
Term-project #1

과목명	오픈소스SW프로젝트
-----	------------

제출일	2023.05.14.(일)
-----	----------------

팀 명	TEAM 20
-----	---------

팀 원	김지민(20204461)
-----	---------------

	장민호(20194668)
--	---------------

	조언욱(20193933)
--	---------------

	차현호(20183968)
--	---------------

I . GitHub repository

프로젝트 깃허브 주소는 [여기](#)에서 확인할 수 있다.

II . Details about implementation

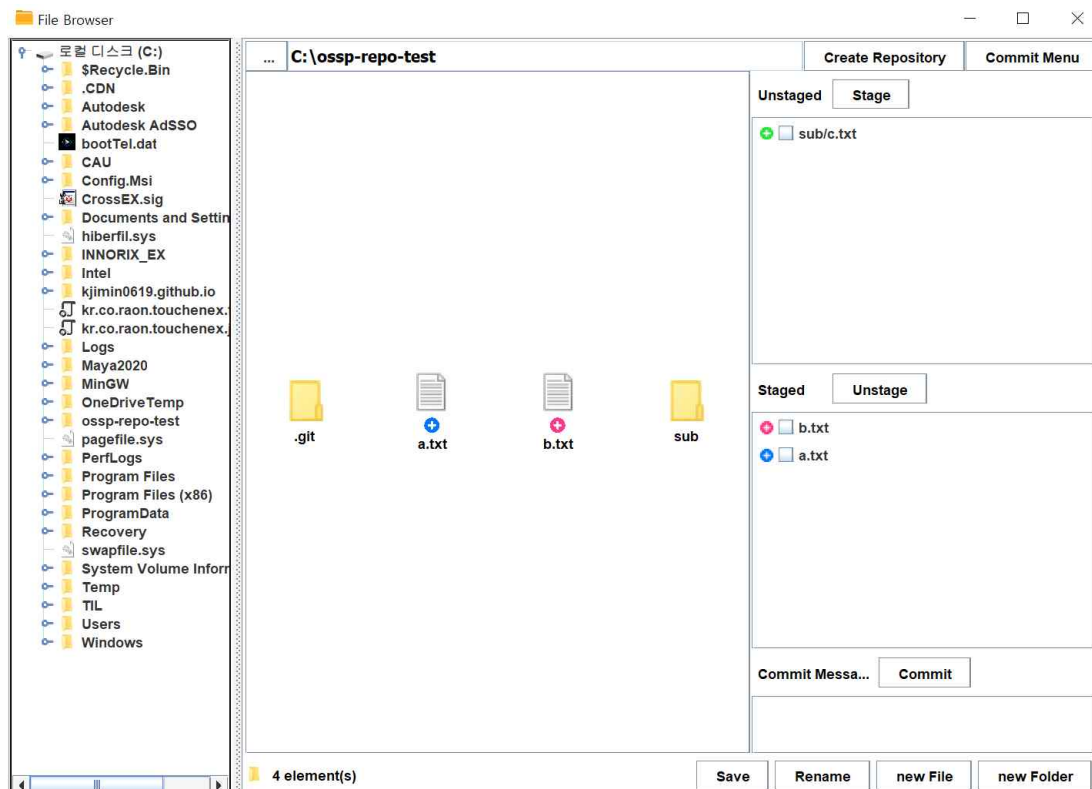
1. 주요 의존성 및 오픈소스

- (1) [JGit](#) : Eclipse Foundation에서 관리하는 Java의 Git 라이브러리이다.
- (2) Swing : Java로 작성된 GUI 라이브러리로 그래픽 인터페이스 구현을 위한 클래스를 제공한다.
- (3) [JGit Cookbook](#) : JGit을 편하게 사용할 수 있도록 example 및 code snippet을 제공하는 오픈소스이다.
- (4) [FileBrowser](#) : Java Swing을 활용한 파일 브라우징 오픈소스로, Eclipse Public License 2.0을 채택하여 소스 코드의 복제, 배포, 수정이 자유롭다. 단, 소스 코드 내 명시된 라이선스 정보를 그대로 유지한 상태에서 재배포해야 한다. 따라서 본 프로젝트에서는 Eclipse Public License 2.0을 적용한다.

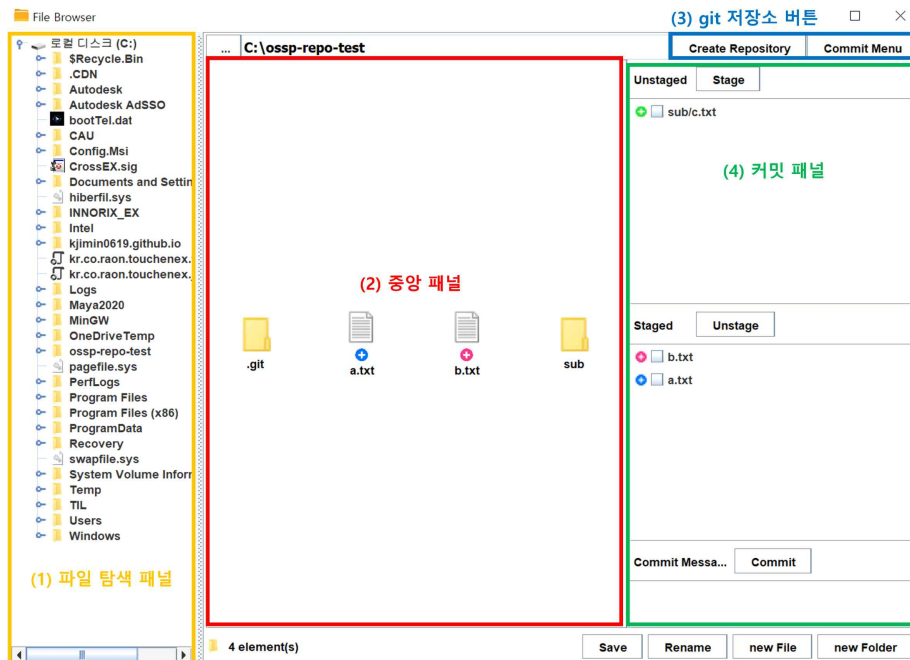
2. Graphical Design

2.1. GUI

프로젝트의 GUI 구성은 <그림1>과 같다.



<그림 1> GUI



<그림 2> GUI 내부 구성 소개

(1) 파일 탐색 패널

- 컴퓨터에 존재하는 파일 및 폴더를 탐색하는 패널이다. 더블클릭으로 원하는 폴더에 접근할 수 있다.

(2) 중앙 패널

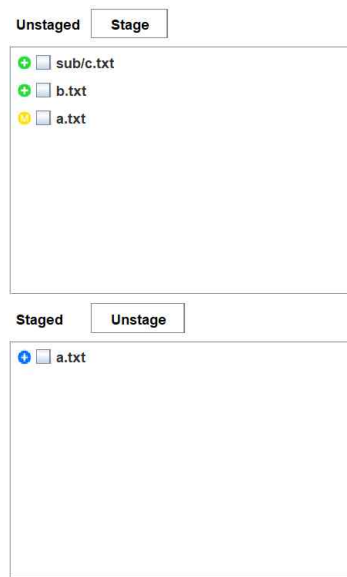
- 폴더의 내부 요소들을 보여준다. 선택된 폴더가 git repository라면 파일의 상태 정보(아이콘)를 함께 제공한다. 이때 local의 현재 상황만 반영하기 때문에 파일의 상태를 중복으로 표시하지 않으며 하위 폴더에 대한 상태 정보는 제공하지 않는다.
- 마우스 우클릭을 통해 파일의 현재 상태별 수행 가능한 git 명령어 정보를 확인할 수 있다.

(3) git 저장소 버튼

- *Create Repository* 버튼으로 일반 폴더를 git repository로 만들 수 있다.
- *Commit Menu* 버튼으로 add, commit 등 git 기능을 수행하는 패널(이하 커밋패널)을 오픈할 수 있다.

(4) 커밋 패널






- 주요 git 기능을 제공하는 패널로 git 저장소에서만 열 수 있다.
- 중앙 패널과 마찬가지로 파일의 상태 정보를 제공한다. 이때 파일의 상태는 크게 *staged*와 *unstaged*로 구분하며 *untracked*, *modified* 등 세부 상태는 아이콘으로 표시한다.
- *unstage*, *staged* 버튼을 통해 특정 파일을 staging area에 올리거나, staging area에 있는 파일을 restore 할 수 있다.
- commit message를 작성하고 하단 *Commit* 버튼을 통해 staging area에 있는 모든 파일을 *committed(unmodified)* 상태로 바꿀 수 있다.
- 커밋 패널에는 한 파일에 대한 여러 상태 정보를 제공한다(<그림 3> 참고).



<그림 3> 파일 a.txt에 대한 중복
상태 표시

2.2. git 상태 아이콘

파일의 상태는 두 가지 범주로 나뉜다.

Unstaged			Staged		
					
Untracked	Modified	Ignored	Added	Changed	Removed

<표 1> 파일 상태 아이콘

(1) Unstaged: 변경된 파일이나 새로 생성한 파일 중 *git add* 명령을 실행하기 전의 상태

- Untracked: 새로 생성된 파일로 git이 추적하지 않는 상태
- Modified: 기존 파일이 수정된 상태
- Ignored: git이 추적하지 않도록 설정한 파일

(2) Staged: 수정된 파일이나 새로 생성한 파일 중 *git add* 명령을 실행하여 git이 추적할 준비가 된 상태

- Added: 새로 생성한 파일이 스테이지 영역에 추가된 상태
- Changed: *unmodified* 상태였던 파일이 수정되고 *git add* 명령을 실행하여 스테이지 영역에 추가된 상태
- Removed: 기존 파일이 삭제되고 *git add* 명령을 실행하여 스테이지 영역에 추가된 상태

(3) 기타 참고사항

- `commit` 명령으로 파일이 `committed(unmodified)` 상태로 바뀌면 해당 파일의 아이콘은 사라진다. 이후 해당 파일에 대한 변경사항이 생기면 그에 상응하는 상태 아이콘이 재생성된다.
- `.gitignore` 파일과 같이 `ignored` 상태에 대한 정보(아이콘)는 제공하지 않는다.

3. 주요 기능

GUI 기반의 Git Repository 관리 서비스를 제공하기 위해 구현된 기능을 소개한다.

3.1. Git repository 생성

`Create Repository` 버튼을 통해 git 저장소를 생성할 수 있다. `.git` 폴더의 유무를 통해 정상적으로 저장소가 생성됐음을 확인할 수 있다. 이미 `.git` 폴더가 생성된 저장소는 중복으로 git repository로 만들 수 없다.


3.2. Version controlling

본 프로젝트에서 지원하는 버전 관리 서비스는 두 가지로 나눌 수 있다.

(1) 파일 상태별 팝업 메뉴 제공

중앙 패널에서 마우스 우클릭 시 파일의 현 상태에 따라 서로 다른 팝업 메뉴가 나타난다. 선택된 메뉴에 맞게 실시간으로 파일의 상태가 변하며 이는 중앙 패널 및 커밋 패널에서 확인할 수 있다. 아래 <표 2>는 파일의 상태에 따른 팝업 메뉴에 대한 예시이다.

파일 상태	팝업 메뉴	해당 git 명령어
 untracked	Add to git -> Rename -> new Folder -> new File -> Open in Desktop	git add
 modified	Add to git Unmodifying -> Rename -> new Folder -> new File -> Open in Desktop	git add git restore

 staged	<div>Unstage changes</div> <div>-> Rename</div> <div>-> new Folder</div> <div>-> new File</div> <div>-> Open in Desktop</div>	git restore --staged
committed(unmodified)	<div>Untracking</div> <div>Delete file</div> <div>Rename tracked file</div> <div>-> Rename</div> <div>-> new Folder</div> <div>-> new File</div> <div>-> Open in Desktop</div>	git rm --cached git rm git mv

<표 2> 파일의 상태에 따른 팝업 메뉴

(2) Staging area 관리 및 Commit 기능 제공

- *Commit Menu* 버튼을 통해 커밋 패널을 열 수 있다. 단, git 저장소가 아닌 폴더에서 커밋 패널은 열리지 않는다.
- 커밋 패널을 통해 Staging area(Staged)에 올라온 파일들의 목록을 확인할 수 있다.
- *Commit* 버튼을 통해 commit 메시지를 작성하고 Staging area에 올라온 파일의 변경 사항에 대한 스냅샷을 만들어 저장할 수 있다.
- *committed(unmodified)*상태가 된 파일들은 즉시 Staging area에서 사라지고 중앙 패널에서 상태 아이콘이 붙지 않는다.
- 커밋 패널의 *Stage*, *Unstage* 버튼을 통해 Staging area에 올라온 파일을 Unstaging 할 수 있으며, 역으로 *Unstaged* 상태의 파일을 손쉽게 Staging area로 올릴 수 있다.



<그림 4> 커밋 패널 예시

III . Guideline to install and run the service

본 프로젝트는 Windows를 타겟 플랫폼으로 하여 IntelliJ IDEA Ultimate 환경에서 개발되었다. JAVA, JDK-17을 사용하고 gradle 8.0을 빌드 도구로 채택하였으며 이와 동일한 개발환경에서 실행하는 것을 권장한다.

프로젝트를 실행하는 순서는 다음과 같다.

1. Github Repository 복제
2. IntelliJ 환경 설정
3. Run

실행 과정에 대한 자세하고 구체적인 가이드라인은 깃허브 README 문서의 [How to Execute](#)에서 확인할 수 있다.

IV . Collaboration history

1. 협업 방식

- 노션을 활용한 [회의록](#) 작성 및 관련 [문서](#) 아카이빙
- 주 1회 오프라인 회의 진행

2. 역할 분담

	GUI	git 주요 기능 구현	기타
김지민	<ul style="list-style-type: none">repository 생성 버튼 제작중앙 패널 속 파일의 상태 별 아이콘 제공	<ul style="list-style-type: none">repository 생성 기능 구현	<ul style="list-style-type: none">제출 문서 및 리드미 작성
장민호		<ul style="list-style-type: none">commit 기능 구현	
조언욱	<ul style="list-style-type: none">우클릭 시 제공되는 팝업 메뉴 제작	<ul style="list-style-type: none">JGit 기반 git 명령어 전체 구현staged & unstaged 상태 전환 및 커밋 패널 업데이트 기능 구현	<ul style="list-style-type: none">코드 개선
차현호	<ul style="list-style-type: none">commit menu 버튼 제작unstaged, stage, commit 버튼, commit message란 등 커밋 패널의 전반적인 ui 생성	<ul style="list-style-type: none">unstaged, staged 파일 구별 및 리스트업 기능 구현	<ul style="list-style-type: none">전반적인 코드 개선 및 로직 재설정리드미 작성

<표 3> 역할 분담

3. 진행 과정

날짜	진행 상황
23-04	적절한 오픈 소스 탐색 및 프로그래밍 언어 선정
23-05-01	eclipse project를 gradle project로 변경
23-05-03	commit menu ui 추가
	커밋 패널 생성 완료
	commit menu, repository creation 버튼 기능 구현 완료

23-05-04	커밋 패널 기능 구현 완료
	팝업 메뉴 기능 구현 완료
23-05-05	파일 상태 아이콘 제작 완료
	커밋 패널 기능 개선
23-05-09	중앙 패널 파일 상태 표시 기능 개선
	커밋 메뉴 버튼 동작 개선
	커밋 패널 기능 개선
23-05-10	레포지토리 하위 폴더 동작 오류 개선 (status에 따른 팝업 메뉴 생성 및 선택 시 기능 동작)
	커밋 버튼 기능 구현 완료
	중앙 패널 파일 상태 아이콘 표시 완성
23-05-11	팝업 메뉴 오류 개선 및 일부 코드 수정
23-05-13	문서 및 리드미 작성 완료

<표 4> 개발 타임라인

프로젝트 개발은 팀원별 브랜치를 만들어 기능을 구현하고 완료 후 merge하는 방식으로 진행되었다. 진행하는 과정에 있어 생성된 브랜치는 9개로 각 브랜치에 대한 설명은 아래와 같다.

- master : 배포 브랜치
- develop : 메인 개발 브랜치
- feature_improvement : 전체 기능 개선을 위한 브랜치
- feature_commit_list : 커밋 패널 디자인 및 아이콘 디자인을 위한 브랜치
- design : 아이콘 디자인 수정을 위한 브랜치
- commit_feature : 커밋 패널 및 팝업 메뉴 기능 구현을 위한 브랜치(git 기능 메인 구현)
- feature_commit, checkSubdirectory : commit_feature 기능 개선을 위한 브랜치
- feature2 : repository 생성 기능 구현을 위한 브랜치
- file_status : 파일 상태 기능 개선을 위한 브랜치
- minho : commit 버튼 기능 구현을 위한 브랜치

4. git commit history

각 브랜치별 구체적인 작업 과정 및 전체 commit history는 아래 주소에 접속하여 확인할 수 있다.

<https://github.com/advicewook/FileBrowser/commits/master>