getting started with git

김덕홍 연구원

2015-03-20

insanehong@NAVER LABS

- Yobi 프로젝트
- KGUG(https://www.facebook.com/groups/kgugs/)

http://insanehong.kr

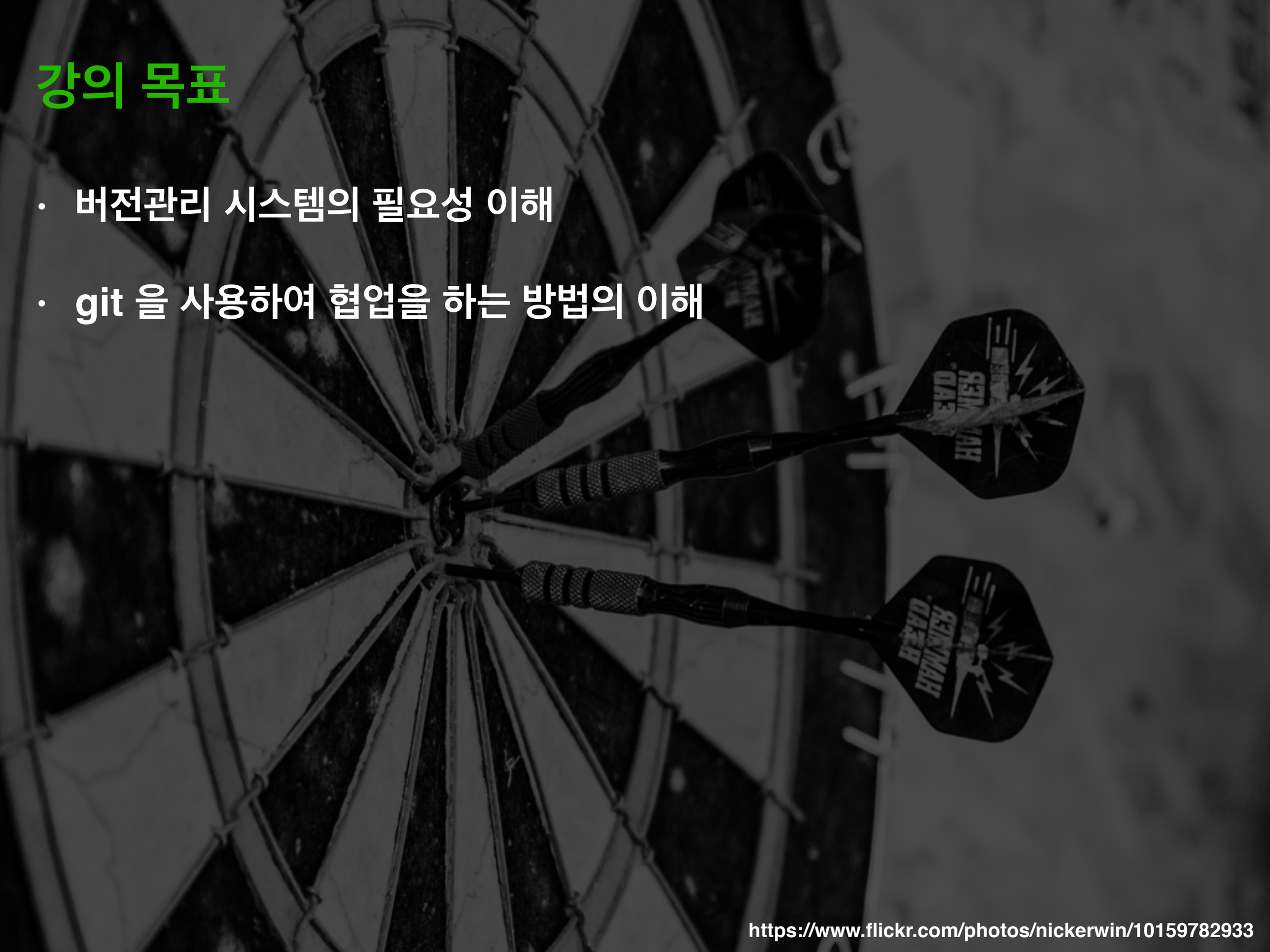
insane.hong@navercorp.com

@insanehong

insanehong

강의 목표

- 버전관리 시스템의 필요성 이해
- git 을 사용하여 협업을 하는 방법의 이해



https://www.flickr.com/photos/nickerwin/10159782933

강의 내용

Basic

- 버전관리 시스템이란?
- git 소개
- git 기초 명령어 - init, add, commit, log, show

Advanced

- git 고급 활용법 - branch, reset, merge, rebase

One more thing

- git & Github 을 활용한 협업 방법

교육에 앞서....

주의! 이 교육은 간접 광고를 포함하고 있지 않습니다.

http://www.dailycal.org/2013/11/21/everybody-macbook/

Recommend

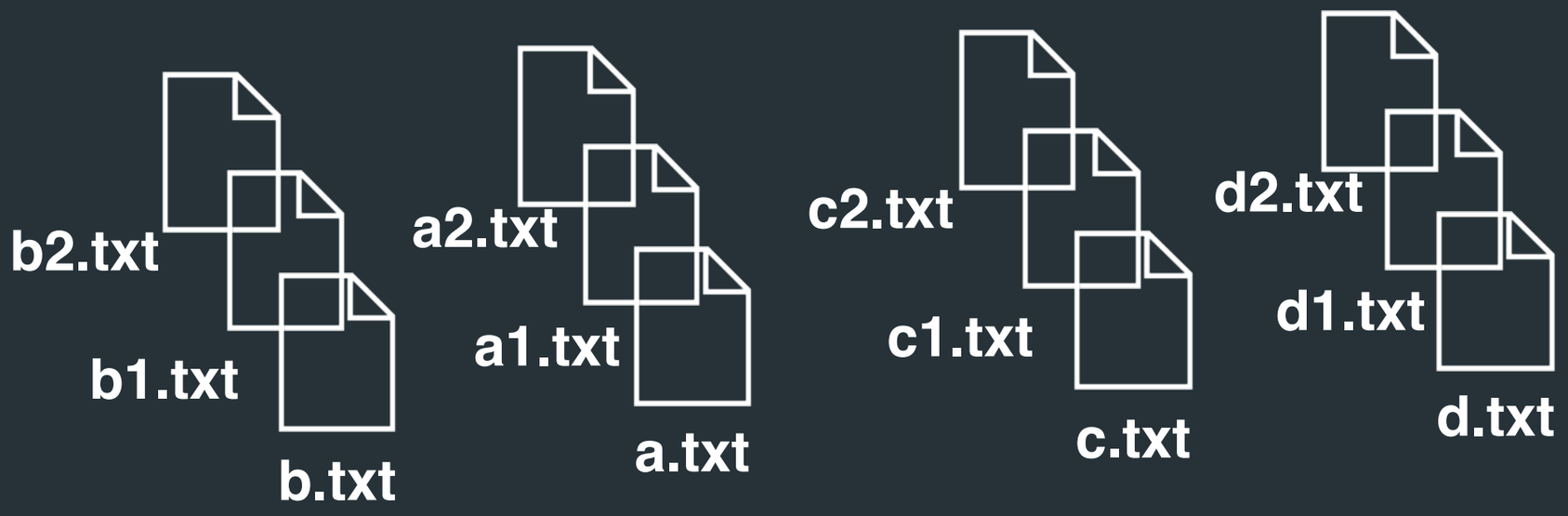
GUI & CLI

버전 관리 시스템 이란?

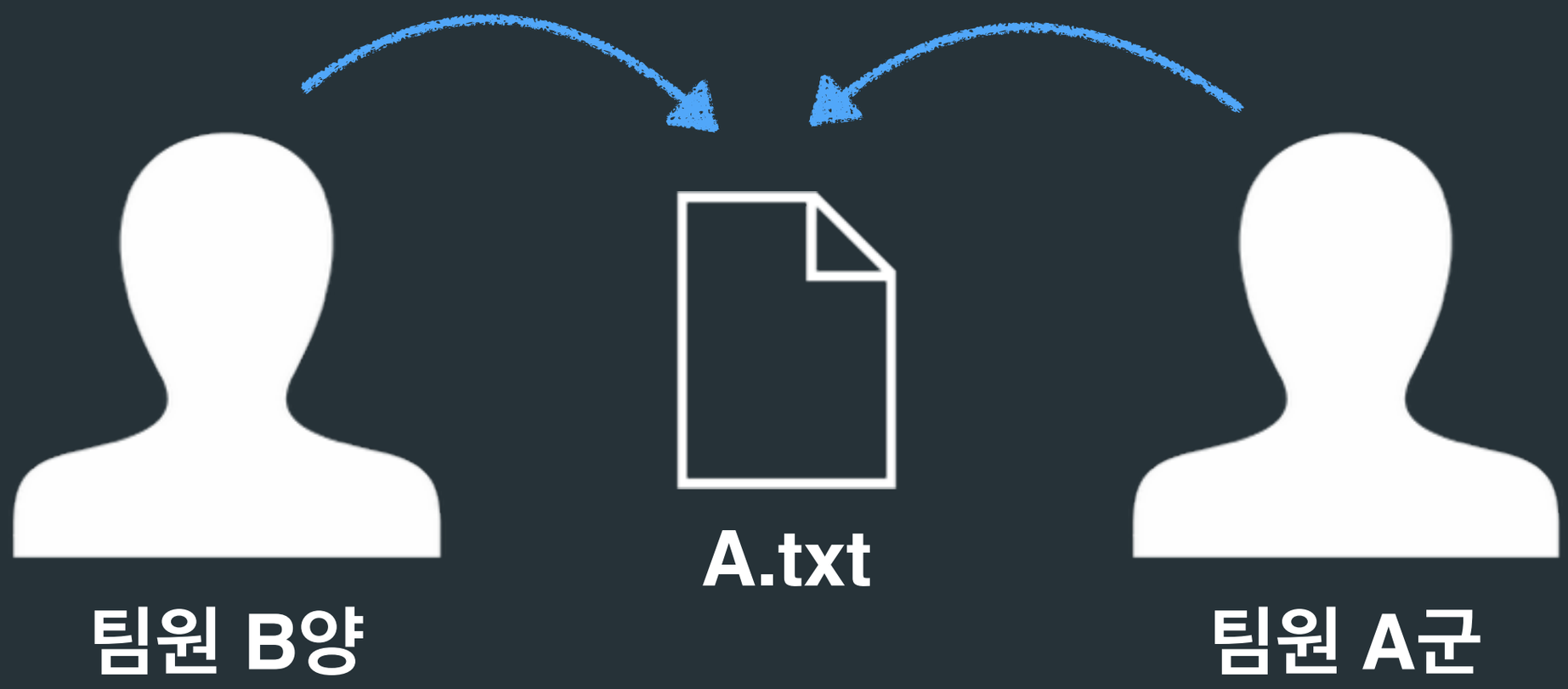
흔한 파일 버전 관리.avi

foo.최초원본.15.02.28.java　　foo.수정본.15.01.08.txt　　foo.수정본.15.02.28.txt

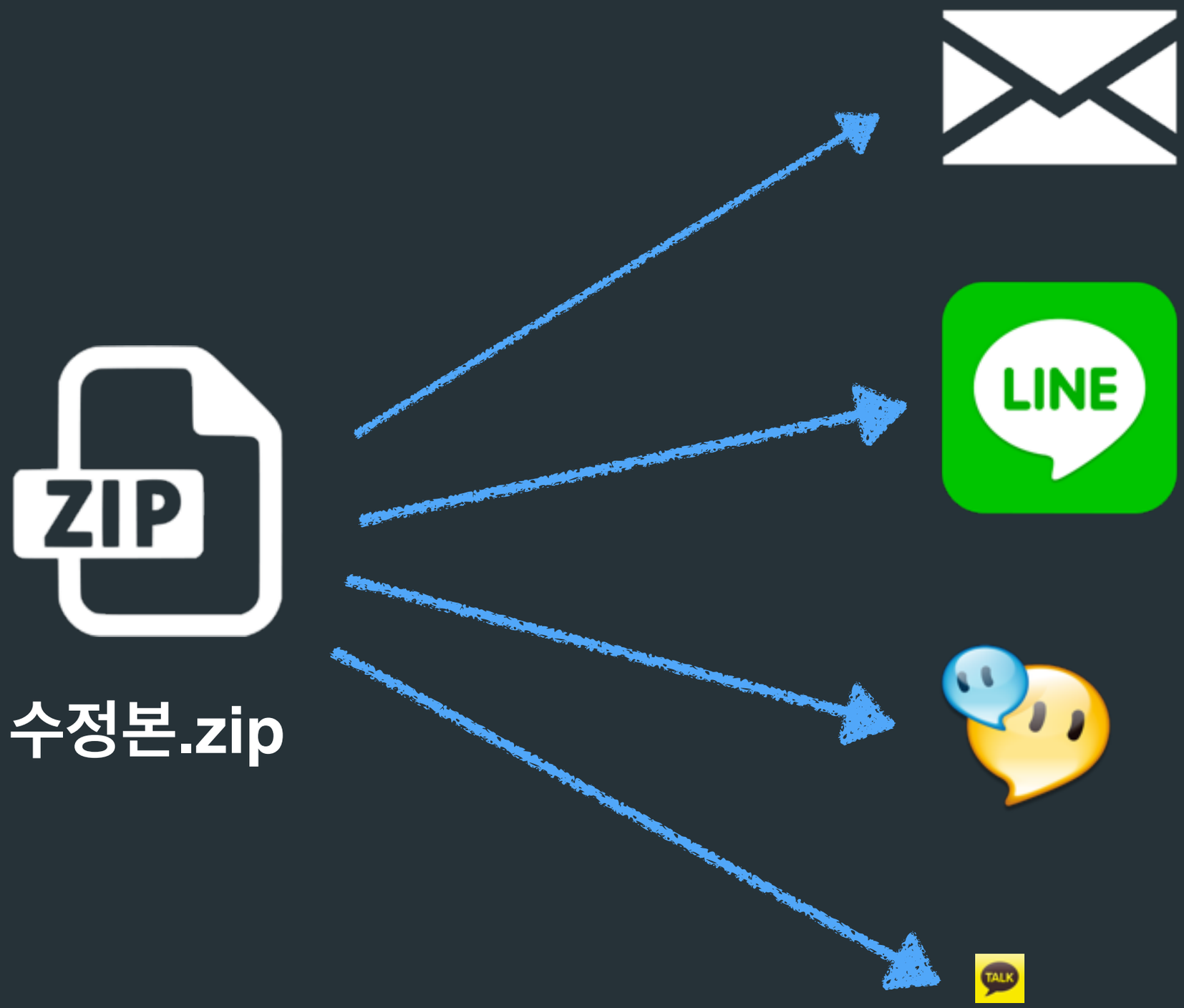
흔한 파일 버전 관리.avi



서로 다른 사람이 같은 파일을 수정 해야 하는 경우



흔한 파일 공유.avi



수정본.zip

그외에도 수많은 상황들

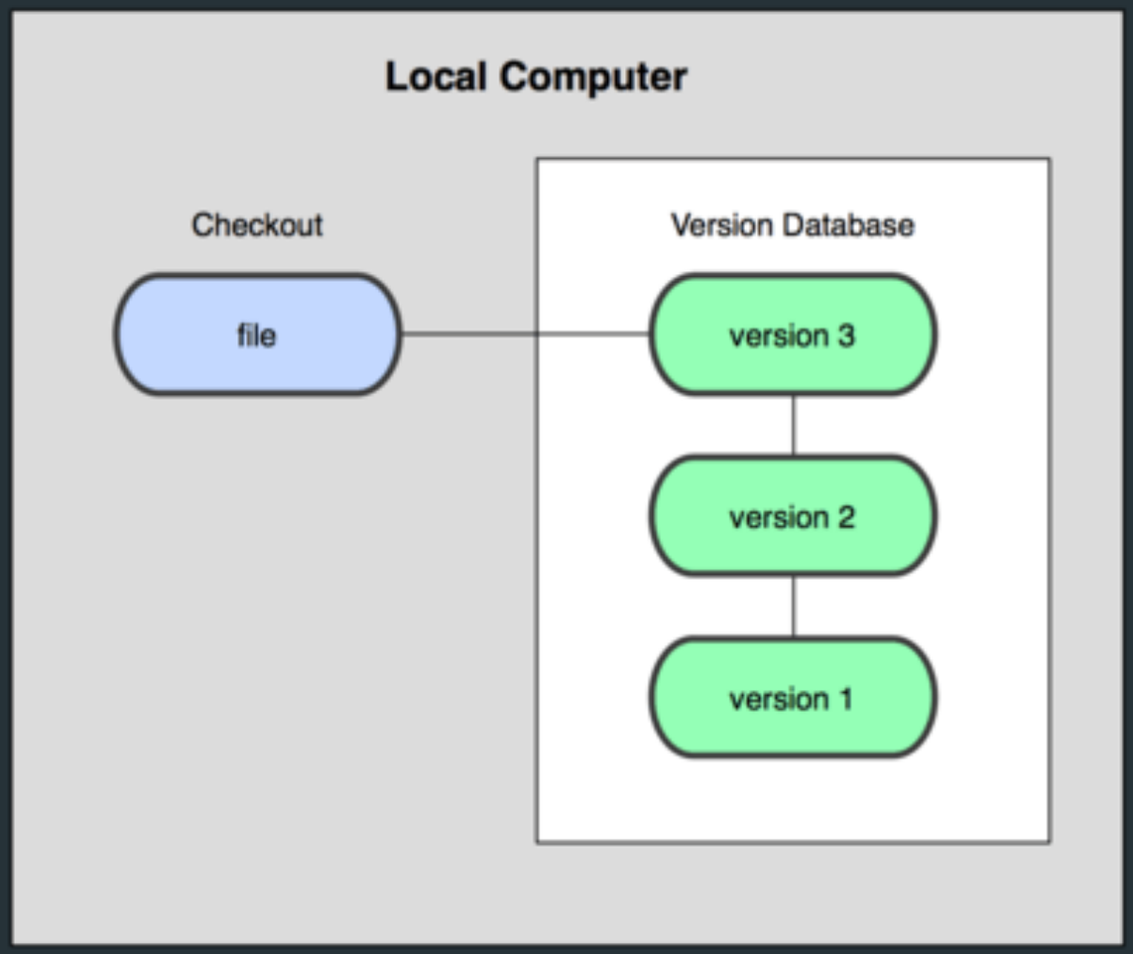
- 개별 파일 혹은 프로젝트 전체를 특정 시점으로 되돌려야 하는 상황
- 문제가 되는 부분을 누가, 언제, 왜 수정했는지 알아야 하는 경우
- 특정 파일을 잃어 버리거나 잘못되어 복구해야 하는 경우 등등

뭔가 좋은 방법이 없을까?

Version Control System

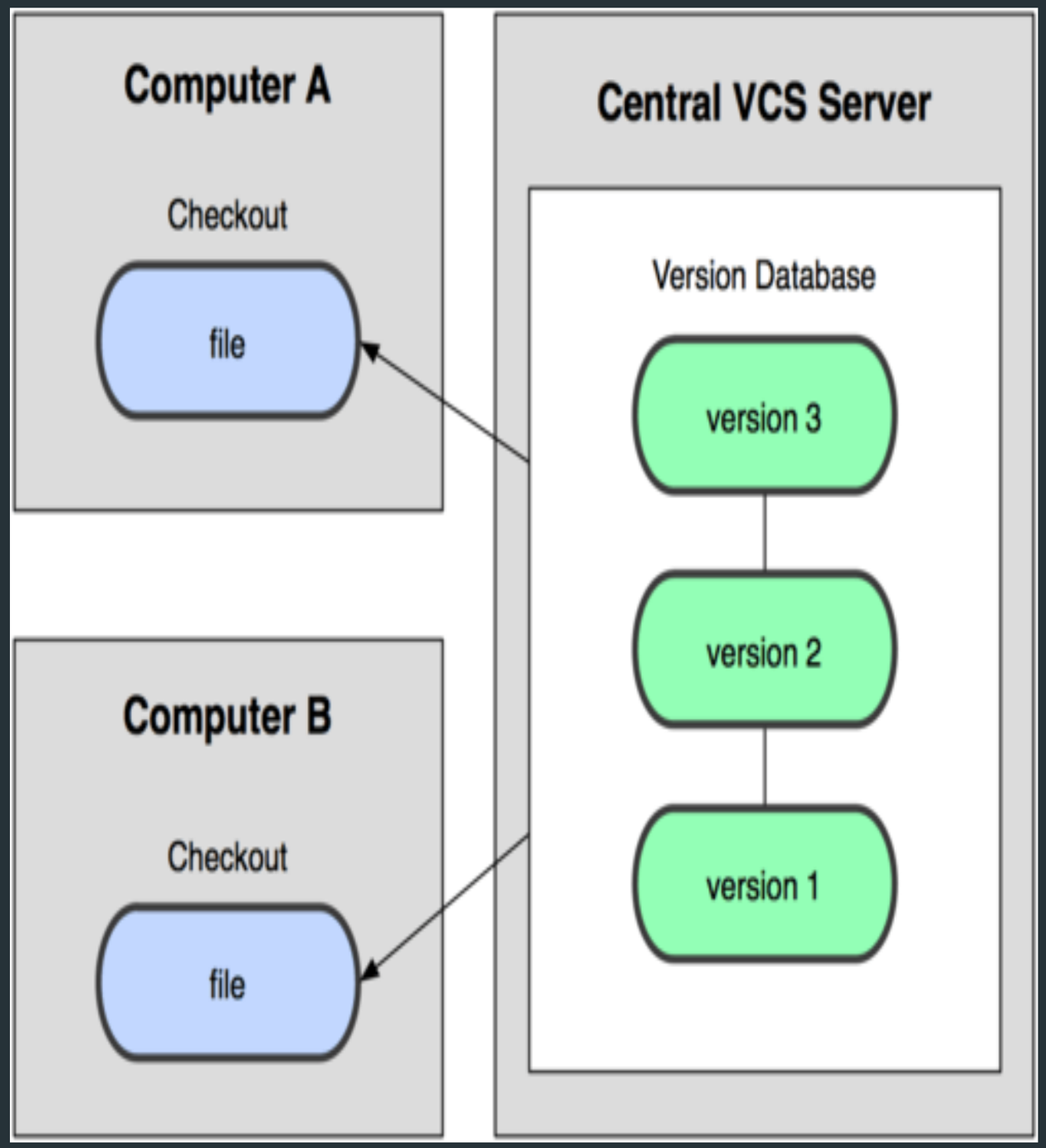
“특정 시점의 파일들의 상태를 저장하고,
필요에 따라 과거 특정 시점의
버전을 다시 불러올 수 있는 시스템 ”

로컬 버전 관리 시스템(LVCS)



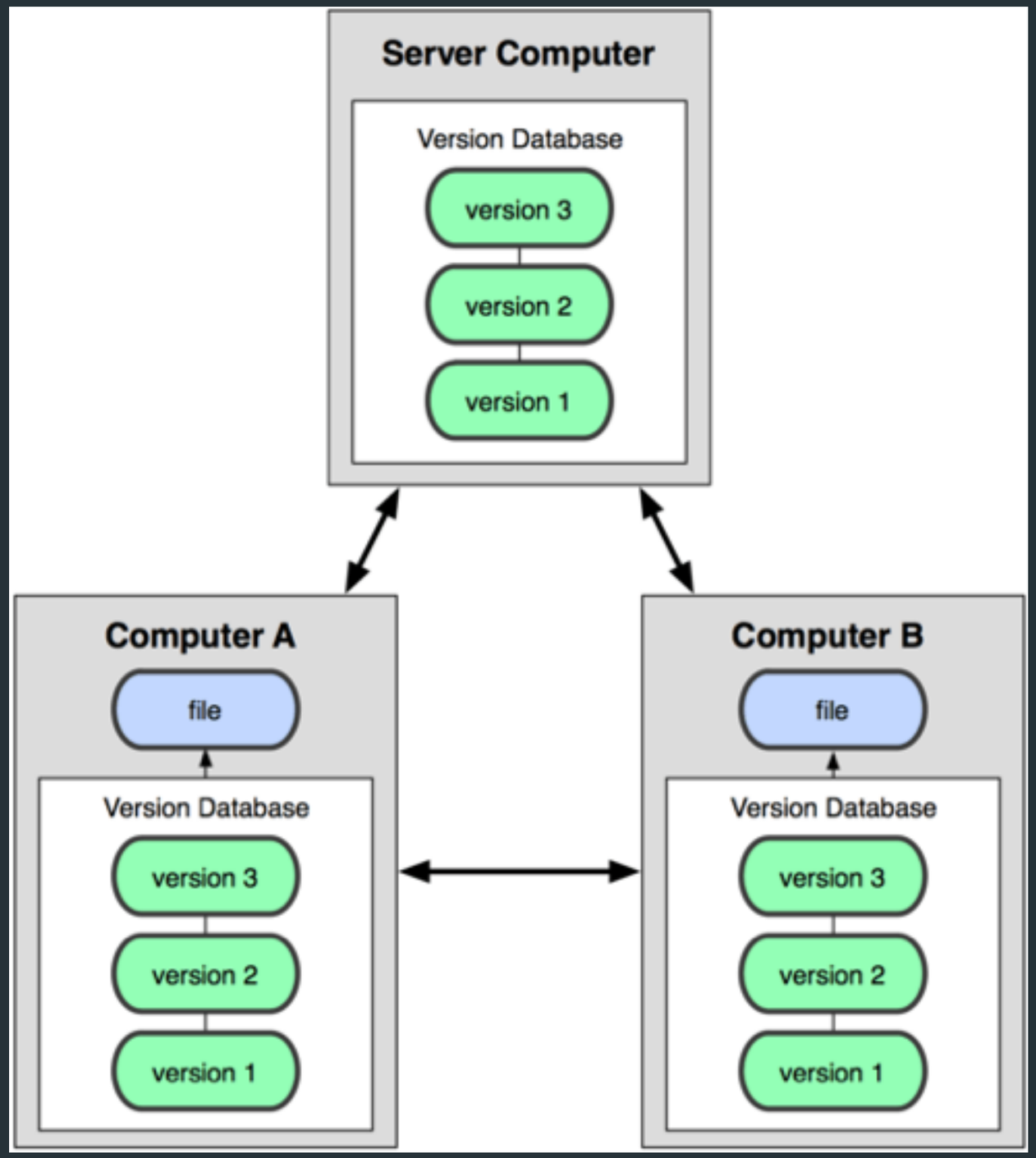
- 간단한 데이터베이스에 파일의 변경 사항을 기록
- 데이터의 차이점들을 특별한 형식의 파일에 저장
- RCS

중앙 집중식 버전 관리 시스템(CVCS)



- 버전 관리의 대표적인 방식
- 하나의 서버와 다수의 클라이언트
- CVS, Subversion, Perforce
- 서버가 다운될 경우 버전관리 불가능
- 서버문제로 스냅샷을 제외한 이력의 손실 가능

분산형 버전 관리 시스템(DVCS)



- CVCS 의 문제를 해결하기 위해 고안
- Git, Mecurial, Bazaar, Darcs 등
- 다수의 원격 저장소
- 다양한 workflow 사용가능

현재 가장 많이 사용되는 VCS

SVN
(Subversion)

git 기초

Git 탄생 배경

누가

Linus Torvalds

언제

2005년

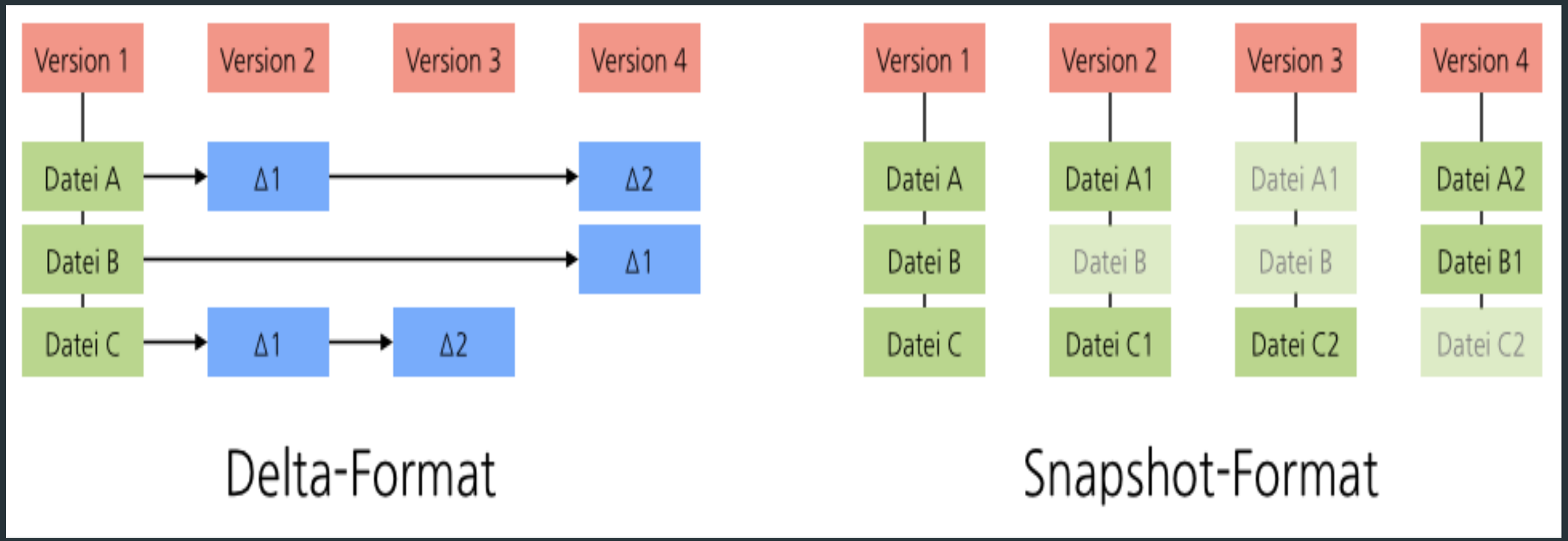
왜

리눅스 커널 소스코드를 관리하던 BitKeeper 가 상용으로
전환되어 버전관리시스템을 직접 개발

Git 개발시 고려한 사항들

- 빠른 속도
- 단순한 구조
- 비선형적인 개발(수천 개의 동시 다발적인 브랜치)
- 완벽한 분산
- 리눅스 커널 같은 대형 프로젝트에도 유용할 것(속도나 데이터 크기 면에서)

Git 은 왜 빠를까?



파일이 달라지지 않았으면 Git은 성능을 위해서 파일을 저장하지 않고
이전 상태의 파일에 대한 링크만 저장한다.

Git의 무결성 관리

- Git은 파일을 이름으로 저장하지 않고 해당 파일의 해시로 저장
- SHA-1 해시를 사용한 체크섬(40자 길이의 16진수 문자열)
  24b9da6552252987aa493b52f8696cd6d3b00373

Git이 관리중인 파일의 3가지 상태

committed

데이터가 로컬 데이터베이스에 안전하게 저장된 상태

Modified

수정한 파일을 아직 로컬 데이터베이스에 저장하지 않은 상태

Staged

로컬 데이터베이스에 저장하기 위해 등록해 놓은 상태

Tracked

Untracked

Staged

Committed

Modified

file

git add

git commit

edit file

git add

Git의 파일관리를 위한 3가지 영역

Git Directory

프로젝트의 메타데이터와 객체 데이터베이스를 저장하는 곳 (.git)

Working Directory

프로젝트의 특정 버전을 Checkout 한 영역

Staging Area(index)

저장할 파일들에 대한 정보를 담고있는 파일 (.git/index)

Git의 파일 관리 3단계

- 워킹 디렉토리에서 파일 추가, 수정, 삭제
- Staging Area에 파일을 Stage해서 커밋할 스냅샷을 만듬
- Staging Area에 있는 파일들을 커밋해서 Git 디렉토리에 영구적인 스냅샷으로 저장

git 설정

Git 설정 파일의 종류

for all users

/etc/gitconfig $ git config --system

for current user

~/.gitconfig $ git config --global

for current repository

~/.git/config $ git config --local

git 설치후 가장 먼저 해주어야 할 설정

user.name

커밋시 author, committer 의 메타데이터로 사용 됨

user.email

커밋시 author, committer 의 메타데이터로 사용 됨

git config 설정 실습

```
~ $ git config --global user.name "YOUR NAME"

~ $ git config --global user.email "YOUR EMAIL"

~ $ git config --list #설정 확인

~ $ git config user.name #지정된 key의  설정만 확인
```

저장소 만들기

git init command

기존 프로젝트를 Git으로 관리하고 싶을 때

```
$ git init
```

새로운 디렉토리를 생성하여 git 저장소를 만들때

```
$ git init (directory path)
```

git 저장소 만들기 실습

```
~ $ cd ~ && mkdir git-workspace && cd ~/git-workspace

~ $ git init git-repo
Initialized empty Git repository /workspace/git-repo/.git/

~ $ cd git-repo

~/git-repo $ ls -al
total 0
drwxr-xr-x   3 insanehong  staff  102  2 26 03:56 .
drwxr-xr-x  21 insanehong  staff  714  2 26 00:32 ..
drwxr-xr-x   9 insanehong  staff  306  2 26 03:56 .git
```

git directory 구조

~/git-repo $ **tree .git**

```
.git
├── HEAD
├── config
├── description
├── hooks
│   ├── applypatch-msg.sample
│   ├── commit-msg.sample
│   ├── post-update.sample
│   ├── pre-applypatch.sample
│   ├── pre-commit.sample
│   ├── pre-push.sample
│   ├── pre-rebase.sample
│   ├── prepare-commit-msg.sample
│   └── update.sample
├── info
│   └── exclude
├── objects
│   ├── info
│   └── pack
└── refs
    ├── heads
    └── tags
```

Working Directory
파일 상태 확인 하기

git status command

working directory의 자세한 내용을 확인 할때

```
$ git status
```

파일들의 변경 상태만 확인 할때

```
$ git status -s
```

현재 Working Directory 상태 확인

```
~/git-repo $ git status
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
```

working directory 상태 변경 실습

```
~/git-repo $ echo "hello world" >> README.md

~/git-repo $ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        README.md

nothing added to commit but untracked files present (use "git add" to track)


~/git-repo $ git status -s
?? README.md
```

Staged 로
파일 상태 변경 하기

git add command

index 에 추가할 파일을 지정 할때

```
$ git add (file path) (file path) (file path)
```

수정된 모든 파일을 추가 할경우

```
$ git add . # 실수를 줄이기 위해 파일 지정 방법을 추천
$ git add -A
```

새로 추가된 파일을 staged 상태로 만들기 실습

```
~/git-repo $ git add README.md

~/git-repo $ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   README.md
```

Committed 로 파일 상태 변경 하기

git commit command

간단한 메세지를 남기며 커밋 할때

```
$ git commit -m "simple message"
```

보다 자세히 메세지를 남기며 커밋할 때 (multi line)

```
$ git commit
```

git commit 실습

```
~/git-repo $ git commit -m "add README.md file"
[master (root-commit) e0d5a92] add README.md file
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

~/git-repo $ git status
On branch master
nothing to commit, working directory clean
```

Git commit 다루기 실습

- LICENSE.md 파일을 추가 하고 커밋하기
- 내용은 http://choosealicense.com/ 에서 MIT License 사용
- 커밋 메세지는 다음과 같이 상세히 남기기

  add LICENSE.md file

  - under the MIT License
  - reference : http://choosealicense.com/

- git 내부에서 어떤 동작이 일어나는지 생각해보면서 실습 해보세요

Git commit 다루기 실습 hint

```
~/git-repo $  touch LICENSE.md

~/git-repo $  git status

~/git-repo $  vi LICENSE.md #내용  저장 닫기

~/git-repo $  git status

~/git-repo $  git add LICENSE.md

~/git-repo $  git status

~/git-repo $  git commit #내용  저장 닫기
```

파일 변경 내용 확인

git diff command

working directory 와 index 를 비교

```
$ git diff
```

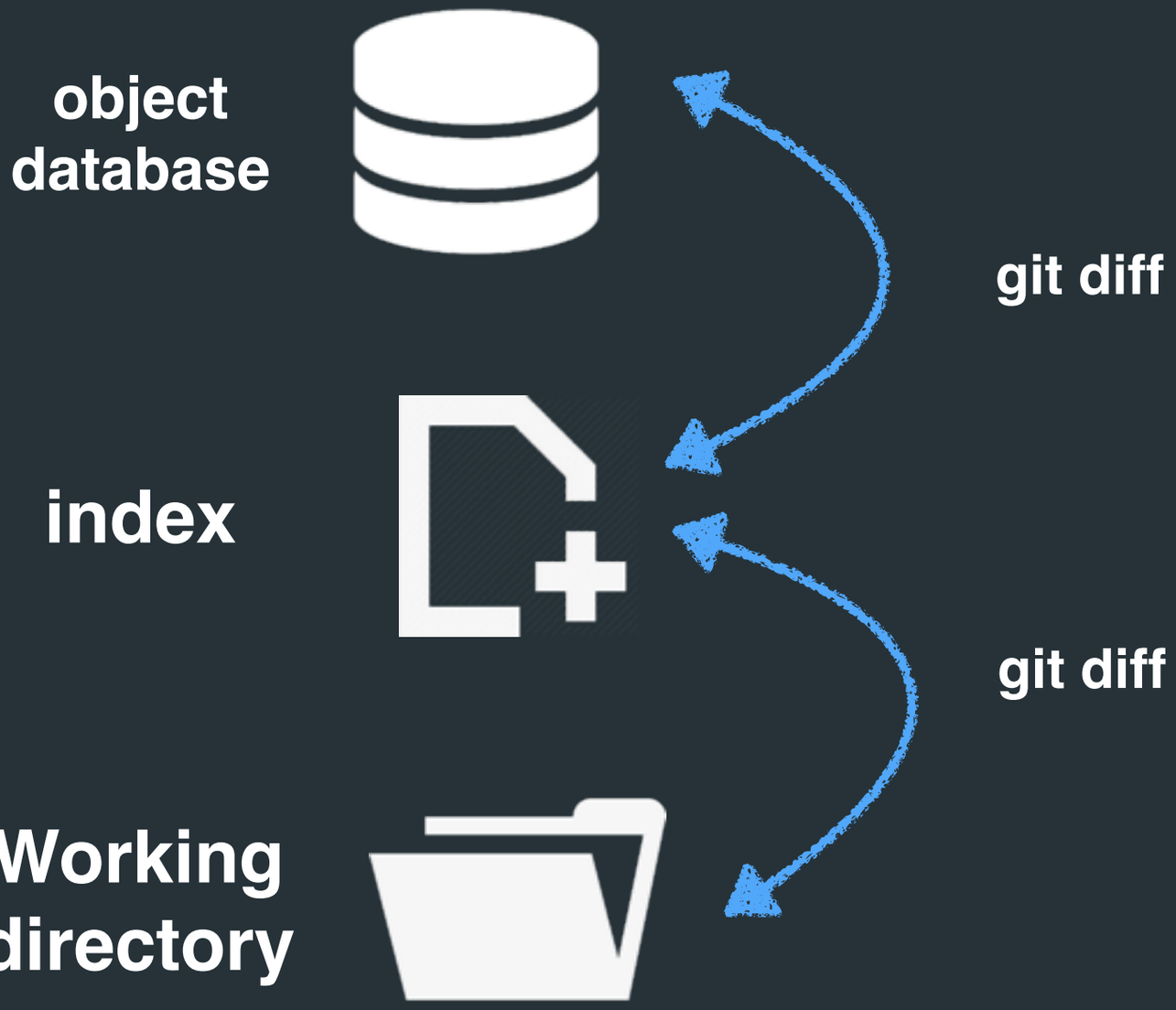
커밋 간 차이점 을 비교

```
$ git diff (commitA)..(commitB)
```

index 와 HEAD를 비교

```
$ git diff --staged
```

git diff command



LICENSE.md 파일 수정하기

LICENSE.md

1. The MIT License (MIT)
2.
3. Copyright (c) [year] [fullname]
4.
5. Permission is hereby granted, free of charge, …

1. The MIT License (MIT)
2.
3. Copyright (c) 2015 insanehong
4.
5. Permission is hereby granted, free of charge, …

git diff 실습

~/git-repo $ **vi LICENSE.md** #파일 수정후 저장 닫기

~/git-repo $ **git diff**

```
diff --git a/LICENSE.md b/LICENSE.md
index 38c6bf2..90ef75a 100644
--- a/LICENSE.md
+++ b/LICENSE.md
@@ -1,6 +1,6 @@
 The MIT License (MIT)

-Copyright (c) [year] [fullname]
+Copyright (c) 2015 insanehong

 Permission is hereby granted, free of charge, to any person obtaining a copy
 of this software and associated documentation files (the "Software"), to deal
```

git diff 실습 - staged diff

```
~/git-repo $ git add LICENSE.md
~/git-repo $ git diff
~/git-repo $ git diff --staged
diff --git a/LICENSE.md b/LICENSE.md
index 38c6bf2..90ef75a 100644
--- a/LICENSE.md
+++ b/LICENSE.md
@@ -1,6 +1,6 @@
 The MIT License (MIT)
-Copyright (c) [year] [fullname]
+Copyright (c) 2015 insanehong

 Permission is hereby granted, free of charge, to any person obtaining a copy
 of this software and associated documentation files (the "Software"), to deal

~/git-repo $ git commit -m "change to year, name on license"
```

커밋 조회

git log command

현재 저장소의 시간순 커밋 히스토리

```
$ git log
```

diff 를 함께 보기

```
$ git log -p [-no]
```

커밋 메세지 조회

```
$ git log --grep <검색어>
```

git log command options

mulilne 으로 작성된 메세지 한줄로 표시하기

```
$ git log --oneline
```

그래프 표현

```
$ git log --graph
```

형식 지정

```
$ git log --pretty=format:"%h %s" --graph
```

git log 실습

~/git-repo $ **git log**

```
commit 1a8236f284791342fb8340f9e38531fa471e2823
Author: insanehong <insanehong@gmail.com>
Date:   Thu Feb 26 12:56:28 2015 +0900

    change to year, name on license

commit 23a1ef80f09efc68149d3723b1793e89ffefc6b1
Author: insanehong <insanehong@gmail.com>
Date:   Thu Feb 26 09:17:13 2015 +0900

    add LICENSE File

    - License under the MIT License
    - reference : http://choosealicense.com/

commit e0d5a92d33f528147e62a87cd5589e9b4d78bb54
Author: insanehong <insanehong@gmail.com>
Date:   Thu Feb 26 04:36:05 2015 +0900

    add README.md file
```

git log 실습 - 최근 2개의 커밋로그를 diff 와 함께 보기

~/git-repo $ **git log -p -2**

```
commit 1a8236f284791342fb8340f9e38531fa471e2823
Author: insanehong <insanehong@gmail.com>
Date:   Thu Feb 26 12:56:28 2015 +0900

    change to year, name on license

diff --git a/LICENSE.md b/LICENSE.md
index 38c6bf2..90ef75a 100644
--- a/LICENSE.md
+++ b/LICENSE.md
@@ -1,6 +1,6 @@
 The MIT License (MIT)

-Copyright (c) [year] [fullname]
+Copyright (c) 2015 insanehong

 Permission is hereby granted, free of charge, to any person obtaining a copy
 of this software and associated documentation files (the "Software"), to deal

commit 23a1ef80f09efc68149d3723b1793e89ffefc6b1
Author: insanehong <insanehong@gmail.com>
Date:   Thu Feb 26 09:17:13 2015 +0900

    add LICENSE File
```

git show command

특정 커밋 조회하기

```
$ git show (commit)
```

이전 단계 커밋 조회하기

```
$ git show HEAD@{no}
```

어제 이루어진 커밋 조회하기

```
$ git show HEAD@{yesterday}
```

git show 실습

~/git-repo $ **git show HEAD**

```
commit 1a8236f284791342fb8340f9e38531fa471e2823
Author: insanehong <insanehong@gmail.com>
Date:   Thu Feb 26 12:56:28 2015 +0900

    change to year, name on license

diff --git a/LICENSE.md b/LICENSE.md
index 38c6bf2..90ef75a 100644
--- a/LICENSE.md
+++ b/LICENSE.md
@@ -1,6 +1,6 @@
 The MIT License (MIT)

-Copyright (c) [year] [fullname]
+Copyright (c) 2015 insanehong

 Permission is hereby granted, free of charge, to any person obtaining a copy
 of this software and associated documentation files (the "Software"), to deal
```

alias

매번 입력하기 귀찮은 옵션들

```
~/git-repo $ git log  --all  --oneline  --decorate --graph

* 1a8236f (HEAD, master) change to year, name on license
* 23a1ef8 add LICENSE File
* e0d5a92 add README.md file
```

git config alias 등록하기

.gitconfig

```
1. [user]
2.         name = insanehong
3.         email = insanehong@gmail.com
4.
5. [alias]
6.         lg = log  --all  --oneline  --decorate --graph
```

git config alias 등록하기 실습

```
~/git-repo $ vi ~/.gitconfig #alias 등록후 저장 닫기

~/git-repo $ git lg
* 1a8236f (HEAD, master) change to year, name on license
* 23a1ef8 add LICENSE File
* e0d5a92 add README.md file
```

Today Tips

Git commit log 를 어떻게 남겨야 할까?

- 커밋은 최대한 의미단위로 묶어서
- 커밋 메세지는 커밋이 담고있는 수정 내용을 충분히 반영 할수 있도록

bad

Updated config file

good

Updated config file with new Minimee cache path directory.

This allows our cache path to be dynamicly absolute based on the environment out code is in.

$config[‘minimee_cache_path’] = $base_path . ‘/min’;35 ERIKREAGAN • EECI

Git 을 사용하기에 괜찮은 GUI Tool 은 없을까?

- CLI 에 익숙하지 않은 사람들 혹은 light 사용자에게는 GUI 가 더 좋은 대안이 될수도 있음.
- git 의 강력하고 다양한 command 들을 활용하기에는 GUI 로는 역부족
- 혹시 GUI 를 고려하고 있다면 SourceTree 를 써볼것을 추천

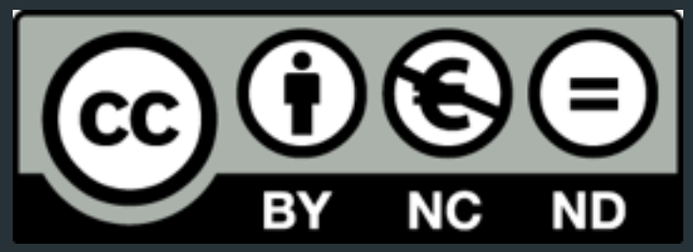
root commit 은 이후에 수정 할수 없다?!

- commit 을 사용자가 자유롭게 다룰수 있다는 것은 git 의 가장 큰 장점 중 한가지
- empty commit

  $ git commit --allow-empty

- .gitignore file 을 이용

Q & A



twitter : @insanehong
email : insane.hong@navercorp.com