깃허브 전반적인 사용법

1. 깃 연결
2. 깃 pull
3. Branch 확인
4. commit & push

1.연결

작업 폴더 이동 -> git init -> git remote add origin

(git remote -v) 로 확인

```
HOST@DESKTOP-8AM9ETF MINGW64 ~/Desktop/git_test
$ git init
Initialized empty Git repository in C:/Users/HOST/Desktop/git_test/.git/

HOST@DESKTOP-8AM9ETF MINGW64 ~/Desktop/git_test (main)
$ git remote add origin https://gh[illegible]@github.
com/intelligence-kim/test

HOST@DESKTOP-8AM9ETF MINGW64 ~/Desktop/git_test (main)
$ git remote -v
origin  https://ghp_2CQjjefbxfAdASrnznm3C4qUiTmosJOS6tSi@github.com/intelligence
-kim/test (fetch)
origin  https://ghp_2CQjjefbxfAdASrnznm3C4qUiTmosJOS6tSi@github.com/intelligence
-kim/test (push)
```

2.Git 푸쉬하기

1. Git pull

```
$ git pull origin main
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 17.01 KiB | 1.70 MiB/s, done.
From https://github.com/intelligence-kim/test
 * branch            main       -> FETCH_HEAD
 + 18df94c...0ff0866 main       -> origin/main  (forced update)
fatal: refusing to merge unrelated histories
```

2. Git add .

```
$ git add .
```

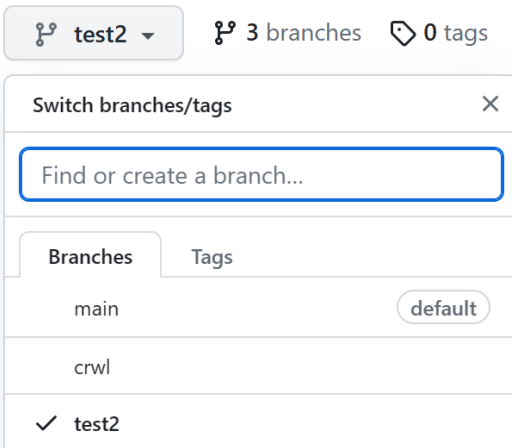
3. Git commit -m 'message'

```
$ git commit -m 'test pc'
on branch test2
nothing to commit, working tree clean
```

4. Git push origin [mybranch]

```
$ git push origin test2
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 786 bytes | 786.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'test2' on GitHub by visiting:
remote:      https://github.com/intelligence-kim/test/pull/new/test2
remote:
To https://github.com/intelligence-kim/test
 * [new branch]      test2 -> test2
```

결과화면



3.명령어

3.1 병합(강제 병합 사용은 자제)

3.1.1 git checkout [branch]

3.1.2 git branch main [branch] -f

3.1.3 git checkout main

3.1.4 git push origin main -f

3.2 브랜치 확인하기

3.2.1 git branch -a (모든 브랜치)

3.2.2 git branch -r (원격)

3.3 브랜치 관련 명령어

3.3.1 git checkout [branch] (브랜치 이동)

3.3.2 git branch –delete [branch] (브랜치 삭제)

3.3.3 git checkout -b [branch] (브랜치 생성)

3.3.4 git pull origin [branch] --allow-unrelated-histories

(프로젝트 기록 병합)

3.4 유저네임 및 이메일

3.4.1 git config user.name "someone"

3.4.2 git config user.email "someone@someplace.com"