

오픈소스 소통을 위한 Git 공부하기

최영락, 마이크로소프트

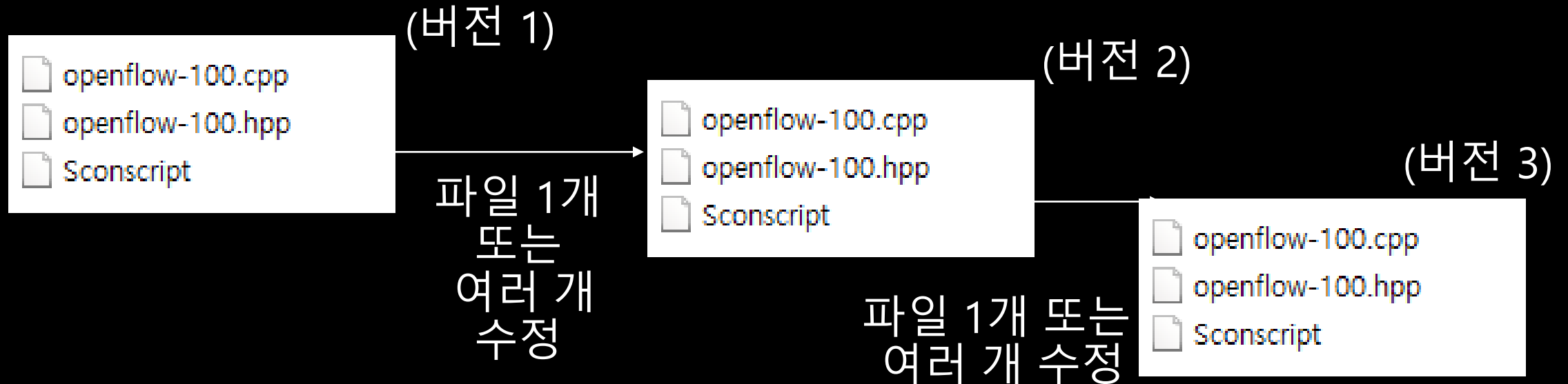


#1. 버전 관리의 중요성



버전 관리란 무엇인가요?

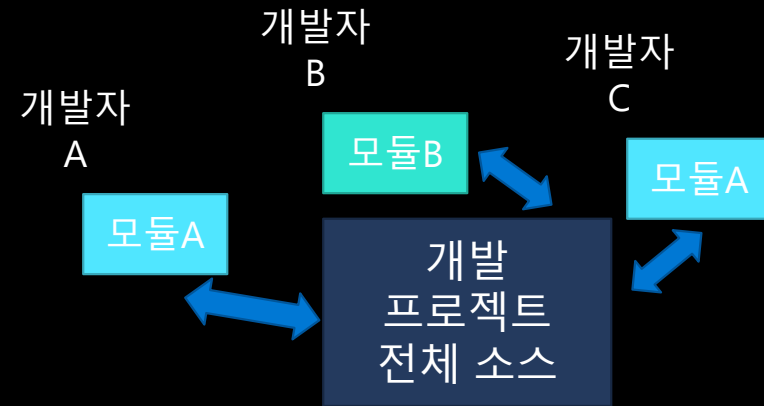
- 소스 하나 또는 묶음을 하나의 버전으로 간주하여 관리합니다.
- 파일/폴더를 추가/수정/삭제하여 사람이 직접 묶음을 버전으로 관리하자고 지정
- 원하는 때 예전 버전 내용 전체를 되돌려 볼 수 있음
- 특히 복잡한 코드 개발할 때는 이전 버전과 비교해 보기가 쉬워짐



버전 관리를 대체 언제 사용하나요?

1) 개발 협업을 위해 사용합니다.

- 전체 개발 소스를 공유하면서 개발 파트를 나누어 공유
- 같은 모듈을 개발하더라도 소스를 서로 공유하면서 개발
 - 이 때, 특정 파일은 변경하지 못하도록 lock을 걸 수도 있습니다 (이걸로 퇴근 못하는 직원 분들도 꽤...)
- 권한 설정을 통해 각 개발자 별로 접근 가능한 소스 목록도 제어하기도 합니다.



2) 개별적으로 버전 등 이력을 관리하기 위해 사용하는 경우도 있습니다.

3) 버전 관리되는 내역 전체를 오픈 소스로 공유하기도 합니다.

다른 버전 관리 도구들은?

- 오픈 소스 버전 관리 도구

- CVS: 90년대 말 – 2000년대쯤? 아는 분들은 사용한다고 들었던 버전 관리 도구
Subversion (SVN)이 대중화되면서 쓰는 사람들이 많이 줄었던 것 같음
- Subversion (SVN): 여러 파일 업로드 중 실패 시 롤백 (원자성, atomicity),
이진 파일 지원, rename (파일 이름 변경도 이력관리) 등 CVS를 대체 가능한
많은 기능이 추가되어 많은 사용자들이 CVS에서 전환된 것으로 알고 있습니다.
오늘날과 같이 Git이 많이 사용되기 전 가장 보편화되었던 버전 관리 도구
- Mercurial: Git와 비슷한 류의 분산 버전 관리 도구라고 합니다.

- 상용 버전 관리 도구

- (제가 써 본 건 Microsoft Visual SourceSafe와 Team Foundation Server밖에..)

'분산' 버전 관리 도구는 머가 다른가요?

- 이를 설명하기 위해 보통 Subversion과 Git를 많이 비교합니다.
- Subversion은 버전 관리를 하려면 반드시 Subversion 서버와 통신하여 버전을 관리해야 합니다.
- 반면, Git는 네트워크가 되지 않아도 우선 자신의 컴퓨터에서 버전 관리를 하고 네트워크 통신이 될 때 서버와 통신해도 됩니다.
- 이런 이야기도 있었습니다.
미국의 개발자들은 인터넷이 안되는 휴양지에 가서도 Git를 이용하면 충분히 개발할 수 있고,
나중에 인터넷 되는 곳에 가서 서버와 쪽 통신하여 업무를 할 수도 있다고..
반면 한국은 인터넷이 아주 잘 되니 Git가 대중화되지 않을 수도 있겠다고...
→ 그런데, 요즘은 Git이 많이 사용되네요 ☺

#2. Git를 소개합니다



Git는 무엇일까요?

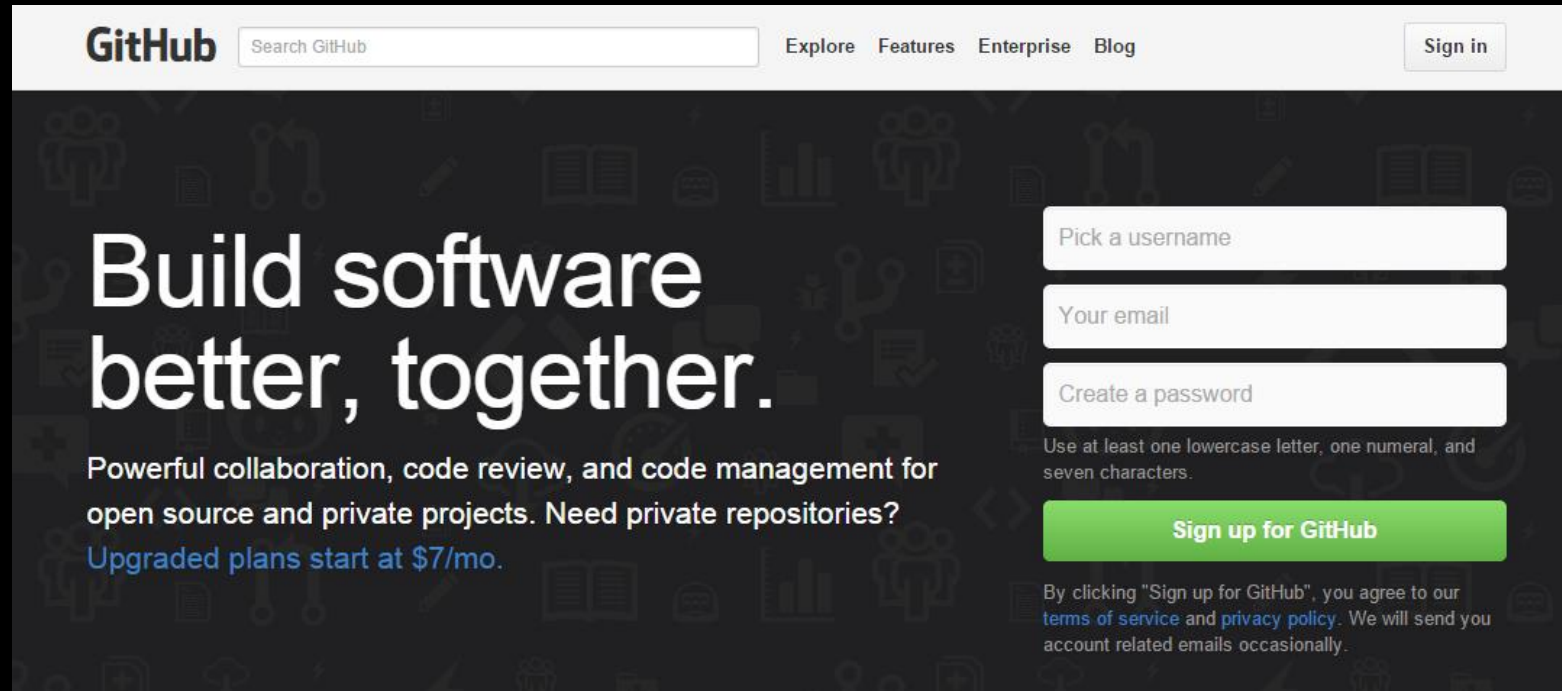
- Git
 - 버전 관리를 위한 도구
 - 리눅스 커널의 창시자, 리누스 토발스께서 직접 만드심
 - BitKeeper라는 상용 도구를 무료 사용하여 버전 관리를 했었는데 무료 사용이 제고되면서 직접 만들게 되었다고 함



Linux Torvalds
(출처: Wikipedia)

깃허브 (GitHub)란 무엇인가요?

- <https://www.github.com>
 - “Git 저장소”를 클라우드라는 공간에서 누구나 사용 & 관리할 수 있는 서비스입니다
 - 이제는 단순히 Git 소스 저장소만을 의미하는 것이 아닌, 개발자 커뮤니티를 위한 공간이 되었습니다
 - Git를 잘 알아야 깃허브를 사용할 수 있겠죠? 😊



The screenshot shows the GitHub homepage with a dark background and white text. The main heading is "Build software better, together." followed by a description of the service. On the right side, there is a sign-up form with three input fields: "Pick a username", "Your email", and "Create a password". Below these fields is a green "Sign up for GitHub" button. At the bottom right, there is a small disclaimer about terms of service and privacy policy.

GitHub Search GitHub Explore Features Enterprise Blog Sign in

Build software better, together.

Powerful collaboration, code review, and code management for open source and private projects. Need private repositories? Upgraded plans start at \$7/mo.

Pick a username

Your email

Create a password

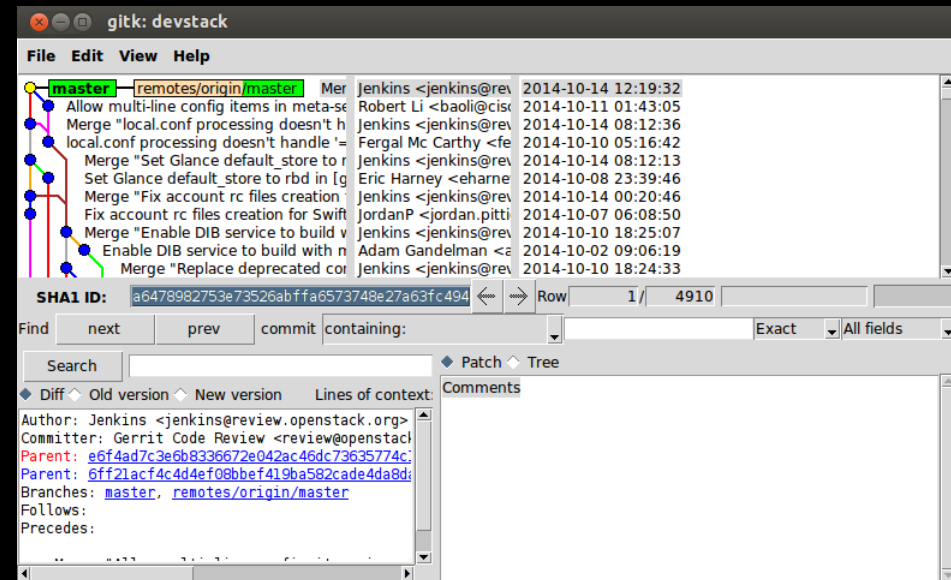
Use at least one lowercase letter, one numeral, and seven characters.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We will send you account related emails occasionally.

Git 명령어들을 꼭 익혀야 하나요?

- 적어도 초창기에는 그랬습니다.
- 지금도 Git를 잘 쓰려면 명령어들을 알면 도움이 많이 됩니다.
- 요즘엔 명령어들을 몰라도 pull, push, commit 등의 기본 개념만 알면 쓸 수 있는 좋은 GUI들이 많습니다.
- GitHub Desktop, TortoiseGit, gitk, ...
- 그래픽 화면으로 나와 명령어보단 쓰기 편합니다. ☺



#3. 소스를 가져오기 위한 Clone



Git 용어부터 살펴봅시다

- Clone (클론)

- 다른 Git 소스 관리 저장소에 있는 내용을 그대로 복제해오는 것을 이야기합니다

[관련 용어]

- Pull (풀, 끌어오기)

- Git 저장소 서버로부터 내 컴퓨터 로컬로 버전 정보 전체를 가져옵니다

- Push (푸시, 밀어넣기)

- 내 컴퓨터 로컬에 저장되어 있던 버전 정보를 Git 저장소 서버로 내보냅니다
-

#4. 변경 단위를 만들기 위한 Commit



Git 용어부터 살펴봅시다

- Commit

- 추가/수정/삭제된 폴더/파일들을 1개 버전으로 간주하여 내 컴퓨터 로컬에 버전 정보를 기록합니다

#5. 브랜치 (Branch) 이해하기



Git 용어를 봅시다

- 브랜치 (Branch)

- 버전들을 묶어서 Branch라고 합니다.
- 기본은 **main** 입니다. (예전에는 master)
- 내 컴퓨터 내에 있는 Branch는 로컬 branch, 외부 서버에 있는 Branch는 리모트 branc라고 합니다.

[관련 용어]

- Tag

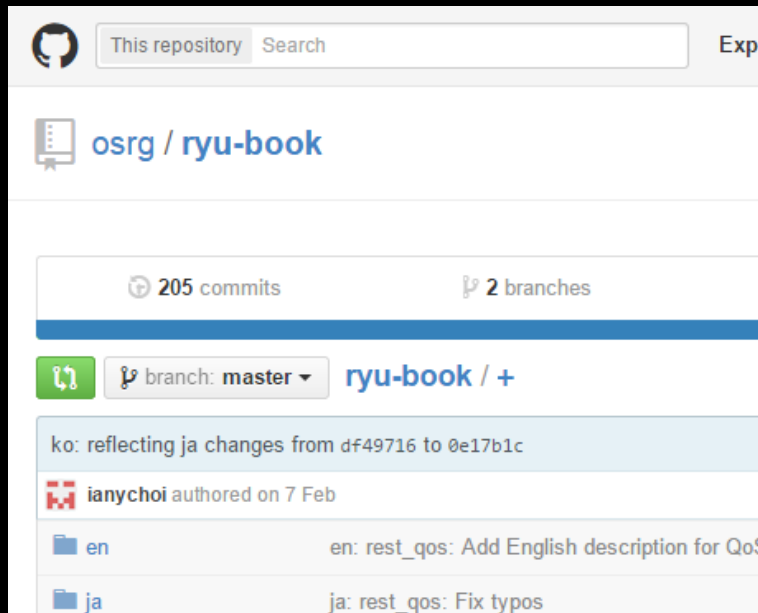
- Branch는 왔다갔다하면서 각각에 대해 내용을 변경하고 버전 관리까지 가능합니다.
 - 반면 Tag는 특정 버전 위치에 대해 나중에 쉽게 찾아갈 수 있도록 이름을 지정해 놓은 것 뿐입니다.
-

#6. 내 저장소에 소스를 푸시하기 위한 Fork

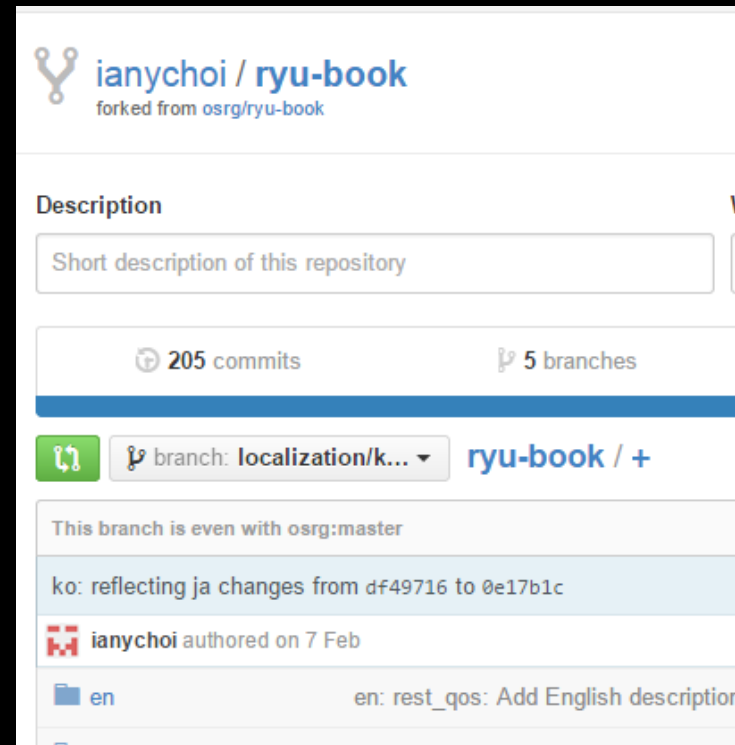


깃허브에서 제공하는 Fork 기능이란?

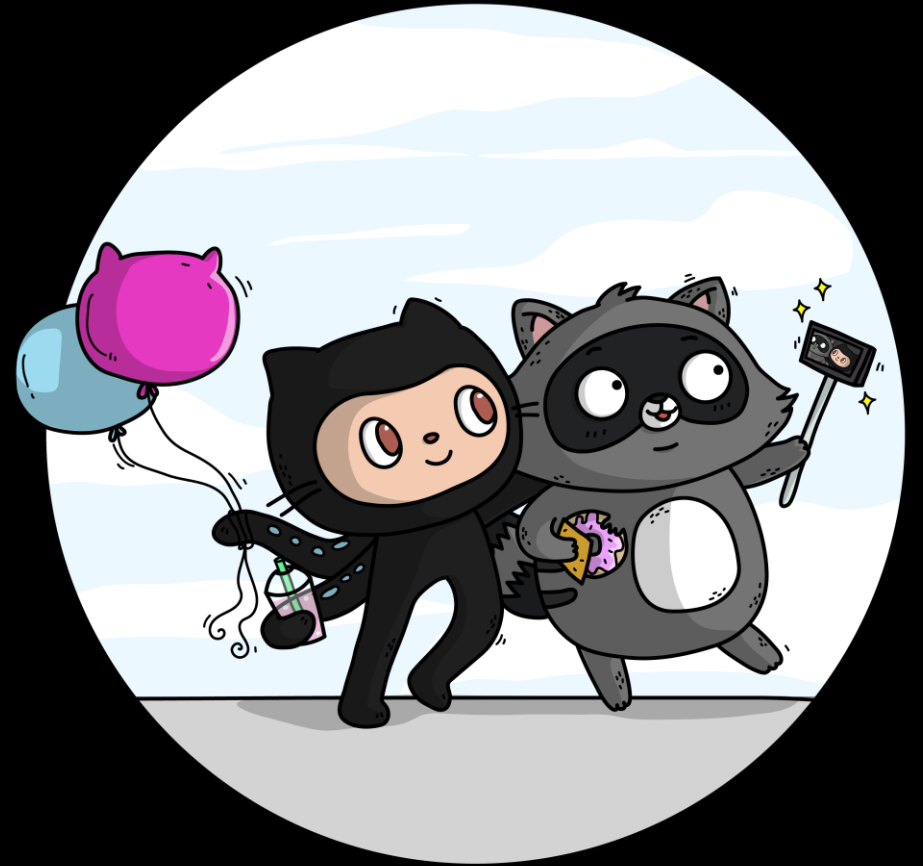
- 다른 사람 저장소를 가져와 제 저장소로 만들어 놓는 기능입니다.



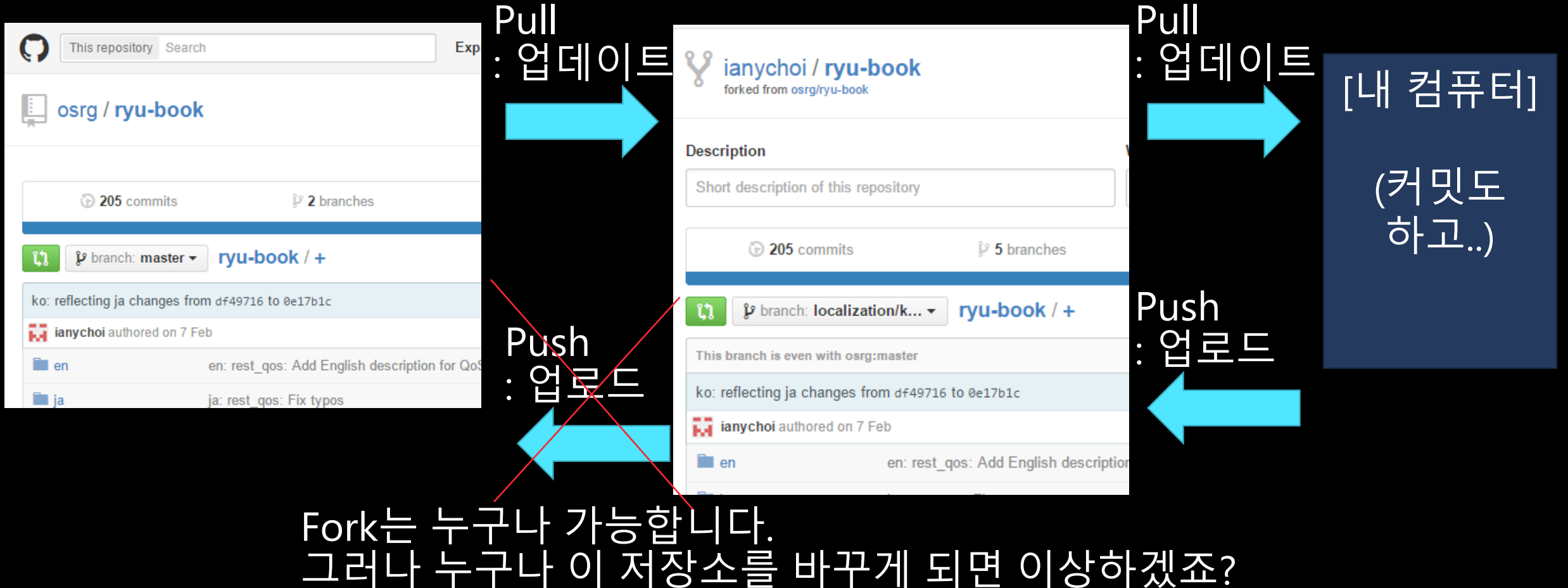
fork



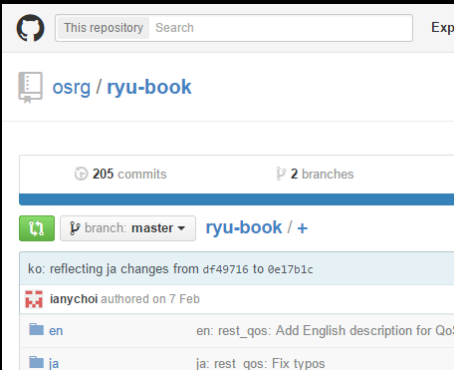
#7. 변경한 소스 반영을 요청하는 풀리퀘스트



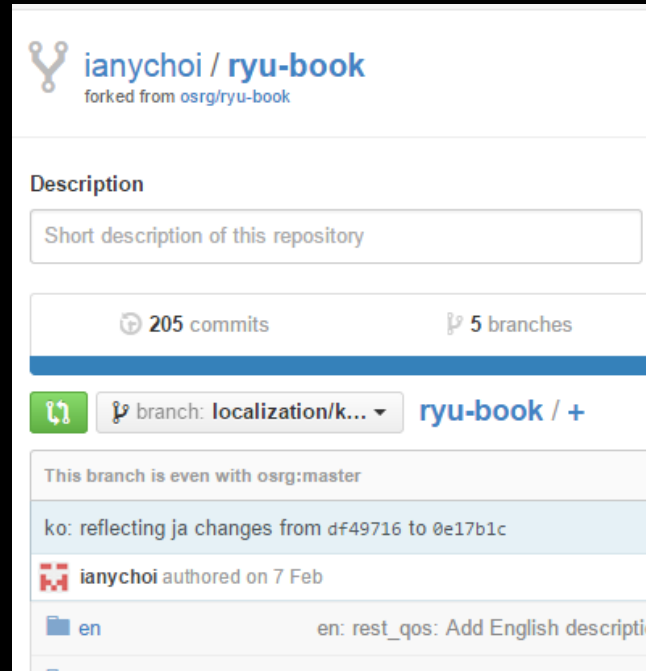
깃허브 – Fork, 다 좋은데 안되는 게 있습니다



깃허브 - 풀리퀘스트 (Pull Request)란?



Pull
: 업데이트



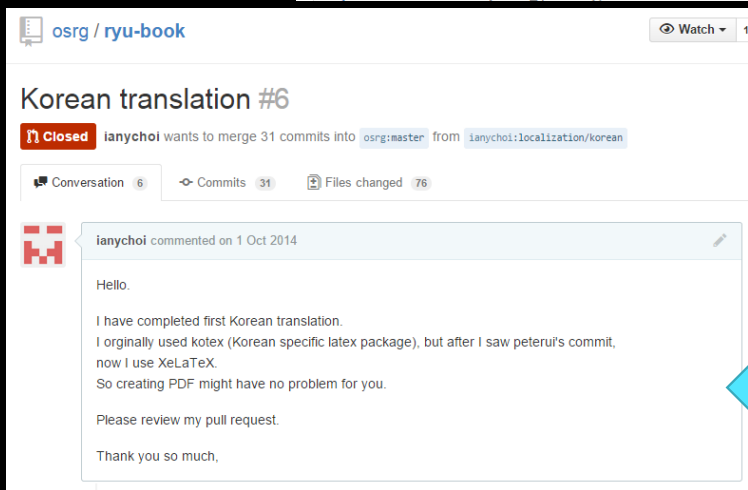
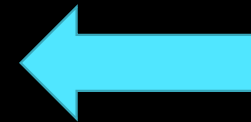
Pull
: 업데이트



[내 컴퓨터]

(커밋도
하고..)

Push
: 업로드



Pull
request
: 검토
부탁합니다



원 저장소 소유자 또는 권한이 있는 자가 검토하여
맘에 들면 가져와서 업데이트하고, 토론/반려 등이 가능합니다.

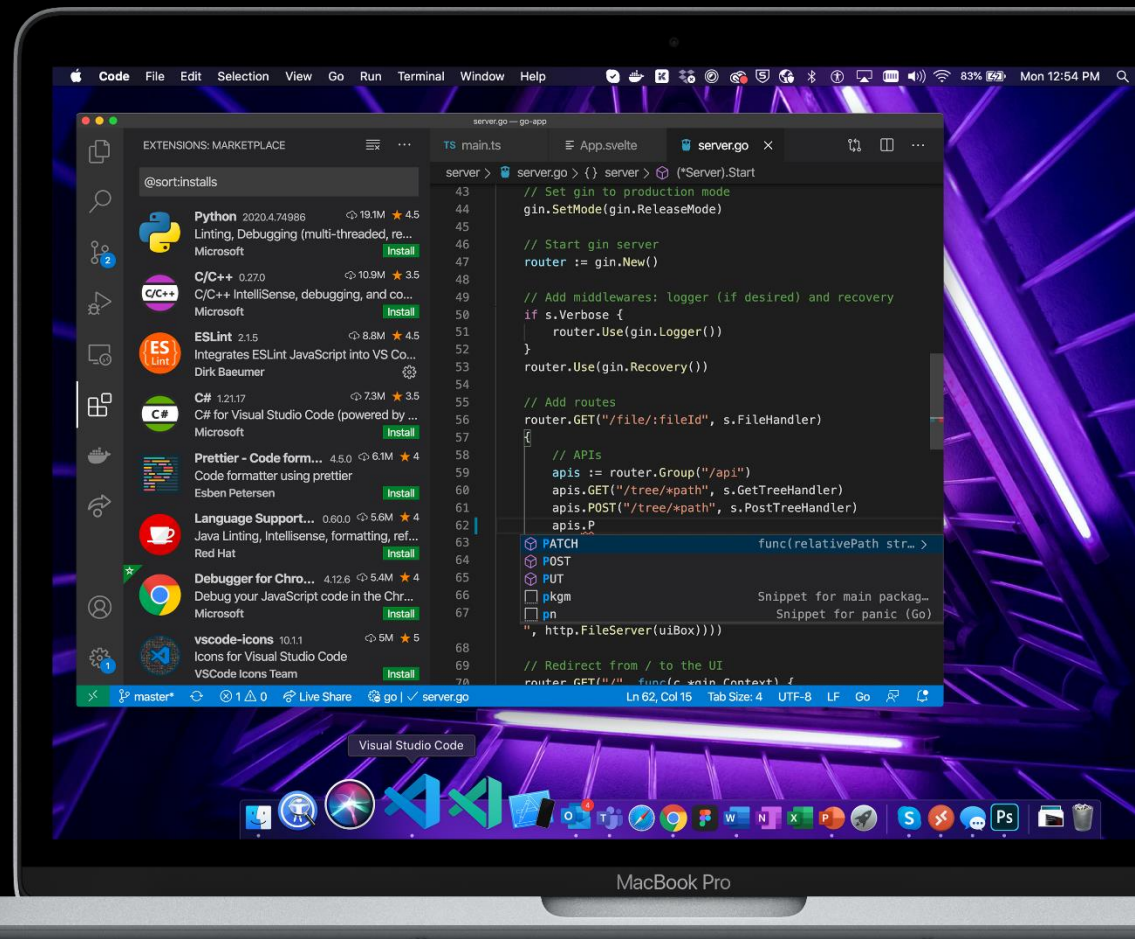
#8. 비주얼 스튜디오 코드로 쉽게 Git 사용하기



Visual Studio Code

- </> Linux, macOS, Windows 모두 무료이면서 오픈소스 방식으로 개발/관리되는 소프트웨어
- 🔧 20,000개 이상 확장 기능이 커뮤니티를 통해 컨트리뷰션이 이루어짐
- <> 로컬, 원격에서 뿐만 아니라 (웹) 브라우저에서도 Codespaces를 통해 개발 가능

code.visualstudio.com



#9. 클로징



참고 리소스

- Microsoft Learn에 깃허브 관련 좋은 내용이 있습니다

- <https://aka.ms/learn-github>



GITHUB을 위한 MICROSOFT LEARN

세계에서 소프트웨어를 빌드하는 곳

필수 Git 및 GitHub 기술을 학습, 개발 및 마스터하고 전 세계 수백만의 개발자 및 회사와 함께 세계 최대 규모의 가장 발전된 개발 플랫폼 중인 GitHub에서 소프트웨어를 빌드, 제공 및 유지 관리하세요. 여기에서 시작하여 경력을 바로 시작하고 재미있는 대화형 모듈 및 경로를 통해 기본 GitHub 학습 목표를 보여 주세요.

PROGRAMS

GitHub 학습 랩을 통해 과정 진행

GitHub 학습 랩을 통해 재미있고 현실적인 프로젝트를 완료함으로써 기술을 성장시키고 진숙한 학습 랩으로부터 조언과 유용한 피드백을 받으세요.

GitHub 학습 랩 살펴보기

PROGRAMS

학생이나 강사인가요?

GitHub Education을 통해 학생, 교사 및 학교는 차세대 소프트웨어 개발을 진행하는 데 필요한 도구와 이벤트에 액세스할 수 있습니다.

프로그램 세부 정보 살펴보기

가이드

GitHub 가이드를 통해 지식 확장

효율적인 일일 GitHub 사용을 위한 기능 데모, 팁 및 워크플로 기본 사항을 살펴보세요.

가이드 찾아보기

기본 GitHub 지식의 권장 학습 경로

학습 경로

GitHub에서 프로젝트의 수명 주기 관리

6시간 23분

초급 | DevOps 엔지니어 | GitHub

학습 경로

GitHub Actions를 사용하여 워크플로 자동화

4시간 40분

초급 | DevOps 엔지니어 | GitHub

학습 경로

Markdown 및 GitHub 페이지를 사용하여 다른 사용자와 협업

3시간 26분

초급 | DevOps 엔지니어 | GitHub

학습 경로

GitHub에서 커뮤니티 기반 소프트웨어 프로젝트 빌드

6시간 35분

초급 | DevOps 엔지니어 | GitHub

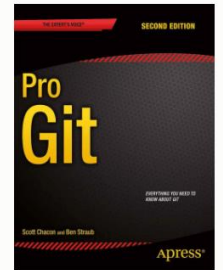
참고 리소스

- Git-scm: <https://git-scm.com/book/ko/v2>
 - 한글로 된 무료 eBook을 제공합니다



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](#).



2nd Edition (2014)

1. 시작하기

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

Download Ebook

