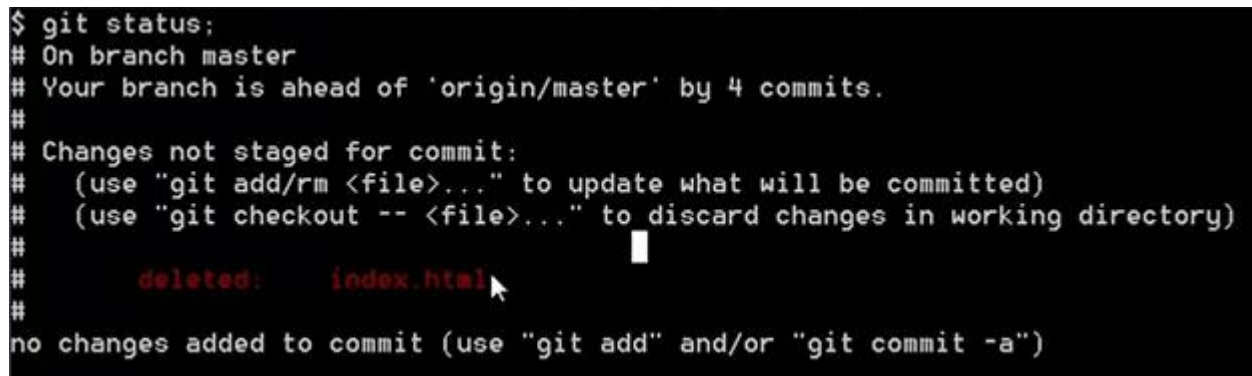


작업 디렉토리의 파일 삭제

- rm 명령어를 사용하여 버전관리 되고 있는 파일(trackd file)을 삭제하면 아래와 같이 표시된다.

```
ls -al
rm index.html
git status
```

A terminal window with a black background and white text. The output of the 'git status' command is shown. It indicates the current branch is 'master' and it is ahead of 'origin/master' by 4 commits. Under the section 'Changes not staged for commit:', it lists 'deleted: index.html' in red text. A mouse cursor is pointing at 'index.html'. The terminal also shows instructions on how to use 'git add/rm' and 'git checkout' to update or discard changes.

```
$ git status;
# On branch master
# Your branch is ahead of 'origin/master' by 4 commits.
#
# Changes not staged for commit:
#   (use "git add/rm <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#       deleted:   index.html
#
no changes added to commit (use "git add" and/or "git commit -a")
```

- 삭제 파일 복구

index.html 파일은 버전 관리되고 있는 파일이므로 rm 명령어를 사용하여 파일을 삭제하면 더이상 커밋대상이 아닌 파일, 즉 버전관리 대상이 아닌 파일로 인식이 된다.

실수로 작업 디렉토리에 있는 파일을 삭제 했을 때는 checkout 명령어를 사용하여 작업 디렉토리에 있는 변경사항들을 취소할 수 있다. 다시 말해 삭제된 파일을 다시 복구 할 수 있다.

```
git checkout -- index.html
```

작업 디렉토리에 있는 파일(tracked file)을 삭제 할 때는 아래와 같이 git rm 명령어를 사용하여 삭제를 하며 아래와 같이 표시된다.

```
git rm index.html
```

※ 원본이 삭제 될 수 있으므로 주의 !!

```

egoing@WIN-NDPPAPTAE3 /C/dev/codingeverybody (master)
$ git status;
# On branch master
# Your branch is ahead of 'origin/master' by 4 commits.
#
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       deleted:    index.html
#

```

git rm 명령어로 삭제한 파일도 tracked 되어 staging 영역에 올라가며 커밋 대상이 되므로 아래와 같이 commit을 해야 파일이 온전히 삭제된다.

```
git commit -m 'delete index.html'
```

```

egoing@WIN-NDPPAPTAE3 /C/dev/codingeverybody (master)
$ git commit -m 'delete index.html'
>
[master 0f24b37] delete index.html
1 file changed, 7 deletions(-)
delete mode 100644 index.html

```

- Git 추적에서 제외하기

또 staging Area에서만 제거하고 워킹 디렉토리에 있는 파일은 지우지 않고 남겨둘 수 있다. 다시 말해 하드디스크에 있는 파일은 그대로 두고 Git만 추적하지 않게 한다. 이것은 .gitignore 파일에 추가하는 것을 빼먹었거나, 대용량 로그 파일이나 컴파일된 파일인 .a 파일과 같은 것을 실수로 추가했을 때 아주 유용하다. --cached 옵션을 사용하여 명령을 실행한다.

```
git rm --cached README
```

여러개의 파일이나 디렉토리를 한꺼번에 삭제할 수도 있다.

```
git rm log/W*.log
```

이 명령은 log/ 디렉토리에 있는 .log 파일을 모두 삭제한다.

```
git rm W*~
```

위 명령은 ~로 끝나는 파일을 모두 삭제한다.

