



## 2강 - 로컬 저장소 생성하기

VS code 설치

Git 설치

간단한 리눅스 명령어

Git을 사용하는 2가지 방법

Git의 새로운 저장소 생성 방법

예제 1. 원하는 위치에 프로젝트 폴더를 생성한다.

예제 2. 프로젝트 폴더를 깃 로컬 저장소로 만들기 - git init

Git의 버전관리 장소 3가지

예제 3. 버전관리할 파일 git에 추가하기 - git add

예제 4. 첫번째 버전 생성하기 - git commit

실습 1. 파일을 2개 추가하고 커밋 연습하기

예제 5. git log의 여러가지 옵션

---

### VS code 설치

- 안내영상 참조 (<https://www.youtube.com/watch?v=lkwYQgtsYko>)


<https://www.youtube.com/watch?v=lkwYQgtsYko>

### Git 설치

- git 다운로드 링크

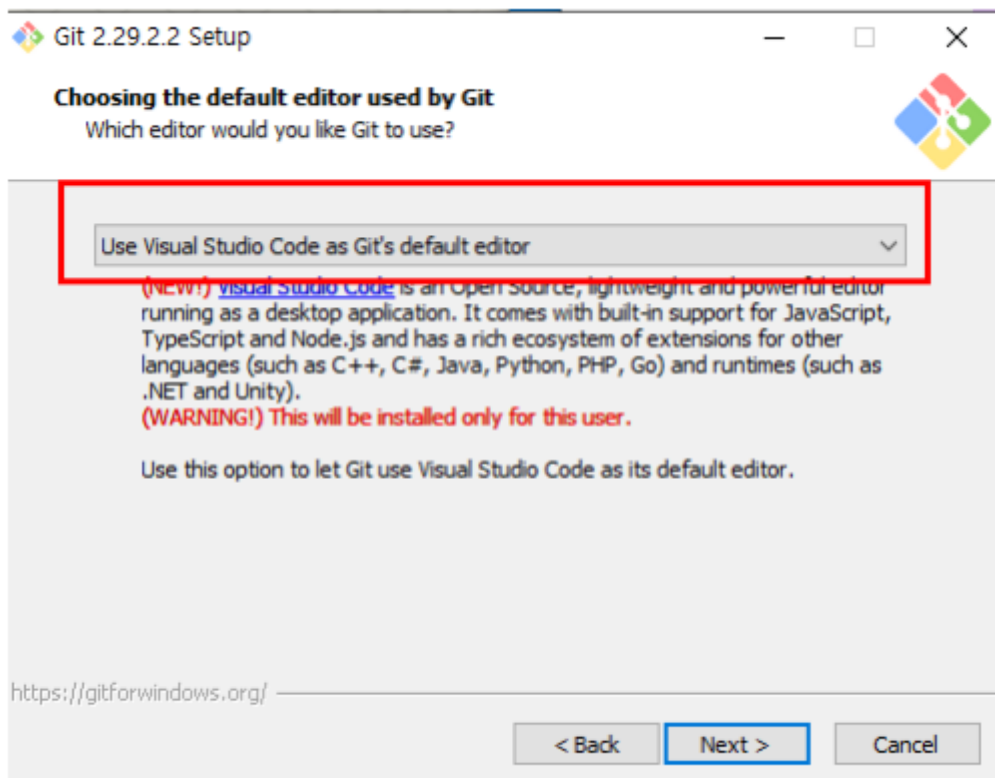
## Git

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local

 <https://git-scm.com/>



- next 쪽 누르다가 아래 화면에서 'Use Visual Studio Code as Git's default editor' 선택



## 간단한 리눅스 명령어

ls	하위 폴더, 파일 보기
ls -l ( ll )	하위 폴더, 파일 자세히 보기
pwd	현재 위치한 폴더 출력
cd	원하는 경로로 이동
mkdir 폴더명	새 디렉토리 생성

## Git을 사용하는 2가지 방법

1. CLI : 명령어 기반 버전관리 (이걸로 공부)
2. GUI : 그래픽 기반 버전관리 툴

## Git의 새로운 저장소 생성 방법

- 기존의 저장소를 복제한다. (clone)
- 기존 버전관리하고 있지 않은 기존 폴더에서 작업을 시작한다. (init)
- 새로운 저장소 역할을 할 폴더를 생성한다. (init)

### 예제 1. 원하는 위치에 프로젝트 폴더를 생성한다.

- 설치한 git의 git bash 프로그램을 실행해주세요.

```
PC - 08@DESKTOP-M0L64VA MINGW64 ~  
$ |
```

- 다음 명령어를 입력하여 폴더를 생성하고 해당 폴더로 이동합니다.

```
$ cd e: // e 드라이브로 이동 (e 드라이브 없을 시 다른 드라이브로 이동)
$ mkdir git-practice // git-practice 폴더 생성
$ cd git-practice // git-practice로 이동
```

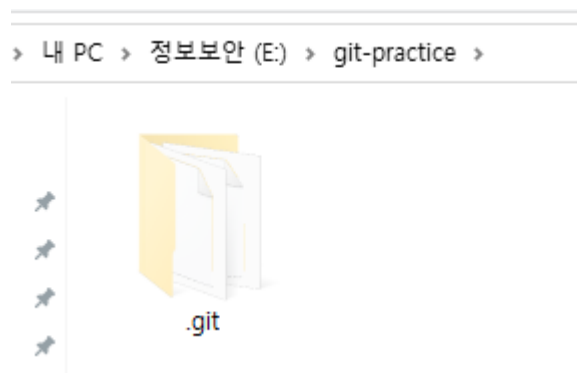
## 예제 2. 프로젝트 폴더를 깃 로컬 저장소로 만들기 - git init

```
$ git init
```

- git init 명령어는 git 버전관리의 시작을 의미함
- 따라서 최초 1회만 입력하면 됨
- git으로 관리되는 프로젝트 폴더는 자동으로 master라는 이름의 브랜치가 생성됨

```
PC - 08@DESKTOP-M0L64VA MINGW64 /e/git-practice (master)
$
```

- 해당 디렉토리로 가서 숨김폴더를 확인해보면 .git 이라는 폴더가 생성됨
- .git 폴더는 삭제시 버전관리 기록이 모두 사라지므로 주의!!




## Git의 버전관리 장소 3가지

1. **Working Tree** : 버전관리에서 등록되지 않은 파일들이 사는 공간
2. **Staging Area** : 버전에 추가될 파일들이 사는 공간
3. **Repository** : 버전으로 등록된 파일들이 사는 공간

### 예제 3. 버전관리할 파일 git에 추가하기 - git add

```
$ git status // git 상태 확인
```

```
$ notepad hello.txt
```

 hello.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

안녕하세요

```
$ git status
```

- hello.txt 파일을 만든 후 git status명령어를 쳐보면 hello.txt파일을 추적한 것을 볼 수 있음.
- 명령어를 잘 읽어보면 commit을 하기 위해 add를 해달라는 내용을 볼 수 있음.

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

hello.txt

nothing added to commit but untracked files present (use "git add" to track)

```
$ git add hello.txt // git에 파일 추가하기
$ git status
```

## 예제 4. 첫번째 버전 생성하기 - git commit

- 커밋은 버전을 생성하는 것을 말함.
- 커밋을 하려면 커밋을 한 사람의 정보가 필요하기 때문에 정보를 등록해야 함.

```
$ git config --global user.name "유저닉네임" // 닉네임은 github, gitlab에서 사용할 닉네임 추천
$ git config --global user.email "유저이메일" // 이메일은 실제 github이나 gitlab 가입메일
```

- 그러면 이제 커밋을 할 준비가 됨.

```
$ git commit -m "커밋 메시지"
$ git log
```

- git log 명령어로 커밋된 버전객체를 확인할 수 있음.

```
$ git log
commit 4eb4bd1a9ec0f942b059cb1d520984403975f517 (HEAD -> master)
Author: soongu <naong@naver.com>
Date: Mon Feb 13 16:19:30 2023 +0900
```

Adding hello.txt for greeting

## 실습 1. 파일을 2개 추가하고 커밋 연습하기

- 배운 깃 명령어를 통해 bye.txt 를 생성하고 텍스트를 3줄 아무거나 입력한 후 `$ git status` 명령어를 통해 상태 확인 후 add commit을 해보세요.
- 또한 food.txt 파일을 생성하고 같은 작업을 반복하세요.

### ▼ 예시 답안 보기

```
PC - 08@DESKTOP-M0L64VA MINGW64 /e/git-practice (master)
$ notepad bye.txt
```

```
PC - 08@DESKTOP-M0L64VA MINGW64 /e/git-practice (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    bye.txt
```

nothing added to commit but untracked files present (use "git add" to track)

```
PC - 08@DESKTOP-M0L64VA MINGW64 /e/git-practice (master)
$ git add bye.txt
```

```
PC - 08@DESKTOP-M0L64VA MINGW64 /e/git-practice (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   bye.txt
```

```
PC - 08@DESKTOP-M0L64VA MINGW64 /e/git-practice (master)
$ git commit -m "Adding bye.txt for the practice1"
[master a831220] Adding bye.txt for the practice1
 2 files changed, 4 insertions(+)
 create mode 100644 bye.txt
 create mode 100644 food.txt

PC - 08@DESKTOP-M0L64VA MINGW64 /e/git-practice (master)
$ git status
On branch master
nothing to commit, working tree clean
```

## 예제 5. git log의 여러가지 옵션

- git log 는 여러 옵션을 통해 다양한 방식으로 커밋 이력을 확인할 수 있음
- 키보드 방향키를 아래로 내리면 로그를 연속해서 볼 수 있음
- Q를 눌러 로그가 길 경우 중간에 빠져 나갈 수 있음
- 초기 실습시에는 ``--all --graph`` 옵션은 브랜치를 활용하지 않았기 때문에 별 차이점이 보이지 않음

```
$ git log --stat    // log를 자세하게 보여줌
$ git log -p       // log를 더 자세하게 보여줌
$ git log --oneline // log를 간략하게 한줄로 보여줌
$ git log --all     // 다른 브랜치의 로그도 모두 보여줌
$ git log --graph   // 브랜치들의 로그를 그래프형식으로 보여줌
```