2-5. 작업 되돌리기

앞에서는 수정한 파일을 스테이지에 올리고 commit하는 방법까지 살펴보았다. 이제부터는 스테이지에 올렸던 파일을 내리거나 commit을 취소하는 등 각 단계로 돌아가는 방법에 대해 알아보자.

**\* 작업 트리에서 수정한 파일 되돌리기 - git checkout**

파일을 수정한 뒤 소스가 정상적으로 동작하지 않는 등의 이유로 **수정한 내용을 취소하고 가장 최신 버전 상태로 되돌려야 할 때**가 있다. 이럴 때 일일이 수정한 소스를 찾아 직접 되돌려야 한다면 아주 번거로울 것이다. 이럴 때 **checkout 명령**을 사용하면 작업 트리에서 수정한 내용을 쉽게 취소할 수 있다. hello.txt 파일을 수정하면서 checkout 명령의 사용법을 알아보자.

1. 먼저 빔을 열고 숫자 ‘3’을 ‘three’로 수정하자.

2. 깃 상태를 확인해보자.

텍스트이(가) 표시된 사진

자동 생성된 설명

**3. checkout 명령 다음에 --를 붙이고 한 칸 띈 다음 파일 이름을 써보자**.



4. 깃 상태와 hello.txt 파일의 내용을 보면 수정 내용이 사라진 것을 확인할 수 있다.

**\* Staging 되돌리기 - git reset HEAD (파일 이름)**

앞에서는 파일의 수정을 취소하고 원래대로 되돌렸다. **이번에는 수정된 파일을 Staging 했을 때, Staging을 취소하는 방법**을 알아보자.

1. 빔을 사용해 hello2.txt를 수정해보자. 기존 내용을 삭제하고 A, B, C, D를 입력해보자.

2. hello2.txt 파일을 Stage에 올려보자.

**2. git reset 명령을 사용해 Stage에서 hello2.txt를 내려보자.**

텍스트이(가) 표시된 사진

자동 생성된 설명

깃 상태를 확인해보면 파일이 Stage에 올라가기 전으로 돌아온 것을 확인할 수 있다.

**(참고) HEAD 다음에 파일 이름을 지정하지 않으면 Stage에 있는 모든 파일을 되돌린다.**

**\* 최신 commit 되돌리기 - git reset HEAD^**

이번에는 수정된 파일을 Staging하고 commit까지 했을 때, **가장 마지막에 한 commit을 취소하는 방법**을 알아보자.

1. hello2.txt 파일에 문자 ‘E’를 추가해보자.

2. git status 명령을 사용해 Staging과 commit을 함께 실행해보자.

텍스트이(가) 표시된 사진

자동 생성된 설명

3. 최신 commit을 돌리려면 **git reset 명령 다음에 HEAD^**를 붙인다. **HEAD^는 현재 HEAD가 가리키는 branch의 최신 commit을 가리킨다.** 해당 명령을 사용하면 commit도 취소되고, Stage에서도 내려진다. 즉, 취소한 파일이 작업 트리에만 남는 것이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

git log 명령으로 확인해도 **commit message가 ‘message4’인 commit이 사라진 것을 볼 수 있다.**

**(참고) git reset 명령의 option**

--soft HEAD^: **최근 commit을 하기 전 상태**로 작업 트리를 되돌린다.

--mixed HEAD^: **최근 commit과 Staging을 하기 전 상태**로 작업 트리를 되돌린다. 옵션 없이 git reset 명령을 사용하면 이 옵션을 기본으로 적용된다.

--hard HEAD^: **최근 commit과 Staging, 파일 수정을 하기 전 상태**로 작업 트리를 되돌린다. 이 옵션으로 되돌린 경우 내용은 복구할 수 없다.

**\* 특정 commit으로 되돌리기 - git reset (commit hash)**

깃에는 파일을 수정하고 commit할 때마다 저장된 버전들이 쌓여 있다. **git reset HEAD^ 명령**으로 최신 commit을 되돌릴 수도 있지만 **특정 버전으로 되돌린 다음 그 이후 버전을 삭제**할 수도 있다. 특정 commit으로 되돌릴 때는 **git reset 명령 다음에 commit hash를 사용**한다.

1. git reset 명령을 연습하기 위해 **몇 개의 commit을 만들어 보자**. 빔을 사용해 hello-git 디렉터리에 rev.txt를 만들고 문자 ‘a’를 입력해보자.

2. rev.txt를 Staging하고 commit(commit message: **R1**)한다.

3. rev.txt에 문자 ‘b’를 추가하고 commit(commit message: **R2**) 한다. 이후 ‘c’, ‘d’도 추가해 각각 commit(Each commit message: **R3, R4**)해준다. (총 4번의 commit을 수행했다.)

4. 4개의 commit중 R2라는 message가 붙은 commit으로 돌아가보자. 단, **reset에서 commit hash를 사용해 되돌릴 때 주의할 점**이 있다. 예를 들어, **reset A**를 입력하면 A commit을 리셋하는 것이 아니라 최근 commit을 A로 리셋한다. **즉, A commit을 삭제하는 것이 아니라 A commit 이후에 만들었던 commit을 삭제하고 A commit으로 이동한다.**

git log 명령의 결과 화면에서 R2 commit의 commit hash를 복사하고 **git reset 명령**을 수행해보자.

텍스트이(가) 표시된 사진

자동 생성된 설명

위와 같이 HEAD가 commit hash 위치로 옮겨졌다는 문구가 출력된다.

--hard 옵션을 사용했으므로 rev.txt의 내용도 a, b만 들어있게 된다.

**\* commit 삭제하지 않고 되돌리기 - git revert**

commit으로 되돌릴 때 수정했던 것을 삭제해도 된다면 git reset 명령을 사용하면 된다. 하지만 **나중에 사용할 것을 대비해 commit을 되돌리더라도 취소한 commit을 남겨두어야 할 때**가 있다. 이때는 **git revert**라는 명령을 사용한다.

1. rev.txt에 ‘e’를 추가해 commit(commit message: R5)을 수행하자.

2. 가장 최근에 commit한 R5 버전을 취소하고 R2로 되돌아가보자.

reset의 경우 R2로 가기 위해 reset 명령 뒤에 R2의 commit hash를 지정했지만 revert 명령의 경우 revert 명령 뒤에 **취소하려고 하는 버전, 즉 R5의 commit hash를 지정**한다.

**revert 명령을 실행하면 깃을 설치할 때 지정했던 기본 편집기가 자동으로 나타나면서 commit message를 입력할 수 있다.** commit message 위에는 어떤 버전을 revert 했는지 나타난다. 문서 맨 위에 revert하면서 추가로 남겨둘 내용이 있다면 입력하고 저장한다.



3. git log를 입력해 확인해보자.

텍스트이(가) 표시된 사진

자동 생성된 설명

log에 R5를 revert한 새로운 commit이 생겼다. 그리고 기존의 R5 역시 사라지지 않는다.

이때 rev.txt의 ‘e’ 문자 역시 없어진다.