

```
$ git init
```

파일을 추적하려면 해당 파일을 저장소에 추가하고 커밋(commit)해야 한다. Git 커밋을 통해 시간의 흐름에 따른 파일 변경 내용을 추적할 수 있다. 예를 들어 한 시간 전에 커밋한 파일과 현재 버전의 파일을 비교할 수 있다.

## 커밋

커밋은 특정 시점의 파일이나 폴더의 상태를 저장하는 것으로 각 커밋은 고유한 식별 코드를 지닌다.


커밋이 무엇인지 알았으니 다음과 같이 하나의 파일을 실제로 커밋해보자.<sup>2</sup>

```
$ git add hello.txt
$ git commit -m "Initial commit"
```

만약 파일이 변경되면 다음 명령을 사용해 파일 상태를 추적할 수 있다.

```
$ git status
```

명령을 실행하면 [그림 1-1]과 같은 화면을 볼 수 있다.



```
youngestdev@Abduls-MacBook-Air:~/Documents/FastAPI-Book
+ FastAPI-Book git:(main) $ git add hello.txt
+ FastAPI-Book git:(main) $ git commit -m "Initial commit"
[main (root-commit) eda7e6c] Initial commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 hello.txt
+ FastAPI-Book git:(main) echo "This is a new addition to the file" > hello.txt
+ FastAPI-Book git:(main) $ git status
On branch 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:   hello.txt

no changes added to commit (use "git add" and/or "git commit -a")
+ FastAPI-Book git:(main) $ git diff hello.txt
+ FastAPI-Book git:(main) $
```

그림 1-1 Git 명령 실행

<sup>2</sup> 옮긴이: `git init`을 실행한 폴더에서 명령을 실행해야 한다. 만약 `hello.txt`라는 파일이 없다면 빈 파일을 만들면 된다.