

Open Source Software

관리 대상 제외



About..

컴퓨터소프트웨어공학과

김 원 일



git의 관리 대상과 제외 대상



• 파일 이력 관리 대상

- git은 기본적으로 모든 종류의 파일을 대상으로 이력을 관리
- 파일의 확장자 유무 등과 관계없이 무조건 관리 대상으로 지정
- LFS(Large File Support)로 큰 파일까지도 관리 대상으로 지정 가능
- 모든 파일을 관리 대상으로 지정하면 높은 비용이 발생
 - 빌드에서 발생하는 중간 단계의 파일은 관리할 필요 없음
 - 실행 파일도 소스 코드 변화에 따라 변화하므로 굳이 관리할 필요 없음
 - 실행 파일은 실행되는 운영체제와 시스템 환경에 따라 오동작할 수 있음
- 관리 대상에서 제외할 파일이나 디렉터리를 기록할 파일 추가 가능
- 기본적으로 "git init"을 통한 저장소 생성에는 만들어지지 않음
- 별도 파일을 추가하여 관리 대상에서 제외시킬 수 있음



.gitignore 파일 - 1



• 파일 버전 관리 예외 정보 기록

- .(숨김 속성) + git + ignore(무시)
- "."으로 시작하는 이유는 리눅스를 기반으로 하기 때문
- 리눅스에서는 파일명이 "."으로 시작하면 숨김 파일
- 사용자에게 노출할 필요가 없으므로 윈도우에서도 숨김 속성으로 처리

```
.gitignore - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

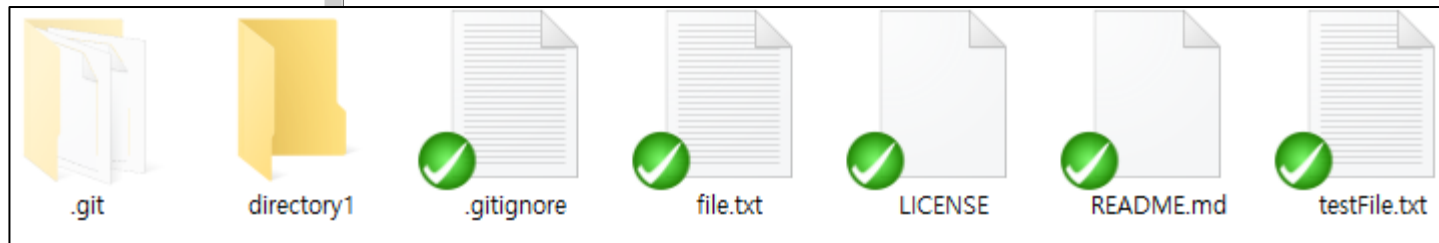
# History files
.Rhistory
.Rapp.history

# Session Data files
.RData

# User-specific files
.Ruserdata

# Example code in package build process
*-Ex.R

Ln 30, Col 23    100%    Windows (CRLF)    UTF-8
```





.gitignore 파일 - 2

- 파일과 디렉터리 모두 지정 가능

- #으로 시작되면 주석으로 처리되어 적용되지 않음
- 상대, 절대 경로를 이용하여 특정 파일/디렉터리 지정도 가능
- 예외로 지정되면 어떠한 추가/수정/삭제가 발생해도 관리하지 않음
- .gitignore도 파일이기 때문에 수정하면 반드시 commit해야 함

```
.gitignore - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

#test
*.text
```

```
# History files
.Rhistory
.Rapp.history

# Session Data files
.RData

# User-specific files
.Ruserdata
```

```
MINGW64:/d/___수업자료/2021/2학기/R/201007001
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .gitignore

no changes added to commit (use "git add" and/or "git commit -a")
YUHAN@YP12624115 MINGW64 /d/___수업자료/2021/2학기/R/201007001 (main)
$
```



.gitignore 파일 - 3

- 파일과 디렉터리 모두 지정 가능

- 확장자가 ".text"라면 파일에 대한 추적을 수행하지 않음
- 새롭게 파일을 추가하더라도 추적 대상에 포함되지 않음
- 디버깅 또는 임시 파일들을 추가하지 않도록 구성할 때 사용

```
MINGW64:/d/___수업자료/2021/2학기/R/201007001
YUHAN@YP12624115 MINGW64 /d/___수업자료/2021/2학기/R/201007001 (main)
$ ls
LICENSE  README.md  directory1/  file.txt  testFile.txt

YUHAN@YP12624115 MINGW64 /d/___수업자료/2021/2학기/R/201007001 (main)
$ vi file.txt

YUHAN@YP12624115 MINGW64 /d/___수업자료/2021/2학기/R/201007001 (main)
$ ls
LICENSE  README.md  directory1/  file.text  file.txt  testFile.txt

YUHAN@YP12624115 MINGW64 /d/___수업자료/2021/2학기/R/201007001 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

YUHAN@YP12624115 MINGW64 /d/___수업자료/2021/2학기/R/201007001 (main)
$
```



관리 대상 제외 실습 - 1



- “.gitignore” 파일
 - 관리 대상 제외 파일/디렉터리 정보를 입력하는 파일
 - 저장소의 최상위에 존재해야만 정상적으로 동작
 - 관대 대상 제외 실습을 위한 저장소를 별도로 생성
 - 저장소에 “.gitignore” 파일 생성 후 아래 내용 입력
 - 입력 후 저장(w)하고 종료(q)

```
MINGW64:/f/suho
# except file : *.text
*.text

# temp file
*.tmp

~
~
~
~
~
~
~

.gitignore[+] [unix] (08:59 01/01/1970) 6,0-1 All
:wq
```



관리 대상 제외 실습 - 2



- 제외 적용 시점 확인

- 제외 대상 정보의 파일도 파일이기 때문에 저장소 추가가 필요
- 아직 관리 대상이 아닌 상태에서 제외 대상 파일 생성
- 정확한 적용 확인을 위해 적용 대상이 아닌 파일도 같이 생성

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

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

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

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



관리 대상 제외 실습 - 3



- 제외 대상 파일 생성

- 관리 대상이 아닌 상태에서 파일이 존재하는 것만으로도 제외 적용
- .gitignore 파일이 생성되면 그 즉시 적용되는 것을 확인 가능

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch test.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch test.text

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        test.c

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

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




관리 대상 제외 실습 - 4



• 제외 대상 파일 삭제

- 적용되어 있는 제외 대상 파일을 바로 삭제한 경우의 상태
- 제외 대상 파일을 삭제하면 즉시 제외 대상 파일도 대상으로 변경
- 즉, .gitignore 파일의 존재 유무 만으로 제외 여부를 즉시 판별

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ rm -rf .gitignore

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        test.c
        test.text

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

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



관리 대상 제외 실습 - 5



- 제외 대상 파일 다시 추가

- 동일한 .gitignore 파일을 다시 생성한 다음의 상태
- 제외 대상 파일인 test.text 파일이 대상에서 제외된 것을 확인 가능
- 따라서 .gitignore 파일은 저장소 생성 시에 만드는 것이 좋음

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi .gitignore

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        test.c

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

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



관리 대상 제외 실습 - 6



• 상태 보관

- 추가 실습을 위해 현재 상태를 그대로 commit하여 보관
- 제외 파일은 그대로 저장소에 존재하는 상태
 - 제외 파일은 저장소에서 지우거나 이동 또는 복사해도 git은 관여하지 않음

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git add .gitignore test.c
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git commit -m "add test files"
[master (root-commit) 1b13bd2] add test files
2 files changed, 6 insertions(+)
create mode 100644 .gitignore
create mode 100644 test.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
test.c  test.text

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



관리 대상 제외 파일 실습 - 1



• 파일 추가

- 저장소에 대상 제외와 관리 대상 파일을 각각 추가
- 제외 대상 파일은 개수와 관계 없이 제외되는 것을 확인

```
MINGW64:/f/suho
unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch except.cpp

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch allow.text

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    except.cpp

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

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
allow.text  except.cpp  test.c  test.text

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```



• 제외 대상 변경

- *.text 제외 문장 앞에 #으로 해당 줄을 주석으로 처리
- 처리 후 저장하고 편집기 종료

[illegible]



관리 대상 제외 파일 실습 - 3



• 파일 수정 후 확인

- .gitignore 파일 수정이 즉시 반영되는 것을 확인할 수 있음
- 현재 상태에서 모든 파일을 관리 상태로 변경

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi .gitignore

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .gitignore

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        allow.text
        except.cpp
        test.text

no changes added to commit (use "git add" and/or "git commit -a")

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



관리 대상 제외 파일 실습 - 4



- 관리 대상으로 모두 추가
 - 생성한 모든 파일들을 관리 대상으로 추가
 - 제외 대상 파일들까지 모두 포함하여 관리대상으로 추가한 상태

```
MINGW64:/f/suho

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git add . ; git commit -m "add files"
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the ne
xt time Git touches it
[master c7b9823] add files
 4 files changed, 1 insertion(+), 1 deletion(-)
 create mode 100644 allow.text
 create mode 100644 except.cpp
 create mode 100644 test.text

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
nothing to commit, working tree clean

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ |
```



관리 대상 제외 파일 실습 - 5



• 파일 수정

- .gitignore 파일을 이전에 주석으로 처리했던 제외 대상의 주석 제거
- 이미 commit한 allow.text 파일을 수정
- 제외 대상에 해당하는 add.text 파일을 생성하고 상태 확인

```
MINGW64:/f/suho
# except file : *.text
*.text

# temp file
*.tmp

~
~
~
~
~
~
~
~
~
~

.gitignore[+] [unix] (22:54 21/03/2025) 6,0-1 All
:wq
```




관리 대상 제외 파일 실습 - 6



• 수정 후의 상태

- 이전 확인에서 .gitignore 파일은 수정 후 즉시 적용되도록 구성
- 새로 추가한 제외는 적용되나 이미 commit된 파일은 적용되지 않음

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi .gitignore

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi allow.text

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch add.text

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .gitignore
        modified:   allow.text

no changes added to commit (use "git add" and/or "git commit -a")

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



관리 대상 제외 파일 실습 - 7



- .gitignore의 적용 시점 문제
 - 이미 commit된 파일에 대해서는 대상 제외가 적용되지 않음
 - 제외 대상 파일이라도 이미 commit되면 관리 대상에 포함
 - 즉, 대상 제외보다 commit된 파일에 대한 처리가 우선 적용
 - 따라서 .gitignore 파일은 저장소 생성 시점에 미리 적용하는 것이 좋음
 - 개발 과정 중에 어쩔 수 없거나 나중에 발견된 경우도 존재
 - 나중에 확인된 상태에서는 전체적으로 정리를 수행해야 함
 - 정리를 위해서는 git의 캐시를 삭제하는 방법이 가장 일반적
 - "git rm -r --cached ." 명령을 저장소 루트에서 실행

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git rm -r --cached .
rm '.gitignore'
rm 'allow.text'
rm 'except.cpp'
rm 'test.c'
rm 'test.text'
```



관리 대상 제외 파일 실습 - 8



• 캐시 삭제 후의 상태

- 캐시 삭제가 되면 저장소의 영향을 받는 모든 파일 삭제가 확인됨
- 실제 삭제가 아니라 git 관리 대상에서의 삭제를 의미함
- 변경이 없었던 관리 대상 파일도 "Untracked" 상태로 변경됨

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    deleted:    .gitignore
    deleted:    allow.text
    deleted:    except.cpp
    deleted:    test.c
    deleted:    test.text

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore
    except.cpp
    test.c

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



관리 대상 제외 파일 실습 - 9



• 파일 변화 적용

- "git add ."를 수행하고 상태를 확인하면 정리된 상태가 확인됨
- 제외 대상 파일에서 commi된 파일들을 제거로 나타남
- .gitignore 수정으로, 추가된 add.text는 목록 자체에서 제외됨

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git add .

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .gitignore
        deleted:    allow.text
        deleted:    test.text

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
add.text  allow.text  except.cpp  test.c  test.text

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



관리 대상 제외 파일 실습 - 10



- commit으로 확정

- 명령 실행하면, 이전에 commit된 2개의 파일이 삭제됨을 확인
- 제외 대상은 수정해도 관리되지 않는 것을 확인할 수 있음

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git commit -m "modify .gitignore and apply"
[master 88c71be] modify .gitignore and apply
3 files changed, 1 insertion(+), 1 deletion(-)
delete mode 100644 allow.text
delete mode 100644 test.text

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi allow.text

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi test.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test.c

no changes added to commit (use "git add" and/or "git commit -a")
```



관리 대상 제외 파일 실습 - 11



- .gitignore 다시 수정

- 제외 되었던 확장자를 다시 적용했을 때의 상태 확인
- 즉시 해당 확장자의 파일들이 "Untracked"로 추가되는 것을 확인

```
MINGW64:/f/suho

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi .gitignore

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .gitignore
        modified:   test.c

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        add.text
        allow.text
        test.text

no changes added to commit (use "git add" and/or "git commit -a")

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```



관리 대상 제외 디렉터리 실습 - 1



• 디렉터리 제외

- 디렉터리 제외 실습을 위해 작업 디렉터리를 제거하거나 새로 생성

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho
$ git init
Initialized empty Git repository in F:/suho/.git/

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi .gitignore

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

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

MINGW64:/f/suho
# directory ignore
dir
test|

.gitignore[+] [unix] (08:59 01/01/1970) 3,5 All
-- INSERT --
```



관리 대상 제외 디렉터리 실습 - 2



• 디렉터리 생성

- 제외 대상으로 지정된 dir 디렉터를 생성
- .gitignore 파일은 commit하지 않아도 적용되어 바로 확인 가능
- 새로 생성된 dir 디렉터리가 추가되지 않는 것을 확인

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

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

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

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

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

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




관리 대상 제외 디렉터리 실습 - 3



- 디렉터리에 파일 추가

- dir 디렉터리에 제외 대상이 아닌 파일을 추가
- 파일은 디렉터리에 종속되는 형식
- 파일은 제외 대상이 아니지만 포함된 디렉터리가 제외되어 자동 제외

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls -al > dir/ls.txt

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

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

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

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



관리 대상 제외 디렉터리 실습 - 4



- 대상이 아닌 디렉터리 생성

- 제외 대상인 dir과 test가 아닌 new 디렉터를 생성
- new 디렉터리는 제외 대상이 아님에도 관리되지 않는 것을 확인

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

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
dir/  new/

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

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

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

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



관리 대상 제외 디렉터리 실습 - 5



• 디렉터리에 파일 추가

- 대상이 아닌 디렉터리에 파일 추가 후 저장소 상태 확인
- “Untracked” 항목에 new/ 디렉터리가 나타남
- 즉, 비어있는 디렉터리는 관리 대상에 포함되지 않는 것을 확인
- 디렉터리를 관리 대상에 포함하려면 내부에 파일을 생성해야 함

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls -al > new/ls.txt

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        new/

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

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



관리 대상 제외 디렉터리 실습 - 6



- 위치에 따른 git status 정보 변화
 - 생성한 new 디렉터리로 이동하여 상태를 확인하면 표시 정보가 변경
 - 현재 위치를 기준으로 정보가 출력됨
 - 디렉터리 내부에 생성된 파일을 나타내지 않고 디렉터를 표시
 - 즉, 디렉터리에 관리 대상 파일이 있으면 디렉터를 표시

```
MINGW64:/f/suho/new
unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ cd new/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho/new (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  ../.gitignore
  ./

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

unange1@DESKTOP-I80QG85 MINGW64 /f/suho/new (master)
$
```



관리 대상 제외 디렉터리 실습 - 7



- git add 실행 결과
 - “Untracked” 상태에서는 디렉터리만 출력
 - “Staged” 상태에서는 디렉터리와 파일명이 정확하게 출력됨

```
MINGW64:/f/suho
unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the ne
xt time Git touches it
warning: in the working copy of 'new/ls.txt', LF will be replaced by CRLF the ne
xt time Git touches it

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   .gitignore
    new file:   new/ls.txt

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ |
```



관리 대상 제외 디렉터리 실습 - 8



- git commit 실행
 - 현재까지 진행된 파일 처리를 추가하여 관리 상태로 전환
 - 디렉터리 또한 관리 대상에서 제외할 수 있음을 확인

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git commit -m "add directory"
[master (root-commit) c242e39] add directory
2 files changed, 11 insertions(+)
create mode 100644 .gitignore
create mode 100644 new/ls.txt

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

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
dir/  new/

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



관리 대상 제외 실습 환경 - 1



• 파일/디렉터리 생성 주의

- 리눅스에서는 파일 확장자가 의미 없기 때문에 확인 필요
- 디렉터리명과 파일명은 동시에 존재할 수 없으므로 확인
- 반대로 파일이 존재하는 경우 디렉터리명을 생성할 수 없음
- 파일을 제외하려다 디렉터리를 제외시켜버리는 상황 발생 가능

📁 .git	2025-03-29 오후 7:09	파일 폴더
📁 dir	2025-03-29 오후 7:04	파일 폴더
📁 new	2025-03-29 오후 7:08	파일 폴더
📄 .gitignore	2025-03-29 오후 7:07	txtfile
📄 dir	2025-03-29 오후 9:13	텍스트 문서

파일 이름 바꾸기







지정한 파일 이름과 같은 이름으로 된 폴더가 이미 있습니다. 다른 이름을 지정하십시오.



새 텍스트 문서.txt
유형: 텍스트 문서
크기: 0바이트
수정한 날짜: 2025-03-29 오후 9:13

다시 시도(R)

취소

 .git	2025-03-29 오후 7:09	파일 폴더	
 dir	2025-03-29 오후 7:04	파일 폴더	
 new	2025-03-29 오후 7:08	파일 폴더	
 asdf	2025-03-29 오후 9:14	파일 폴더	
 .gitignore	2025-03-29 오후 7:07	txtfile	1KB
 asdf	2025-03-29 오후 9:13	파일	0KB

폴더 이름 바꾸기

지정한 폴더 이름과 같은 이름으로 된 파일이 이미 존재합니다. 다른 이름을 지정하십시오.



새 폴더
만든 날짜: 2025-03-29 오후 9:14

다시 시도(R)

취소



관리 대상 제외 실습 환경 - 2



• 대상 제외 실습

- 관리 대상에서 디렉터리는 마지막에 "/"를 붙여서 표시하여 구분이 편리
- 디렉터리를 제외하면 내부의 대상 파일이라도 모두 제외
 - 제외 디렉터리에 관리 대상 파일이 포함되어 있는 경우
 - 디렉터리의 하위 디렉터리 지정의 경우
- *(와일드 카드)를 이용한 대상 제외 방법 확인
- 확인을 위한 .gitignore 파일을 아래와 같이 지정

```
MINGW64:/f/suho
1 # directory ignore
2 dir/
3 test/view/
4
5 # except files
6 *.text
7 tmp.*
8 net*.o
9 wide*.*
10 *bio*.k
11 *key_*.
12
~
.gitignore[+] [unix] (23:05 29/03/2025) 12,0-1 All
:wq
```




관리 대상 제외 실습 환경 - 3



• .gitignore 파일 내용

- dir/ 디렉터리와 test/ 디렉터리의 하위 디렉터리 view/ 디렉터리 제외
- *.text : 파일명 상관 없이 확장자가 "text"인 모든 파일 제외
- tmp.* : 확장자가 무엇이든 상관 없이 파일명 tmp인 모든 파일 제외
- net*.o : 파일이름의 시작이 net인 모든 파일의 확장자가 o인 모든 파일 제외
- wide*.* : 파일 이름의 시작이 wide인 모든 파일은 확장자와 관계 없이 제외
- *bio*.k : 파일 이름 중간에 bio가 포함되고, 확장자가 k인 모든 파일 제외
- *key_*. * : 파일 이름 중간에 key_가 포함되어 있는 모든 파일 제외

```
MINGW64:/f/suho
1 # directory ignore
2 dir/
3 test/view/
4
5 # except files
6 *.text
7 tmp.*
8 net*.o
9 wide*.*
10 *bio*.k
11 *key_*. *
12
~
.gitignore[+] [unix] (23:05 29/03/2025) 12,0-1 All
:wq
```



관리 대상 제외 실습 환경 - 4



• 실습 디렉터리 생성

- 실습을 위한 디렉터리를 생성하고, "git init"으로 저장소 초기화
- .gitignore 파일을 이전 페이지 예제와 동일하게 작성

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho
$ git init
Initialized empty Git repository in F:/suho/.git/

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ vi .gitignore

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master

No commits yet

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

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

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



관리 대상 제외 실습 환경 - 5



- 제외 파일 commit

- 제외 대상 파일인 .gitignore를 관리 대상으로 추가
- 실습 도중 수정 등을 방지하기 위해 먼저 관리 상태로 추가
- 이전 정보가 남은 상태에서는 정확한 확인이 어려울 수 있음

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git add .gitignore
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git commit -m "apply .gitignore"
[master (root-commit) 51b670a] apply .gitignore
1 file changed, 12 insertions(+)
create mode 100644 .gitignore

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
nothing to commit, working tree clean

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

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



관리 대상 제외 디렉터리 실습 - 1



• 실습 디렉터리 생성

- 제외 대상 디렉터리가 어떻게 적용되는지 확인하기 위해 생성
- 제외 대상 디렉터리인 dir/, test/, test/view/ 디렉터리 생성
- 제외 대상 디렉터리가 아닌 project/ 디렉터리 생성
- 확인이 필요한 project/dir/ 디렉터리 생성

```
MINGW64:/f/suho

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

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

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ mkdir test/view

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

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ mkdir project/dir

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



관리 대상 제외 디렉터리 실습 - 2



• 디렉터리의 제외 적용 확인 - 1

- 디렉터리 제외 확인을 위해 제외 대상 파일이 아닌 파일을 생성
 - 확장자와 파일명 제외에 해당하지 않도록 간단하게 "hong.c"로 생성
- 제외 대상이 아닌 디렉터리는 추적되므로 파일을 생성하지 않음
 - 파일들을 생성하고 git status로 저장소 상태 확인

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch dir/hong.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch test/hong.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch test/view/hong.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    test/

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

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



관리 대상 제외 디렉터리 실습 - 3



• 디렉터리의 제외 적용 확인 - 2

- dir/은 명확하게 제외되었으나, test/는 제외 확인이 어려움
- 확인을 위해 test/ 디렉터를 Staged 상태로 변경하면 파일이 보임
- test/view에 생성된 파일은 제외되었으나 test/에 생성된 파일은 적용
- test/view/ 경로에서 test/는 제외 대상이 아님을 알 수 있음
- 하위 디렉터리 포함 제외 경로는 해당 경로만을 제외함
- test/로 지정하면 전체 제외로 지정됨

```
MINGW64:/f/suho

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ git add test/

unangel@DESKTOP-I8OQG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   test/hong.c

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



관리 대상 제외 디렉터리 실습 - 4



• 디렉터리의 제외 적용 확인 - 3

- 제외 대상 디렉터리 내부에 다수 디렉터리가 존재할 경우의 처리
- test/ 내부에 디렉터리와 파일을 추가로 생성하면 추적 대상으로 적용

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ mkdir test/source

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ mkdir test/pptx

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch test/source/hong.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch test/pptx/hong.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   test/hong.c

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    test/pptx/
    test/source/

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



관리 대상 제외 디렉터리 실습 - 5



• 디렉터리의 제외 적용 확인 - 4

- 다수의 하위 디렉터를 갖는 디렉터리의 제외 대상 적용
 - 디렉터리마다 경로를 입력하여 제외 대상이 되는 디렉터리 입력이 필요
 - 최상위 디렉터를 지정하면 하위 디렉터리 여부와 관계 없이 모두 제외
- 즉, 디렉터리 경로는 지정한 경로에서만 제외가 적용
 - test/view/ 형식의 제외 경로에 test/는 제외 대상이 아님
 - test/ 형식의 입력은 test/ 내부의 모든 디렉터리와 파일을 제외
 - pptx/, source/의 상위 디렉터리 test/는 제외가 아니므로 추적 대상에 포함
 - 다수의 디렉터리 중 하나의 디렉터리만 적용하려면 제외 대상을 모두 지정해야 함

```
MINGW64:/f/suho
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    test/pptx/
    test/source/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls test/
hong.c pptx/ source/ view/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```




관리 대상 제외 디렉터리 실습 - 6



• 대상 파일 확인

- "git add"로 모든 파일/디렉터리 추가하여 확인
- test/view/ 디렉터리를 제외한 모든 디렉터리의 파일이 추가됨
- 디렉터리 경로를 제외할 경우 주의해야 함
- 대상/제외 경로에 따라 테스트를 통해 반드시 확인할 것

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git add .

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   test/hong.c
    new file:   test/pptx/hong.c
    new file:   test/source/hong.c

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



관리 대상 제외 파일 실습 - 1



- 파일 실습을 위해 commit
 - 이전 제외 정보가

```
MINGW64:/f/suho
unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   test/hong.c
    new file:   test/pptx/hong.c
    new file:   test/source/hong.c

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git commit -m "add hong.c"
[master 6387bf6] add hong.c
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test/hong.c
 create mode 100644 test/pptx/hong.c
 create mode 100644 test/source/hong.c

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
nothing to commit, working tree clean

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ |
```



관리 대상 제외 파일 실습 - 2



- 파일 확장자와 이름 제외

- 파일 확장자가 .text라면 디렉터리와 관계 없이 제외
 - *.text 제외 규칙으로 a1.text, zip.text 파일은 추적에서 제외
- 고정된 파일 이름 또한 확장자와 관계 없이 제외
 - tmp.* 제외 규칙으로 tmp.c , tmp.cpp 파일은 추적에서 제외

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch a1.text ; touch zip.text ; touch tmp.c ; touch tmp.cpp

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   test/hong.c
    new file:   test/pptx/hong.c
    new file:   test/source/hong.c

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



관리 대상 제외 파일 실습 - 3



• 파일 제외 주의

- 파일 제외 규칙은 앞서부터 적용되어 해당 파일 제외 규칙을 확인
- *.text 규칙은 파일명과 관계 없이 마지막에 .text를 만나면 적용
 - *.text.txt 파일의 확장자까지는 일치하나 뒤의 .txt가 존재하여 불일치
- tmp.* 규칙도 시작부분이 동일하면 뒤의 내용은 관계 없이 제외

```
MINGW64:/f/suho
unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch a2.text.txt ; touch tmp.cpp.txt

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   test/hong.c
    new file:   test/pptx/hong.c
    new file:   test/source/hong.c

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    a2.text.txt

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```



관리 대상 제외 파일 실습 - 4



- ***(와일드 카드)를 적용한 파일명**

- net*.o 규칙은 파일명 시작이 net이고, 확장자가 o인 모든 파일 제외
- 확장자가 "pptx"인 파일과 "c"인 파일은 관리 대상임을 확인
- "network.o", "Network.o" 파일은 대상에서 제외됨을 확인
- 파일명 전체와 규칙을 비교하여 처리하는 것을 알 수 있음

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch network.o ; touch netWork.pptx ; touch Network.o ; touch network.c

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    network.pptx
    network.c

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

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



관리 대상 제외 파일 실습 - 5



• 파일명 대/소문자 구별

- UNIX/Linux 계열 운영체제는 대/소문자를 모두 구분하므로 주의
- 윈도우 운영체제는 파일명에 대/소문자를 구분하지 않음
- "network.o"와 "Network.o"는 동일한 파일로 인식
- "network.o" 파일을 하나만 확인할 수 있으나 Linux에서는 2개로 표시

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
dir/  netWork.pptx  network.c  network.o  project/  test/

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

```
wikim@wikim-VirtualBox: ~/work

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

wikim@wikim-VirtualBox:~/work$ touch network.o ; touch netWork.pptx ; touch Network.o ; touch network.c
wikim@wikim-VirtualBox:~/work$ ls
Network.o  netWork.pptx  network.c  network.o
wikim@wikim-VirtualBox:~/work$
```



관리 대상 제외 파일 실습 - 6



- ***(와일드 카드)를 적용한 파일/확장자**
 - 이전 정보의 출력 방지를 위해 사용한 파일을 삭제
 - "wide*.*" 규칙은 파일명과 확장자에 모두 적용되는 규칙
 - *는 특수기호까지 포함하여 적용되므로 "Wide_Old.c" 파일도 제외
 - 확장자의 *는 파일명 시작이 "wide" 이기만하면 모두 적용되는 규칙

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ rm -rf net*

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch widao.c ; touch widETv.o ; touch Wide_Old.c ; touch aWide.pc

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        aWide.pc
        widao.c

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

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



관리 대상 제외 파일 실습 - 7



• * 사이의 파일명

- 이전 정보의 출력 방지를 위해 사용한 파일을 삭제
- “*bio*.k”은 파일명 사이에 bio가 포함된 파일명 제외 규칙
- 예제로 생성한 모든 파일이 제외 되었음을 확인할 수 있음
- 즉, *는 다양한 문자열과 문자가 없는 것도 포함하는 것을 확인
- 파일명에 특정 문자열이 있고, 확장자가 “k”인 파일을 제외하는 규칙

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ rm -rf *.c *.pc *.o

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch bio.k ; touch abioop.k ; touch abio.k ; touch bio_b.k

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
nothing to commit, working tree clean

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




관리 대상 제외 파일 실습 - 8



- 특수 기호와 모든 파일 확장자

- 이전 예제에서 관리 대상이 없으므로 그대로 다음 예제 진행
- `"*key_*.*"` 규칙은 `"key_"` 문자열이 포함된 모든 파일을 의미
- `*`는 모두 또는 아무것도 없는 것을 의미
 - `"key_."` 파일 또한 제외 규칙에 포함됨
- 예제에서는 `KEY_` 파일을 제외하면 모두 제외되는 것을 확인
 - 윈도우에서는 대/소문자 구분이 없고, `."`이 없으므로 대상이 아님

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ touch key_.file ; touch A_key_a.c ; touch KEY_ ; touch key_.

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        KEY_

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

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



Untracked 및 제외 파일 삭제 - 1



• 추가 파일의 삭제

- 수정된 상태가 아니라 새롭게 추가된 파일들이 필요 없는 경우가 존재
- 테스트 등을 위해 새롭게 추가 또는 복사된 파일들의 삭제가 필요
- "git clean" 명령으로 저장소의 파일들을 삭제할 수 있음
- 테스트를 진행한 현재 디렉터리의 상태는 아래와 같음

```
MINGW64:/f/suho

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
A_key_a.c  abio.k    bio.k     dir/      key_.file project/
KEY_       abioop.k  bio_b.k   key_.     key_a.cpp test/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ echo 1; ls dir/ ; echo 2; ls project/ ; echo 3 ; ls project/dir/ ; echo 4; ls test/
1
hong.c
2
dir/
3
4
hong.c pptx/ source/ view/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$
```



Untracked 및 제외 파일 삭제 - 2



• 파일 삭제 - 1

- "git clean -f"로 모든 Untracked 파일을 삭제할 수 있음

```
MINGW64:/f/suho
unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        KEY_

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

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git clean -f
Removing KEY_

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
nothing to commit, working tree clean

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
A_key_a.c  abioop.k  bio_b.k  key_.      key_a.cpp  test/
abio.k     bio.k     dir/     key_.file  project/

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



Untracked 및 제외 파일 삭제 - 2



• 파일 삭제 - 2

- 제외 대상으로 지정된 파일들은 불필요하게 남아 있을 수 있음
- "git clean -f -x"로 제외 대상인 파일들을 모두 삭제할 수 있음

```
MINGW64:/f/suho
unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git status
On branch master
nothing to commit, working tree clean

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
A_key_a.c  abioop.k  bio_b.k  key_.      key_a.cpp  test/
abio.k     bio.k     dir/      key_.file  project/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git clean -f -x
Removing A_key_a.c
Removing abio.k
Removing abioop.k
Removing bio.k
Removing bio_b.k
warning: failed to remove key_.: No such file or directory
Removing key_.file
Removing key_a.cpp

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ |
```

파일이 명확하지 않은 경우는 삭제 실패



Untracked 및 제외 파일 삭제 - 3



• 디렉터리 삭제

- 제외 대상이나 비어 있는 디렉터리를 모두 삭제 가능
- "git clean -f -d -x"로 제외 대상 디렉터리와 내부 파일 모두 삭제
- 제외 파일이 포함되어 있는 "dir/" 디렉터리도 삭제
- 디렉터리 내에 파일이 없는 "project/", "project/dir/" 디렉터리도 삭제

```
MINGW64:/f/suho
unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
dir/  key_.  project/  test/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git clean -f -d -x
Removing dir/
warning: failed to remove key_.: No such file or directory
Removing project/
Removing test/view/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
key_.  test/

unange1@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ |
```



Untracked 및 제외 파일 삭제 - 4



- “git clean” 명령어

- 추적되지 않는 모든 작업 디렉터리들과 파일들을 삭제
- 현재 디렉터리 기준으로 시작되며 재귀적으로 모두 삭제가 가능
 - “-f” : 파일을 강제로 삭제하라는 옵션
 - “-d” : 옵션을 명시하지 않으면 현재 디렉터리만 기준으로 처리
옵션을 명시하면 재귀적으로 모든 디렉터리를 검사하여 디렉터리도 삭제
 - “-x” : .gitignore 파일에 의해 추적되지 않는 파일도 모두 삭제
 - “-X” : .gitignore에 의해 무시되는 모든 파일들을 삭제
 - “-n” : 주어진 옵션으로 삭제될 파일 목록을 출력하는 옵션. 실제 삭제가 발생하지 않음

```
MINGW64:/f/suho

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ ls
A_key_a.c  abio.k    bio.k     dir/      key_.file  project/
KEY_       abioop.k  bio_b.k   key_.     key_a.cpp  test/

unangel@DESKTOP-I80QG85 MINGW64 /f/suho (master)
$ git clean -X -n
Would remove A_key_a.c
Would remove abio.k
Would remove abioop.k
Would remove bio.k
Would remove bio_b.k
Would remove key_.
Would remove key_.file
Would remove key_a.cpp
```