

GitHub

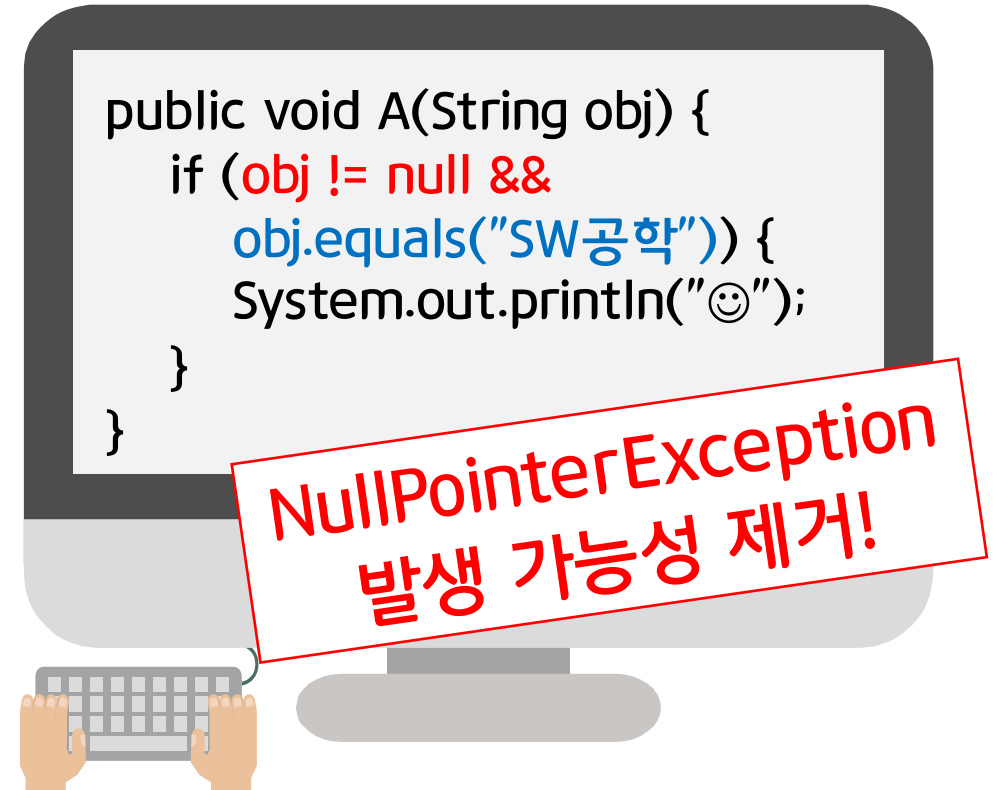
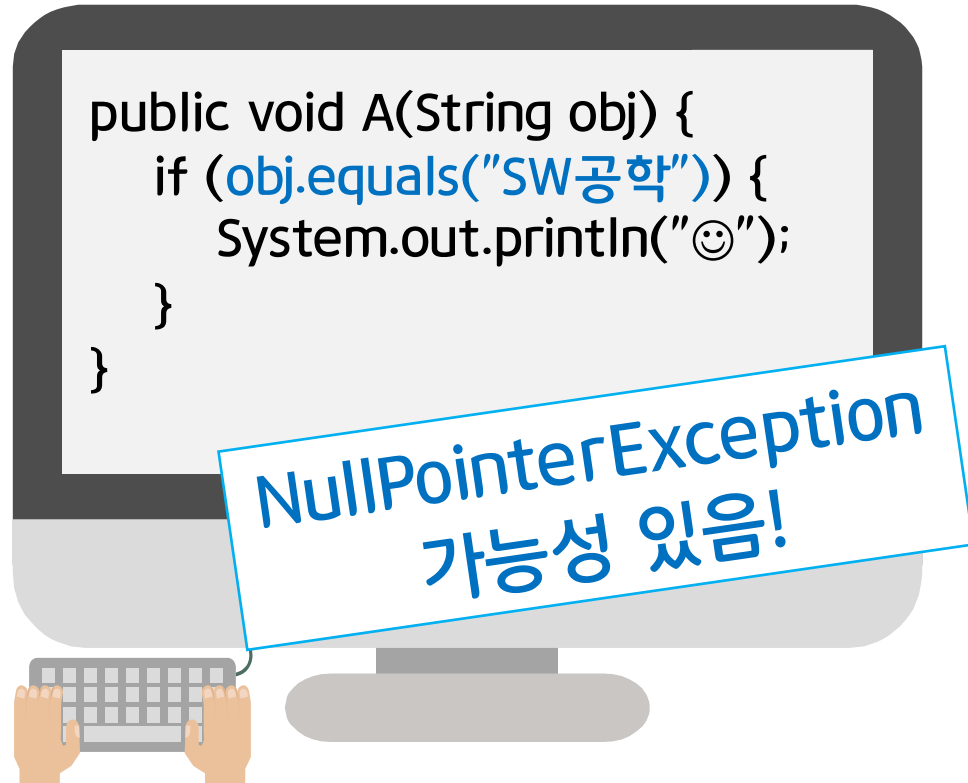
Desktop으로 시작하는 버전 관리 (Version Control)














버전(Version)

- 의미 있는 변경
 - '기능 개선' 혹은 '버그 수정' 등



버전 관리(Version Control)

- 정의
 - 의미 있는 변경 이력(change history)을 관리하는 것.
- 필요성
 - 변화되지 않은 소프트웨어는 없다.
 - 변화되지 않는다? 누구도 사용하지 않는다!
 - 따라서 버전 관리도 필수이다.

Name	
	120525_문서_업데이트.txt
	120604_문서.txt
	120605_문서_수정판.txt
	120605_문서_수정판2.txt
	120605_문서_최신 복사.txt
	120605_문서_최신.txt
	120605_문서.txt
	1200602_문서.txt
	문서_회의용.txt

▲ 일반인도 하는 버전 관리 예시

그렇게 똑똑하지 못한 버전 관리 사례

각 버전의 산출물을
모두 저장해 놓자!

report_v1.txt

모두들 그것을 알고 있는가?
컴퓨터교육과의 으뜸은
♡♡♡ 교수이다.



report_v2.txt

모두들 그것을 알고 있는가?
大 컴퓨터교육과의 으뜸은
갓 ♡♡♡ 교수이다.



report_v3.txt

모두들 그것을 알고 있는가?
大 컴퓨터교육과의 으뜸은
갓 ♡♡♡ 교수, **ㅇㅈ?**



report_v4.txt

모두들 그것을 알고 있는가?
大 컴퓨터교육과의 으뜸은
갓 ♡♡♡ 교수, **ㅇㅈ? ㅇㅇ**

N번 변경이 발생할 때마다,
공간 비용이 선형 증가한다.

# report_v1.txt	3KB
# report_v2.txt	4KB
# report_v3.txt	4KB
# report_v4.txt	5KB

저장소(Storage)

개선책: 원본 1개를 저장하고, 원본에 대한 변경 이력을 저장해 놓을 경우?

report.txt

모두들 그것을 알고 있는가?
컴퓨터교육과의 으뜸은
♡♡♡ 교수이다.

2번째 줄 앞에 **大** 삽입
3번째 줄 앞에 **갓** 삽입

모두들 그것을 알고 있는가?
大 컴퓨터교육과의 으뜸은
갓 ♡♡♡ 교수이다.

‘이다.’을 ‘, ㅇㅈ?’으로 수정

모두들 그것을 알고 있는가?
大 컴퓨터교육과의 으뜸은
갓 ♡♡♡ 교수, **ㅇㅈ?**

report.txt

변경로그.txt

#2 ... 2번 줄 앞에 **大** 삽입
3번 줄 앞에 **갓** 삽입
#3 ... 이다.을 , ㅇㅈ?으로
수정

버전 관리 시스템(Version Control System)

- 정의
 - 개발자의 버전 관리 편의를 도모한 시스템

- 종류

- 로컬 전용

- SCCS

1970년대 ~ 1980년대 중반

“Just For Me!”

“나 혼자 개발한다!”

- 클라이언트-서버

- SubVersion (SVN)

1980년대 ~ 2000년대 초반

“2인 이상의 개발자가 협력하기 시작!”

“한 곳에 모아놓고 버전을 관리하자.”

- 분산

- Git

2000년대 중반 ~

“저장 장소를 분산하자!”

“서버 연결이 끊겨도
로컬에서 작업할 수 있어야 한다!”

로컬 전용 버전 관리

- 혼자서만 버전 관리를 해야 할 때 적절.

report.txt

모두들 그것을 알고 있는가?
컴퓨터교육과의 으뜸은
♡♡♡ 교수이다.

2번째 줄 앞에 大 삽입
3번째 줄 앞에 갓 삽입

모두들 그것을 알고 있는가?
大 컴퓨터교육과의 으뜸은
갓 ♡♡♡ 교수이다.

‘이다.’을 ‘, ㅇㅈ?’으로 수정

모두들 그것을 알고 있는가?
大 컴퓨터교육과의 으뜸은
갓 ♡♡♡ 교수, ㅇㅈ?

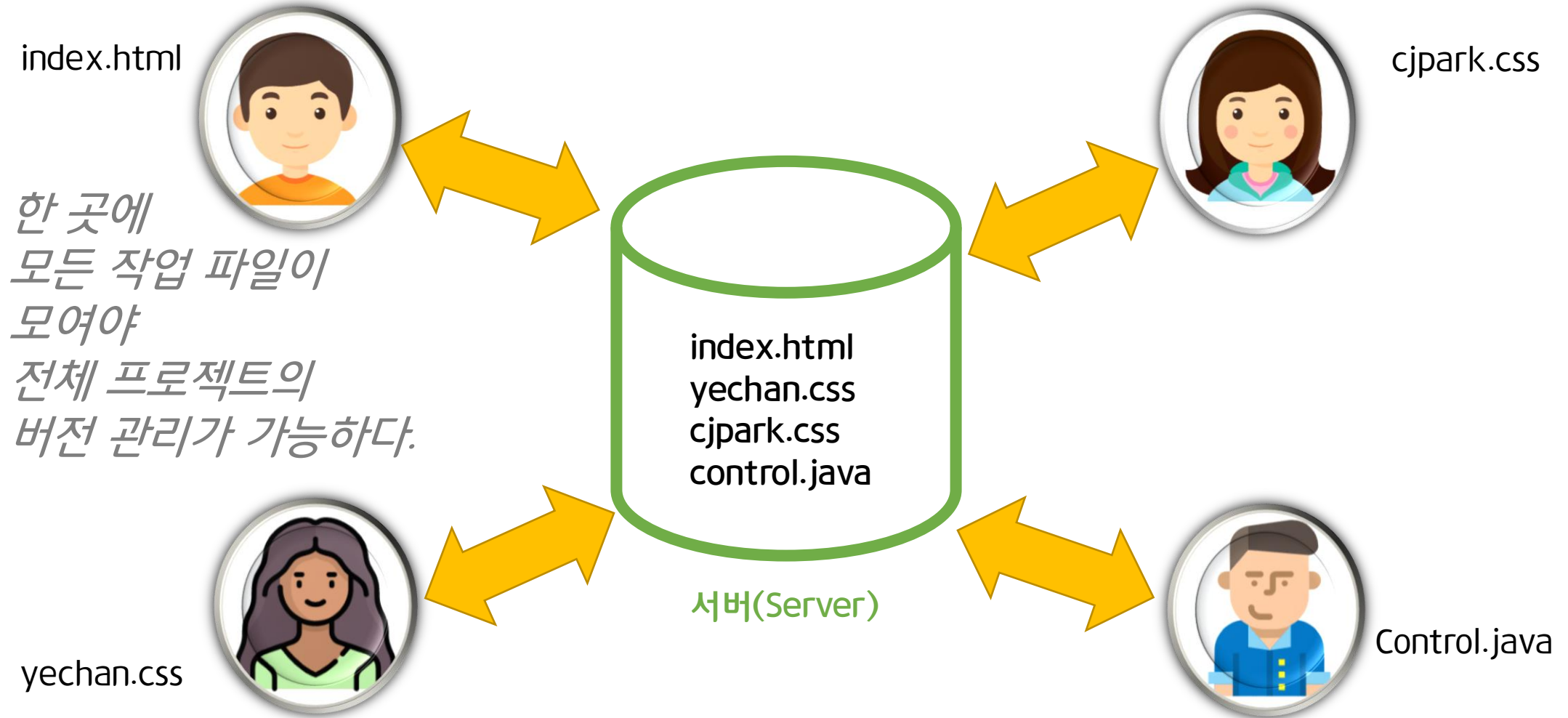
report.txt

변경로그.txt

#2 ... 2번 줄 앞에 大 삽입
3번 줄 앞에 갓 삽입
#3 ... 이다.을 , ㅇㅈ?으로
수정

클라이언트-서버 버전 관리

- 여럿이 협업할 때 일관된 버전 관리를 위해



분산 버전 관리

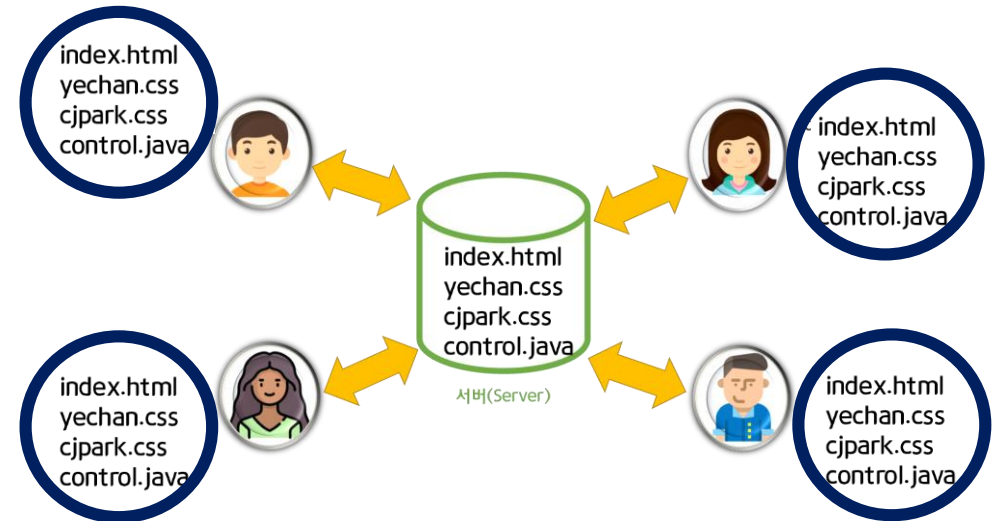
- 기존 클라이언트-서버 버전 관리 + alpha

- alpha?

- 모든 프로젝트의 산출물을 서버에만 두지 말자.
- 모든 프로젝트의 산출물을 **로컬에도 복사해 두자**. (분산해서 2군데 저장하자.)

- 분산의 유익?

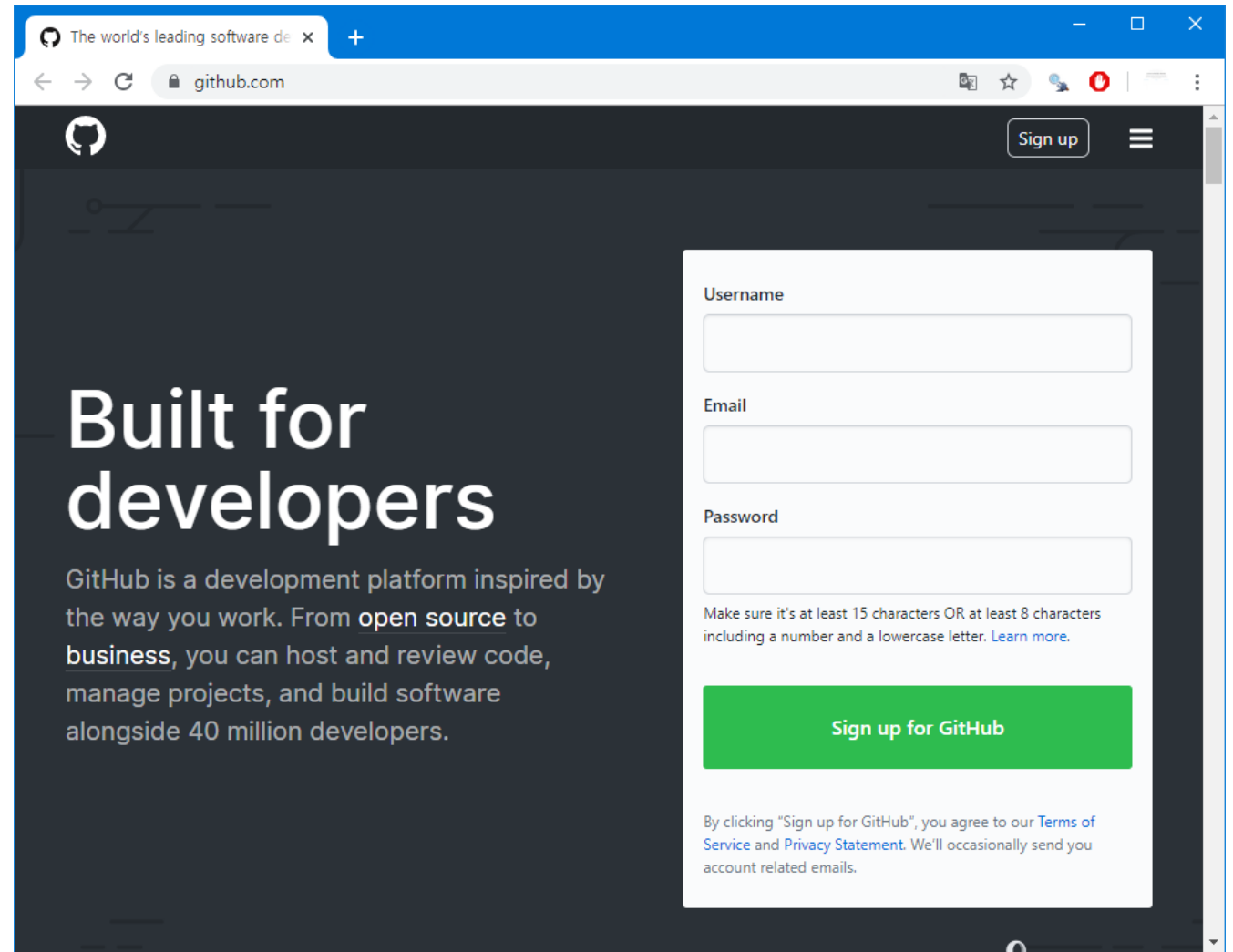
- 다음과 같은 상황에도 개발이 가능하다.
 - 서버의 상태가 불안정할 때
 - 클라이언트가 서버에 접속 불가능한 상태일 때





GitHub

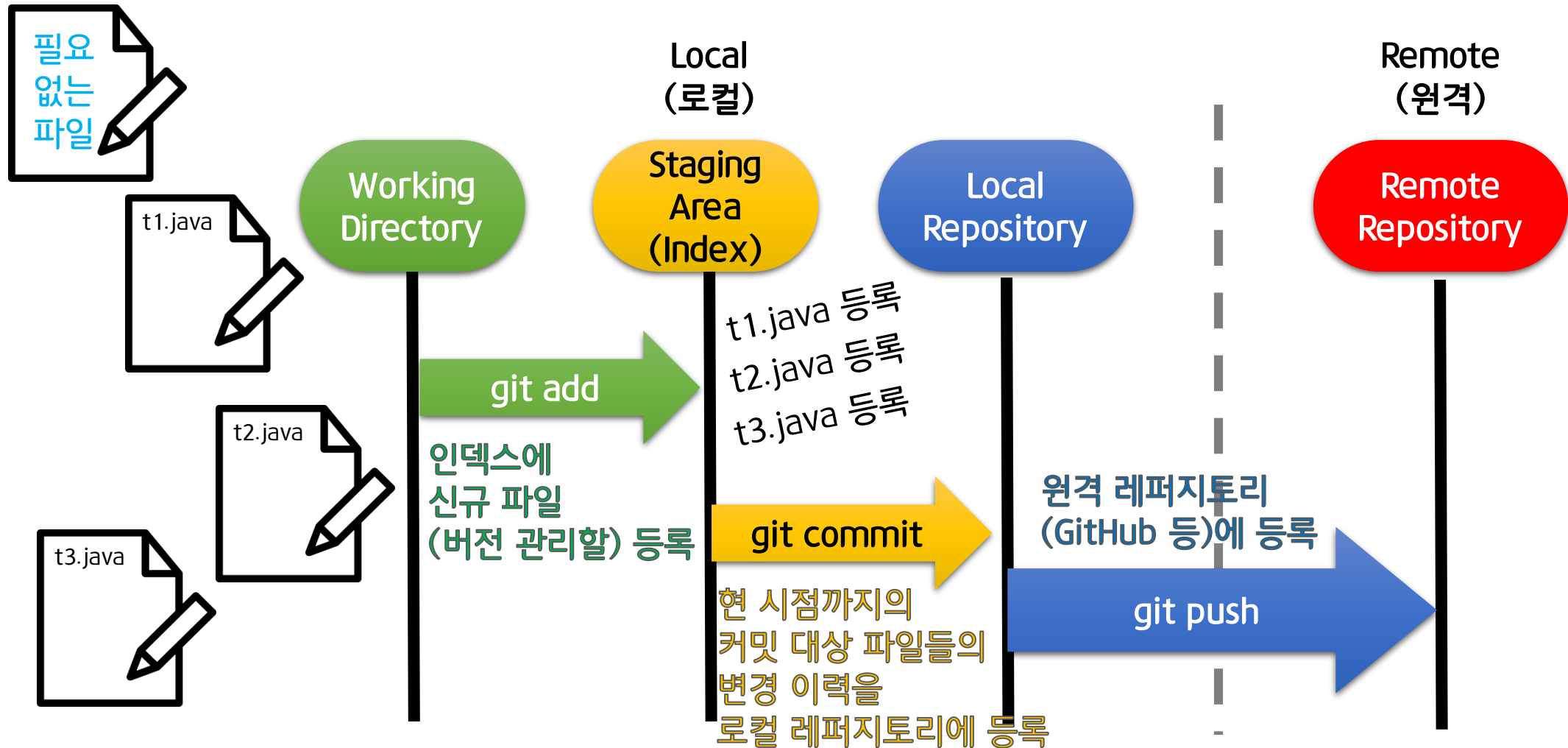
- 분산 버전 관리 도구인 Git을 사용하는 프로젝트를 지원하는 웹 호스팅 서비스.
- 무료 서버 공간을 획득했다고 생각하면 좋음. 단, 내 소스를 누군가에게 보여주어야만 하는 것이 기본 원칙.
- <https://github.com/>



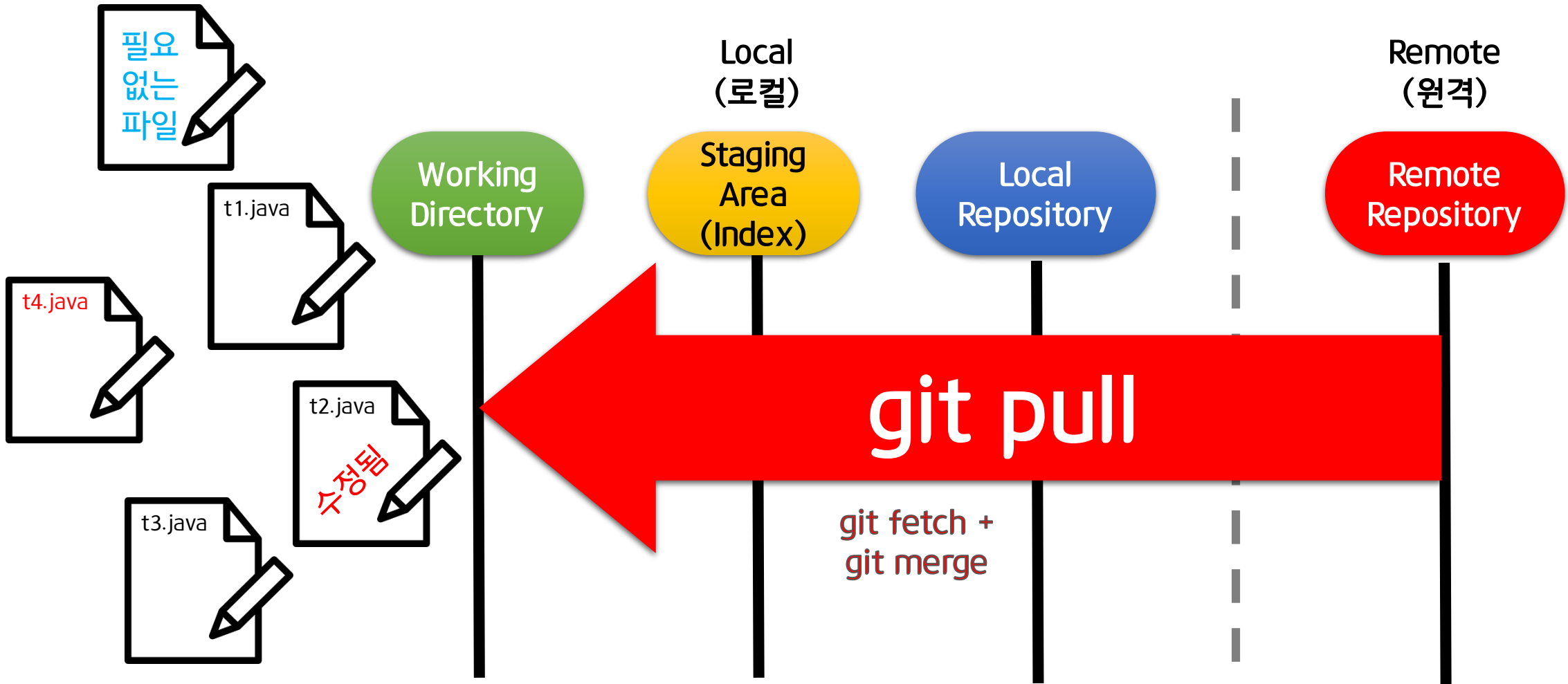
Git과 GitHub를 이용한 버전 관리 시 알고 있어야 할 각 공간의 명칭

- 작업 디렉토리(Working Directory)
 - 개발자의 컴퓨터 디스크 등
- 스테이징 영역 혹은 인덱스(Staging Area or Index)
 - 버전 관리 대상인, 즉 커밋 대상인 파일을 등록해 놓는 공간.
- 로컬 저장소(Local Repository)
 - 새로 생성된 파일과 그 파일의 변경 이력을 저장해 놓는 공간.
 - GitHub 등의 원격 서버의 분산된 공간.
- 원격 저장소(Remote Repository)
 - GitHub 등의 원격 서버 내 작업 공간.

GitHub를 원격 레퍼지토리로 갖는 Git 기반 분산 버전 관리 개요 ①



GitHub를 원격 레퍼지토리로 갖는 Git 기반 분산 버전 관리 개요 ③



버전 관리가 불필요한 파일은?

- 임시로 생성되는 백업 파일
- 컴파일 된 파일
- 입출력 데이터나 로그 파일
- IDE에서 사용하는 프로젝트 관리 파일
- 기타 사용자가 공유를 원치 않는 파일
- 임시로 생성한 파일



GitHub 실습 관련 URL

- 다음 사이트를 참고하세요.

- GitHub 계정 생성

- <https://recipes4dev.tistory.com/159>

- GitHub Repository(자기 저장소) 생성

- <https://greeksharifa.github.io/github/2018/06/29/github-usage-02-create-project/>

- GitHub Desktop 설치 및 GitHub와 연동

- [https://github.com/cau-cmclab/sku-cmclab.github.io/wiki/%EA%B9%83%ED%97%88%EB%B8%8C-%EB%8D%B0%EC%8A%A4%ED%81%AC%ED%83%91\(GitHub-Desktop\)-%EC%82%AC%EC%9A%A9](https://github.com/cau-cmclab/sku-cmclab.github.io/wiki/%EA%B9%83%ED%97%88%EB%B8%8C-%EB%8D%B0%EC%8A%A4%ED%81%AC%ED%83%91(GitHub-Desktop)-%EC%82%AC%EC%9A%A9)



오픈소스 개발이란?

- 쉽게 말해 “**거인의 어깨 위에 올라서라.**”
 - 거인: 오픈소스
 - 어깨 위에 올라선 우리: 오픈소스를 활용하여 개발하는 자.



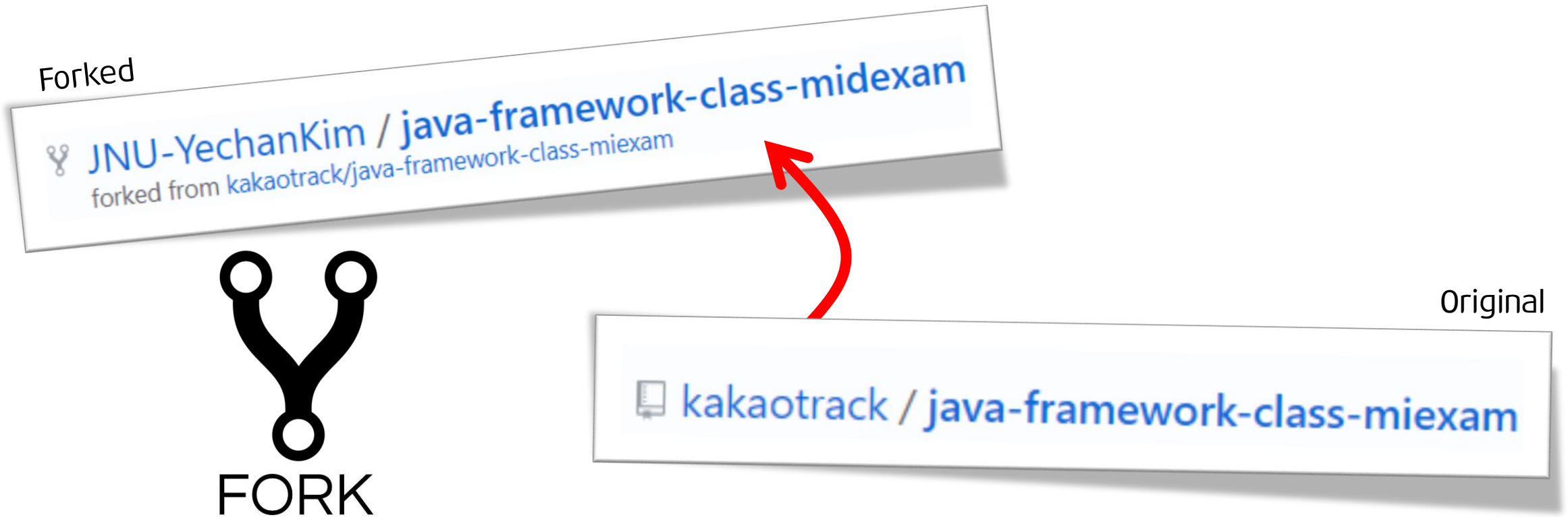
- 목적
 - 본질(Essential)에 집중한 개발 실현
 - 서로 공유(Share)하여 더 나은 SW의 발전 도모

GitHub에서 fork(clone의 확장)



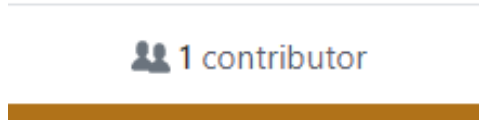
GitHub에서 fork(clone의 확장)

- fork: GitHub 커뮤니티 사이트에 공개된 다른 누군가의 Repository를 내 GitHub 공간으로 복제(clone)하는 행위.



GitHub에서 fork의 유익

- 오픈 소스의 기여자(Contributor)가 될 수 있다.
 - 원작자의 repo를 Fork한 뒤, 내가 더 좋게 개선한 소스를 원작자에게 pull해달라고 request할 수 있다.
 - ⇔ 원작자에게 자신이 개선한 소스를 반영해달라고 요청할 수 있다.
 - 원작자가 위 요청을 수락하면, 당신은 원작자의 repo에 대한 기여자가 된다.



- 저작권 의식 제고. (정보 문화 소양.)
 - 누군가의 소스를 그냥 다운로드 받지 않고, 인용하였음을 표시해 놓는 것은 당연한 일이다.



GitHub Desktop 설치

- 다음 사이트를 참고하세요.

- <https://m.blog.naver.com/PostView.nhn?blogId=yingbbang&logNo=221487342730&proxyReferer=https%3A%2F%2Fwww.google.com%2F>



Class Activity #1

- GitHub 계정 만들기
- GitHub에 내 레퍼지토리 생성하기
- GitHub.com에서 다른 개발자의 레퍼지토리 fork하기
- 내 로컬 환경에 fork한 레퍼지토리 내려 받기
- 다른 개발자의 레퍼지토리에 pull request하여 기여자 되기

Class Activity #2

- 레퍼지토리에서의 버전 롤백(version rollback) 연습하기
- git commit, git push 연습하기

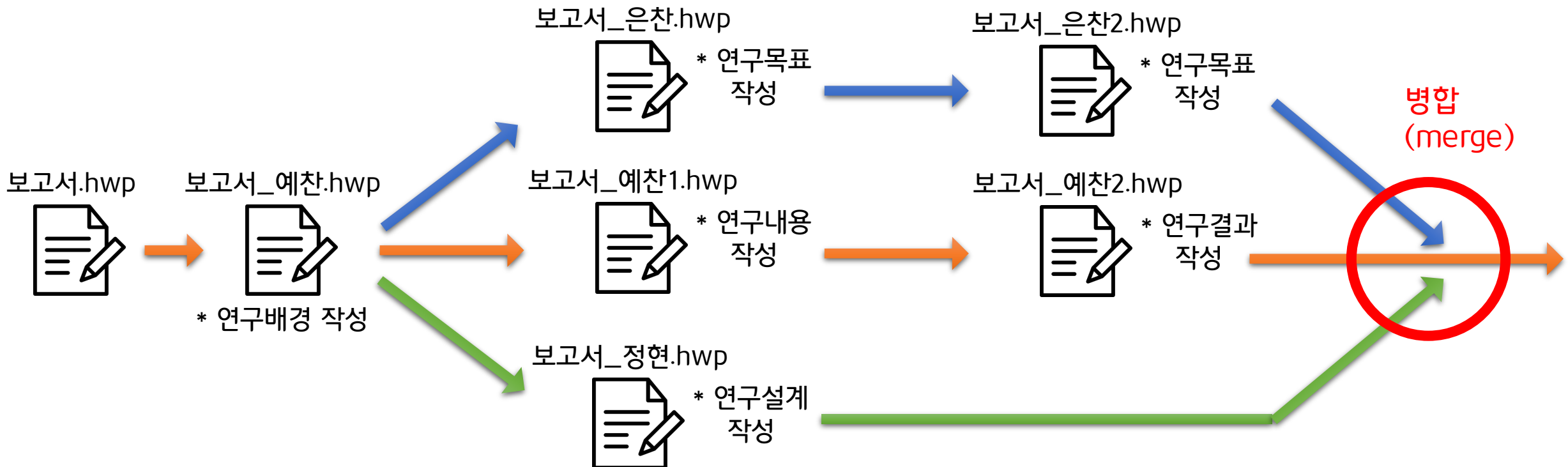


브랜치(Branch)



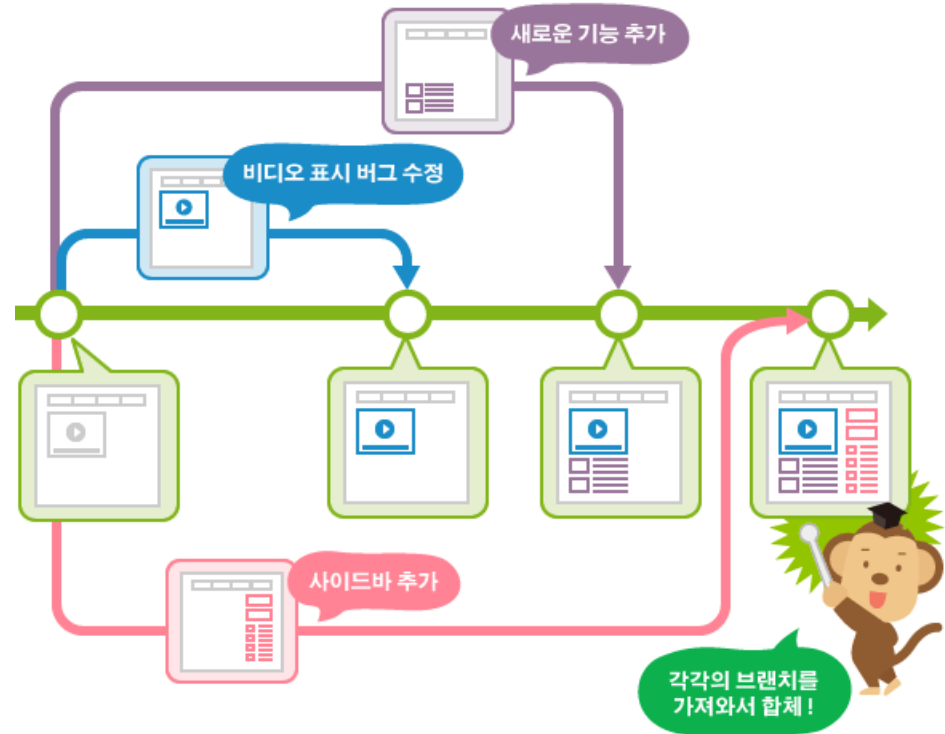
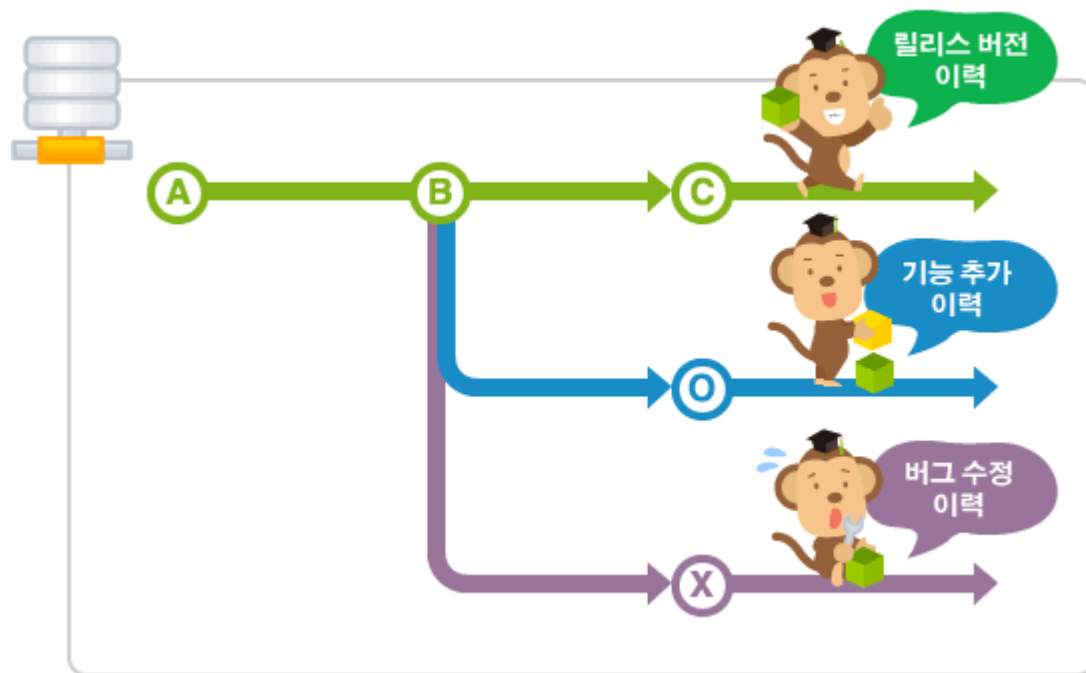
- 필요성

- 개발을 하다 보면 소스를 여러 개로 복사해야 하는 일이 자주 생긴다.



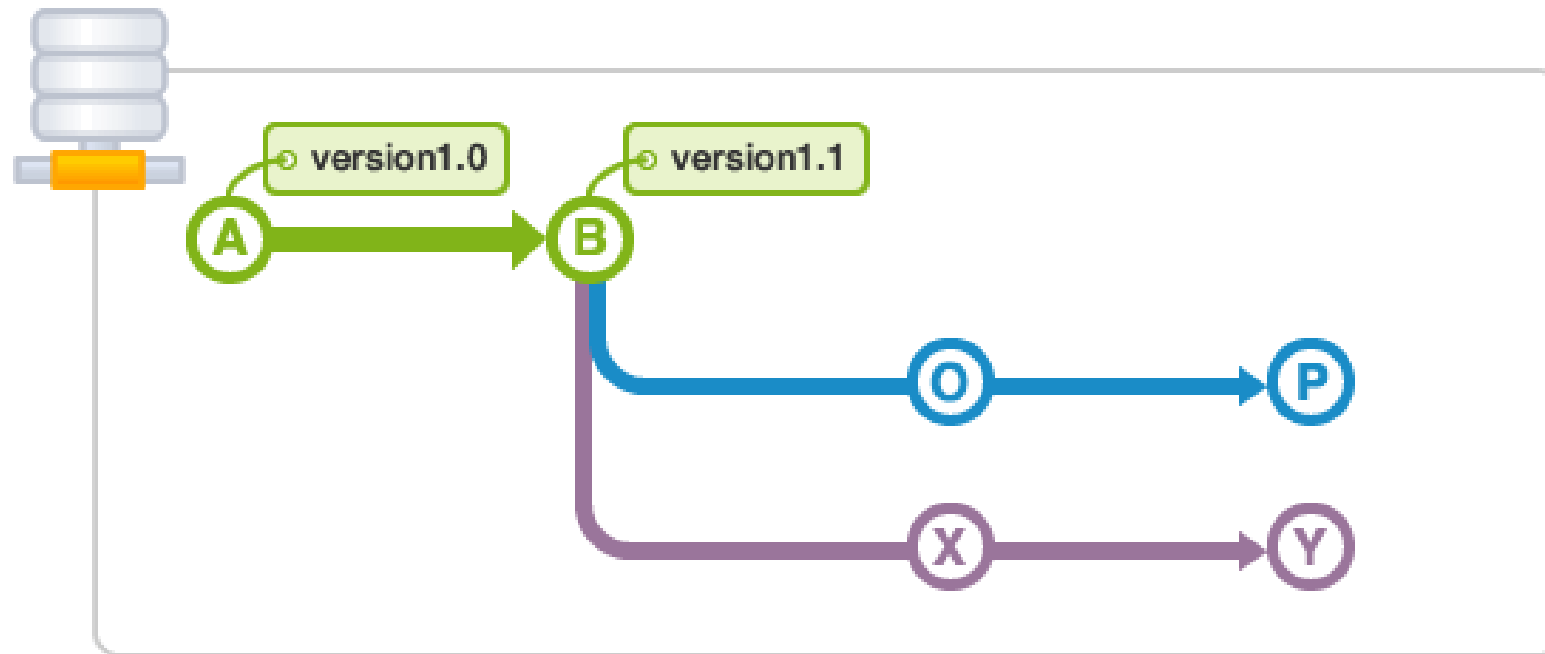
브랜치(Branch)

- 모든 버전 관리 시스템은 브랜치를 지원한다.
- Git에서도 브랜치를 지원하므로, 이 기능을 활용하는 방법을 숙지하면 협업에서 유리하다.



태그(Tag)

- 태그: 커밋을 참조하기 쉽도록 알기 쉬운 이름을 붙이는 행위 (aliasing)
 - 일반 태그: 이름만 붙이는 태그
 - 주석 태그: 이름 + 태그에 대한 설명 + 서명 + 태그를 만든 자에 대한 설명



버전 관리 관련하여 공부하기 좋은 자료

- **오픈소스개발방법론** (카카오 트랙 교과목)
 - 교과목 목표
 - 오픈소스가 무엇인지 안다.
 - 오픈소스 라이선스의 종류와 유의점을 열거할 수 있다.
 - 버전 관리(VC)의 유익을 설명할 수 있다.
 - GitHub가 제공하는 웹 호스팅 서비스를 이용할 수 있다.
 - GitHub를 이용한 버전 관리를 프로젝트에 접목할 수 있다.
 - 오픈소스 기반의 팀 프로젝트를 수행할 수 있다.
- **생활코딩** (이정훈 교수 수업 말고, egoing이 운영하는 온라인 사이트)
 - 버전 관리 관련 강좌 URL: <https://opentutorials.org/course/3838>

참고문헌

1. <https://medium.com/cardstack/introducing-gitchain-add61790226e>, 2019.11.25. 인용
2. 박찬정(2018), 소프트웨어공학 학부 강의자료 – 버전 관리
3. 정인상(2018), 소프트웨어공학 학부 강의자료 – 버전 관리
4. <https://backlog.com/git-tutorial/kr>, 2019.11.25. 인용
5. 최은만(2014), 새로 쓴 소프트웨어공학, 정익사