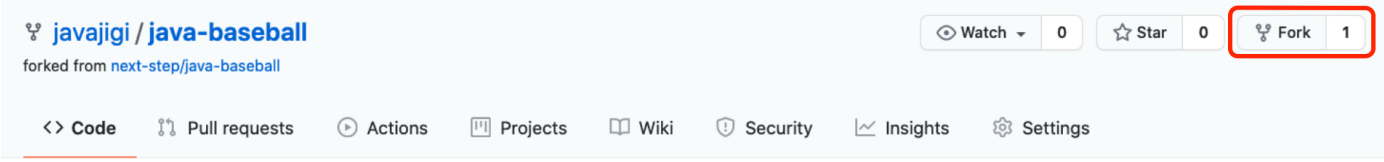
📖 미니과제 진행 가이드

1. 저장소를 내 계정으로 포크하기

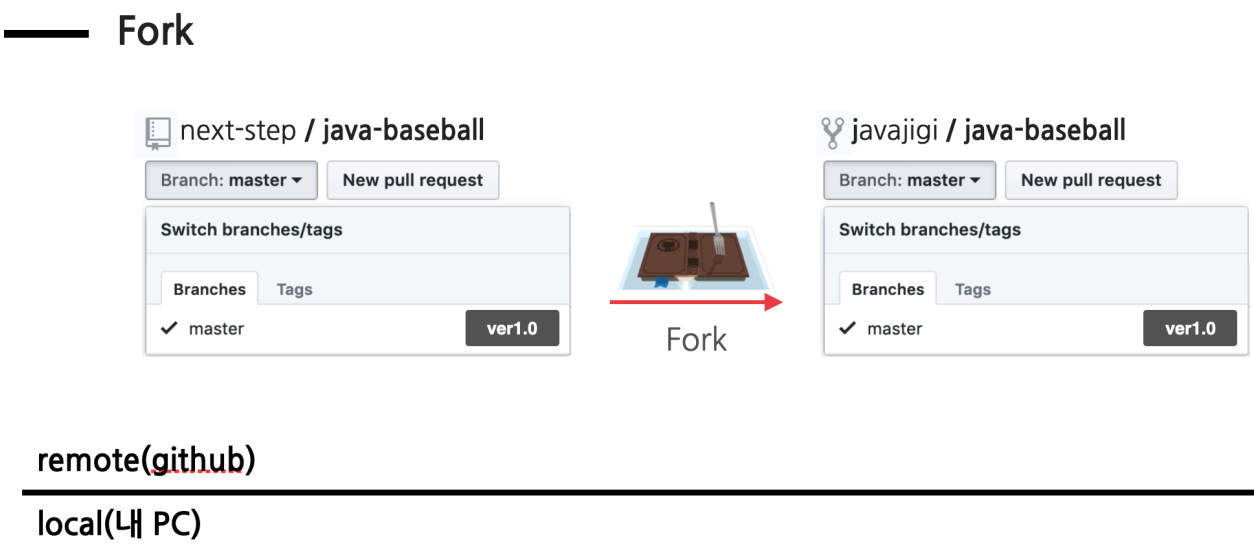
아래 이미지와 같이 next-step 저장소의 오른쪽 상단에 있는 포크 버튼을 클릭하여 포크합니다. 앞으로 모든 미션은 **개인 계정의 포크된 저장소를 활용**하게 됩니다.



포크는 `upstream` 저장소라고도 하는 **원본 저장소에 영향을 미치지 않고 프로젝트를 변경**하도록 해줍니다. 저장소를 포크한 후, 원본 저장소에서 업데이트를 가져와 포크를 최신 상태로 유지하고, **풀 리퀘스트**를 사용하여 포크에서 **원본 저장소로 변경 내용을 제안**할 수 있습니다.

실행 결과

포크 완료 후 저장소 상태는 아래와 같습니다.



2. 포크된 저장소를 내 컴퓨터로 클론하기

저장소를 클론하려는 적절한 디렉터리로 이동하여 터미널에 다음과 같은 명령을 입력합니다.

```
git clone https://github.com/{사용자_이름}/{저장소_이름}.git
cd {저장소_이름}
```

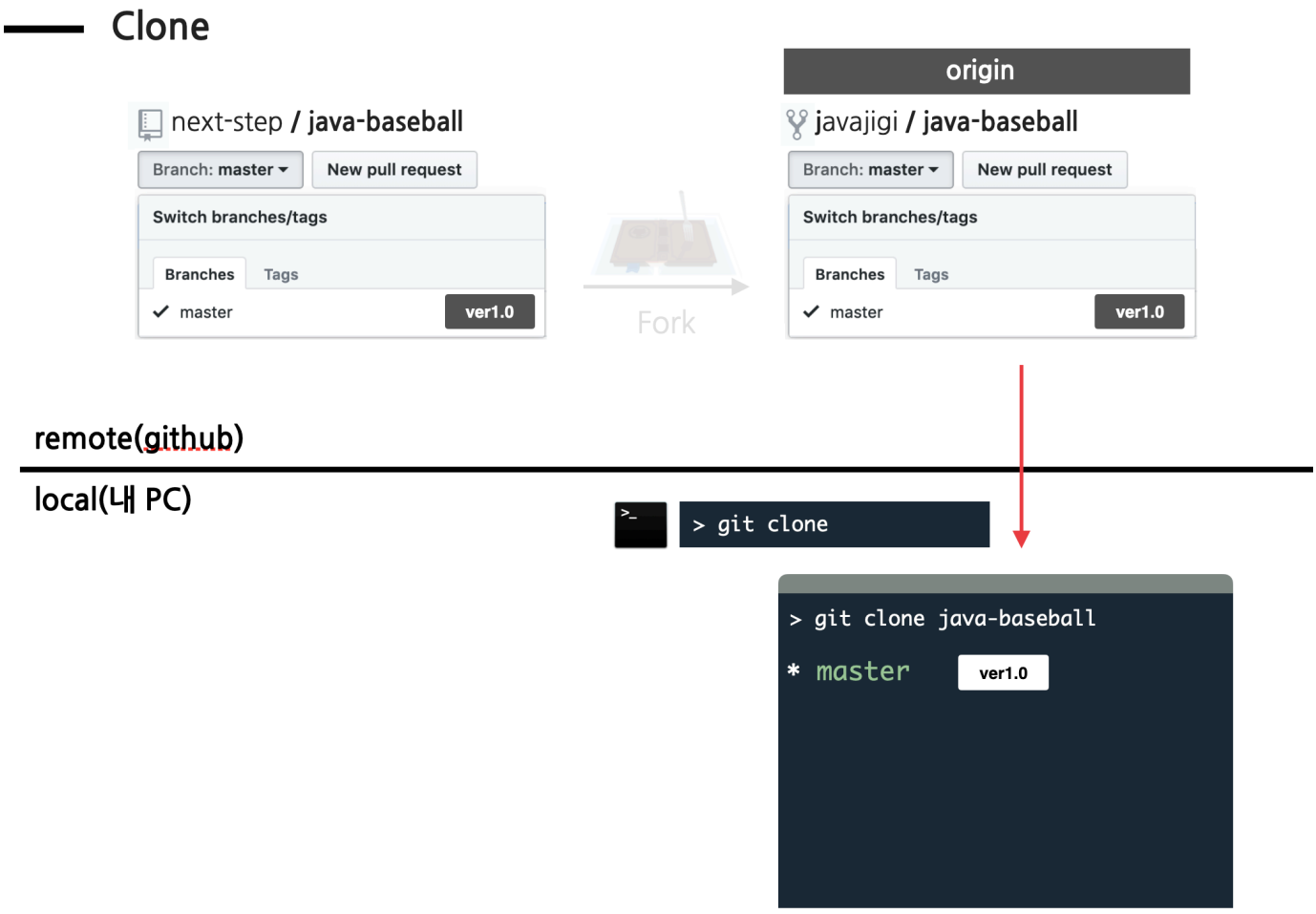
e.g.

```
git clone https://github.com/next-step/java-baseball-precourse.git
cd java-baseball-precourse
```

클론은 **GitHub에 있는 저장소를 내 컴퓨터로 복사하는 과정**입니다. 저장소를 GitHub에서 컴퓨터로 복제하면 병합 충돌을 더 쉽게 해결하고, 파일을 추가 또는 제거하고, 더 큰 커밋을 푸시할 수 있습니다.

실행 결과

클론이 완료된 후 저장소의 상태는 아래와 같습니다.



3. 기능 구현을 위한 브랜치 만들기

터미널에서 다음 명령을 입력하여 브랜치를 만듭니다.

```
git checkout -b {사용자_이름}
```

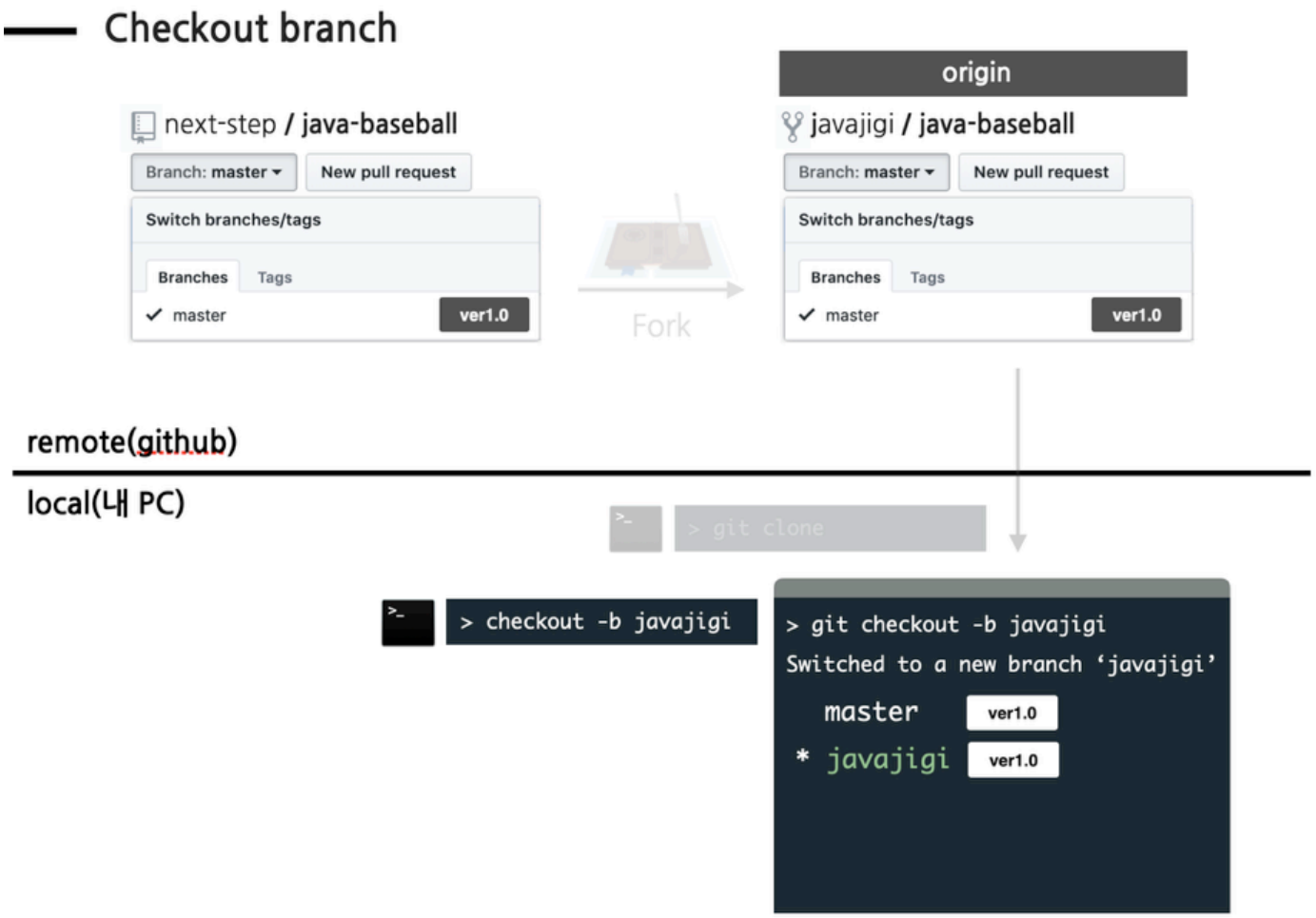
e.g.

```
git checkout -b javajigi
```

모든 버전 관리 시스템은 **브랜치**를 지원합니다. 개발 중에는 코드를 여러 개의 복사본으로 복사해야 하는 경우가 종종 있습니다. 코드 전체를 복사한 후에는 **원본 코드와 독립적으로 개발**할 수 있으며, 이렇게 독립적으로 개발하는 것을 브랜치라고 합니다.

실행 결과

브랜치를 만든 후에는 아래와 같은 상태가 되어야 합니다.

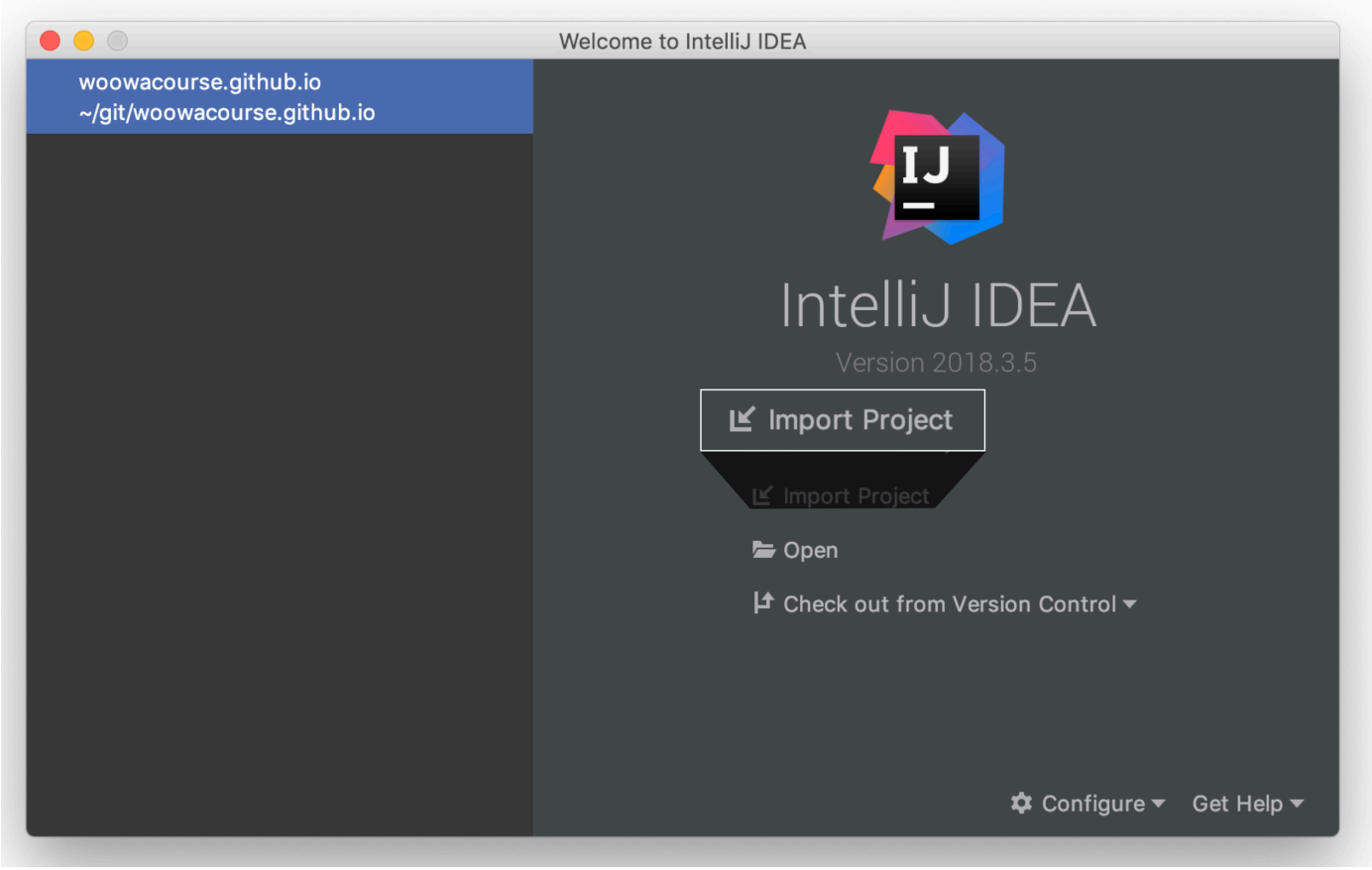


4. 통합 개발 환경(IDE)으로 가져오기

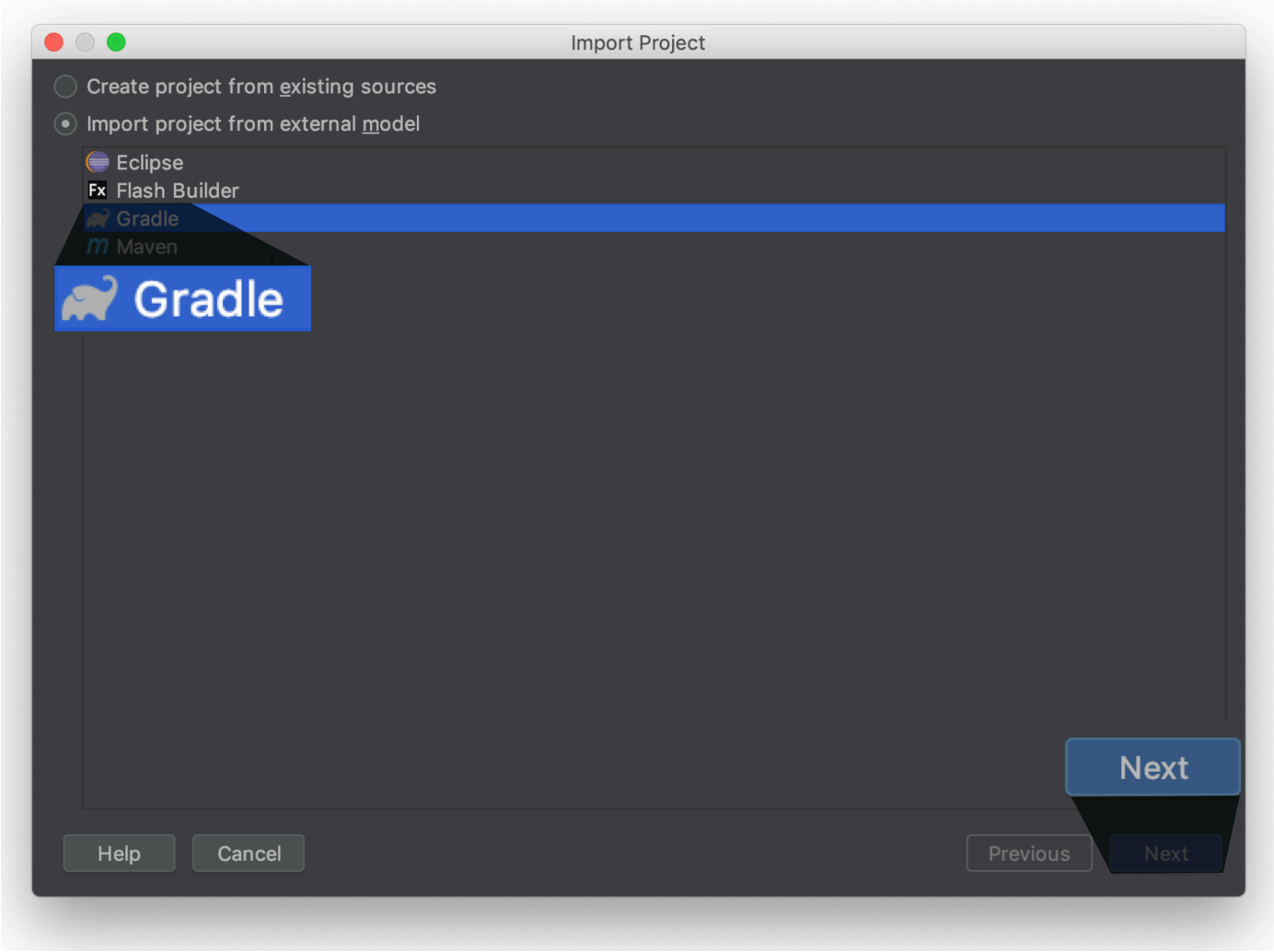
미션 진행을 위해 클론한 저장소를 즐겨 사용하는 통합 개발 환경(IDE)으로 가져옵니다.

4.1. IntelliJ IDEA으로 가져오기

1. IntelliJ IDEA를 실행합니다.
2. 아래 화면에서 Import Project를 선택합니다.



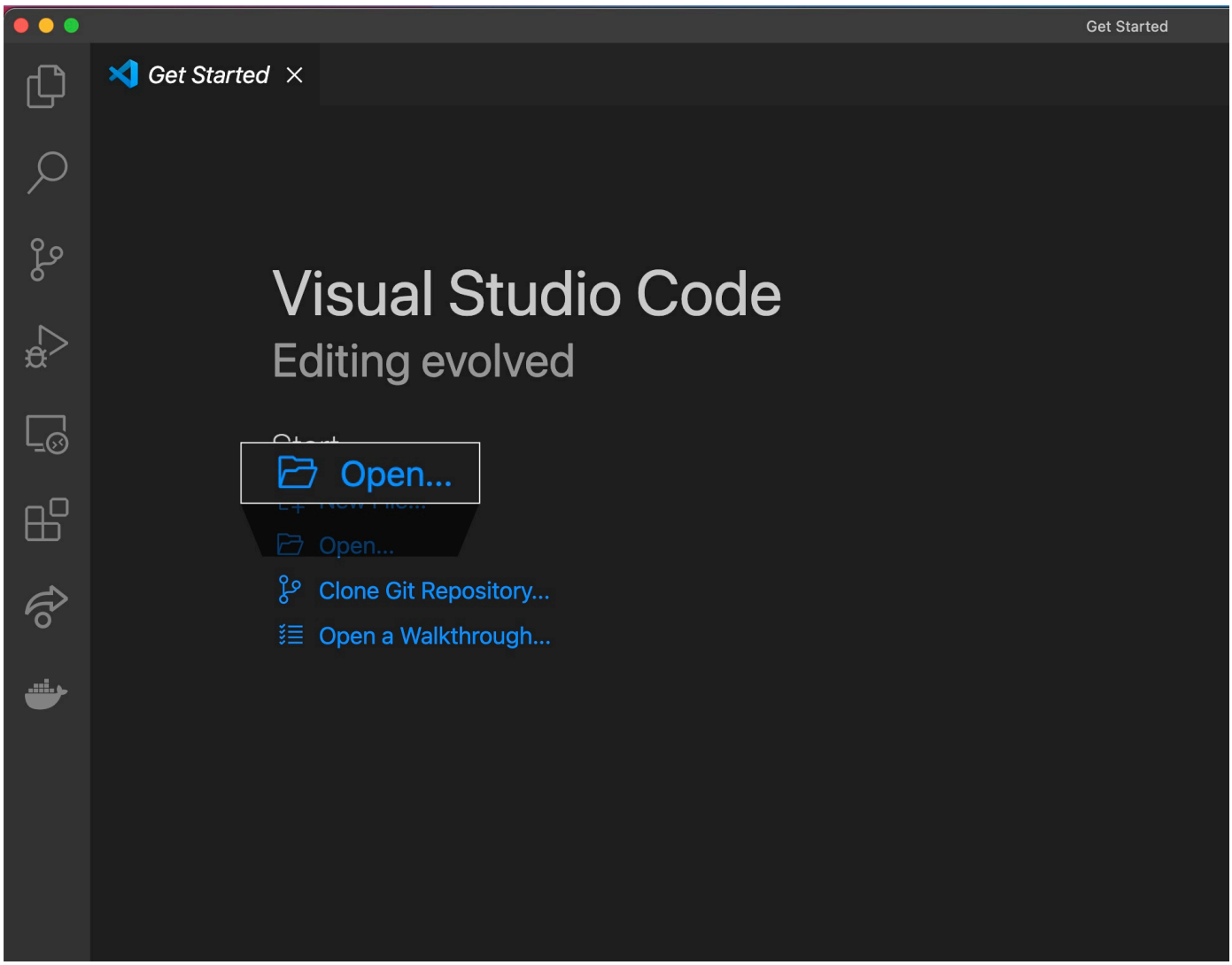
3. 앞서 클론한 저장소의 디렉터리를 선택합니다.
4. 아래와 같이 "Import Project from external model"를 선택하고 Gradle -> Next를 선택합니다.



5. 다음 화면에서 Finish를 클릭하여 가져오기를 완료합니다.

4.2. Visual Studio Code으로 가져오기

1. Visual Studio Code를 실행합니다.
2. 아래 화면에서 Open을 선택합니다.



3. 앞서 클론한 저장소의 디렉터리를 선택합니다.

5. 기능 구현하기

미션 요구 사항을 파악하고 기능을 구현하세요.

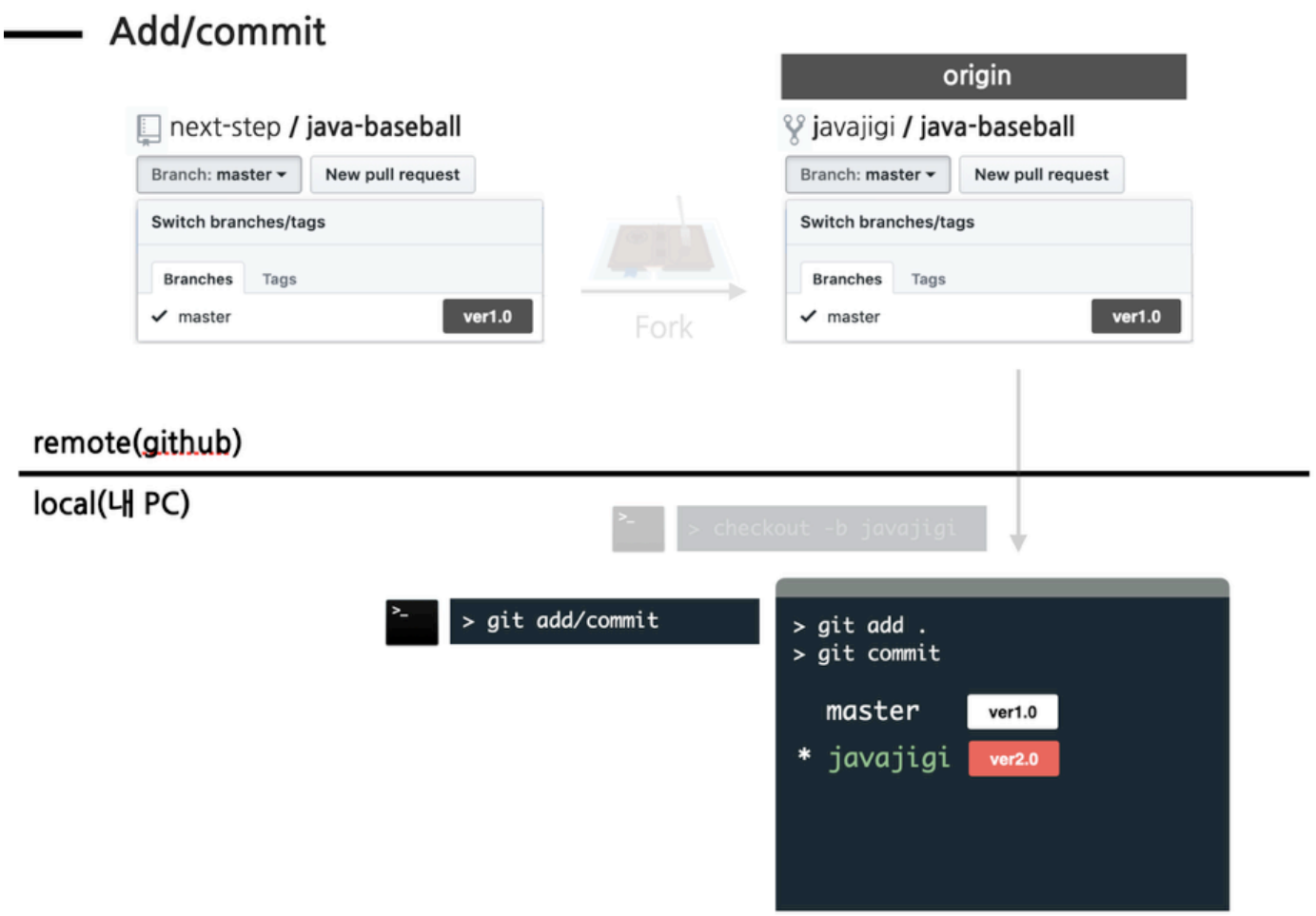
6. 기능 구현 후 git add, commit

기능 구현을 완료한 후 `git add` 및 `git commit` 명령을 사용하여 변경 사항을 로컬 저장소에 반영합니다.

```
git status # 변경된 파일 확인
git add -A (또는 .) # 변경된 모든 파일을 한 번에 반영
git commit -m "message" # 작업 내용을 메시지에 쓰기
```

실행 결과

기능 구현을 완료하고 `git add` 및 `git commit` 명령을 실행한 후의 상태는 아래와 같습니다.



7. 원격 저장소에 올리기

로컬에서 `git commit` 명령을 실행하면 **원격 저장소가 아닌 로컬 저장소에만 반영**됩니다. 원격 저장소에도 동일하게 반영하려면 `git push` 명령을 사용합니다.

```
git push origin {사용자_이름}
```

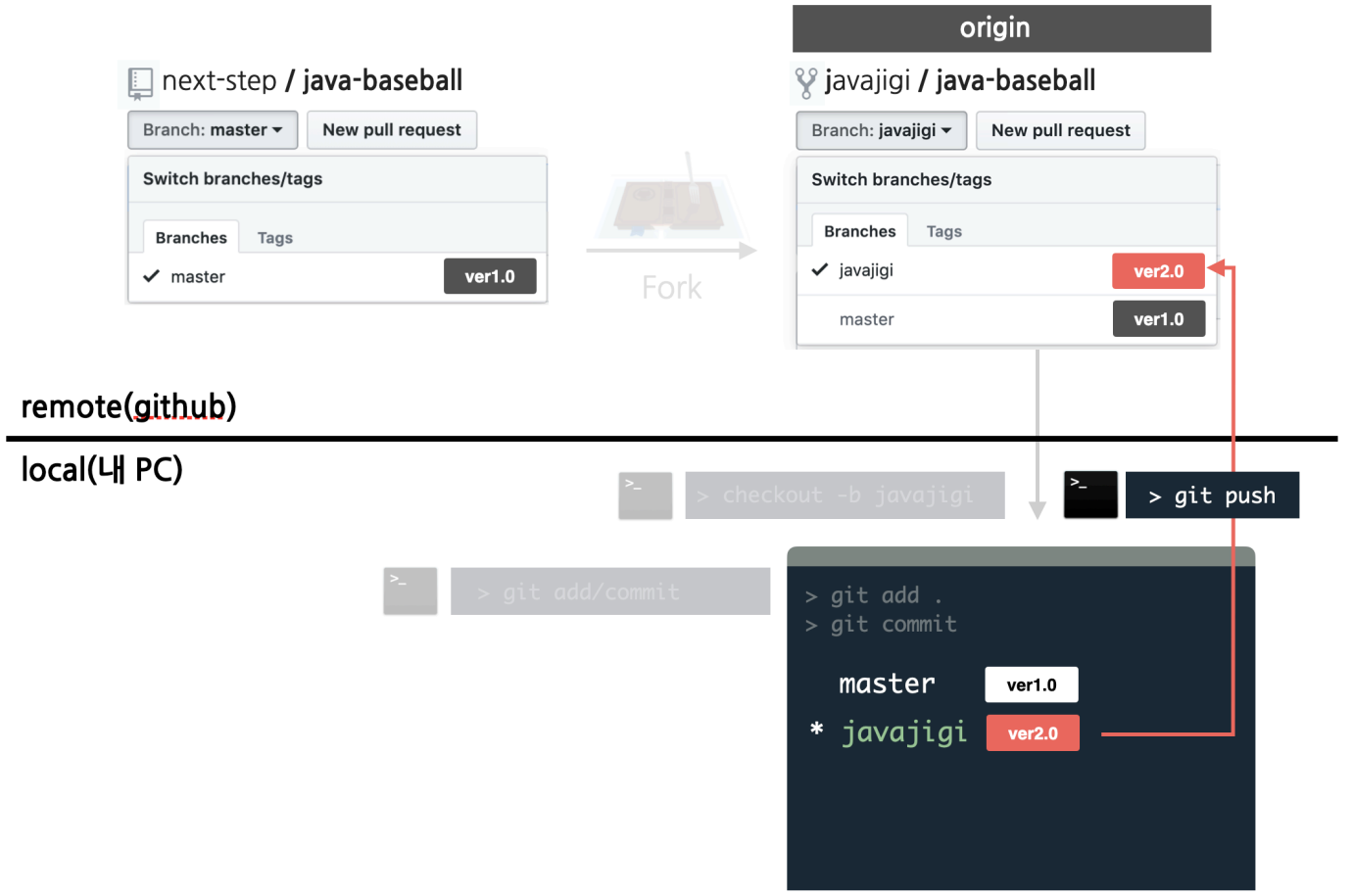
e.g.

```
git push origin javajigi
```

실행 결과

`git push` 명령을 실행한 후의 상태는 아래와 같습니다.

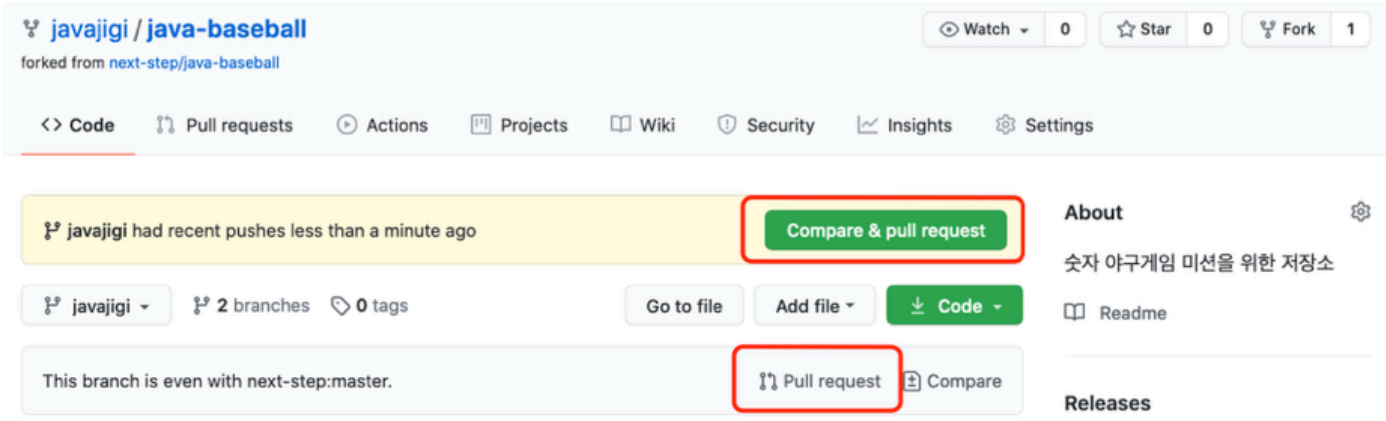
Push



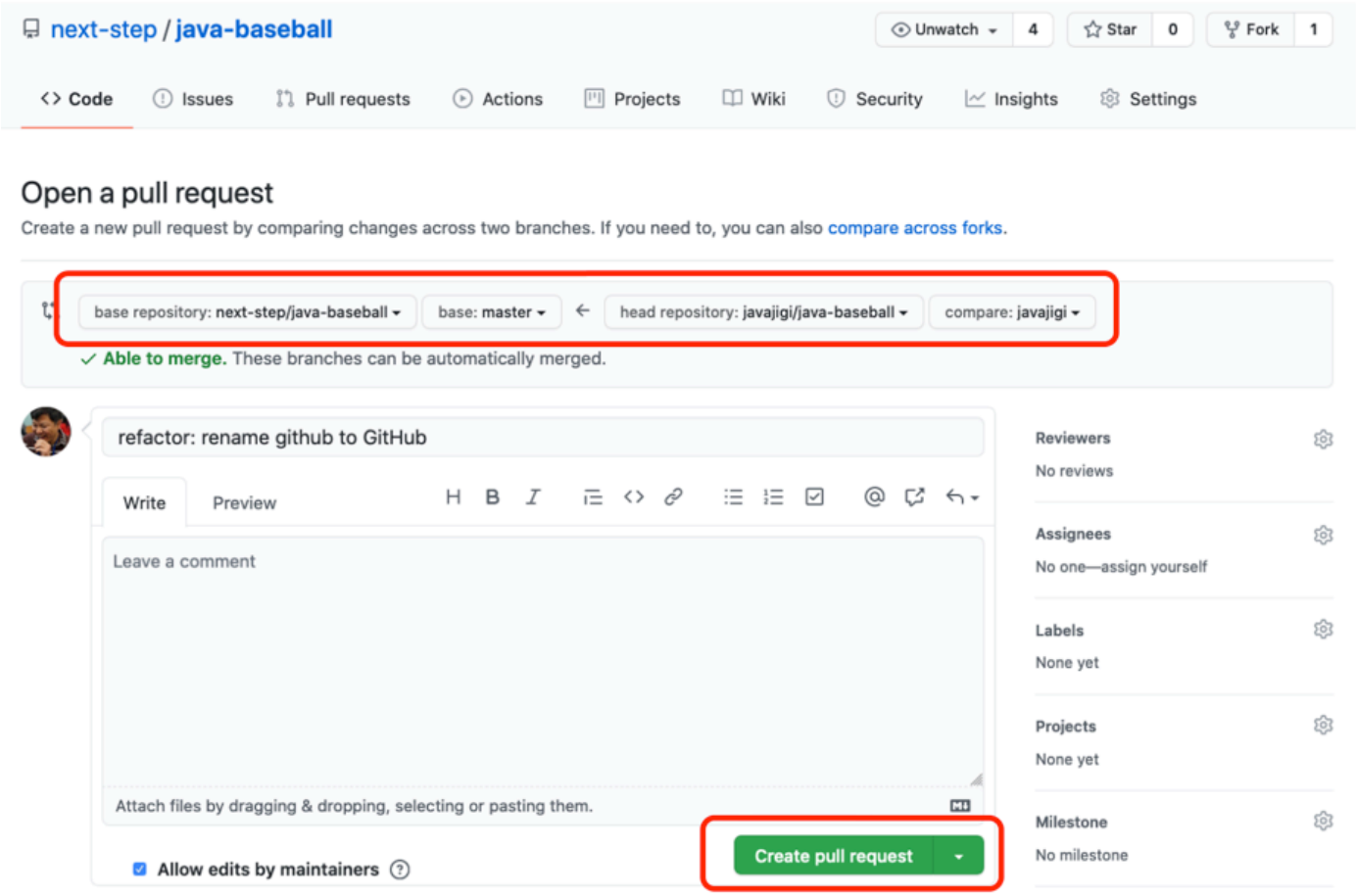
8. GitHub에서 풀 리퀘스트 만들기

풀 리퀘스트는 **코드 리뷰를 요청하는 데 사용하는 GitHub에서 제공하는 기능**입니다. 풀 리퀘스트는 원본 저장소의 `main` 브랜치와 이전 단계에서 원격 저장소에 올린 브랜치를 기반으로 합니다.

1. 브라우저에서 GitHub 원격 저장소에 접근합니다.
2. 브랜치를 작업 브랜치로 변경합니다.
3. 브랜치 오른쪽에 있는 "New pull request" 버튼을 클릭합니다.

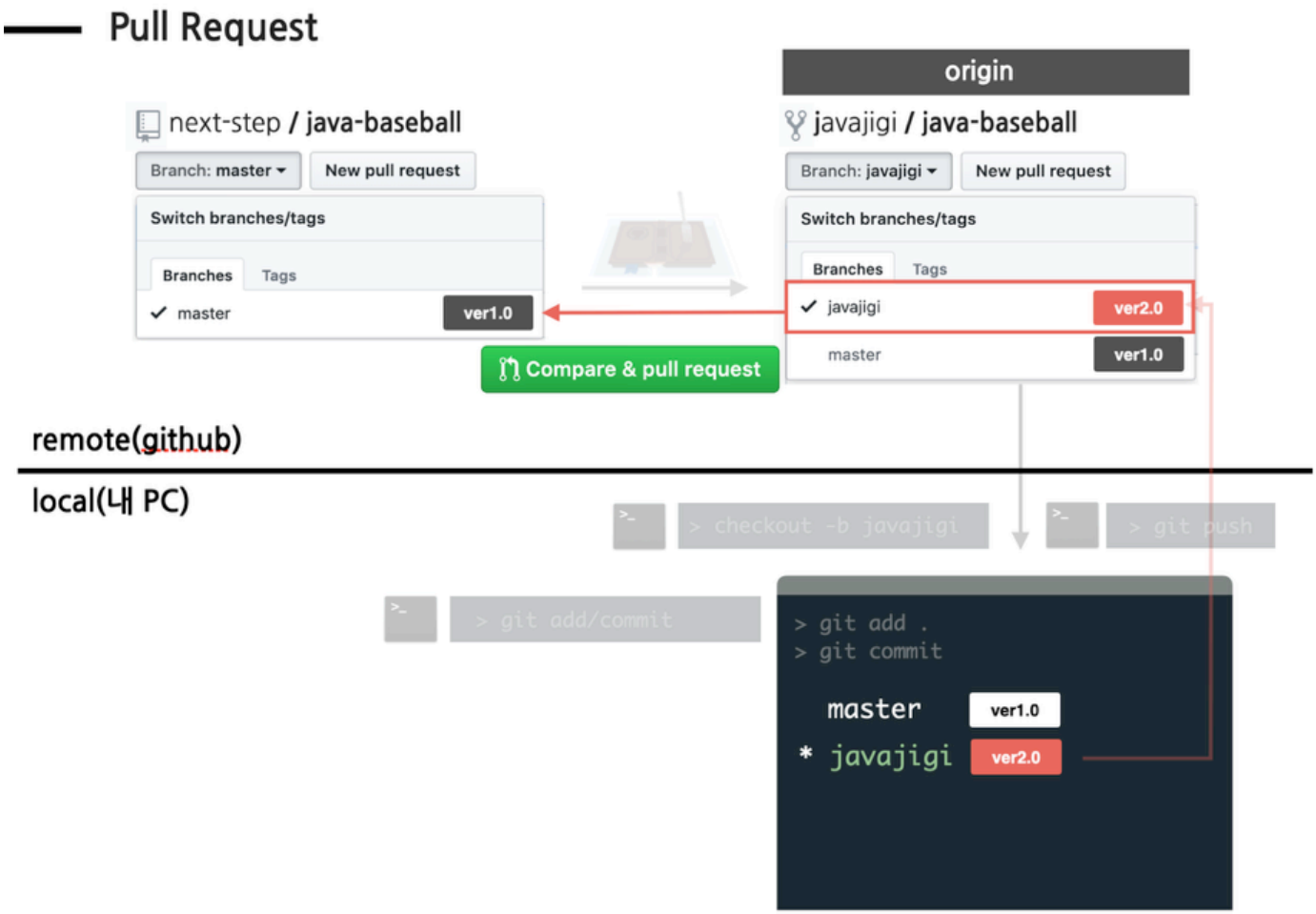


4. 현재 미션에서 작업한 내용을 입력하고 "Create pull request" 버튼을 클릭해 풀 리퀘스트를 만듭니다.



실행 결과

풀 리퀘스트를 만들었을 때의 상태는 아래와 같습니다.


Pull Request
origin
next-step / java-baseball
javajigi / java-baseball
Branch: master
New pull request
Branch: javajigi
New pull request
Switch branches/tags
Branches
Tags
master
ver1.0
Switch branches/tags
Branches
Tags
javajigi
ver2.0
master
ver1.0
Compare & pull request
remote(github)
local(내 PC)
> checkout -b javajigi
> git push
> git add/commit
> git add .
> git commit
master
ver1.0
* javajigi
ver2.0

9. 미니과제 제출하기