

# Git & GitHub

◆ 깃 시작하기

정수아

# Contents

**01** 깃이란 무엇인가

**02** 깃 프로그램의 종류

**03** 깃 설치하기

**04** 깃 환경 설정

**05** 기본 리눅스 명령어

**06** 빔(Vim) 사용하기



01

깃이란 무엇인가

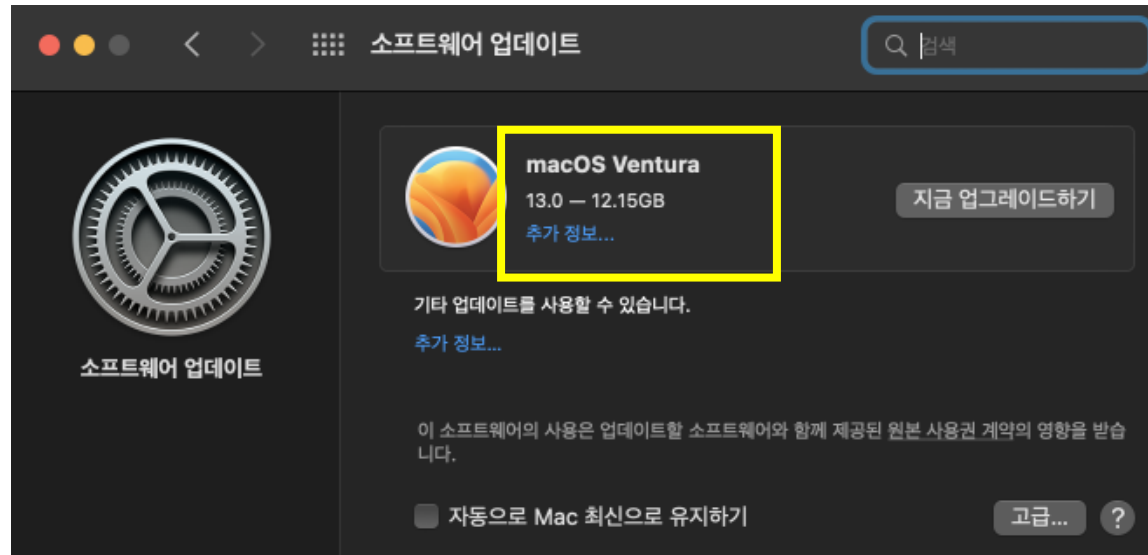
# 깃(Git)의 탄생

## ❖ 깃(Git)

- 2005년 리누스 토르발스가 처음 개발
- 특징
  - 컴퓨터 파일의 변경사항을 추적
  - 사용자들의 파일 작업을 조율하기 위한 분산 버전 관리 시스템
- 개발자들은 깃을 통해 수많은 소스 코드를 효율적으로 관리



# 버전 관리

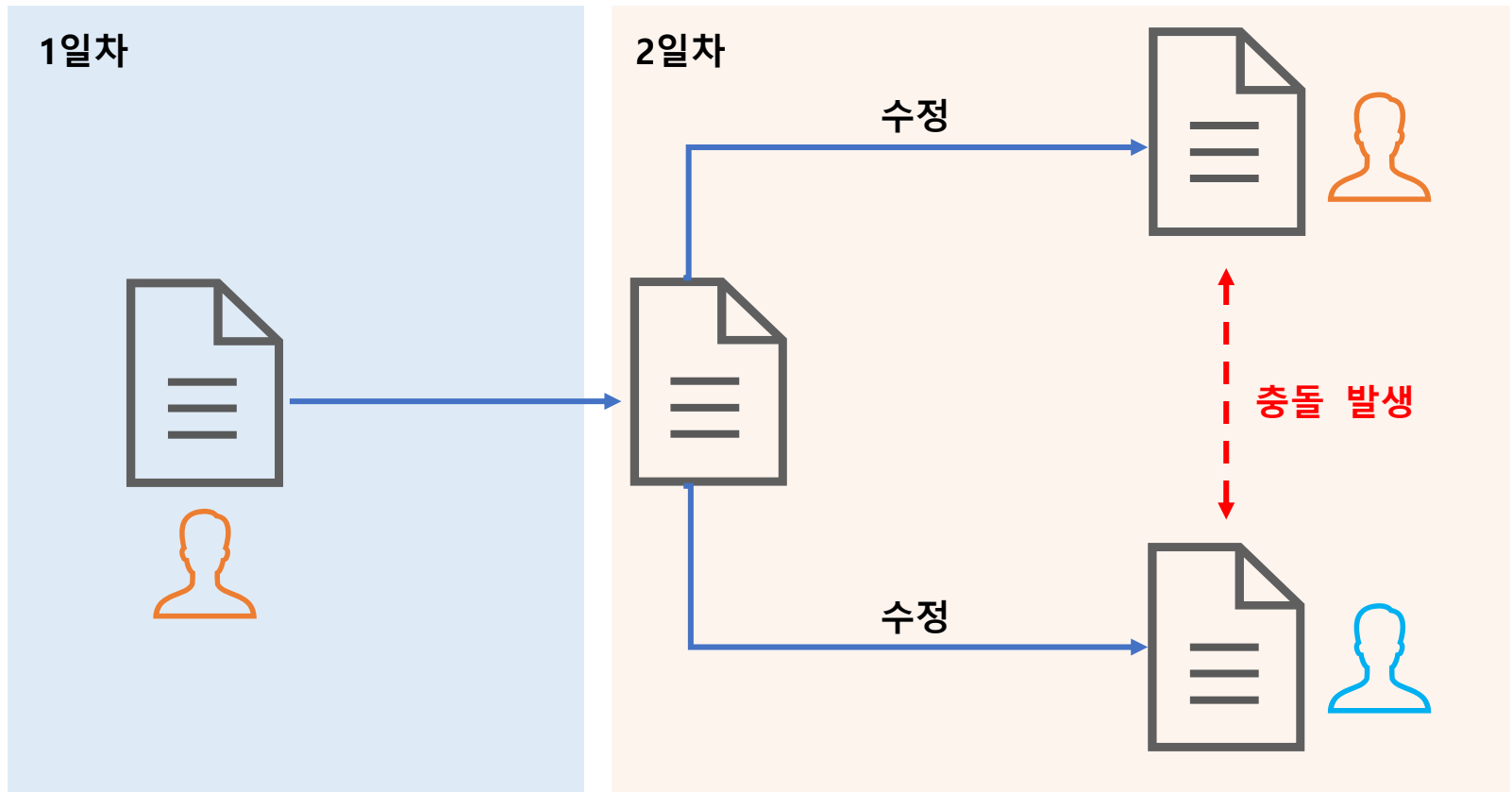


## ❖ 버전 관리의 필요성

- 업데이트를 체계적으로 관리하기 위해서는 소스 코드에 언제, 어떤 변화가 있었는지 기록하고 추적할 수 있어야 함

# 버전 관리 시스템

## ❖ 버전 관리 시스템의 필요성



# 깃의 특징

- ❖ 버전 관리(Version Control)
- ❖ 백업(Backup)
- ❖ 협업(Collaboration)

# 깃의 특징

## ❖ 버전 관리(Version Control)

- 문서를 작성한 뒤 원본을 남겨두고 수정한 내용을 저장해야 하는 경우
- '다른 이름으로 저장'을 주로 사용



원본



수정



최종



진짜최종

☆ 깃을 사용한다면? ☆

파일 이름은 유지하면서 문서를 수정할 때마다 언제 수정했는지, 어떤 것이 변경되었는지 기록 가능

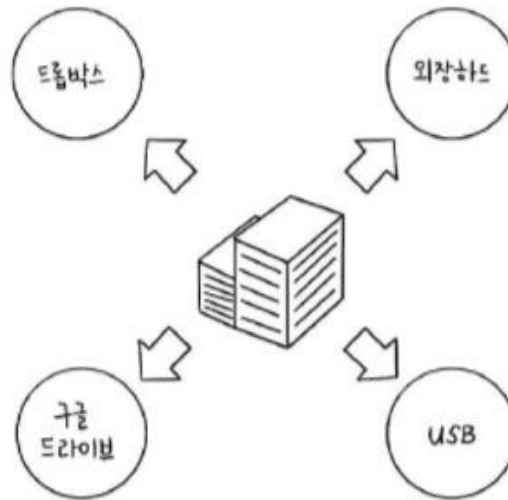


# 깃의 특징

## ❖ 백업(Backup)

- 백업이란?

- 현재 내 컴퓨터에 있는 자료를 다른 곳에 복제
- USB 디스크, 외장하드, 드롭박스(Dropbox) 등



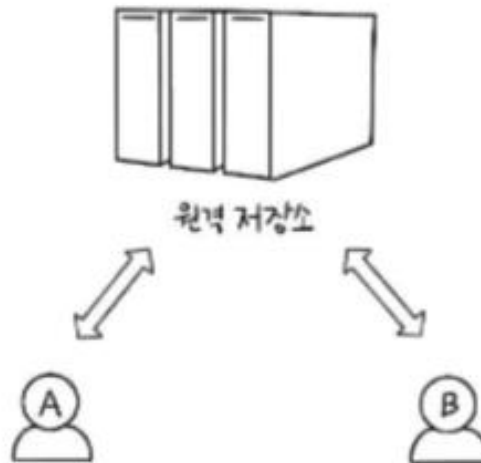
☆ 깃허브(GitHub) ☆

깃 파일을 위한 원격 저장소 또는 온라인 저장소

# 깃의 특징

## ❖ 협업(Collaboration)

- 깃허브와 같은 온라인 서비스를 사용하면 여러 사람이 함께 일할 수 있음



### ☆ 깃허브(GitHub)의 장점 ☆

- 팀원들이 파일을 편하게 주고받으면서 일할 수 있음
- 누가 어느 부분을 수정했는지 기록에 남음

# 깃의 특징

## ❖ 버전 관리(Version Control)

- 문서를 수정할 때마다 언제 수정했는지, 어떤 것을 변경했는지 편하고 구체적으로 기록하기 위한 버전 관리 시스템

## ❖ 백업(Backup)

- 현재 컴퓨터의 자료를 인터넷 상의 공간(원격 저장소)에 백업
- 주로 깃허브(GitHub) 사용

## ❖ 협업(Collaboration)

- 원격 저장소를 통해 여러 사람이 함께 일할 수 있음
- 누가, 언제, 어느 부분을 수정했는지 기록 가능



02

## 깃 프로그램의 종류

# 깃 프로그램

## ❖ 깃허브 데스크톱(GitHub Desktop)

- 깃허브에서 제공하는 프로그램
- 그래픽 사용자 인터페이스(GUI)로 구현
- 사용이 쉬워 누구나 배울 수 있음
- 기본적인 기능만 제공

## ❖ 토터스깃(TortoiseGit)

- 윈도우 전용 프로그램
- 윈도우 탐색기의 빠른 메뉴에 추가되는 프로그램

## ❖ 소스트리(Source Tree)

- 깃의 기본 기능부터 고급 기능까지 제공
- 사용법은 복잡하지만 익숙해지면 자유롭게 깃을 활용할 수 있음

# 깃 프로그램

## ❖ 커맨드 라인 인터페이스(CLI)

- 터미널에 직접 명령을 입력해서 깃을 사용하는 방식
- 리눅스 명령어 및 깃 명령어를 알아야 사용 가능
- 대부분의 개발자들은 이 방식으로 깃을 사용

# 깃 프로그램

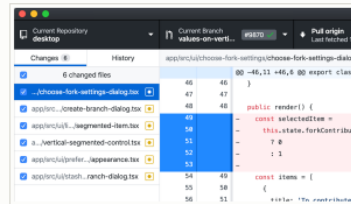
## ❖ 더 많은 깃 프로그램

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

### GUI Clients

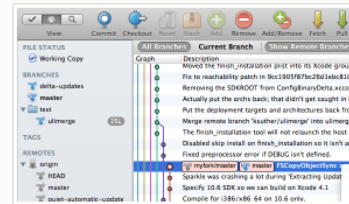
Git comes with built-in GUI tools for committing (`git-gui`) and browsing (`gitk`), but there are several third-party tools for users looking for platform-specific experience.

If you want to add another GUI tool to this list, just [follow the instructions](#).

[All](#)[Windows](#)[Mac](#)[Linux](#)[Android](#)[iOS](#)

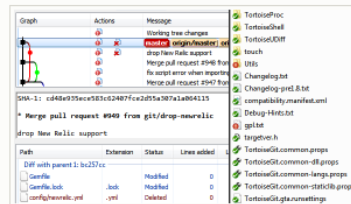
#### GitHub Desktop

Platforms: Mac, Windows  
Price: Free  
License: MIT



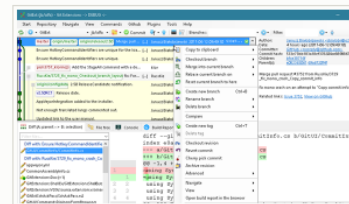
#### SourceTree

Platforms: Mac, Windows  
Price: Free  
License: Proprietary



#### TortoiseGit

Platforms: Windows  
Price: Free  
License: GNU GPL



#### Git Extensions

Platforms: Linux, Mac, Windows  
Price: Free  
License: GNU GPL



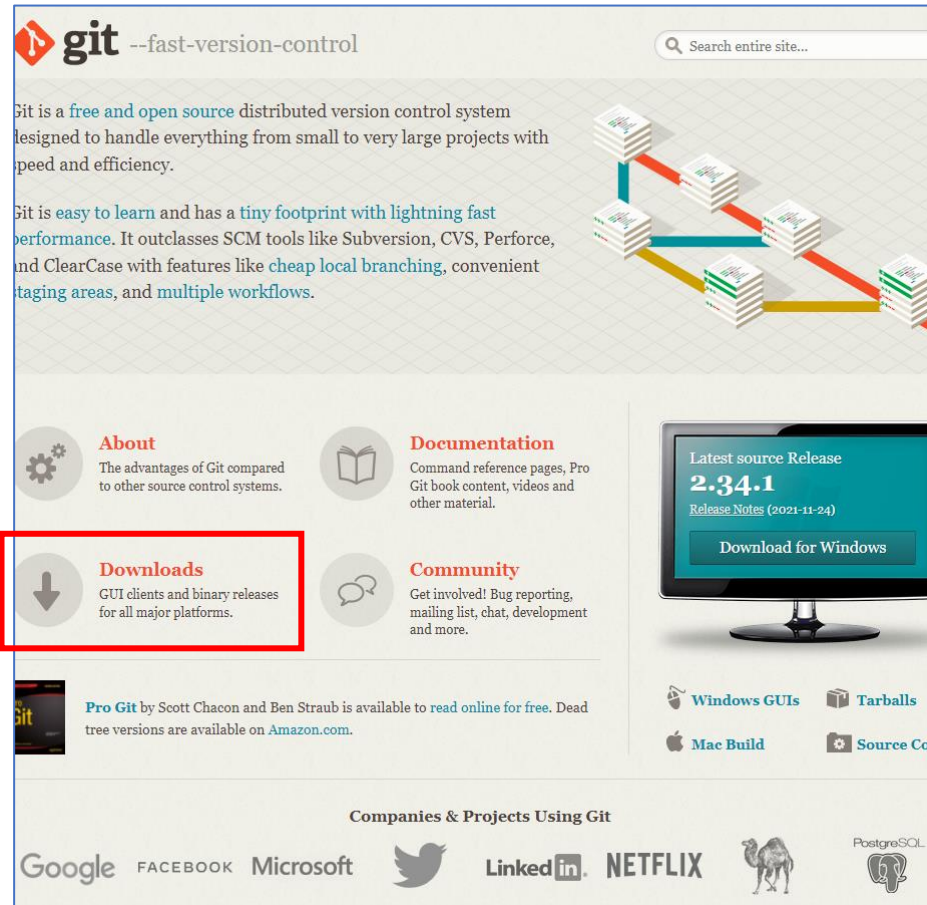
03

## 깃 설치하기

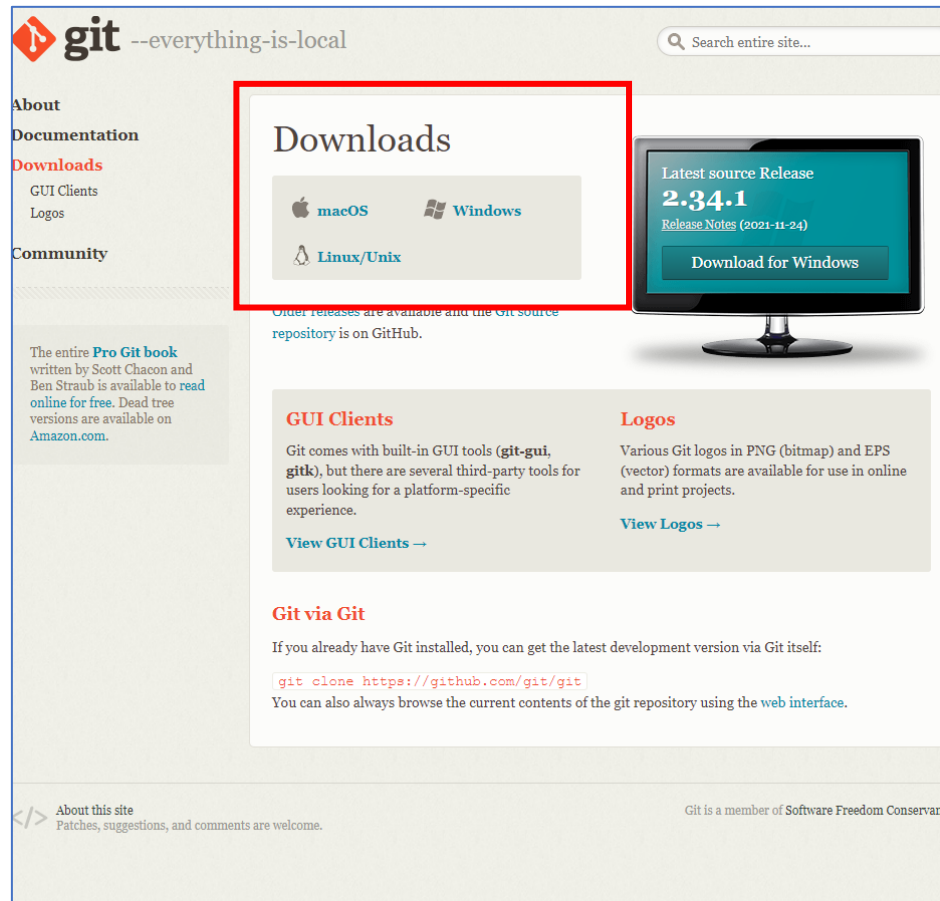


# 깃 설치하기 - Windows


❖ <https://git-scm.com/>



# 깃 설치하기



# 깃 설치하기 - Windows

 **git** --local-branching-on-the-cheap

Search entire site...

**About**  
**Documentation**  
**Downloads**  
GUI Clients  
Logos  
**Community**

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

## Download for Windows

[Click here to download](#) the latest (2.37.0) 64-bit version of **Git for Windows**. This is the most recent maintained build. It was released **6 days ago**, on 2022-06-27.

~~Other Git for Windows downloads~~

**Standalone Installer**  
[32-bit Git for Windows Setup.](#)  
[64-bit Git for Windows Setup.](#)

**Portable ("thumbdrive edition")**  
[32-bit Git for Windows Portable.](#)  
[64-bit Git for Windows Portable.](#)


**Using winget tool**  
Install [winget tool](#) if you don't already have it, then type this command in command prompt or Powershell.  

```
winget install --id Git.Git -e --source winget
```


The current source code release is version **2.37.0**. If you want the newer version, you can build it from [the source code](#).

### Now What?


Now that you have downloaded Git, it's time to start using it.



**Read the Book**  
Dive into the Pro Git book and learn at your own pace.

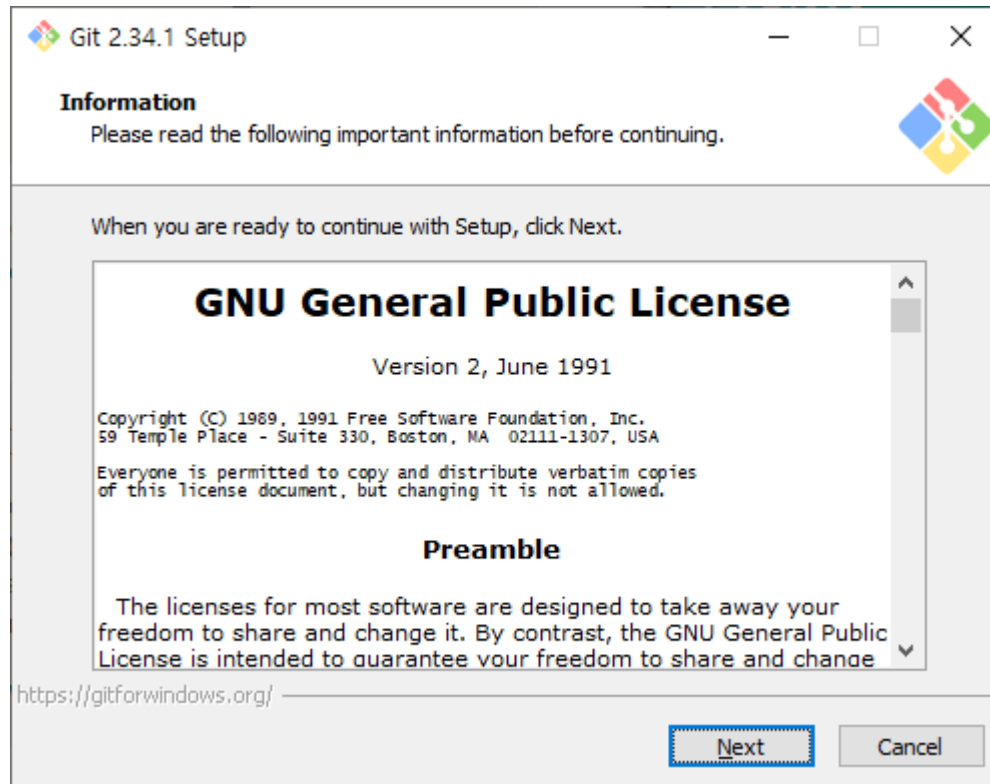


**Download a GUI**  
Several free and commercial GUI tools are available for the Windows platform.

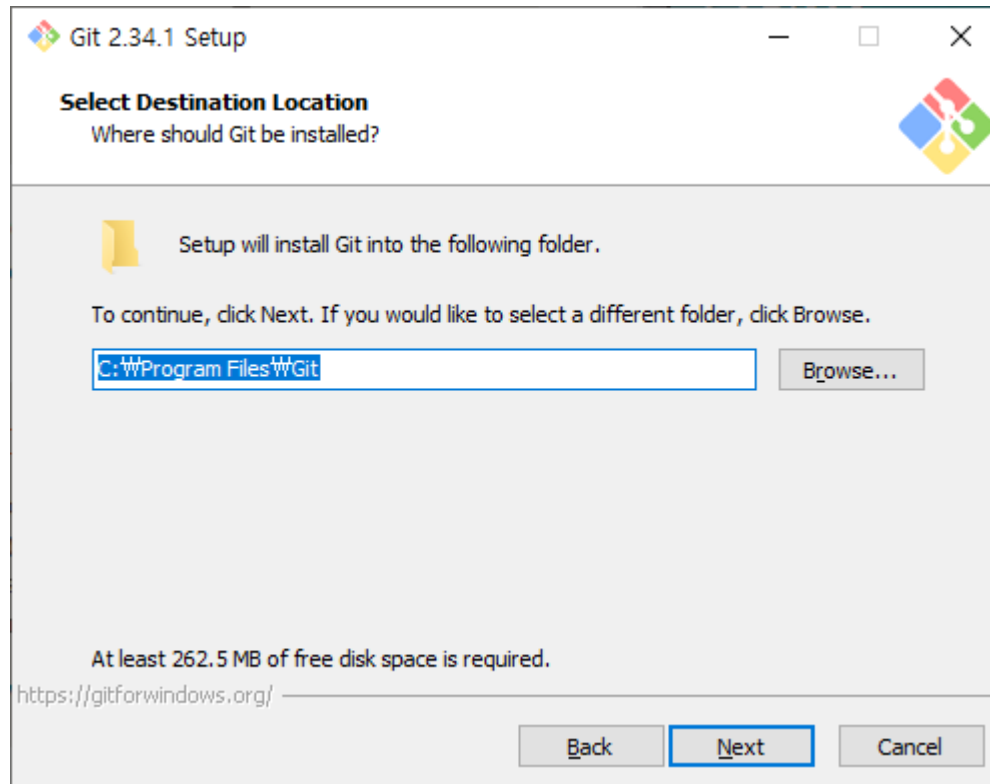


**Get Involved**  
A knowledgeable Git community is available to answer your questions.

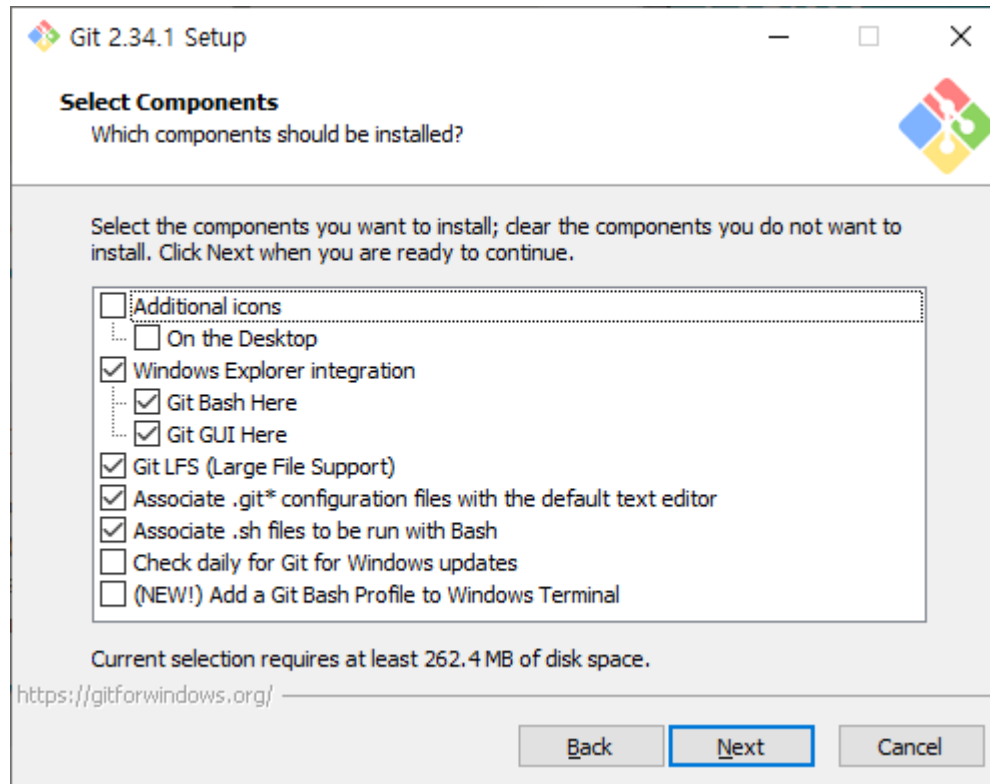
# 깃 설치하기 - Windows



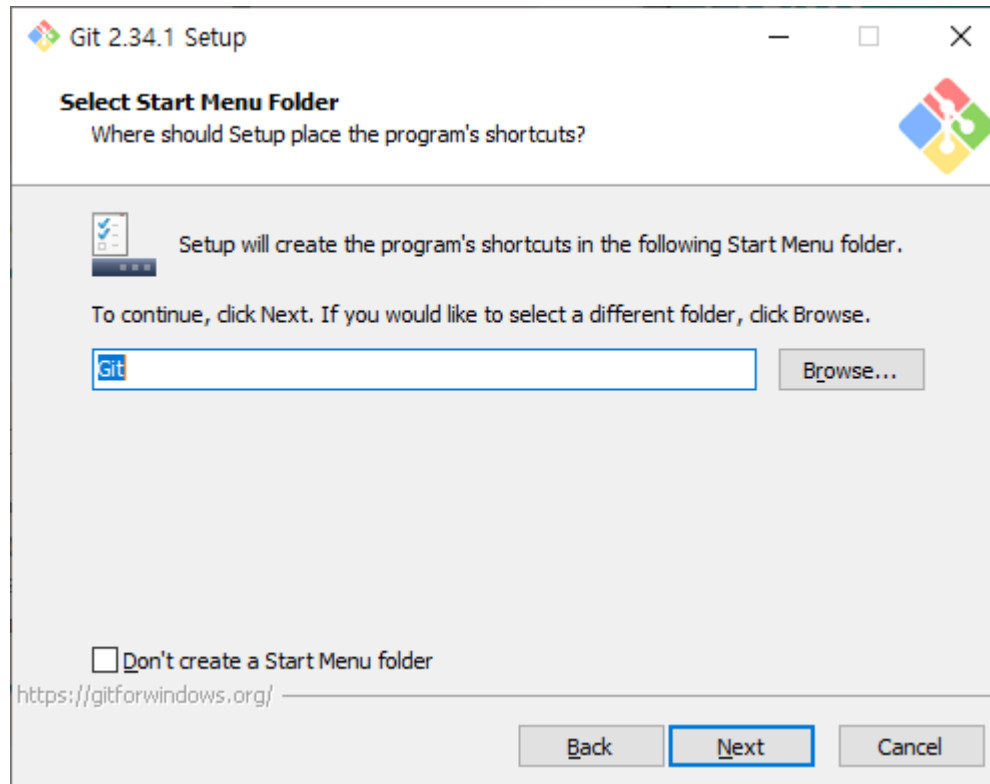
# 깃 설치하기 - Windows



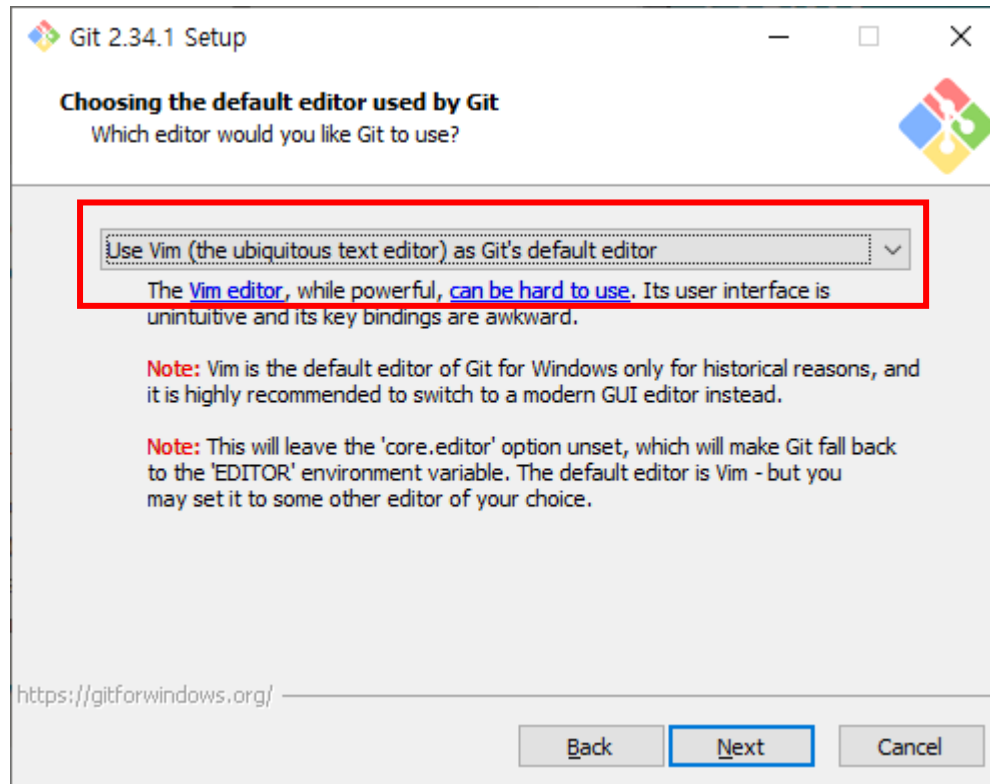
# 깃 설치하기 - Windows



# 깃 설치하기 - Windows

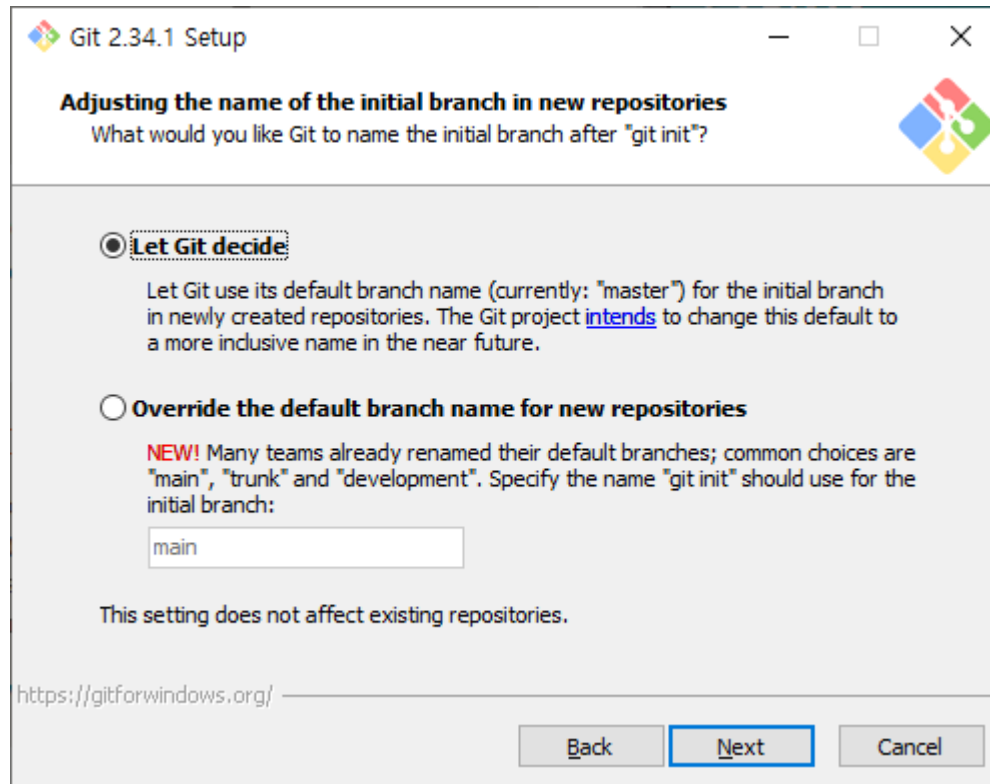


# 깃 설치하기 - Windows

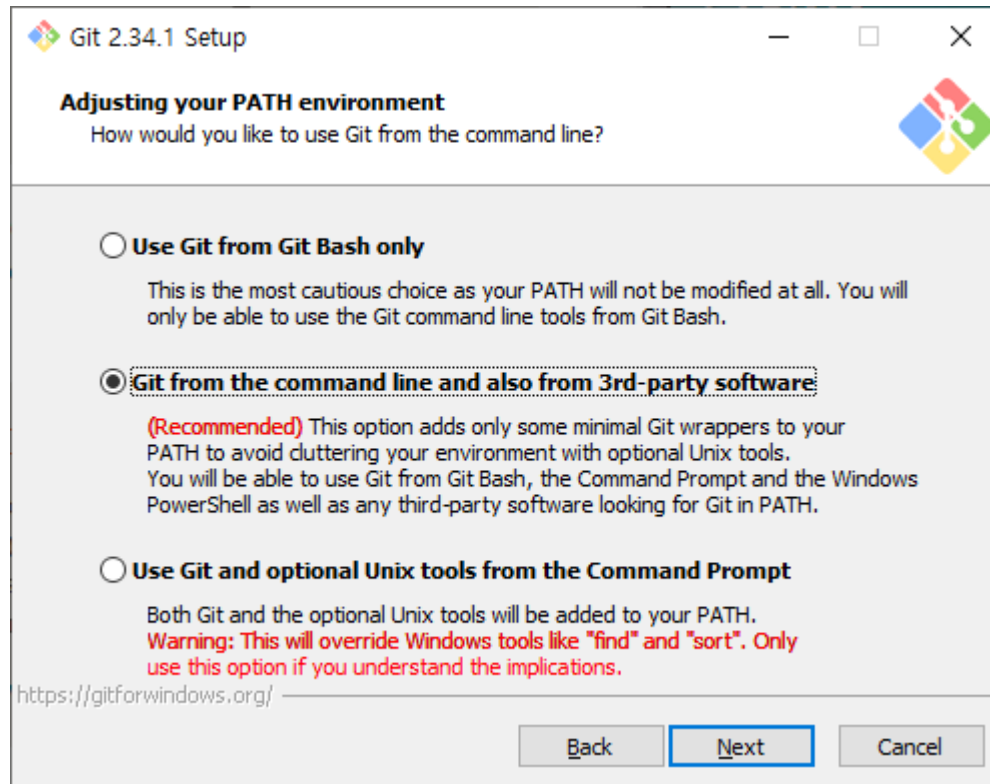




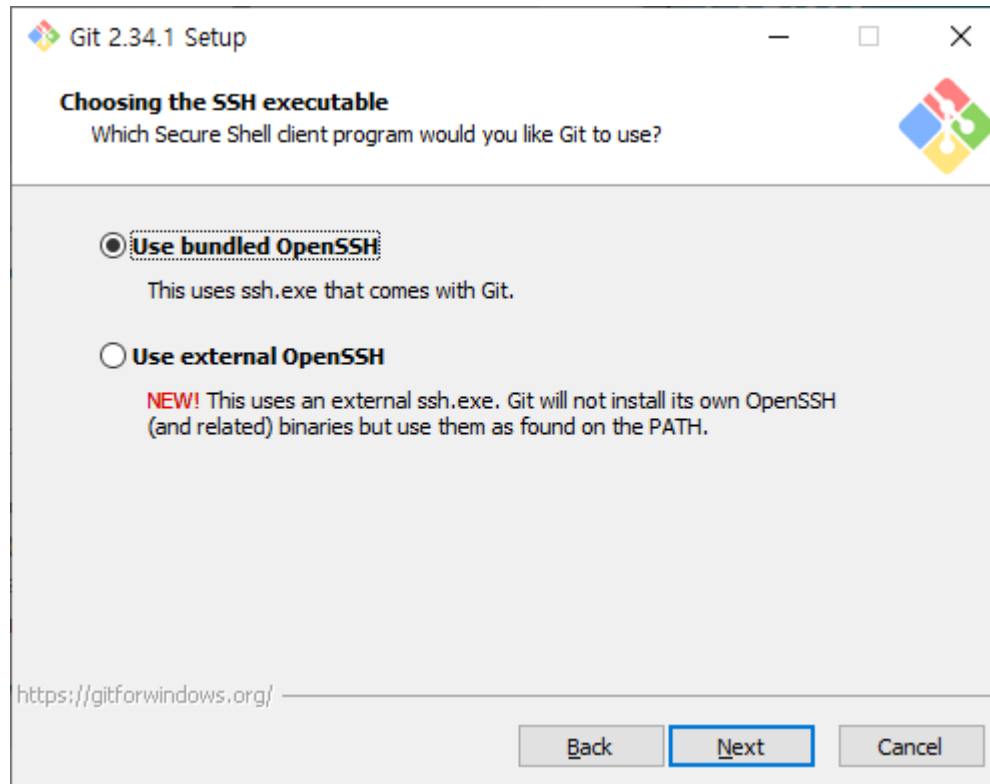
# 깃 설치하기 - Windows



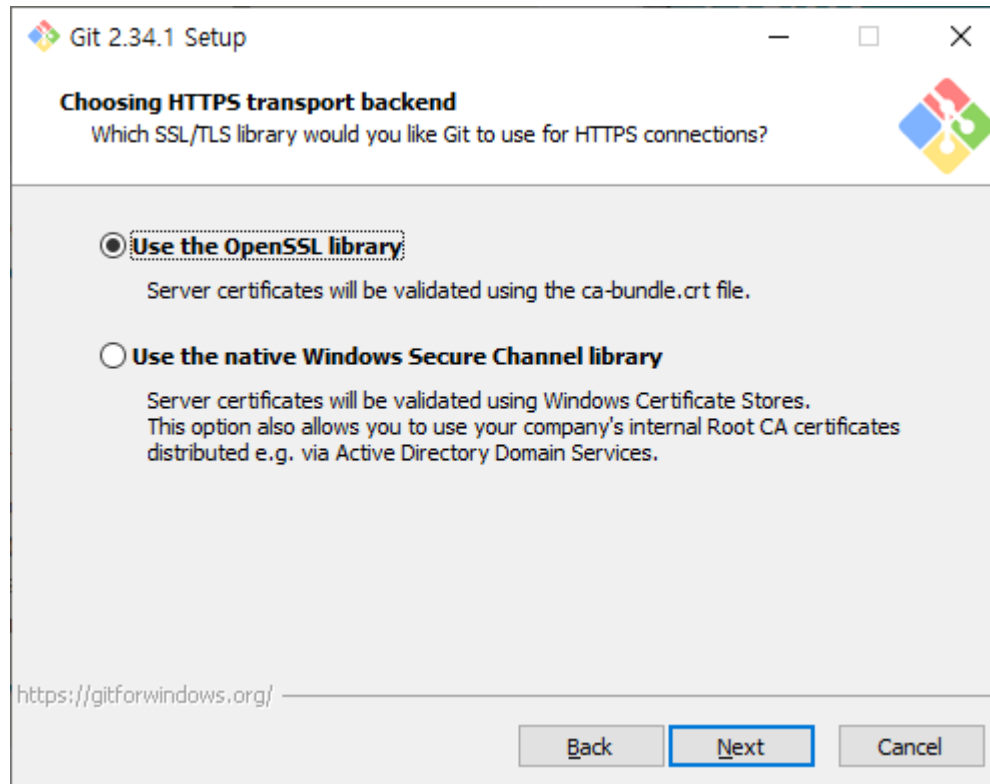
# 깃 설치하기 - Windows



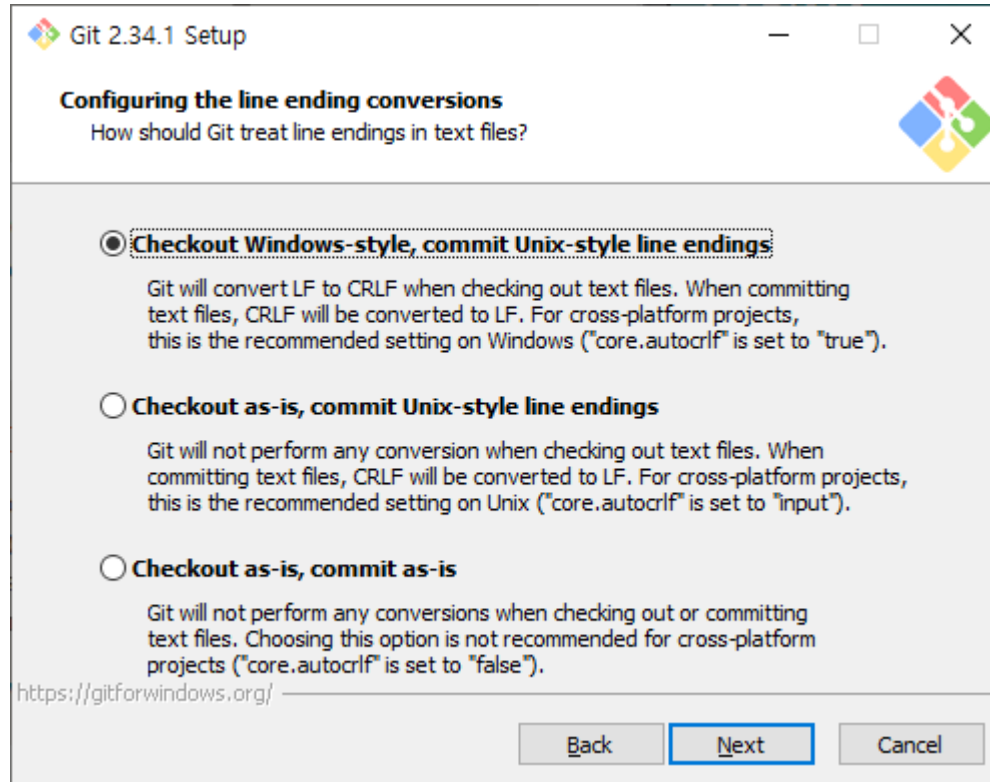
# 깃 설치하기 - Windows



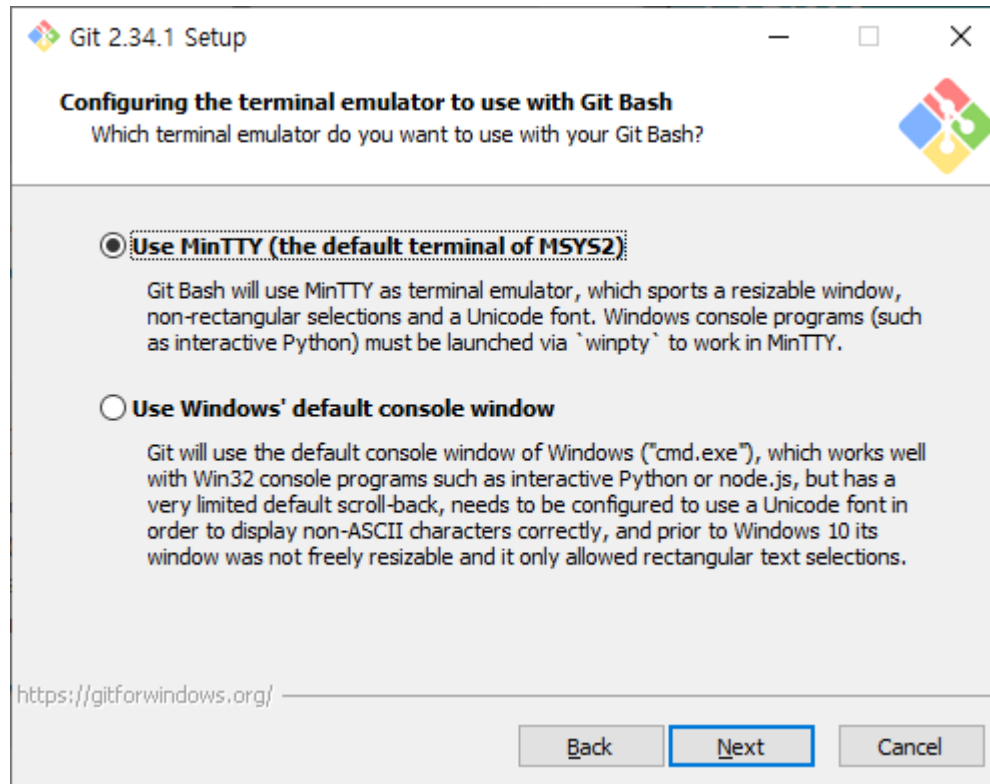
# 깃 설치하기 - Windows



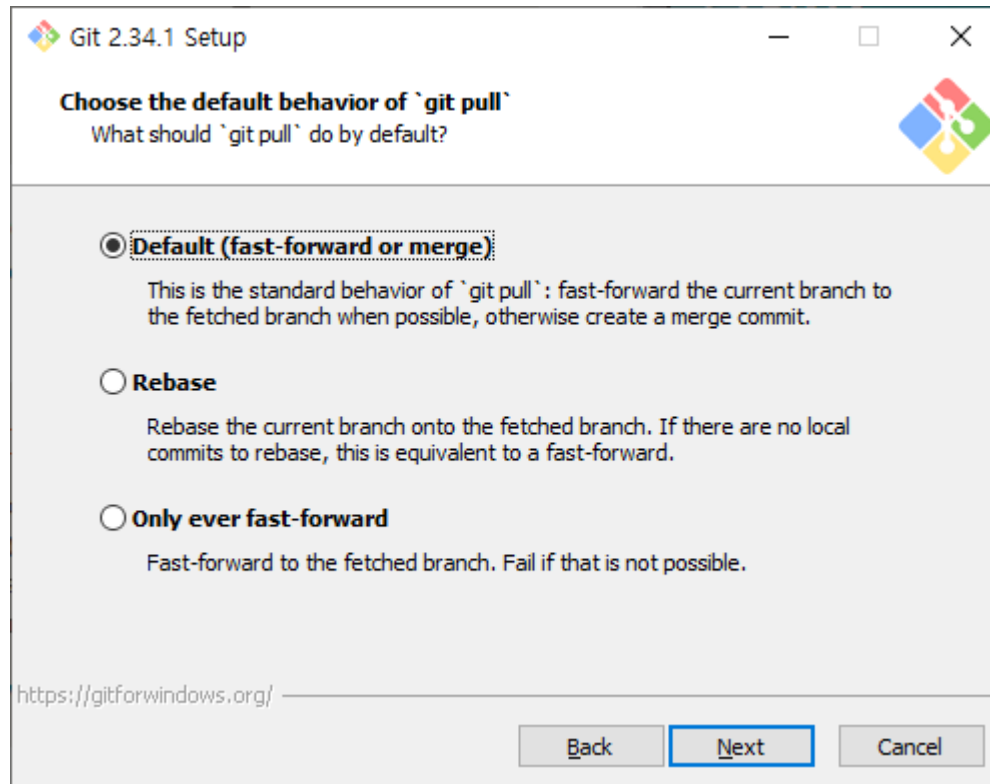
# 깃 설치하기 - Windows



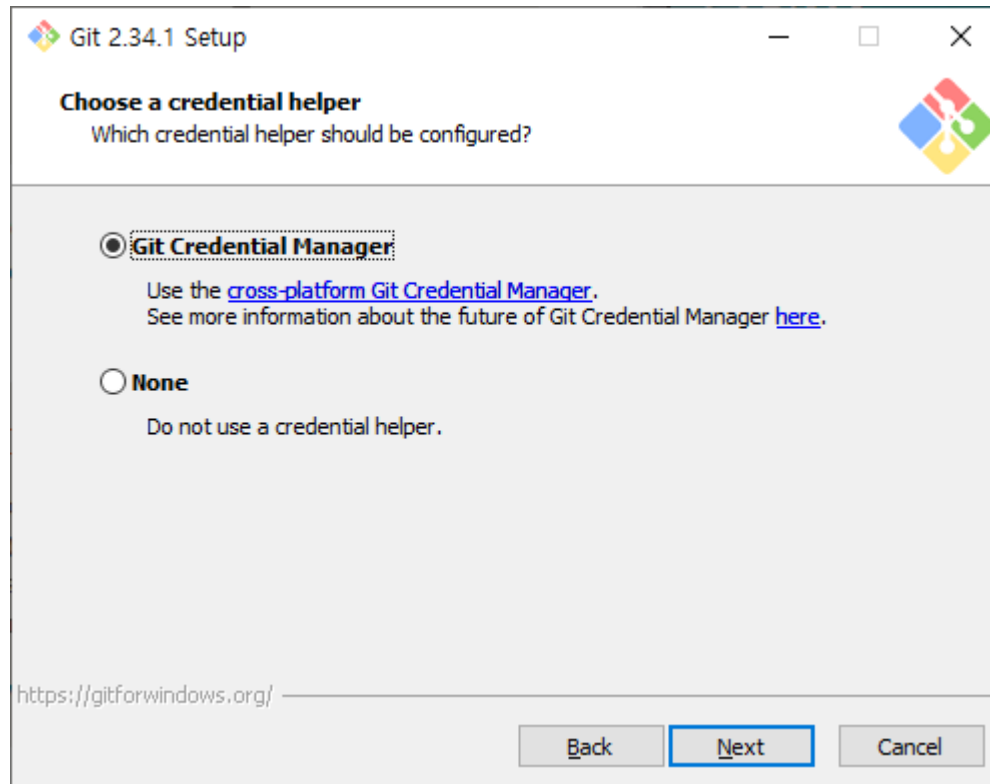
# 깃 설치하기 - Windows



# 깃 설치하기 - Windows

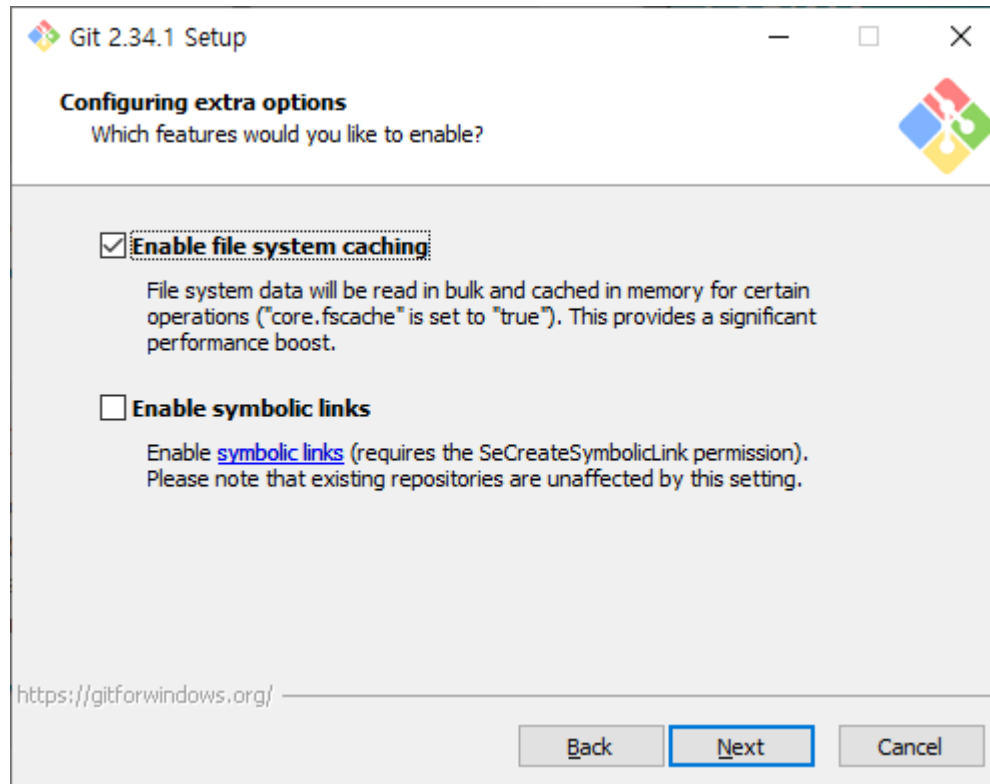


# 깃 설치하기 - Windows

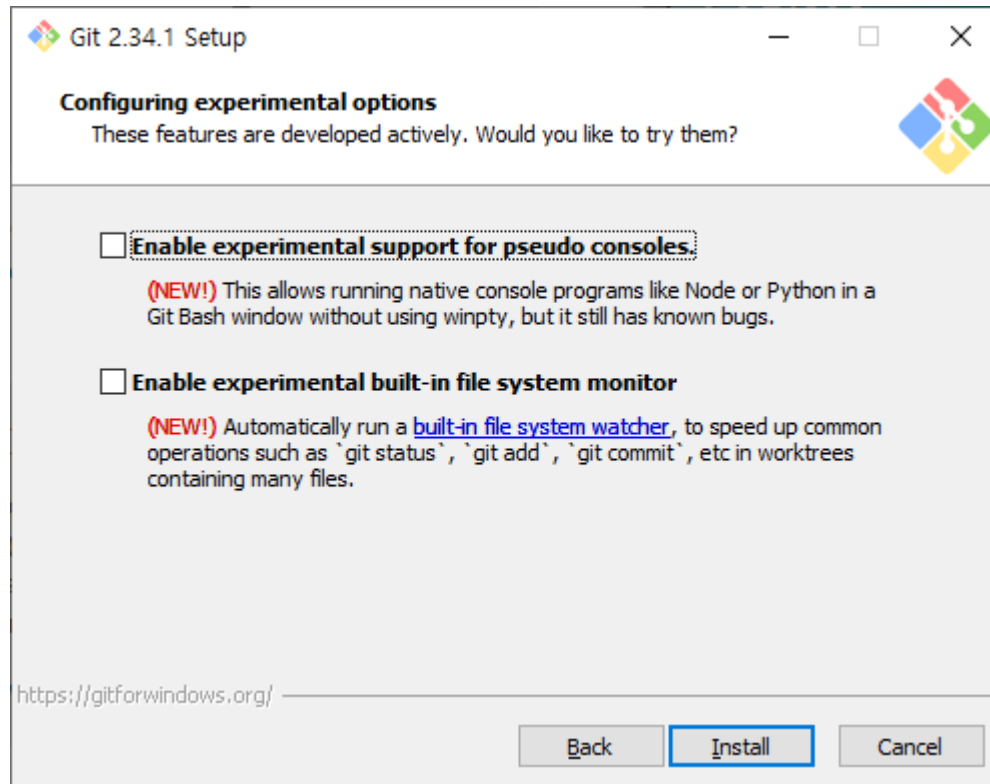




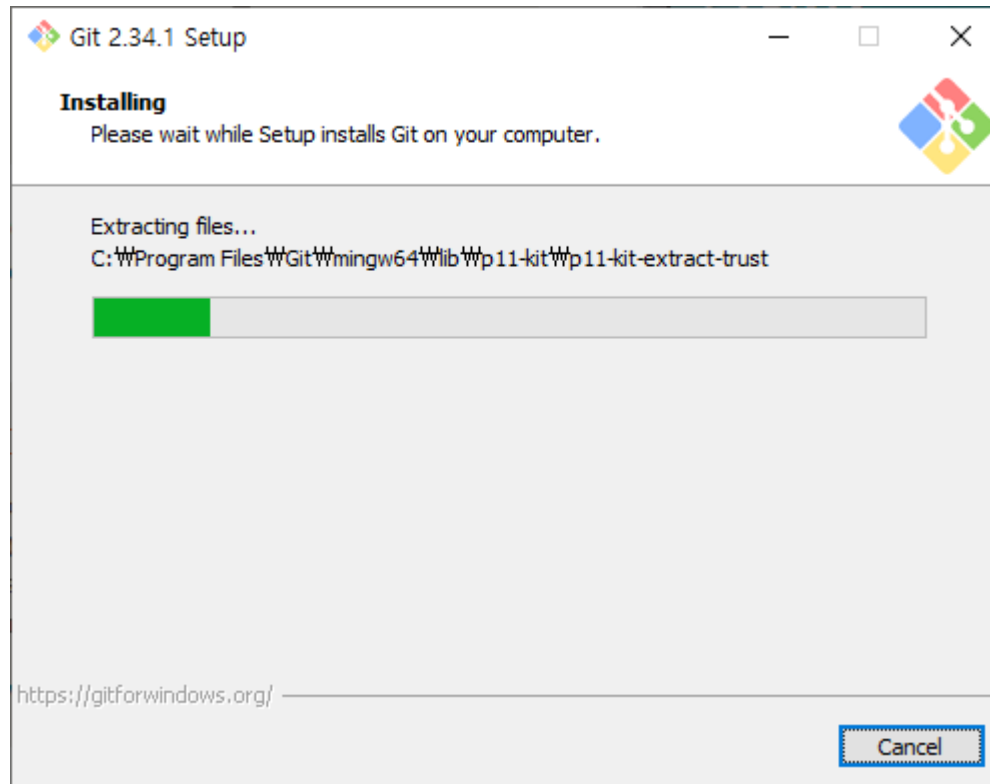
# 깃 설치하기 - Windows



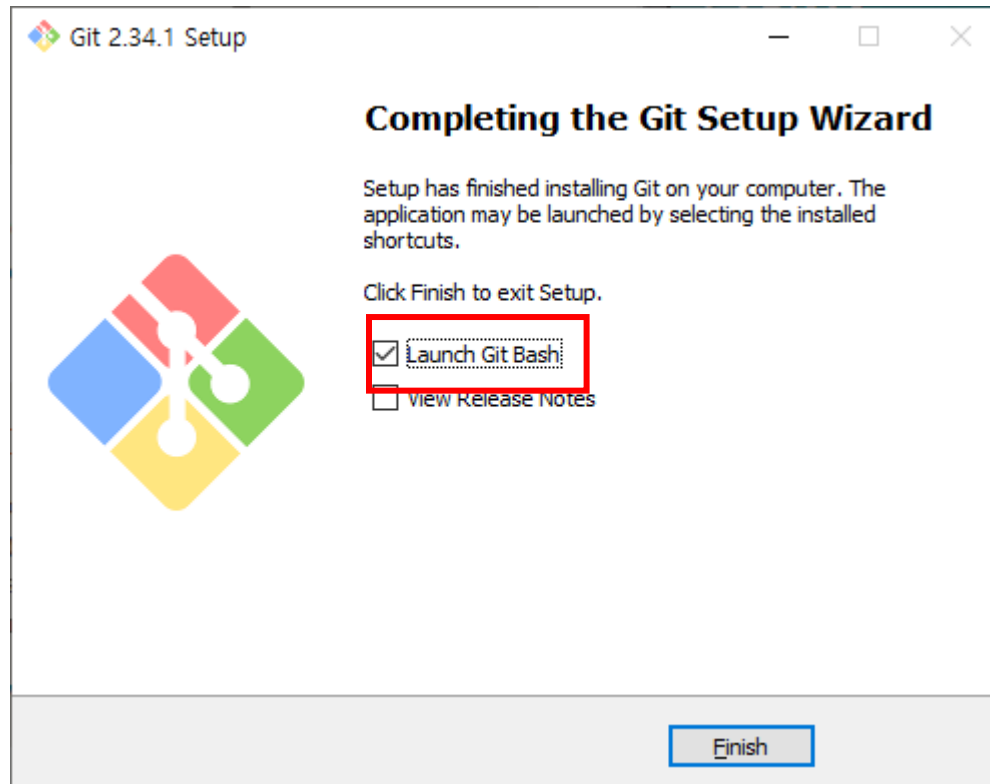
# 깃 설치하기 - Windows



# 깃 설치하기 - Windows

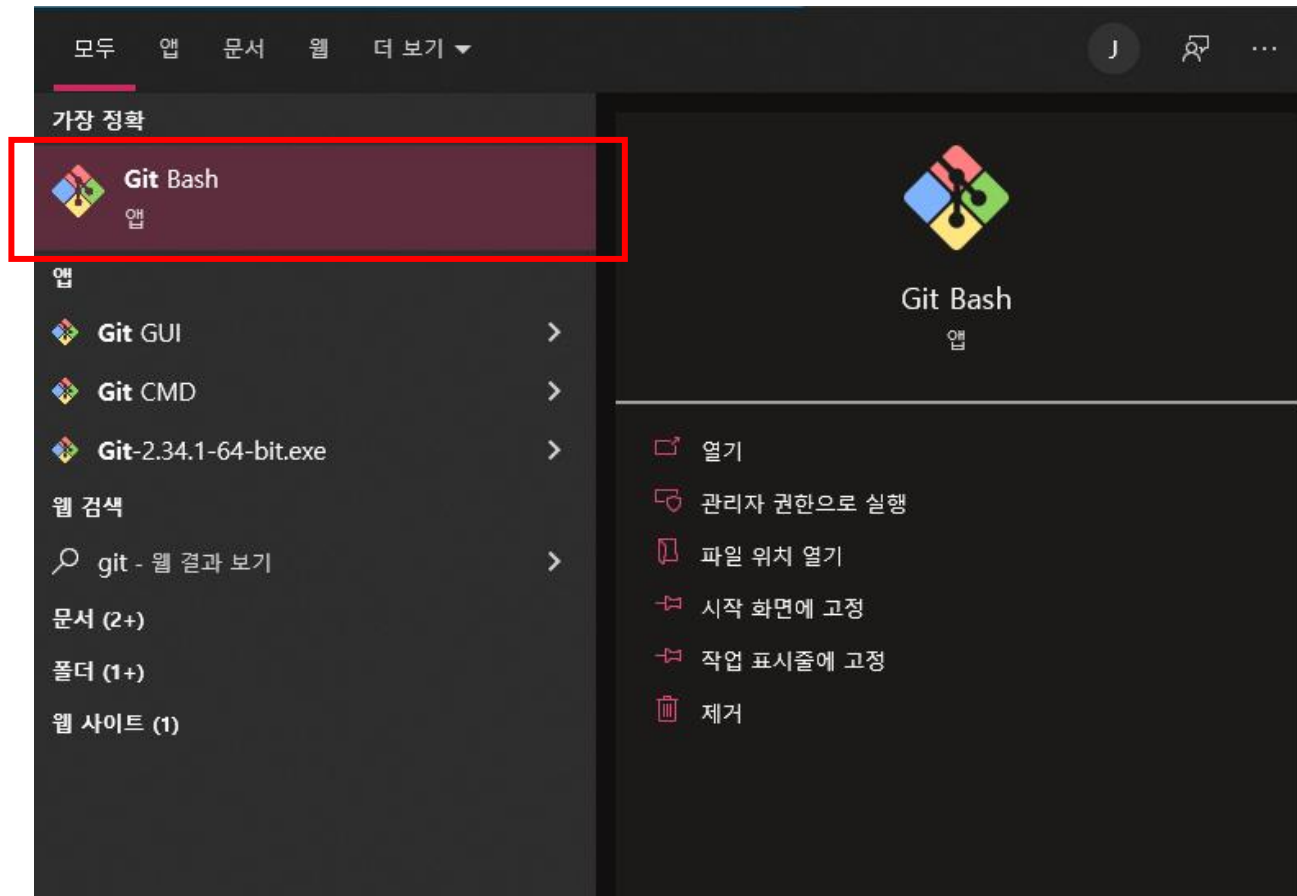


# 깃 설치하기 - Windows



# 깃 설치하기 - Windows

❖ 윈도우 검색 창에 git 입력 후, [Git Bash] 클릭



# 깃 설치하기 - macOS

## ❖ homebrew 설치하기

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

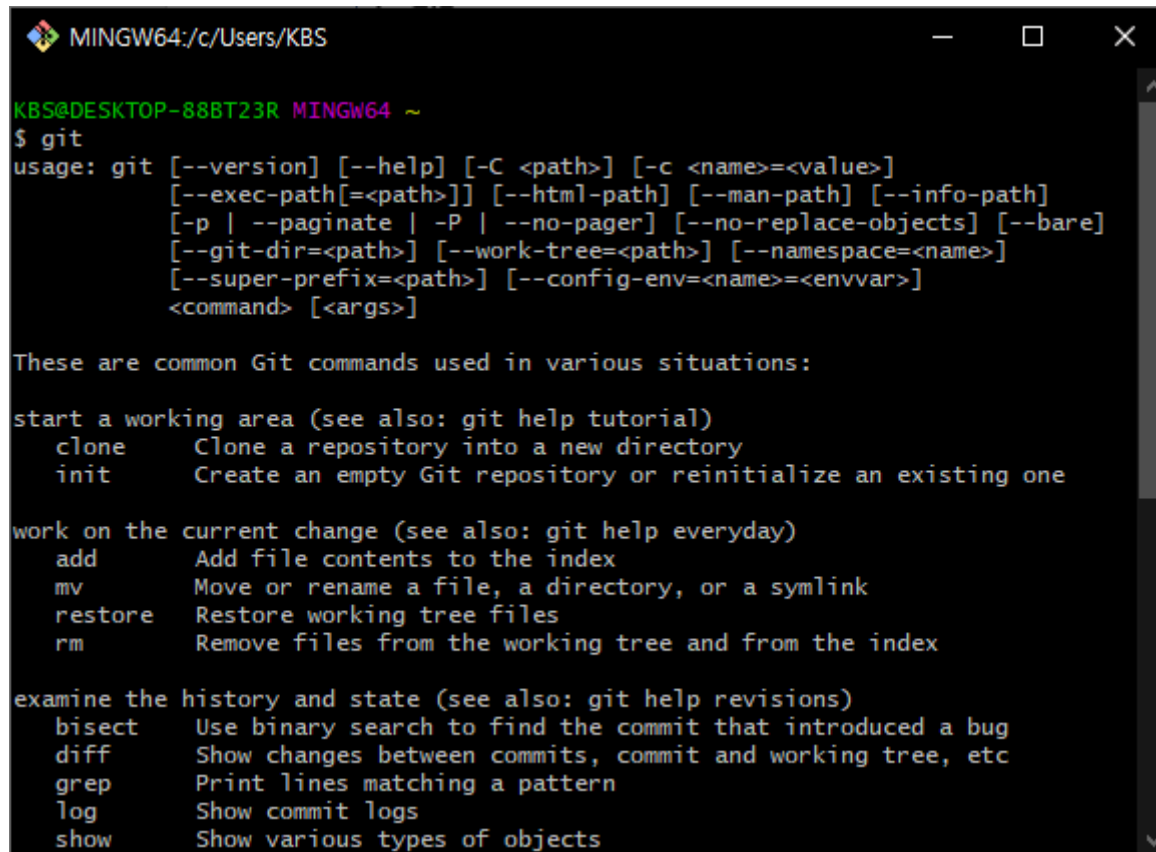
## ❖ 깃 설치하기

```
$ brew install git
```

# 깃 설치하기

## ❖ Git Bash 화면에 다음 명령어 입력

```
$ git
```



```
MINGW64:/c/Users/KBS
KBS@DESKTOP-88BT23R MINGW64 ~
$ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      [--super-prefix=<path>] [--config-env=<name>=<envvar>]
      <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
    clone      Clone a repository into a new directory
    init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
    add        Add file contents to the index
    mv         Move or rename a file, a directory, or a symlink
    restore    Restore working tree files
    rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
    bisect     Use binary search to find the commit that introduced a bug
    diff       Show changes between commits, commit and working tree, etc
    grep       Print lines matching a pattern
    log        Show commit logs
    show       Show various types of objects
```

# 환경 설정

## ❖ 사용자 정보 입력

- 사용자 이름

```
$ git config --global user.name "Sooa"
```

- 사용자 이메일

```
$ git config --global user.email "Sooa@gmail.com"
```





05

## 기본 리눅스 명령어

# 리눅스 명령어

## ❖ 현재 디렉터리 위치 경로

```
$ pwd
```



```
MINGW64:/c/Users/KBS  
KBS@DESKTOP-88BT23R MINGW64 ~  
$ pwd  
/c/Users/KBS
```

# 리눅스 명령어

## ❖ 현재 디렉터리에 안에 있는 파일 또는 디렉터리 확인

```
$ ls
```

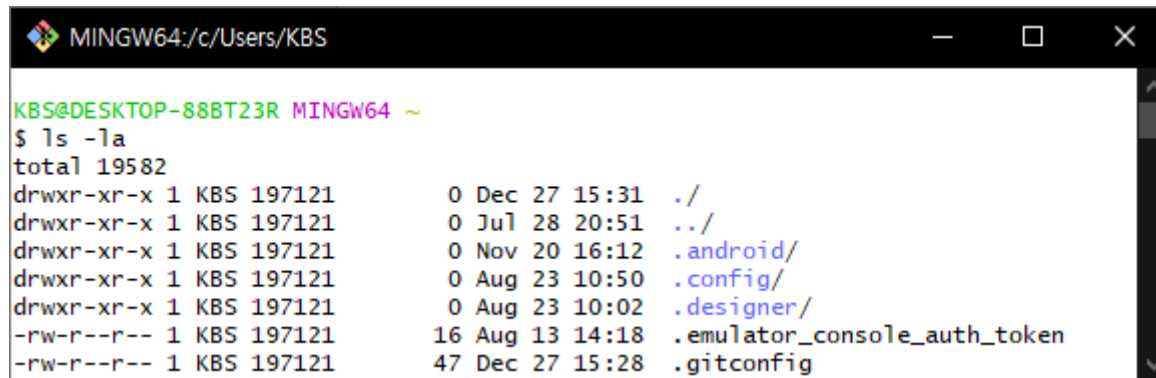


```
MINGW64:/c/Users/KBS
KBS@DESKTOP-88BT23R MINGW64 ~
$ ls
'3D Objects'/
AndroidStudioProjects/
AppData/
'Application Data'@
Contacts/
Cookies@
Desktop/
Documents/
```

# 리눅스 명령어

## ❖ 파일과 디렉터리 상세 정보 확인

```
$ ls -la
```



The screenshot shows a terminal window titled "MINGW64:/c/Users/KBS". The prompt is "KBS@DESKTOP-88BT23R MINGW64 ~". The command entered is "\$ ls -la". The output is as follows:

```
total 19582
drwxr-xr-x 1 KBS 197121      0 Dec 27 15:31 ./
drwxr-xr-x 1 KBS 197121      0 Jul 28 20:51 ../
drwxr-xr-x 1 KBS 197121      0 Nov 20 16:12 .android/
drwxr-xr-x 1 KBS 197121      0 Aug 23 10:50 .config/
drwxr-xr-x 1 KBS 197121      0 Aug 23 10:02 .designer/
-rw-r--r-- 1 KBS 197121 16 Aug 13 14:18 .emulator_console_auth_token
-rw-r--r-- 1 KBS 197121 47 Dec 27 15:28 .gitconfig
```

# 리눅스 명령어

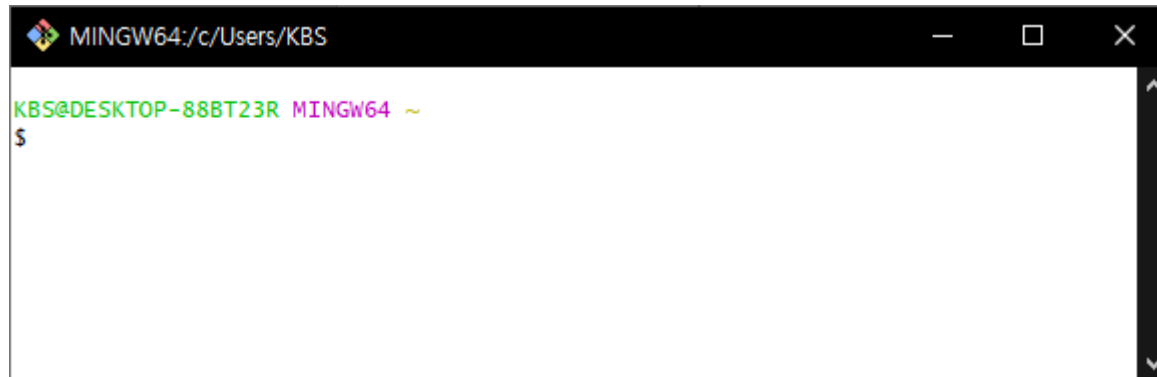
## ❖ ls 명령어 옵션

옵션	설명
-a	숨김 파일과 디렉터리를 함께 표시
-l	파일이나 디렉터리의 상세 정보를 함께 표시
-r	파일의 정렬 순서를 거꾸로 표시
-t	파일 작성 시간 순으로(내림차순) 표시

# 리눅스 명령어

## ❖ 화면 지우기

```
$ clear
```



# 리눅스 명령어

## ❖ 디렉터리 이동

```
$ cd
```

- 현재 위치에서 상위 디렉터리로 이동

```
$ cd ..
```



# 리눅스 명령어

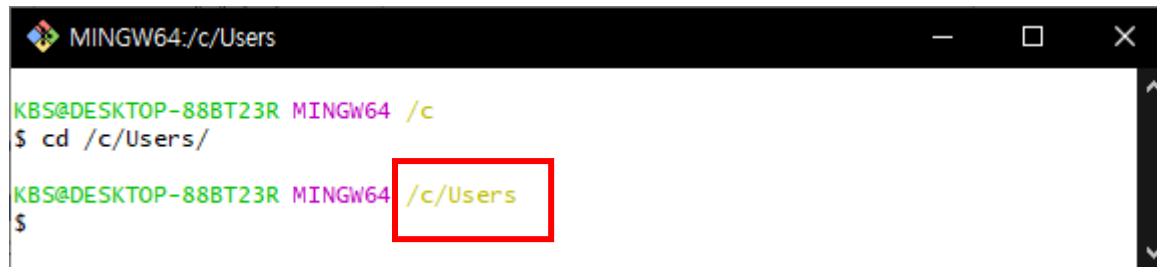
## ❖ 디렉터리 이동

- 현재 위치에서 하위 디렉터리로 이동

```
$ cd 하위_디렉터리_이름
```

- /c/Users 디렉터리로 이동

```
$ cd /c/Users
```



The screenshot shows a Windows command prompt window titled "MINGW64:/c/Users". The prompt is "KBS@DESKTOP-88BT23R MINGW64 /c". The user has entered the command "\$ cd /c/Users/". The prompt has changed to "KBS@DESKTOP-88BT23R MINGW64 /c/Users", and the path "/c/Users" is highlighted with a red box. The prompt is now "\$".

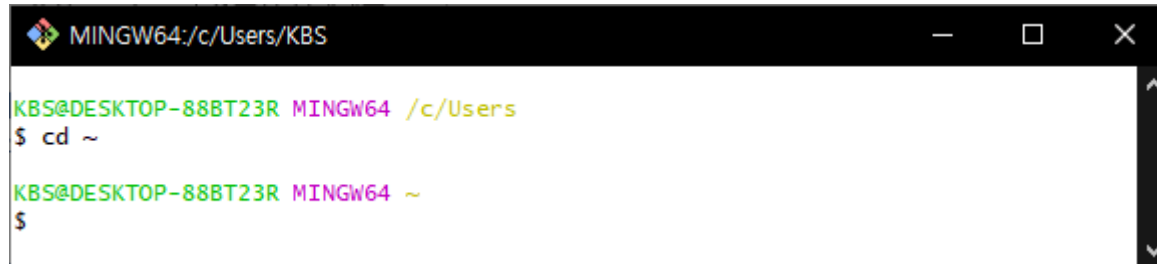


# 리눅스 명령어

## ❖ 디렉터리 이동

- 홈 디렉터리로 이동

```
$ cd ~
```



```
MINGW64:/c/Users/KBS  
KBS@DESKTOP-88BT23R MINGW64 /c/Users  
$ cd ~  
KBS@DESKTOP-88BT23R MINGW64 ~  
$
```

# 리눅스 명령어

## ❖ cd 명령어 옵션

옵션	설명
~	<ul style="list-style-type: none"><li>▪ 현재 사용자의 홈 디렉터리</li><li>▪ c/Users/계정이름</li></ul>
./	현재 사용자가 작업 중인 디렉터리
../	현재 디렉터리의 상위 디렉터리

# 리눅스 명령어

## ❖ 디렉터리 생성

```
$ mkdir 생성할_디렉터리_이름
```

- **홈 디렉터리** 안에 있는 **Documents 디렉터리**에 **test 디렉터리** 생성

```
$ cd ~/Documents  
$ mkdir test
```

# 리눅스 명령어

## ❖ 디렉터리 생성

- test 디렉터리 생성 확인

```
$ ls
```



```
MINGW64:/c/Users/KBS/Documents
KBS@DESKTOP-88BT23R MINGW64 ~/Documents
$ ls
Embarcadero/  'My Videos'@  desktop.ini
HeidiSQL/    'Python Scripts'/  test/
Mobizen/     UltraVNC/        '카카오톡 지정 Office 서식 파일'/'
'My Music'@  'Visual Studio 2019'/  '카카오톡 받은 파일'/'
'My Pictures'@ aa.json
KBS@DESKTOP-88BT23R MINGW64 ~/Documents
$
```

# 리눅스 명령어

## ❖ 디렉터리 삭제

```
$ rm -r 삭제할_디렉터리_이름
```

- 홈 디렉터리 안에 있는 Documents 디렉터리에 test 디렉터리 삭제

```
$ rm -r test  
$ ls
```



```
MINGW64:/c/Users/KBS/Documents  
KBS@DESKTOP-88BT23R MINGW64 ~/Documents  
$ rm -r test  
KBS@DESKTOP-88BT23R MINGW64 ~/Documents  
$ ls  
Embarcadero/  'My Videos'@  desktop.ini  
HeidiSQL/    'Python Scripts'/  '사용자 지정 Office 서식 파일'/  
Mobizen/     UltraVNC/        '카카오톡 받은 파일'/  
'My Music'@  'Visual Studio 2019'/  
'My Pictures'@ aa.json
```



06

## 빔(Vim) 사용하기

# 빔(Vim) 사용하기

## ❖ 빔(Vim)이란?

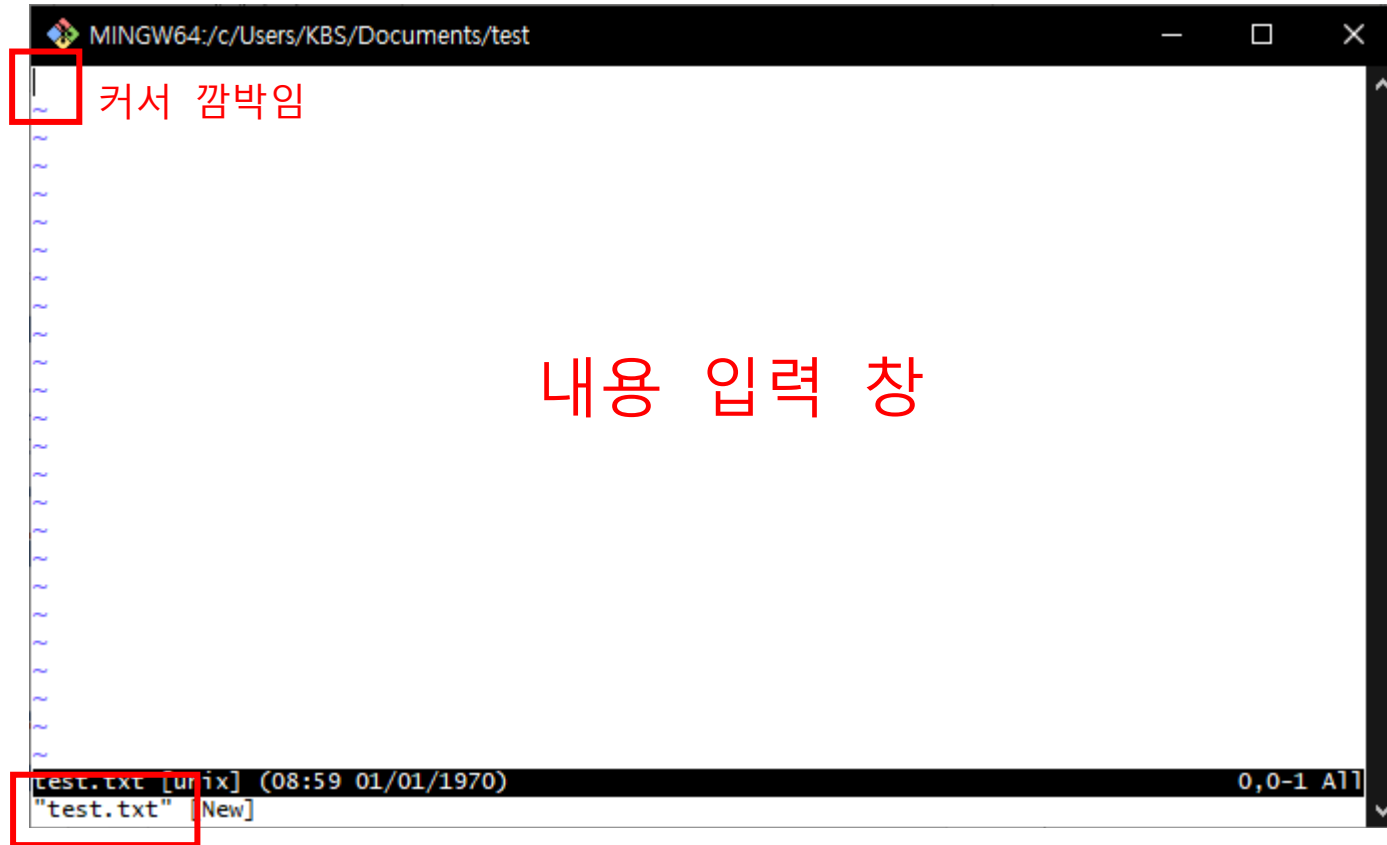
- 터미널에서 사용할 수 있는 대표적인 문서 편집기
- 홈디렉터리/Documents에 test 디렉터리 생성 후 이동

```
$ cd ~/Documents  
$ mkdir test  
$ cd test
```

- 현재 디렉터리(test)에 test.txt 파일 생성

```
$ vim test.txt
```

# 빔(Vim) 사용하기

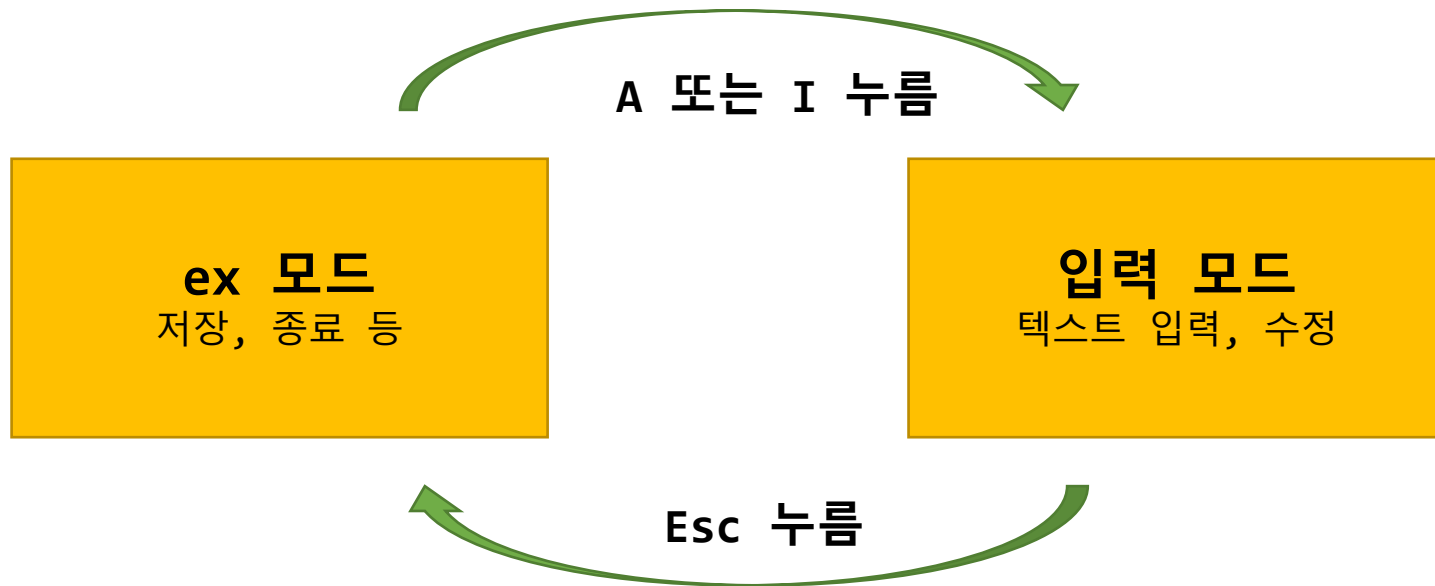


현재 열려 있는 파일 이름



# 빔(Vim) 사용하기

## ❖ 입력 모드와 ex 모드



# 빔(Vim) 사용하기

## ❖ 텍스트 입력하기

MINGW64: c:/Users/KBS/Documents/test

안녕하세요!  
vim 편집기를 사용하고 있습니다.

test.txt[+] [un] x (08:59 01/01/1970) 2,44-32 All  
-- INSERT --

입력 모드

# 빔(Vim) 사용하기

## ❖ 내용 저장하기

- ex 모드로 돌아가야 함
  - Esc 키를 누름

A screenshot of a Windows command prompt window titled "MINGW64: c:/Users/KBS/Documents/test". The window shows a Vim editor interface. The first line contains the Korean text "안녕하세요!" followed by "vim 편집기를 사용하고 있습니다.". Below this are several tilde (~) characters representing blank lines. At the bottom, the status bar displays "test.txt[+] [unix] (08:59 01/01/1970)" on the left and "2,43-31 All" on the right. A red dashed rectangle highlights the bottom-left corner of the window, specifically the area around the status bar and the first few lines of the file.

# 빔(Vim) 사용하기

## ❖ 내용 저장하기

- 콜론(:)을 입력하면 "INSERT"가 있던 자리에 텍스트를 입력할 수 있음
- :wq 입력 후, Enter 키 누름

A screenshot of a Windows command prompt window titled "MINGW64: c:/Users/KBS/Documents/test". The window shows the output of running "vim test.txt". The first two lines are Korean text: "안녕하세요 !" and "vim 편집기를 사용하고 있습니다.". Below these are several tilde (~) characters representing blank lines. At the bottom, the status bar displays "test.txt[+] [unix] (08:59 01/01/1970)" and "2,43-31 All". A red box highlights the ":wq" command entered at the prompt.

# 빔(Vim) 사용하기

## ❖ ex 모드 명령어

옵션	설명
:w 또는 :write	편집 중이던 문서를 저장
:q 또는 :quit	편집기를 종료
:wq	편집 중이던 문서를 저장하고 종료
:q!	문서를 저장하지 않고 편집기를 종료

# 빔(Vim) 사용하기

## ❖ 텍스트 문서 내용 확인하기

```
$ cat 파일이름
```

- test.txt 파일 내용 확인

```
$ cat test.txt
```



```
MINGW64:/c/Users/KBS/Documents/test
KBS@DESKTOP-88BT23R MINGW64 ~/Documents/test
$ cat test.txt
안녕하세요!
vim 편집기를 사용하고 있습니다.
```

**THANK 😊 YOU**