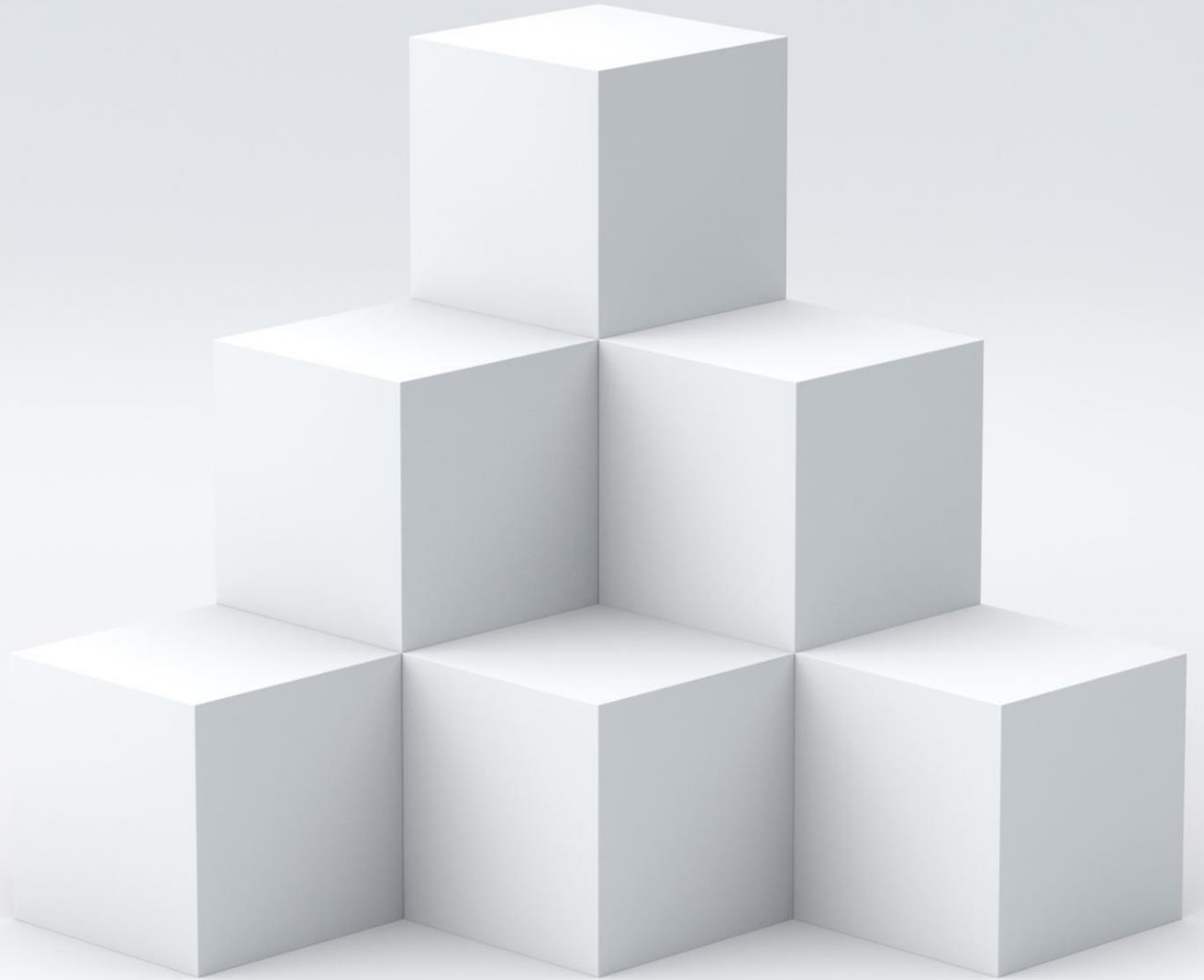


Software Engineering

- git -

Professor Han-gyoo Kim

2022



Git



git-scm.com (git book <https://git-scm.com/book/en/v2>)

- fast version control system
- open source (mature)
- github은 매우 유용한 서비스이지만 git의 일부는 아님
- github은 remote repository (git server) 중 하나
- 2005 Linux
- 분산 VCS (but all version control activities are local, no connection required) – 팀 공유 작업에 최적화된 도구
- 다수 개발자 사이에 협력을 용이하게 하는 도구
- 가장 기본적이고 필수적인 DevOps 도구
- 누가 언제 무엇을 수정하였는지 tracking – content (not file) oriented
- 수정된 내용을 되돌릴 수 있음
- Local / Remote Repository

Official Pro Git book (2nd ed. 2014)

[←](#) [→](#) [↻](#) [git-scm.com/book/ko/v2](#)

About

Documentation

- Reference
- Book**
- Videos
- External Links

Downloads

Community

This book is available in [English](#).
Full translation available in

- [azərbaycan dili,](#)
- [български език,](#)
- [Deutsch,](#)
- [Español,](#)
- [Français,](#)
- [Ελληνικά,](#)
- [日本語,](#)
- [한국어,](#)

Book

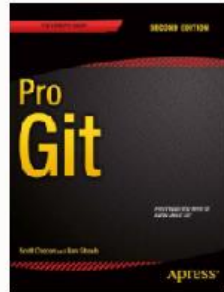
The entire Pro Git book, written by Scott Chacon and Ben Straub and published by Apress, is available here. All content is licensed under the [Creative Commons Attribution Non Commercial Share Alike 3.0 license](#). Print versions of the book are available on [Amazon.com](#).

1. 시작하기

- 1.1 버전 관리란?
- 1.2 짧게 보는 Git이 역사
- 1.3 Git 기초
- 1.4 CLI
- 1.5 Git 설치
- 1.6 Git 최초 설정
- 1.7 도움말 보기
- 1.8 요약



2. Git의 기초

- 2.1 Git 저장소 만들기
- 2.2 수정하고 저장소에 저장하기

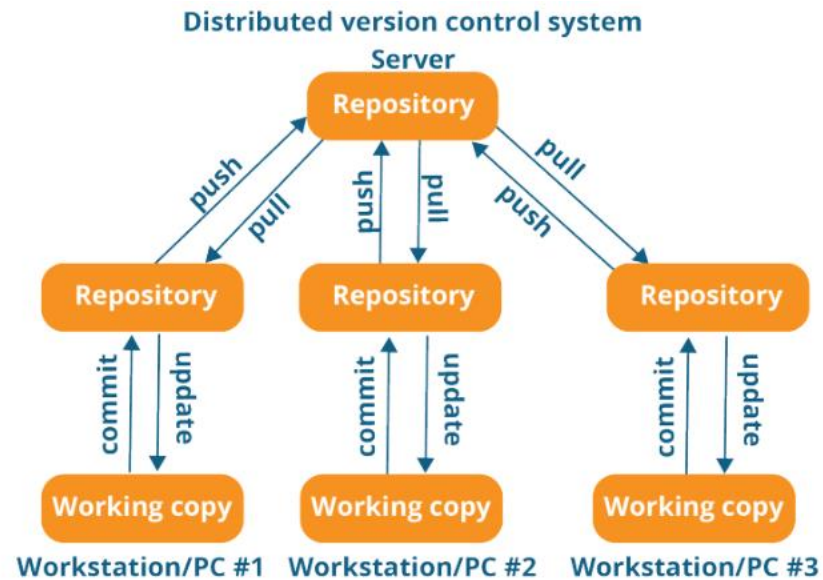
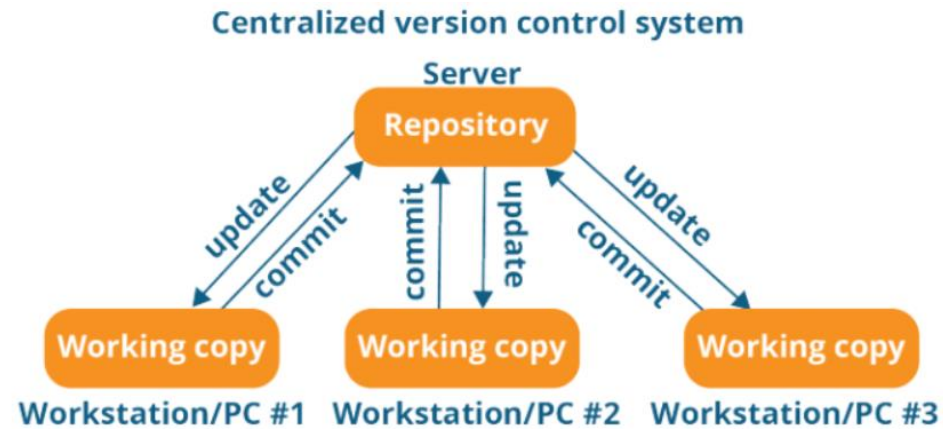


2nd Edition (2014)

Download Ebook

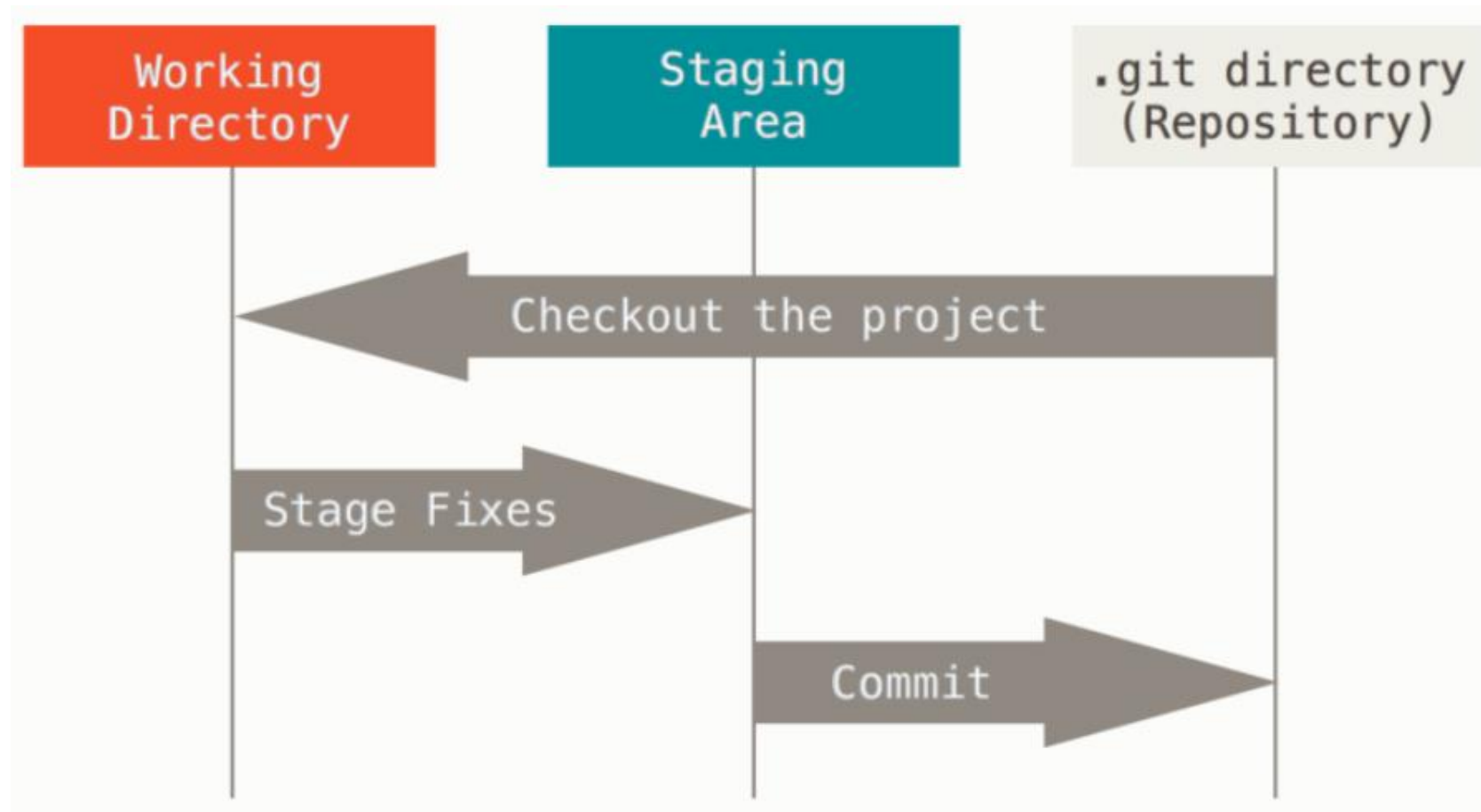


Git distributed architecture



- 빠름
- Internet 없어도 작업
- Main repository에 영향을 미치기 전에 개발자간 상의
- Crash safe

Git structure



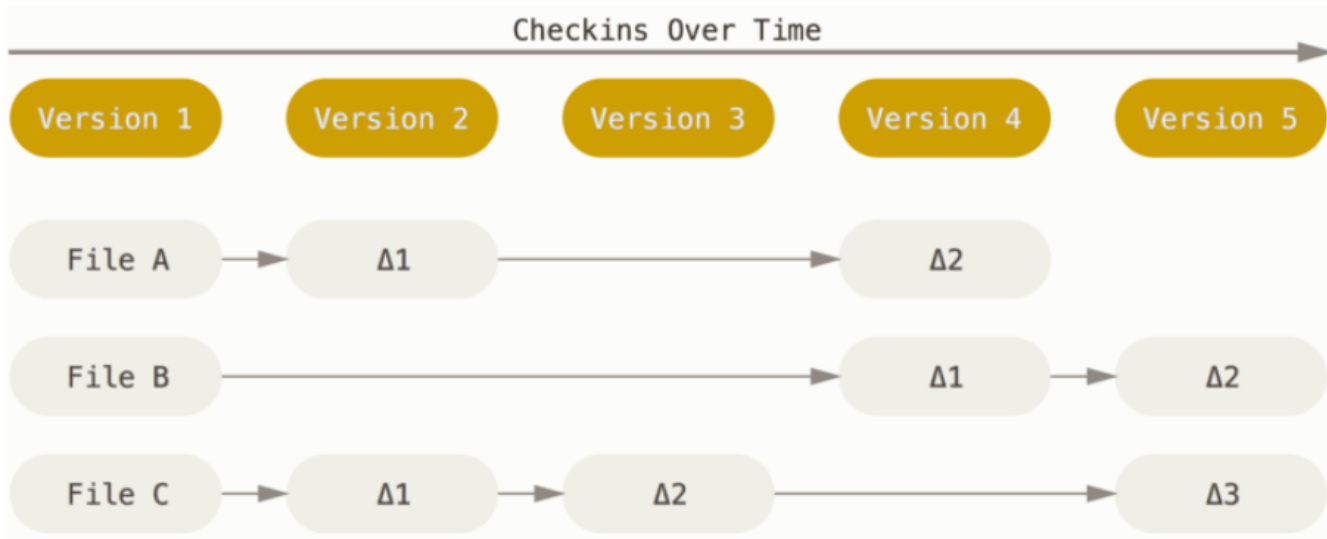
File view

modified

staged

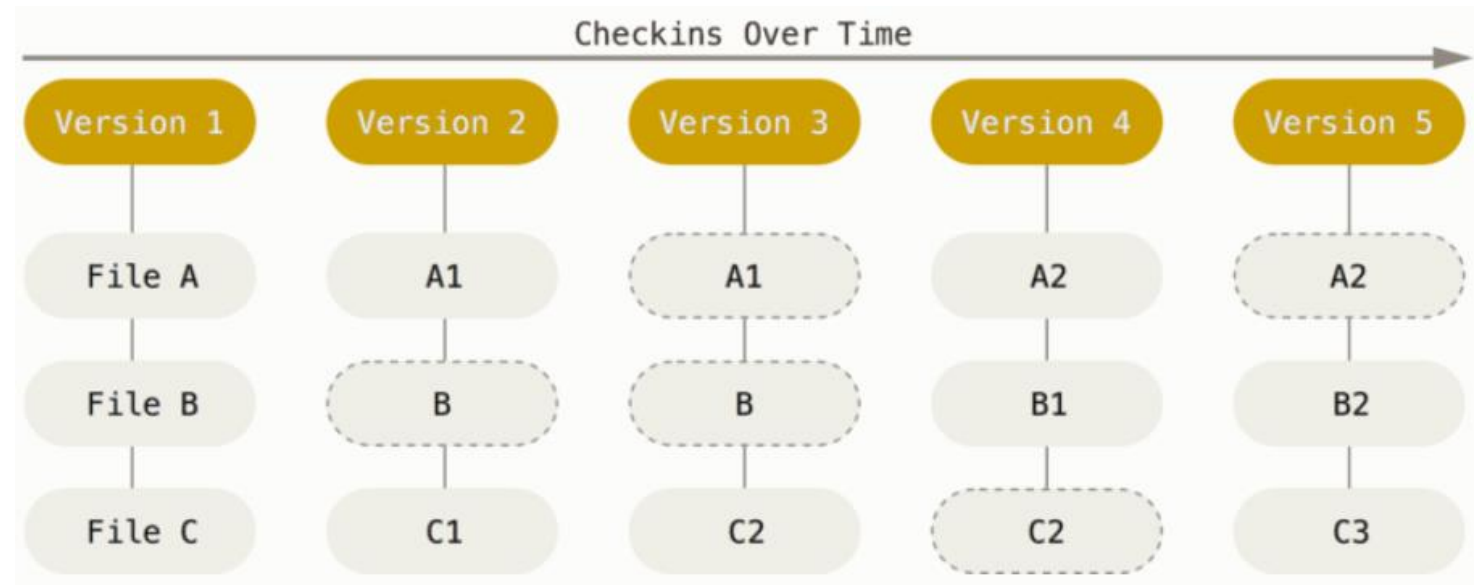
committed

git's view = snap shot stream

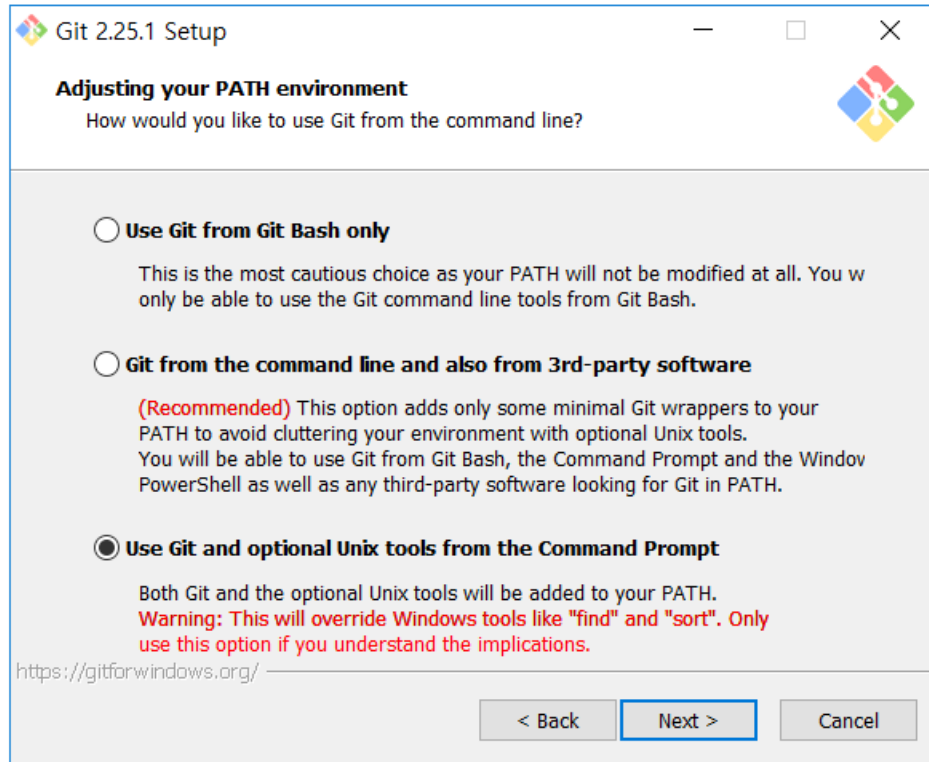


<= Delta based vc

Git : content snap shots =>



Git on Windows



Git on ubuntu

```
$ sudo apt-get install git // git-all
```

Git vs Github

- Git과 Github은 version control 소프트웨어(git)와 remote repository 웹서비스(github)
- Github을 사용하면 사용자의 클라이언트 기계(PC)에서 직접 사용자의 github 계정 안의 repository에 파일을 올려 놓거나 다운 받을 수 있으며 사용법은 매우 직관적이고 쉬움
- 우리 수업에서는 git을 내가 만든다면 어떻게 만들 수 있는지 즉 git 소프트웨어가 어떻게 동작하는지를 공부하기 위해 github에 대해서는 간단하게 소개하고, github에서 제공되는 편리한 기능을 사용하지 않음
- Github 외에 몇 가지 유사한 서비스가 제공되고 있음
- Github 과 같은 기능을 하는 remote git repository를 만드는 것이 어렵지 않음
- 우리가 DevOps를 공부하는 이유는 사용법을 배우는 것이 아니라 각각의 도구가 무엇이며 우리가 직접 만든다면 어떻게 만들 수 있겠는지를 생각하여 DevOps 능력을 배양하는 것

Git set up

- Git (home) directory – 여러 director에 git을 실행 할 수 있으며 각각의 directory는 독립적인 git 소프트웨어의 홈
- `sudo apt-get install git // git-all`
- Git init git-home
 - `mkdir git-home`
 - `cd git-home`
 - `git init`
- `.git/config` 파일
- `git --local <verb> // --local0| default`

Git commands - basics

```
$ git --help
```

```
$ git init    // initialize local git repository, creates .git folder
```

```
$ git add <file>  // staging, i.e., add file(s) to Index
```

```
$ git status    // check status of working tree
```

```
$ git commit    // commit changes in index to local repository
```

```
                // git commit . -am 'commit comment'
```

// commit 한 모든 것은 언제나 복구할 수 있으나, commit하지 않고 수정하거나 잃어 버린 것은 복구할 수 없다


```
$ git config --global user.name 'Han-gyoo Kim'
```

```
$ git config --global user.email 'hgkim@hongik.ac.kr'
```

```
$ git log  // history
```

Git commands – branch and merge

```
$ git branch <branch name>
```

// 옆 가지( 를 만들어 기존 문서를 수정하여 저장

// commit한 그 어떠한 것도 복구할 수는 있으나, 수정할 때는 branch를 하는 습관을 드리는 것이 필요 (branch는 새로운 directory를 만드는 것이 아니라 git의 특정 버전에 속하는 오브젝트들을 가리키는 포인터)

```
$ git merge // 수정본을 원본으로
```

// commit object = 현재 commit의 메타데이터+pointer to previous commit object

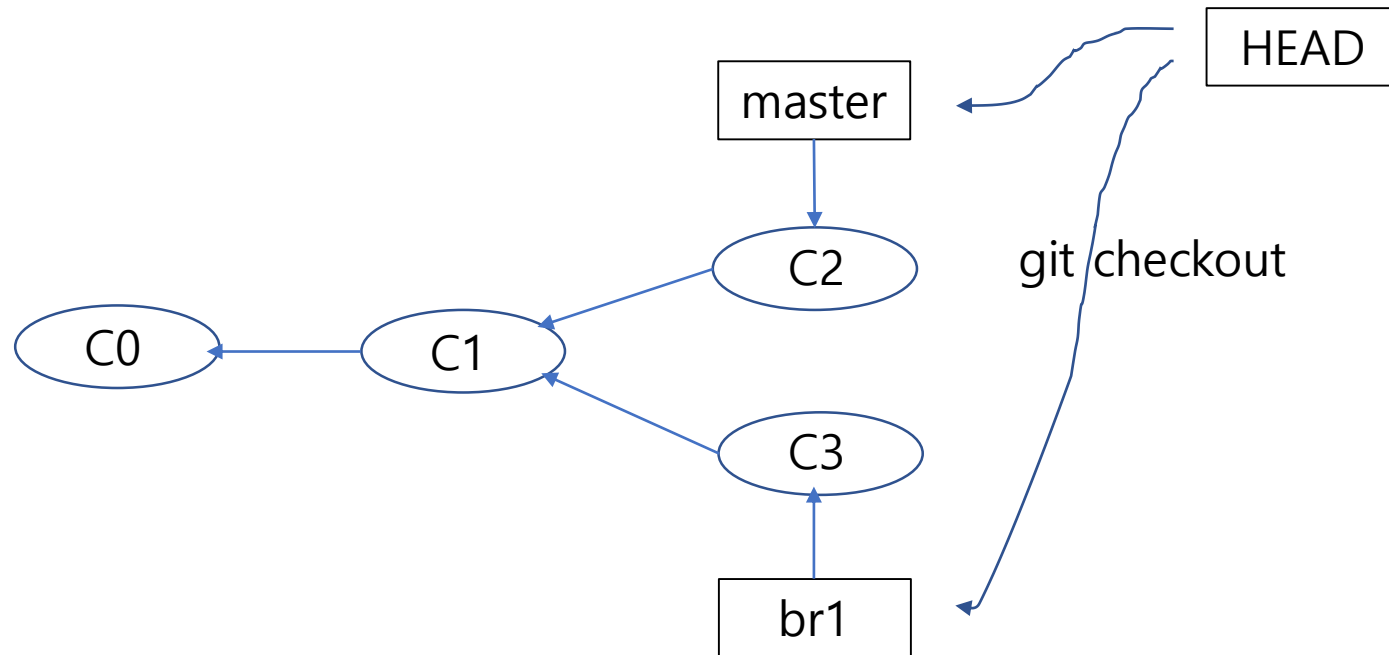
// git branch 는 commit들 사이를 이동할 수 있게 하는 포인터

// 최초로 commit 하면 default로 master branch 생성, 이후 git branch 명령을 통해 새로운 수정본을 가리키는 브랜치 이름으로 브랜치 생성

// 핫은 HEAD라는 이름의 포인터를 사용하며 HEAD는 현재 작업 중인 local branch를 가리킨다 (git branch 이전에는 master branch를 가리키며 git branch를 실행하여도 git checkout 이전에는 여전히 master branch를 가리킨다)

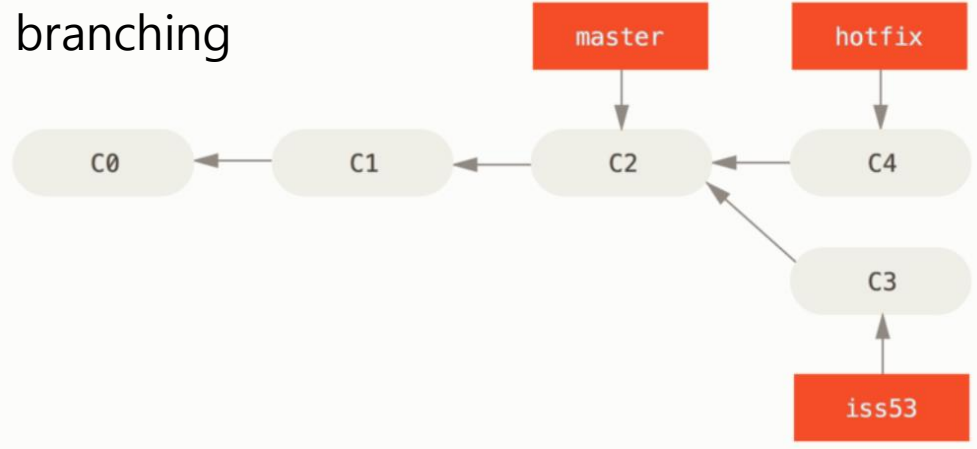
```
$ git checkout <branch name>
```

Branches – link of commit objects

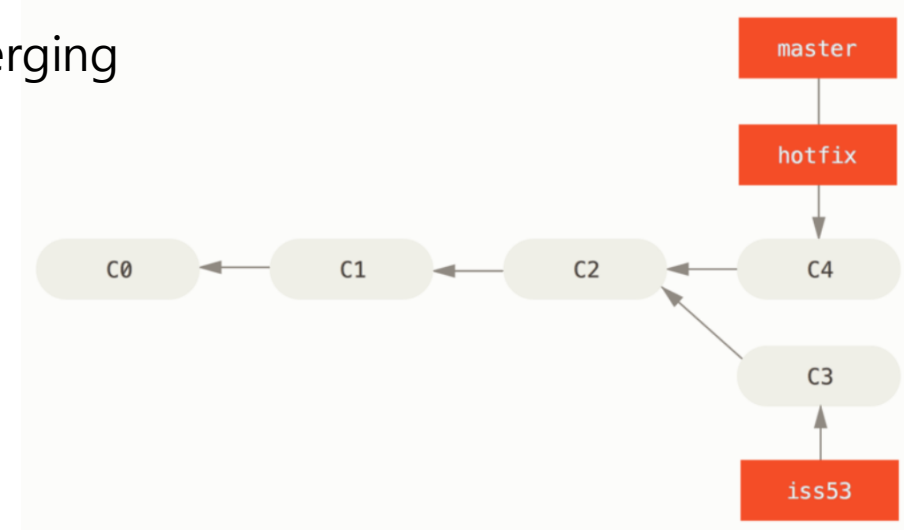


Git branch – fast forward

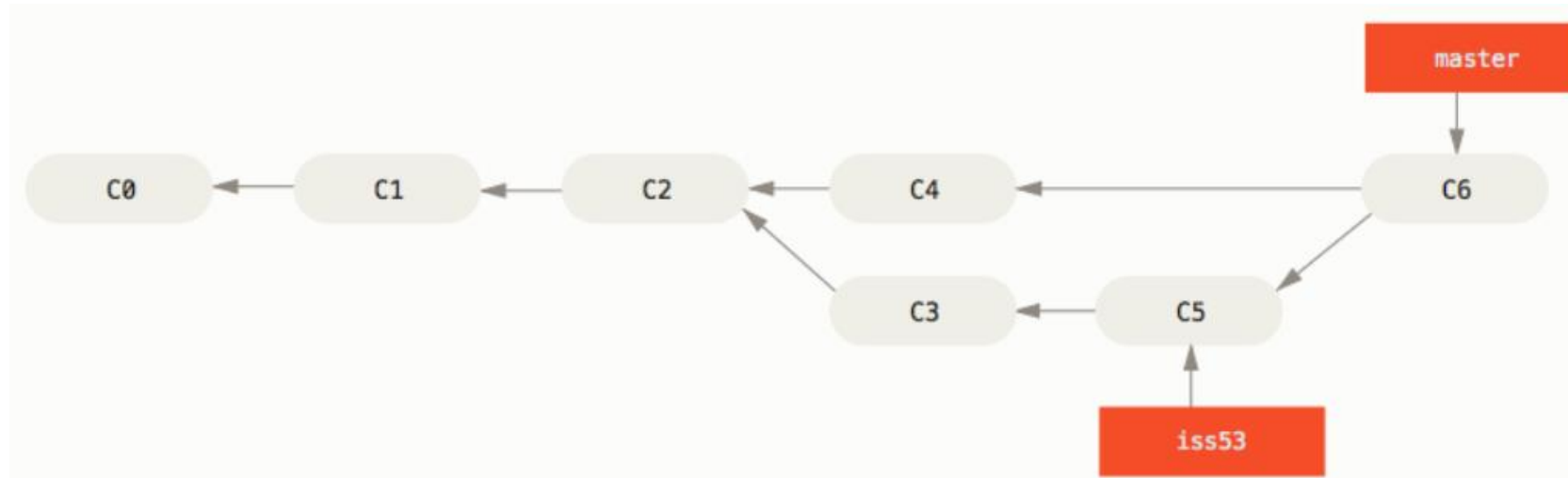
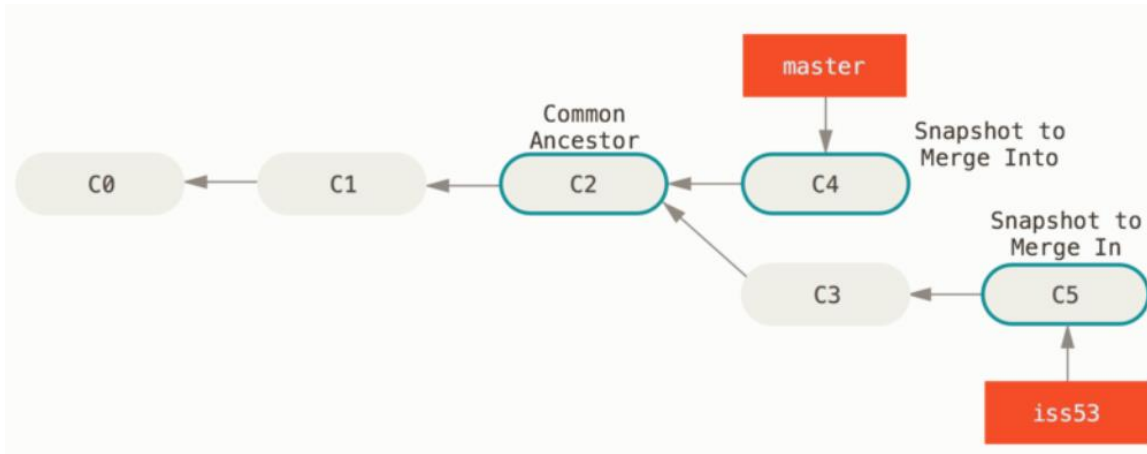
branching



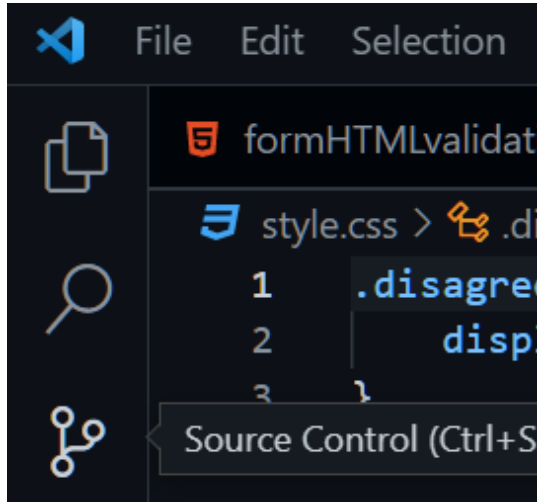
merging



Git branch – 3 way merge



Git integration to VS Code



- 우리는 ubuntu 기계에서 작업하였으나, 만일 여러분이 PC나 Mac에서 작업을 수행하는 경우에는 모든 명령을 terminal이나 powerShell에서 수행할 수 있으나, 보다 편리하게 editor에서도 git을 사용할 수 있음
- VS code가 대표적인 예
- MS가 github을 인수, VS code는 MS의 freeware
- Vs code에서 git을 이용하는 것은 새로운 내용은 없으므로 각자 연습하기로 함