

SILO Lab

GitHub 개발환경 가이드

| 기술개발팀 김시현

INDEX

01 GitHub란?

- Git, GitHub란
- 사전지식

02 GitHub 사용 기초

- Git설치
- GitHub 계정생성
- 레퍼지토리 생성
- 파일 업로드

03 Visual Studio Code GIT협업

- Vscode설치
- 기초셋팅
- 버전관리
- Vscode GIT협업하기

04 코드스페이스

05 GitHub실습과제

01. GitHub란?



01. GitHub란?

1. Git, GitHub란

2. 사전지식

깃(Git)이란

컴퓨터파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 분산버전관리 시스템.



즉 모든 파일의 변화를 기록하는 것

CLI(Command Line Interface)기반

* 깃에서 버전이란?

흔히 알고 있듯이 문서를 수정하고 저장할 때마다 생기는 것이라고 생각하면 쉽다. 깃과 같은 버전 관리 시스템을 이용하여 버전을 관리하면 원래 파일 이름은 그대로 유지하면서 파일에서 무엇을 변경했는지를 변경 시점마다 저장할 수 있다.

01. GitHub란?

1. Git, GitHub란

2. 사전지식

GitHub란



분산 버전 관리 툴인 깃(Git) 저장소 호스팅을 지원하는 웹 서비스

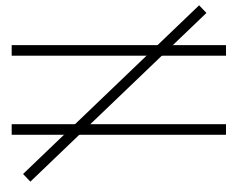
즉, 인터넷에서 코드를 업로드 할 수 있는 사이트

01. GitHub란?

1. Git, GitHub란

2. 사전지식

깃과 깃허브는 다르다!!!



01. GitHub란?

1. Git, GitHub란

2. 사전지식



업로드



업로드



01. GitHub란?

1. Git, GitHub란

2. 사전지식

“ 개발자에게는 없어서는 안될 서비스!”

GitHub의 장점

- 소스 코드를 주고받을 필요 없이 같은 파일을 여러 명이 동시에 작업하는 병렬개발이 가능
- 프로젝트관리, 배포, 이슈 추적 기능 등 프로젝트 진행에 있어 편리한 기능들을 지원
- 히스토리를 볼 수 있음.
- 빠르다

GitHub의 필요성

- 비대면 작업이 활성화되는 요즘 온라인 협업도구의 필요성 증가
- 깃허브에 남겨진 결과물들이 포트폴리오 역할
- 깃허브에는 많은 양의 오픈소스 프로젝트들이 있어 프로그래밍 하는 사람들이 깃허브에서 많은 정보를 얻고 공부할수 있음.

01. GitHub란?

1. Git, GitHub란

2. 사전지식

사전지식 1 : 가장 중요한 두가지

- 커밋(commit) : 파일을 추가하거나 변경 내용을 저장소에 저장하는 작업
- 푸시(push) : 파일을 추가하거나 변경 내용을 원격 저장소에 업로드하는 작업

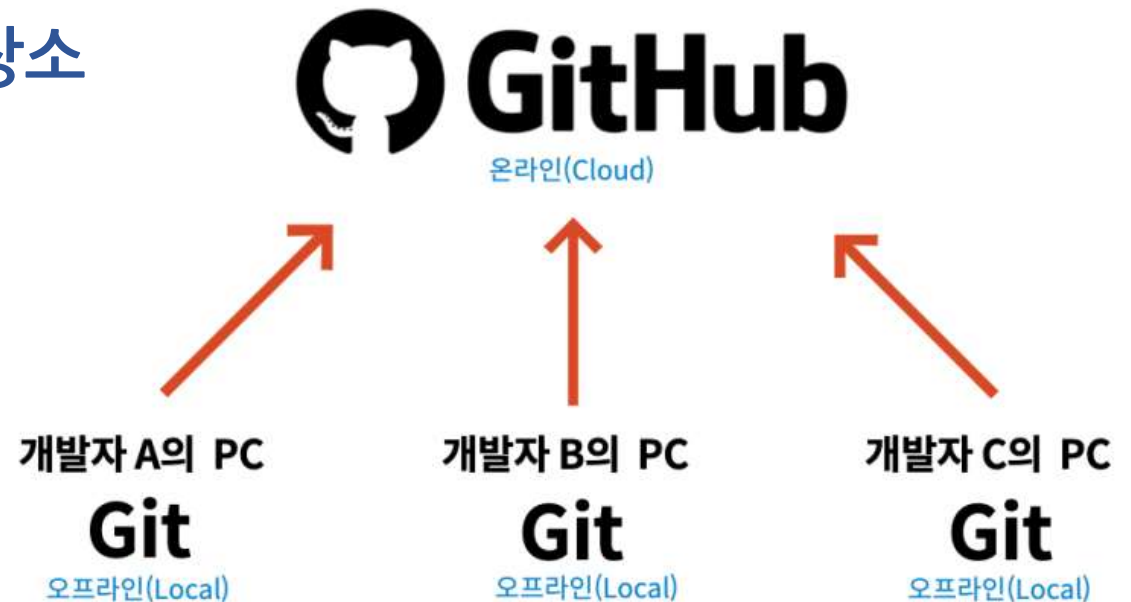
01. GitHub란?

1. Git, GitHub란

2. 사전지식

사전지식 2 : 로컬 저장소와 원격 저장소

- 로컬저장소 : 자신의 컴퓨터에 있는 저장소
- 원격저장소 : 서버 등 네트워크에 있는 저장소



01. GitHub란?

1. Git, GitHub란

2. 사전지식

사전지식 3 : 브랜치(branch)

소프트웨어 개발은 현재 출시하고 있는 버전의 유지 보수를 하면서 새로운 기능 추가 및 버그 수정을 할 수 있다. 이러한 병렬로 수행되는 여러 버전 관리를 위해 GitHub에는 브랜치(branch)라는 기능이 있다.

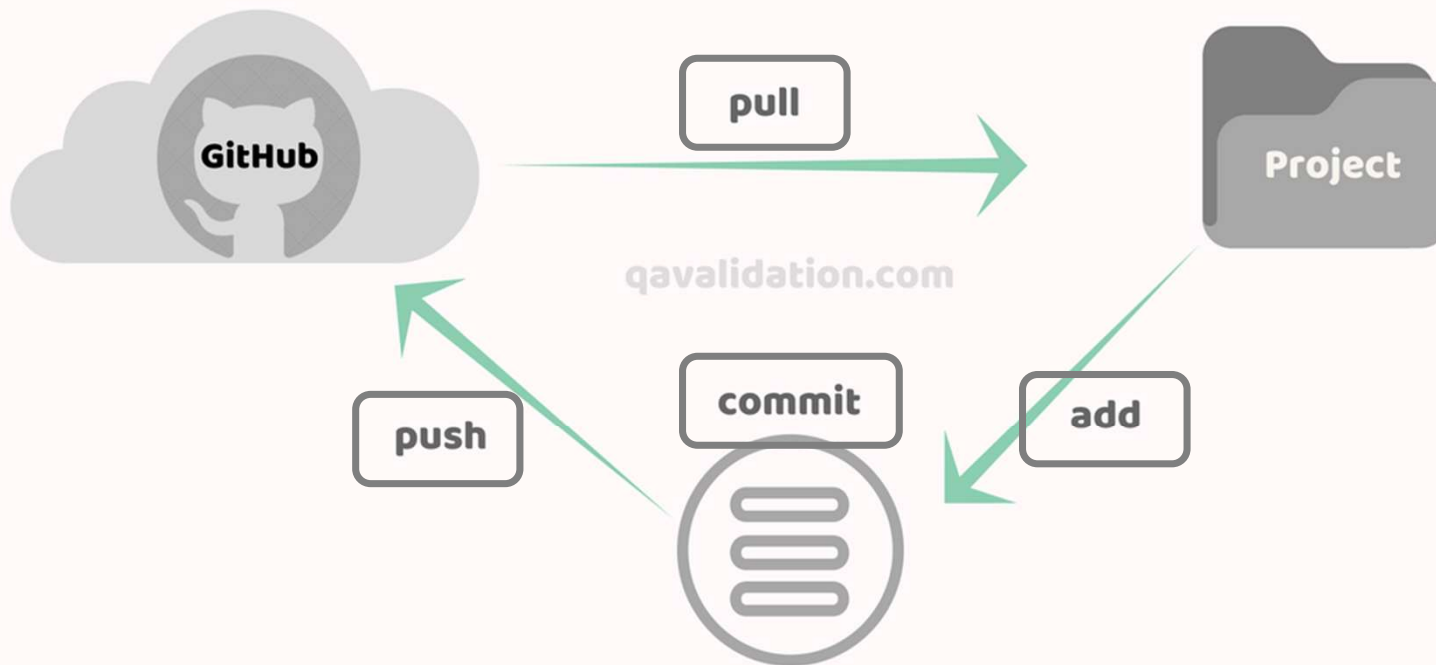
브랜치(branch)의 지점은 역사의 흐름을 분기하여 기록해 나가는 것이다. 분기 한 지점은 다른 지점의 영향을 받지 않기 때문에 같은 저장소에서 각 개발을 해 나갈 수 있다.

01. GitHub란?

1. Git, GitHub란

2. 사전지식

Git PUSH PULL



다운받자! pull

변경했다! add

확정했다! commit

업로드하자! push

여기까지 한 것

개념 익히기

02. GitHub사용 기초

02. GitHub사용 기초

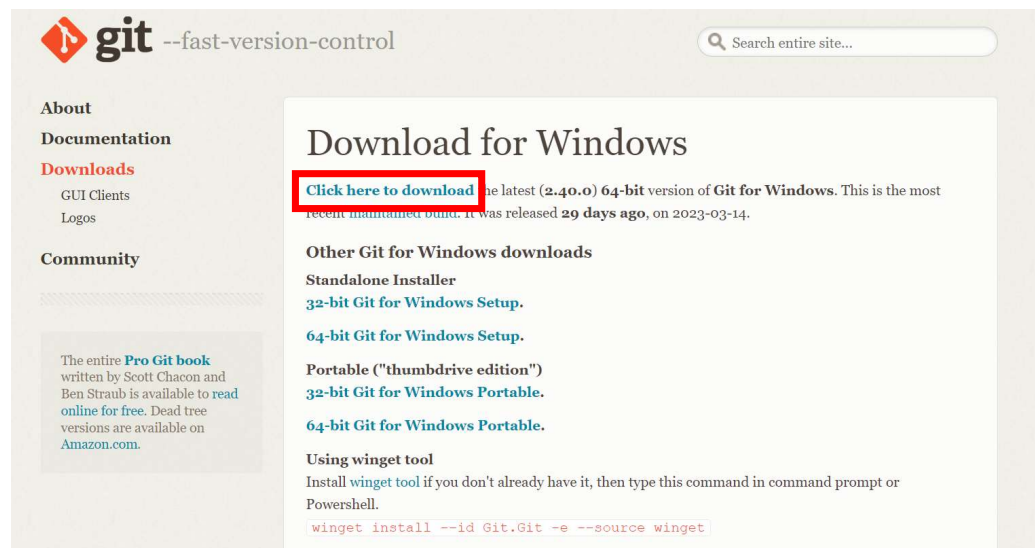
1. Git설치

2. Github계정생성

3. 레퍼지토리생성

4. 파일업로드

① 사이트 접속 <https://git-scm.com/download/win>



운영체제에 맞게 선택하여 다운로드

02. GitHub사용 기초

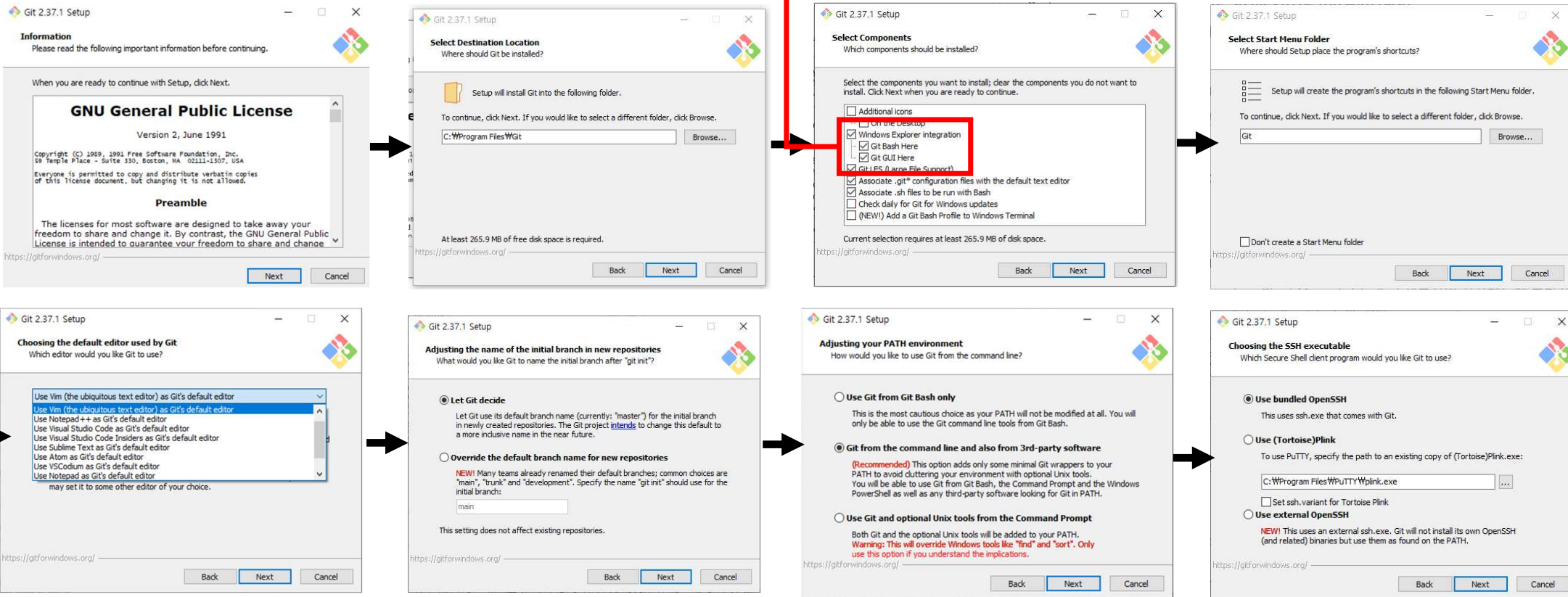
1. Git설치

2. Github계정생성

3. 레퍼지토리생성

4. 파일업로드

② 설치진행



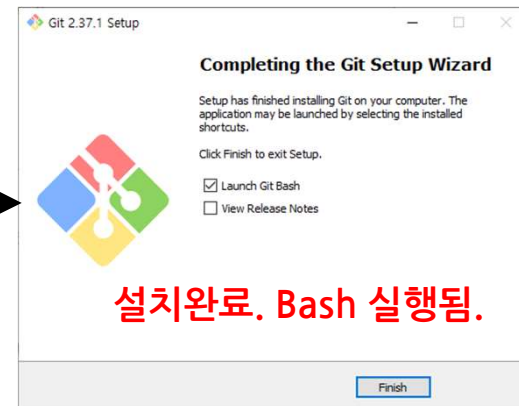
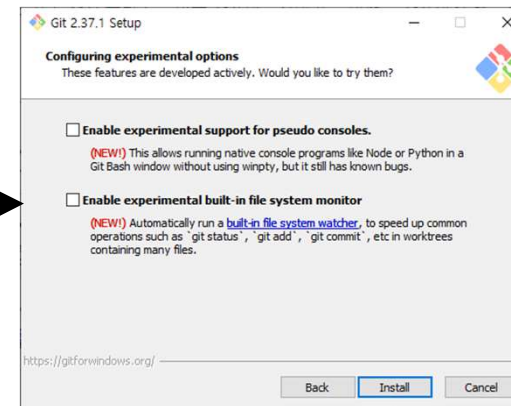
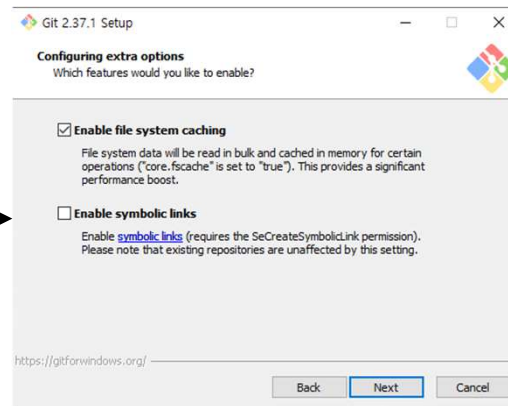
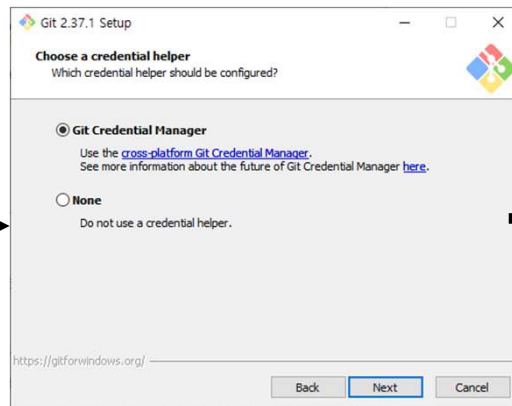
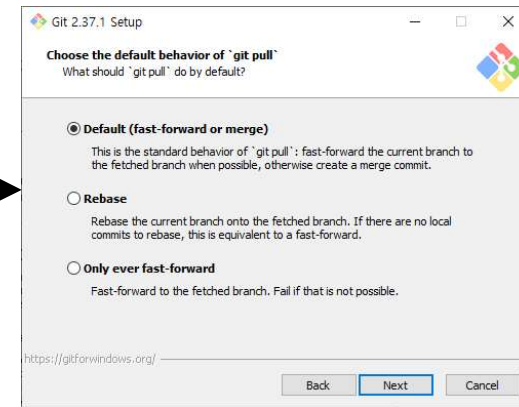
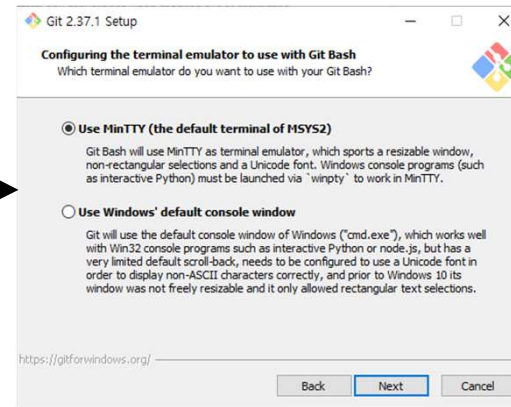
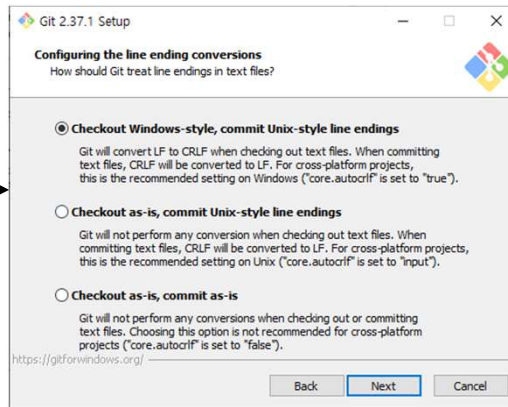
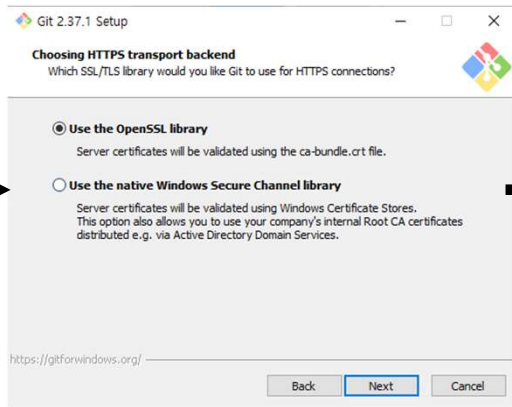
02. GitHub사용 기초

1. Git설치

2. Github계정생성

3. 레퍼지토리생성

4. 파일업로드



설치완료. Bash 실행됨.

02. GitHub사용법

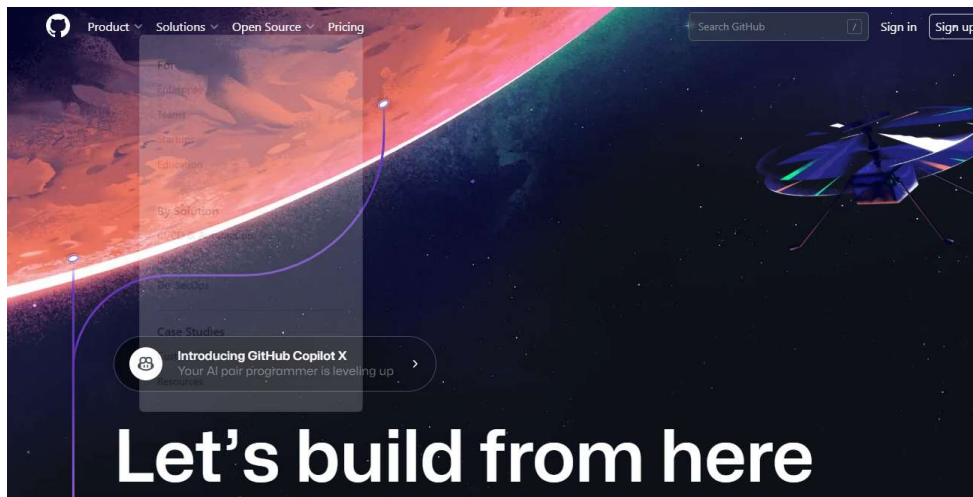
1. Git설치

2. Github계정생성

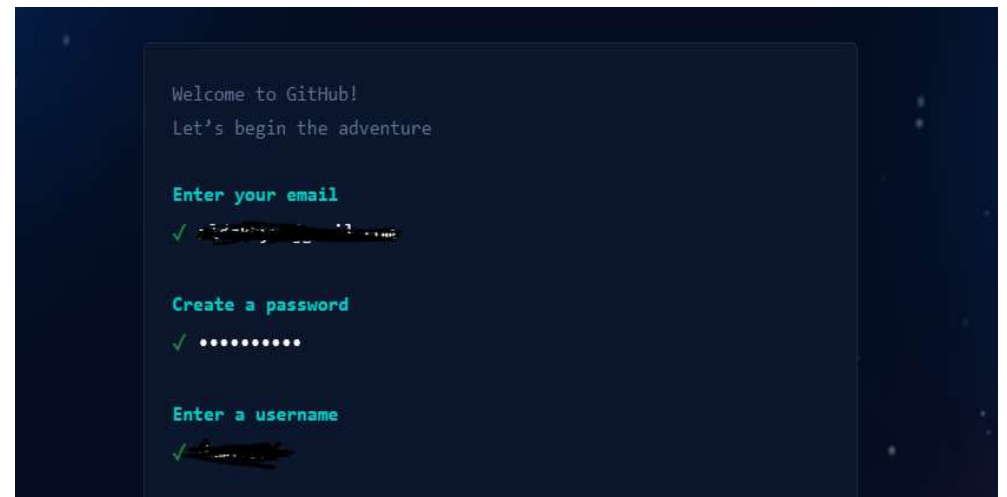
3. 레퍼지토리생성

4. 파일업로드

① 사이트 접속 <https://github.com/>



② 회원가입(sign up)



02. GitHub사용법

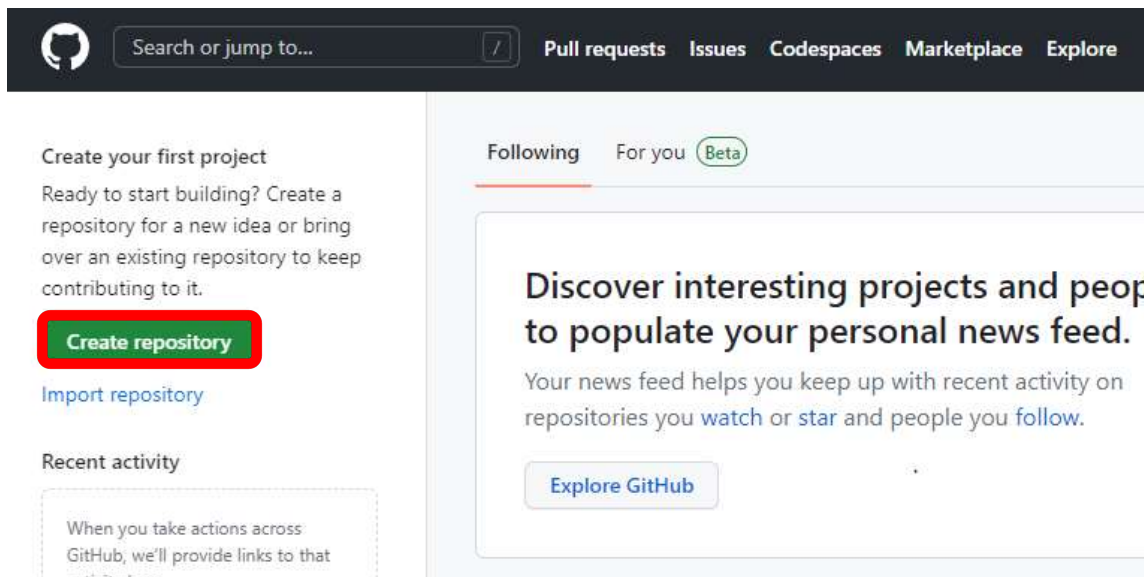
1. Git설치

2. Github계정생성

3. 레퍼지토리생성

4. 파일업로드

Create repository



repository(저장소)

깃허브에 파일을 업로드하려면
repository 만드는 게 우선.
원하는 대로 만들 수 있음.

02. GitHub사용법

1. Git설치

2. Github계정생성

3. 레퍼지토리생성

4. 파일업로드

❖ Repository 생성시 참고

- ① **Repository name** : 프로젝트 이름을 적어준다. 해당이름으로 저장소의 URL이 결정된다.
- ② **Description** : 프로젝트에 대한 간단한 설명을 덧붙인다. 안해도 됨.
- ③ **Public/Private** : 저장소의 공개여부이다. Private면 비공개로 저장되지만 4인이상시 유료이다. Public 하기
- ④ **Add a README file**: 프로젝트를 소개하는 문서이다. 안해도 됨.
- ⑤ **ADD.gitignore**: github에 업로드하지 않을 문서를 지정하는 파일이다. 이 파일은 주로 IDE에서 자동으로 생성해주므로, 안해도 됨.
- ⑥ **Choose a license**: 저작권 및 오픈소스 관련 라이선스를 설정할 수 있다. 개인용이면 굳이 안해도 된다,

Create repository 누르면 생성 완료!!

02. GitHub사용법

1. Git설치

2. Github계정생성

3. 레퍼지토리생성

4. 파일업로드

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

HamToriS / SILO_test1 (Public) Pin Unwatch 1

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or **HTTPS** SSH `https://github.com/HamToriS/SILO_test1.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# SILO_test1" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/HamToriS/SILO_test1.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/HamToriS/SILO_test1.git
git branch -M main
git push -u origin main
```

이것이 깃허브 레퍼지토리의 주소이다! 클립보드에 저장해놓기.

02. GitHub사용법

1. Git설치

2. Github계정생성

3. 레퍼지토리생성

4. 파일업로드

◆ Github에서 repository 에 파일 올리기

EX)

• 파일이름입력

Ham_first / SILO in main

<> Edit new file

Preview

- ‘파일명’ + ‘/’: 폴더로 생성됨. (‘/’ 지우면 복구.)
- 최종입력 칸은 ‘파일명’ + ‘.md’
- md파일에 설명 적기
- 아래 **Commit changes** 버튼 누르면 완성

Ham_first / SILO / sihyun.md in main

<> Edit file

Preview

1 2023년, 예시로 작성해봄.
2

여기까지 한 것

[기본셋팅]

GIT설치, GitHub원격저장소생성

03. Visual Studio Code GIT협업

이제 할 것

[협업을 위한 기본설정/ 협업방법]

Vscode설정, pull, commit, push

03. Visual Studio Code GIT협업

1. Vscod설치

2. 기초셋팅

3. 버전관리

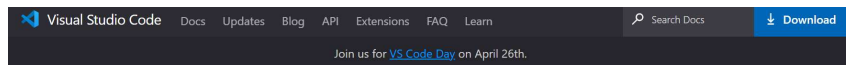
4. VScode GIT협업하기



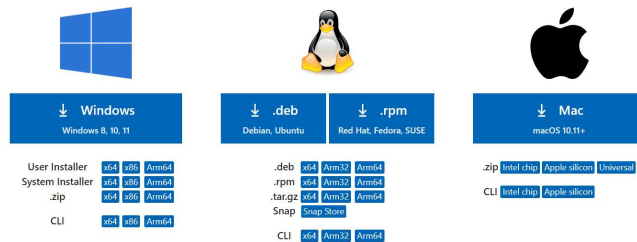
VScode란

마이크로소프트가 Window, macOS, 리눅스용으로 개발한 소스 코드 편집기이다.
거의 모든 언어가 지원된다.

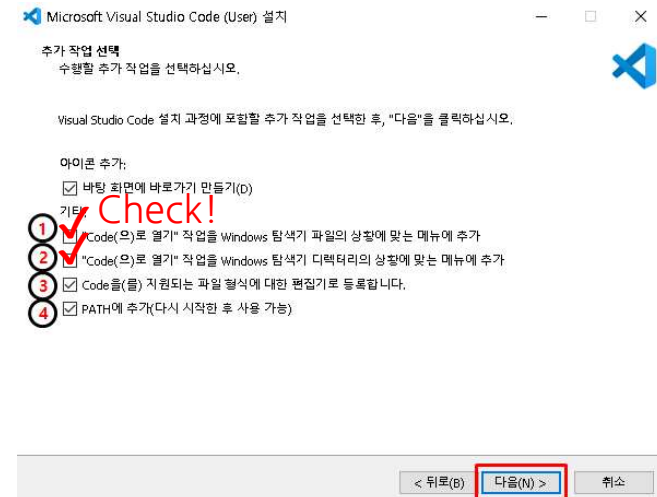
① 사이트 접속 <https://code.visualstudio.com/download>



Download Visual Studio Code
Free and built on open source. Integrated Git, debugging and extensions.



② 설치 시 주의



03. Visual Studio Code GIT협업

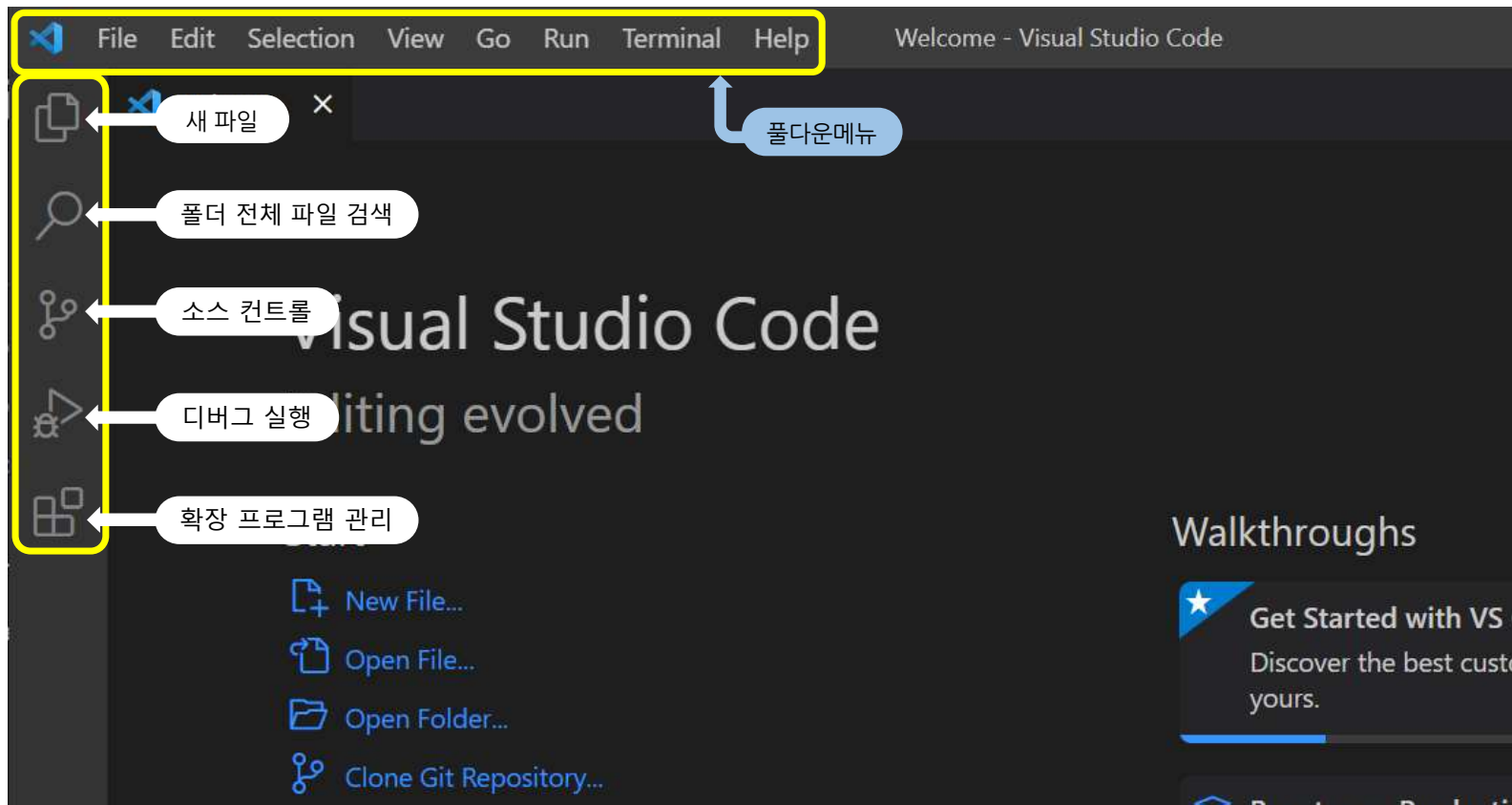
1. Vscode설치

2. 기초설정

3. 버전관리

4. VScode GIT협업하기

◆Vscode 화면 인터페이스구성



*한국어로설정하는법

Ctrl + Shift + p 를 눌러 명령 팔레트를 표시합니다. display를 입력한 후 표시된 명령 목록 중 Configure Display Language 를 선택합니다.

03. Visual Studio Code GIT협업

1. Vscode설치

2. 기초셋팅

3. 버전관리

4. VScode GIT협업하기

◆ Git : User name & email 설정하기



Git Bash
실행

```
silolab_ksh@SILO-KSH MINGW64 ~
$ git config --global user.name remote
silolab_ksh@SILO-KSH MINGW64 ~
$ git config --global user.email zldzkhyun@gmail.com
```

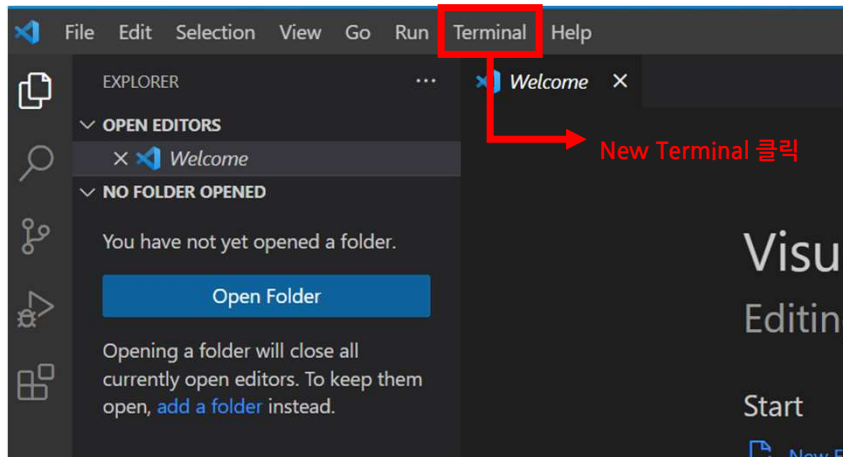
이름:

이름은 아무거나 해도 상관없다.

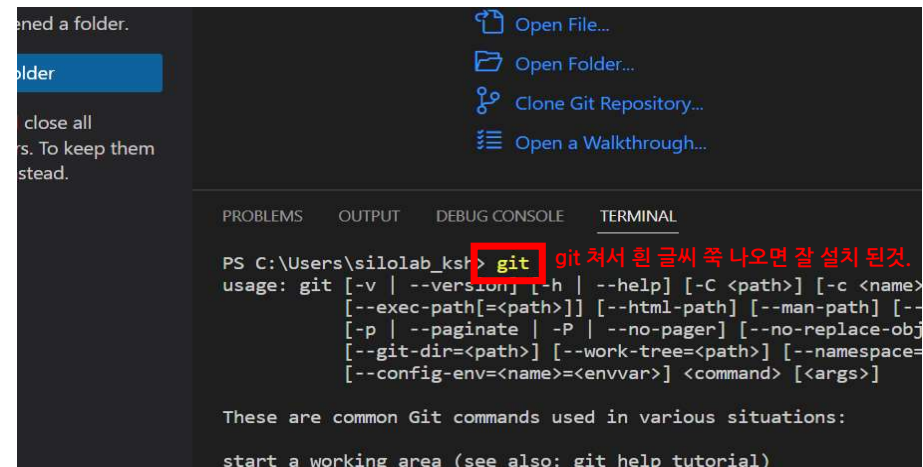
★깃허브 이메일적기:

이메일로 사용자 인증을 진행하기 때문에 입력을 잘해야 함.

◆ Vscode접속해서 git 설치 잘 됐는지 확인하기



New Terminal 클릭



4. VScode GIT협업하기



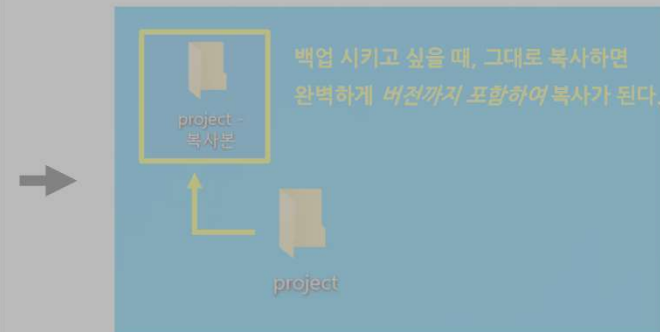
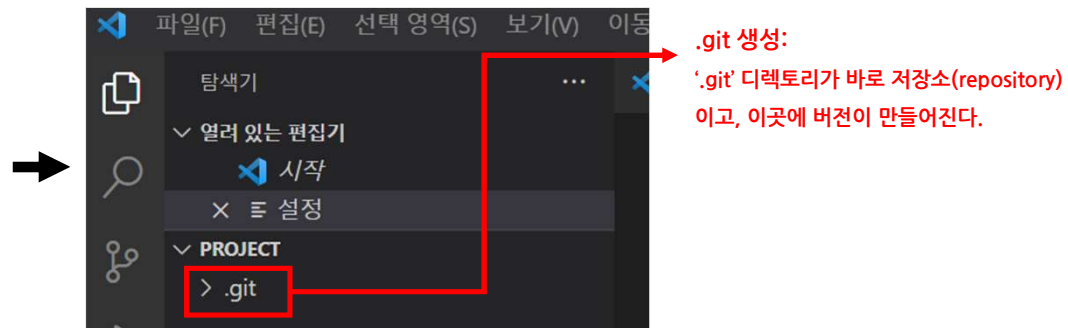
03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

3. 버전관리

4. VScode GIT협업하기



03. Visual Studio Code GIT협업

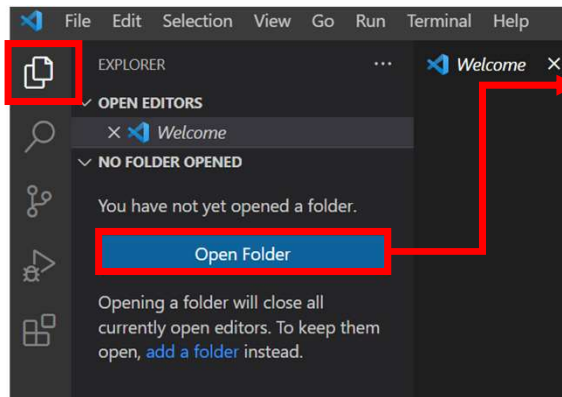
1. VScode란

2. 기초셋팅

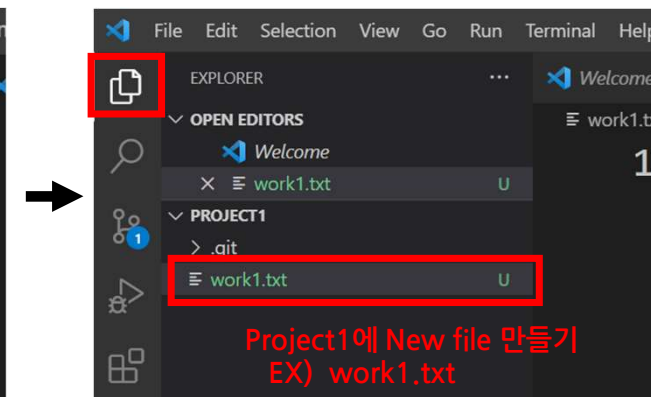
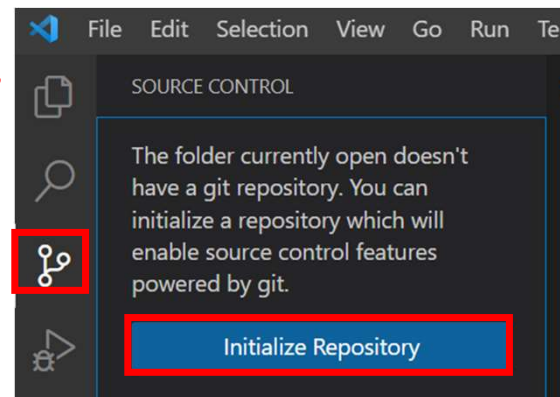
3. 버전관리

4. Vscode GIT협업하기

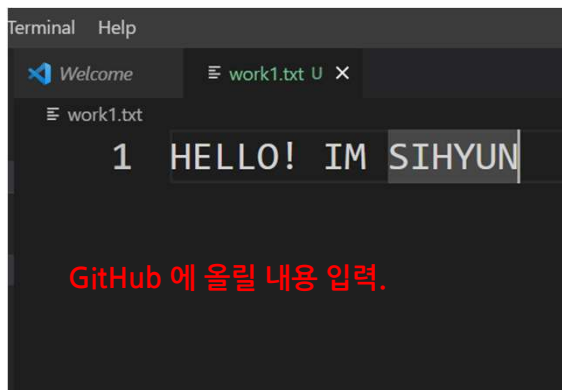
◆ Vscode에서 Github 레퍼지토리로 push 하기



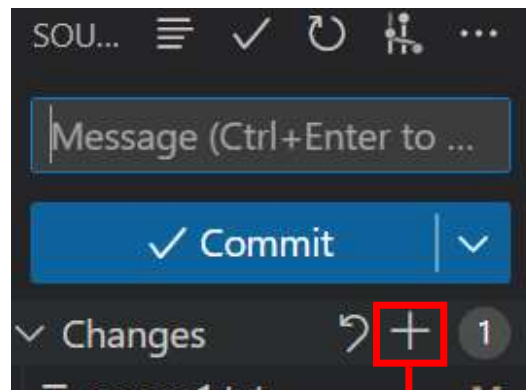
깃허브에 올릴 폴더 선택:
EX) 바탕화면에 'project1'
폴더 만들어서 폴더선택.



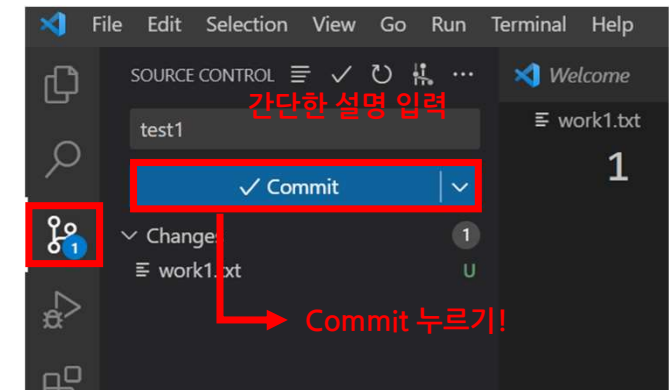
Project1에 New file 만들기
EX) work1.txt



GitHub 에 올릴 내용 입력.



Commit전 준비단계:add



Commit 누르기!

여기까지 한 것

[Vscode에서 파일만들어서 commit함]

*Commit이란?

Push하기전 변경사항 확정하고, 저장해 두는 것.

03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

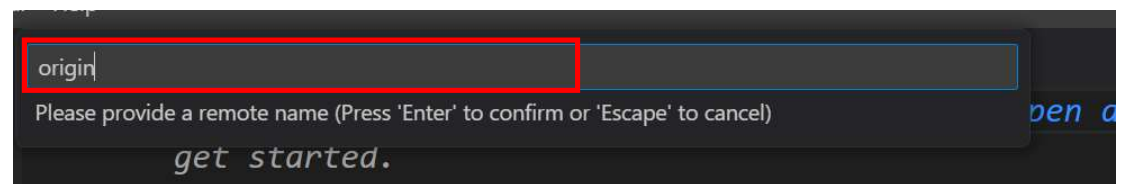
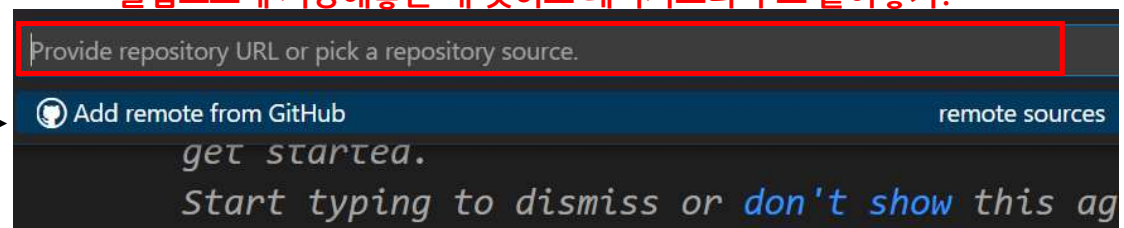
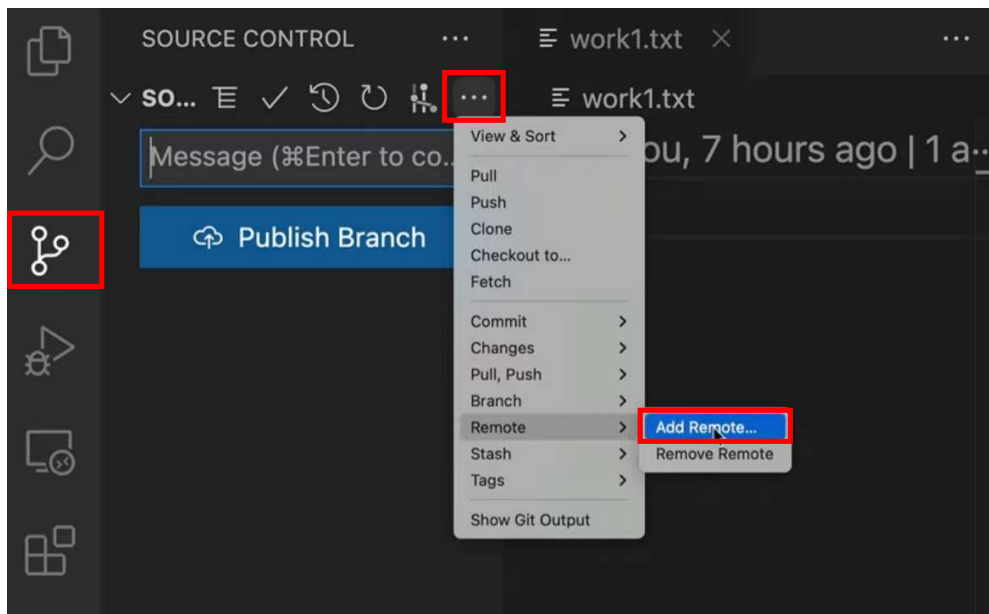
3. 버전관리

4. Vscode GIT협업하기

◆ 원격저장소 설정하기(Add remote)

Why? 그래야 내 컴퓨터에서 수정한 내용 깃허브 원격저장소로 push 할수있음.

클립보드에 저장해놓은 내 깃허브 레퍼지토리 주소 붙여넣기!



원격저장소의 별명을 지정해야함. origin으로 적어라.

여기서부터 'origin'이란 방금 추가한 주소의 저장소를 나타내는것!

03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

3. 버전관리

4. Vscode GIT협업하기

◆ Git Graph설치



설치가 끝나면 Vscode 좌측 하단 파란바에 Git Graph가 있고 누르면 나온다.

03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

3. 버전관리

4. Vscode GIT협업하기

◆ log확인(1) ... Why? Commit의 히스토리 확인하기

터미널창 ctrl+shift+`

```
PS C:\Users\silolab_ksh\Desktop\TES> git log --oneline --all --graph
* a9138fe (HEAD -> master) test1
```

이건 commit message로 적었던거고,
버전이라고 한다.

HEAD가 master 를 가리키고있고, master는 test1을 가리키고 있다.

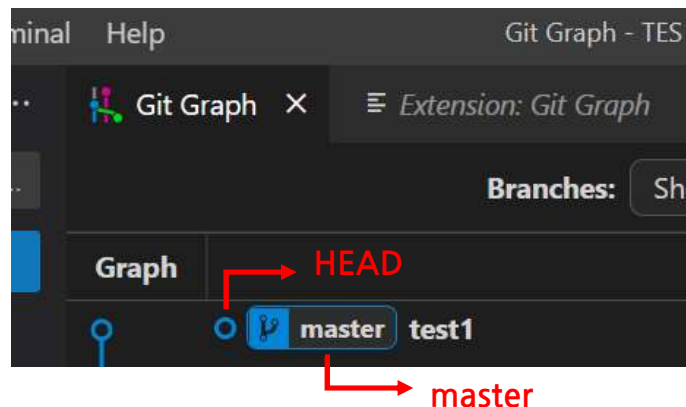
지역저장소(내컴퓨터)의 최신 버전이 test1이라는 뜻이다.

터미널창에

git log --oneline --all --graph 입력

(log를 볼건데 한줄로 모두 그래프형식으로 보겠다는뜻)

Git graph로 보는법



HEAD와 master가 같은 파란색으로 돼있으면 HEAD가 master를 가리키고 있다는 의미.

03. Visual Studio Code GIT협업

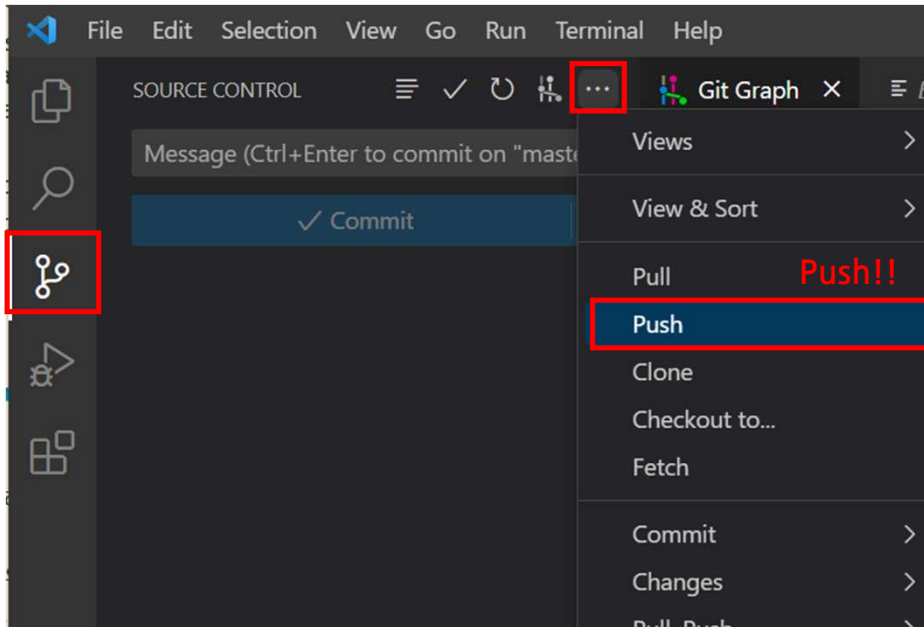
1. VScode란

2. 기초셋팅

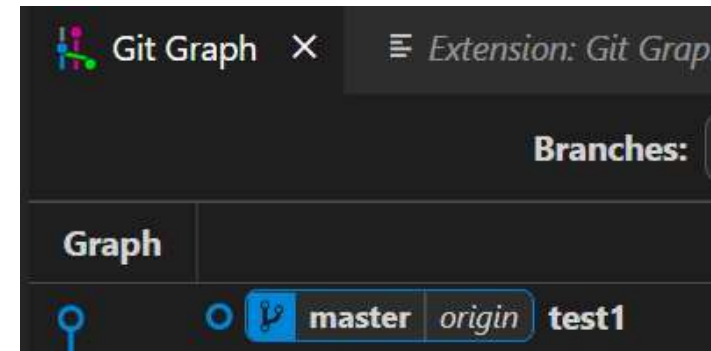
3. 버전관리

4. Vscode GIT협업하기

◆ 원격저장소로 push하기



Git Graph



Git Graph를 보면 변경되어 있다.

Origin 이라는 master가 생긴것.

지역저장소의 버전을 원격저장소로 push하기 성공

03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

3. 버전관리

4. Vscode GIT협업하기

터미널로확인

```
PS C:\Users\silolab_ksh\Desktop\TES> git log --oneline --all --graph
* a9138fe (HEAD -> master, origin/master) test1
```

원격저장소의 master 최신 버전을 가리킴!

지역저장소의 master 최신 버전을 가리킴!

지역저장소의 최신버전과, 원격저장소의 최신버전이 둘 다 test1을 가리키고 있다.

즉, 동기화가 끝났다는 뜻.

Push 성공!!

03. Visual Studio Code GIT협업

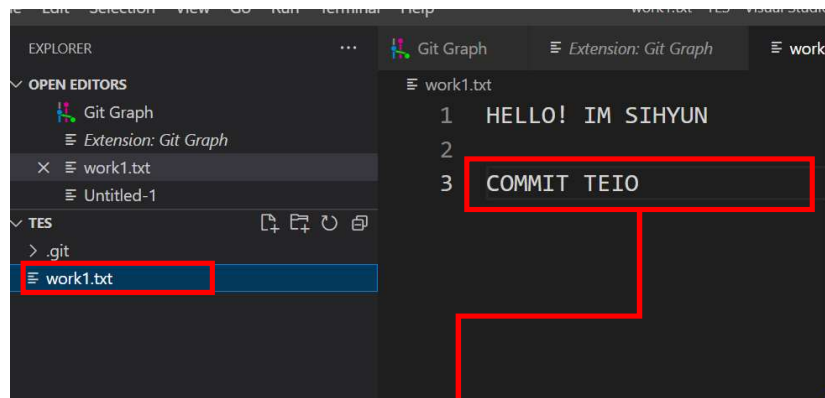
1. VScode란

2. 기초셋팅

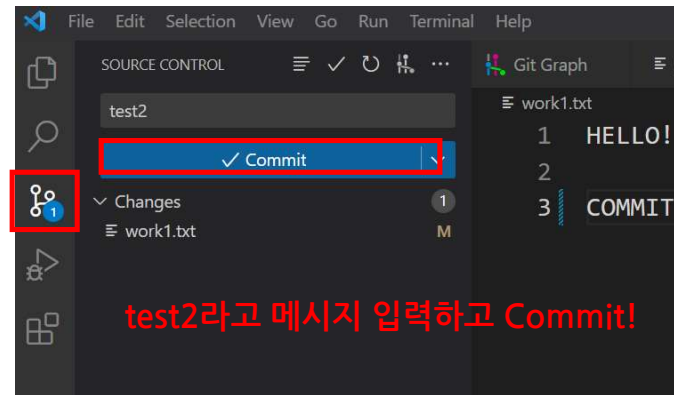
3. 버전관리

4. Vscode GIT협업하기

◆ log확인(2)



Work1.txt파일에 아무 말이나 추가해보자.

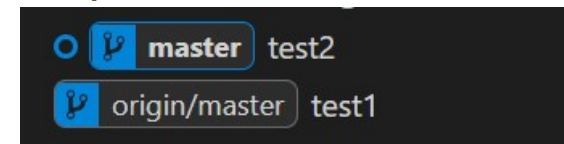


test2라고 메시지 입력하고 Commit!

터미널창

```
PS C:\Users\silolab_ksh\Desktop\TES> git log --oneline --all --graph
* 47e7ae0 (HEAD -> master) test2
* a9138fe (origin/master) test1
```

Git Graph



원격저장소의 최신버전은 test1 이고, 지역저장소의 최신버전은 test2이다.

Test1은 원격저장소로 올라갔지만, test2는 아직 원격저장소로 안 올라갔다!

03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

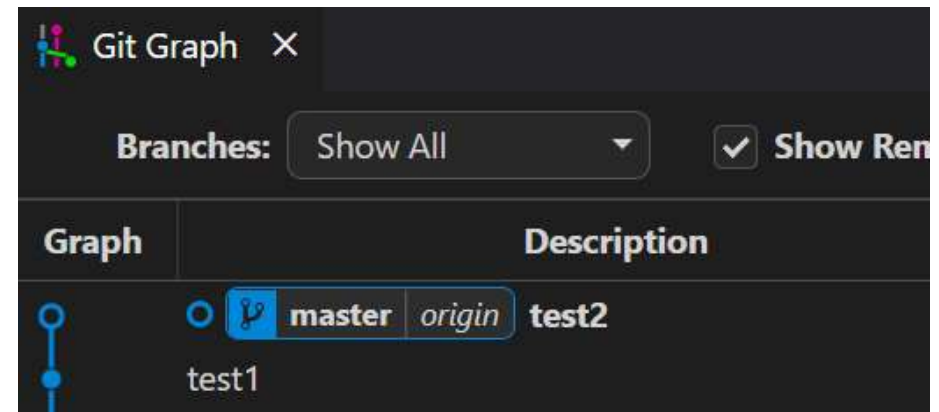
3. 버전관리

4. Vscode GIT협업하기

이제 test2를 원격저장소로 push 해보자!

그런 다음 Git Graph 또는 터미널창을 통해 확인해보면
, 원격저장소 origin의 최신버전이 test2로 바뀐 것을 볼수있다.

즉, 제일 최근에 변경된 내용이 test2 버전이라는 뜻!



```
PS C:\Users\silolab_ksh\Desktop\TES> git log --oneline --all --graph
* 47e7ae0 (HEAD -> master, origin/master) test2
* a9138fe test1
```

03. Visual Studio Code GIT협업

1. VScode란

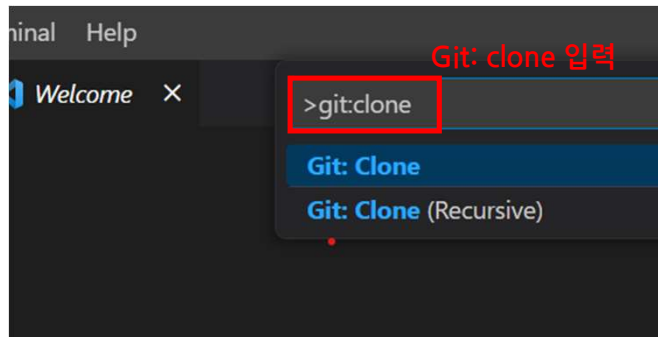
2. 기초셋팅

3. 버전관리

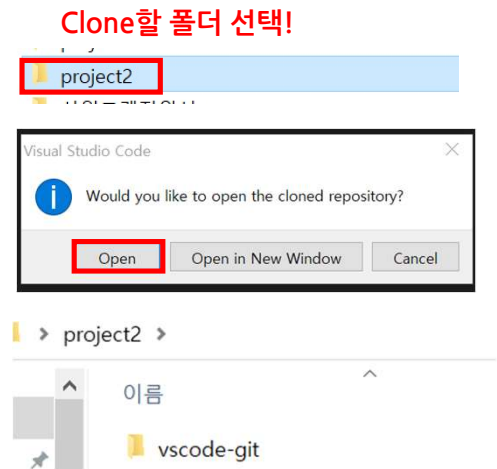
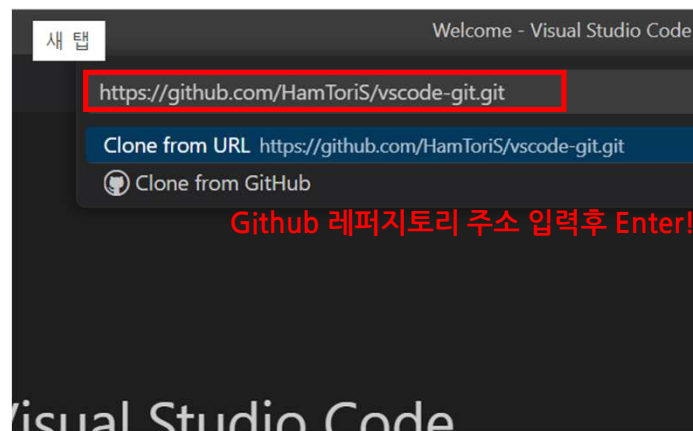
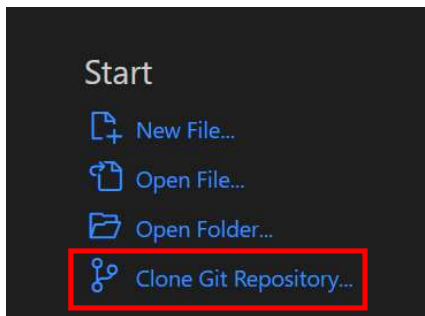
4. Vscode GIT협업하기

◆ Github 원격저장소의 레퍼지토리를 vscode지역저장소로 Clone하기

Ctrl+shift+p 해서 명령팔레트 열기



or



Clone하고 폴더 확인해보면
원격저장소Github에서 만든
레퍼지토리 내용이 저장돼있다.

03. Visual Studio Code GIT협업

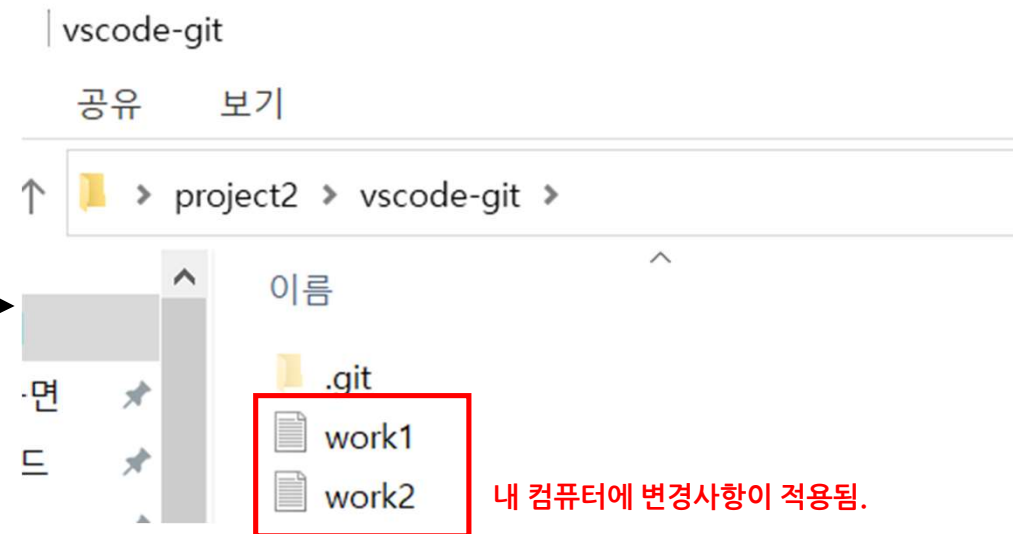
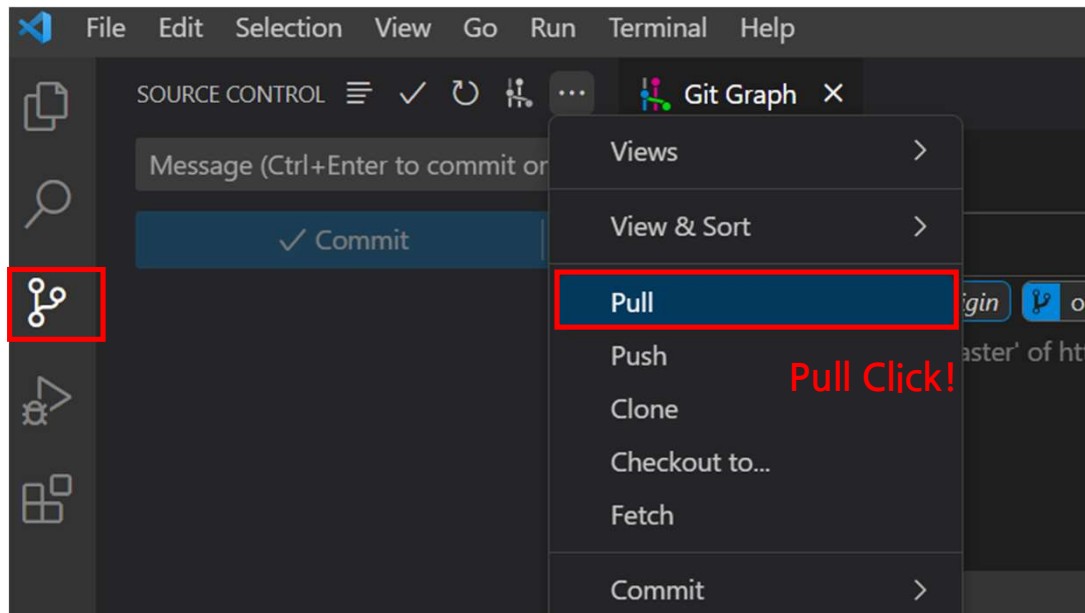
1. VScode란

2. 기초셋팅

3. 버전관리

4. Vscode GIT협업하기

◆ Github 원격저장소에 수정된 내용 내 폴더로 Pull 하기



Pull 완료!!

03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

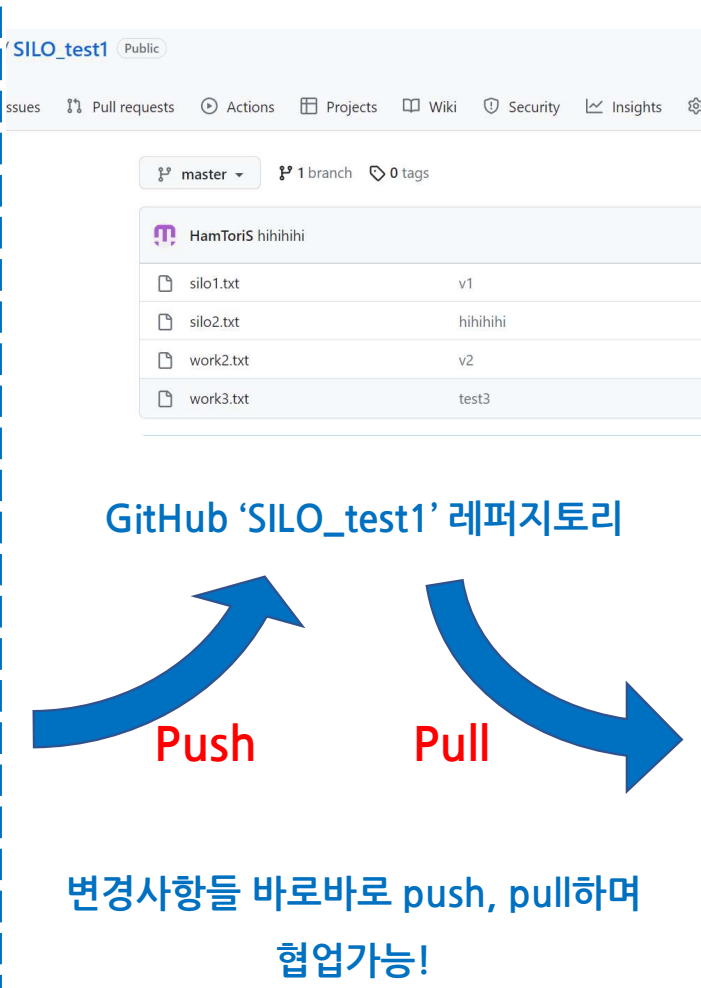
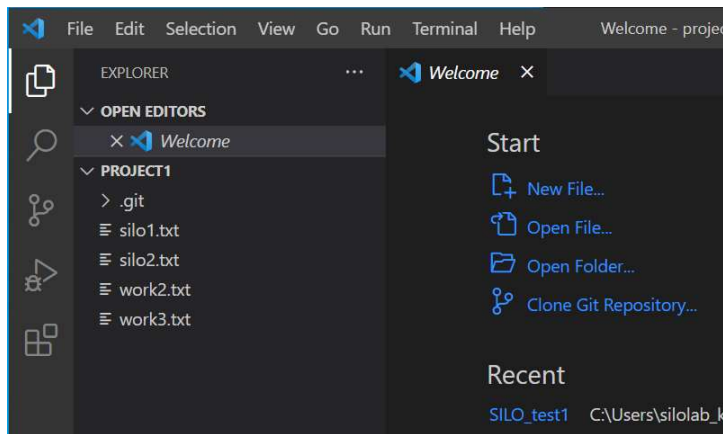
3. 버전관리

4. Vscode GIT협업하기

◆ Push ,Pull 정리!!!

깃허브 'SILO_test1' 레퍼지토리를
원격저장소(remote)로 설정한

A의 VScode



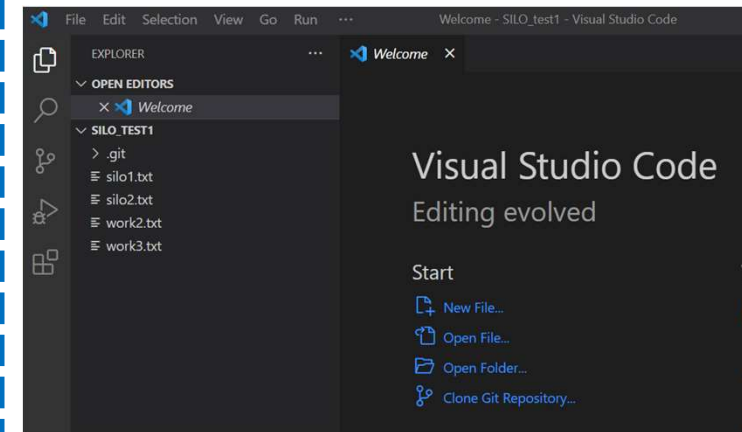
★주의할점!!!★

Push 하기전, 무조건 Pull 해야함!!

이유는 뒤에설명

깃허브 'SILO_test1' 레퍼지토리를
자신의 컴퓨터에 Clone한

B의 VScode



03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

3. 버전관리

4. Vscode GIT협업하기

★주의할점★

Pull 하고 Push 해야하는 이유

브랜치와 관련있어서 그렇다.

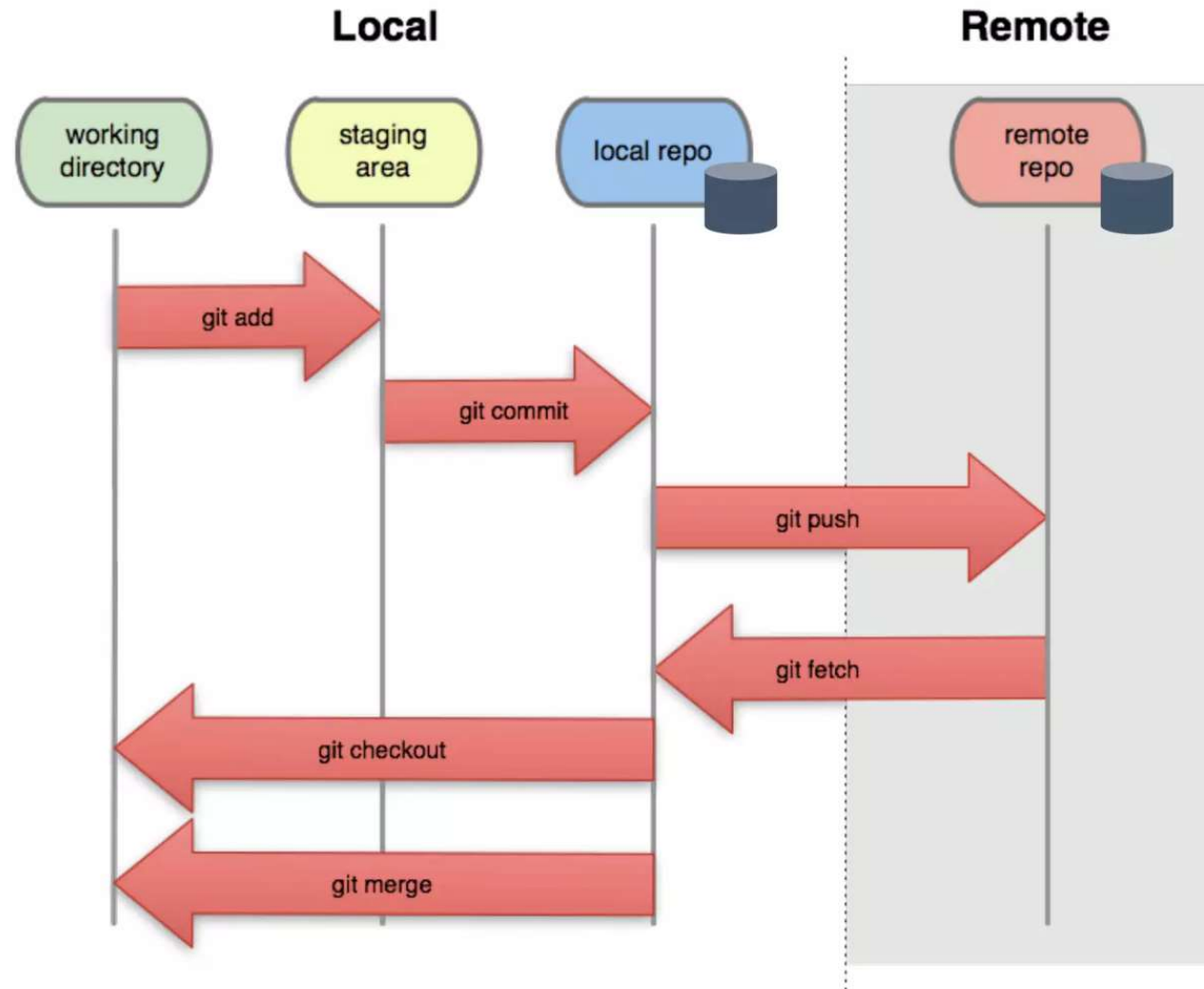
A가 변경내용'a'를 push하면 'a'가 최신버전인데,

B가 바로 뒤에 변경내용'b'를 버전이 꼬여서 거부된다.

그러므로 B는 먼저 pull을 통해 원격저장소의 최신버전을 등록을 하고,

자신의 것을 push를 해야 변경내용이 올라간다.

◆ Git을 이용한 작업흐름



03. Visual Studio Code GIT협업

1. VScode란

2. 기초셋팅

3. 버전관리

4. Vscode GIT협업하기

협업하기

협업을 위해선 동료들
원격저장소로 추가해야함.

03. Visual Studio Code GIT협업

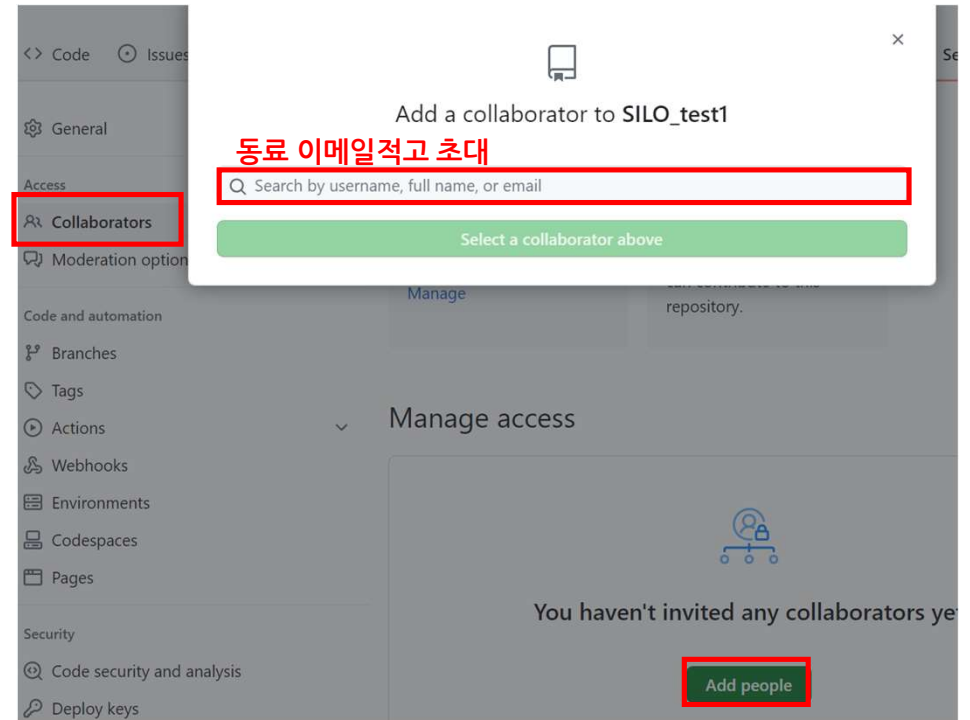
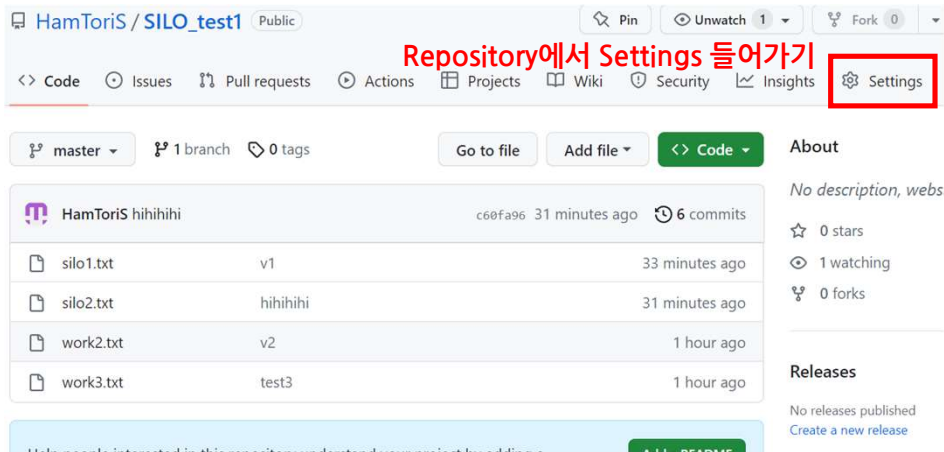
1. VScode란

2. 기초셋팅

3. 버전관리

4. Vscode GIT협업하기

◆ 협업하기



Ep.1 기본셋팅

팀장

로컬에서 프로젝트
기본셋팅



Github 레포지토리 업로드



팀원

팀장 레포를 **fork** 해옴



Fork 해온것 **본인 로컬에 clone** 하기

04. 코드스페이스

◆ 코드스페이스란?

2018년 마이크로소프트가 GitHub인수하면서 생김.



코드스페이스 한마디로

vscode를 너네 컴퓨터에서 돌리지 말고 우리가 좋은 컴퓨터를 줄 테니 그 위에서 돌려라!

깃허브랑 바로 연동도 되니까 코드관리도 더 편할거고
개발환경 셋팅에 신경 쓰지 말고 개발에만 집중해!

◆ 코드스페이스 사용이유

Why ? 내 컴퓨터에서 작업하면 되지 코드스페이스에서 왜?

개발할 때는 코드가 돌아가는 환경은 중요하다.

Window 나 mac이냐에 따라서 완전히 똑같은 코드나 프로그램이더라도 안 돌아갈 수가 있다.

환경이 다른 문제로 코드가 맞 가는 문제를 방지하는 것.

개발하는 환경과 배포될 환경을 일치시키는 것.


그리고 귀찮게 프로그램 이것저것 다 깔고 할 필요 없다.

참고: <https://youtu.be/2hZ-Q7h-9gE>

◆ 가격정보

지불 세부 사항

🕒 얼마나 자주 요금을 청구하시겠습니까?

☒ 매년 지불 1개월 무료 이용  ☐ 매월 지불

🔍 얼마나 많은 좌석을 포함하시겠습니까?

- 1 +

좌석

GitHub 팀에서 각 시트 비용은 연간 \$44입니다.

📄 청구 정보

비즈니스/기관 이름 *

VAT/GST ID

주소(사서함, 회사명, c/o) *

주소 입력란 2(아파트, 스위트, 유닛)

도시 *

우편번호

특정 국가의 경우 필수

국가/지역 *

시/도

당신의 나라를 선택하십시오 ⇅

특정 국가의 경우 필수

결제 이메일 *

결제 수단으로 계속


📄 지불 방법

오늘까지

\$44.00


🔍 할인 가격은 신용 카드 또는 PayPal로 결제하는 신규 연간 고객을 위한 것입니다. 첫째 이후에는 가격이 변경될 수 있습니다. GitHub는 가격 변경 최소 30일 전에 알림 이메일을 보내드립니다.

“ GitHub Actions를 사용하는 CI/CD를 사용하면 GitHub에서 바로 빌드, 테스트 및 배포할 수 있습니다. 빌드 시간을 80분에서 10분으로 줄였습니다.

 **존 패리스**
Pinterest의 엔지니어링 설계자

지불 세부 사항

🕒 얼마나 자주 요금을 청구하시겠습니까?

☐ 매년 지불 1개월 무료 이용  ☒ 매월 지불

🔍 얼마나 많은 좌석을 포함하시겠습니까?

- 1 +

좌석

GitHub 팀에서 각 시트 비용은 월 \$4입니다.

📄 청구 정보

비즈니스/기관 이름 *

VAT/GST ID

주소(사서함, 회사명, c/o) *

주소 입력란 2(아파트, 스위트, 유닛)

도시 *

우편번호

특정 국가의 경우 필수

국가/지역 *

시/도

당신의 나라를 선택하십시오 ⇅

특정 국가의 경우 필수

결제 이메일 *


결제 수단으로 계속

📄 지불 방법

오늘까지

\$4.00

“ GitHub Actions를 사용하는 CI/CD를 사용하면 GitHub에서 바로 빌드, 테스트 및 배포할 수 있습니다. 빌드 시간을 80분에서 10분으로 줄였습니다.

 **존 패리스**
Pinterest의 엔지니어링 설계자



<https://github.com/pricing>

Get the complete developer platform.

How often do you want to pay?

Monthly

Yearly

How often do you want to pay?

Monthly

Yearly

Get 1 month free

Free

The basics for individuals and organizations

\$0

per month
forever

Join for free

- > Unlimited public/private repositories
- > Automatic security and version updates
- > 2,000 CI/CD minutes/month
Free for public repositories
- > 500MB of Packages storage
Free for public repositories
- > New issues & projects (in limited beta)
- > Community support

MOST POPULAR

Team

Advanced collaboration for individuals and organizations

~~\$4~~ \$3.67

per user/month
for the first 12 months*

Continue with Team

팀 계정으로만 만들수 있음.

← Everything included in Free, plus...

- > Access to GitHub Codespaces
- > Protected branches
- > Multiple reviewers in pull requests
- > Draft pull requests
- > Code owners
- > Required reviewers

Team

Advanced collaboration for individuals and organizations

\$4 per user/month

Continue with Team

- ← Everything included in Free, plus...
- > Access to GitHub Codespaces
- > Protected branches
- > Multiple reviewers in pull requests
- > Draft pull requests

Enterprise

Security, compliance, and flexible deployment

~~\$21~~ \$19.25

per user/month
for the first 12 months*

Start a free trial

Contact Sales

- ← Everything included in Team, plus...
- > Enterprise Managed Users
- > User provisioning through SCIM
- > Enterprise Account to centrally manage multiple organizations
- > Environment protection rules
- > Audit Log API



GitHub Codespaces

Your instant dev environment.

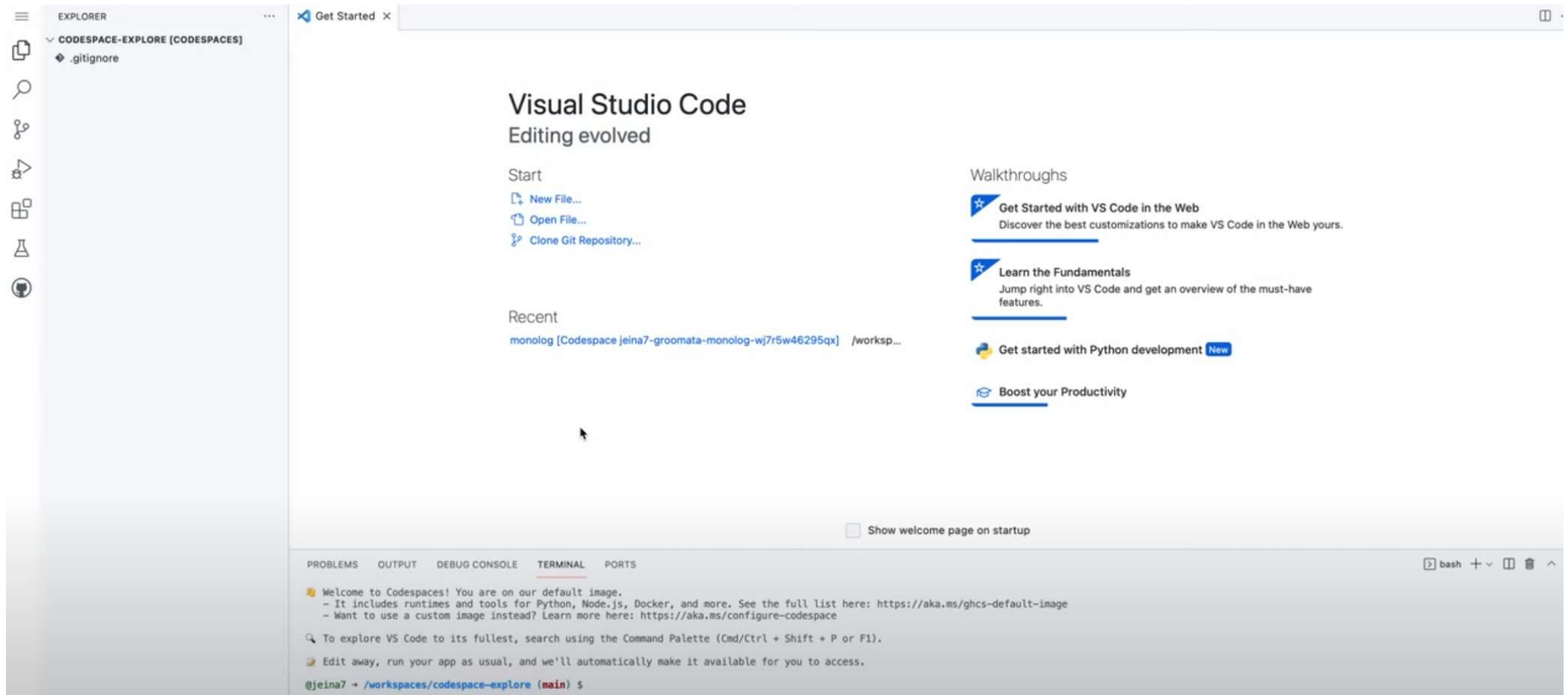


Project requirements

Codespaces can scale to the complexity of your project requirements. Select the machine size that best suits your work:

- ☐ **2 cores, 4GB RAM** **\$0.18**
Static web apps, small databases, commandline applications /hr
- ☐ **4 core, 8GB RAM** **\$0.36**
Dynamic web apps, relational databases, analytics /hr
- ☐ **8 core, 16GB RAM** **\$0.72**
Multi-container applications, content management systems /hr
- ☐ **16 core, 32GB RAM** **\$1.44**
Compute-intensive database workloads, complex web apps /hr
- ☒ **32 core, 64GB RAM** **\$2.88**
Compute-intensive applications (AI and deep learning) /hr

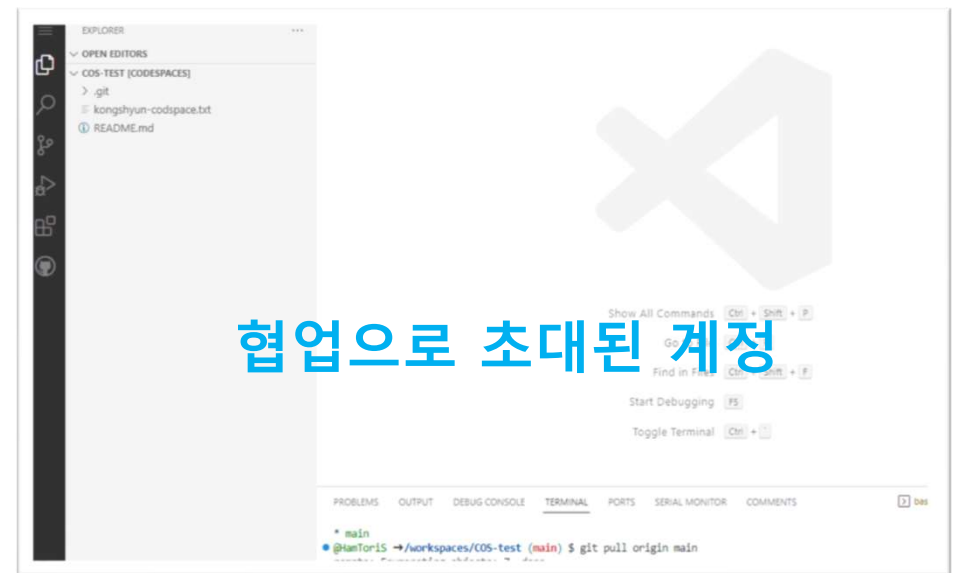
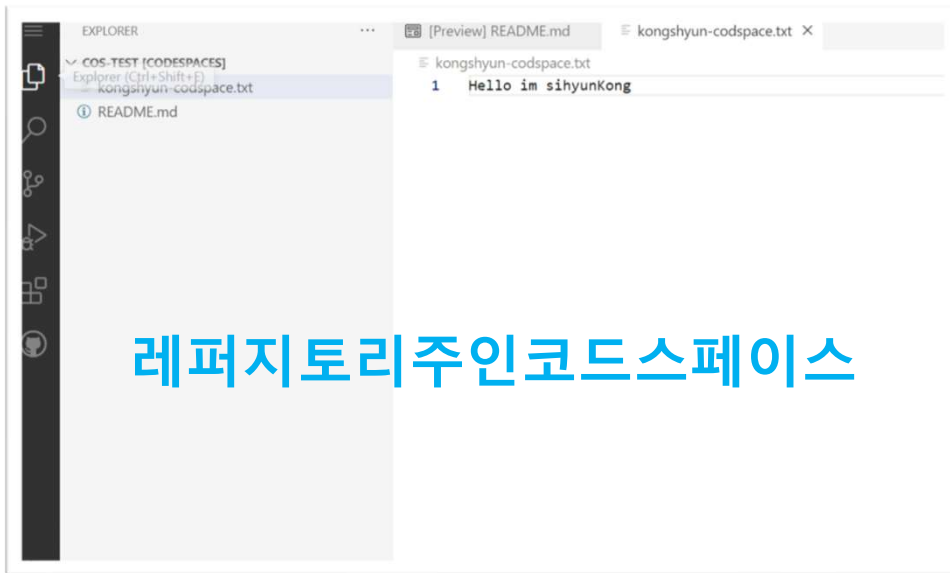
◆ 코드스페이스시작화면



웹에서 하는 vscode, git등 개발환경은 다 셋팅 돼있음.

코드스페이스연동

COS-TEST레퍼지토리 코드스페이스에 접속한 두 계정



설정, 확장 프로그램들 동기화 된다.

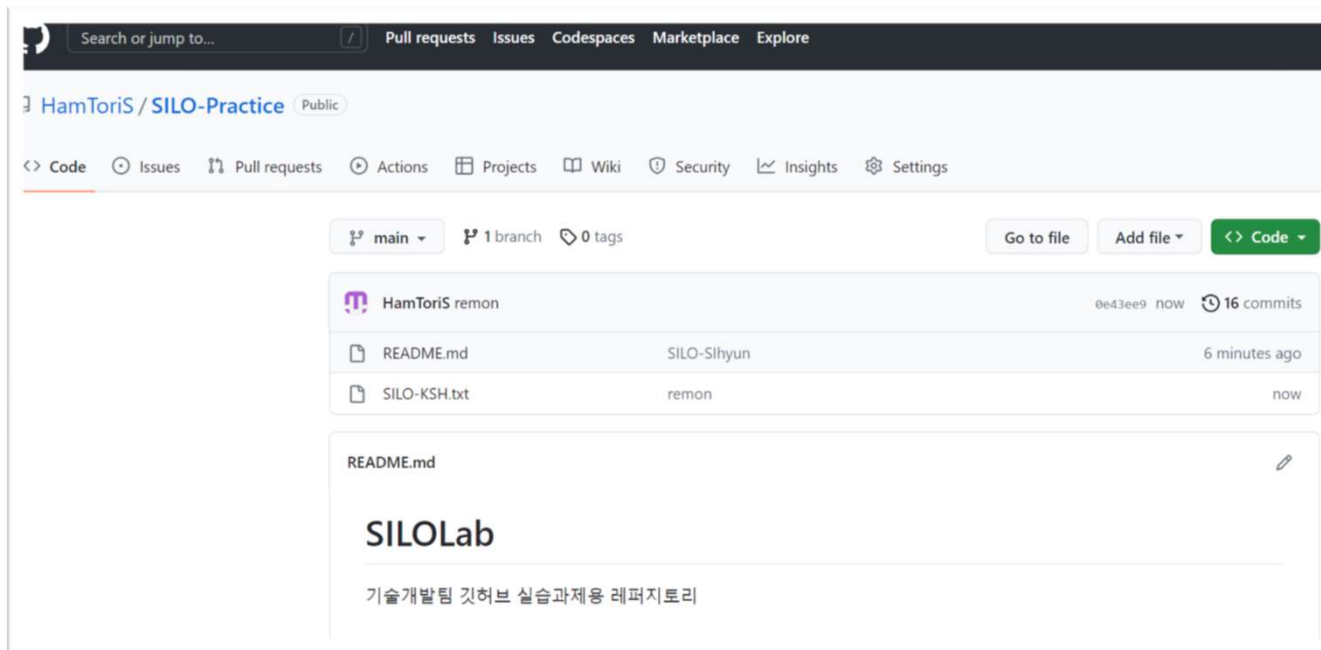
Turn on the setting sync.하면됨.

05. 깃허브실습과제

[과제개요] 원격저장소와 자신의 로컬저장소 동기화 시키기.

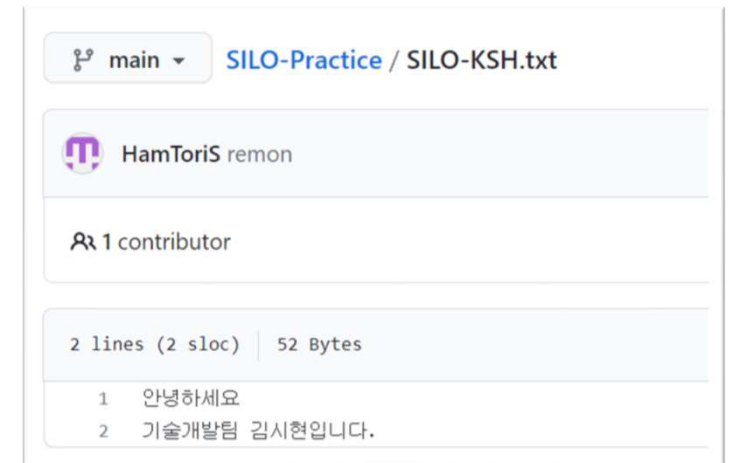
- Clone, Pull, Push 하기

- 원격저장소주소: <https://github.com/HamToriS/SILO-Practice.git>



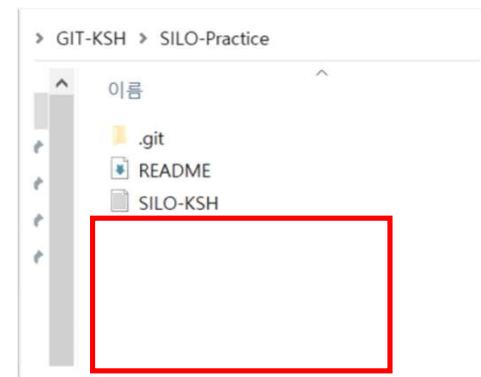
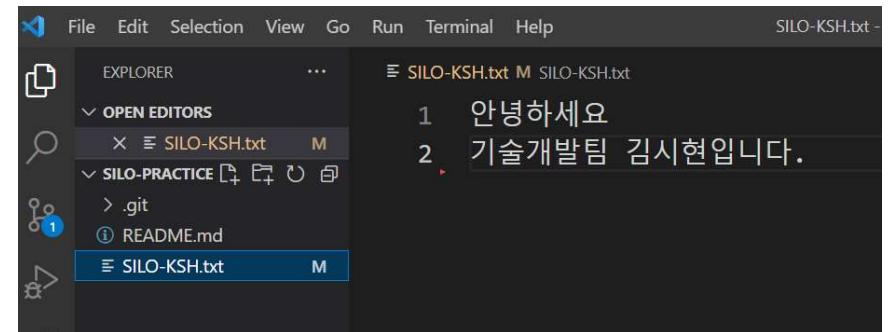
‘SILO-Practice’라는 레퍼지토리를 만들었습니다.

‘SILO-KSH.txt’파일이 있습니다. Commit 메시지는 ‘remon’으로 했습니다.



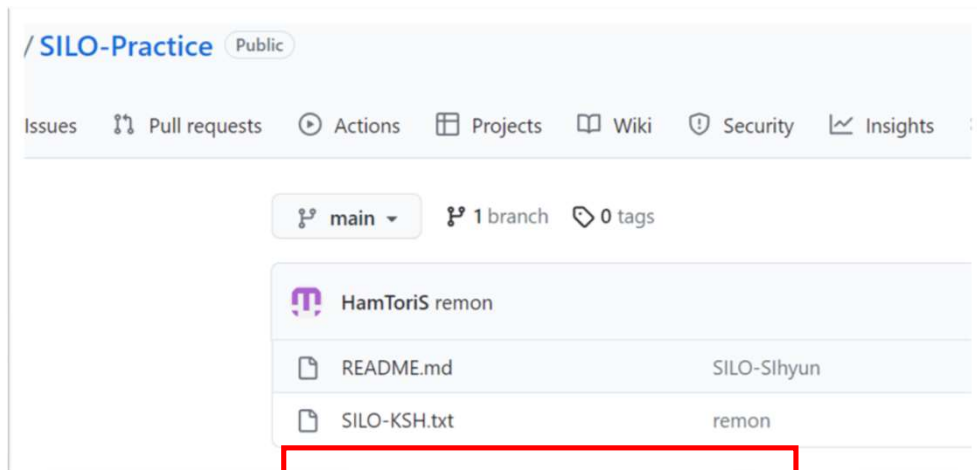
[과제설명]

- ① 'SILO-Practice' 레퍼지토리를 자신의 컴퓨터 Vscode로 **Clone**하기.
...Clone하는 폴더 위치는 바탕화면에 'GIT-이름이니셜'폴더 생성하여 하기.
- ② Vscode에서 New file 만들기.
...file내용은 옆 내용과 같이. 이름만 다르게.
- ③ **Pull**하고나서 , 만든 파일을 원격저장소로 **push** 하기
...**Commit** 메시지는 '과일이름' 으로하기. ex) remon
- ④ 다른 사람이 만든 파일이 내 컴퓨터 폴더로 들어온 지 확인후 캡처.

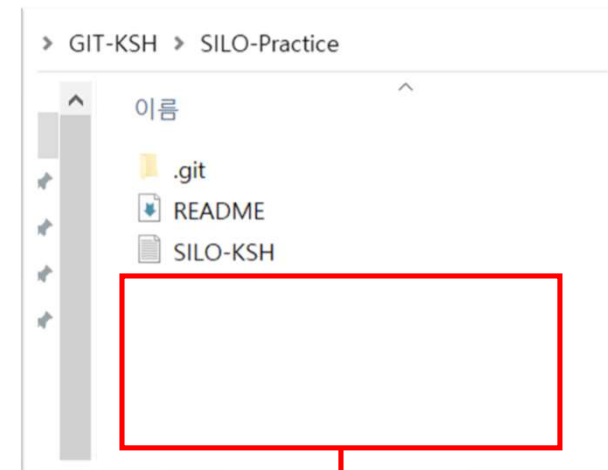


[과제목표]

1



2



이곳에 팀원들 파일이 다 들어오면 과제성공!

완성후 1번, 2번 사진처럼 캡처해서 김시현 한테 개인 메신저로 보내주세요!



기술개발팀 김시현