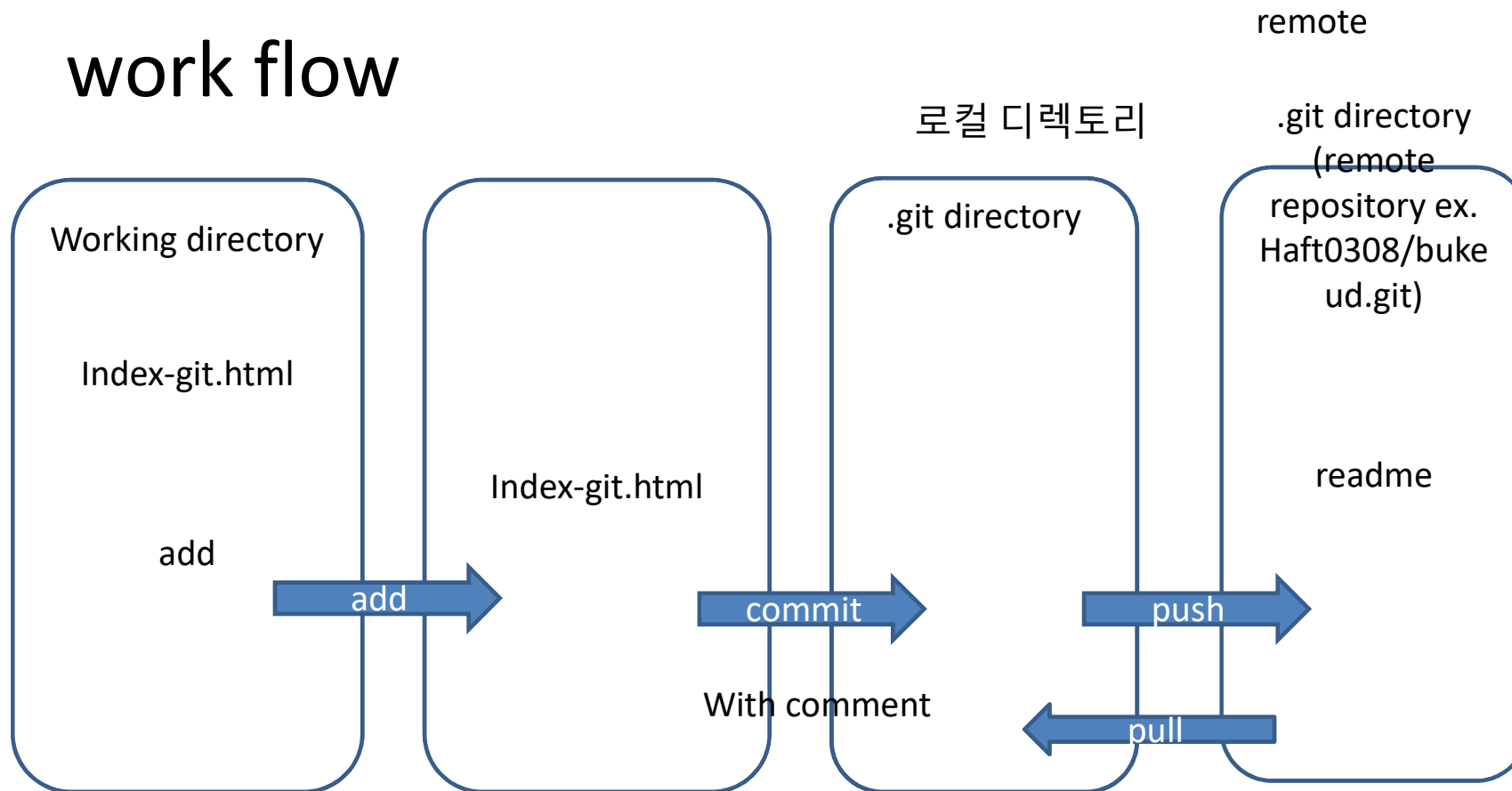


github

github 사용 소스 코드 올리기

1. github가입, git 프로그램 다운로드
2. github 에 repository(저장소)추가
3. 로컬의 공유할 폴더와 repository와 연결
4. 올리고자 하는 파일을 임시공간(staging area)에 add
5. commit으로 변경사항 확정 후
6. 임시공간(staging area)의 내용을 repository에 반영(push)

work flow



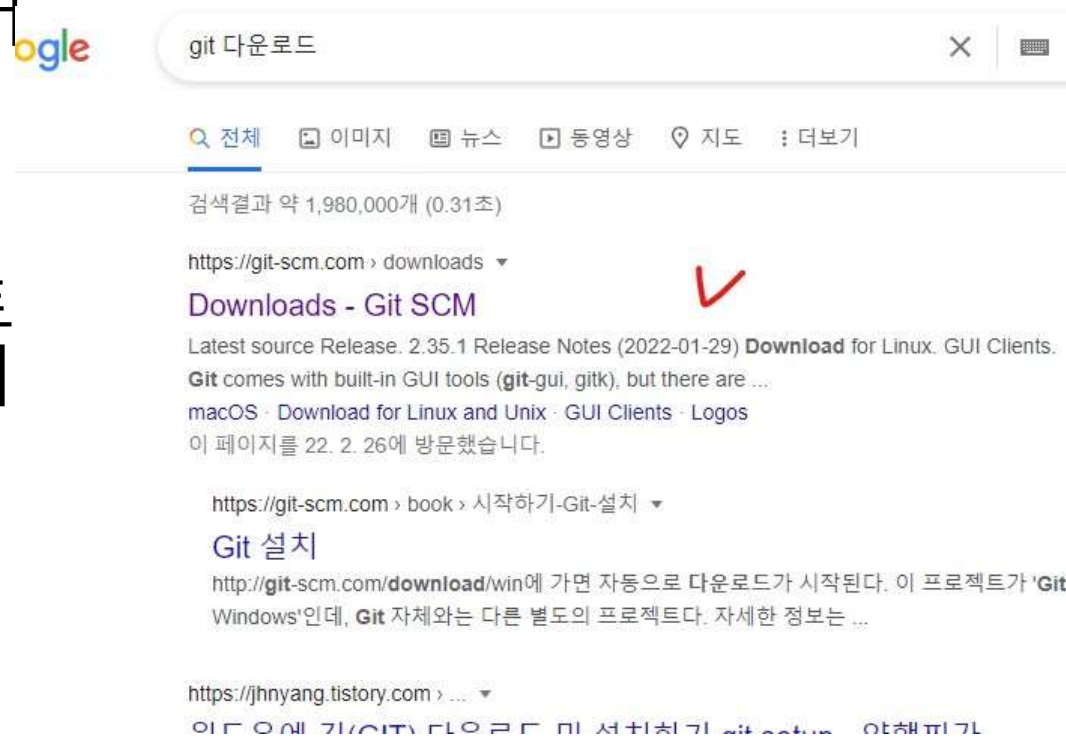
1. github

- 프로그램 소스를 관리하는 시스템
- 프로그램 작업시 협업하는 시스템
- <https://github.com/>
- 회원등록
- 블로그 처럼 나만의 독특한 프로
필로 어필.

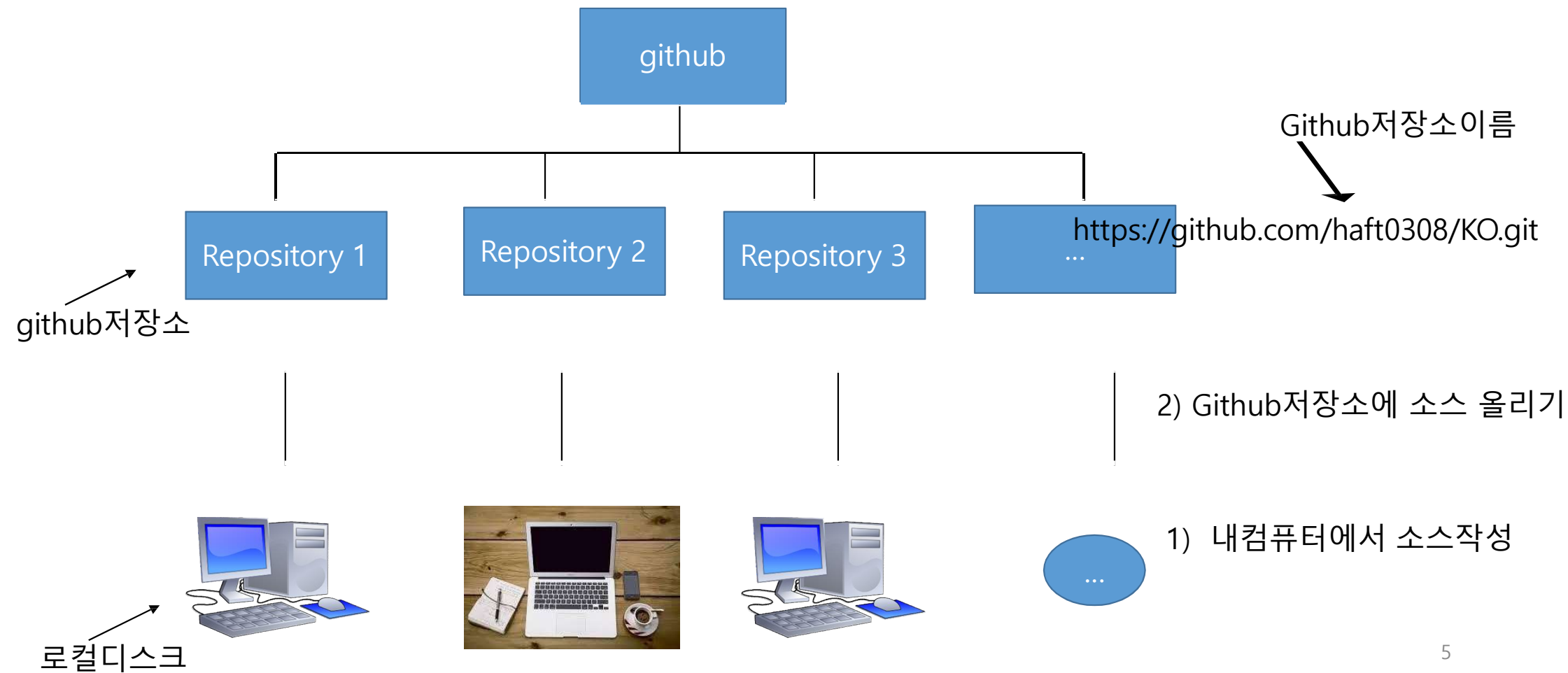
git

- github에 소스를 올리고, 버전관리하고 브랜치 관리할 수 있는 프로그램

- <https://google.com> 사이트에서 "git 다운로드" 검색어를 입력, 기본값으로 설치.



구조



3. 공유할 로컬 컴퓨터 소스로 이동

- 로컬 컴퓨터에서 github에 올리고자 하는 폴더를 정하자. d:\git_source를 만들어 공유하도록 하자
- 예)'d:\git_source'
- Vscode프로그램의 터미널에서 작업
- 공유하고자 하는 폴더로 이동(위의 폴더와 같이.
- `cd ../../../../d/git_source` 와 같이 해당 디렉토리로 이동

3. 공유할 로컬 컴퓨터 소스로 이동

- 로컬 컴퓨터에서 github에 올리고자 하는 폴더를 정하자. 이클립스의 workspace를 공유하도록 하자
- 예)d:\Eclipse-workspace
- 이클립스>터미널에서 작업
- 공유하고자 하는 폴더로 이동
- `cd ../../../../d/eclipse-workspace`와 같이 해당 디렉토리로 이동

git 환경설정

- git config --global user.name "koheeseon"
- git config --global user.email haft0308@gmail.com

로컬폴더와 github연결

- git init :처음 github와 로컬의 폴더를 연결시킬때.
- git remote add origin <https://github.com/haft0308/xx.git>
 - ✓(각자 할당된 영어이름.git으로 입력)
 - ✓잘못 입력하면, 지우고 다시 하자
 - ✓ git remote remove <https://github.com/haft0308/xx.git>
 - ✓변경하기
 - ✓ git remote set-url origin <https://github.com/haft0308/xx.git>
- git remote -v :연결된 사항(repository) 확인하기


로컬폴더 작업내용 임시공간에 적용

- `git add .` :폴더내 모든 파일을 올리자
- 혹은 `git add hello.java`를 통해 해당 파일만 올리자
- `git status` : 상태 확인
- `git commit -m "first"`
- 여기까지 수행하면 내가 올릴 소스 변경사항을 임시 공간에 add해 놓고, 금 곳고 first라고 이름지음.

임시공간 적용사항을 repository에 반영

- `git push origin master` : master브랜치로 github에 push.
- 웹페이지 github의 repository 선택해서 보면, 로컬의 소스가 반영되어 있음.

추가 변경사항 github에 반영하기

- 추가 수정사항이 있  음.
- `git add .`
- `git commit -m "change variable"`
- `git push origin master`
- Repository에 소스 내용과 동기화
 - (단, pull하기전에 내가 작업하던 사항은 commit으로 금그어놓고 난 후)
 - `git pull origin master`

집에 가서 github의 소스 다운로드

- 터미널에서 작업하고자 하는 경로로 이동 후
- git clone <https://github.com/haft0308/KO.git> (폴더명)
- 폴더에 데이터 다운로드함.

※ git 명령어(1/2)

분류	명령어	내용 설명
<새로운 저장소 생성>	\$ git init	.git 하위 디렉토리 생성 (폴더를 만든 후, 그 안에서 명령 실행 => 새로운 git저장소 생성)
<저장소 복제/다운로드(clone)>	\$ git clone <https:... URL> \$ git clone /로컬/저장소/경로 \$ git clone 사용자명@호스트:/원격/저장소/경로	기존 소스 코드 다운로드/복제 로컬 저장소 복제 원격 서버 저장소 복제
<추가 및 확정(commit)>	\$ git add <파일명> \$ git add * \$ git add -A \$ git commit -m "커밋 메시지"	커밋에 단일 파일의 변경 사항을 포함 (인덱스에 추가된 상태) 커밋에 파일의 변경 사항을 한번에 모두 포함 커밋 생성 (실제 변경사항 확정)
<가지(branch)치기 작업>	\$ git status \$ git branch \$ git branch <브랜치이름> \$ git checkout -b <브랜치이름> \$ git checkout master \$ git branch -d <브랜치이름> \$ git push origin <브랜치이름> \$ git push -u <remote> <브랜치이름> \$ git pull <remote> <브랜치이름>	파일 상태 확인 브랜치 목록 새 브랜치 생성 (local로 만듦) 브랜치 생성 & 이동 master branch로 되돌아 오 브랜치 삭제 만든 브랜치를 원격 서버에 전송 새 브랜치를 원격 저장소로 push 원격에 저장된 git 프로젝트의 현재 상태를 다운받 고 + 현재 위치한 브랜치로 병합

※ git 명령어(2/2)

분류

<변경 사항 발행(push)>

명령어

\$ git push origin master

\$ git push < remote > <브랜치이름>

\$ git push -u < remote > <브랜치이름>

\$ git remote add origin <등록된 원격 서버 주소>

\$ git remote remove <등록된 클라우드 주소>

내용 설명

변경사항 원격 서버에 업로드

커밋을 원격 서버에 업로드

커밋을 원격 서버에 업로드

클라우드 주소 등록 및 발행
(git에게 새로운 원격 서버 주소 알림)

클라우드 주소 삭제

<갱신 및 병합(merge)>

\$ git pull

원격 저장소의 변경 내용이 현재 디렉토리에 가져와지고(fetch) 병합(merge)됨

\$ git merge <다른 브랜치이름>

현재 브랜치에 다른 브랜치의 수정사항 병합

\$ git add <파일명>

각 파일을 병합할 수 있음

\$ git diff <브랜치이름> <다른 브랜치이름>

변경 내용 merge 전에 바뀐 내용을 비교할 수 있음

<태그tag 작업>

\$ git log

현재 위치한 브랜치 커밋 내용 확인 및 식별자 부여됨

<로컬 변경사항 return 작업>

\$ git checkout -- <파일명>

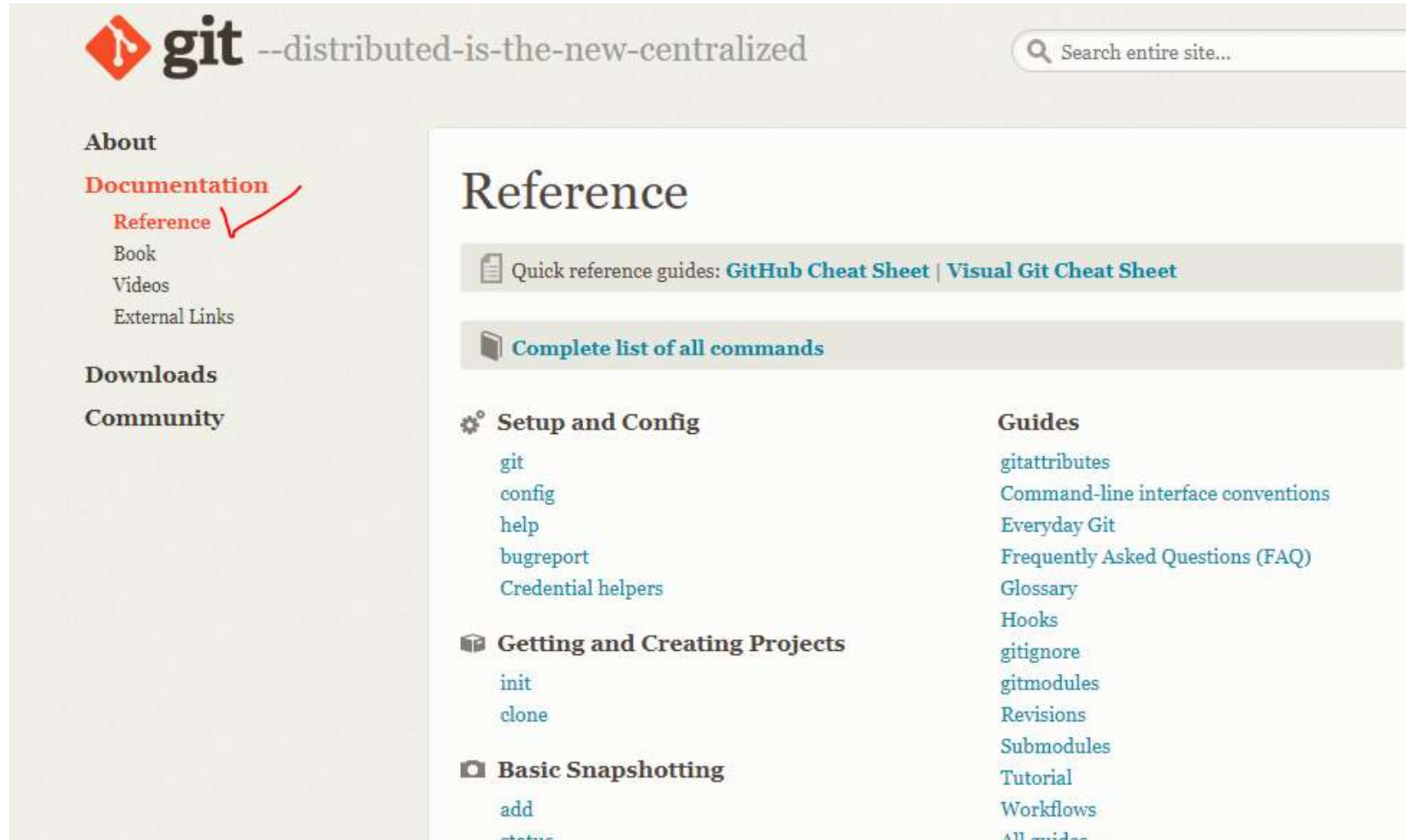
로컬의 변경 사항을 변경 전으로 되돌림

\$ git fetch origin

원격에 저장된 git프로젝트의 현 상태를 다운로드

참조해야할 문서

<https://git-scm.com/docs> 사이트 > document > reference



The screenshot shows the Git documentation website. The header features the Git logo and the tagline "--distributed-is-the-new-centralized". A search bar is located in the top right corner. The left sidebar contains navigation links: "About", "Documentation" (highlighted in red with a red checkmark), "Reference" (also highlighted in red with a red checkmark), "Book", "Videos", "External Links", "Downloads", and "Community". The main content area is titled "Reference" and includes a section for "Quick reference guides" with links to "GitHub Cheat Sheet" and "Visual Git Cheat Sheet". Below this is a link to the "Complete list of all commands". The "Reference" section is divided into three columns: "Setup and Config" (with links to git, config, help, bugreport, and Credential helpers), "Getting and Creating Projects" (with links to init and clone), and "Basic Snapshotting" (with links to add and status). A "Guides" column on the right lists various guides: gitattributes, Command-line interface conventions, Everyday Git, Frequently Asked Questions (FAQ), Glossary, Hooks, gitignore, gitmodules, Revisions, Submodules, Tutorial, Workflows, and All guides.

git --distributed-is-the-new-centralized

Search entire site...

About

Documentation

Reference ✓

Book

Videos

External Links

Downloads

Community

Reference

Quick reference guides: [GitHub Cheat Sheet](#) | [Visual Git Cheat Sheet](#)

[Complete list of all commands](#)

Setup and Config

- [git](#)
- [config](#)
- [help](#)
- [bugreport](#)
- [Credential helpers](#)

Getting and Creating Projects

- [init](#)
- [clone](#)

Basic Snapshotting

- [add](#)
- [status](#)

Guides

- [gitattributes](#)
- [Command-line interface conventions](#)
- [Everyday Git](#)
- [Frequently Asked Questions \(FAQ\)](#)
- [Glossary](#)
- [Hooks](#)
- [gitignore](#)
- [gitmodules](#)
- [Revisions](#)
- [Submodules](#)
- [Tutorial](#)
- [Workflows](#)
- [All guides](#)

※ github로 팀프로젝트하기

- 1. Github에서 소스코드 다운로드
 - git clone 주소 폴더이름
 - git clone <https://github.com/KO.git> KO (폴더이름)
- 폴더이름은 선택사항. (즉 없어도 됨) 폴더 이름을 줄 경우에는 그 폴더가 새로 생성이 되면서 그 안에 코드들이 다운로드가 되고, 폴더이름을 안 줄 경우에 github 프로젝트 이름으로 폴더가 자동으로 생기고 그 안에 코드들이 다운로드됨.

2. github에서 내 브랜치(branch)만들고 해당 브랜치(공간)로 이동

- git checkout -b 브랜치이름

3. 내 브랜치에 소스코드 업데이트하기

- git add .
- git commit -m "first commit"
- git push origin 브랜치이름

4. 내가 작업한 사항을 master에게 반영요청.(pull request)

- Create pull request
- (master는 pull request를 눌러 merge가 진행될 수 있게 한다. master 브랜치에 반영됨)

4. 마스터 브랜치에 소스 가져오기(pull)

- `git pull origin master`
- pull을 하기전에는 기존에 소스코드들을 commit을 먼저 해놔야 한다 (내가 작성하던 소스에 대해 commit하고 받아야함)

5. 브랜치끼리 이동하는 법

- `git checkout` 브랜치이름