

Open Source Software

git 소프트웨어 설치와 Bash 터미널

About..

컴퓨터소프트웨어공학과

김 원 일



git 다운로드 - 1

- git 설치 프로그램 다운로드

- <http://git-scm.com/> 사이트 접속 및 최신 버전 다운로드



The screenshot shows the Git website homepage. At the top left is the Git logo and the tagline "--local-branching-on-the-cheap". To the right is a search bar with the placeholder text "Type / to search entire site...". Below the header, there are two paragraphs of text describing Git as a free and open source distributed version control system, designed to handle projects of all sizes with speed and efficiency. It also mentions that Git is easy to learn, has a tiny footprint, and lightning fast performance, outclassing other SCM tools like Subversion, CVS, Perforce, and ClearCase. To the right of the text is a diagram illustrating branching and merging with stacks of papers and colored lines. Below the text and diagram are four sections: "About" (advantages of Git), "Documentation" (command reference, Pro Git book, videos), "Downloads" (GUI clients and binary releases), and "Community" (bug reporting, mailing list, chat). On the right side, there is a monitor displaying the "Latest source Release 2.48.1" with a red box around the "Download for Windows" button.

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

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

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 branching**, convenient **staging areas**, and **multiple workflows**.

About
The advantages of Git compared to other source control systems.

Documentation
Command reference pages, Pro Git book content, videos and other material.

Downloads
GUI clients and binary releases for all major platforms.

Community
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.48.1
[Release Notes \(2025-01-13\)](#)
Download for Windows



• Architecture 확인

- Windows 10/11 : 탐색기의 "내 PC"의 오른쪽 버튼 메뉴의 "속성"

장치 사양

장치 이름	YP12624115
프로세서	Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz 3.00 GHz
설치된 RAM	16.0GB
장치 ID	36135D8A-6B44-4206-B307-FE81A33FDE7D
제품 ID	00328-10000-00001-AA746
시스템 종류	64비트 운영 체제, x64 기반 프로세서
펜 및 터치	이 디스플레이에 사용할 수 있는 펜 또는 터치식 입력이 없습니다.

PC의 이름 바꾸기

복사 ^

장치 사양

장치 이름	
프로세서	13th Gen Intel(R) Core(TM) i5-13400F 2.50 GHz
설치된 RAM	32.0GB(31.8GB 사용 가능)
장치 ID	
제품 ID	
시스템 종류	64비트 운영 체제, x64 기반 프로세서
펜 및 터치	이 디스플레이에 사용할 수 있는 펜 또는 터치식 입력이 없습니다.



• 버전 선택

- 32bit/64bit 중 시스템에 맞는 버전을 선택하여 다운로드
- "Standalone Installer"에서 CPU에 맞게 다운로드
- "Portable"은 프로그램 압축 형식으로 환경 설정의 수동 진행 필요

The screenshot shows the Git website's 'Download for Windows' page. The header includes the Git logo and the tagline '--local-branching-on-the-cheap'. A search bar is located in the top right. The left sidebar contains links for 'About', 'Documentation', 'Downloads' (highlighted), 'GUI Clients', 'Logos', and 'Community'. The main content area is titled 'Download for Windows' and features a call to action to download the latest (2.48.1) 64-bit version of Git for Windows, noting it is the most recent maintained build released 21 days ago. Below this, there are sections for 'Other Git for Windows downloads' including 'Standalone Installer' (with links for 32-bit and 64-bit Setup) and 'Portable ("thumbdrive edition")' (with links for 32-bit and 64-bit Portable). A footer note mentions the 'Pro Git book' by Scott Chacon and Ben Straub is available online for free, with dead tree versions on Amazon.com.

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

Search: Type / to 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 (2.48.1) 64-bit version of **Git for Windows**. This is the most recent [maintained build](#). It was released **21 days ago**, on 2025-02-13.

Other Git for Windows downloads

Standalone Installer

[32-bit Git for Windows Setup.](#)

[64-bit Git for Windows Setup.](#)

Portable ("thumbdrive edition")

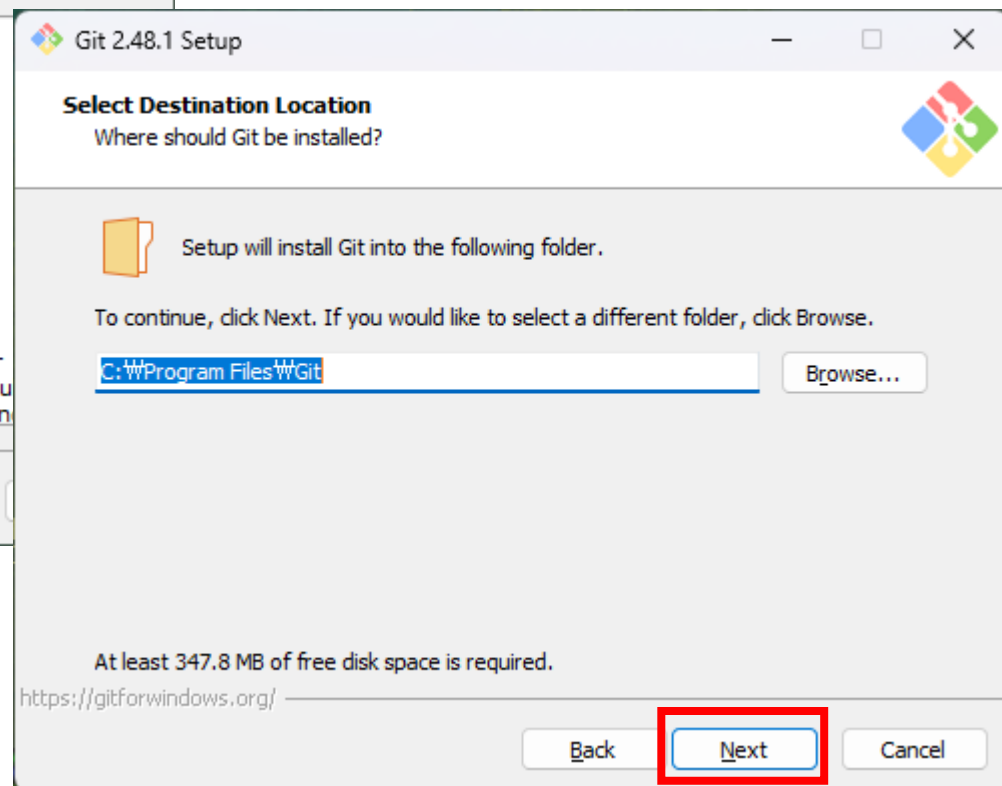
[32-bit Git for Windows Portable.](#)

[64-bit Git for Windows Portable.](#)



git 설치 - 1

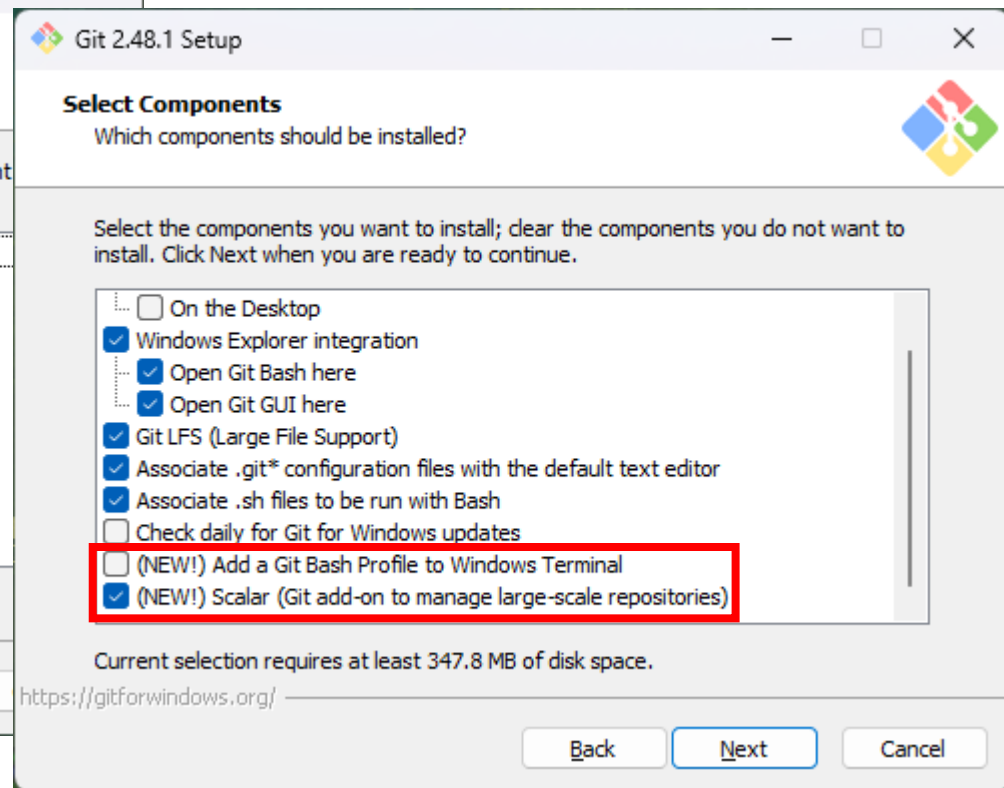
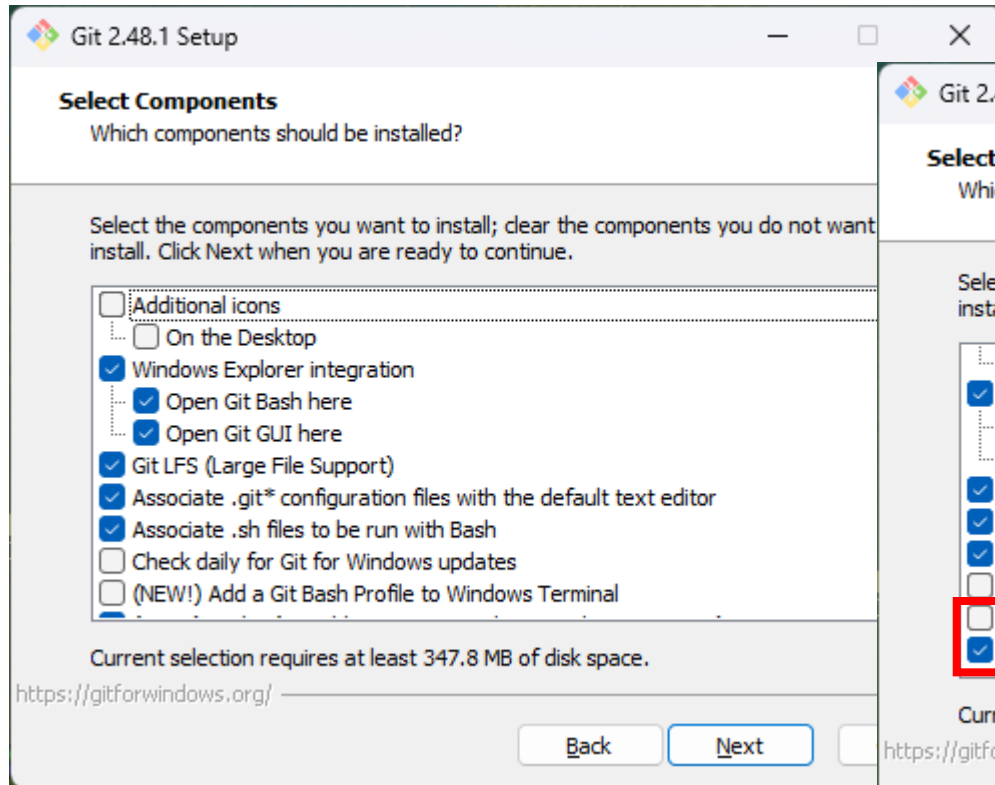
• 라이선스 동의와 설치 경로 설정





git 설치 - 2

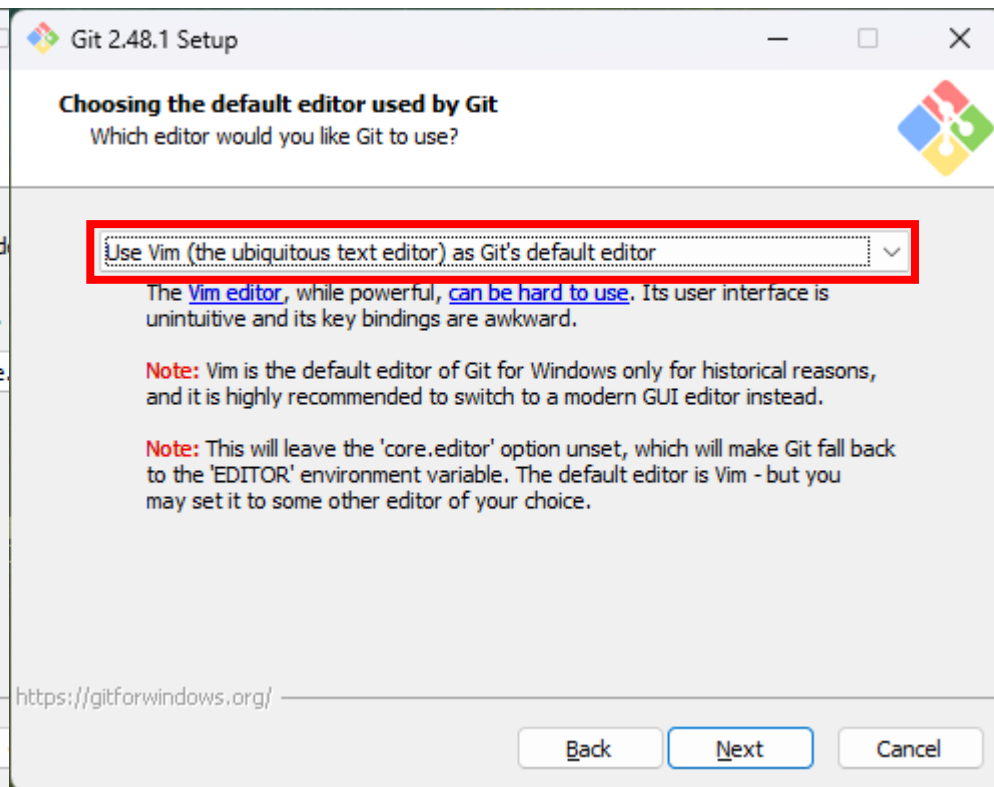
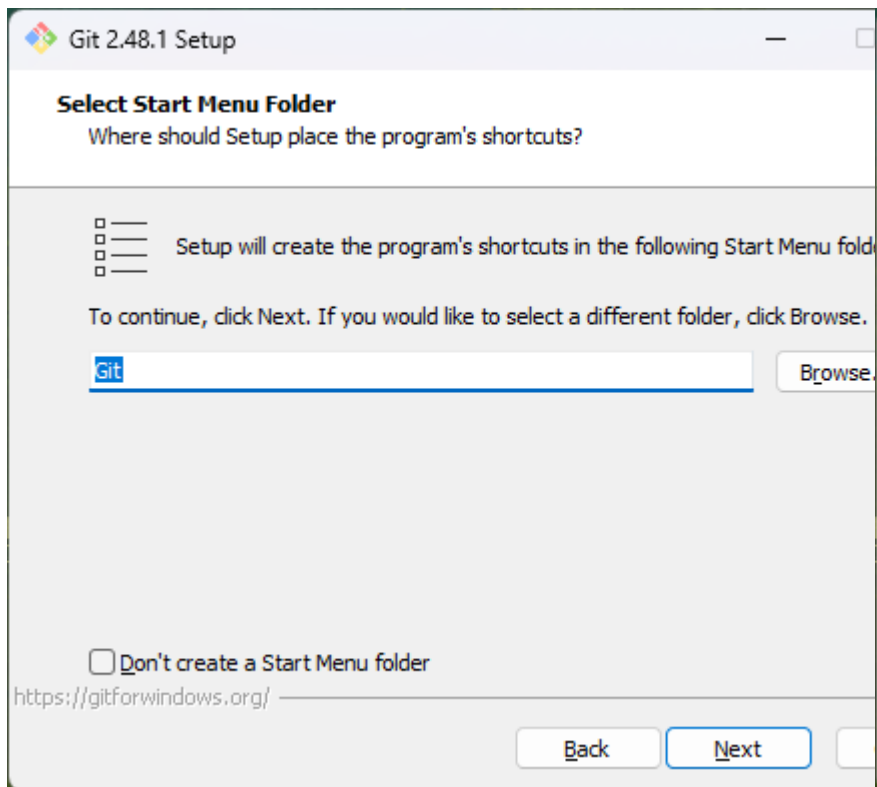
- 설치 도구와 시작 메뉴 이름 설정
 - 설치 도구는 선택하거나 그대로 진행
 - (NEW!)로 표시된 항목은 테스트 기능





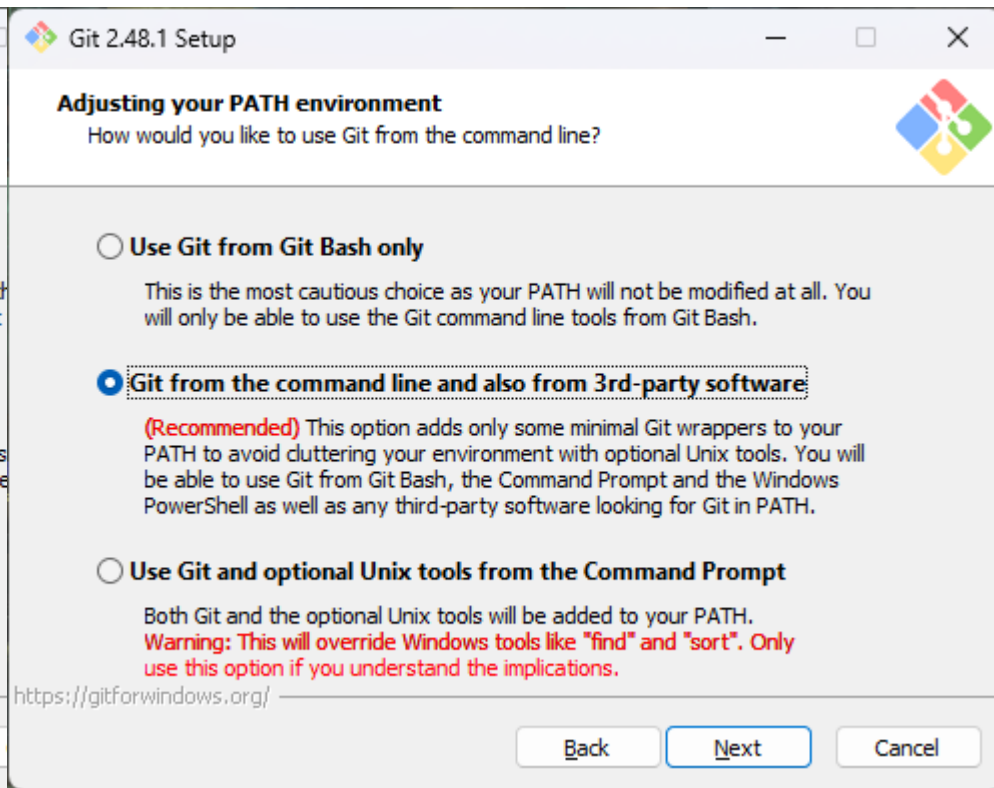
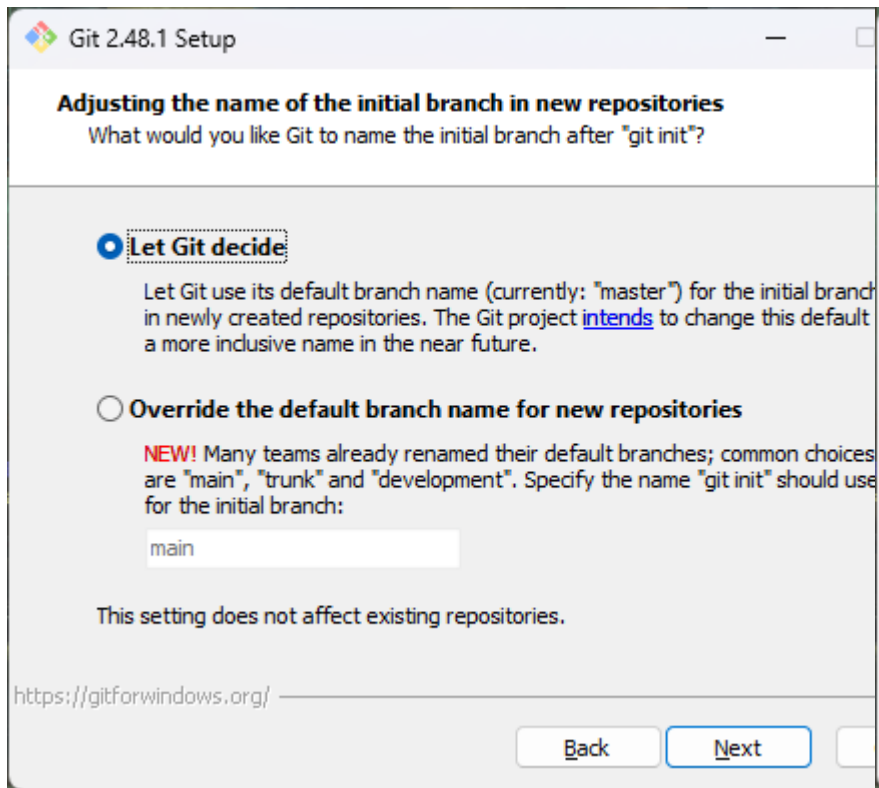
git 설치 - 3

- 문서 편집기와 브랜치 처리
 - 프로그램 폴더 생성 여부 설정
 - Linux 수업과 연계를 위해 Vim을 편집기로 설정
 - Vim 편집기를 사용하기 위한 실습을 추후 진행 예정



git 설치 - 4

- 초기 브랜치 이름 설정과 환경 설정
 - 저장소 초기 이름에 대한 설정
 - 보통 그대로 사용하지만, 필요한 경우 변경하여 사용할 수 있음
 - git 명령어 경로 환경 설정으로 윈도우 프로그램에서도 사용 가능

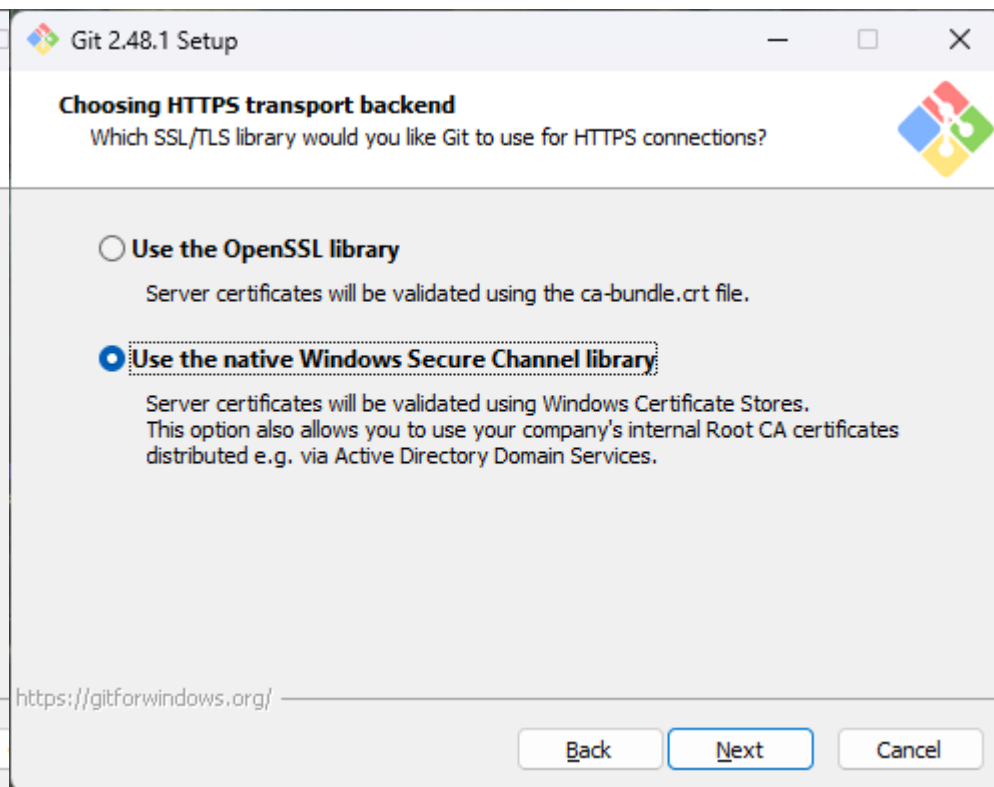
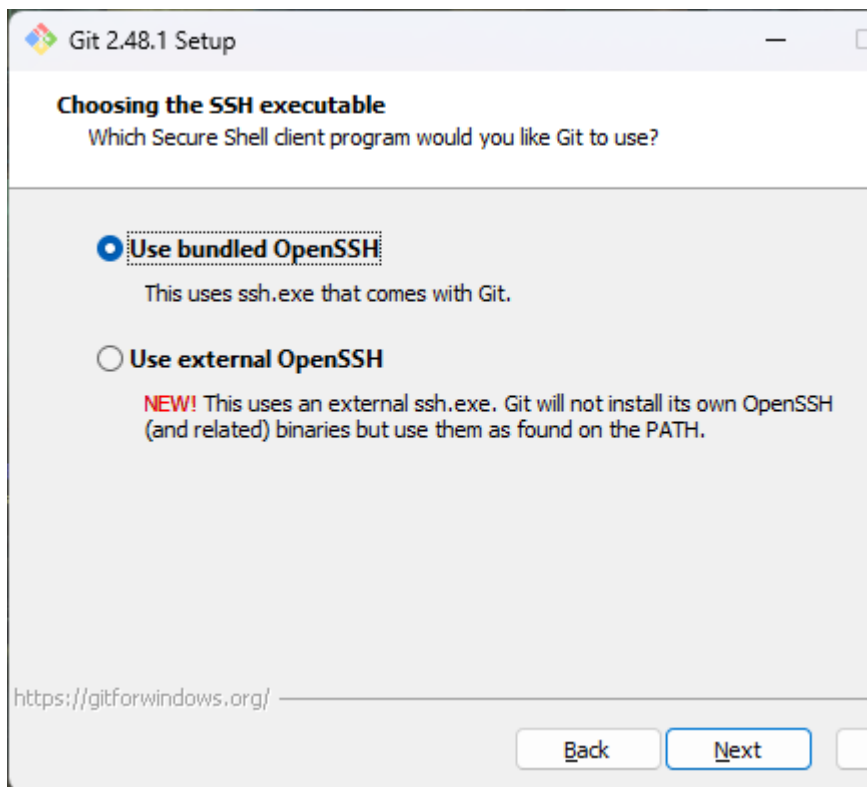




git 설치 - 5

• 소스코드 처리 방식 설정

- SSH(Secure SHell) : 보안 강화 통신 채널 프로그램 선택
- SSL/TLS 라이브러리 선택
- 두 옵션은 별도 프로그램, 인증서를 가지고 있으면 지정해서 사용 가능

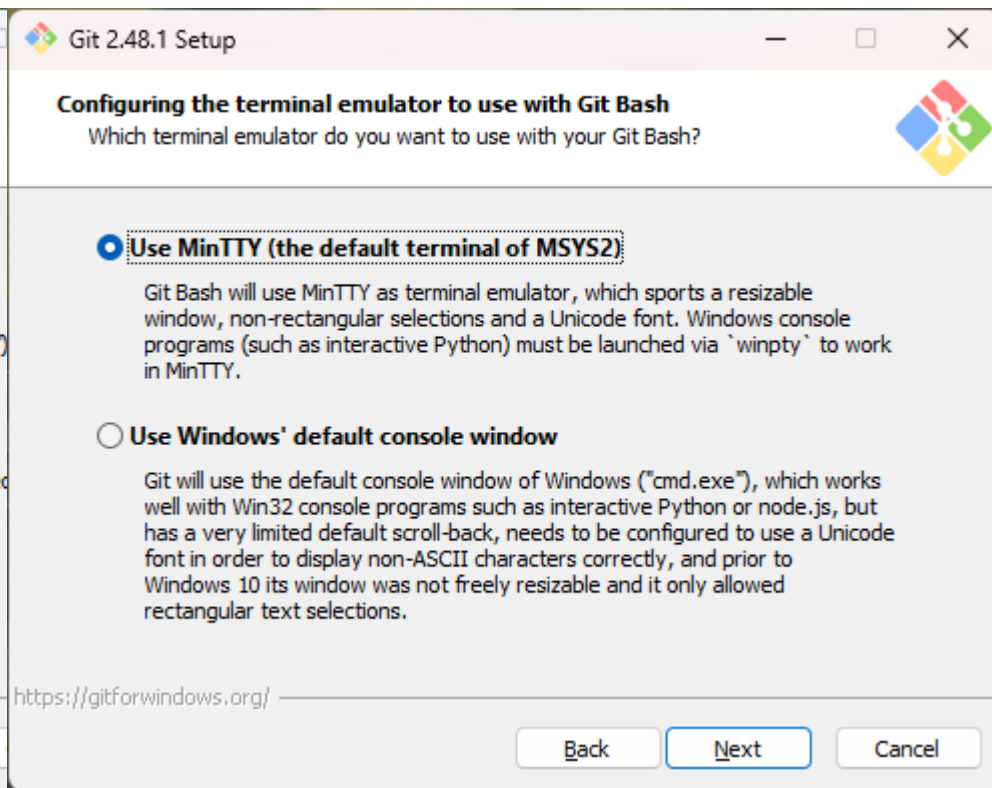
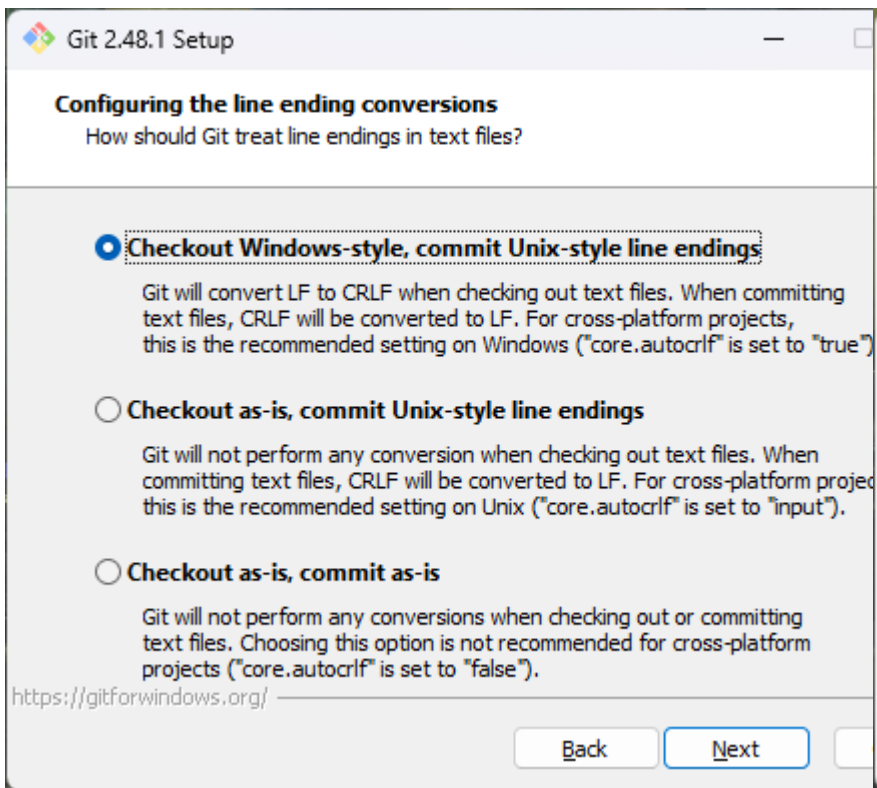




git 설치 – 6

• 라인 끝에 대한 처리 방식과 터미널 설정

- Windows 또는 UNIX 스타일로 지정
- git 제공 터미널 사용 또는 윈도우 터미널 사용 여부 선택

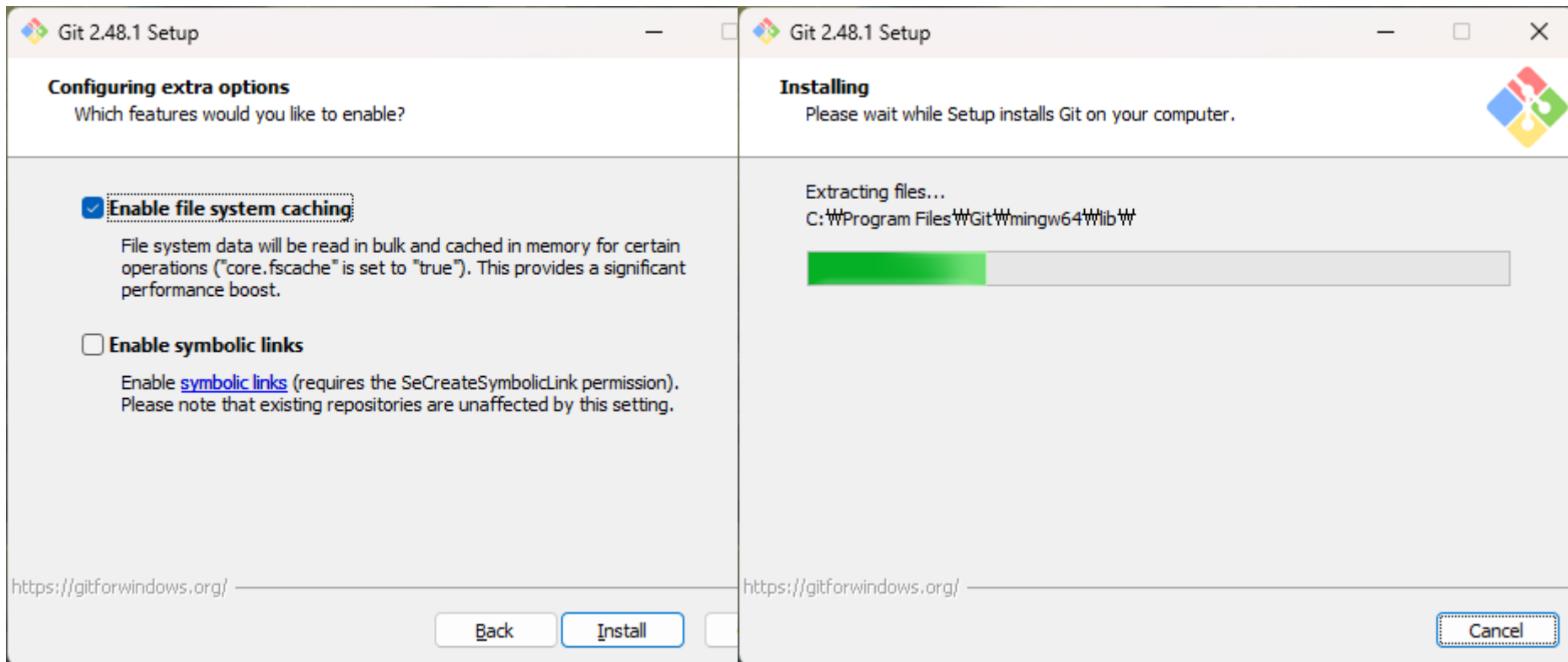




git 설치 - 7

• 추가 옵션과 설치

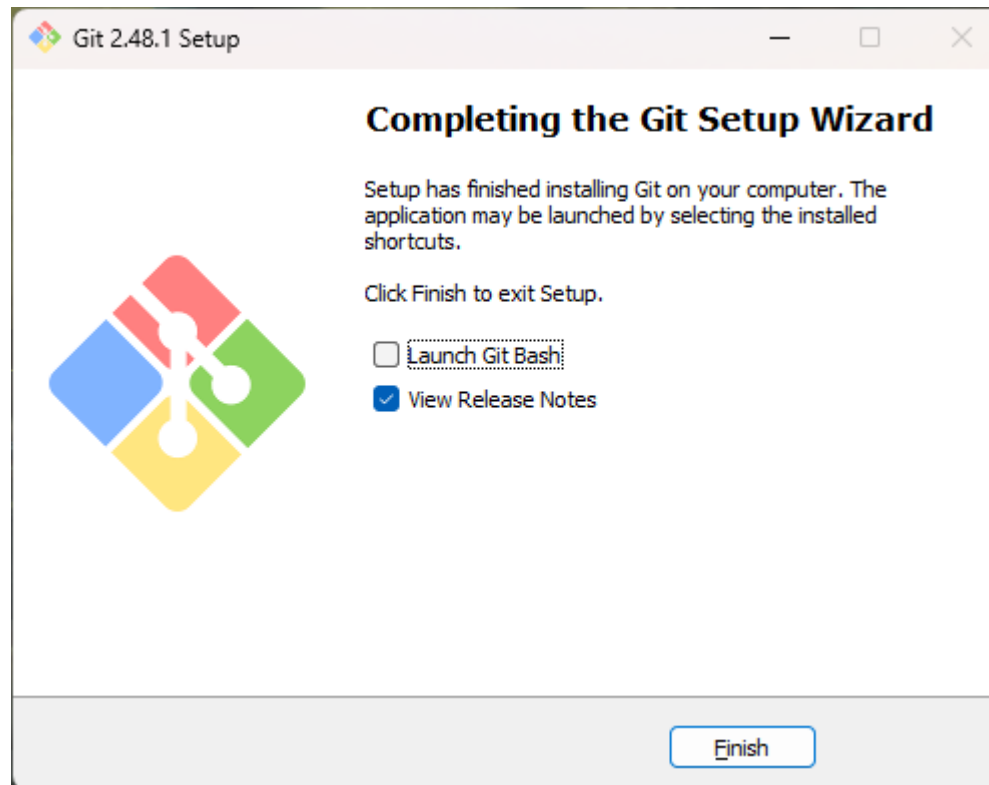
- 파일 시스템 캐시 활성화 선택 → 속도 향상
- 심볼릭 링크 사용 여부 선택 → 바로 가기 동작 여부



git 설치 - 8

- 설치 종료

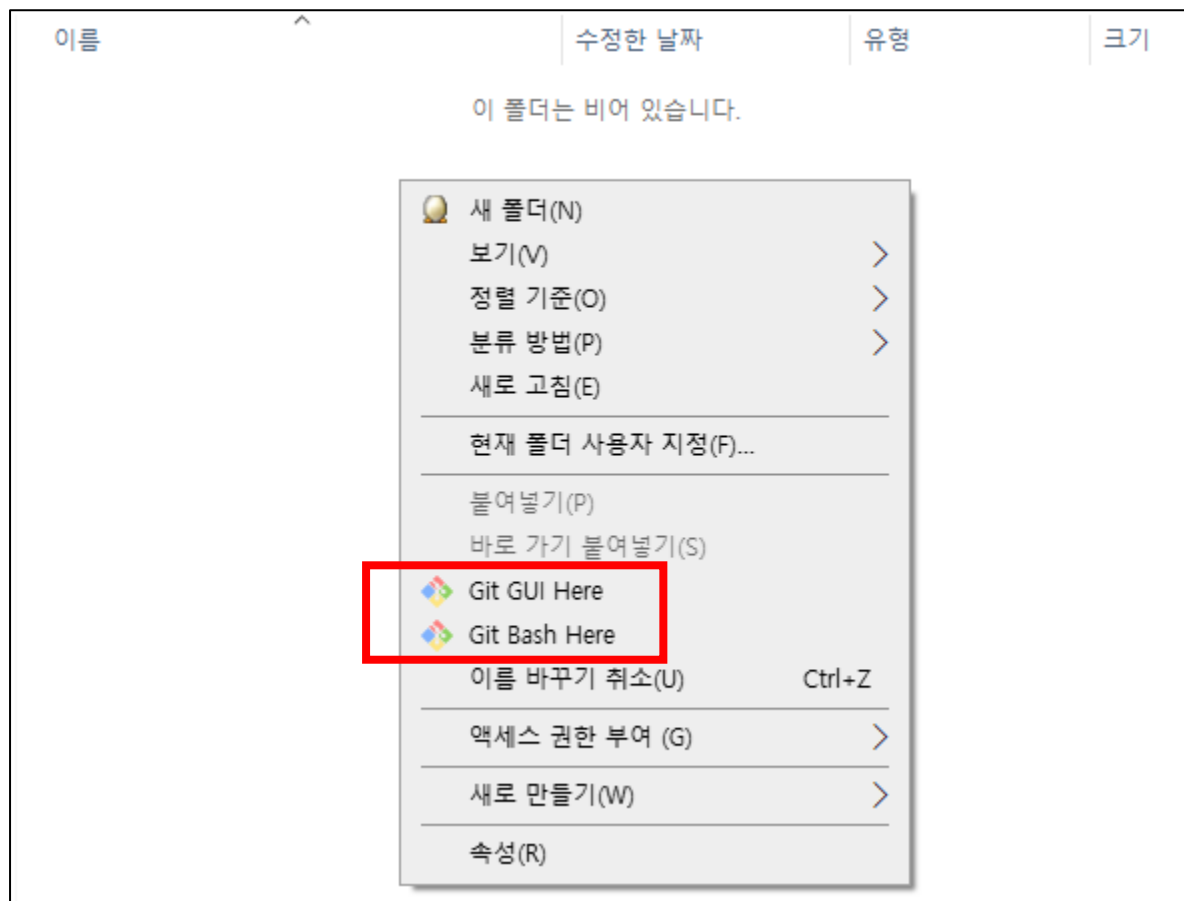
- git bash 즉시 실행 여부 선택
- 버전 주요 내용 문서 확인 여부 선택



git 설치 - 9

- 탐색기에서 설치 확인

- 빈 곳에서 오른쪽 클릭 시, git 메뉴의 추가 여부 확인 가능



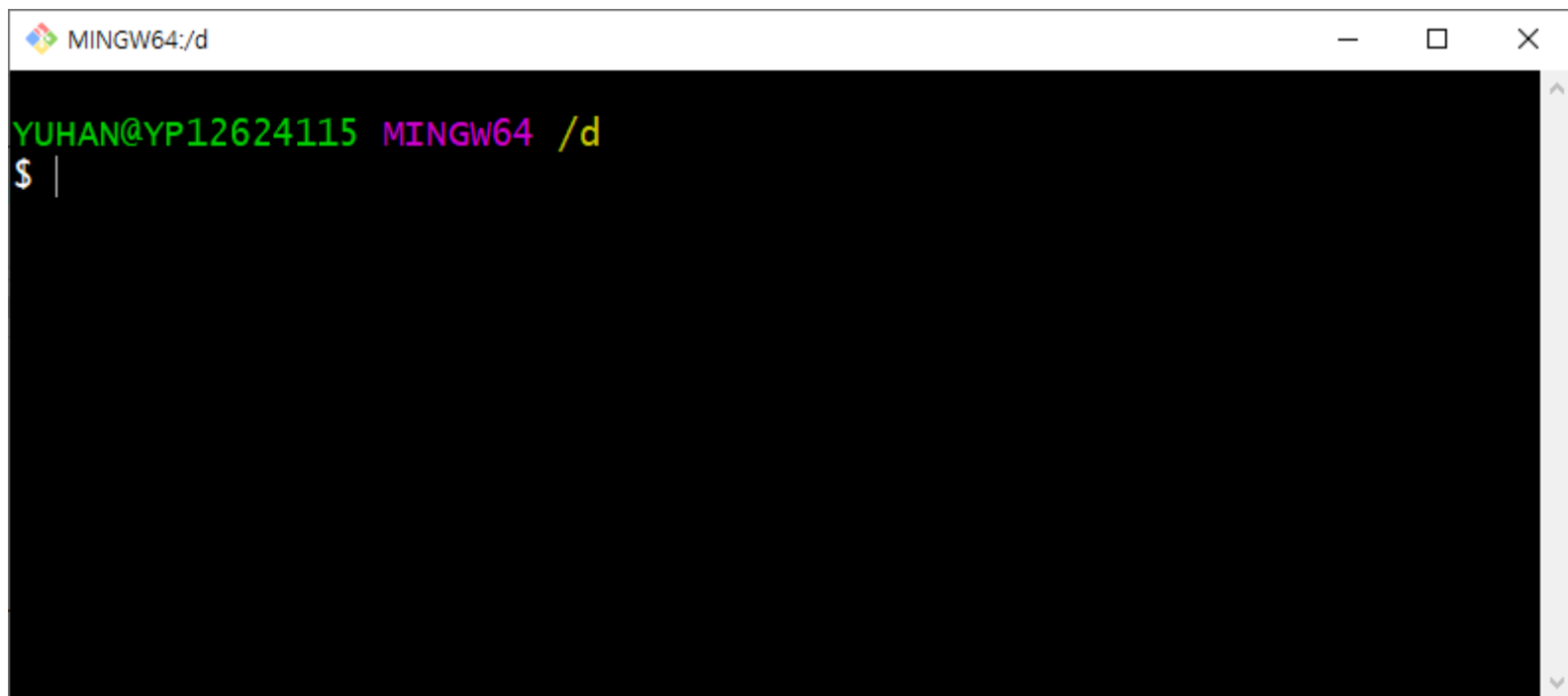


git bash – 1



- bash 프로그램

- CUI(Command User Interface) 형식의 명령 실행 프로그램
- 명령어를 입력하고, ENTER 키로 실행을 요청하는 형태
- 프로그램 입/출력이 모두 글자로 이루어지는 인터페이스
- 검은 바탕에 흰 글씨로 예전 인터페이스를 사용. 변경 가능



```
MINGW64:/d

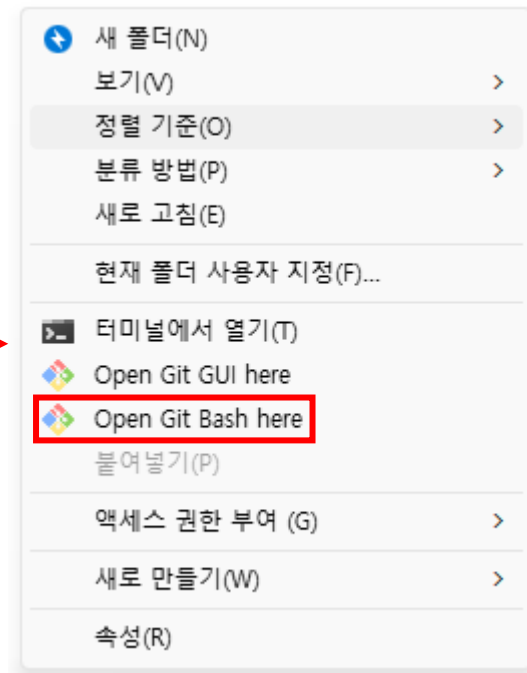
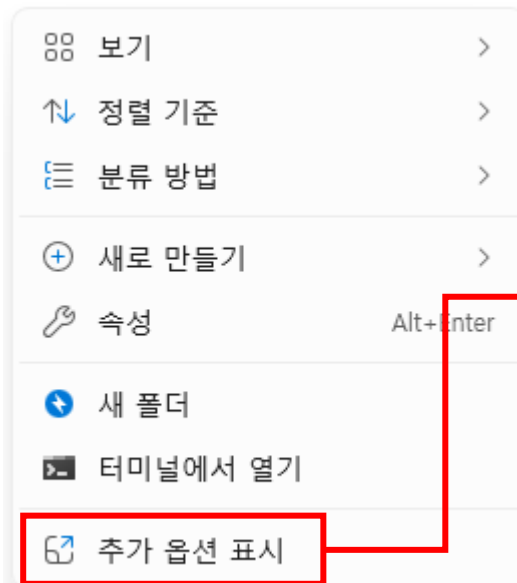
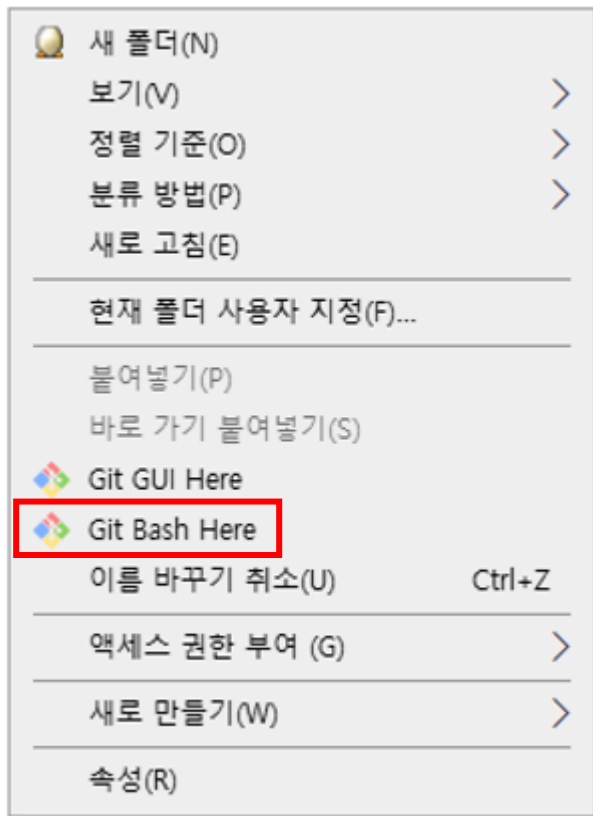
YUHAN@YP12624115 MINGW64 /d
$ |
```



git bash – 2

• git bash 실행

- Windows 10에서는 탐색기의 오른쪽 버튼으로 바로 확인 및 실행
- Windows 11에서는 탐색기 오른쪽 버튼의 추가 옵션에서 실행





git bash – 3



- 윈도우의 CUI 프로그램

- cmd.exe, command.exe, conhost.exe, powershell.exe 등

- 리눅스에서는 필수적으로 사용할 수 있어야 함

- 다수의 GUI(Graphic User Interface)가 개발되어 있는 상태임
- 한글화와 윈도우와 같이 사용할 수 있는 프로그램들이 개발되고 있음
- 하모니카 리눅스 : CUI를 완전히 대체하기 위한 국내 리눅스 프로젝트
- GUI로 해결하지 못하거나 CUI를 반드시 사용해야하는 경우도 존재
- 개발자는 당연히 CUI를 통해 다양한 작업을 수행해야 함
- CUI로만 해결할 수 있는 문제들이 존재
- 리눅스에서는 터미널(Terminal)로 명명하고 있음
- bash는 리눅스 셸(Shell)의 종류 가운데 하나

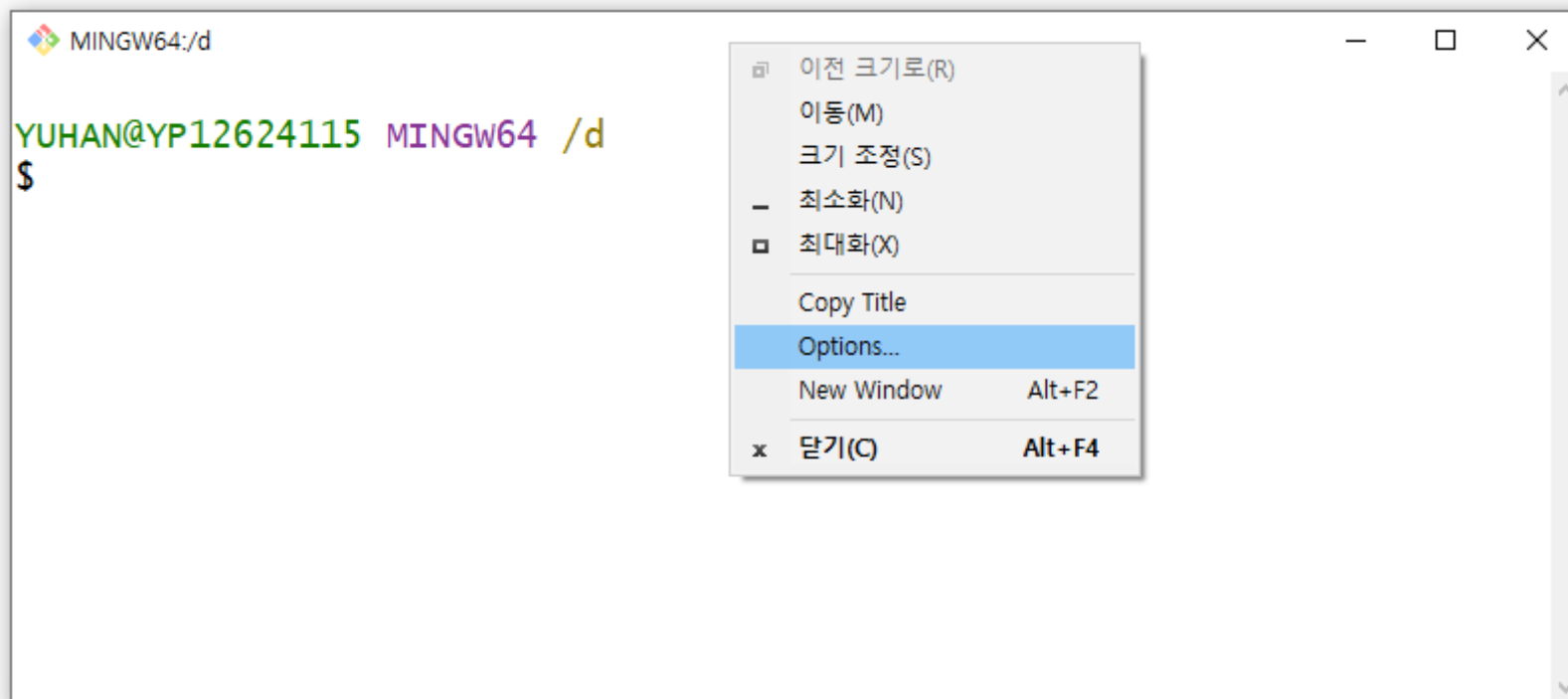


git bash – 4



• 환경 설정

- 터미널의 기본 설정을 변경하여 사용할 수 있음
- 터미널 상단의 타이틀 바에서 오른쪽 버튼으로 옵션 선택
- 폰트, 색상 등 터미널을 사용하는데 필요한 내용 수정 가능



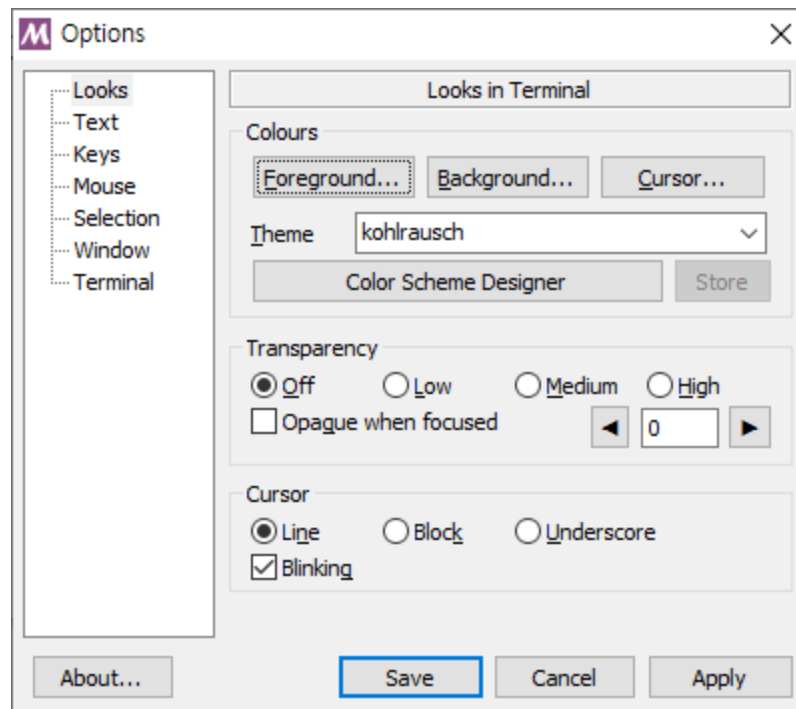


git bash – 5



- **bash 터미널 환경 설정**

- 터미널의 표시와 폰트 및 색상 등을 변경할 수 있음
- 필요에 따라 다양한 기능의 변경과 수정이 가능
- 본인에게 알맞은 터미널 환경을 찾는 과제



Open Source Software

터미널 실습과 활용



About..

컴퓨터소프트웨어공학과

김 원 일



터미널 실습 - 1



• 터미널 학습의 필요성

- 도구는 사용이 가장 어려운 환경에서 배워야 나중에 쉽게 사용이 가능
- 쉬운 환경에서 배우면 어려운 환경에서는 더 사용이 어려움
- 2학기부터 리눅스 과목이 3학기에 걸쳐 진행
- 리눅스는 현재 서버 구축으로 90% 이상 점유된 상태
 - 윈도우 프로그램에서 접속하는 서버는 리눅스 서버
 - 웹 서버는 거의 대부분이 리눅스 서버를 사용
 - 모바일 프로그램도 접속하는 서버는 리눅스를 사용
- 개발자로 리눅스에서 작업을 진행해야 하므로 필수적으로 학습해야 함
- 터미널로 프로그램을 배워야 GUI를 쉽게 사용 가능
- GUI 프로그램부터 사용하면 나중에 터미널로는 업무가 어려움
- 실제 서버들은 GUI가 필요 없는 경우가 많아 CUI가 강제되기도 함



터미널 실습 - 2



- git bash 터미널
 - 리눅스 터미널 환경을 사용할 수 있도록 구성
 - 간단한 리눅스 명령어를 사용할 수 있도록 프로그램 구성
 - 일반적인 리눅스의 환경에서 파일 관련 정보 확인 및 실행 가능
 - 터미널 화면에서 확인 가능한 정보
 - 컴퓨터 로그인 아이디, 컴퓨터 이름, 현재 경로와 사용자 권한(# - 관리자)

```
MINGW64:/d/내 드라이브/2025
unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$
```



명령어 실습 - 1



• ls(LS) 명령어 - 1

- "ls" 명령어 : 현재 경로의 파일/디렉터리 정보를 출력
- 파일은 기본 색으로, 디렉터리는 색이 있고 "/"가 붙어 있음
- 윈도우 CUI의 동일한 명령도 사용 가능
 - dir 명령도 동일하게 현재 디렉터리의 파일/디렉터리 정보를 출력
 - 명령어를 완전히 동일하게는 사용할 수 없으나 기본 사용은 가능

MINGW64:/d/내 드라이브/2025	MINGW64:/d/내 드라이브/2025
unangel@DESKTOP-I8OQG85 \$ ls desktop.ini git/ oss	unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025 \$ dir desktop.ini git oss
unangel@DESKTOP-I8OQG85 \$ dir desktop.ini git oss	unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025 \$ dir /a dir: cannot access '/a': No such file or directory
unangel@DESKTOP-I8OQG85 \$	unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025 \$ dir /p dir: cannot access '/p': No such file or directory
	unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025 \$



명령어 실습 - 2



• ls 명령어 - 2

- 윈도우의 cmd.exe 프로그램 실행
- git bash에서 실행되지 않는 옵션으로 실행 가능
- 터미널에서 수행 가능한 다양한 작업들이 존재
- 윈도우 터미널과 연동하여 작업 가능하도록 설정도 가능

The image shows two overlapping Windows Command Prompt windows. The background window is titled '명령 프롬프트' and shows the output of the 'dir /a' command, listing files and folders with their attributes and timestamps. The foreground window is titled '명령 프롬프트 - dir /p' and shows the output of the 'dir /p' command, which displays the same directory listing but with additional formatting, including file sizes and permissions.

```
C:\Users\unangel>dir /a
C 드라이브의 볼륨: OS
볼륨 일련 번호: 3C19-4691

C:\Users\unangel 디렉터리

2025-03-10 오전 12:31
2025-01-29 오후 02:23
2025-03-10 오전 12:31
2021-08-24 오전 12:08
2025-03-09 오후 03:57
2022-06-01 오후 08:22
2021-07-21 오후 02:49
2025-01-30 오전 12:19
2025-01-29 오후 02:23

C:\Users\unangel>dir /p
C 드라이브의 볼륨: OS
볼륨 일련 번호: 3C19-4691

C:\Users\unangel 디렉터리

2025-03-10 오전 12:31 <DIR> .
2025-01-29 오후 02:23 <DIR> ..
2025-03-10 오전 12:31 18 .bash_history
2021-08-24 오전 12:08 <DIR> .dotnet
2025-03-09 오후 03:57 35 .minttyrc
2022-06-01 오후 08:22 <DIR> .VirtualBox
2021-07-21 오후 02:49 <DIR> 3D Objects
계속하려면 아무 키나 누르십시오 . . . |
```



명령어 실습 - 3



• ls 명령어 - 3

- "ls" 명령어 뒤에 옵션으로 정보를 다양한 형태로 확인 가능
- "ls -a" : 현 디렉터리의 모든 파일/디렉터리 정보 출력
 - 모든 파일/디렉터리는 숨김 파일 등을 포함한 전체 파일
- "ls -l" : 현 디렉터리의 파일/디렉터리를 리스트 형태로 출력
 - 리스트로 파일/디렉터의 추가적인 정보를 제공

```
MINGW64:/d/내 드라이브/2025

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ ls -a
./  ../  desktop.ini  git/  oss/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ ls -l
total 1
-rw-r--r-- 0 unangel 197121 246 Mar 10 10:13 desktop.ini
drwxr-xr-x 0 unangel 197121  0 Mar  5 23:20 git/
drwxr-xr-x 0 unangel 197121  0 Mar 10 15:19 oss/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$
```

◆ 명령어 실습 - 4

• ls 명령어 - 4

- 터미널 명령어들은 옵션을 조합하여 사용할 수 있는 프로그램들이 존재
- "ls" 명령어도 "ls -a" 와 "ls -l" 옵션을 조합하여 사용 가능
- 조합하면 모든 파일/디렉토리를 리스트 형식으로 출력
- 명령어 옵션 조합을 통해 보다 많은 화면 정보 출력 가능

```
MINGW64:/d/내 드라이브/2025
unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ ls -al
total 1
drwxr-xr-x 0 unangel 197121  0 Nov 26 08:43 ./
drwxr-xr-x 0 unangel 197121  0 Mar 10 10:13 ../
-rw-r--r-- 0 unangel 197121 246 Mar 10 10:13 desktop.ini
drwxr-xr-x 0 unangel 197121  0 Mar  5 23:20 git/
drwxr-xr-x 0 unangel 197121  0 Mar 10 15:19 oss/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ |
```



명령어 실습 - 5



• 리눅스 파일 정보 - 1

- "ls" 명령어가 출력하는 내용들에 대한 요약 정보
- total 1 : 파일은 1개
- drwxr-xr-x : 파일에 대한 권한
- unangel : 파일 소유자의 ID
- Mar 10 15:19 : 생성 일자와 시간

```
MINGW64:/d/내 드라이브/2025

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ ls -al
total 1
drwxr-xr-x 0 unangel 197121  0 Nov 26 08:43 ./
drwxr-xr-x 0 unangel 197121  0 Mar 10 10:13 ../
-rw-r--r-- 0 unangel 197121 246 Mar 10 10:13 desktop.ini
drwxr-xr-x 0 unangel 197121  0 Mar  5 23:20 git/
drwxr-xr-x 0 unangel 197121  0 Mar 10 15:19 oss/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ |
```

❖ 명령어 실습 - 6

• 리눅스 파일 정보 - 2

- `"./"` : 리눅스의 다양한 경로 표현에서 현재 디렉토리를 나타냄
- 경로 표현에서 `"."`은 항상 현재 디렉토리를 나타냄
- `"../"` : 리눅스의 다양한 경로 표현에서 상위 디렉토리를 나타냄
- 경로 표현에서 `".."`은 항상 상위 디렉토리를 나타냄

```
MINGW64; d:/내 드라이브/2025

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브/2025
$ ls ../../2025
desktop.ini  git/  oss/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브/2025
$ ls
desktop.ini  git/  oss/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브/2025
$ ls git/..
desktop.ini  git/  oss/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브/2025
$ |
```



명령어 실습 - 7



- “cd” 명령어

- “**C**hange **D**irectory” 명령어는 작업 디렉터리 변경을 수행
- 사용법 : cd 변경할 디렉터리의 절대 또는 상대 경로
- 절대 경로 : 디렉터리 위치를 처음부터 기술하는 경로
 - Ex) d:\2025 → d 드라이브의 2025 디렉터리
- 상대 경로 : 현재 디렉터리 위치를 기준으로 경로를 기술하는 경로
- 절대/상대 경로 입력에 따라 다른 파일을 지정할 수도 있음
- CUI에서는 현재 디렉터리 위치를 반드시 알아야 정상 작업이 가능

- “pwd” 명령어

- “Print Working Directory” 명령어는 현재 작업 디렉터리를 출력
- 사용법 : pwd [옵션 없음]
- 어느 위치에서나 항상 현재 디렉터리 경로를 출력하는 명령어



• 디렉터리 이동 실습

- "cd ." 또는 "cd ./" : 현재 디렉터리로 이동하라는 명령
 - 현재 디렉터리 이동이기 때문에 실제 이동은 발생하지 않음
 - 보통 현재 디렉터리 내의 파일을 정확하게 지정하고자 할 때 사용
- "cd .." 또는 "cd ../" : 상위 디렉터리로 이동하라는 명령
- "pwd"로 항상 현재 디렉터리를 확인

```
MINGW64:/d/내 드라이브

unangle1@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ cd .

unangle1@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ cd ..

unangle1@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브
$ pwd
/d/내 드라이브

unangle1@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브
$
```




• 화면 초기화 명령어

- "clear" 옵션 없이 사용. cmd.exe에서는 "cls"가 동일 동작
- 터미널에서 작업하던 내용을 모두 제거하고 초기화 상태로 되돌림
- 이전 출력 정보가 모두 삭제되기 때문에 사용 시 주의 필요

```
MINGW64:/d/내 드라이브/2025

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브
$ cd 2025/

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브/2025
$ ls -al
total 1
drwxr-xr-x 0 unangel
drwxr-xr-x 0 unangel
-rw-r--r-- 0 unangel
drwxr-xr-x 0 unangel
drwxr-xr-x 0 unangel

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브/2025
$ clear
```

```
MINGW64:/d/내 드라이브/2025

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$
```



• 디렉터리 생성

- mkdir : bash에서 디렉터를 생성하는 명령어
- 사용법 : mkdir 디렉터리 이름 또는 경로/디렉터리 이름

```
MINGW64:/d/tmp/suho

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ ls
modify.txt
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ mkdir test/version
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ ls test/
version/
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$
```



• 디렉터리 삭제

- rmdir : bash에서 디렉터를 삭제하는 명령어
- 사용법 : rmdir 디렉터리 이름 또는 경로/디렉터리 이름
- 디렉터리에 파일/디렉터리가 존재하면 삭제할 수 없음
- 내부 파일/디렉터를 삭제하면 디렉터리 삭제가 가능

```
MINGW64:/d/tmp/suho

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ rmdir test/
rmdir: failed to remove 'test/': Directory not empty

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ rmdir test/version/

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ rmdir test/

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ |
```



명령어 실습 - 12



• 파일 생성

- touch : 파일을 생성하거나 파일의 정보를 업데이트
- 사용법 : touch 파일명 또는 경로/파일명
- 파일명이 존재하지 않는 경우에 파일을 생성

```
MINGW64:/d/tmp/suho

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ ls
modify.txt  original.txt  test/

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ touch test.txt

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ ls
modify.txt  original.txt  test/  test.txt

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$
```

◆ 명령어 실습 - 13

• 파일 업데이트

- touch 명령으로 파일 정보의 업데이트 수행
- 업데이트 시 파일명에 *을 사용하면 모든 파일을 대상으로 지정함

```
MINGW64:/d/tmp/suho
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ ls -l
total 2
-rw-r--r-- 1 YUHAN 197121 113 Mar 11 08:36 modify.txt
-rw-r--r-- 1 YUHAN 197121 119 Mar 11 08:35 original.txt
drwxr-xr-x 1 YUHAN 197121 0 Mar 12 08:18 test/
-rw-r--r-- 1 YUHAN 197121 0 Mar 12 08:23 test.txt

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ touch modify.txt

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ ls -l
total 2
-rw-r--r-- 1 YUHAN 197121 113 Mar 12 14:06 modify.txt
-rw-r--r-- 1 YUHAN 197121 119 Mar 11 08:35 original.txt
drwxr-xr-x 1 YUHAN 197121 0 Mar 12 08:18 test/
-rw-r--r-- 1 YUHAN 197121 0 Mar 12 08:23 test.txt

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$
```



• 파일 내용 보기 - 1

- cat : 파일을 열어 파일 내용을 화면에 출력
- 사용법 : cat 파일명 또는 경로/파일명
- 파일 내용 전체를 터미널 화면에 그대로 출력
- 파일이 길어도 전체를 보여주기 때문에 원하는 위치를 찾기 어려움

```
MINGW64:/d/tmp/suho

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ cat original.txt
이 파일은 원본 파일입니다 .

3번째 줄에 내용이 있습니다 .
2번째 줄은 비워 놓았습니다 .
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ cat modify.txt
이 파일은 원본 파일입니다 .
2번째 줄에 내용이 있습니다 .
비워 놓은 줄이 없습니다 .
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ |
```



• 파일 내용 보기 - 2

- less : 파일을 열어 파일 내용을 화면에 페이지 단위로 출력
- 사용법 : less 파일명 또는 경로/파일명
- 파일 내용 전체를 페이지 단위로 터미널 화면에 그대로 출력
- 파일이 길어도 페이지 단위로 전체를 출력하여 내용 확인이 편리
- 스페이스 바로 다음 페이지 이동하고, 'q'로 종료

```
MINGW64:/d/tmp/suho
YUHAN@YP12624115 MINGW64 /d/tmp$ less long_text.txt
"Chief Justice Roberts, Vice President Harris. Speaker Pelosi, Leader
Schumer, McConnell, Vice President Pence, my distinguished guests and
my fellow Americans, this is America's day.

This is democracy's day. A day of history and hope of renewal and reso
lve through a crucible for the ages. America has been tested anew and
America has risen to the challenge. Today, we celebrate the triumph no
t of a candidate, but of a cause, the cause of democracy. The people,
the will of the people, has been heard and the will of the people has
been heeded.

We've learned again that democracy is precious. Democracy is fragile.
At this hour, my friends, democracy has prevailed.

long_text.txt
```




• 파일 내용 보기 - 3

- head : 파일의 전체 내용에서 처음 N개의 줄을 화면에 출력
- 사용법 : head 파일명 또는 경로/파일명
- 파일 처음 부분의 내용을 간단하게 확인하기 위해 사용

```
MINGW64:/d/tmp/suho
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ head long_text.txt
"Chief Justice Roberts, Vice President Harris. Speaker Pelosi, Leader
Schumer, McConnell, Vice President Pence, my distinguished guests and
my fellow Americans, this is America's day.

This is democracy's day. A day of history and hope of renewal and reso
lve through a crucible for the ages. America has been tested anew and
America has risen to the challenge. Today, we celebrate the triumph no
t of a candidate, but of a cause, the cause of democracy. The people,
the will of the people, has been heard and the will of the people has
been heeded.

We've learned again that democracy is precious. Democracy is fragile.
At this hour, my friends, democracy has prevailed.

From now, on this hallowed ground, where just a few days ago, violence
sought to shake the Capitol's very foundation, we come together as on
e nation, under God, indivisible to carry out the peaceful transfer of
power, as we have for more than two centuries.

As we look ahead in our uniquely American way: restless, bold, optimis
tic, and set our sights on the nation we can be and we must be.

YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ |
```



• 파일 내용 보기 - 3

- tail : 파일의 전체 내용에서 마지막 N개의 줄을 화면에 출력
- 사용법 : tail 파일명 또는 경로/파일명
- 파일 마지막 부분의 내용을 확인하기 위해 사용

```
MINGW64:/d/tmp/suho
YUHAN@YP12624115 MINGW64 /d/tmp/suho
$ tail long_text.txt

Let me know in my heart when my days are through.

America, America, I gave my best to you.

Let's add. Let us add our own work and prayers to the unfolding story
of our great nation. If we do this, then when our days are through, ou
r children and our children's children will say of us: They gave their
best, they did their duty, they healed a broken land.

My fellow Americans, I close the day where I began, with a sacred oath
before God and all of you. I give you my word, I will always level wi
th you. I will defend the Constitution. I'll defend our democracy. I'll
```



명령어 실습 - 18



• 파일 및 디렉터리 삭제 - 1

- rm : 파일을 삭제하는 명령이나 디렉터리도 삭제 가능
- 사용법 : rm 파일명 또는 경로/파일명
- 파일이나 디렉터를 삭제할 수 있는 명령

```
MINGW64:/f/suho

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls
delete.txt  modify.txt  original.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ rm delete.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls
modify.txt  original.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ |
```



• 파일 및 디렉터리 삭제 - 2

- rm은 기본적으로 디렉터를 삭제할 수는 없음
- 옵션 -rf 입력으로 파일이나 디렉터를 삭제할 수 있음
- 사용법 : rm -rf 파일/디렉터리명 또는 경로/파일/디렉터리명
- 휴지통 이동이 아니라 완전 삭제이기 때문에 복구 불가능
- 하위 디렉터리나 파일이 존재해도 무시하고 완전 삭제하므로 주의

```
MINGW64:/f/suho

unange1@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ mkdir test

unange1@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ rm test
rm: cannot remove 'test': Is a directory

unange1@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ rm -rf test

unange1@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ |
```



• 파일 복사

- cp : 파일을 복사하는 명령어
- 사용법 : cp 경로/원본파일명 경로/복사할파일명
- 존재하는 파일을 복사하여 새로운 파일을 생성
- 경로를 지정하여 지정 경로에 복사할 수 있음

```
MINGW64:/f/suho

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls
modify.txt  original.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ cp modify.txt test.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls
modify.txt  original.txt  test.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ |
```



• 파일 및 디렉터리 이동

- mv : 파일/디렉터를 이동하는 명령어
- 사용법 : mv 경로/원본파일/디렉터리명 경로/이동할파일/디렉터리명
- 존재하는 파일/디렉터를 다른 위치로 이동하는 명령어

```
MINGW64:/f/suho
unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls
modify.txt  original.txt  test/  tv.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ mv tv.txt test/

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls
modify.txt  original.txt  test/

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls test/
tv.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ |
```



• 파일 및 디렉터리 이름 변경

- mv 명령어를 이용하여 파일/디렉터리 이름을 변경
- 이동할 경로를 동일 경로로 지정하면 이름 변경과 동일
- 파일과 디렉터리 모두 동일하게 이름 변경이 가능

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 $ ls
modify.txt  original.

unangel@DESKTOP-I80QG85 $ mv modify.txt mod.t

unangel@DESKTOP-I80QG85 $ ls
mod.txt  original.txt

unangel@DESKTOP-I80QG85 $
```

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
mod.txt  original.txt  test/

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ mv test/ test_1

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
mod.txt  original.txt  test_1/

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```



• 파일 상태 정보 보기

- file : 파일/디렉터리 종류에 대한 정보를 확인
- 사용법 : file 경로/파일|디렉터리명
- 파일이나 디렉터리의 종류를 판별하여 출력하는 명령어

```
MINGW64:/f/suho

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ file modify.txt
modify.txt: Unicode text, UTF-8 text, with CRLF line terminators

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ file test/
test/: directory

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$
```




• 파일/디렉터리 정보 보기

- stat : 파일 시스템에 등록된 파일/디렉터리에 대한 정보를 출력
- 사용법 : stat 경로/파일|디렉터리명
- 파일|디렉터리가 파일 시스템 상에 어떻게 인식되고 있는지를 확인
- 파일|디렉터리에 대한 세부적인 정보를 확인 가능

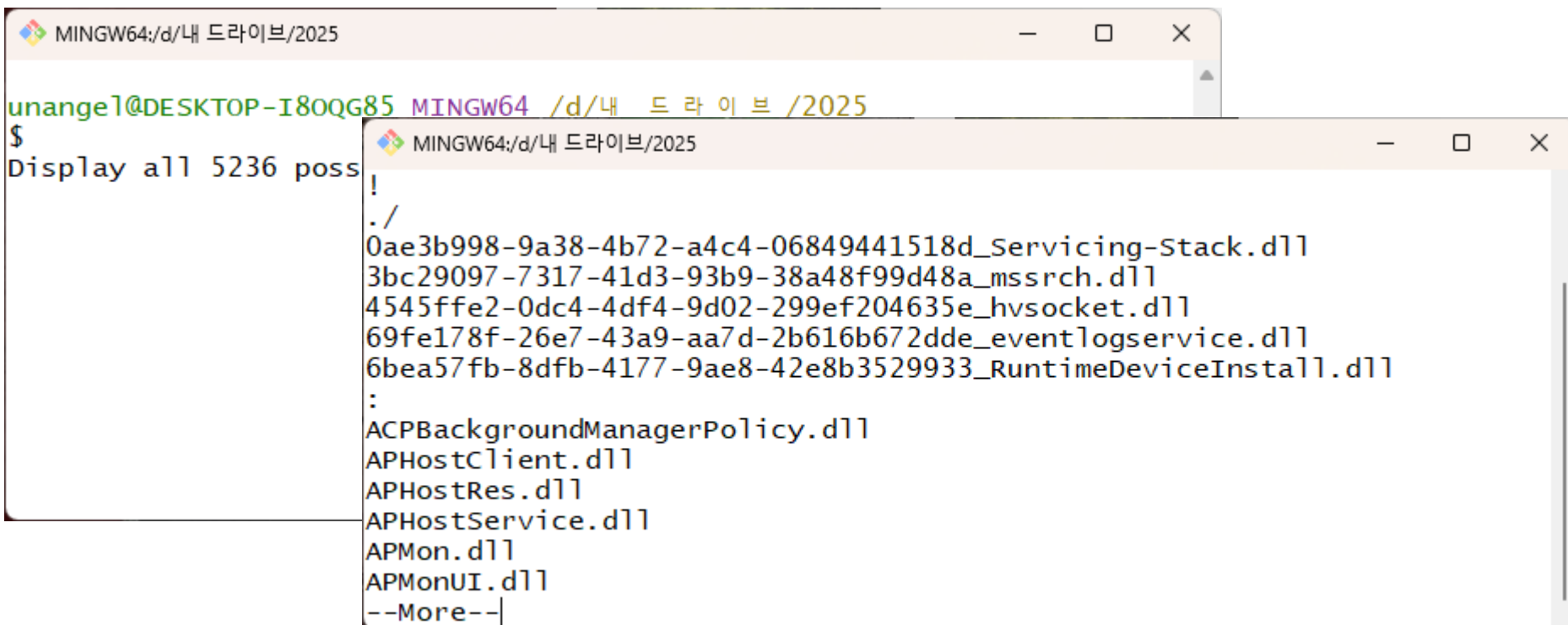
```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ stat mod.txt
  File: mod.txt
  Size: 644
ln
Device: 160a2df4h/369
Access: (0644/-rw-r--
OWN)
Access: 2025-03-12 22
Modify: 2025-03-12 19
Change: 2025-03-12 22
  Birth: 2025-03-12 19
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ stat test_1/
  File: test_1/
  Size: 0          Blocks: 0          IO Block: 65536  directory
Device: 160a2df4h/369765876d  Inode: 281474977091767  Links: 1
Access: (0755/drwxr-xr-x)  Uid: (197609/ unangel)   Gid: (197121/ UNKN
OWN)
Access: 2025-03-12 22:16:45.884012100 +0900
Modify: 2025-03-12 22:16:38.593673500 +0900
Change: 2025-03-12 22:29:01.737649300 +0900
  Birth: 2025-03-12 22:15:52.989323900 +0900
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ |
```

❖ 터미널 활용 - 1

• 명령어 찾기 - 1

- 명령어를 항상 암기해야 하는 것이 가장 어려운 문제
- 터미널 사용을 위한 간단한 팁으로 "tab" 키 활용
- "clear" 명령어로 초기화 한 다음 "tab" 연속 두 번 입력
- 엔터 키 입력 없이 "y"는 전체 정보 출력, "n"은 동작 중지



```
MINGW64:/d/내 드라이브/2025
unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$
Display all 5236 poss
!
./
0ae3b998-9a38-4b72-a4c4-06849441518d_servicing-stack.dll
3bc29097-7317-41d3-93b9-38a48f99d48a_mssrch.dll
4545ffe2-0dc4-4df4-9d02-299ef204635e_hvsocket.dll
69fe178f-26e7-43a9-aa7d-2b616b672dde_eventlogservice.dll
6bea57fb-8dfb-4177-9ae8-42e8b3529933_RuntimeDeviceInstall.dll
:
ACPBackgroundManagerPolicy.dll
APHostClient.dll
APHostRes.dll
APHostService.dll
APMon.dll
APMonUI.dll
--More--
```



터미널 활용 - 2



• 명령어 찾기 - 2

- "tab"을 이용한 정보 출력은 내용이 매우 많은 경우가 대부분
- 현재 화면 구성에서 출력할 수 있는 만큼만 출력하고 입력을 대기함
- 이때 스페이스 바, "y"는 한 페이지씩 내용을 추가 출력
- 엔터 키는 하나씩 내용을 추가 출력
- "q", "n", 백 스페이스는 출력 동작을 정지하고 셸로 다시 되돌아감

```
MINGW64:/d/내 드라이브/2025
0ae3b998-9a38-4b72-a4c4-06849441518d_Servicing-Stack.dll
3bc29097-7317-41d3-93b9-38a48f99d48a_mssrch.dll
4545ffe2-0dc4-4df4-9d02-299ef204635e_hvsocket.dll
69fe178f-26e7-43a9-aa7d-2b616b672dde_eventlogservice.dll
6bea57fb-8dfb-4177-9ae8-42e8b3529933_RuntimeDeviceInstall.dll
:
ACPBackgroundManagerPolicy.dll
APHostClient.dll
APHostRes.dll
APHostService.dll
APMon.dll
APMonUI.dll

unange1@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ |
```



터미널 활용 - 3

• 명령어 찾기 - 3

- "tab"은 "자동 완성"과 비슷한 기능으로 동작
- 특정 명령어의 전체 명령어가 기억나지 않는 경우에 "tab" 사용
- "pwd" 명령이 생각나지 않는 경우 "p" 입력 후 "tab" 2회 입력
- "p"로 시작하는 사용 가능한 명령어/파일을 확인할 수 있음

```
MINGW64:/d/내 드라이브/2025

unange1@DESKTOP-I80QQ...
$ p
Display all 269 possible completions:
PATHPING.EXE
PCPKsp.dll
PCShellCommonProxyStub.dll
PING.EXE
PNPXAssoc.dll
PNPXAssocPrx.dll
POSyncServices.dll
PSEvents.dll
PSHED.DLL
PSModuleDiscoveryProvider.dll
PackageInspector.exe
PackageStateChangeHandler.dll
PackagedCWALauncher.exe
PasswordEnrollmentManager.dll
--More--
```



터미널 활용 - 4



• 명령어 찾기 - 4

- 한 글자 입력 후 검색은 전체가 나오기 때문에 몇 글자 정도는 암기 필요
- "pwd"의 "pw"까지 입력 후 "tab" 2회 입력
- "pw"로 시작하는 명령어/기능의 목록을 출력
- 한 페이지(터미널 설정)를 넘지 않는 경우 사용자 선택 없이 출력
- 출력 후 "tab"입력 전의 문자열인 "pw"까지 자동으로 출력

```
MINGW64:/d/내 드라이브/2025

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ pw
pwd                pwlauncher.dll    pwrshmsg.dll      pwrshsip.dll
pwd.exe            pwlauncher.exe    pwrshplugin.dll   pwsso.dll

unangel@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ pw|
```

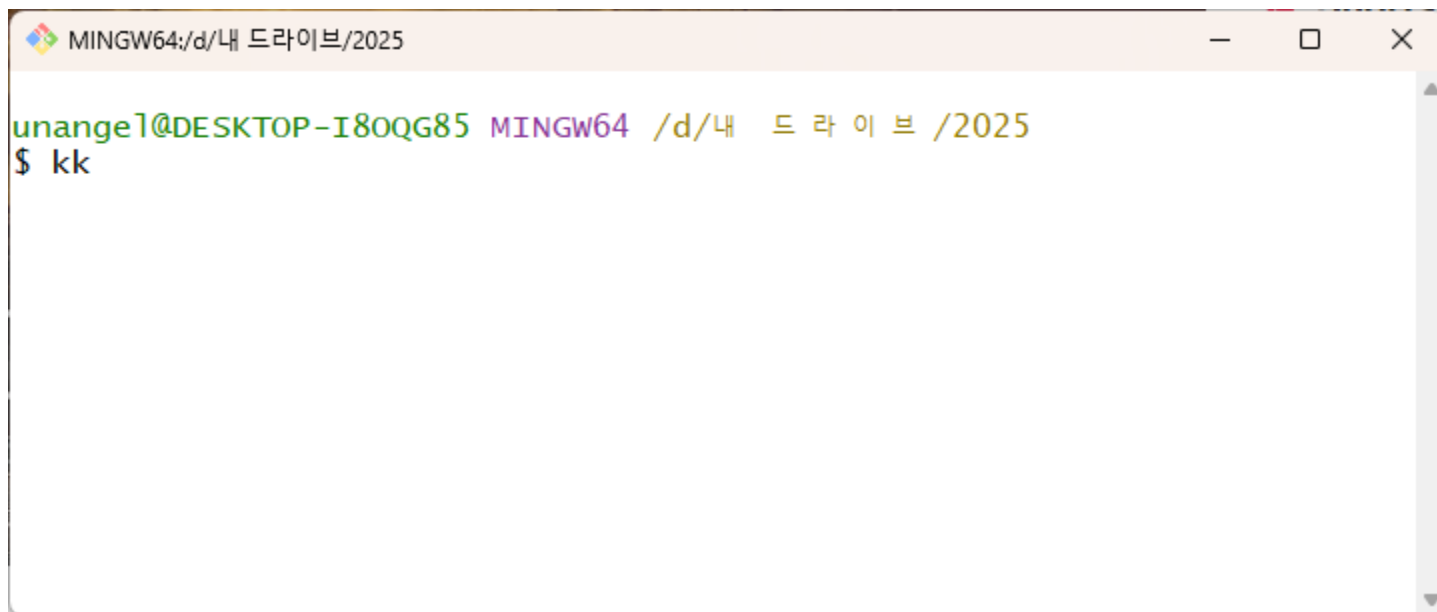


터미널 활용 - 5



• 명령어 찾기 - 5

- "tab"키 입력이 동작하지 않는 경우는 오타일 경우
- "kk"를 입력하고, "tab" 2회 입력 시, 반응이 없음
- "kk"로 시작하는 명령어/기능이 없다는 뜻

A screenshot of a MINGW64 terminal window. The title bar shows the path "MINGW64:/d/내 드라이브/2025". The terminal content shows the prompt "unange1@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025" followed by the command "\$ kk".

```
MINGW64:/d/내 드라이브/2025
unange1@DESKTOP-I8OQG85 MINGW64 /d/내 드라이브 /2025
$ kk
```



터미널 활용 - 6



- **터미널 명령어 입력**

- 터미널 상에서 이전에 입력한 명령어의 재 입력은 괴로운 일
- 셸에서 입력한 입력 정보 이력은 저장되고 관리됨
- 새로 터미널을 실행해도 해당 정보는 계속 유지됨
- 키보드 화살표 상/하로 이전 입력 명령어 확인, 수정 및 실행 가능

- **터미널에서 경로와 파일 이름 찾기**

- 명령어 또는 경로를 쉽게 완성하기
- "tab" 키를 이용하여 경로나 명령어 또한 자동 완성이 가능
- 전체 경로를 확인하거나 파일들을 확인할 때도 사용
- 경로를 모두 외울 필요 없이 파일이나 디렉터리 지정이 가능
- 한글로 되어 있는 경로와 파일까지 모두 자동으로 완성 가능



터미널 활용 - 7



- 다중 명령어 사용

- 연속 명령어 사용은 ";"로 구분하여 한번에 실행 가능
- 한 줄에 여러 명령어를 동시에 실행한 결과를 확인할 수 있음
- 오래 걸리는 작업들을 한 줄에서 한꺼번에 지시하여 실행 가능

```
MINGW64:/d/GoogleDrive/2025

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ pwd ; ls ; cd . ; pwd ; ls
/d/GoogleDrive/2025
desktop.ini  git/  oss/
/d/GoogleDrive/2025
desktop.ini  git/  oss/

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ |
```




• 명령어 도움말 보기

- CUI에서 명령어에 대한 실행 방법 등을 확인하기가 쉽지 않음
- 대부분의 명령어 뒤에 "--h" 또는 "--help"로 확인
- "-h" 옵션은 일반적인 옵션으로 인식될 수 있어 "--" 형식으로 사용
- 형식이 잘못된 경우 사용 방법을 출력하도록 구성되어 있음

```
MINGW64:/d/GoogleDrive/2025

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ ls -h
desktop.ini  git/  oss/

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ ls --h
ls: ambiguous option -- h
Try 'ls --help' for more information.

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ |
```



- 명령어 버전 확인

- 프로그램의 버전 확인
- "-v" 또는 "--version"을 명령어 뒤에 입력
- 대부분의 CUI 프로그램들은 동일한 옵션을 지원하고 있음

```
MINGW64:/d/GoogleDrive/2025

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ ls --v
ls (GNU coreutils) 8.32
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/g
pl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Richard M. Stallman and David MacKenzie.

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ |
```



• 명령어 매뉴얼 보기

- 프로그램의 명령어 사용법과 예제 등을 매뉴얼로 제공
- "man" 명령어로 매뉴얼 확인이 가능
- bash 터미널에서는 정상적으로 지원되지 않음
- 아래는 실제 리눅스에서 man 명령어를 사용한 화면

```
unangel@unangel-gooroom: ~/work
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
unangel@unangel-gooroom: ~/work$ man
What manual page do you want?
unangel@unangel-gooroom: ~/work$ man ls
```

```
unangel@unangel-gooroom: ~/work
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
LS(1) User Commands LS(1)
NAME
    ls - list directory contents
SYNOPSIS
    ls [OPTION]... [FILE]...
DESCRIPTION
    List information about the FILES (the current directory by default).
    Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-
    fied.
    Mandatory arguments to long options are mandatory for short options
    too.
    -a, --all
        do not ignore entries starting with .
    -A, --almost-all
        do not list implied . and ..
    --author
Manual page ls(1) line 1 (press h for help or q to quit)
```



• 터미널 작업 중단

- 터미널에서의 "ctrl + c"는 현재 작업을 중단하라는 의미
- 상호작용 프로그램의 실행을 중단 또는 배치 작업 실행 중단 시에 사용
- 프로그램의 동작을 강제종료를 위해 가장 많이 사용
- "nslookup"이 상호작용 프로그램의 대표적인 예

```
MINGW64:/f/suho

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ cat
kdkdkdkdkdkdkdkdkd
kdkdkdkdkdkdkdkdkd
this is a example
this is a example
yes
yes
but
but

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ |
```



터미널 활용 - 12



- 이전 사용 명령어

- 직전에 사용한 명령어를 바로 다시 사용할 경우
- "!명령어"로 이전에 사용한 명령어를 즉시 사용 가능
- 명령어의 단순한 재실행으로 현재 상태와 다른 실행이 될 수 있음

```
MINGW64:/f/suho
205 ps
206 clear
207 cat
208 nslookup
209 clear
210 ls
211 clear
212 history
213 ls -al
214 cd ..
215 cd suho/
216 history

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```

```
MINGW64:/f/suho
210 ls
211 clear
212 history
213 ls -al
214 cd ..
215 cd suho/
216 history

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ !cd
cd suho/
bash: cd: suho/: No such file or directory

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```



• 이전 입력 명령어 재실행

- history 명령어는 이전에 입력했던 명령어들의 이력을 출력
- 이전에 입력한 명령어들을 입력했던 순서를 포함하여 보관하고 있음
- 특정 번호에 해당하는 명령을 수행하려면 "!"를 이용

```
MINGW64:/f/suho
$ history
1 cd ..
2 ls
3 clear
4 ls
5 ls
6 dir
7 dir /a
8 dir /p
9 clear
10 ls
11 dir
12 dir /a
13 dir /p
14 exit
15 ls -a
16 ls -al
```

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ !4
ls
mod.txt  original.txt  test_1/

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ |
```



• 출력 내용 보관

- 화면 출력 내용을 파일로 보관하는 방법
- 화면 출력은 표준 출력(1) 디스크립터를 이용하는 것을 의미
- 즉, `printf()`를 통해 화면에 출력하는 내용은 파이프를 통해 보관 가능
- 사용법 : 명령어 실행에 따른 출력 > 보관할 파일명

```
MINGW64:/f/suho
unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ ls -al > ls.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ cat ls.txt
total 7
drwxr-xr-x 1 unangel 197121  0 Mar 13 00:04 ./
drwxr-xr-x 1 unangel 197121  0 Mar 11 23:21 ../
drwxr-xr-x 1 unangel 197121  0 Mar 11 23:27 .git/
-rw-r--r-- 1 unangel 197121  0 Mar 13 00:04 ls.txt
-rw-r--r-- 1 unangel 197121 644 Mar 12 19:12 mod.txt
-rw-r--r-- 1 unangel 197121 661 Mar 13 00:00 original.txt
-rw-r--r-- 1 unangel 197121 150 Mar 13 00:00 test.patch
drwxr-xr-x 1 unangel 197121  0 Mar 12 22:16 test_1/

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$
```



- 터미널 종료

- 터미널의 종료 "exit" 입력
- 터미널의 안정적인 종료 명령어

```
MINGW64:/d/GoogleDrive/2025

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ quit
bash: quit: command not found

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ bye
bash: bye: command not found

YUHAN@YP12624115 MINGW64 /d/GoogleDrive/2025
$ exit
```




기타 리눅스 명령어



- **제한적인 명령어만 사용**

- git bash는 리눅스 환경과 유사한 환경을 만들어주기만 함
- 실제 리눅스 운영체제와 동일한 환경은 아님
- 주로 git을 활용하기 위한 명령어들만 사용 가능하도록 구성
- 실제 명령어의 실행과 확인은 리눅스를 설치해야 가능
- 추가적인 내용은 수업에서 간단히 진행할 예정
- 리눅스 관련 수업 진행을 위해서는 반드시 리눅스 OS가 필요
- 별도 설치 없이도 WSL을 활용하는 방법도 존재
- 2학기부터 리눅스 수업이 시작되는데 선행해도 문제 없음