♨[Group Study] – Git기본2 by jinuman

* **변경사항 확인하기**

Git을 쓰는 가장 중요한 효용중 하나! : commit을 통한 변경사항 확인!

1. 버전간 차이점을 알 수 있고 과거의 어느 시점의 내용을 알 수 있다.
2. 과거로 돌아갈 수 있다.

제일 중요한 효용은 협업!

$git log -p : 로그에서 출력되는 버전간 차이점을 출력하고 싶을 때

$git log "commit\_ID" : commit ID를 copy paste 하면 해당 시점 버전 이전의 버전들의 log를 볼 수 있다.

$git diff "commit\_ID".."commit\_ID" : 특정 버전 간의 차이점을 비교할 수 있다

git diff : add & commit 하기 전에 실수 한 것이 없는지 마지막으로 확인하는 확인작업 -> commit하기 전 확인하는 습관을 들이자

* **과거의 버전으로 돌아가기**
  1. reset (hard, soft, mixed)

$git reset "commit\_ID" --hard : 해당 commit\_ID 의 버전을 최신버전으로 함

$git reset HEAD~1 : Head부터 한칸 뒤로 최신 commit을 이동!

주의할 점 : 디렉토리의 버전들을 인터넷에 공유한 후에는 절대로 reset을 하지 말자!

1. revert

$git revert "commit\_ID" : 특정 commit만을 되돌린다.

$git revert "commit\_ID".."commit\_ID" : 범위를 주어서 여러 개의 commit을 되돌린다.

* reset vs revert

reset이 이력을 단순하게 할 수 있어서 좋아 보인다.

그러나 굉장히 위험한 방법이라고 한다.

중간의 어떤 특정 commit만을 취소하려고 한다면 revert를 써야 한다.

reset을 쓰면 그 이후 commit들이 다 날아가 버리기 때문에

그리고 원격에 push를 한 상태에서는 무조건 revert를 해야 한다.

reset을 하게 되면 reset하기 전의 상태로 되돌리기 전까지 push를 할 수 없다.

!!결론 : Github(원격)으로 협업 중이라면 reset은 과감히 기억에서 지워버리자..

* **Wanna change commit message?? 커밋 메세지 수정하고 싶니??**

1. HEAD 커밋 메시지만 수정할 때

$git commit --amend

2. HEAD 이전 커밋 메시지 수정할 때(예: 3번째 전까지)

$git rebase -i HEAD~3

변경을 원하는 커밋을 pick에서 edit로 수정.

$git commit --amend와 $git rebase --continue를 edit로 수정한 커밋 개수만큼 실행. 해보면 알 것.

(도중에 취소하고 싶을 때) $git rebase --abort

* **브랜치를 쓰는 이유?** 
  + 학부 단계에선 주로 test해보고 싶을 때 브랜치를 따서 작업 후 괜찮으면 merge / 그냥 해본거면 브랜치를 냅두거나 삭제 (master와는 독립적)
  + 현업에서는 고객에 요구에 따른 다양한 버전을 release해야 할 경우 branch를 써서 maintain가능