git 기본 사용

$ git init [project]
- 프로젝트를 git repository로 만들기
- 현재 하부 폴더에 새로운 저장소 생성
- [project] 없으면 현재 폴더를 저장소로 생성

$ git clone [git repository 주소]
- git 저장소를 로컬 저장소에 복제

$ git config user.name “[name]”
- 현재 로컬 저장소에 적용할 사용자 정보를 설정

$ git add
- commit 전까지 로컬 스테이징 영역에 변경사항을 저장

$ git commit -m “[메시지]”
- 스테이지 영역(로컬)에 올라온 변경된 내용을 원격저강소에 올리기

$ git status
- 작업 디렉터리와 스테이징 역역의 상태를 표시.
- add 목록 확인, commit 할 파일 확인 가능

$ git log
- commit 기록을 조회
- --graph 옵션으로 commit 기록을 시각화 하여 볼 수 있음

$ git diff
- Working Directory와 Staging Area 사이의 차이를 확인
$ git diff HEAD
- Working Directory Head Commit에 대한 Change 확인

commit 되돌리기

$ git reset [commit hash]
- 대상 커밋 해시와 현재 커밋 사이의 모든 커밋이 제거
- 전체 파일이 특정 커밋으로 돌아감

$ git revert [commit]
- 커밋의 변경사항을 제거하기 위해 새로운 커밋을 만들기
- 1,2,3 커밋이 있을 때 reset 하면 2,3 커밋이 삭제되므로 2만 삭제하고 싶을 때 revert 이용

$ git commit --amend
- 가장 최근 커밋, 커밋 메시지를 수정할 수 있다.

git branch

$ git branch [branch]
- 브랜치명을 지정하여 새로운 브랜치 생성
- 브랜치는 다른 브랜치 작업 이력과 독립적이다.

$ git checkout [branch]
- 대상 브랜치로 이동

$ git merge [branch]
- 지정 커밋을 HEAD 커밋과 병합

$ git rebase [base]
- 현재 분기를 base로 리베이스
- base는 커밋ID, 분기 이름, 태그 , HEAD..

원격 저장소

$ git remote add [name] [git repository 주소]
- 나의 원격 저장소를 이름을 붙여 등록

$ git fetch [remote] [branch]
- 원격 저장소의 브랜치를 가져와 확인
- 내용을 확인 후 merge 하므로 pull 했을때 자동 merge 되면서 발생할 수 있는 충돌을 막음

$ git pull [name]
- 원격 저장소 이름으로 변경 내용을 로컬 저장소로 가져와 병합

$ git push [name] [branch]
- 대상 원격 저장소의 대상 브랜치에 변경내용 올리기