
Open source Software
오픈소스 소프트웨어
02. 버전관리 개요

학습 개요

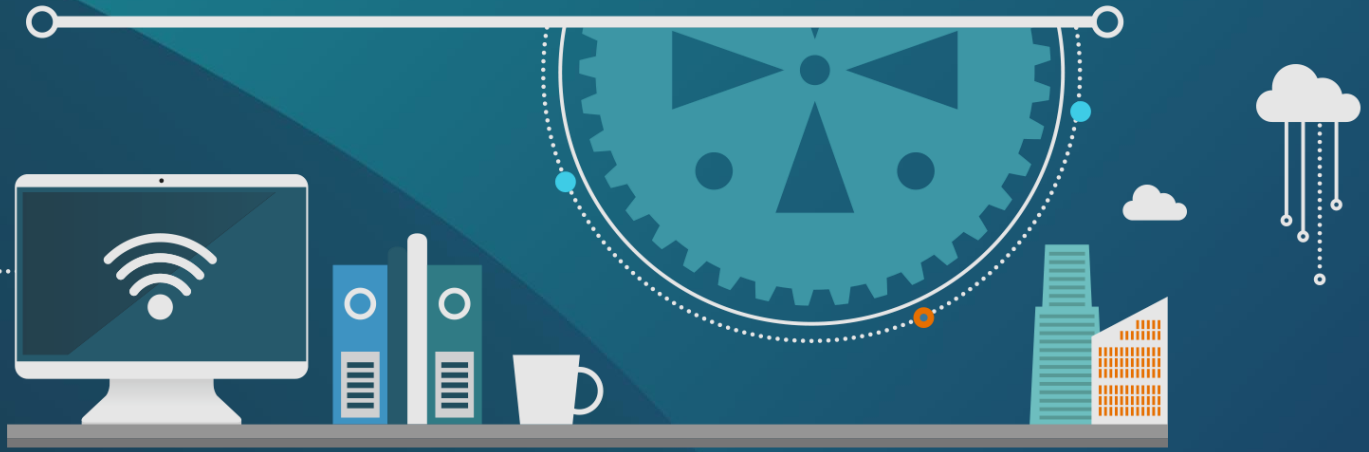
1. 버전관리의 필요성과 개념
2. 버전관리 도구의 인터페이스 방식
3. 커밋과 버전관리
4. 원격저장소와 지역저장소
5. Clone, push, pull, add, commit

학습 목표

1. 버전관리를 이해할 수 있다.
2. 버전관리 도구의 인터페이스 방식을 설명할 수 있다.
3. 원격저장소와 지역저장소를 이해하고 기본 용어를 설명할 수 있다

LESSON 01

버전관리 이해



1 버전관리 이해

버전 관리 필요성

- 지난 어느 시점의 파일이나 소스의 내용을 확인해 보고 그 상태로 되돌리고 싶을 때
  - 이름으로 관리의 문제
    - 이름을 수정하면서 관리
    - 관리할 파일이 많다면
      - ➡ 여러 이름의 새로운 폴더에 여러 파일을 복사해 관리

<출처> : https://backlog.com/git-tutorial/kr/intro/intro1_1.html

버전 관리 개념

파일의 추가 및 수정 이력(추적) 관리

- 누가(다중 사용자)
- 저장소가 여러 개인 경우(어느 저장소에서)
- 어느 파일을
- 언제
- 어디를
- 어떻게
  - 추가
  - 수정
  - 내용

1 버전관리 이해

버전 관리란?

- 시간 흐름에 따라 파일 집합에 대한 변경 사항을 추적, 관리
  - the version control system tracks changes to a file or set of files over time
  - 버전의 저장과 백업
    - 변경사항의 자세한 확인
  - 여러 사용자에 대한 버전 이력 추적관리
    - 소스 내용의 충돌에 대한 처리
    - 어떤 파일이 언제 어떻게 삭제되고 추가 됐는지 확인
- 필요하면 과거 어느 시점으로 이동
  - 마치 “시간 여행”

명령어 줄 인터페이스(Command Line Interface) 방식

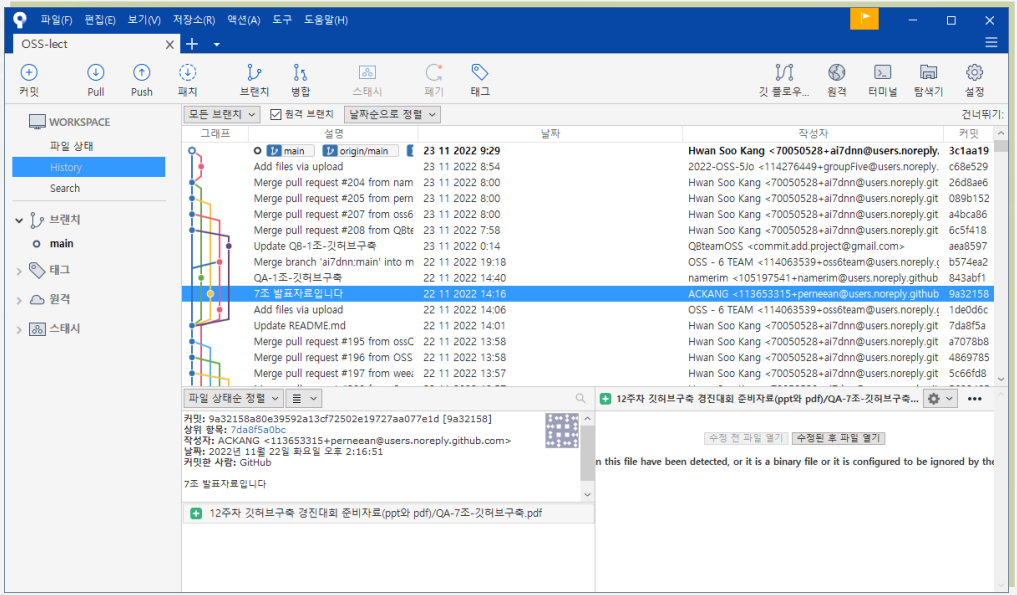
깃 도구 방식 중의 하나

- 텍스트(문자 text)로 명령을 입력하고 결과도 텍스트로 표시되는 인터페이스 방식
- 윈도의 명령 프롬프트(prompt)나 유닉스의 쉘(shell)

1 버전관리 이해

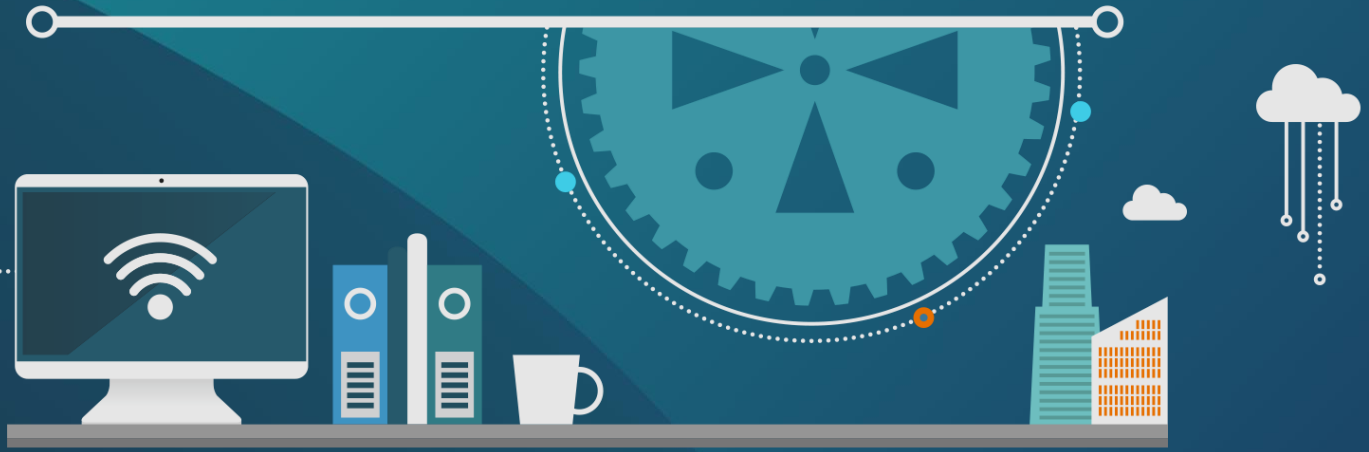
그래픽 사용자 인터페이스(Graphic User Interface) 방식

- 깃 도구 방식 중의 또 다른 하나
  - 윈도처럼 그래픽 대화 화면에서 마우스와 텍스트의 입력방식으로 명령을 입력하고 결과가 표시되는 인터페이스 방식
- 소스트리



LESSON 02

커밋과 버전관리



2 커밋과 버전관리

버전 관리의 커밋 commit

커밋(commit): 사전적 의미: ~적어두다

버전관리의 커밋

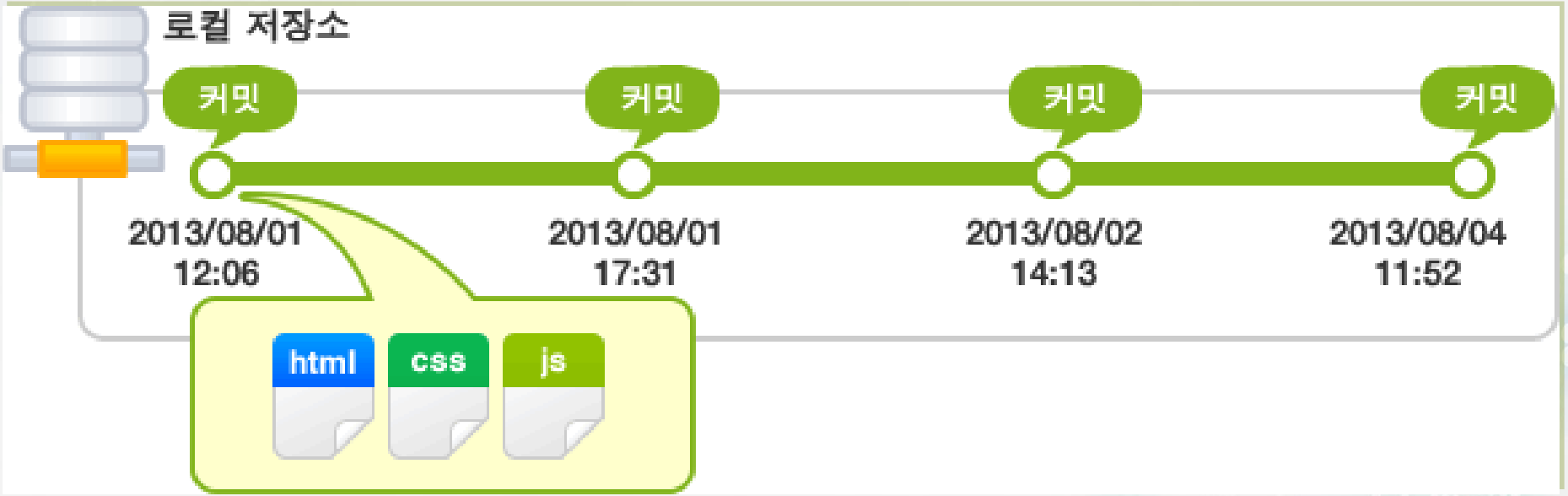
- 저장소의 현 상태를 저장하는 행위
  - 현 상태를 담는 스냅샵 사진을 찍는 것에 비유
- 파일 집합의 변경 내용을 깃 저장소에 기록하는 작업
  - 어느 시점의 파일 집합(폴더)의 추가/변경 사항을 저장소에 기록
    - ➡ 이전 커밋 상태부터 현재 상태까지의 변경 이력이 기록된 커밋이 생성

2 커밋과 버전관리

4번의 커밋 commit

시간순으로 저장

- 최근 커밋부터 거슬러 올라가면 과거 변경 이력과 내용을 알 수 있음



2 커밋과 버전관리

커밋 추적 관리

커밋

- 저장소의 현재 상태를 저장

저장소

- 연속된 커밋으로 관리
- 파일이 달라지지 않았으면 파일을 새로 저장하지 않음
  - 단지 이전 상태의 파일에 대한 링크만 저장

가장 최근 커밋 포인터 HEAD

가장 최근의 커밋을 가리키는 포인터: HEAD

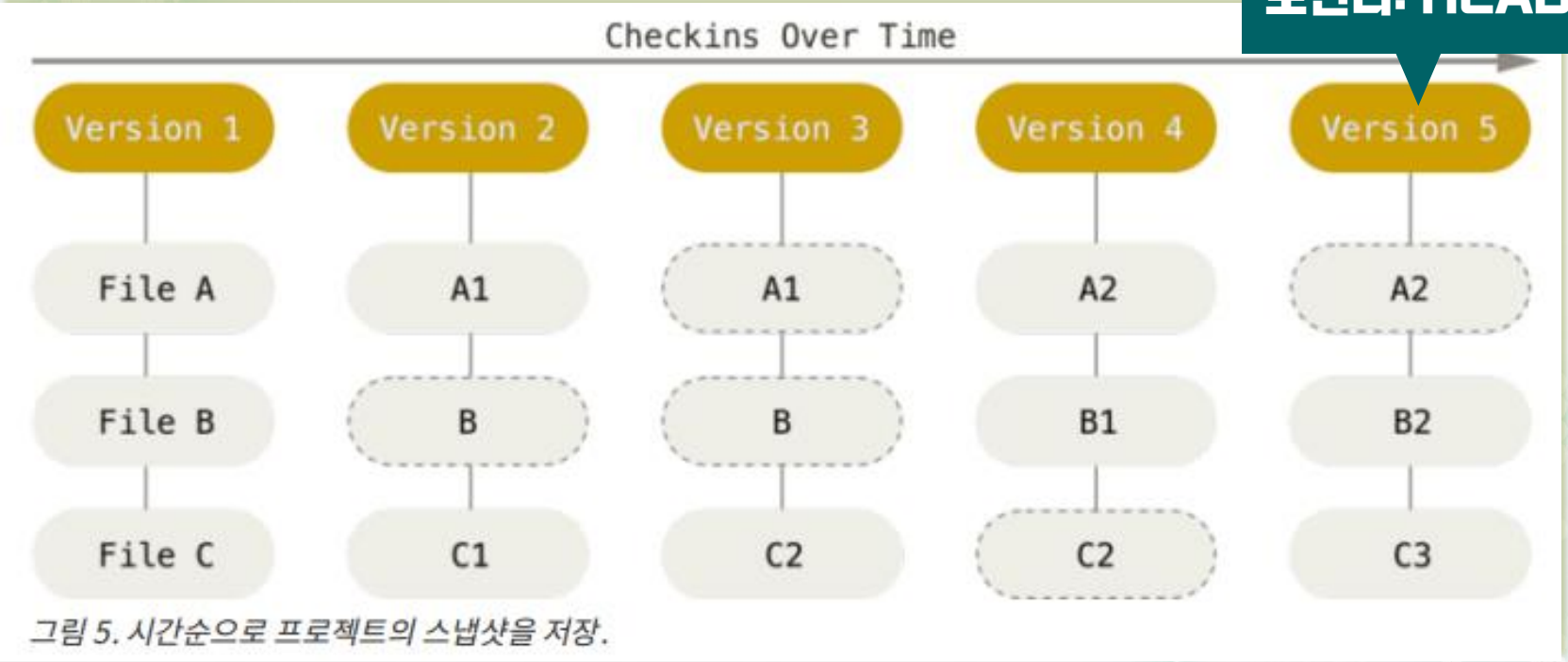


그림 5. 시간순으로 프로젝트의 스냅샷을 저장.

<출처> : https://git-scm.com/book/ko/v2/%EC%8B%9C%EC%9E%91%ED%95%98%EA%B8%B0-Git-%EA%B8%B0%EC%B4%88

이력을 관리하는 저장소

저장소(Git repository)

- 파일이나 폴더를 저장해 두는 곳
  - 파일이 변경 이력 별로 구분되어 저장

<출처> : https://backlog.com/git-tutorial

원격저장소와 지역 저장소

일반적으로 두 종류의 저장소를 제공

- 원격 저장소(Remote Repository)
  - 파일이 원격 저장소 전용 서버에서 관리되며 여러 사람이 함께 공유하기 위한 저장소
- 지역 저장소(Local Repository)
  - 내 PC에 파일이 저장되는 개인 전용 저장소

원격(서버) 저장소와 지역 저장소의 명령 1/2

Clone

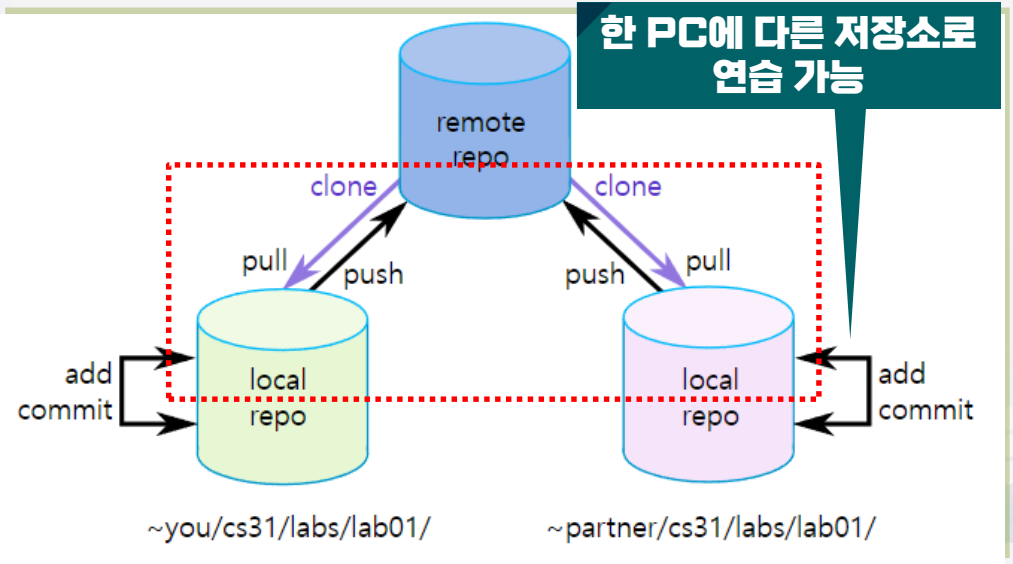
- 서버의 원격 저장소를 지역 저장소에 복제

Push

- 서버인 원격저장소로 올리기

Pull

- 지역 저장소로 내리기

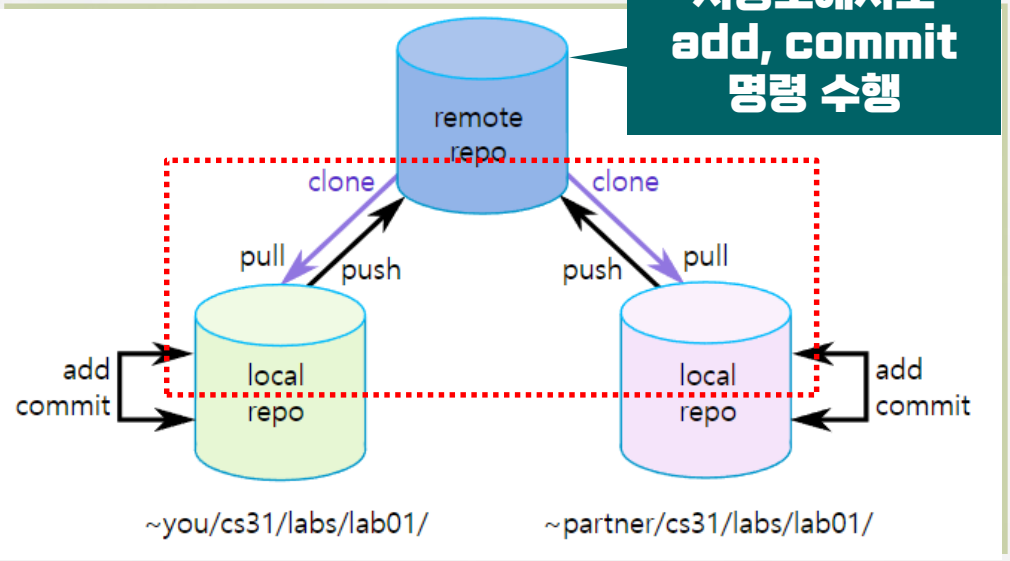


원격(서버) 저장소와 지역 저장소의 명령 2/2

파일을 저장소에 저장하는 명령어

- add
- commit

서버의 원격 저장소에서도 add, commit 명령 수행



원격(remote) 저장소와 지역(local) 저장소로 이동

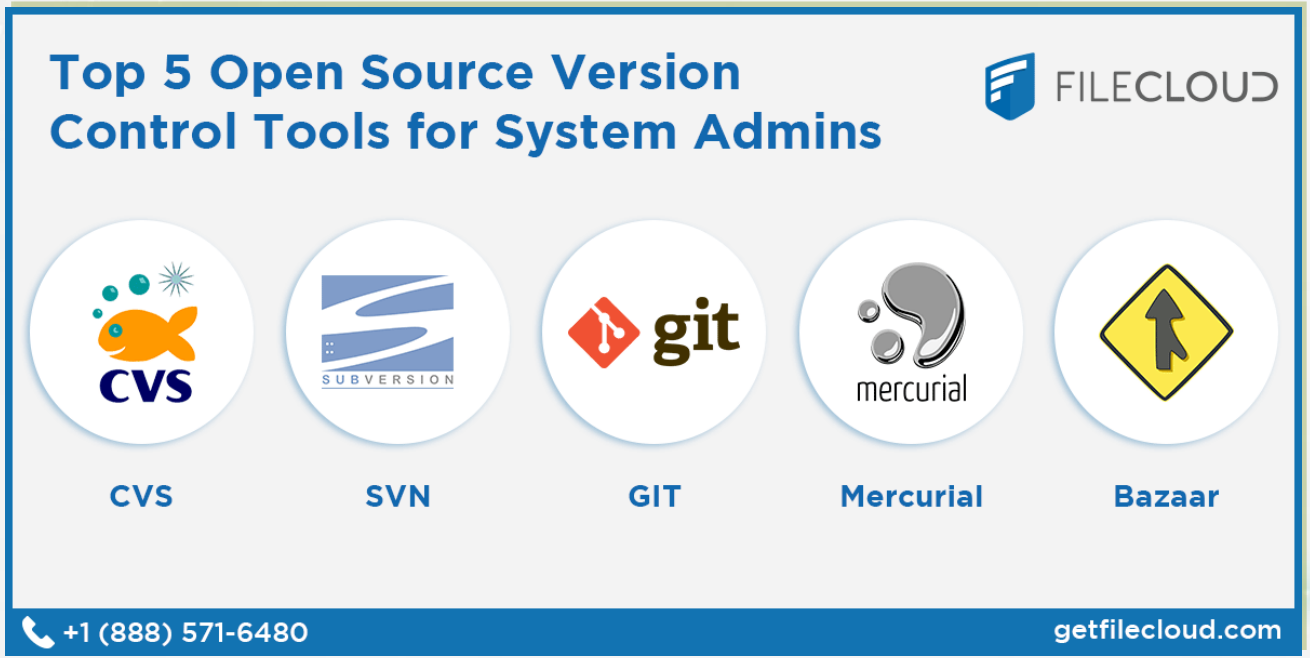
Push와 pull 활용

- Push: 원격저장소 올리기
  - 평소에는 내 PC의 지역 저장소에서 작업하다가
    - ➡ 작업한 내용을 공개하고 싶을 때에 원격 저장소에 업로드(push)
- Pull: 원격 저장소에서 지역 저장소로 내리기
  - 원격 저장소에서 여러 사람이 작업한 파일을
    - ➡ 지역 저장소로 가져(pull)올 수도 있음

다양한 버전관리시스템

Git

- 가장 많이 활용되는 분산 버전 관리 시스템



Summary

버전관리의 정의

- 시간 흐름에 따라 파일 집합에 대한 변경 사항을 추적, 관리

버전관리의 필요성

- 과거 지점의 버전으로 돌아가 누가, 무엇을 수정했는지 파악 가능

버전관리 도구의 인터페이스방식

- CLI와 GUI
  - Working folder, Working tree, Work space

Summary

커밋(commit)

- 저장소의 현 상태를 저장하는 행위이자 깃 명령어

HEAD

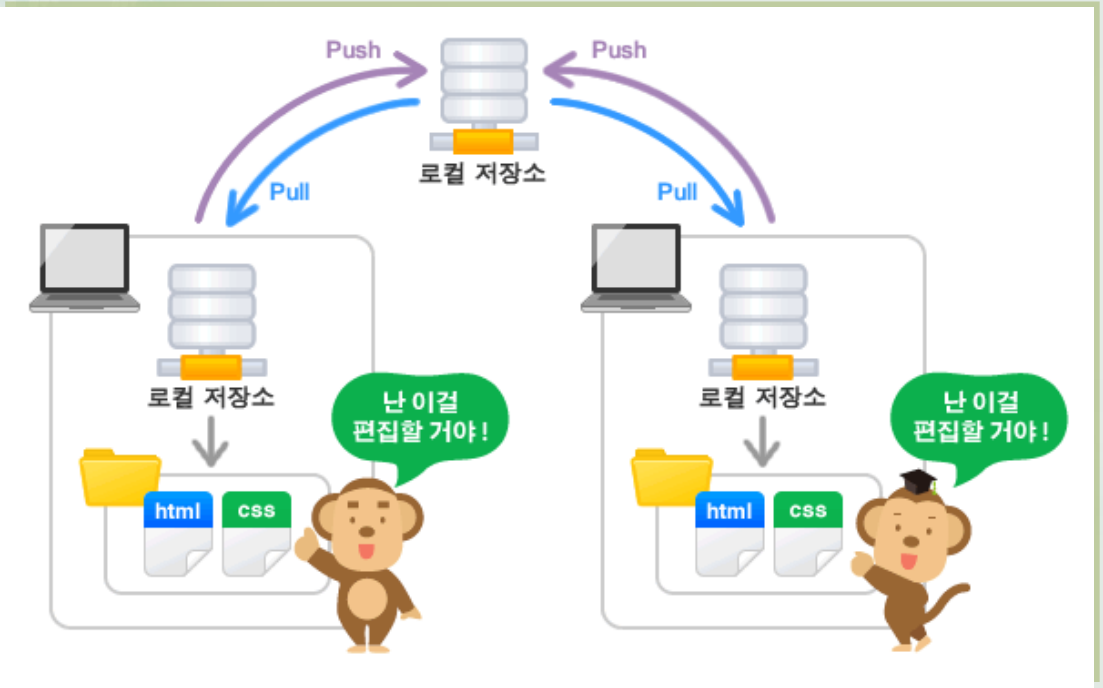
- 가장 최근의 커밋을 가리키는 포인터

저장소(Git repository)

- 파일이나 폴더, 버전관리를 위한 관련 파일을 저장해 두는 곳

Summary

Push와 Pull



<출처> : https://backlog.com/git-tutorial