

<기본설정>

\$ git --version

-> git 버전 확인 (버전이 제대로 나오면 git 다운로드 성공!)

\$ git config --global user.name "이름"

-> user name 설정

\$ git config --global user.email "이메일"

-> user email 설정 (버전(변경 사항)에 오류가 발생했을 시 연락을 할 수 단으로 사용)

\$ git config --list

-> 모든 설정값을 읽는 명령어 (마지막 줄에 user.name과 email 확인 가능!)

<주 명령어>

\$ git config --global init.defaultBranch main(master)

-> 기본 브랜치 이름(메인으로) 설정

\$ git config --get init.defaultBranch

-> 기본 브랜치 이름 확인

\$ git init

-> .git 폴더 생성(숨김 폴더, Git이 관리하는 프로젝트 내역 저장)

-> 새로운 git 저장소(repository) 생성

\$ git clone

-> 다른 사람이 이미 생성해 놓은 Git 원격 저장소 내려받기

\$ git add 파일명.확장자명 / git add .

-> 커밋 전까지 변경사항 모아놓음 (.은 모든 파일을 모아놓는 것)

\$ git status

-> 상태 확인 (Changes to be committed : 커밋 작업 가능)

\$ git commit / git commit -m “메세지 내용”

-> 변경 사항을 저장 (새로운 버전으로 만들어 local repository에 저장)

-> 메시지 내용을 포함해 커밋

\$ git diff

-> 수정 내역(변경 사항) 확인 (j : 올리기, k : 내리기)

\$ git log / git log --branches

-> 현재 브랜치의 수정 사항 확인 / 로컬에 존재하는 모든 브랜치 이력 확인

\$ git branch “브랜치명”

-> 새로운 브랜치 생성

\$ git branch

-> 현재 존재하는 브랜치 표시 (*은 현재 선택된(checkout된) 브랜치)

\$ git checkout ”브랜치명”

-> 브랜치 checkout (다른 브랜치로 이동)

\$ git checkout -b “브랜치명”

-> 새 브랜치 생성과 동시에 해당 브랜치로 이동

\$ git push origin ”브랜치명”

-> git remote 브랜치 생성

\$ git branch -set-upstream-to origin/”브랜치명”

-> branch local remote 연동

\$ git checkout main -> git merge “브랜치명”

-> main 브랜치로 이동한 후 main 브랜치에 병합하고자 하는 브랜치 병합

\$ git branch -d ”브랜치명” / git branch -D ”브랜치명”

-> 병합된 브랜치 제거 (브랜치로 checkout(이동) 후 브랜치 삭제)

\$ git branch -D “브랜치명”
-> 병합되지않은 브랜치 제거

<동일한 파일 수정시 충돌(CONFLICT)로 병합 실패가 될 경우>

1. \$ git status로 현재상태 확인 => 병합 하지 못한 파일 확인 가능
2. <<<<<<<<<< HEAD와 =====, >>>>>>>>>> develop으로 구분,
HEAD는 현재 브랜치 (git merge 명령어를 실행한 브랜치)
develop은 git merge의 명령어의 대상이 되는 브랜치
(git merge 브랜치명 <- 이거)
3. 두 내용을 확인해 필요한 내용만 남기거나 수정
5. 수정 완료후 git commit (충돌을 수정한 내용 커밋)
6. 자동으로 커밋 메시지 생성됨