Git 명령어 정리

1. Git 유저 및 기본설정

 # git init

> 현재 위치에서 저장소 생성

 # git config --global user.name "[사용자명]"

> Git 환경에서 사용자 이름 지정

 # git config --global user.email "[사용자이메일]"

> Git 환경에서 사용자 이메일 지정

 # git status

> Git의 상태를 확인

2. Commit 명령어

- Github 업로드 순서 : 스테이징 -> 커밋 -> 푸쉬

 # git add [파일명.확장자명]

> 변경 내용을 스테이지에 올림 (스테이징)

 # git commit -m [메시지명]

> 스테이징된 변경내용을 커밋

 # git commit -am [메시지명]

> 스테이징과 커밋 한번에 진행 (한번이라도 커밋한 적이 있는 파일만 가능)

 # git log

> 커밋 리스트 조회
>
> - --oneline , --pretty=oneline 등 옵션 추가 시 한줄로 표기

 # git show [커밋 id]

> 특정 커밋 내역 확인

 # git diff [이전커밋 id] [이후 커밋 id]

> 커밋 수정 내역 비교

 # git checkout [커밋 해시]

> 지정한 커밋 해시로 이동

 # git reset HEAD^

> 현재 HEAD의 이전 커밋으로 되돌리기

 # git reset HEAD~n

> 현재로부터 n번째 이전 커밋으로 되돌리기

 # git reset [커밋 해시]

> 지정한 커밋 해시로 이동하고 커밋을 취소

```
* reset의 3가지 옵션
-------------------------------------------------------------
# git reset --soft [커밋ID]  -> head 만 바뀜
# git reset --mixed [커밋ID] -> staging 도 그 때로 바뀜
# git reset --hard [커밋ID]  -> working 디렉토리 / staging 모두 그 때로 바뀜
-------------------------------------------------------------
```

 # git revert [커밋 해시]

> 지정한 커밋 해시의 변경 이력을 취소

★ 커밋 수정하는법

```
# git add .
# git commit --amend  : 최신 커밋 수정
```

3. Branch 명령어

 # git branch

> 브랜치 조회하기

 # git branch [브랜치명]

> 새로운 브랜치 생성

 # git checkout [브랜치명]

> 브랜치 이동
>
> - -b 옵션 붙이면 -> 브랜치 만들고 바로 이동

 # git branch -d [브랜치명]

> 브랜치 삭제

 # git log --branches --graph

> 각 브랜치의 커밋을 그래프로 표시

 # git merge [브랜치명]

> 해당 브랜치를 master 브랜치와 병합
>
> - --edit : 병합 후 바로 커밋 메시지 수정
> - --no-edit : 커밋 메시지 수정없이 바로 병합

 # git merge --abort

> merge 취소하기

4. Github 원격 저장소

 # git remote add [remote명] [저장소 주소]

> 원격 저장소에 연결

 # git remote -v

> 원격 저장소에 연결됐는지 확인 (remote명 확인)

 # git push [remote명] [branch명]

> 원격 저장소로 업로드 (push)
>
> - -u 옵션 입력시 앞으로 이 설정을 자동 기억 -> git push 만 입력시 자동 push

 # git pull

> 원격 저장소의 커밋 다운로드 (pull)
>
> - 보통 저장소와 커밋 내역이 다를때 동기화에 사용

 # git clone [원격 저장소 주소]

> 해당 주소 저장소 복제

- 원격 저장소 파일수정 순서
    1. fetch로 내 파일과 다른점(변경점) 확인
    2. checkout으로 fetch로 가져온 브랜치로 이동
    3. fetch로 가져온 브랜치 merge로 master와 병합
    4. commit push

 # git fetch [branch명]

> 원격 저장소에서 변경 내용을 확인 (merge 전 바뀐 내용이 있는지 확인)

> -> diff -> checkout -> merge -> push

 # git remote remove [remote명]

5. 파일 보관 명령어 (stash)

 # git stash

> 작업트리의 수정내용(add후 인덱스 내용)을 stash에 따로 보관하기

> = git stash save

 # git stash list

> 보관한 내용 목록을 출력

 # git stash apply

> 보관한 내용을 적용

> or git stash apply stash@{1}

 # git stash drop

> 보관한 내용 중 가장 최근 항목을 삭제

> or git stash drop stash@{1}

 # git stash pop

> stash를 apply하고 제거(drop)하기

6. 기타 명령어

 # git config --global alias.[별명] [사용할 명령어] - alias 적용

> 별명을 명령어와 연결시켜 간편하게 사용하기

```
> ex) # git config --global alias.cm commit ( commit을 cm으로 줄여서 사용 )
```

 # git tag [태그이름] [커밋 ID]

> 커밋에 태그를 달아서 편리하게 조회하기

```
> ex) # git tag Testtag 1b217  - Tag 달기
      # git tag  - Tag 목록 조회하기
      # git show Testtag  - 해당 Tag 조회하기
```