝트가 진행되었습니다. 아쉽게도 개발 단계에서 본 부분이 제대로 활용되지 않았습니다. 후에 첫 prototype이 완성된 이후 github의 활용도가 조금씩 늘어나기 시작하였습니다.

## 3.3 느낀 점

Github를 정확히 활용하는 방법을 터득하게 될 경우, 모든 팀원들이 같은 컴퓨터에서 작업하듯이 프로젝트를 진행하는 것이 가능해질 것으로 느꼈습니다. 무엇보다 git bash에서 git 커맨드를 활용한 프로젝트의 관리는 push와 pull을 통하여 서로의 일관성을 높이는데 뛰어난 역할을 할 것으로 기대합니다. 다음 프로젝트에서 활용하기 위한 첫 경험이라고 보여집니다.