

모바일 레포트 1주차

GIT과 GITHUB와 vs code & github

이름: 김강현

학번: 202032291

1. GIT 정의

GIT이란, 소스코드를 효율적으로 관리하기 위해 만들어진 "분산형 버전 관리 시스템"입니다. 것은 2005년에 리눅스 베네딕트 토르발스의 의해 개발된 시스템입니다.

2. GIT을 쓰는 이유

- ① 소스코드의 변경 이력 확인이 쉽습니다.
- ② 특정 시점에 저장된 버전과 비교하거나 특정시점으로 되돌아갈 수 있습니다.

3. GIT 관련 프로그램

1) GITHub Desktop

간단 설명

GITHub를 서비스 하고 코드 편집기 A노미를 개발하는 GITHub에서 만든 어플리케이션입니다. 특이사항으로는 모 회사가 MICROSOFT입니다.

장점

- 예쁘며, 빠릅니다, 멈춘다는 느낌이 없습니다.
- 커밋정보 - 변경된 정보 확인이 편합니다.
- 오픈 소스 - 자유롭게 커스텀하고, 빌드 할 수 있습니다.

단점

- 한글이 지원되지 않습니다.
- merge, branch의 시각화가 안됩니다.

2) GIT EXTENSIONS

간단 설명

정확히는 GIT의 확장 기능들이 포함된 소프트웨어입니다.

2008년부터 역사가 시작되었고, 개발자 친화적 소프트웨어입니다.

장점

시각화 - merge, branch 등 시각화가 아주 잘되어 있습니다.

2) GIT Extensions

장점

- 오픈소스 - 자유로이 커스텀하고, 빌드할 수 있습니다.
- 빠르고, 안정적 - 무려 20여년째 개발되었고, 빠르고 안정적입니다.
- 유클릭 - 어플리케이션이 아닌 GIT repository가 있는 폴더에서 유클릭시, git을 이용 할 수 있습니다.

단점

- 안 예쁨 - UI가 투박하며, 애니메이션이 없어서 버벅거린다고 느낍니다.
- 아이콘 보다는 텍스트, 직관 적이기 보다는 세부적입니다.

3) Tortoise Git

간단 설명

2015년에 출시된 거북이라고 불리는 꽤 유명한 프로그램입니다. SVN 버전 거북이도 있는데 SVN이 Git에 점유율이 낮았기며, 개발된 소프트웨어입니다. 1) CSVN: Sub Version의 약자며, 버전 관리 시스템입니다. SVN이라고 하면 거의 이 소프트웨어를 써서, Git으로 넘어오면서 이 소프트웨어를 그대로 쓰는 사람이 많습니다.

장점

- 한글을 지원하나 공식 홈페이지에서 별도로 설치 해야 합니다.

단점

- 도움말을 정독하고 사용해야 합니다.

4. Github 정의

깃허브 (Github)는 분산 버전 관리 툴인 깃 (Git)을 쓰는 프로젝트-트리를 지원하는 웹 호스팅 서비스입니다.

5. Github 용어

1. Repository: 말 그대로 파일이나 폴더를 저장해두는 저장소입니다.
2. Commit: 파일을 추가, 변경한 내용을 저장소에 저장하는 작업
3. Push: 추가, 변경한 내용을 원격 저장소에 업로드 하는 작업

5. Github에 프로젝트 올리기

5. Github에 프로젝트 올리기

1. 코드로 올리기

* `git --version` : git 설치후 버전 확인

* `git init` : 새로운 repository를 만들고 Git 프로젝트 관리할 때 이용

* `git test` : Git은 현재 레파지토리에 모든 변경 사항을 추적관리 합니다. 그 후 `git test` 라는 파일을 생성합니다.

* `git status` : 작업 디렉토리나 스테이징 영역의 상태를 표시합니다. 즉, 어떤 파일이 변경되었는지 등을 확인 할 수 있습니다.

* `git add` "파일 이름" 0

`git add .`

`git add git-test`

이 파일을 staging area에

추가 시킴

* `git commit -m "commit message"` 0 commit 명령어

`git commit -m "first commit"` 0

* `git log` : 프로젝트 내 커밋 내역을 확인 할 수 있습니다.

6. VS Code란?

비주얼 스튜디오 코드 (Visual Studio Code)는 Microsoft에서 제작한 무료 코드 편집기로 맥, 리눅스, 윈도우에서 모두 실행이 가능하기 때문에 많은 사람들에게 이용되고 있습니다.

7. Git에서 쓰는 명령어

1. `git init` : git 저장소를 생성하는 코드

2. `git status` : 현재 git 상태 나타냄

3. `git add (파일 이름)` : 특정 파일을 스테이징 영역에 추가 합니다.

4. `git add -A` : 새로운 파일, 수정한 파일, 삭제한 파일 모두 추가 합니다.

5. `git add .` : 새로운 파일, 수정한 파일, 삭제 파일

Staging Area (코드 변경 사항을 관리하고 버전 제어를 역할을 합니다)에 남아 있음

주석

Git에서 스테이징 영역은 작업 디렉토리와 실제로 커밋되기 전 중간 단계를 스테이징 영역이라 합니다.

7. Git에서 쓰는 명령어

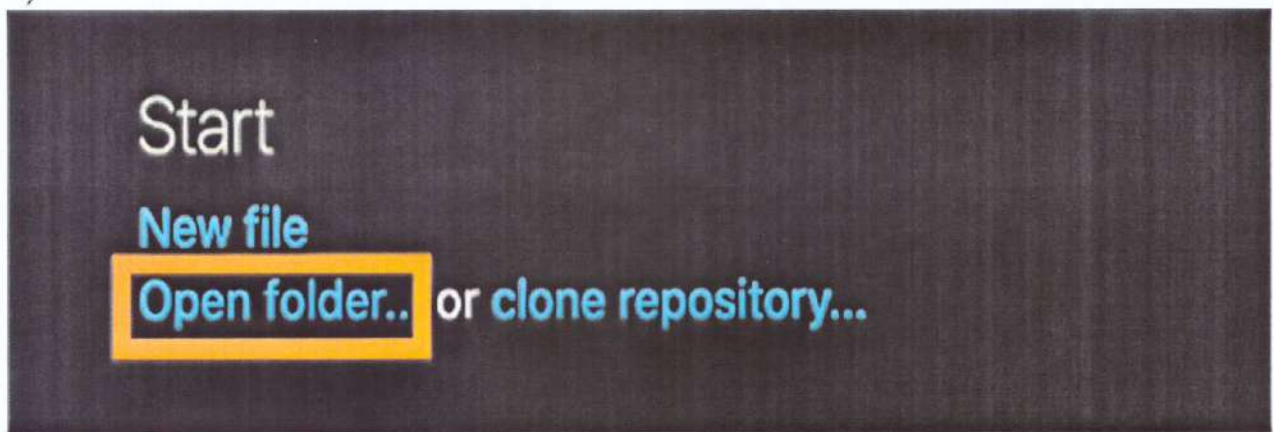
| Git에서 쓰는 명령어 | 역할 |
|------------------------------------|---|
| git add * | 새로운 파일, 수정한 파일 추가 삭제된 파일은 staging Area에서 삭제 가능 |
| git commit -m "message" | 메세지와 함께 commit |
| git commit -a | add 해주고 commit 메세지는 따로 적기 |
| git commit -am "message" | add 해주고 메세지와 함께 commit |
| git remote remove <등록된 클라우드 주소> | 클라우드 주소 삭제 |
| git branch | 브랜치 목록 (브랜치: 새기능 / 버그 수정 시 사용해서 다른 팀원의 작업과 분리한다.) |
| git branch <브랜치 이름> | 브랜치 생성 (local로 만듦) |
| git push origin <브랜치 이름> | 만든 브랜치를 원격서버로 전송 |
| git (check out -b <브랜치 이름> | 브랜치 생성 및 이동 |
| \$ git clone <https... URL> | 기존 소스코드 다운로드 복제 |
| git clone / 로컬저장소 / 경로 | 로컬 저장소 복제 |
| git push <remote> <브랜치 이름> | 커밋을 원격서버에 업로드 |

8. VS Code 단축키

| | |
|------------------|---------------------|
| Ctrl + P | 파일 열기 |
| Ctrl + W | 현재 열린 파일 닫기 |
| Ctrl + S | 파일 저장 |
| Ctrl + D | 중복사 |
| Ctrl + L | 라인 전체 선택 |
| Ctrl + Tab | 태비딩 |
| Ctrl + Q | 이전에 사용한 파일 열기 |
| Ctrl + Shift + K | 라인 삭제 |
| Ctrl + I ~ J | 열린 파일 숫자키로 이동 |
| Ctrl + Shift + P | Command Palette를 오픈 |

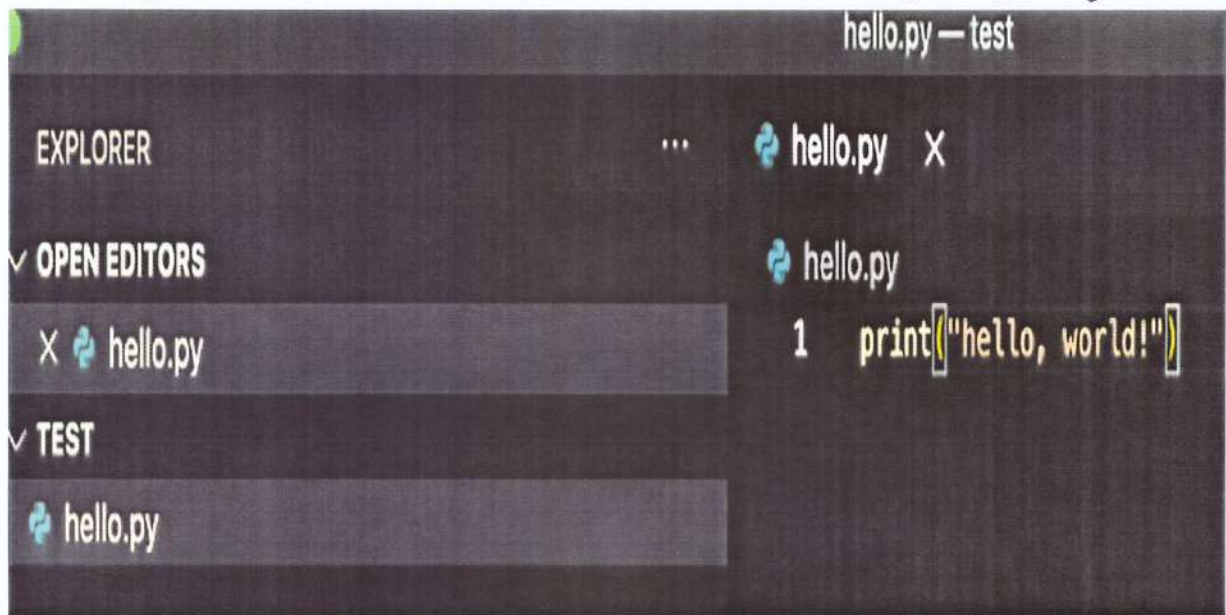
1. VS code 열기 > VS code 로 github 에 파일 업로드 하기

VS code 를 켜고 open folder 를 클릭합니다.

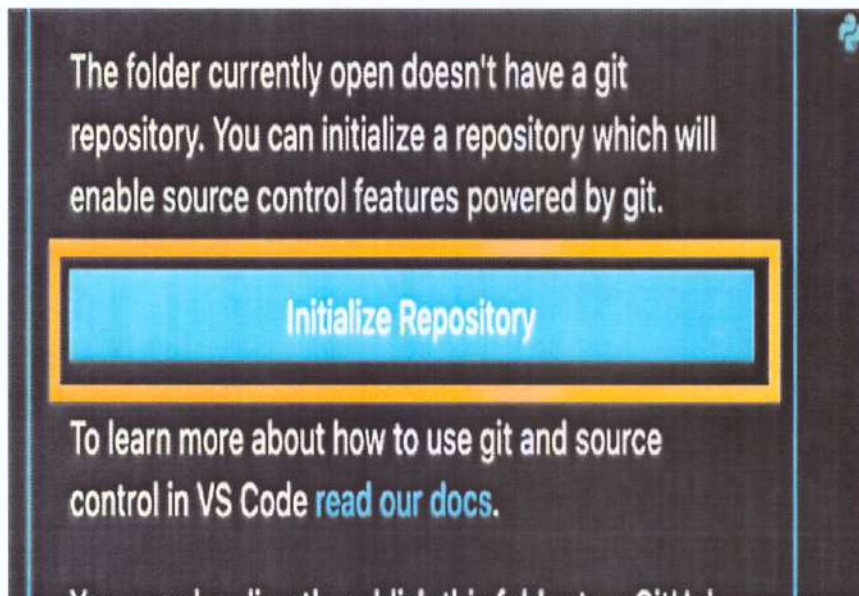


↳ 저는 test 라는 폴더를 선택했습니다.
github 에 올리고자 하는 파일이 속한 폴더를 열어줍니다.

→ 파일 추가를 해서, hello.py 파일을 만들어 볼게요



→ 브랜치 아이콘을 클릭 하세요



→ initial Repository를 클릭하면, 로컬 Repository가 초기화 됩니다.

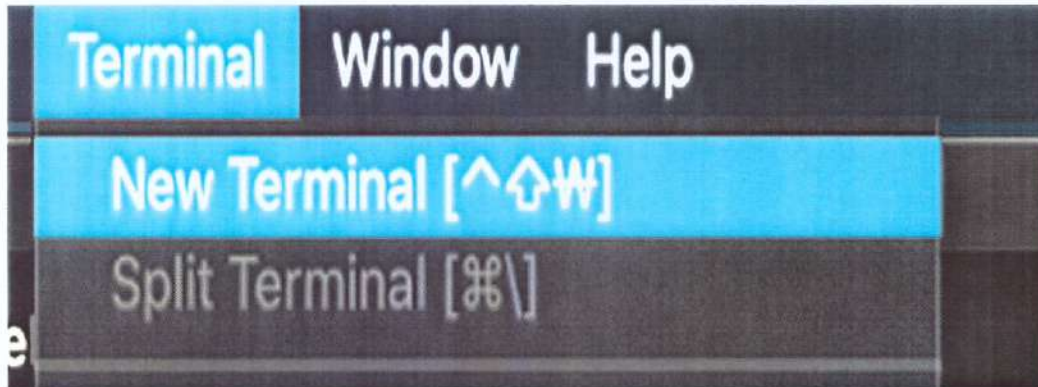
*로컬 Repository가 초기화 된다는 말은 '이제서야 이 폴더에 있는 파일을 git 명령어로 이쁘게 관리한다'고

1) Remote setting

- remote setting은 작업중인 파일을 github의 어떤 repository에 업로드 하는 과정입니다.

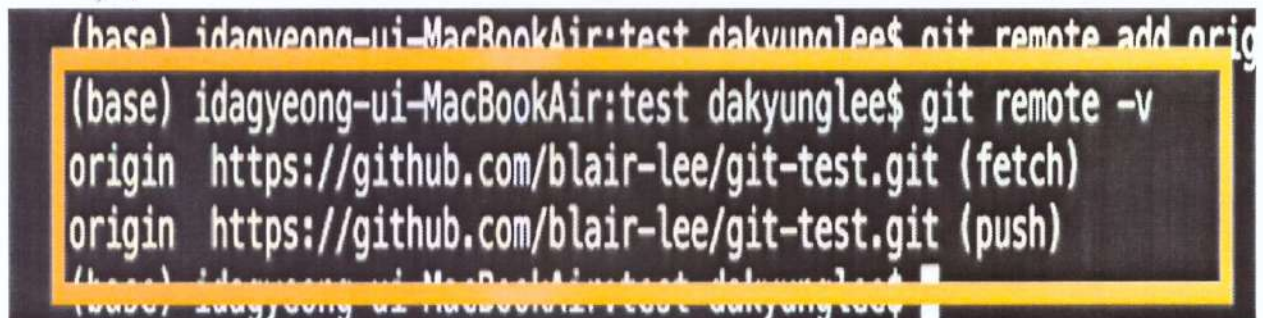


→ Terminal의 New Terminal을 클릭합니다.



→ 새 Terminal 창을 여는 것입니다.

→ Command line에 git remote -v를 입력합니다.



→ github의 repository 주소를 확인 할 수 있습니다.



→ 2) Commit 대상 선택하기

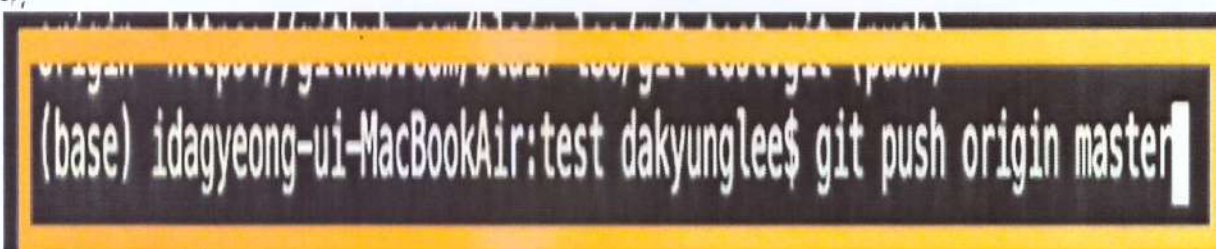
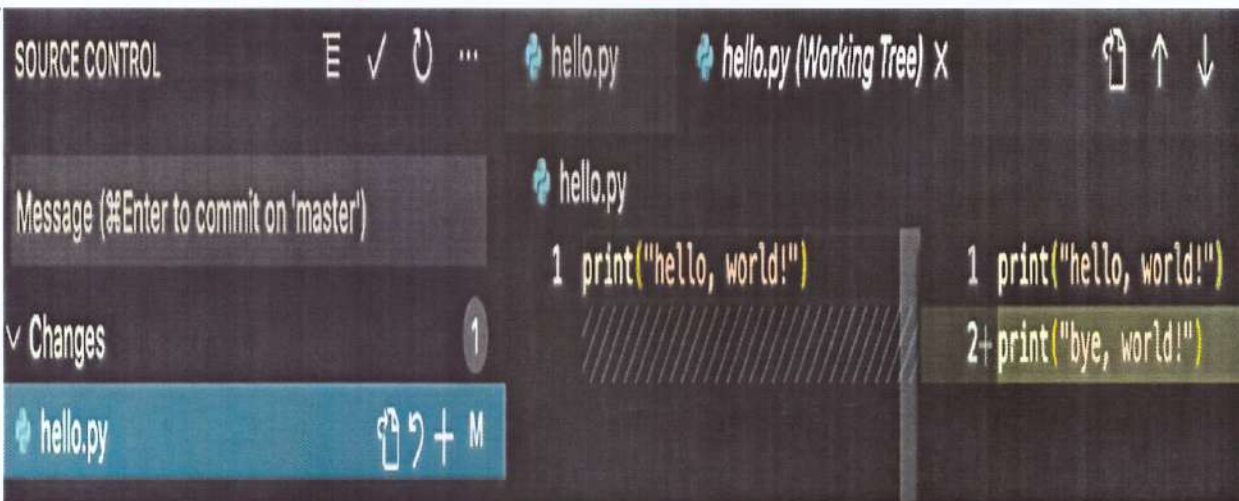
3) Commit



* Commit은 로컬에서 발생한 변경 사항을 기록하고 저장합니다.

↳ √를 누르면 commit이됩니다.

hello.py
파일을
수정하
거나,
새로운
파일을
추가하면
다시
나타납니다.



↳ command line에 git push origin master
이렇게 합니다.



4) 결과 확인

blair-lee first commit

9f028b8 23 minutes ago 1 commit

hello.py

first commit

2) 레퍼지토리 생성하기



New repository

Import repository

New gist




New organization

New project

→ 새로운 레퍼지토리를 생성합니다.


1) Repository 이름과 Description 을 입력합니다.


Owner * Repository name *

  / 

Great repository names are short and memorable. Need inspiration? How about [studious-tribble?](#)

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.


☐  **Private**
Only those who you've given access to can see and commit to this repository.

Initialize this repository with:

☒ Add a README file
[Learn more](#)

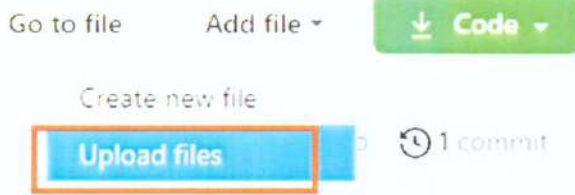
☐ Add .gitignore
Choose which files not to track from a list of templates. [Learn more](#)

☐ Choose a license
Tell the world what they can and can't do with your code. [Learn more](#)

To, will get  [fork](#) as the default branch. Change the default name in your [settings](#).

↳ Add a README file 을 체크하고 Create repository 를 클릭해서 repository 를 생성합니다.

2) Add file > Upload files 를 클릭 합니다.



→ 더 채로 올리지 않고 파일만 올립니다.

Drag files here to add them to your repository

Or [choose your files](#)

3) 업로드 후 아래쪽 commit changes에
Commit changes
설명을 적어줍니다.

[Add files via upload](#)

[Add an optional extended description...](#)

[Settings](#)

4) commit 완료 되면 위쪽에 settings 를
클릭합니다.

5) 아래 내려오면 GitHub Pages가 있습니다.
GitHub Pages (Check it out here)를 클릭합니다.

Pages settings now has its own dedicated tab! [Check it out here](#)

6) source에 select branch를 main으로
변경하고 save를
클릭합니다.

Source
GitHub Pages is currently disabled. Select a source below to enable

Branch: main ▾

/ (root) ▾

Save

Select branch

Select a branch

name using the gh-

✓ main

None

9. 줄서

1. GIT 정의 부분 참조 자료 > [Git] Git이란?

-정의 및 용어 정리 - Tstaya

2. GIT을 쓰는 이유 부분 참조 > [Git] Git이란? 정의 및

-용어 정리 - Tstaya

3. GIT 관련 프로그램 참조 > GUI Git 클라이언트 종류와

-장단점 - Tstaya

4. 'GitHub 정의'와 '5. Git hub 용어' 부분 참조

↳ [GitHub] GitHub란? (간단정리)

5. GitHub에 프로젝트 올리기 작성시 참고 자료

→ [Git] Git Hub 레파지토리 (Repository)
생성하고 소스 올리기

6. 'VS Code란?' 작성시 참고한 자료 > VS Code란?

7. Git에서 쓰는 명령어 작성시 참고한 자료

→ [Git Hub] Git 명령어 모음

8. VS Code 단축키 작성시 참고 자료

→ [Visual Studio Code] 기본 ~ 유용한 단축키 모음

1. VS Code 열기 [VS Code]로 GitHub에 파일
업로드 하기 작성시 참고 한 자료

→ VS Code에서 Git hub 업로드 하는 방법 (광수님
ㅋㅋ)

2. 레퍼지토리 생성하기

→ 깃허브. 웹 페이지 호스팅 만들기