GIT

git

- git
  - ✓ 컴퓨터 파일의 변경 사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 버전 관리 시스템(Version Control System) 또는 이를 위한 명령어
  - ✓ git은 2005년에 리눅스 커널 개발을 위해 초기 개발에 기여한 다른 커널 개발자들과 함께 2005년에 Linus Benedict Torvalds가 Linux 소스 코드를 관리할 목적으로 개발
  - ✓ git은 GNU 일반 공중 사용 허가서 v2 하에 배포되는 자유 소프트웨어
  - ✓ git의 장점
    - 이력 기록 및 추적 - 깃은 상세 이력이 기록되기 때문에 프로젝트에서 발생한 문제를 해결하는데 도움을 줌
    - 전 세계의 수많은 사용자가 사용 중
    - 서버 역할을 하는 원격 저장소 와 각 개발자의 로컬 저장소에 깃은 소스 코드 와 변경 이력을 분산 저장하기 때문에 원격 저장소에 문제가 생겨도 로컬 저장소를 이용하여 복원 가능
    - 변경 이력 병합 가능

git

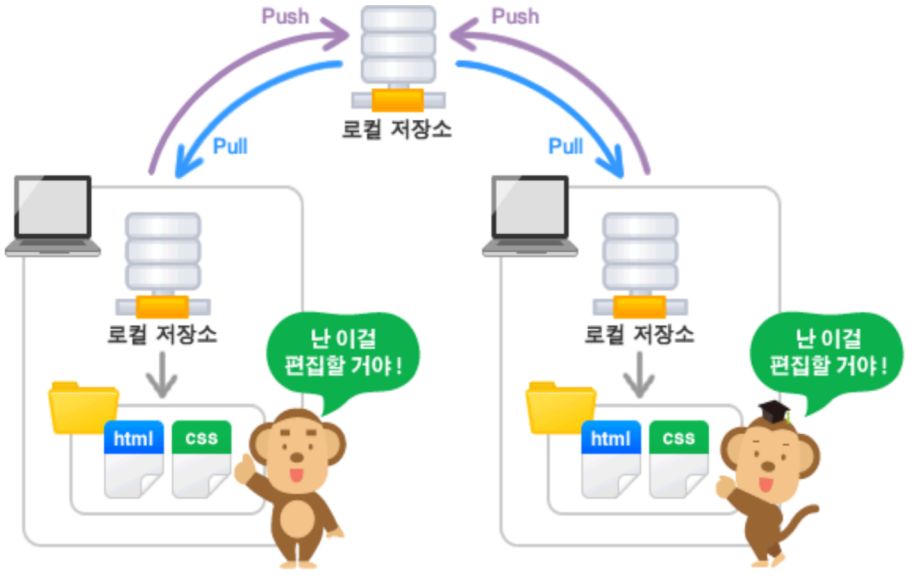
- ❖ Cloud Server Service
  - ✓ Git Hub
    - Ruby on Rails로 작성된 분산 버전 관리 툴인 git의 원격 저장소 호스팅 서비스
    - 영리적인 서비스와 오픈 소스를 위한 무상 서비스를 모두 제공
    - git이 텍스트 명령어 입력 방식인데 반해 git hub는 그래픽 유저 인터페이스(GUI)를 제공
    - git hub는 pastebin(텍스트 데이터를 저장하고 공유할 수 있는 온라인 저장소) 과 유사한 서비스인 Gist 와 Wiki를 Repository 마다 운영하고 있으며 git 저장소를 통해 고칠 수 있음
    - https://github.com 에 접속하여 계정 생성 및 로그인 후 작업 가능
    - git hub는 깃 프로젝트 저장소 역할 외에도 다양한 기능을 제공하는데 github Action 과 git hub Deployment API를 이용하면 빌드 및 배포 자동화를 구성할 수 있고 Project Boards를 이용해 프로젝트 협업 가능
    - Public 저장소는 무료로 생성할 수 있고 Private 저장소는 작업자 3인 이하인 경우에는 무료이고 설치형 버전인 Enterprise는 유료
    - 가장 인기있는 git 저장소 호스팅 서비스

git

- ❖ Cloud Server Service
  - ✓ GitLab
    - Ruby on Rails 로 만들어진 오픈소스 저장소 솔루션
    - 비공개 저장소를 참여 인원 수에 관계없이 무제한으로 생성할 수 있어 이곳을 사용하는 기업들이 많이 늘어남
    - 저장소의 최대 용량도 5GB
    - GitHub보다 훨씬 체계적인 이슈 트래커 기능과 CI(지속적 통합) 서비스 등을 제공
    - GitLab CE, GitLab EE라는 설치형 소프트웨어가 있으며 GitLab CE는 무료로 EE는 유료로 사용할 수 있음
    - 로컬 서버 시스템으로 마련: 오픈소스 솔루션이라서 별도 추가 비용 없이도 내부망 서버에 무료로 설치가 가능
  - ✓ BitBucket
    - 이슈 관리 시스템인 지라(Jira)를 만든 Atlassian이 모기업이어서 지라와 연동이 쉬움
    - 5명 이하 팀이면 공개 저장소 및 비공개 저장소 생성 무료이고 그 이상이면 월 3~6달러 부담

git

- ❖ 원격 저장소 와 로컬 저장소
  - ✓ git은 원격 저장소와 로컬 저장소 두 종류의 저장소를 제공
  - ✓ 로컬 저장소(Local Repository)
    - 내 PC에 파일이 저장되는 개인 전용 저장소
    - 새로 만들거나 원격 저장소를 로컬 저장소로 복사해서 생성
  - ✓ 원격 저장소(Remote Repository): 파일이 원격 저장소 전용 서버에서 관리되며 여러 사람이 공유하기 위한 저장소



git

- git 설치
  - ✓ Windows
    - https://git-scm.com/downloads에 접속해서 Windows 버전 다운로드 후 설치
    - https://www.sourcetreeapp.com/ 에서 sourcetree(git을 gui로 사용할 수 있도록 해주는 소프트웨어) 설치
  - ✓ Mac
    - git 설치
      - 다운로드 받아서 설치: https://git-scm.com/downloads에 접속해서 Mac 버전 다운로드 후 설치
      - brew를 이용한 설치: brew install git
    - https://www.sourcetreeapp.com/ 에서 sourcetree를 다운로드 받아서 설치
  - ✓ Linux
    - https://git-scm.com/download/linux 에서 각 배포 판 설치 방법 확인
    - Ubuntu
      - 최신 패키지 정보 업데이트

        sudo apt-get update
      - git 설치

        sudo apt-get install git

git

- ❖ git 설치 확인
  - ✓ git 버전 확인

    git --version
  - ✓ 깃의 기본 브랜치 이름을 main 으로 설정

    git config --global init.defaultBranch main
  - ✓ 사용자 설정
    - ● 명령어 - 현재 프로젝트에 등록
      - ◆ git config user.name 사용자이름
      - ◆ git config user.email 이메일
    - ● 명령어 - 모든 프로젝트에 등록
      - ◆ git config --global user.name 사용자이름
      - ◆ git config --global user.email 이메일
    - ● 확인
      - ◆ git config --global user.name
      - ◆ git config --global user.email

git

- ❖ git 설치 확인
  - ✓ 설정 값 확인
    - git config --list

credential.helper=osxkeychain

user.name=itggangpae

user.email=ggangpae1@gmail.com

init.defaultbranch=main

core.excludesfile=/Users/adam/.gitignore_global

difftool.sourcetree.cmd=opendiff "$LOCAL" "$REMOTE"

difftool.sourcetree.path=

mergetool.sourcetree.cmd=/Applications/Sourcetree.app/Contents/Resources/opendiff-w.sh "$LOCAL" "$REMOTE" -ancestor "$BASE" -merge "$MERGED"

mergetool.sourcetree.trustexitcode=true

http.postbuffer=1048576000

git

- ❖ 기본 개념
  - ✓ 작업 영역
    - Working Directory
      - ◆ 이력 관리 대상(tracked) 파일들이 위치하는 영역
      - ◆ 저장소 디렉토리에서 .git 디렉토리를 제외한 공간
      - ◆ 작업 중인 파일이나 코드가 저장되는 공간
      - ◆ Work Tree 라고도 함
    - Staging Area
      - ◆ 이력을 기록할(Commit) 대상 파일들이 위치하는 영역
      - ◆ .git 디렉토리 아위에 파일 형태로 존재
      - ◆ Index 라고도 함
    - Local Repository
      - ◆ 이력이 기록된(Commited) 파일들이 위치하는 영역
      - ◆ .git 디렉토리에 이력 관리를 위한 모든 정보가 저장되고 관리됨

git

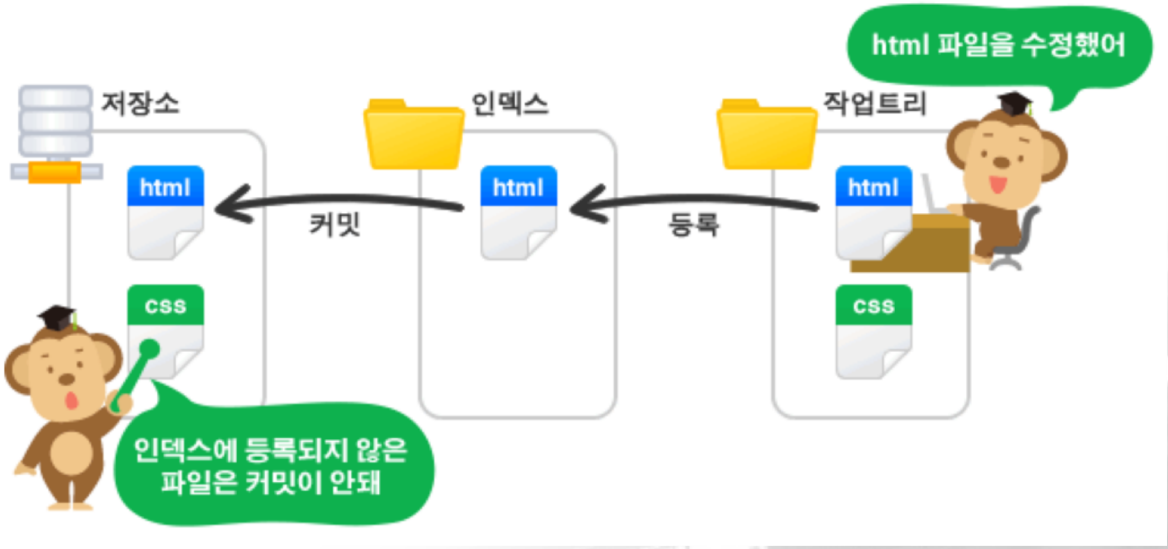
- 기본 개념
  - ✓ 작업 영역

git

❖ 기본 개념

✓ Work Tree 와 Index

- git에서는 작업 중인 디렉토리를 작업 트리(Work Tree)라고 부름
- Commit을 실행하기 전의 저장소와 작업 트리 사이에 존재하는 공간이 인덱스



git

- 기본 개념
  - ✓ Work Tree 와 Index
    - git의 Commit 작업은 작업 트리에 있는 변경 내용을 저장소에 바로 기록하는 것이 아니라 그 사이 공간인 인덱스에 파일 상태를 기록(staging)하게 되어 있기 때문에 저장소에 변경 사항을 기록하기 위해서는 기록하고자 하는 모든 변경 사항들이 인덱스에 존재해야 함
    - 10개의 파일을 수정했지만 그 중에 7개만 저장소에 공개하고 싶을 때 변경한 10개의 파일 중 7개를 선택하는 작업이 인덱스에 등록 또는 스테이징 이라 표현하는 작업
    - 인덱스란 공간이 중간에 있는 덕분에 작업 트리 안에 있는 Commit이 필요없는 파일들을 Commit에 포함하지 않을 수 있고 파일에서 내가 원하는 일부 변경 사항만 인덱스에 등록해 Commit할 수 있음

git

- ❖ 기본 개념
  - ✓ Commit
    - 파일 및 디렉토리의 추가/변경 사항을 로컬 저장소에 기록하는 것이 Commit
    - Commit을 수행하면 이전 Commit 상태부터 현재 상태까지의 변경 이력이 기록된 Revision이 생성됨
    - Commit은 순서대로 저장되므로 최근 Commit 부터 거슬러 올라가면 과거 변경 이력과 내용을 알 수 있음

git

- ❖ 기본 개념
  - ✓ Commit
    - 각 Revision에는 영문/숫자로 이루어진 40자리 고유 이름이 붙는데 저장소에선 이 40자리 이름을 보고 각 Revision을 구분하고 선택
    - 버그 수정, 기능 추가 등 특별한 의미가 있는 업데이트를 작업 별로 구분해서 각각 Commit하면 나중에 이력을 보고 특정 변경 내용을 찾기 쉬움
    - Commit은 이렇게 이력을 남기는 중요한 작업이기 때문에 Commit을 수행할 땐 Commit 메시지를 필수로 입력해야 하는데 메시지가 없으면 Commit이 실행되지 않음
    - git에서 권장하는 메시지 형식
      - ◆ 1번째 줄 - Commit 내의 변경 내용을 요약
      - ◆ 2번째 줄 - 빈 칸
      - ◆ 3번째 줄 - 변경한 이유

git

- 기본 개념
  - ✓ Working 디렉토리에서 관리되는 파일의 상태
    - Modified: 관리 중인 파일에 수정 사항이 감지되었지만 커밋이 되지 않은 상태
    - Staged: 감지 된 파일의 수정 사항이 Staging Area로 이동된 상태
    - Commited: 파일의 수정 사항에 대한 이력 저장이 완료된 상태
    - Untracked: 저장소에서 관리되지 않는 파일

git

❖ 기본 개념

✓ 작업 흐름