Git Local Repository

무엇을 할 수 있을까?

버전 관리(Version Control), 백업(Backup), 협업(Collaboration)

2005년 리누스 토르발스(Linus Torvalds)

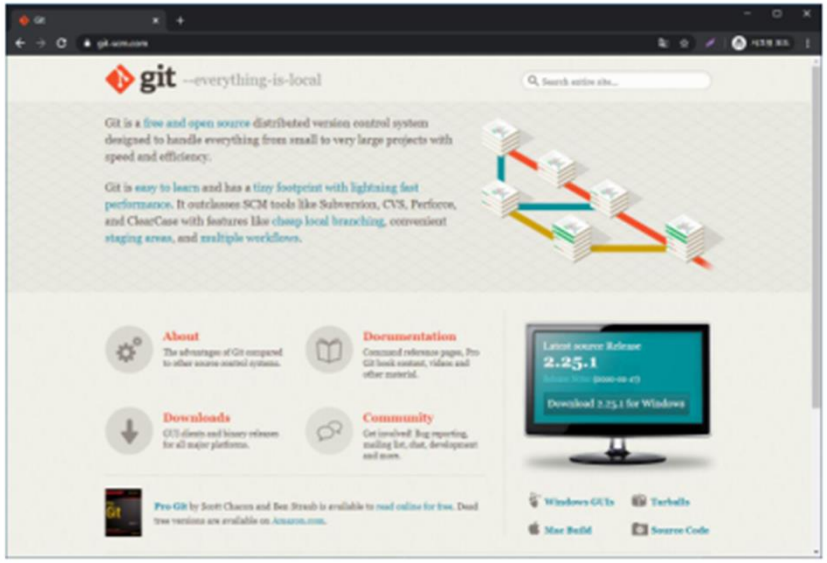
1. 버전 관리
   수정 날짜와 어떤 파일이 변경되었는지 구체적으로 기록된다

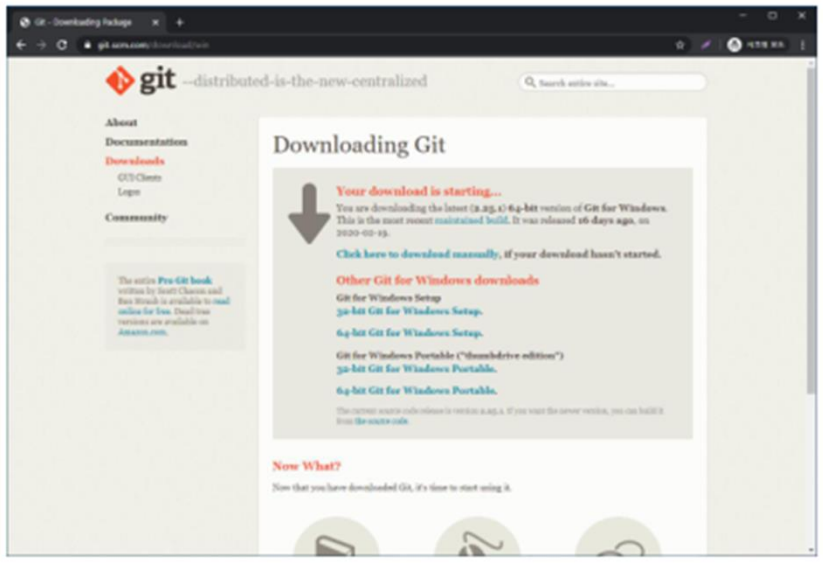
2. 백업
   드롭 박스나 구글 드라이버와 같이 온라인 저장소를 제공

3. 협업
   여러 사람이 Git에 저장된 파일을 가져와 수정 및 재 업로드가 가능하다.

Git 설치

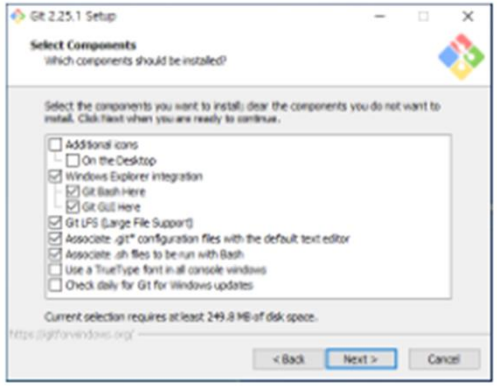
https://git-scm.com/

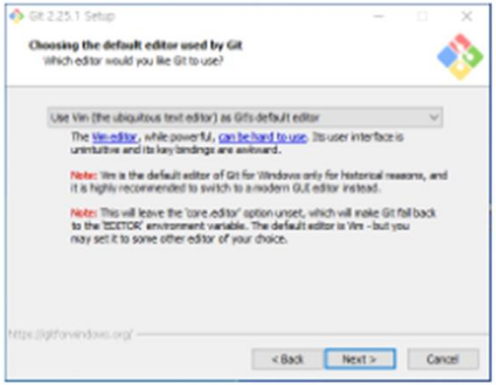




Git 설치

https://git-scm.com/

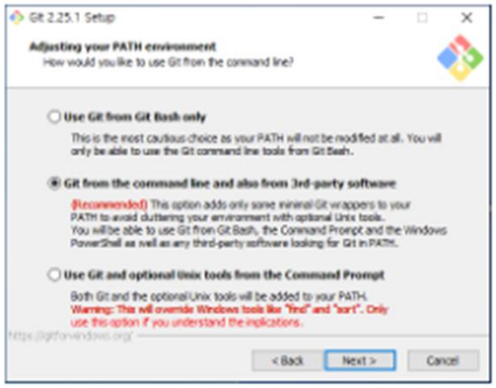


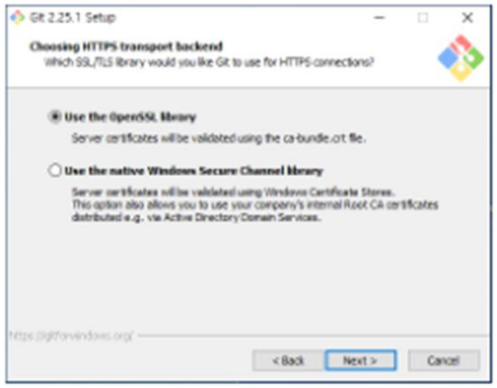


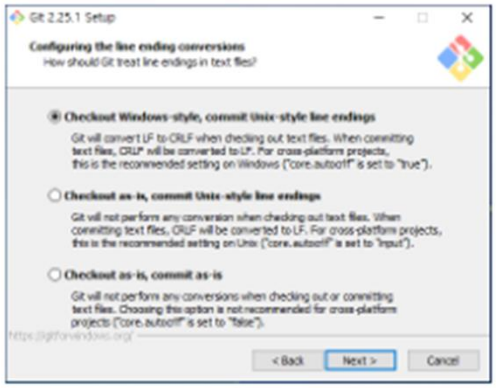
Git 기본 편집기는 Vim. VSCode나 nodtepad++ 등 선택 가능.

Git 설치

https://git-scm.com/



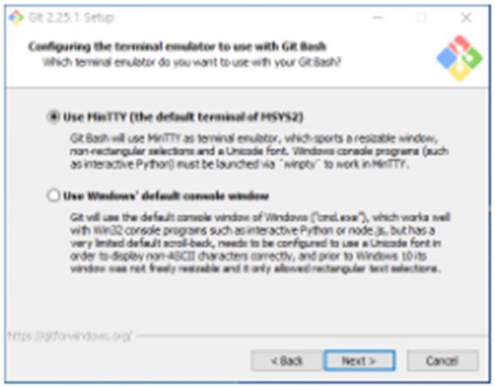


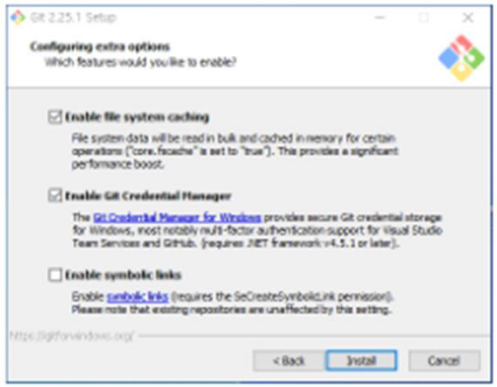


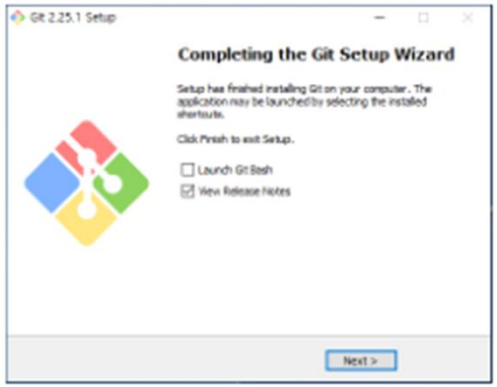
커맨드 라인 편집은 Gig from the commond line and from 3rd-party software.
보안 서버 접속 방법은 Use the OpenSSL library
텍스트의 끝 줄 바꿈 처리 방식은 Checkout Window-style, commit Unix-style line endings

Git 설치

https://git-scm.com/



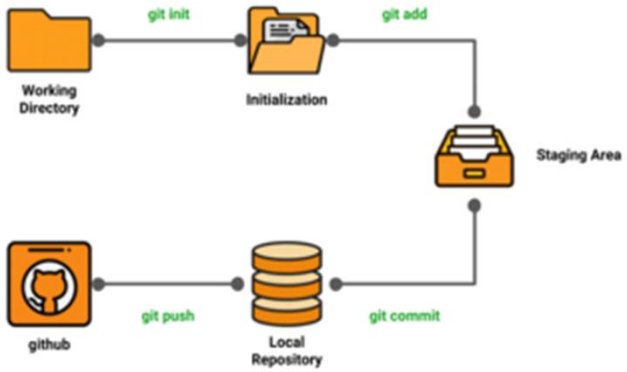




터미널 에뮬레이터 UseMinTTY (the default terminal of MSYS2)

Git Repository
저장소

저장소는 파일이나 디렉토리의 변경 내역(commit)을 관리하는 장소.
"로컬 저장소"와 **"원격 저장소"**가 있다.



로컬 저장소
- 개인 PC의 작업 디렉토리에 있는 저장소.
- 사용자 개인의 변경 이력을 관리.
- 작업 디렉토리의 .git이라는 숨겨진 디렉터리가 저장소의 실체.
- 작업 디렉토리에서 **git init** 명령으로 생성된다.

원격 Repository
서버에 있는 저장소. 여러 사용자가 변경 내용을 공유하는 저장소.

Git commit / staging

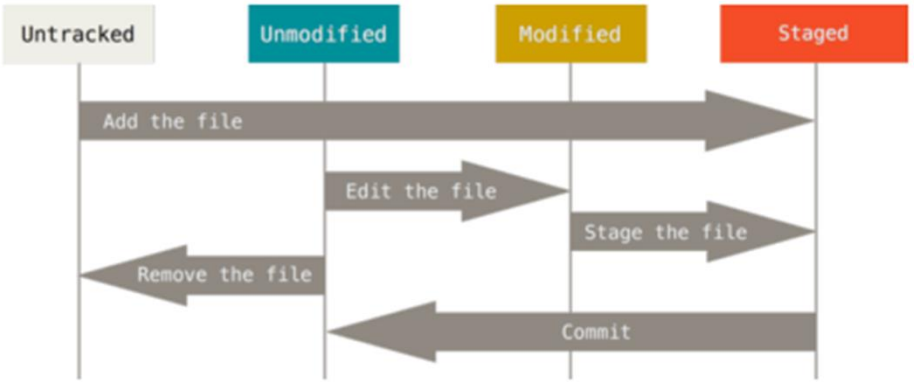
commit과 staging

commit

변경 이력을 로컬 저장소에 등록하는 것을 commit 이라고 한다.

Staging

stage 영역에 넣으려면 git add로 파일을 지정하고 색인을 생성.



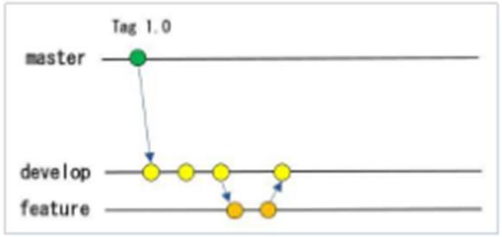
Git branch

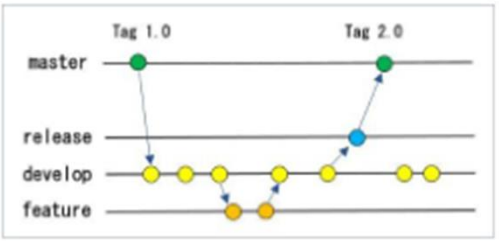
분기 – branch

브랜치

브랜치는 변경 내용을 분기하여 추진해 나가는 기능. 기본이 되는 master 브랜치가 기본적으로 있다.

지점을 이동하는 것을 체크 아웃이라고 한다. (master만 있는 경우 master 브랜치에 체크 아웃 된 상태라고 말할 수 있다.)





Git branch

분기 – branch

추적 브랜치

추적 브랜치는 정확히 "원격 추적 브랜치"라 한다. 그 이름대로 원격 변경을 추적하는 브랜치이다.

Remote에 위치: 원격 브랜치

로컬에 위치: 로컬 브랜치, 원격 추적 브랜치

Git branch

분기 – branch

fetch

원격 브랜치의 변경 이력을 추적 브랜치로 가져 오는 것을 패치라고 한다.

추적 지점의 현재 위치를 가리키는 포인터는 FETCH_HEAD가 된다. (로컬 브랜치는 HEAD).

git diff FETCH_HEAD .. HEAD

로, 추적 브랜치와 로컬 브랜치의 차이를 볼 수 있다.

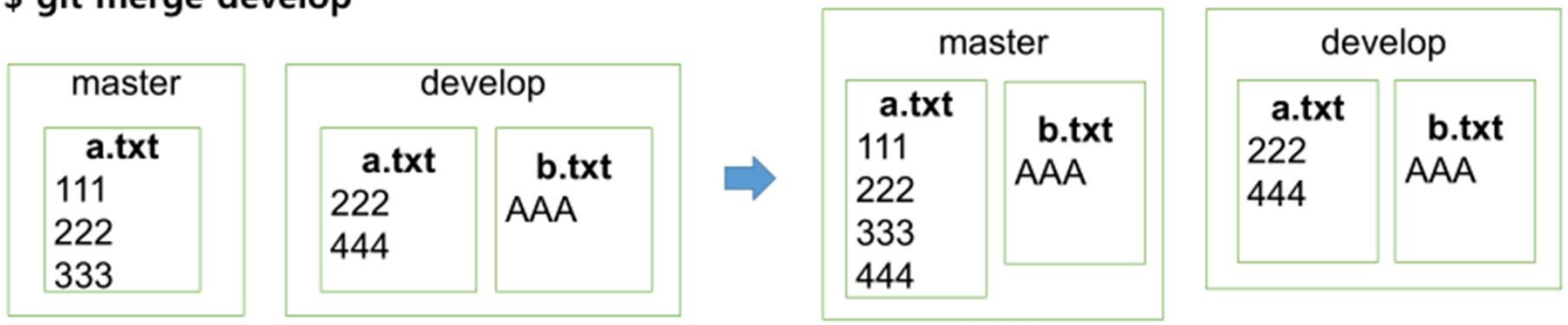
Git branch

분기 – branch

merge

로컬 브랜치에 추적 브랜치를 병합

$ git merge develop



Git pull / push

pull과 push

풀 (pull)

가져 오기(fetch) 및 병합(merge)을 동시에 실행하는 것을 pull 이라고 한다.
원격 브랜치의 변경 이력을 로컬 브랜치로 한번에 가지고 올 수 있다.

푸시 (push)

풀(pull)을 다운로드에 비유하면 푸시는 업로드이다.
로컬 브랜치의 변경 이력을 원격 브런치에 한번에 업로드한다.

Git clone

복사 - clone

clone은 원격 저장소의 내용 (commit 된 변경 이력)을 로컬 저장소에 복제하는 것이다.

업무 프로젝트에 참가한 경우
1. PC의 작업 디렉토리에 로컬 저장소 만들기(작업 디렉토리는 작업 트리가 된다)
2. 원격 저장소의 내용을 로컬 저장소에 복제(clone)
가 정상적인 흐름

Git HEAD

HEAD 지정

HEAD는 브랜치의 현재 위치를 가리키는 포인터로, 간단히 말하면 현재 commit ID의 별칭.

· HEAD

현재 위치(끝)를 가리키는 포인터

· 현재 위치보다 앞

물결표로 현재 위치 이전의 위치를 지정 할 수 있다.

HEAD~1: 하나 전

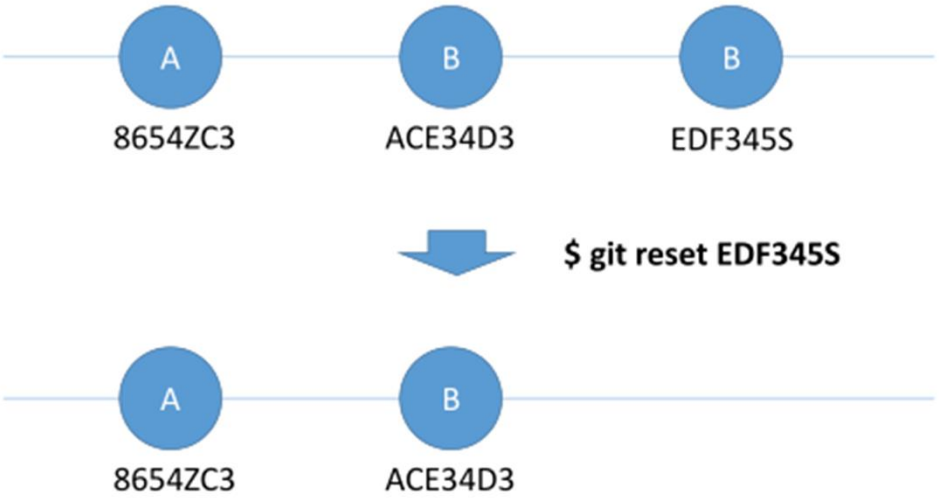
HEAD~2: 2 개의 전

Git RESET

reset

지정한 commit까지 재 설정.

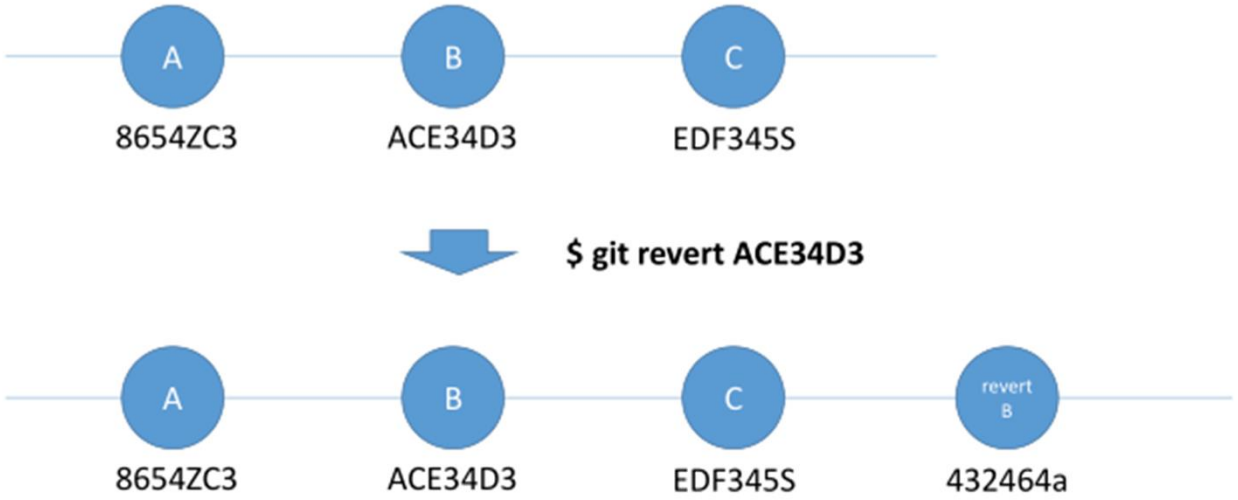
기본적으로 commit과 인덱스(staging 영역)은 재설정이며 옵션으로도 지정할 수도 있다.



Git revert

복구 - revert

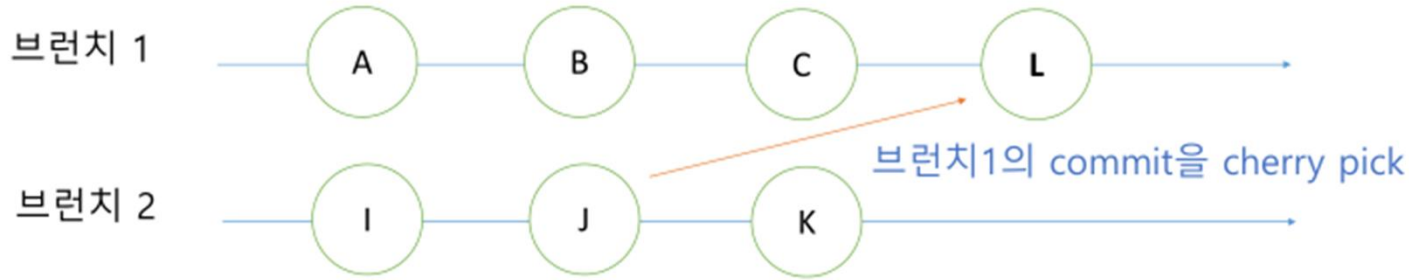
복구(revert)는 "commit을 참조"하여 commit을 만듭니다.
commit B를 참조하여 commit



Git cherry-pick

특정 commit – cherry-pick

체리 픽은 특정 commit을 도입.
다음의 경우 commit L는 실질적으로 C와 J를 병합 한 것이다.



Git stash

변경 숨김 [임시 저장] – stash

stash는 작업 트리의 변경을 일시적으로 숨기고 싶은 경우에 사용. [임시 저장]
필요하게 된 stash는 목록에서 제거 할 수 있다.

작업하는 동안 다른 일을 해야 할 때 (긴급 버그 대응 등)에 활용됩니다. 흐름으로서는
(1) 작업 트리 변경 사항을 숨김 (stash)
(2) 숨겨 놓은 목록을 확인 (stash list)
(3) 숨김 내용을 복구 (stash apply)
(4) 목록에서 제거 할 수도 있다 (stash drop)

Git tag

commit에 알기 쉬운 이름을 지정 - tag

commit에 알기 쉬운 이름을 지정하려면 태그를 사용한다.
Commit C는 EDF345S로 commit ID로 지정해도 좋지만 v1.1와 태그 이름으로 지정할 수도 있다. 로그에 표시