<생활 코딩> - 지옥에서 온 Git 강의 요약

(멋사 강의자료와 조금 다름)

**[수업 소개] – 12분**

Git – 버전 관리 시스템 (version Control system)

Version Control System이란?

- 이미 우리는 하고 있었음. Ex) 팀플 발표 자료 (최종, 최종최종, …) 등과 같이 파일의 이름을 여러가지로 바꿔가면서 여러 버전의 파일 저장

- 목표 : 파일의 이름을 바꾸지 않고 버전 관리를 하자.

-> 백업, 이전 상태로 돌아가기, 협업

Git은 매우매우매우 복잡하고 어려움.

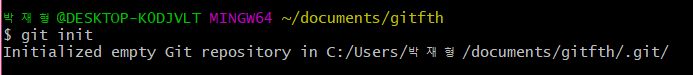
Codeonweb.com 에서 git 실습 가능

**[git init] – 4분**

Pwd 명령어(리눅스) : 현재 위치 알려줌

mkdir 명령어(리눅스) : 폴더 생성

init -> 현재 디렉토리에 작업을 하겠다는 것을 깃에게 알려주는 기능

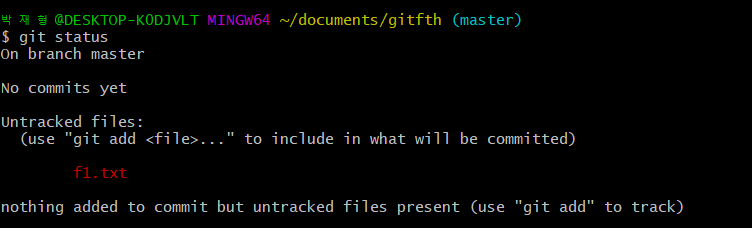


여러가지 버전들에 대한 정보는 모두 git 폴더에 저장됨

**[git add] – 6분**

우선 텍스트 파일 생성 (예시) – vim, cat 등 이용

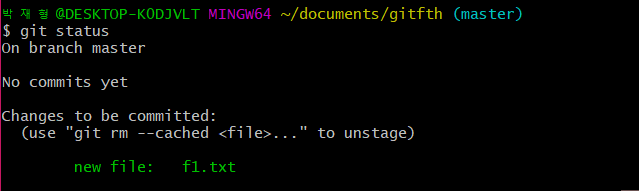
Git status



Untracked files : 지금 생성한 f1.txt 파일은 버전 관리가 되고 있는 디렉토리인 gitfth안에 존재하지만, 이 파일을 깃한테 버전 관리를 시작하라고 얘기하기 전까지는 이를 무시함.

-> 관리해라 : git add f1.txt

(후 결과)



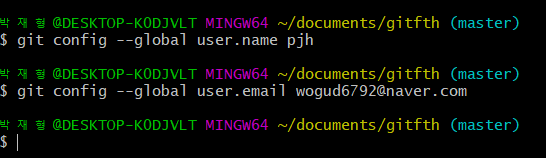
Git add 가 도입된 이유 : 임시로 필요한 파일들은 포함하면 안되므로

**[버전 만들기 (commit)] – 7분**

저장소에 상태가 어떤지를 살펴볼 때는 git status 명령어 사용

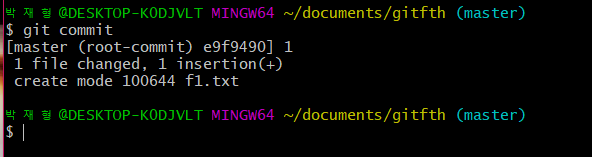
Git config --global user.name pjh

Git config –global user.email [wogud6792@naver.com](mailto:wogud6792@naver.com)

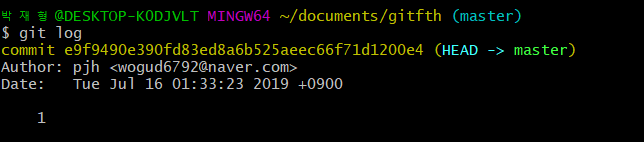


위 과정은 한번만 하면 됨. 누가 작업했는지를 알게 하기 위함.

Git commit 명령어 누르고 안에 파일 수정



Git log 명령어로 현재 버전이 잘 만들어졌는지 확인



깃에서 어떤 새로운 파일이 생겼을 때, 그 파일에 대한 버전 관리를 깃에게 명령할 때에도

Git add를 사용함.

어떤 파일에 이미 버전 관리가 되고 있는 파일 ex) f1.txt 의 수정이 끝나고, 버전을 생성할 때에도 또 git add를 해주어야 함.

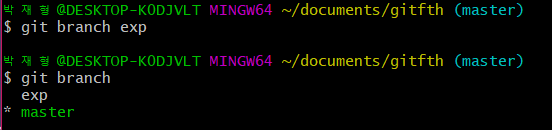
즉, 최초로 추적할 때에도 add, 파일이 수정되어 버전을 만들기 전에도 add

**[branch 소개] – 7분**

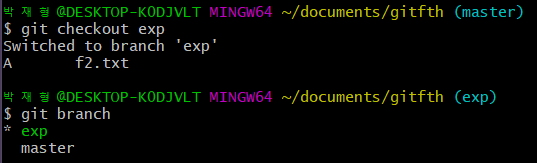
Git branch 명령어 입력하면 현재 branch들 보여줌



Git branch 이름 명렁어 입력하면 branch 생성



Git checkout [이름] 하게 되면 해당 이름의 branch로 이동



(exp branch로 이동한 것을 볼 수 있음)

이 때, exp branch의 내부를 보면 master와 동일함

즉, 마스터 branch와 exp branch는 같은 상태. => branch를 생성하면 생성한 branch는 현재 속해있던 branch의 상태를 그대로 복사함 **<중요>**

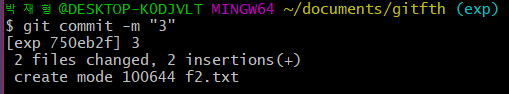
기존에 master에는 1이라는 이름의 버전 하나만 있는 상황



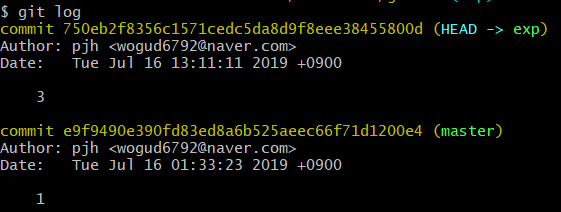
f1.txt 들어가서 내용 추가



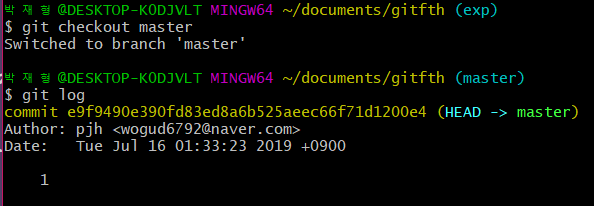
Git add 해줌



Git commit 해줌



Git log 해보면 3이라는 이름의 버전이 추가된 것을 알 수 있음

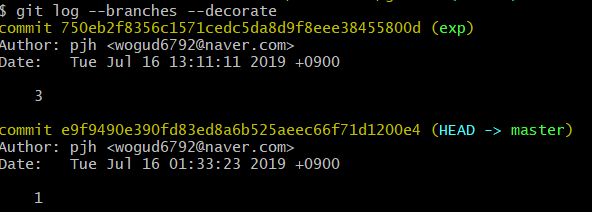


master 다시 확인해보면 1이라는 이름의 버전만 존재하는 것을 알 수 있음

=> 어느 branch에 속해있냐에 따라 파일이 완전히 달라짐

**[branch 정보확인] – 11분**

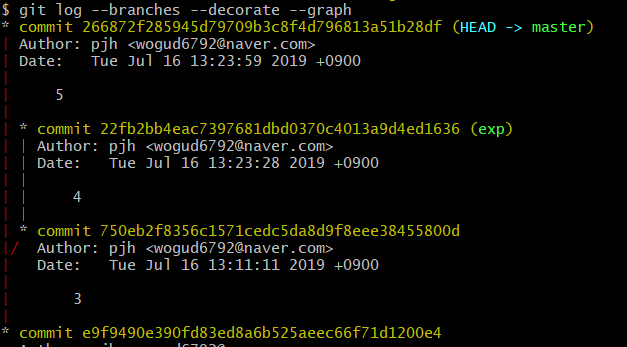
Git log --branches --decorate



Head가 가리키는 branch가 현재 branch 위치

Git log --branches --decorate --graph --oneline

Branch의 조상을 보여줌

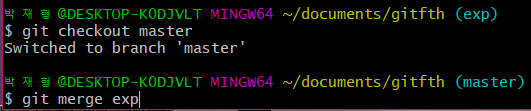


Git log master..exp => master에는 없고 exp에는 있는 버전을 보여줌

Git log exp..master => exp에는 없고 master에는 있는 버전을 보여줌

**[branch 병합] – 7분**

Exp를 master에 합쳐주기 위해서는 master로 checkout한 뒤에, master에서 merge



Git branch -d [이름] 명령어 입력하면 branch 삭제