Git 명령어

Git Study

local repository(Commend Line)

Working tree -> Staging Area -> Repository

파일작업하는 곳 버전을 만들려고 하는 파일들 만들어진 버전

1. 먼저 경로에 가서 버전생성

명령어 설명

git init Intialize repository(.git파일)

2. 작업한 Working tree에서 Staging Area로 올리는 작업

명령어 설명

git add . 전체 파일 올라감

git add 파일명 원하는 파일만 올라감

3. git에게 버전을 만들게 하는 단계

명령어 설명

git commit 기본 editor가 나와 여러가지 문장으로 commit massage 작성가능

git commit -m "버전설명" Staging Area에 있는 파일들 버전으로 만듦(-m은 메세지 작성)

git commit -am "버전설명" add와 commit 한번에 가능, but untracted 상태에서는

안됌, 최초 한번은 add해야함

기타

명령어 설명

ls -al 현재 경로에 있는 파일들을 보여줌

pwd 현재경로

cat 파일명 파일안에 있는 데이터 보여줌

git status 현재 상태(add 되었는지 안되었는지 알려줌)

clear clear

git log commit 된것 다 보여줌

git log --stat commit 안에 여러개의 파일 변경사항 보기

git log -p 어떠한 점들이 변경되어는지 보여줌

git log --all 모든 브랜치 보여줌

git log --all --graph 시각적표현

git log --all –graph --oneline 한줄

git diff 변경사항들 보여줌(버전을 만들기전에 최종적 검토)

git reset --hard **HEAD가 브랜치를 제어해서 버전을 바꾸는 것**

**commit한 버전을 가르킨다. 삭제의 느낌**

git checkout 버전id 버전을 만든 시점으로 돌아감(버전id는 git log통해서)

지워진것은아님!!, **HEAD를 바꾸는 것! 브랜치를 가르킨다.**

**변경의 느낌**

git checkout master 다시 최신으로 돌아옴

git commit –amend commit 메시지 수정

### git l 설정방법 ###  
1) nano ~/.gitconfig 를 열어주세요.  
2) 아래 내용을 입력하면, git l 단축명령어를 사용할 수 있습니다.  
[alias]  
l = log --all --graph --oneline

버전삭제!!!

명령어 설명

git reset --hard 버전id 저 버전id으로 reset하겠다.(되겠다.)

--hard는 버전 뿐만아니라 수정하고 있는것도 지운다.

--soft는 수정하는 것은 유지

버전되돌리기!!

명령어 설명

git revert 버전id 맨위에 있는 버전 전 버전으로 되돌린다.

기존의 커밋은 내버려두고 commit의 변화만 취소되므로 전버전으로 간다.

주의할점! 하나씩 차례대로(역순으로) 단계적으로 revert를 해야함.

원격저장소(github)와 연결(remote& push)

명령어 설명

git remote add origin 깃페이지복사 githup과 연결하기

git remote githup과 연결된 레파지토리 명나옴

git remote -v githup과 연결된 주소명 나옴

git push githup에 올리기(처음에 git push --set-upstream origin master해야함)

원격저장소 복제(clone)

명령어 설명

git clone 깃페이지복사 (mkdir) 현재 경로에 저장소 복제(mkdir을 사용시 레퍼지토리명 변경가능)

원격저장소 pull

협업하고 있는 컴퓨터에서 수정한 내용을 pull을 먼저해서 수정된 내용을 자신의 컴퓨터에 pull하고 작업을 시작한다.

pull -> 작업 -> commit -> push

명령어 설명

git pull

깃 branch & conflict

명령어 설명

git log --all --graph --oneline 한줄로 브랜치의 구성(그래프)를 그림으로 쉽게 보여줌

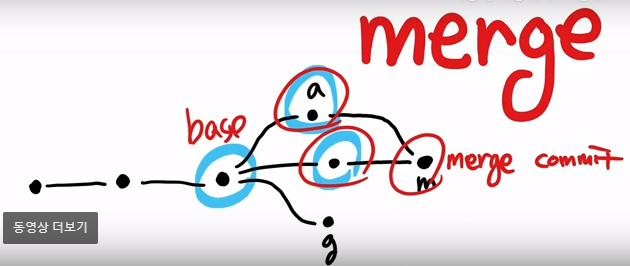
git branch 현재 브랜치를 모두 보여줌

git branch 브랜치이름 현재 있는 브랜치의 버전에서부터 브랜치이름이라는 브랜치 생성

git checkout 브랜치이름 브랜치이름이라는 버전으로 이동, 텍스트로 바로바뀜

깃 merge

용어 1. base: 브랜치가 쪼개졌을 때의 공통의 조상을 말한다.



위에 그림처럼 Master에 버전에 a를 병합하고 싶다.

merge에는 크게 **두 가지 상황**이 존재한다.

**1**. 서로 다른 파일을 수정했을 때 문제 x

**2**. 서로 같은 파일을 수정했을 때

-서로 다른 부분 수정 했을 때 문제 x

-서로 같은 부분 수정 했을 때 문제 o -> conflict 발생 사용자가 직접 수정해야 함

os에서는 같은 파일 다른 부분을 수정하면 파일 하나를 더 생성하게 된다. 그러면 이름이 달라진다. 그러나, 깃은 os와 달리 같은 파일이라도 다른 부분을 수정하면 자동적으로 merge해주는 기능이 있다. 깃은 같은 파일 같은 부분을 수정할 때에는 conflict가 발생한다.

1. 먼저 master 브랜치로 이동한다.
2. git merge 명령어를 통해서 병합한 브랜치를 생성한다.

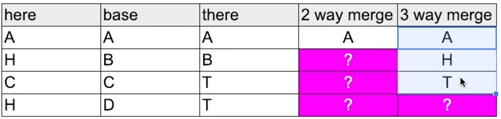
명령어 설명

git merge 병합하고자 하는 브랜치명 여기서 중요한것은 병합하기전에 브랜치의 위치 중요,

현재 브랜치로 merge됨!

3 way merge

이 방법은 더 적은 충돌을 가능하게 한다.



**Collaboration**

pull -> 작업 -> commit -> push

git fetch -> git merge FETCH\_HEAD(최신 버전 자동으로 가져옴) -> commit -> push

remote 브랜치만 가져온다. 즉, 현재 내가 작업하고 있는 것이 상대방 최신 버전으로 업데이트는 되지 않고 버전만 가져오고 내가 작업하는 것은 그대로 유지!

**Git patch 🡪 오픈소스프로젝트!!**