



Git 실습 1

📅 날짜	@2024년 1월 4일
☰ 태그	Cloud 세희

Git

리눅스와 같은 환경을 구축하도록 도와주는 tool

- windows powershell: 여러가지 쉘 관련 기능 가지고 있음
 - 이에 맞는 쉘 문법이 따로 존재
- 만약 모든 것이 처음이라면 공식문서를 한번 읽어보기 추천
- git 홈페이지 가면 document랑 무료 교습 제공하고 있음
- yaml 확장자 파일: 명세하는 파일

Git - Book

This book is available in
English.

 <https://git-scm.com/book/ko/v2>

처음에 설치하고 help 쳐보기

```
PS C:\Users\2022-PC(T)-4> git -help init
```

```
unknown option: -help
```

```
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=  
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]  
        [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--n  
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]  
        [--config-env=<name>=<envvar>] <command> [<args>]
```

- 윈도우와 맥에서 엔터 방식 차이로 인한 오류 방지

```
git config --global core.autocrlf true
```

- 코드

```
git help -a    #Git의 모든 명령어들
git (명령어) -h    #해당 명령어의 설명과 옵션 보기
git help (명령어)    #해당 명령어의 설명을 볼 수 있는 사이트 제시
git config --global init.defaultBranch main #branch 명 변경하기
git status    #현재 상태 보여주기
```

- git은 기업에게 아주 중요한 자산이기 때문에 유출되면 안돼!
 - 보안적인 issue

▼ Git 기초

- vs code 진행
- 들어가자마자 좌측에 내가 작업하고자 하는 폴더 선택
- 터미널 열고 git bash
- 내가 작업한 폴더로 들어가서 보기 > 표시 > 숨긴 항목 들어가면 작업 내용이 저장되어 있음(로컬 저장소)
 - 이 폴더 지우면 다 날아감 ㅋㅋㅋㅋㅋㅋ
 - git commit을 하면 여기에 저장
- git init을 git bash 터미널에 수행
- 여기서 작업을 수행하고 옆에 source control을 보면 우리의 변경 사항이 존재함 → commit을 누르기 전까지 폴더는 비어있다.
- 파일 옆에 U가 떠있다는 것은 git이 아직 손대지 않았다. git이 파일을 인식하지 못함

```
git status
git add lion.yaml
```

```

$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        lions.yaml
        tigers.yaml

nothing added to commit but untracked files present (use "git add" to track)

2022-PC(T)-4@DESKTOP-OTU9SAO MINGW64 /c/src/hello (master)
$ git add lions.yaml

2022-PC(T)-4@DESKTOP-OTU9SAO MINGW64 /c/src/hello (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   lions.yaml

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        tigers.yaml

```

파일이 등록되고 source control을 봤을 때 상태가 변경된 것을 볼 수 있음

add를 수행한다고 해서 파일 저장이 된 것은 아님. 저장은 commit으로 수행한다. add를 함으로써 추적이 가능하게 됨(누가 어떻게 어떤 작업을 했다.)

```

# 모든 파일 add하기
git add .

```

※ 반드시 **.git** 폴더가 작업하는 폴더 내부에 있어야 한다.

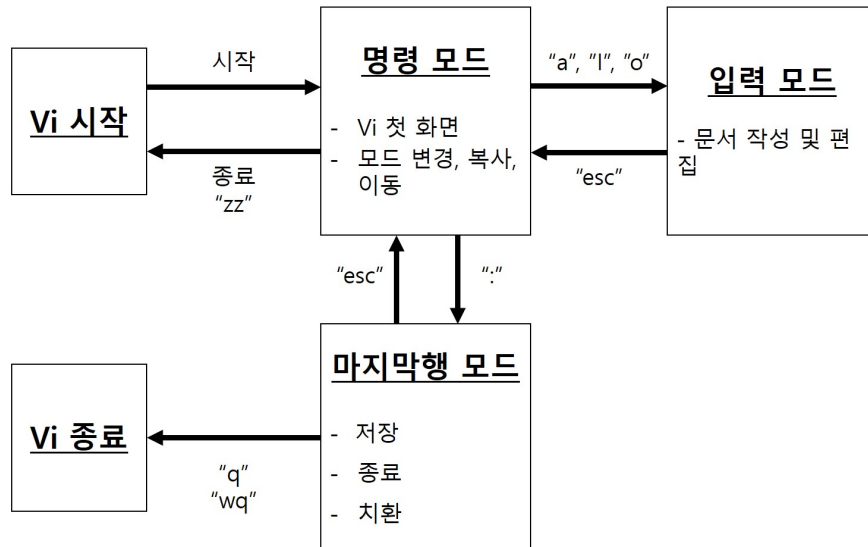
없다면 파일을 추적하지 못한다.

작업을 하다가 필요없는 파일 원격 저장소에 upload할 필요는 없다.

이런 파일의 이름을 **.gitignore**로 명명하면 git이 추적을 하지 못한다.

.gitignore에 올릴 필요가 없는 파일을 넣어 놓는다.

```
# VI editor
$ vi
>> vi 모드로 진입
i 입력 후 원하는 text 작성, esc 누르고 모드 나오고
:wq 입력 -> 저장 후 종료
```



- **\$git log**

```
2022-PC(T)-4@DESKTOP-OTU9SAO MINGW64 /c/src/hello (master)
$ git log
commit afe23471118f733ab96b59526f529aca1f5c71aa (HEAD -> master)
Author: sehee <sally001234@gmail.com>
Date: Thu Jan 4 13:49:42 2024 +0900

i FIRST COMMIT
```

commit 한 log를 볼 수 있다.

commit 뒤에 있는 고유한 해시 값을 통해 복구, 복원이 가능함

- 문서 여러 개 수정하고 status 확인해보기

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    lions.yaml
        modified:   tigers.yaml

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        leopards.yaml

no changes added to commit (use "git add" and/or "git commit -a")
```

무엇이 변경되었는지 볼 수 있음

```
2022-PC(T)-4@DESKTOP-OTU9SAO MINGW64 /c/src/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   leopards.yaml
        deleted:    lions.yaml
        modified:   tigers.yaml
```

git add를 수행함으로써 추적 가능해짐

- Vi editor없이 바로 commit이 가능하게
 - “안에 있는게 commit 문구
 - ***\$git commit -m “Replace Lions with Leopards”***
 - ***\$git commit -am “Replace Lions with Leopards”(add와 commit 한꺼번에)***
 - 새로 추가한 파일은 -am해도 한 번에 commit 되지 않음
 - 기존에 관리하던 파일은 수행

```

2022-PC(T)-4@DESKTOP-OTU9SAO MINGW64 /c/src/hello (master)
$ git log
commit 40a59cb45dca4cfb2cf195385670603421713498 (HEAD -> master)
Author: sehee <sally001234@gmail.com>
Date: Thu Jan 4 14:07:18 2024 +0900

    Replace Lions with Leopards

commit afe23471118f733ab96b59526f529aca1f5c71aa
Author: sehee <sally001234@gmail.com>
Date: Thu Jan 4 13:49:42 2024 +0900

    i FIRST COMMIT

```

commit log

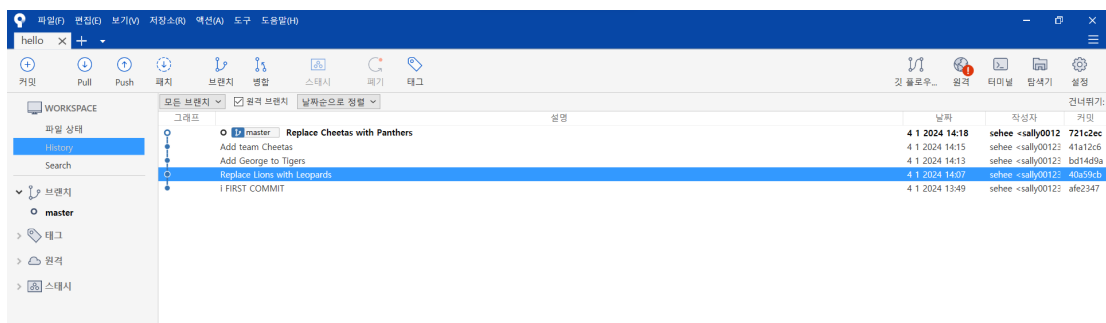
- 실습!

- 기존 파일 지우고, 파일 하나 추가하고, 기존 파일 수정하고!
- git add . → git commit -m "commit change 수정 내역 text"

```

2022-PC(T)-4@DESKTOP-OTU9SAO MINGW64 /c/src/hello (master)
$ git commit -m "Replace Cheetas with Panthers"
[master 721c2ec] Replace Cheetas with Panthers
3 files changed, 9 insertions(+), 9 deletions(-)
delete mode 100644 cheetas.yaml
create mode 100644 panthers.yaml

```



source tree 화면. 5번 commit 내역 확인 완료

- **reset**

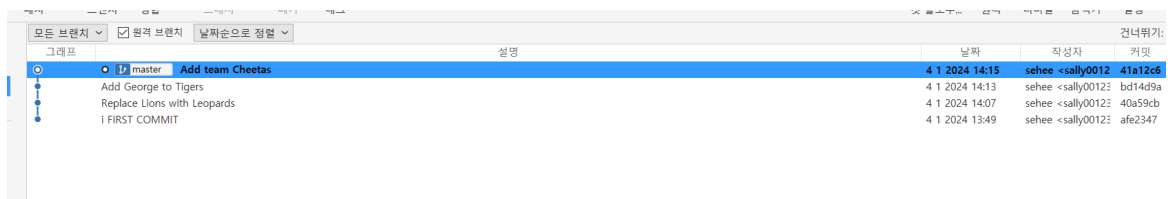
이전으로 돌아갈 때 아예 commit 내용을 지워버린다.

문제는 지워버리기 때문에 그 동안의 작업 내역 및 누가 reset을 했는지 모름

따라서 로컬에서 reset을 진행할 때는 .git 파일을 복사해서 백업 해놓는게 좋음

```
$ git reset --hard (commit의 고유 해시값)
```

해시값은 log를 통해 얻거나 sourcecetree에서 우클릭하고 SHA값 얻기 클릭

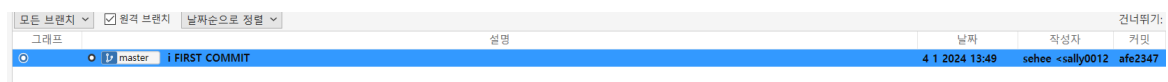


그래프	설명	날짜	작성자	커밋
4 1	Add team Cheetas	2024 14:15	sehee <sally0012>	41a12c6
4 1	Add George to Tigers	2024 14:13	sehee <sally0012>	bd14d9a
4 1	Replace Lions with Leopards	2024 14:07	sehee <sally0012>	40a59cb
4 1	FIRST COMMIT	2024 13:49	sehee <sally0012>	afe2347

사진을 보니 이전 commit으로 리셋하면 그 중간에 있는 내용이 삭제된 것을 볼 수 있다.

- 첫번째 commit으로 리셋해보자~

- SourceTree에서는 원하는 부분에서 우클릭하면 “**이 커밋까지 모든 브랜치를 초기화**”를 클릭.



그래프	설명	날짜	작성자	커밋
4 1	FIRST COMMIT	2024 13:49	sehee <sally0012>	afe2347

내 금쪽 같은 commit 다 사라짐

- 백업한 걸 다시 보니 git이 이 파일을 제대로 인지하고 있지 못한 상태

```

2022-PC(T)-4@DESKTOP-OTU9SAO MINGW64 /c/src/hello (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    leopards.yaml
        deleted:    panthers.yaml
        modified:   tigers.yaml

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        lions.yaml

no changes added to commit (use "git add" and/or "git commit -a")

```

```

# 해당 폴더를 다시 인식하게 만든다.
$git reset --hard

```

- **revert(되돌리기)**

파일의 상태를 **해시값 시점의 상태를 되돌리나** 그 중간에 있는 내역들을 삭제하지는 않는다.

```

# commit text 다시 입력할 수 있음, commit 내역에 +1
$git revert (해시값)

```

실습 - SourceTree로 진행

1. 변경사항 만들고 커밋하기

GIT 실습 3 (Reset, Revert)
 leopards.yaml 삭제
 .gitignore 에 .config 추가
 hello.txt 추가 (내용 자유)

커밋 메시지(커밋 버튼): Commit with SourceTree
 파일 옆 + 버튼 또는

모두 스테이지에 올리기

2. revert

Add George to Tigers 의 수정사항 되돌려보기

해당 커밋에 마우스 우클릭 - 커밋 되돌리기

커밋 추가 확인

tigers.yaml에서 Geoge가 추가된 내용이 삭제됨

3. reset

Replace Cheetas with Panthers 시점으로 되돌려보기

해당 커밋에 마우스 우클릭 - ... 이 커밋으로 초기화

선택지에서 Hard 선택

이후 시점의 commit 사항 모두 삭제

▼ Git - Branch

- 프로젝트 하나 이상의 모습으로 관리해야 할 때!
- 여러 작업들이 각각 독립되어 진행될 때
- 나중에 한꺼번에 합치는 것은 merge!
 - 각각 작업 후 merge를 통해 합친다.

- **Code**

```
# 브랜치 만들기
$ git branch add-coach

# 브랜치 switch하기
$ git switch add-coach

# 동시에 만들면서 switch하기
$ git switch -c new-teams(new name)

# 브랜치 이름 바꾸기
$ git branch -m to-delete(기존 이름) to-ease(새 이름)
```

```
# 브랜치 삭제하기
$ git branch -d to-ease(브랜치 이름)

# 여러 브랜치의 내역 편리하게 보기
$ git log --all --decorate --oneline --graph
```

- add_coach라는 브랜치에 적용된 사항은 main에 적용되지 않음

```
panthers.yaml
1  team: Panthers
2
3  manager: Sebastian
4
5  members:
6    - Violet
7    - Stella
8    - Anthony
9    - Freddie
```

main

```
panthers.yaml
1  team: Panthers
2
3  manager: Sebastian
4
5  coach: Teddy
6
7  members:
8    - Violet
9    - Stella
10   - Anthony
```

add_coach branch

- Rebase(비추천)
대상 브랜치로 가지를 전부 옮겨 붙기
- **Merge**

```
# 내가 존재하는 branch와 다른 브랜치 합치기
$ git merge new-teams(합치고 싶은 브랜치)
```

만약 merge를 취소하고 싶다면 reset —hard를 줘서 이전 commit을 삭제할 수도 있겠다!