git과 github

참고

https://shortcuts.tistory.com/8

요약

1. 깃과 깃허브는 상시 연동.
   깃 : 깃은 상태 변화를 체크해주고 로컬에서 사용됨
   깃허브 : public 공간. 공간만 제공해줌
2. 깃허브에 원격 저장소 (repository) 만들기
3. 로컬 저장소 만들기
4. 로컬 저장소와 원격 저장소 (이하 레포) 연결
5. 로컬 → 레포로 파일 올리기
6. 한 번 설정했으면 아래 명령어로 변경사항 연동시키기

```
$ git pull origin main
$ git add .
$ git commit -m "commit message"
$ git push origin main
```

첫 번째 줄의 pull은 해당하는 경우에만 하면 됨.

해당하는 경우 : 다른 팀원이 main 브랜치에 뭔가 추가해서, 내 컴퓨터의 내용과 차이가 생길 수 있다. 이 경우 그 팀원이 만든 변경사항을 내 컴퓨터에도 반영하기 위해 pull을 먼저 해줘야 한다.

(까먹고 push 해도 pull 하라고 할거임)

명령어 총 정리

```
# 레포 처음 만들면

## 로컬 저장소 만들고
$ git init

## 원격 저장소(repo)와 연결
$ git remote add origin [repo 주소]

// tip 1. 연결된 원격 저장소
$ git remote -v

// tip 2. 기존 원격 저장소와의 연결 삭제 (만약 잘못 연결했을 경우 등)
$ git remote rm origin

## 현재 브랜치 확인
$ git branch

// tip 1. 브랜치(branch) 이름 변경 (-m 이후 A에서 B로)
$ git branch -m master main

// tip 2. 브랜치 기본(defalut)이름 설정 (repo 만들때마다 항상 main으로
$ git config --global init.defaultBranch [브랜치 이름]

## README.md 가져오기
// 원격저장소의 파일 가져오기
$ git pull origin [브랜치 이름]

## add -> commit -> push
$ git add .

// 현재 브랜치에서 변경된 파일 목록 확인
$ git status

// tip 1. 만약 add 했는데, 이를 취소, 즉 장바구니(staging area)에서
$ git rm --cached -r . (add한 파일 모두 취소)
$ git rm --cahced [파일] (특정 파일만 취소)

$ git commit -m "commit message"
```

```
## push 하기
$ git push origin [브랜치 이름]

// 푸시할 때 항상 main 브랜치에 푸시하도록
$ git push -u origin main

// 다음부터는 아래처럼 생략 가능
$ git push
$ git pull

# 다음 작업 부터는?
$ git pull origin main
$ git add .
$ git commit -m "commit message"
$ git push origin main
```