# Git 사용 방법 정리(commit, push, pull request, merge 등

## Git

### Git 개념 및 명령어 정리

- 개인 개발을 넘어, 공동 개발에서 효율적인 코드 형상 관리를 하기 위함

### Git 영역

#### (1) Working Directory (Local)

- 개인 코드 작성

#### (2) Staging 영역

- git add 를 통해서 수정된 코드를 올리는 영역

#### (3) Repository

- git commi 을 통해서 최종 수정본을 제출

---

## Git 작업 플로우

## (1) 저장소 Repository 생성

- 원하는 폴더에서 git bash 실행 후

$git init

- 또는 기존 github에 있는 저장소를 내 로컬로 복제할 수도 있습니다.

$git clone(git 저장소의 URL)

## (2)staging 영역에 추가

- 코드 수정이 완료되면 staging 영역에 추가합니다.

$git add .

현재 디렉토리에 있는 업데이트 된 파일을 전부 스테이징 영역으로 추가합니다.

- 또는,

$ git add -A

수정된 파일 전부를 스테이징 영역에 추가합니다.

$git status

로 현재 add 내역을 확인할 수 있습니다.

## (3)Repository에 commit

$git commit -m "feat : README.md update"

- -m은 메세지의 약자이고, 뒤에 ""안에 공유할 메시지 내용을 적어준다.

## (4) 원격 저장소에 push, 업데이트 된 내용은 pull

- 내 local 디렉토리로 부터 원격저장소(Remote repository)로 보내기 위해서는 push 명령어를 사용합니다.
- 그 전에 원격 저장소와 내 로컬을 연결해야 한다.

### 원격 저장소 연결(github)

git remote add origin (원격 저장소 github URL)

- origin은 remote repository의 이름이며, 다른 이름으로 설정해도 무방하다.

### push

$ git push origin master

- origin 이라는 원격저장소의 master 브랜치에 푸쉬합니다.

### pull

- 또한 다른 사람이 원격 저장소(Remote repository)에 업데이트한 파일이 있을 때, 원격저장소와 내 로컬 저장소의 상태를 동일하게 만들기 위해 pull을 이용합니다.

$ git pull

## 그 외 알아두면 좋은 명령어

### 변경 내용 확인

$ git diff

### staging 취소 (unstage)

$ git reset

- 또는

$ git reset —hard

- working directory 까지도 변경전 상태로 되돌림 즉, 변경한 내용이 모두 소멸 됨으로 신중하게 사용

### Commit 정리

- 여러개의 commit을 하나의 commit 으로 정리

$ git rebase -i

- 직전과 금번 커밋을 하나로 정리

$ git commit —amend

### branch 확인

- 현재 설정된 브랜치 앞에 * 가 붙습니다.

$ git branch

### branch 생성 및 변경

$git branch mybranch

- 새로운 브랜치 mybranch를 생성했습니다.

$git checkout mybranch

-

기존 브랜치(master)에서 새로운 브랜치(mybranch)로 전환합니다.

따라서 바뀐 브랜치에서 commit을 하면, 원래 브랜치에는 업데이트가 안되고, 새로운 브랜치에서만 업데이트가 됩니다.

- 바뀐 브랜치를 사용하여 push 진행

$ git push origin mybranch

- origin 이라는 원격저장소의 mybranch로 push합니다.

## Pull Request와 Merge

commit을 한다고 최종 코드가 수정되는 것은 아닙니다.

개인이 commit을 했으면, 관리자가 이 코드를 리뷰하고 바꿀것이 있으면 수정해달라고 다시 요청해야하기 때문입니다. 그 과정을 알아봅니다.

## 1) Pull Request 발행 (Review 의뢰자)

github에 접속 후, 원격저장소에 들어가서 해당 commit의 pull request 버튼을 누르면 Reviewer에게 풀리퀘스트 메시지 전송

## 2) Review & Comment (Review 수행자)

리뷰 후 comment 할 것이 있으면 comment 버튼 클릭

## 3) Comment 대응 (Review 수행자)

local에서 코드 수정 후 원래와 같은 방식으로 commit 수행

```
$ git add .
$ git commit -m "커밋 메시지"
$ git push
```

## 4) Review 및 병합 (Review 수행자)

리뷰 후 최종 결과 만족 시 병합(merge) → github에서 pull request merge 버튼 클릭