Git05_revert

Git 온라인 리소스

Git - git-revert Documentation

- git revert 는 특정 커밋의 변경 사항을 취소하는 새로운 커밋을 생성합니다.
- 원본 커밋은 삭제되지 않고, 새로운 커밋이 추가됩니다.
- 새로운 커밋은 원본 커밋의 변경 사항을 "반대로" 적용합니다.
- git revert는 히스토리를 변경하지 않으므로, 이미 원격 저장소에 푸시된 커밋을 취소할 때 유용합니다.

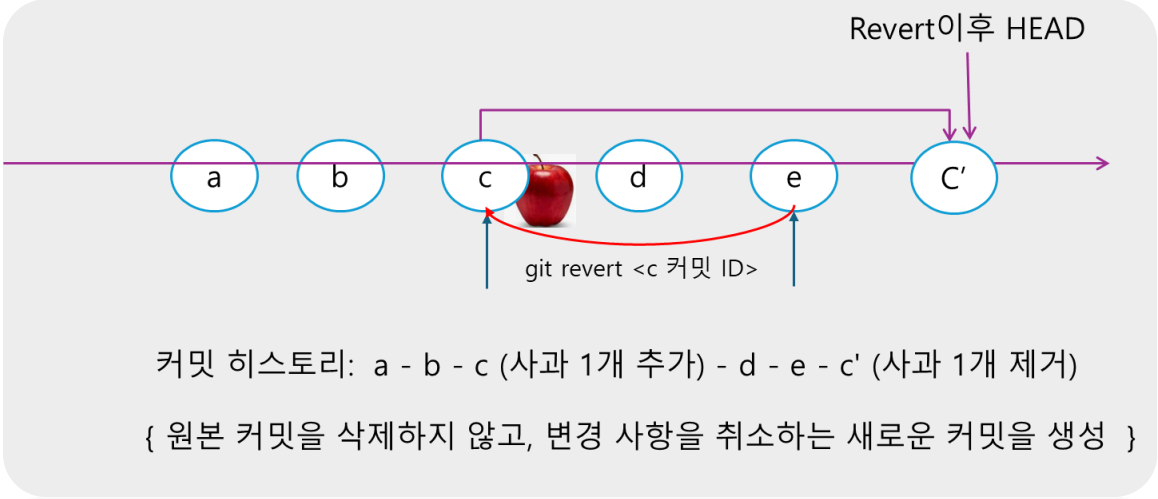


fig 1 revert

fig 1 revert 는 a - b - c (사과 1 개 추가) - d - e - c' (사과 1 개 제거)라는 커밋 히스토리를 가지고 있으며 이때 c 커밋은 "사과 1 개 추가"라는 변경 사항을 가지고 있습니다. c' 커밋은 c 커밋의 변경 사항을 "취소"하는 변경 사항을 가지고 있습니다. c' 커밋은 c 커밋을 삭제하는 것이 아니라, c 커밋의 변경 사항을 "반대로" 적용하는 새로운 커밋입니다.

git _revert 실습

Q1. 위 fig1과 같은 히스토리를 구현해보자

```
git init
echo "File A" > a.txt && git add a.txt && git commit -m "Add a.txt"
echo "File B" > b.txt && git add b.txt && git commit -m "Add b.txt"
echo "사과 하나 추가" > c.txt && git add c.txt && git commit -m "Add c.txt"
echo "File D" > d.txt && git add d.txt && git commit -m "Add d.txt"
echo "File E" > e.txt && git add e.txt && git commit -m "Add e.txt"
git log --oneline
git status
```

MINGW64:/d/myTest/myrevert_test

```
Dominica@Dominica MINGW64 /d/myTest/myrevert_test
$ git init
Initialized empty Git repository in D:/myTest/myrevert_test/.git/

Dominica@Dominica MINGW64 /d/myTest/myrevert_test (master)
$ echo "File A" > a.txt && git add a.txt && git commit -m "Add a.txt"
echo "File B" > b.txt && git add b.txt && git commit -m "Add b.txt"
echo "사 과  하 나  추 가 " > c.txt && git add c.txt && git commit -m "Add c.txt"
echo "File D" > d.txt && git add d.txt && git commit -m "Add d.txt"
echo "File E" > e.txt && git add e.txt && git commit -m "Add e.txt"
[master (root-commit) 95a6ad7] Add a.txt
 1 file changed, 1 insertion(+)
 create mode 100644 a.txt
[master 954bf33] Add b.txt
 1 file changed, 1 insertion(+)
 create mode 100644 b.txt
[master ecfd269] Add c.txt
 1 file changed, 1 insertion(+)
 create mode 100644 c.txt
[master 563a8cf] Add d.txt
 1 file changed, 1 insertion(+)
 create mode 100644 d.txt
[master c71703b] Add e.txt
 1 file changed, 1 insertion(+)
 create mode 100644 e.txt

Dominica@Dominica MINGW64 /d/myTest/myrevert_test (master)
$ git log --oneline
c71703b (HEAD -> master) Add e.txt
563a8cf Add d.txt
ecfd269 Add c.txt
954bf33 Add b.txt
95a6ad7 Add a.txt
```

Q2. git revert HEAD~2 명령어를 실행후 "Add c.txt" 커밋이 revert 되어 새로운 커밋이 생성된 것을 확인해보자.

```
Dominica@Dominica MINGW64 /d/myTest/myrevert_test (master)
$ git revert HEAD~2 # "Add c.txt" 커밋을 리버트 (사과 삭제)
[master 209814a] Revert "Add c.txt"
 1 file changed, 1 deletion(-)
 delete mode 100644 c.txt

Dominica@Dominica MINGW64 /d/myTest/myrevert_test (master)
$ git log --oneline
209814a (HEAD -> master) Revert "Add c.txt"
c71703b Add e.txt
563a8cf Add d.txt
ecfd269 Add c.txt
954bf33 Add b.txt
95a6ad7 Add a.txt

Dominica@Dominica MINGW64 /d/myTest/myrevert_test (master)
$
```

git log --oneline 명령어를 통해 커밋 히스토리를 확인해보면, 최신 커밋 메시지가 "Revert "Add c.txt"" 로 시작하는 것을 볼 수 있습니다. 이는 "Add c.txt" 커밋을 되돌리는 새로운 커밋이 생성되었음을 의미합니다.

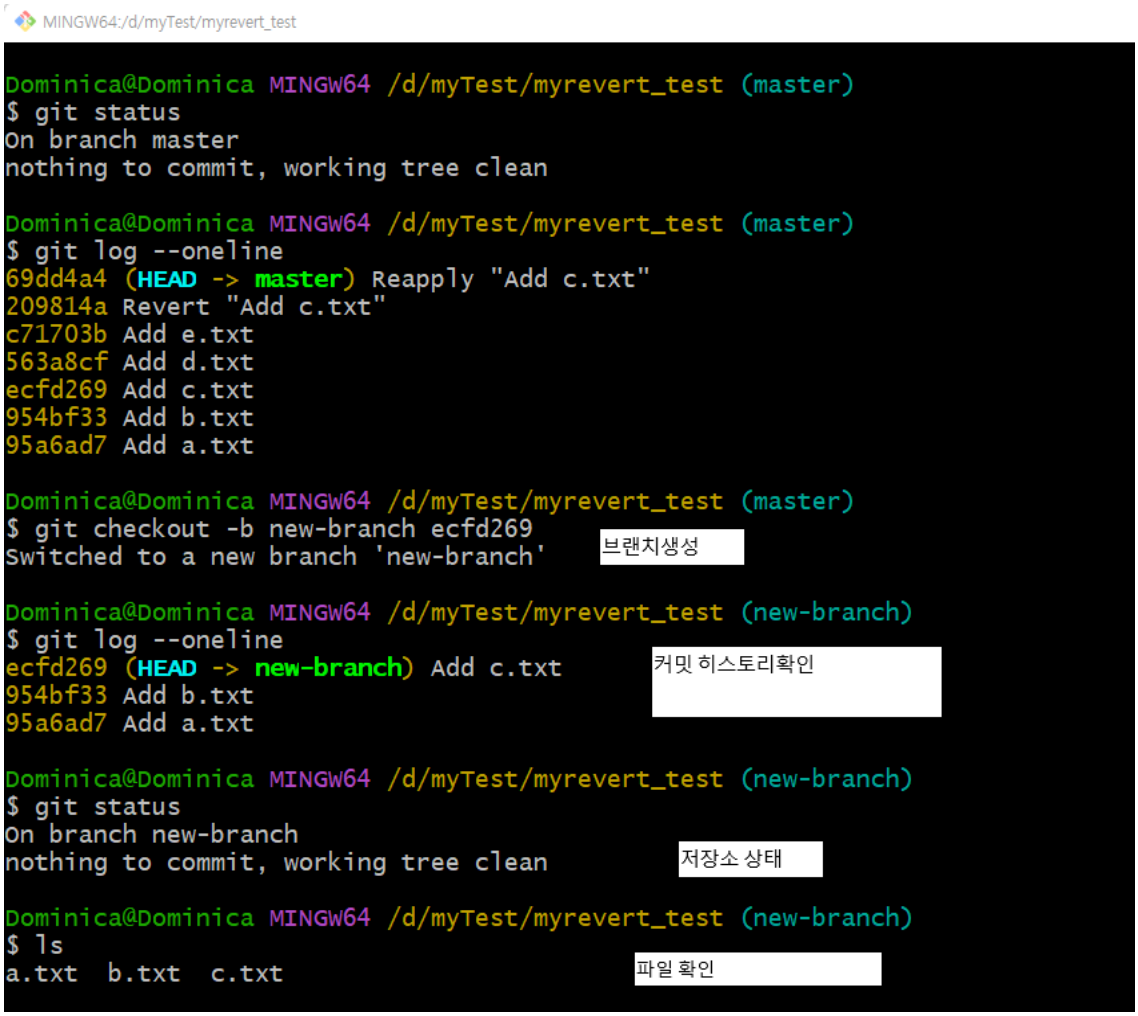
또한, git revert 명령어 실행 시 c.txt 파일이 삭제되었다는 메시지(1 file changed, 1 deletion(-))가 출력된 것을 보아 "Add c.txt" 커밋에서 추가했던 c.txt 파일이 revert 되면서 워킹디렉토리에서 삭제된 것을 알 수 있습니다.

Q3. git revert를 사용하여 생성된 리버트 커밋을 취소해보자.

git revert HEAD # 또는 git revert 209814a

Q4. "Add c.txt" 커밋(ecfd269)으로 이동하고 새로운 브랜치를 생성해보자

- git checkout -b new-branch ecfd269 명령어를 사용하여 "Add c.txt" 커밋으로 이동하고 새로운 브랜치를 생성했습니다.
- 새로운 브랜치에서 "Add c.txt" 커밋 시점의 코드를 기반으로 작업을 진행할 수 있습니다.
- 기존 master 브랜치는 영향을 받지 않습니다.



Q5. `git checkout` **명령어를 사용하여 파일을 복원하면 워킹 디렉토리의 파일이 변경되는 것을 확인하자.**

```
git checkout 563a8cf -- d.txt
git checkout c71703b -- e.txt
git add d.txt e.txt
git commit -m "Restore d.txt and e.txt".
```

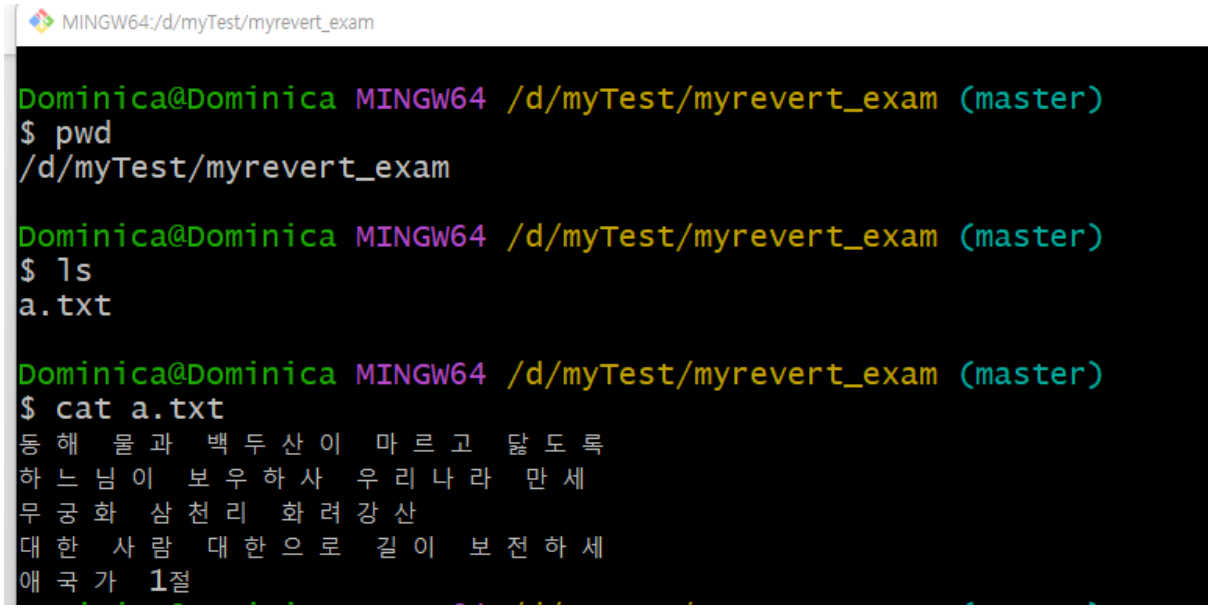
```
Dominica@Dominica MINGW64 /d/myTest/myrevert_test (new-branch)
$ git checkout 563a8cf -- d.txt
git checkout c71703b -- e.txt
git add d.txt e.txt
git commit -m "Restore d.txt and e.txt"
[new-branch 7e0dc46] Restore d.txt and e.txt
 2 files changed, 2 insertions(+)
 create mode 100644 d.txt
 create mode 100644 e.txt

Dominica@Dominica MINGW64 /d/myTest/myrevert_test (new-branch)
$ git log --oneline
7e0dc46 (HEAD -> new-branch) Restore d.txt and e.txt
ecfd269 Add c.txt
954bf33 Add b.txt
95a6ad7 Add a.txt

Dominica@Dominica MINGW64 /d/myTest/myrevert_test (new-branch)
$ ls
a.txt  b.txt  c.txt  d.txt  e.txt
```

Q6. **모든 내용을 new-branch 브랜치의 변경 사항을 master 브랜치에 반영해보자**

```
git rebase master
git checkout master
ls
```

실습

Q1. 다음과 같은 환경으로 시작한다.



Q2. 현재 상태를 저장한다.

현재 워킹 디렉토리에는 a.txt 파일이 있으며, 파일 내용은 애국가 1절입니다. 이 상태를 Git 저장소에 커밋하세요. 커밋 메시지는 "[애국가] add a.txt"로 지정합니다.

Q3. b.txt 파일 추가 및 커밋하자

다음과 같은 내용을 가진 b.txt 파일을 생성하고, Git 저장소에 커밋하세요. 커밋 메시지는 "[애국가] add b.txt"로 지정합니다.

```
무궁화 삼천리 화려강산
```

Q4. c.txt 파일 추가 및 커밋해보자

다음과 같은 내용을 가진 c.txt 파일을 생성하고, Git 저장소에 커밋하세요. 커밋 메시지는 "[애국가] add c.txt"로 지정합니다.

대한 사람 대한으로 길이 보전하세

Q5. c.txt 파일을 추가한 커밋을 리버트하자

Q6. b.txt 파일 내용 수정 및 커밋해보자.

b.txt 파일에 다음 내용을 추가하고, Git 저장소에 커밋하세요. 커밋 메시지는 "[애국가] modify b.txt"로 지정합니다.

동해 물과 백두산이 마르고 닳도록

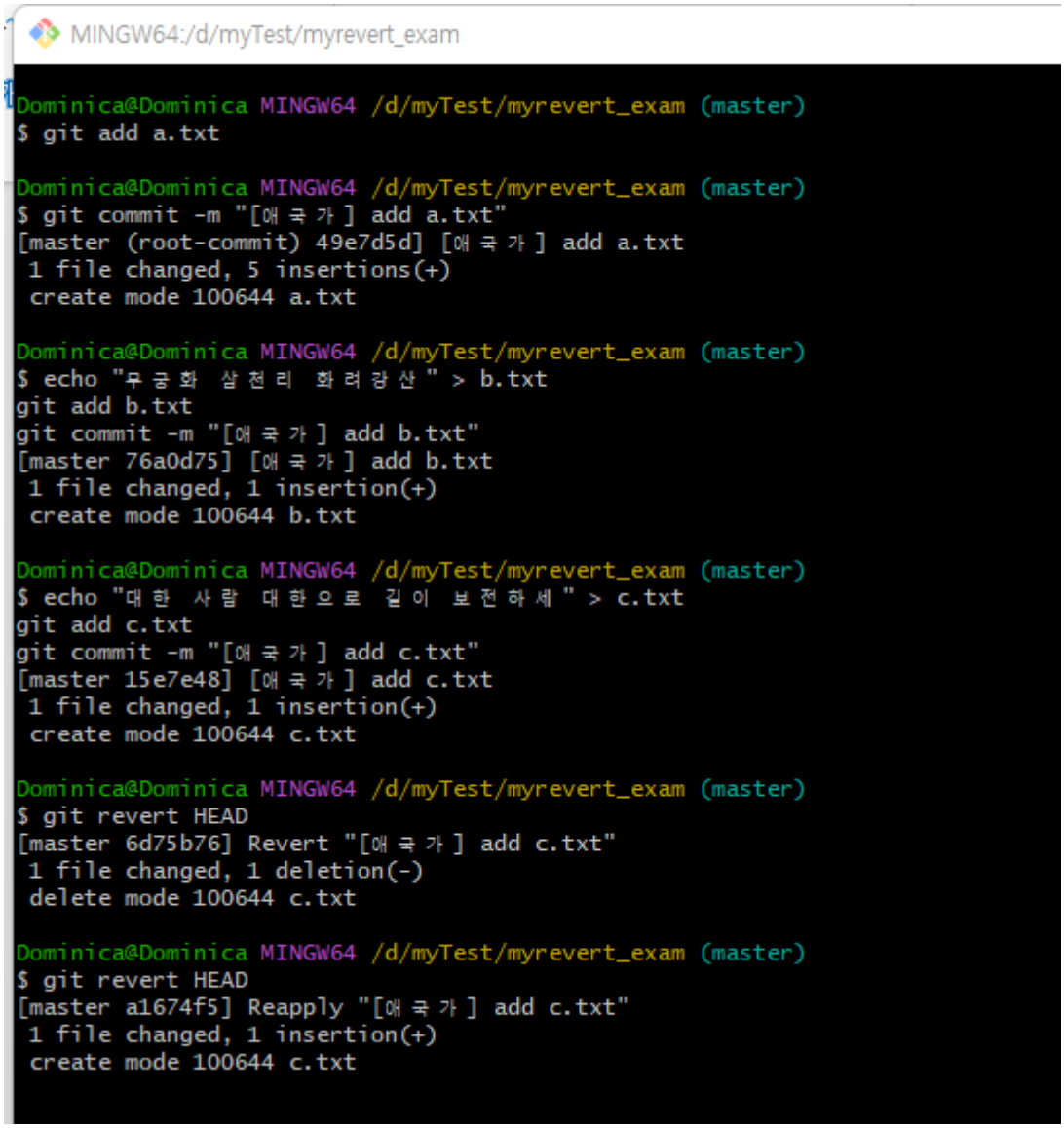
Q7. a.txt 파일 삭제 및 커밋해보자.

txt 파일을 삭제하고, Git 저장소에 커밋하세요. 커밋 메시지는 "[애국가] remove a.txt"로 지정합니다.

Q8. 현재까지의 커밋 히스토리를 확인 해보자 .

Q9. 현재 Git 저장소의 상태를 확인해보자.

실행단계확인


MINGW64:/d/myTest/myrevert_exam
Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ git add a.txt
Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ git commit -m "[애 국 가] add a.txt"
[master (root-commit) 49e7d5d] [애 국 가] add a.txt
1 file changed, 5 insertions(+)
create mode 100644 a.txt
Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ echo "무 궁 화 삼 천 리 화 려 강 산 " > b.txt
git add b.txt
git commit -m "[애 국 가] add b.txt"
[master 76a0d75] [애 국 가] add b.txt
1 file changed, 1 insertion(+)
create mode 100644 b.txt
Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ echo "대 한 사 람 대 한 으 로 길 이 보 전 하 세 " > c.txt
git add c.txt
git commit -m "[애 국 가] add c.txt"
[master 15e7e48] [애 국 가] add c.txt
1 file changed, 1 insertion(+)
create mode 100644 c.txt
Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ git revert HEAD
[master 6d75b76] Revert "[애 국 가] add c.txt"
1 file changed, 1 deletion(-)
delete mode 100644 c.txt
Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ git revert HEAD
[master a1674f5] Reapply "[애 국 가] add c.txt"
1 file changed, 1 insertion(+)
create mode 100644 c.txt

```
Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ echo "동해 물과 백두산이 마르고 닳도록" >> b.txt
git add b.txt
git commit -m "[애국가] modify b.txt"
[master d9fa155] [애국가] modify b.txt
 1 file changed, 1 insertion(+)

Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ git rm a.txt
git commit -m "[애국가] remove a.txt"
rm 'a.txt'
[master a7c19b4] [애국가] remove a.txt
 1 file changed, 5 deletions(-)
 delete mode 100644 a.txt

Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ git log --oneline
a7c19b4 (HEAD -> master) [애국가] remove a.txt
d9fa155 [애국가] modify b.txt
a1674f5 Reapply "[애국가] add c.txt"
6d75b76 Revert "[애국가] add c.txt"
15e7e48 [애국가] add c.txt
76a0d75 [애국가] add b.txt
49e7d5d [애국가] add a.txt

Dominica@Dominica MINGW64 /d/myTest/myrevert_exam (master)
$ git status
On branch master
nothing to commit, working tree clean
```