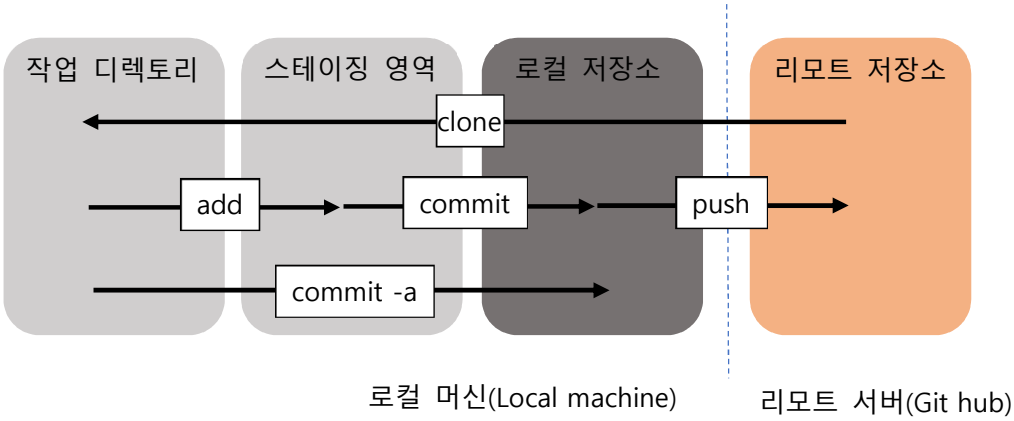

1. 저장소 복제(git clone)

/$ **cd ~/.**

~$ **git clone https://github.com/userID/test**

Cloning into 'test'...

remote: Counting objects: 14, done.

remote: Compressing objects: 100% (5/5), done.

.....생략

Checking connectivity... done.

~$ **ls -al | grep test**

drwxr-xr-x 9 rpi rpi 4096 Jan 21 05:48 test

2. 상태보기(git status)

~/test$ **git init**

~/test$ **echo "Just some text" > newfile.txt**

~/test$ **git status**

On branch master

Your branch is up-to-date with 'origin/master'.

Untracked files:

……생략

Nothing to commit, untracked files present(use "git add" to track)

추가할 파일 예외설정

~/test$ **echo "*.o" > .gitignore**

~/test$ **more .gitignore**

*.o

~/test$ **touch testobject.o**

~/test$ **git status**

On branch master master

Your branch is up-to-date with 'origin/master'.

Untracked files:

(use "git add<file>..." to include in what will be committed)

.gitignore

Newfile.txt

Nothing to commit, untracked files present (use "git add" to track)

testobject.o 파일이 git status상태에서 무시

3. 스테이징 영역에 추가(git add)

~/test$ **git add .**

~/test$ **git status**

Your branch is up-to-date with 'origin/master'.

Change to be commited:

(use "git reset HEAD <file>..." to unstage)

new file: .gitignore

new file: newfile.txt

스테이징 영역으로부터 파일을 지우려면 ~/test$ git rm somefile.txt

4. 로컬 저장소에 커밋(git commit)

~/test$ **git config --global user.name "Name"**

~/test$ **git config –global user.email mail@myEmail.com**

이 값들은 리눅스 사용자 계정에 대해 설정되므로 다음에 로그인할 때 남아있다.

more ~/.gitconfig 를 타이핑 해서 정보를 확인 할 수 있다.

로컬 Git 저장소로의 파일 추가를 영구적으로 커밋할려면 git commit명령을 사용한다.

~/test$ **git commit -m "Testing the repository"**

[master 3eea9a2] Testing the repository

2 files changed, 2 insertions(+)

create mode 100644 .gitignore

create mode 100644 newfile.txt

git commit -a 는 수정된 파일을 로컬 저장소로 직접 커밋하므로 add를 호출할 필요가 없다.

5. 리모트 저장소에 푸시(git push)

이 단계를 수행하기 위해서는 계정이 필요하다. git push 명령은 어떠한 코드 갱신이든 리모트 저장소로 푸시한다.

~/test$ **git config –global push.default simple**

~/test$ **git push**

Username for 'https://github.com': **UserID**

Password for 'https://UserID@github.com': **MyPassword**

Counting objects: 4, done

.....생략

~/test$ **git pull**

Already up-to-date

6. 브랜치 생성(git branch)

git checkout mybranch로 해당 브랜치로 전환이 가능하다.

~/test$ **Git branch mybranch**

~/test$ **git checkout mybranch**

Switched to branch 'mybranch'

~/test$ **touch testmybranch.txt**

~/test$ **ls**

testmybranch.txt

~/test$ **git add .** ← 스테이징에 추가

~/test$ **git status**

On branch mybranch

(use "git reset HEAD <file>..." to unstage ...이하생략

~/test$ **git add .**

~/test$ **git status**

On branch mybranch

Change to be commited:

(use "git reset HEAD <file>..." to unstage)

New file : testmybranch.txt

변경사항을 로컬 저장소의 mybranch 브랜치에 커밋할 수 있다.

이것은 mybranch에만 영향을 미치고 마스터 브랜치에는 영향을 주지 않는다.

~/test$ **git commit -m "Test commit to mybranch"**

~/test$ **git status**

On branch mybranch

~/test$ **git checkout master** <- 브랜치를 마스터로 전환후

Swtiched to branch 'master'

~/test$ **git merge mybrach** <- mybranch를 마스터 브랜치에 병합하여 put 하면

마스터 리모트 저장소에 적용

7. 브랜치 삭제

~/test$ **git branch -d mybranch**

참고문헌

데릭 몰로이, Exploring Raspberry pi, (옮긴이 최용, 2018년 04 05일) 3장1부 위키북스