
Open source Software
오픈소스 소프트웨어
29. [실습] 버전 되돌리기 reset

학습 개요

1. 과거 버전으로 되돌리기
2. reset 옵션 --hard --mixed --soft
3. 작업 디렉토리(폴더), 스테이징 영역, 깃 저장소

학습 목표

1. 과거 버전으로 완전히 되돌리는 방법을 이해할 수 있다.
2. 버전 되돌리는 reset 명령을 수행할 수 있다.
3. reset과 checkout 차이를 이해하고 활용할 수 있다.

LESSON 01

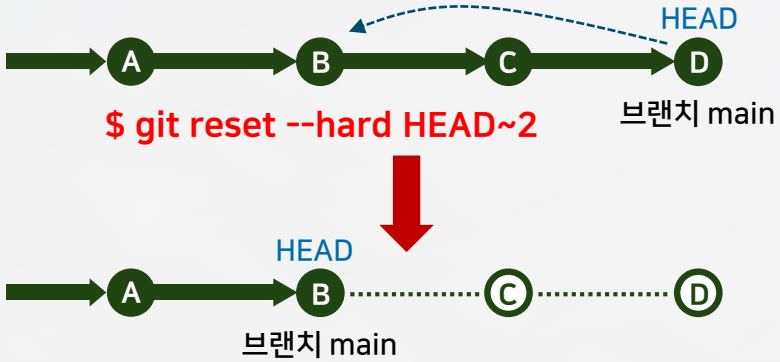
버전 되돌리기 reset

1 버전 되돌리기 reset

기능 reset 이해

커밋 이력에서 이전 특정 커밋으로 완전히 되돌아가는(roll back) 방법

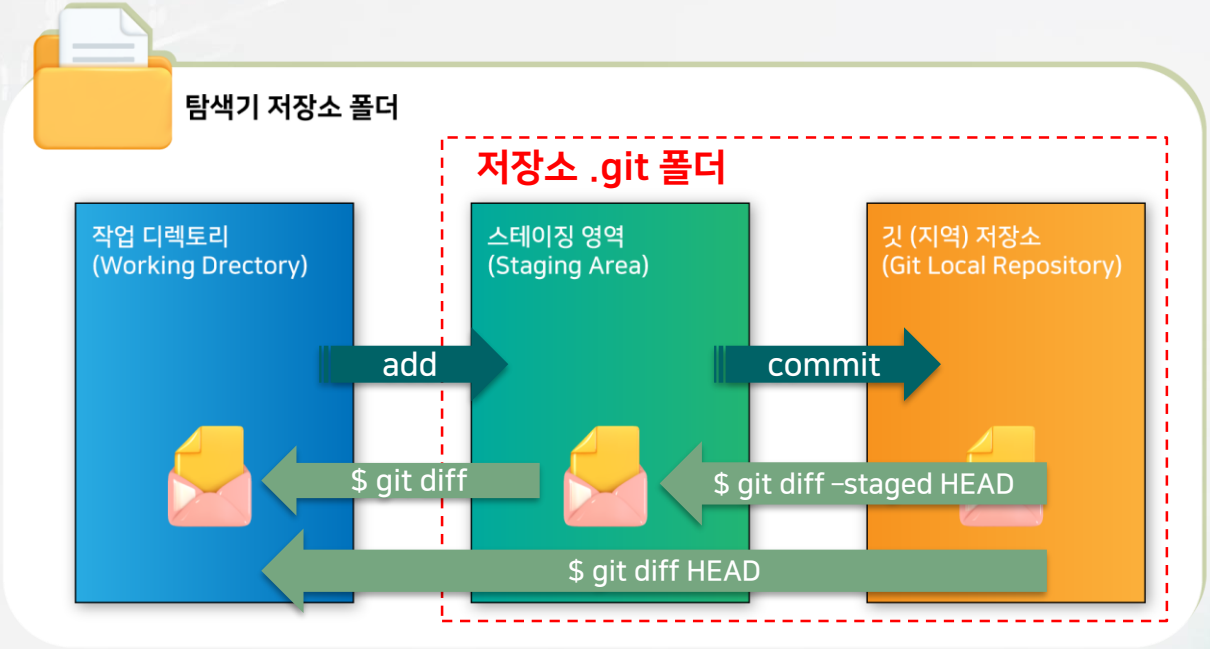
- 시계를 뒤로 맞추는 '타임 머신'과도 같음
- 이동되는 해당 커밋 이후의 이력은 모두 사라지므로 주의가 필요
  - 새로운 커밋이 생성되지 않음
- 깃 저장소는 이전 커밋 내용으로 수정
  - 다만 reset 이전에 있던 작업 폴더와 스테이지 영역을 내용을 어떻게 유지할지가 관건



1 버전 되돌리기 reset

깃 3 영역

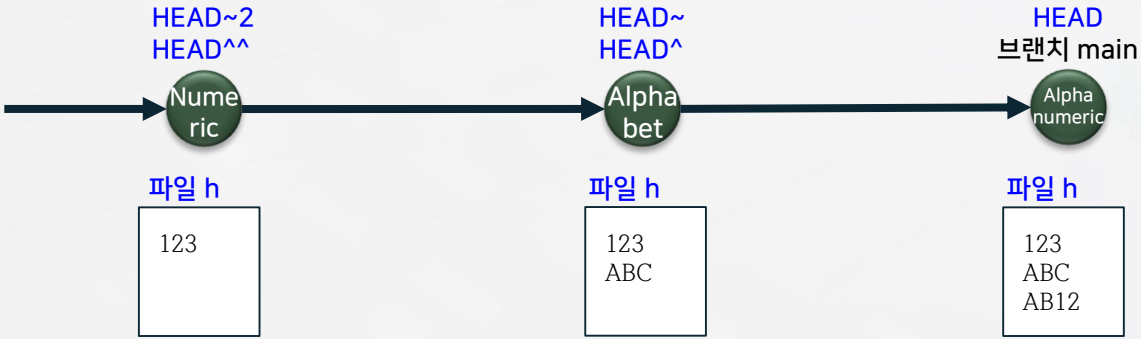
- 작업 디렉토리(폴더), 스테이징 영역, 깃 저장소



[실습] 버전 되돌리기 reset

간편히 준비, 현재 상태

작업 폴더 (파일 h)	스테이지 영역 (파일 h)	깃 저장소 (파일 h)
123 ABC AB12 [1, 2] {1, 2}	123 ABC AB12 [1, 2]	123 ABC AB12



1 버전 되돌리기 reset

준비 단계까지

저장소 grst

```
$ git init grst
$ cd grst

$ echo 123 > h
$ git add h
$ git commit -m Numeric

$ echo ABC >> h
$ git commit -am Alphabet
$ echo AB12 >> h
$ git commit -am Alphanumeric

$ echo '[1, 2]' >> h
$ git add h
$ echo '{1, 2}' >> h
$ git status
```

1 버전 되돌리기 reset

버전 되돌리기 reset 전에 수행

- 버전 되돌리기(옵션 --hard)를 하면 현재 상태가 모두 제거되므로
  - 현재의 작업 디렉토리와 스테이징 영역을 임시 저장에 저장
    - $ git stash
  - 확인
    - $ git stash list
  - 현재 상태 확인
    - $ git status

```
$ git stash
$ git stash list
$ git status
```

1 버전 되돌리기 reset

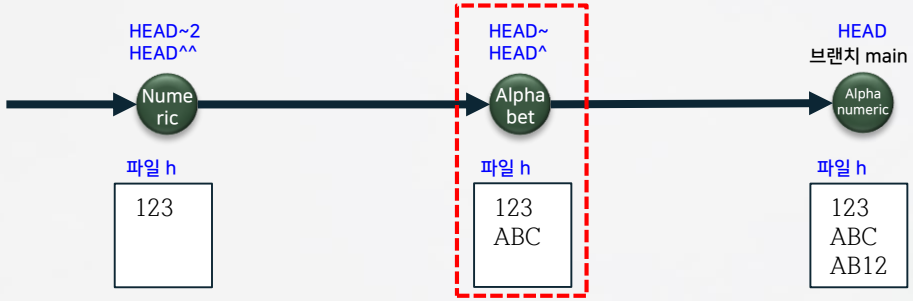
reset 옵션 --hard

$ git reset --hard HEAD~

- 지정된 HEAD~의 내용으로 작업 폴더와 스테이지 영역, 깃 저장소가 모두 복사·수정
  - reset 전에 있던 작업 폴더와 스테이지 영역에 작업 내용이 모두 사라지므로 주의가 필요

```
$ git log --oneline
$ git reset --hard HEAD~

$ cat h
$ git log --oneline
$ git diff
$ git diff HEAD
$ git diff --staged
```



작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC AB12

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC	123 ABC	123 ABC

```
$ git reset --hard HEAD~
```

다시 원래 상태로 복원

reset 이전의 커밋 ID를 ORIG_HEAD에 저장

- $ git reset --hard ORIG_HEAD

```
$ git reset --hard ORIG_HEAD
$ git status
$ cat h

$ git log --oneline
$ git diff
$ git diff HEAD
$ git diff --satged
```

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC	123 ABC	123 ABC

$ git reset --hard ORIG_HEAD

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC AB12

1 버전 되돌리기 reset

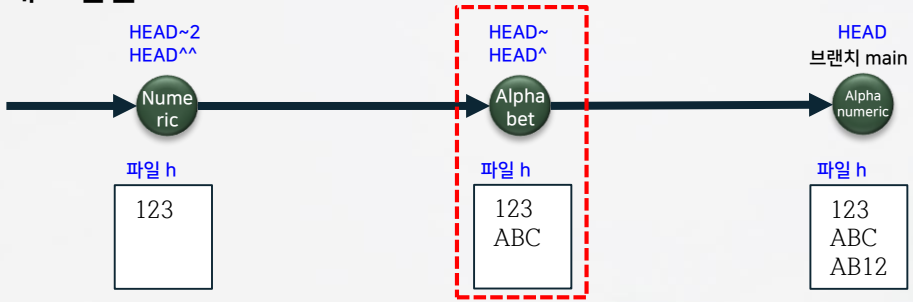
reset 옵션 --mixed

$ git reset --mixed HEAD~

- 지정된 HEAD~(커밋 메시지 Alphabet)의 내용으로 스테이지 영역과 깃 저장소가 모두 복사·수정
  - 커밋 메시지 Alphanumeric의 로그 이력과 함께 당시의 스테이지 영역, 깃 저장소 내용이 모두 사라짐, 다만 작업 폴더의 내용은 이전 그대로 남음

```
$ git log --oneline
$ git reset --mixed HEAD~

$ cat h
$ git log --oneline
$ git diff
$ git diff HEAD
$ git diff --staged
```



작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC AB12

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC	123 ABC

```
$ git reset HEAD~
```

다시 원래 상태로 복원

reset 이전의 커밋 ID를 ORIG_HEAD에 저장

- $ git reset --hard ORIG_HEAD

```
$ git reset --hard ORIG_HEAD
$ git status
$ cat h

$ git log --oneline
$ git diff
$ git diff HEAD
$ git diff --satged
```

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC	123 ABC

$ git reset --hard ORIG_HEAD

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC AB12

1 버전 되돌리기 reset

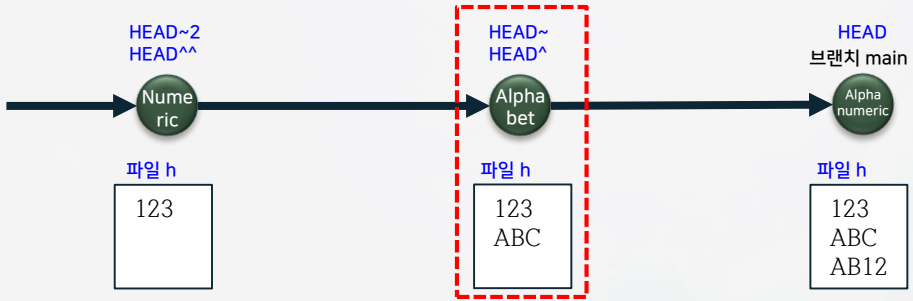
reset 옵션 soft

$ git reset --soft HEAD~

- 지정된 HEAD~(커밋 메시지 Alphabet)의 내용으로 깃 저장소만 복사·수정
- 커밋 메시지 Alphanumeric의 로그 이력은 사라짐
  - 작업 폴더와 스테이지 영역의 내용이 모두 이전 그대로 남음

```
$ git log --oneline
$ git reset --soft HEAD~

$ cat h
$ git log --oneline
$ git diff
$ git diff HEAD
$ git diff --staged
```



작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC AB12

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC

$ git reset --soft HEAD~

1 버전 되돌리기 reset

다시 원래 상태로 복원

reset 이전의 커밋 ID를 ORIG_HEAD에 저장

- $ git reset --hard ORIG_HEAD

```
$ git reset --hard ORIG_HEAD
$ git status
$ cat h

$ git log --oneline
$ git diff
$ git diff HEAD
$ git diff --satged
```

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC

$ git reset --hard ORIG_HEAD

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC AB12

1 버전 되돌리기 reset

작업 디렉토리와 스테이징 영역 수정

3회 커밋과 이후의 다음 상태로 복원

- 작업 폴더와 스테이징 영역을 임시 저장에 저장해 놓았음
- 임시 저장을 다시 불러오면 됨
  - 스테이징 영역까지 복원하려면 옵션 --index 사용

$ git stash apply --index

```
$ git stash apply --index
$ git status
$ cat h
$ git log --oneline

$ git diff
$ git diff HEAD
$ git diff --satged
```

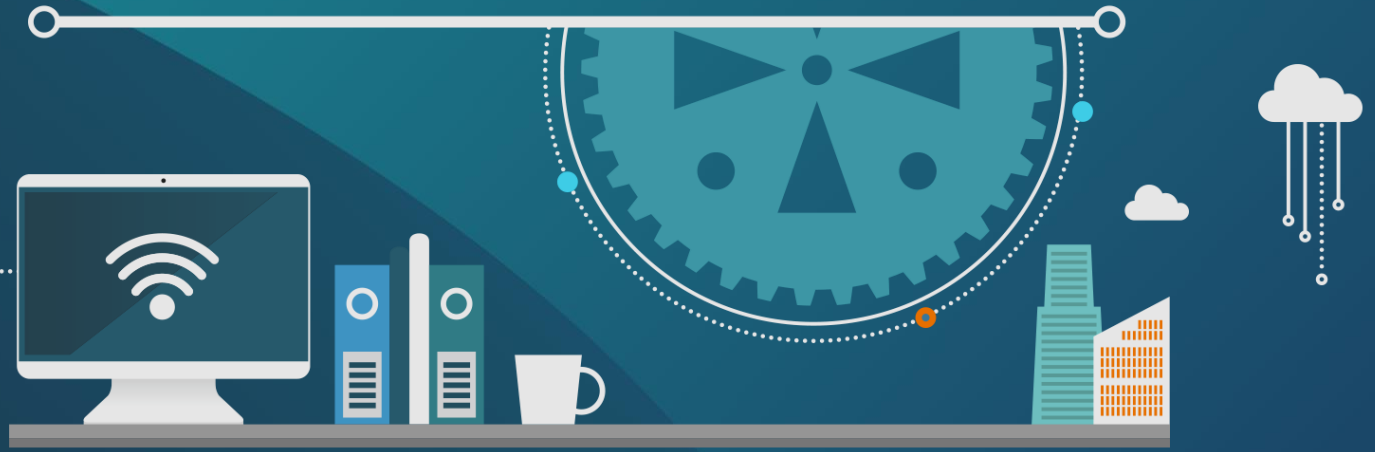
작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12	123 ABC AB12	123 ABC

$ git stash apply --index

작업 폴더(파일 h)	스테이지 영역(파일 h)	깃 저장소(파일 h)
123 ABC AB12 [1, 2] {1, 2}	123 ABC AB12 [1, 2]	123 ABC AB12

LESSON 02

checkout과 reset 비교

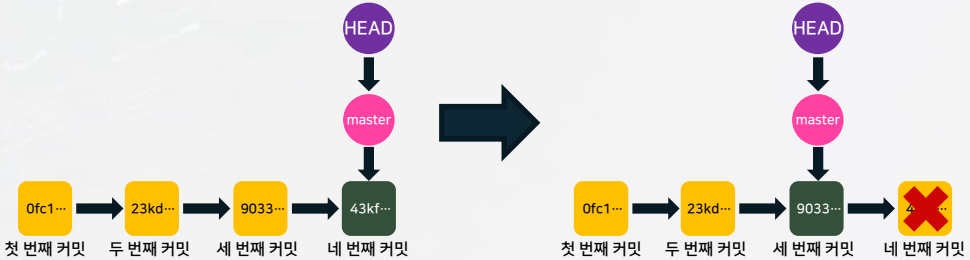


2 checkout과 reset 비교

reset과 checkout 비교

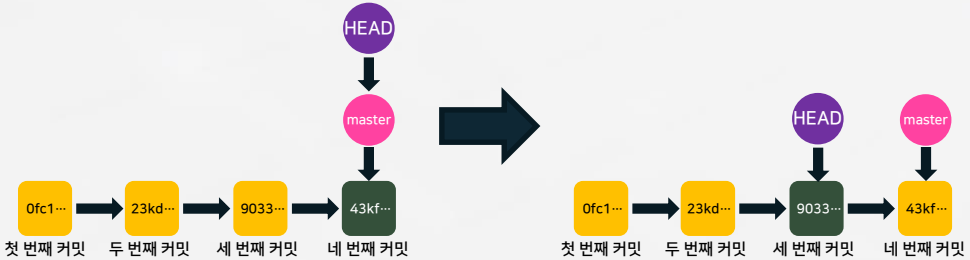
$ git reset 9033

- 브랜치의 마지막 커밋을 수정하는 명령



$ git checkout 9033

- HEAD 포인터를 브랜치 마지막 커밋 이전으로 이동하는 명령



2 checkout과 reset 비교

checkout으로 과거 여행 복습

상태가 깔끔해야 checkout 가능

- Nothing to commit, working tree clean

```
$ git checkout HEAD~
오류발생

$ git reset --hard HEAD
$ git status

$ git checkout HEAD~
$ cat h
$ git log --oneline
$ git checkout main

$ git checkout HEAD~2
$ cat h
$ git log --oneline
$ git checkout -
```

Summary

HEAD~2의 내용으로 작업 디렉토리와 스테이징 영역, 깃 저장소에 복사

- $ git reset --hard HEAD~2

HEAD~2의 내용으로 스테이징 영역과 깃 저장소에 복사

- $ git reset --mixed HEAD~2

HEAD~2의 내용으로 깃 저장소에 복사

- $ git reset --soft HEAD~2

이전에 수행한 reset을 바로 취소하는 명령

- $ git reset --hard ORIG_HEAD