

# Github 중급

# 준비사항

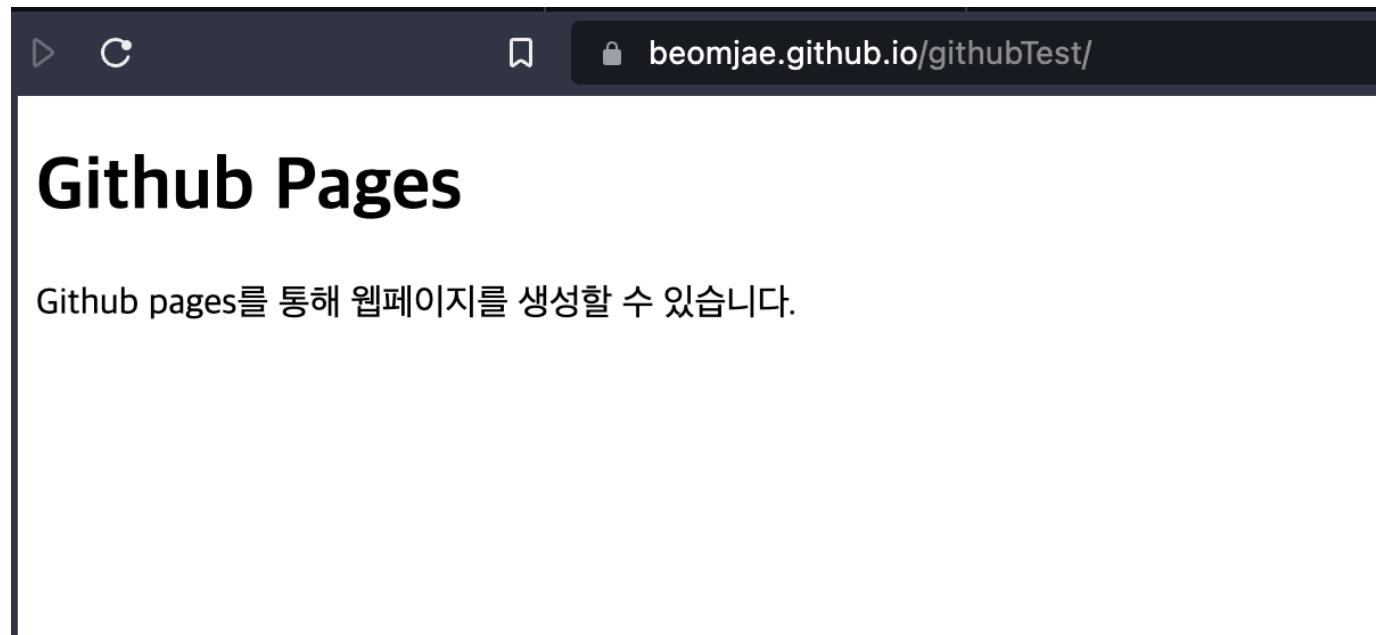
- Github 가입
- Fork 설치

받아오고  
추가하고  
올리고  
보내는 것



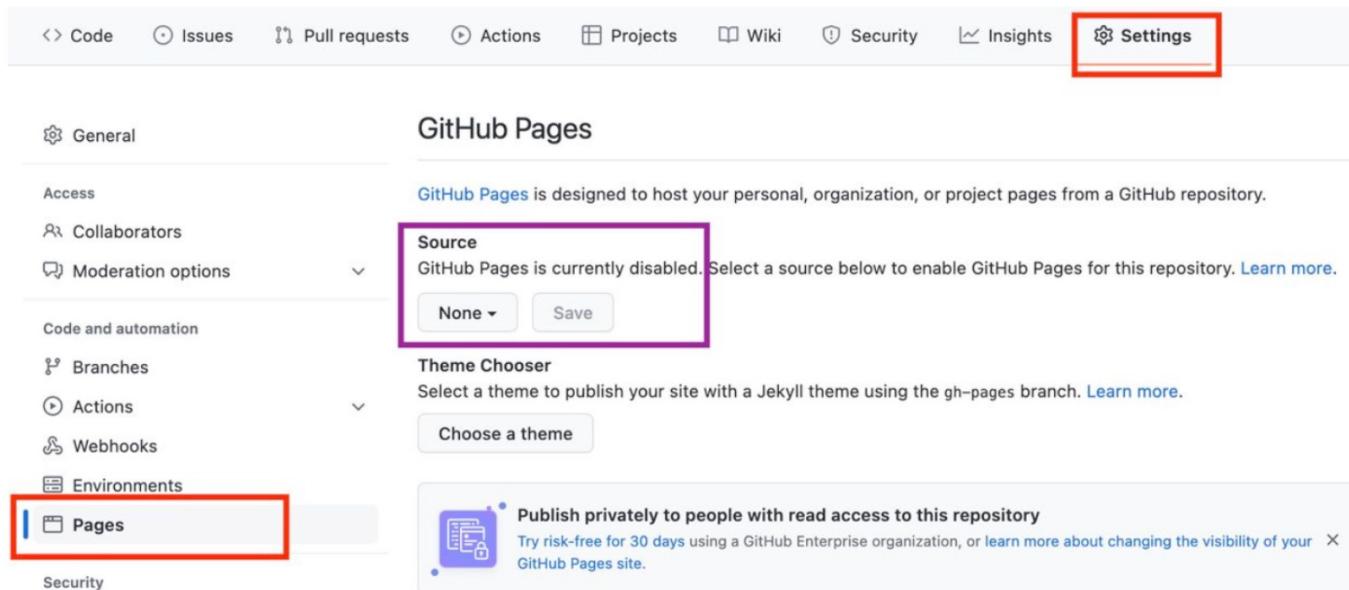
# 1. Pages 사용하여 웹 페이지 생성하기

<https://beomjae.github.io/githubTest/>



# 1. Pages 사용하여 웹 페이지 생성하기

- Github 프로젝트 생성하기 (readme 파일 포함)
- Index.html 파일 추가하기
- Settings – Pages에서 설정하기



Branch: main ▾

/ (root) ▾

Save

Select branch

Select branch

main

None

All theme using the gh-pages branch. [Learn more.](#)

read access to this repository

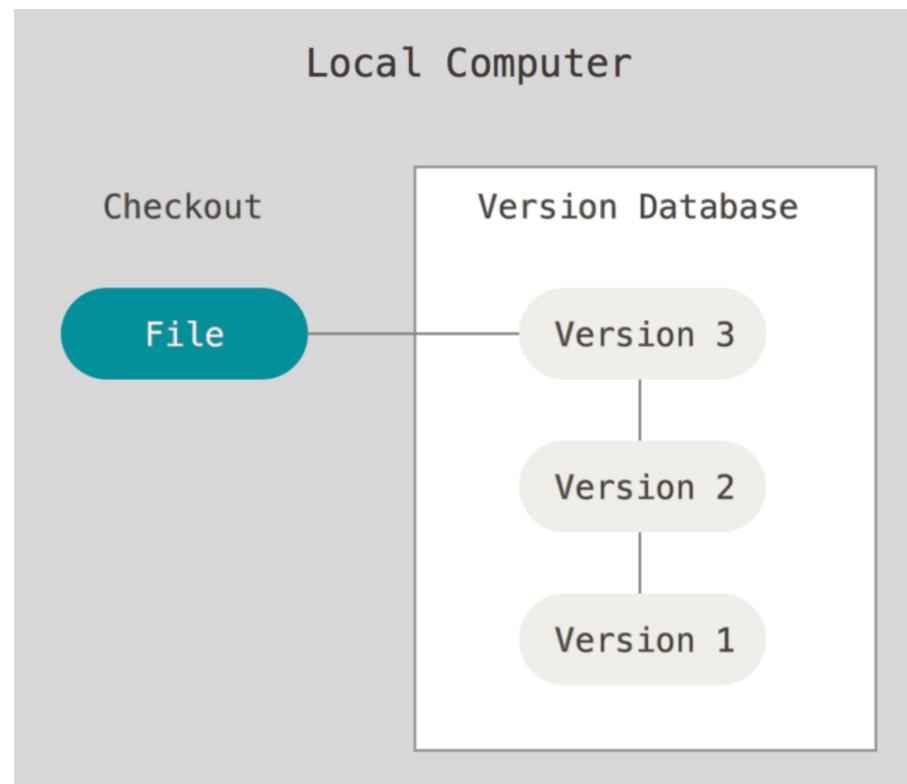
Try risk-free for 30 days using a GitHub Enterprise organization, or [learn more about changing the visibility of your GitHub Pages site.](#)

<https://{계정이름}.github.io/{저장소명}>

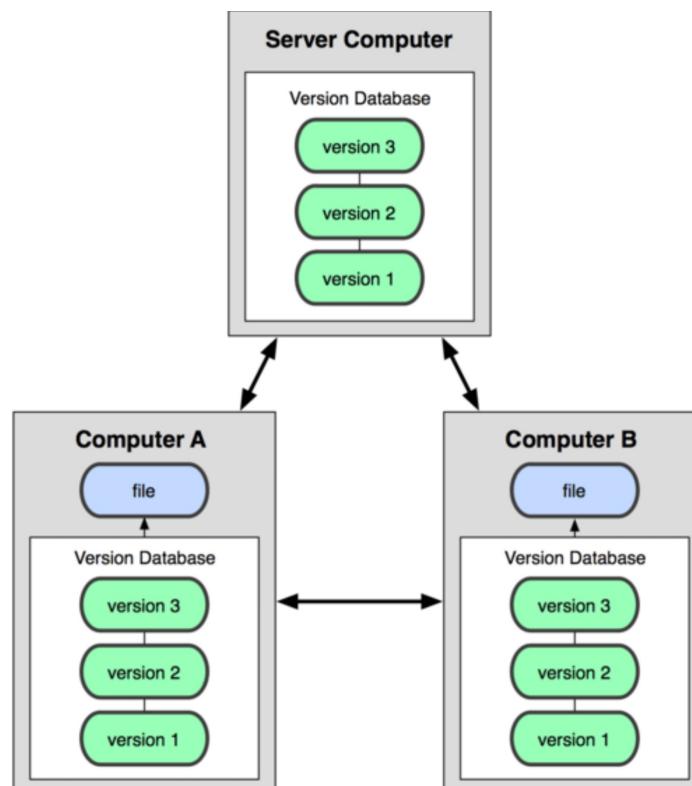
Git : 분산버전관리시스템

GitHub : Git 원격 저장소

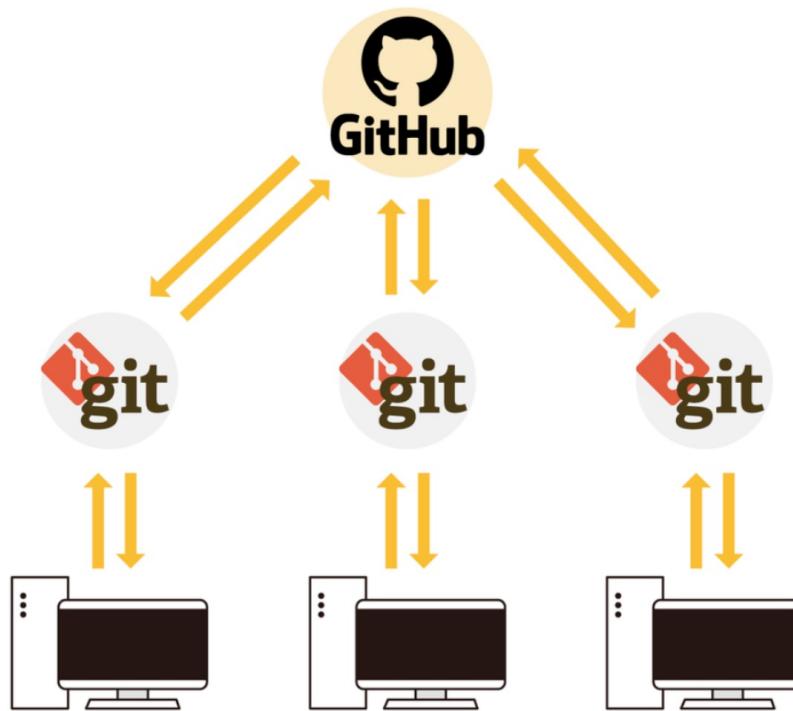
# 로컬 저장소



# 분산버전관리



# GitHub



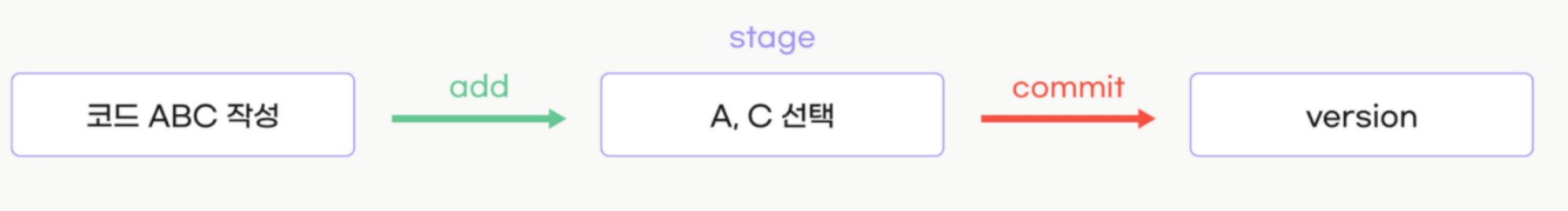
## 2. 저장소 만들고 커밋하기

- git-test 폴더를 만들고 초기화(init) 하기
- 새 파일을 만들고 add 후 commit 하기

# git add .

# git commit -m “메세지”

## ■ add



# git status

```
$ git status  
On branch master  
nothing to commit, working tree clean
```

# git diff

```
$ git diff
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory
diff --git a/README.md b/README.md
index 9c59e24..66a52ee 100644
--- a/README.md
+++ b/README.md
@@ -1 +1,2 @@
 first
+second
```

# git log

```
$ git log
commit 65d39e39bb587a32fffc1823bd8f683258f9b9f1 (HEAD -> master)
Author:
Date:   Tue Dec 7 15:08:42 2021 +0900

    first commit
```

# git log --pretty=oneline

## 커밋내용 한줄로 보기

```
7a9d697617a739b3883a160994a7b5581af86bdb (HEAD -> main, origin/main) ✓useCallback 함수 최적화
3f197d0c50c3e3c400c3c43cc0b574f4cb8745a4 ✓React.memo 렌더링 최적화
dc1745ce46432c8e495d5431a54b70ae5ab3df62 ✨List 컴포넌트 분리
d28473447dd75fc228a5985c658a9aeffd21a1e2 🔍파일명 수정
e8a541f6adaaf591f06f1f11a3b98178b80f9de0 ✨List 컴포넌트 Drag & Drop 기능 추가
701859a98292ecc5a6831e5725fff2c4d014b58b ✏Form 컴포넌트 작업
f370fc82d61e4b0fc6425bed0eb664f1b5fce7a5 ✏Form 컴포넌트 분리
f9bfce919972457dee5dc8188288a585412793c9 🔍불필요한 텍스트 삭제
```

# git log --graph

## 로그를 그래프로 보기

```
* commit a11408b247a669a7a311dabfe55cdd3117c0fd46
| \
| Merge: 666fcfbf 5865249
| Author: heyoni <hhheyoni@gmail.com>
| Date: Mon May 1 14:38:11 2023 +0900
|
|   Merge branch 'fix/#131-BE-스터디-등록-및-수정시에-스터디-제목의-글자수를-50자로-변경-필요' into dev
|
* commit 586524994150d9824df14a97861494074266150a (origin/fix/#131-BE-스터디-등록-및-수정시에-스터디-제목의-글자수를-50자로-변경-필요)
| \
| Author: heyoni <hhheyoni@gmail.com>
| Date: Mon May 1 14:31:46 2023 +0900
|
|   fix: migration 파일 추가
|
* commit 666fcfbfc06ef900cae2b9290d3fe717787de08c2
| \
| Merge: dd3898b 5803425
| Author: heyoni <hhheyoni@gmail.com>
| Date: Tue Apr 25 17:24:11 2023 +0900
|
|   Merge branch 'fix/#135-BE-스터디-종료시간이-시작시간보다-이전일-경우-완료로-표시되어-버리는-이슈' into dev
|
* commit 58034256373fd422128cf00639616b9ce297a9a5 (origin/fix/#135-BE-스터디-종료시간이-시작시간보다-이전일-경우-완료로-표시되어-버리는-이슈)
| \
| Author: heyoni <hhheyoni@gmail.com>
| Date: Tue Apr 25 17:18:36 2023 +0900
|
|   fix: 완료 표기 수정
|
|     - 종료시간 < 시작시간 (day를 고려하지 않고 24시간 기준)일 경우 종료시간에 1일을 더 해주어 해결
|     - 주석 추가
|
* commit dd3898ba90c4bbeb1ae2239cffa92b239ae070c5 (origin/feat/#96-BE-이메일-회원가입-로그인-기능)
| \
| Author: heyoni <hhheyoni@gmail.com>
| Date: Tue Apr 25 15:51:14 2023 +0900
```

# git log --graph --pretty=oneline

## 커밋내용 한줄로 표시 & 그래프 보기

```
/
| * 7f04b4cbafb4cb6fc008a0199b4197ed5ee31eb8 (origin/fix/#128-upload-image-resize, fix/#128-upload-image-resize) fix : [BE] 이 미지
|/
| * acbc682cfa17f64a497ea016051d11ede8fbf901 change production server stop script
| * 5aaa1d22728bc133b9a873090859f15cb2e639d2 Readme 수정
| * d694ca9968264c0471fc5da49418eadd50c2fc06 (origin/fix/#124-BE-스터디 조회에러-이슈, fix/#124-BE-스터디 조회에러-이슈) #124
 관심 스터디 중 마감 예정인 스터디 조회시 에러 증상 해결
| * cfc49ce018c45c6d3407e3fb8710a1eb805ea498 Merge branch 'fix/#83-BE-검색-관련-이슈' into dev
| \
| * 5cea0ec9980fb9295c3d6f99d37315f44faaad45 (origin/fix/#83-BE-검색-관련-이슈) fix: offline 검색 관련 수정
| * 313f764697062149be4fafb3de621415aebcbcd7 (origin/fix/#123-BE-Subject-관련-이슈) fix: migration dependencies 수정
| * 3b9bd85421b0f8a62883eb2db8fefd1721ec0f87 fix: subject 관련 수정
| * 867b7db824a57a551f6604b6c522737bab78b827 Merge branch 'feat/#109-BE-댓글-기능-수정' into dev
| \
| * 55a08ab5c3c7c99ee9a515e5f099f4b3ceeb14b3 (origin/feat/#109-BE-댓글-기능-수정) fix: 병합하면서 충돌된 부분 수정
| * 0fab6b7b6b638748b1a37cc329ca49a90dcc99bd Merge branch 'fix/#83-BE-검색-관련-이슈' into dev
| \
| \
| * 090ba7e7d8b1417bbfe26e34d0101c7fdb711b6b fix: comment가 비밀 댓글일 경우 에러 수정
* | 9941173e8c36b0474672068c3cb136c298a10a86 Merge branch 'feat/#109-BE-댓글-기능-수정' into dev
| \
| \
```

# .gitignore

```
$ git log
commit 65d39e39bb587a32fffc1823bd8f683258f9b9f1 (HEAD -> master)
Author:
Date:   Tue Dec 7 15:08:42 2021 +0900

    first commit
```

# .gitignore 규칙

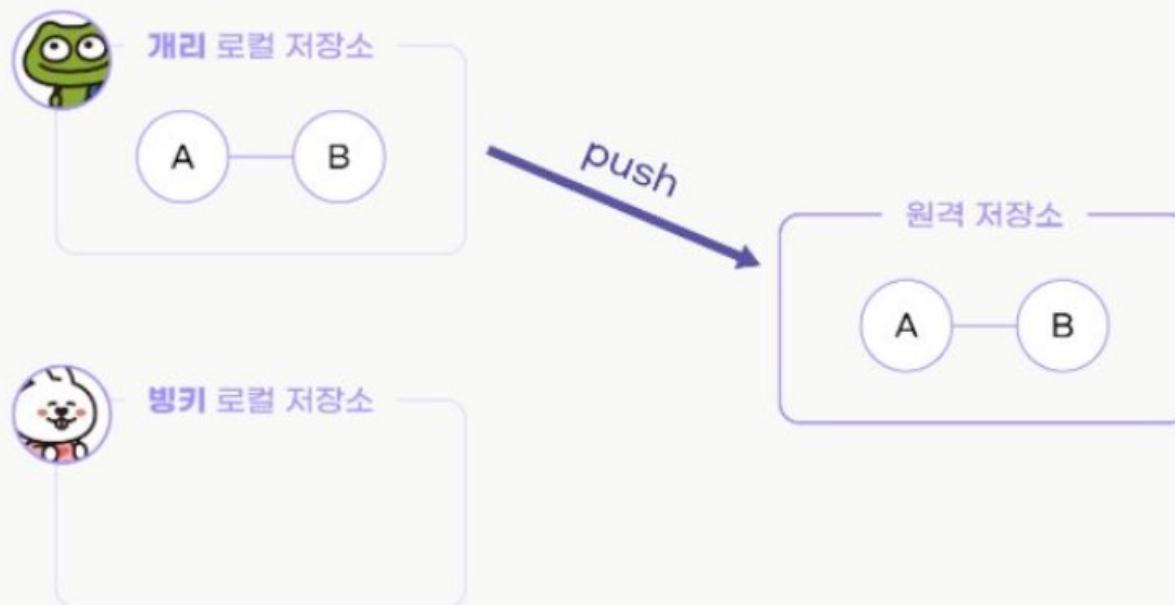
```
# a comment - 이 줄은 무시한다.  
# 확장자가 .a인 파일 무시  
*.a  
# 웃 줄에서 확장자가 .a인 파일은 무시하게 했지만 lib.a는 무시하지 않는다.  
!lib.a  
# 루트 디렉토리에 있는 TODO파일은 무시하고 subdir/TODO처럼 하위디렉토리에 있는 파일은 무시하지 않는다.  
/TODO  
# build/ 디렉토리에 있는 모든 파일은 무시한다.  
build/  
# `doc/notes.txt`같은 파일은 무시하고 doc/server/arch.txt같은 파일은 무시하지 않는다.  
doc/*.txt  
# `doc` 디렉토리 아래의 모든 .txt 파일을 무시한다.  
doc/**/*.txt
```

# git clone

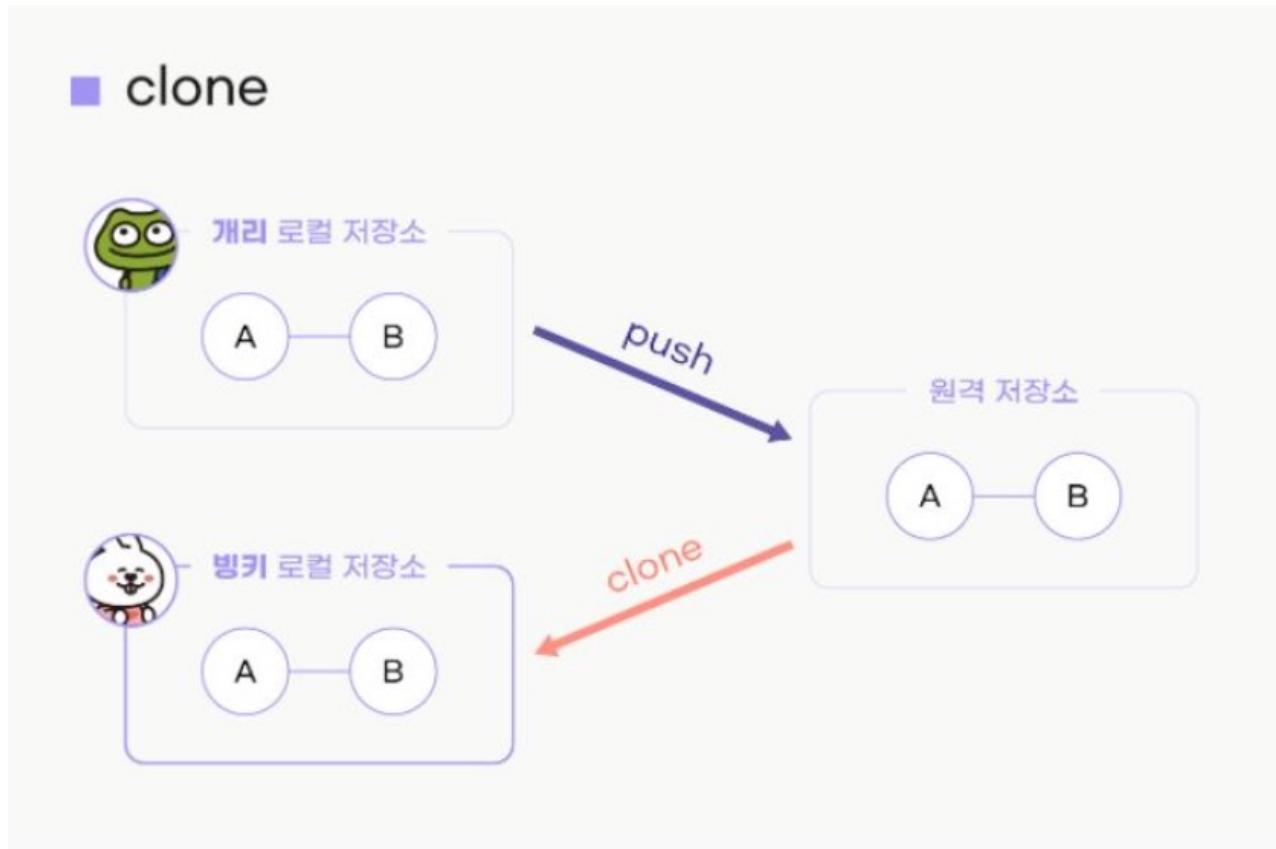
- 원격 저장소의 코드를 내 컴퓨터로 받아오는 것

# Clone 하기 전

## ■ clone



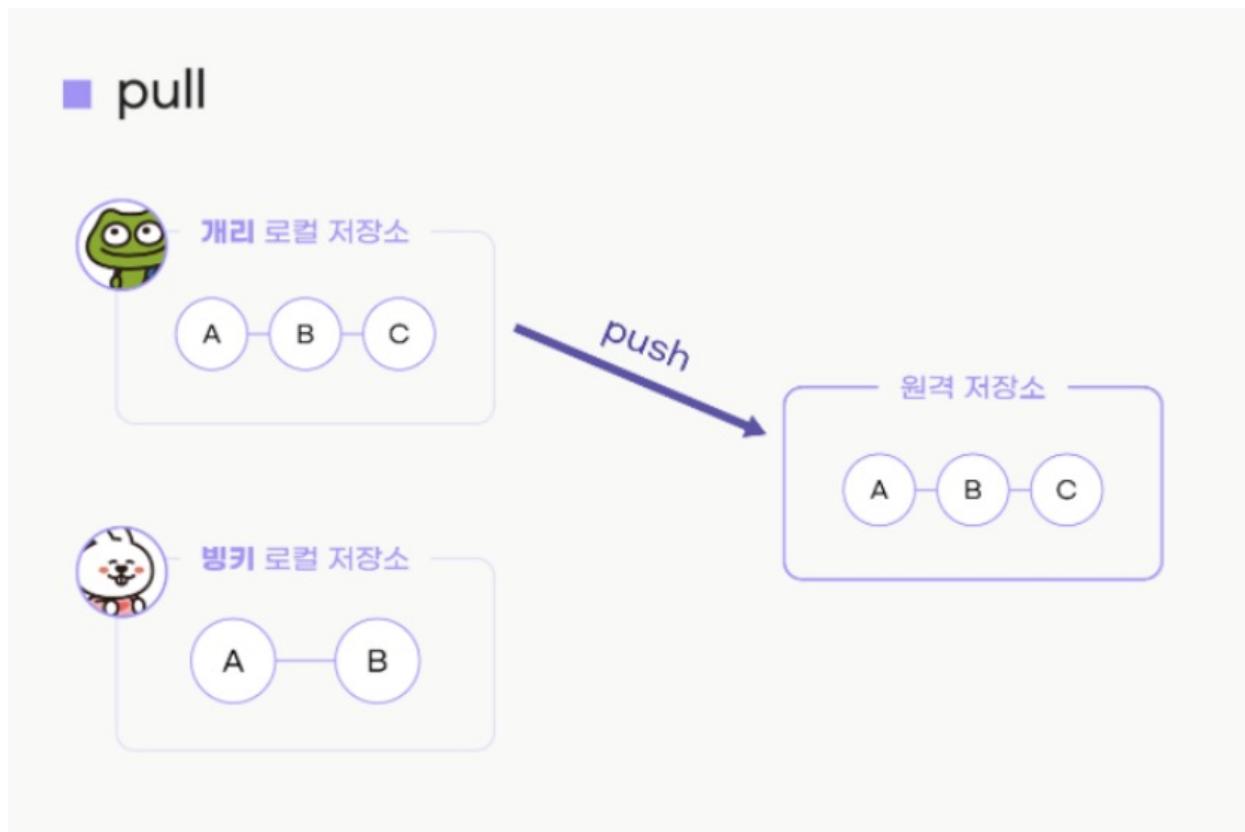
# Clone 한 후



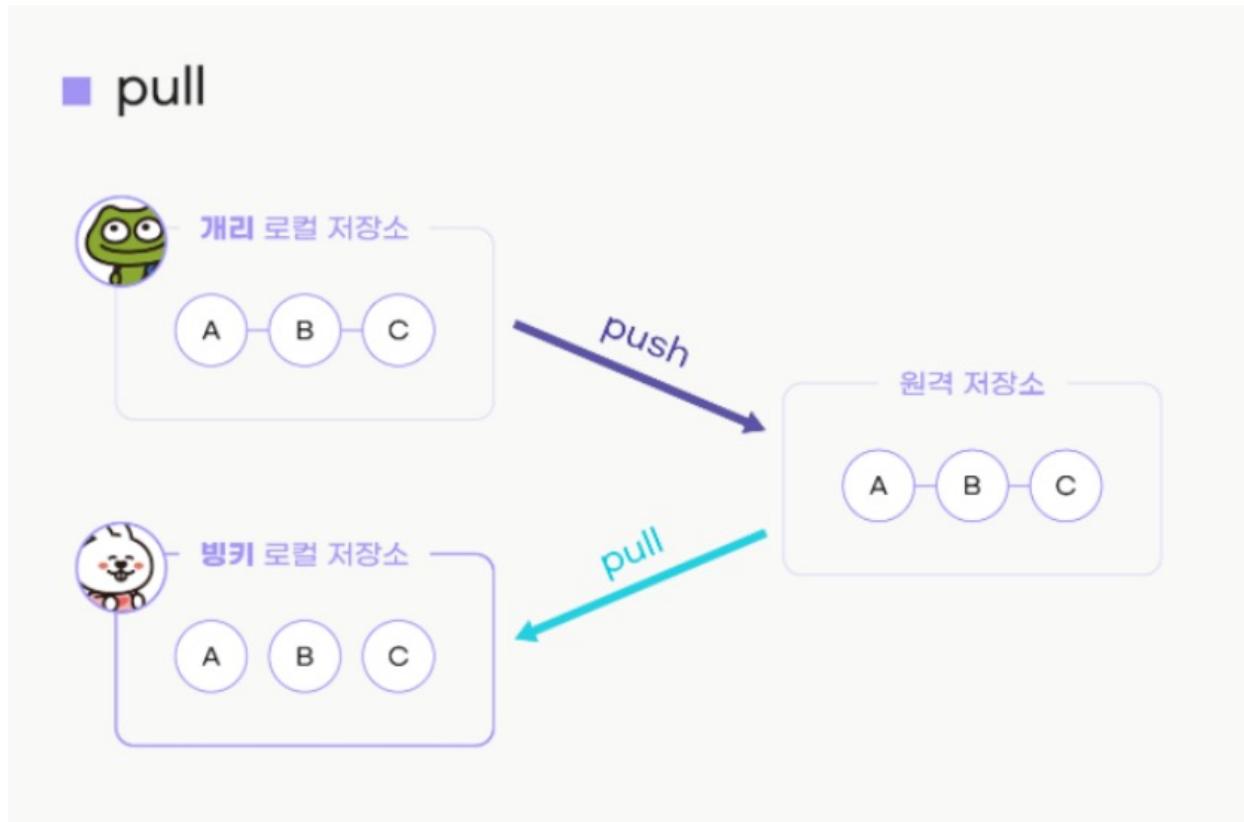
# git pull

- 원격 저장소에 업데이트 된 데이터를 가져와서 합칩니다.

# pull 하기 전



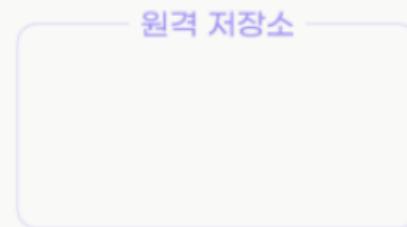
# pull 한 후



# git push

- 원격 저장소에 로컬저장소의 변경내역을 업데이트합니다.

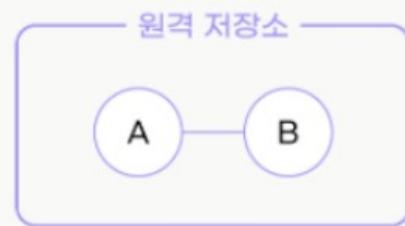
## ■ push

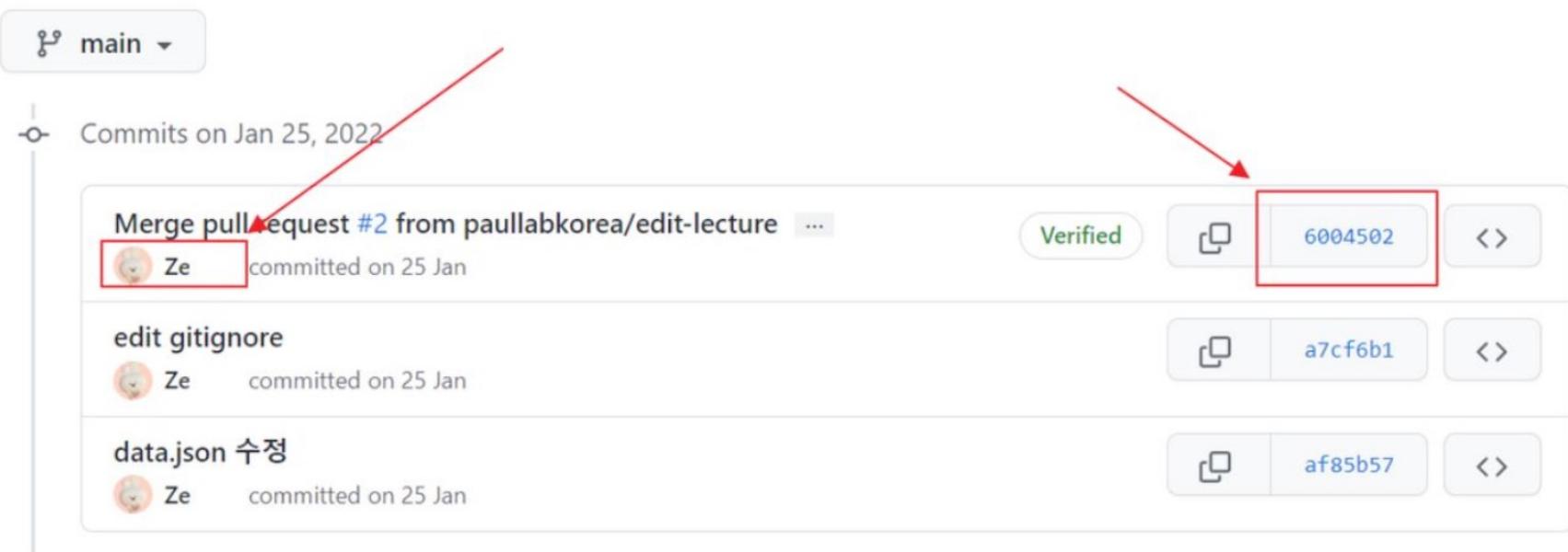
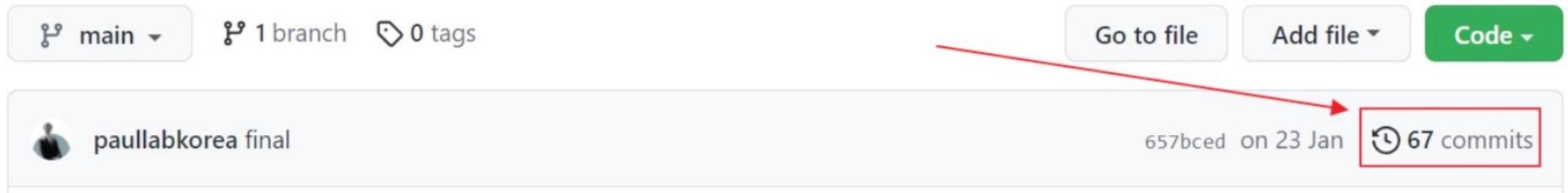


## ■ push



push





# 실습 1

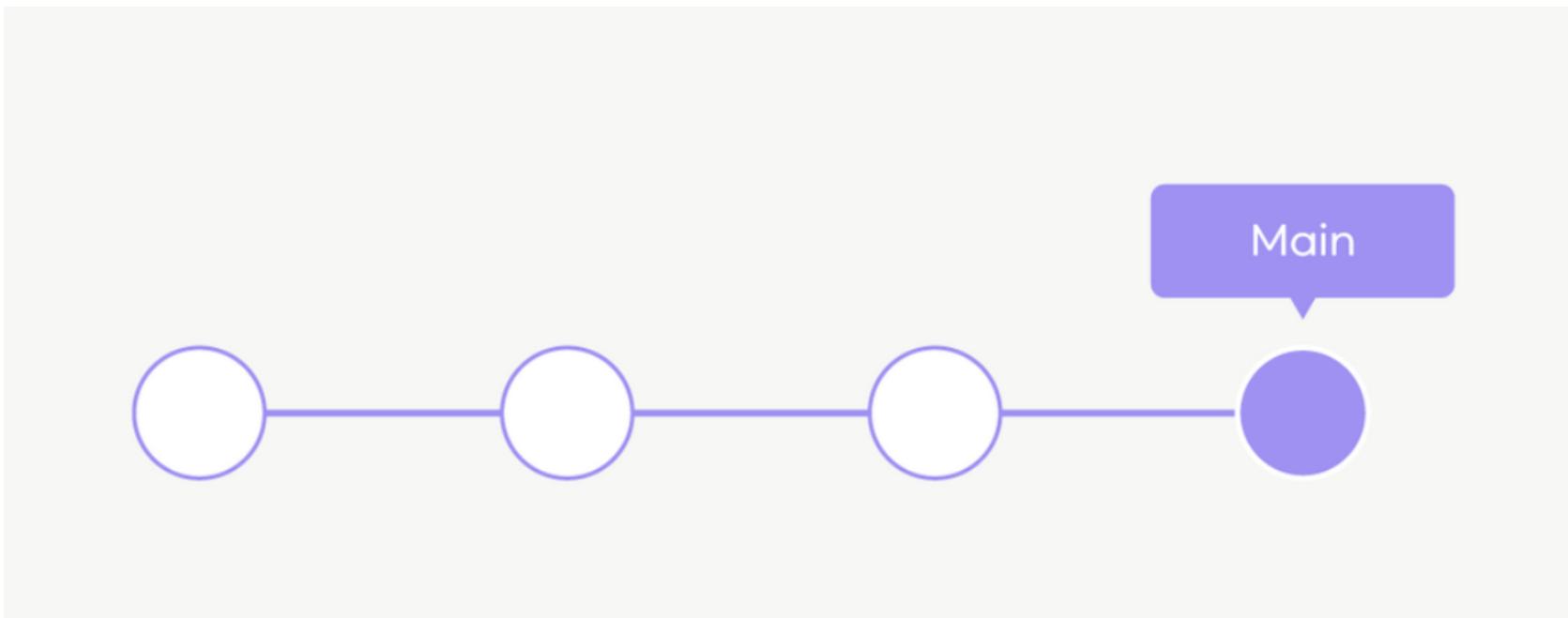
- 로컬 터미널에서 git-test-1 이라는 폴더를 만듭니다.
- index.html 파일을 생성후 내용 작성 후 add, commit 합니다.
- Github에서 git-test-1 이라는 저장소를 생성합니다.
- git-test-1 저장소에 내 코드를 push 합니다.
- Github pages를 이용해서 URL을 생성합니다.

디스코드에 공유해주세요~

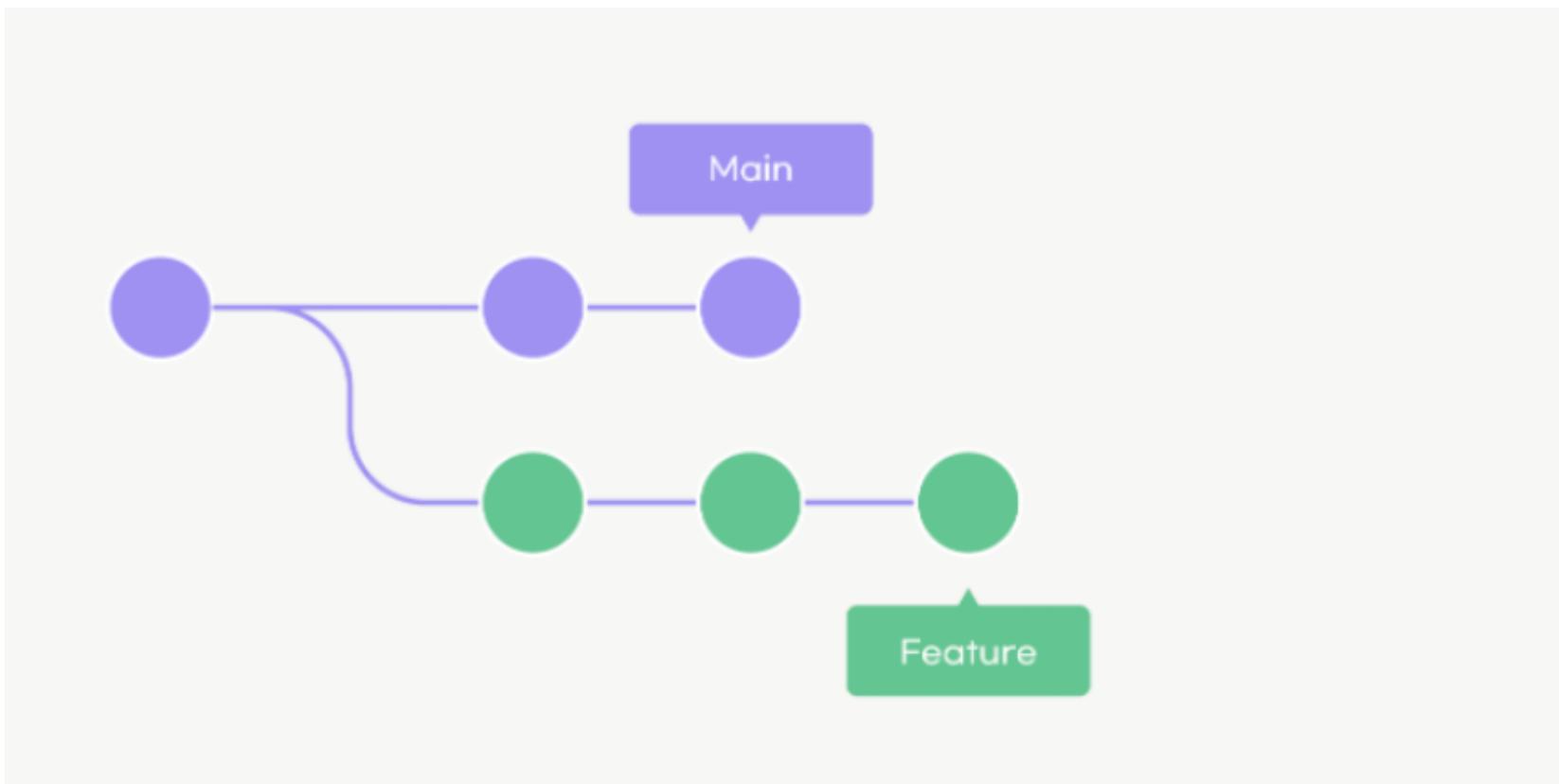
# 브랜치 (Branch)

독립적인 작업을 할 수 있는 공간

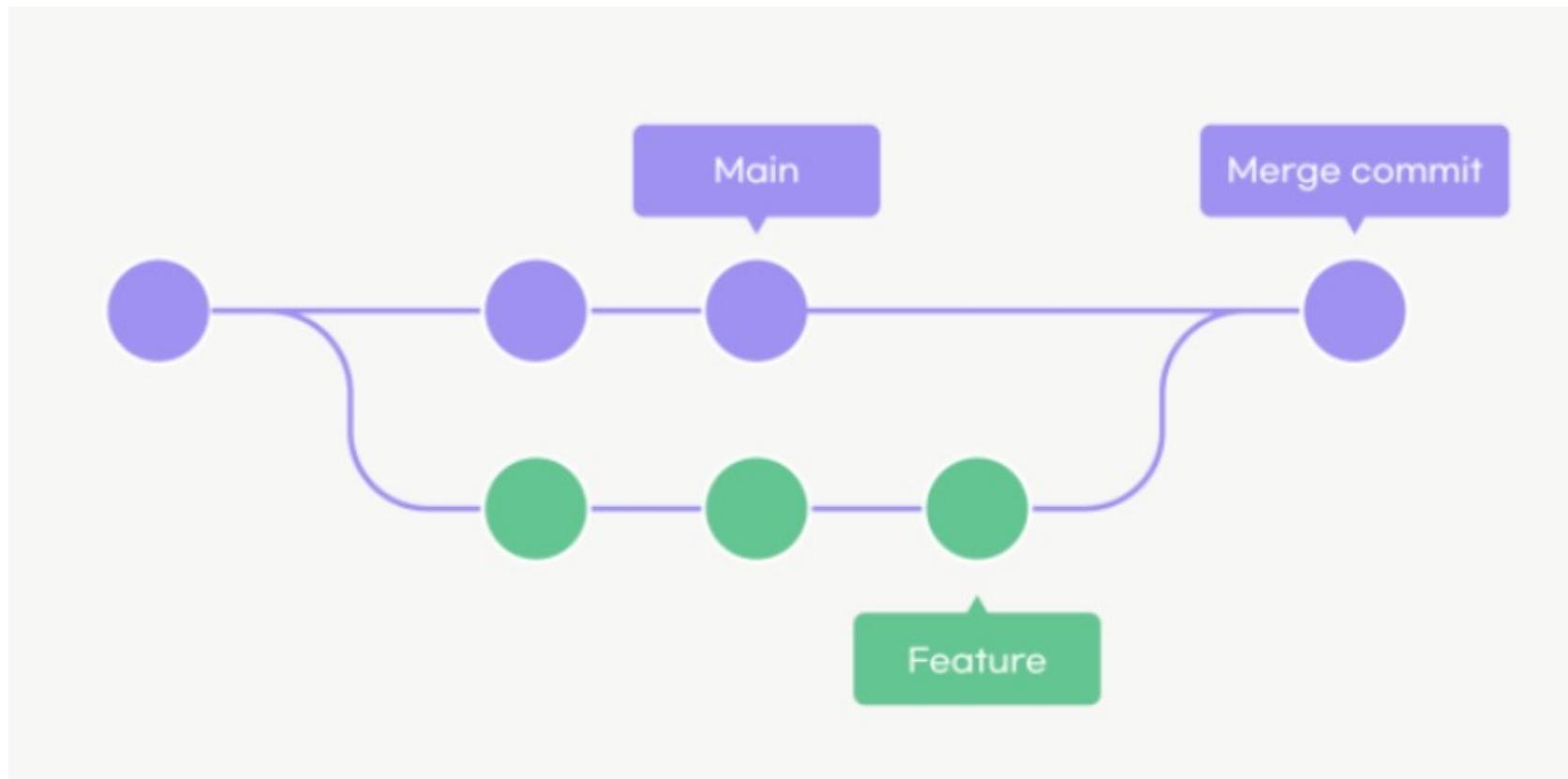
# Main 브랜치



# Feature 브랜치



# 브랜치 통합(merge)



# 현재 브랜치 목록과 현재 브랜치 확인

```
$ git branch
```



# Branch 만들기

```
$ git branch Gary
```



# Checkout – 브랜치 이동

```
$ git checkout Gary
```



checkout -b : 브랜치 새로 만들어 이동

git checkout -b <브랜치명>

# switch – 브랜치 이동

- git switch gary



# switch --c : 브랜치 새로 만들어 이동

- git switch --c gary2



# restore – 파일 수정 내용 복원

- git restore 파일명

```
$ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```



```
$ git status
On branch main
nothing to commit, working tree clean
```

# restore --staged : 스테이지 올린것 빼기

- git restore --staged 파일명

```
Changes to be committed:  
(use "git restore --staged <file>..." to unstage)  
modified:   index.html
```



```
Changes not staged for commit:  
(use "git add <file>..." to update what will be committed)  
(use "git restore <file>..." to discard changes in working directory)  
modified:   index.html  
  
no changes added to commit (use "git add" and/or "git commit -a")
```

# reset : 스테이지 올린것 빼기

- git reset HEAD 파일명

```
Changes to be committed:  
(use "git restore --staged <file>..." to unstage)  
modified:   index.html
```



```
Changes not staged for commit:  
(use "git add <file>..." to update what will be committed)  
(use "git restore <file>..." to discard changes in working directory)  
modified:   index.html  
  
no changes added to commit (use "git add" and/or "git commit -a")
```

# branch 삭제

- git branch -D <삭제할 브랜치 명>

# merge 합병하기

- git checkout main



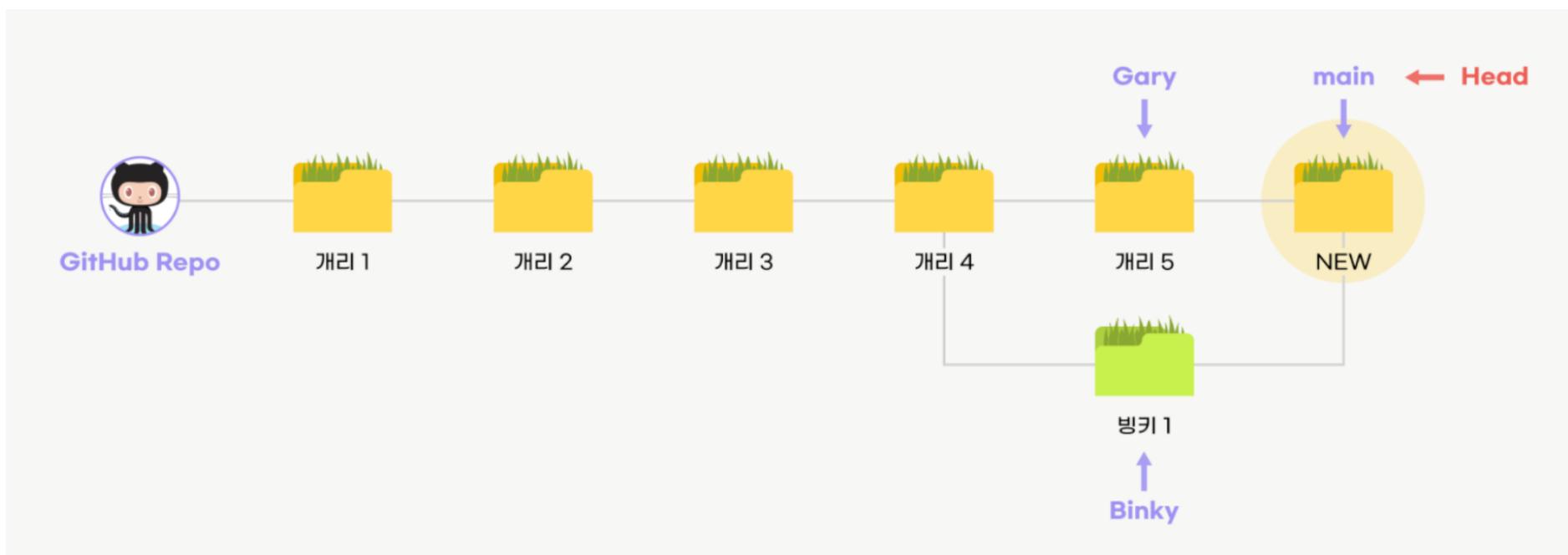
# merge 합병하기

- git merge binky



# merge 합병하기

- git merge gary



# conflict 충돌 해결하기

```
hello world  
hello Gary
```

Gary 브랜치

```
hello world  
hello binky
```

main 브랜치

```
<<<<< HEAD (Current Change)  
hello world  
hello binky  
=====  
>>>>> branchName (Incoming Change)  
hello world  
hello Gary
```

# conflict 충돌 해결하기

conflict는 두가지 변경 내역을 비교한 다음에 원하는 코드를 남기는 것입니다.

```
hello world  
hello binky Gary
```

이후 **add > commit > push** 를 해주시면 됩니다.

# Pull Request

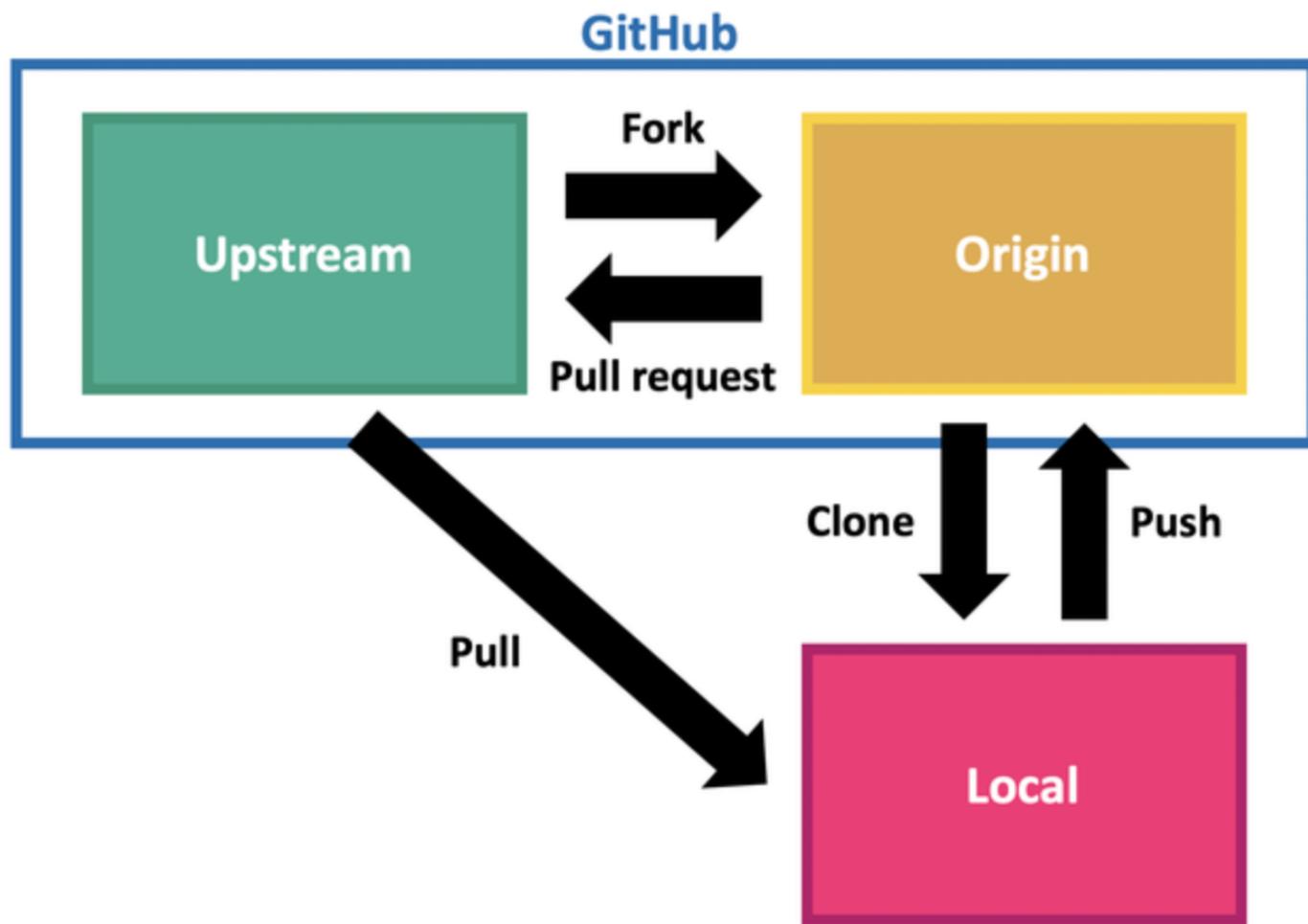
- 브랜치 생성 후 커밋하기
- 푸시 하기
- Pull Request 생성하기
- 코드리뷰 하기
- Pull Request 머지하기

# 실습 2 - 브랜치에서 작업후 Pull Request

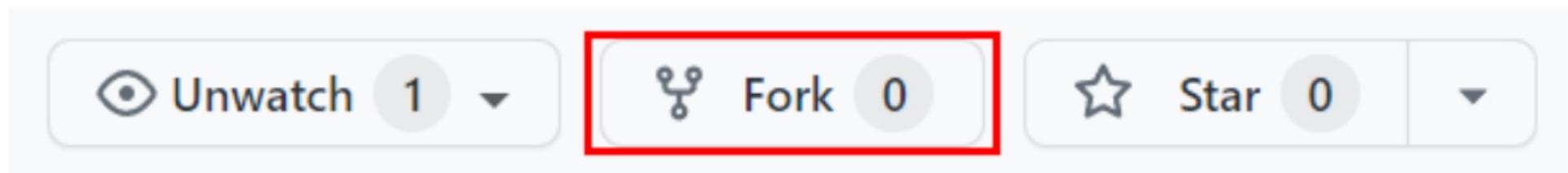
- 실습1에서 작업한 git-test-1 디렉토리에서 develop 브랜치 생성 후 이동
- index.html 파일 수정후 커밋 & 푸시(develop branch)
- Pull Request 생성하기
- Pull Request merge 하기

# Fork



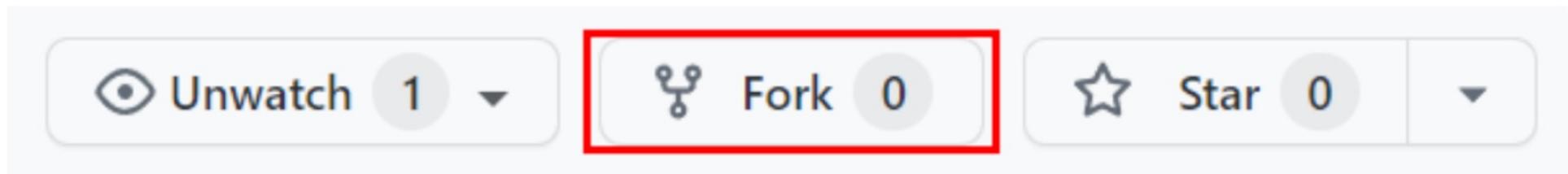


<https://github.com/beomjae/githubTest>



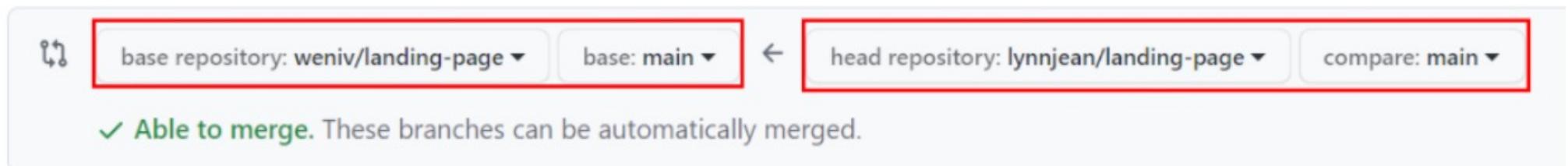
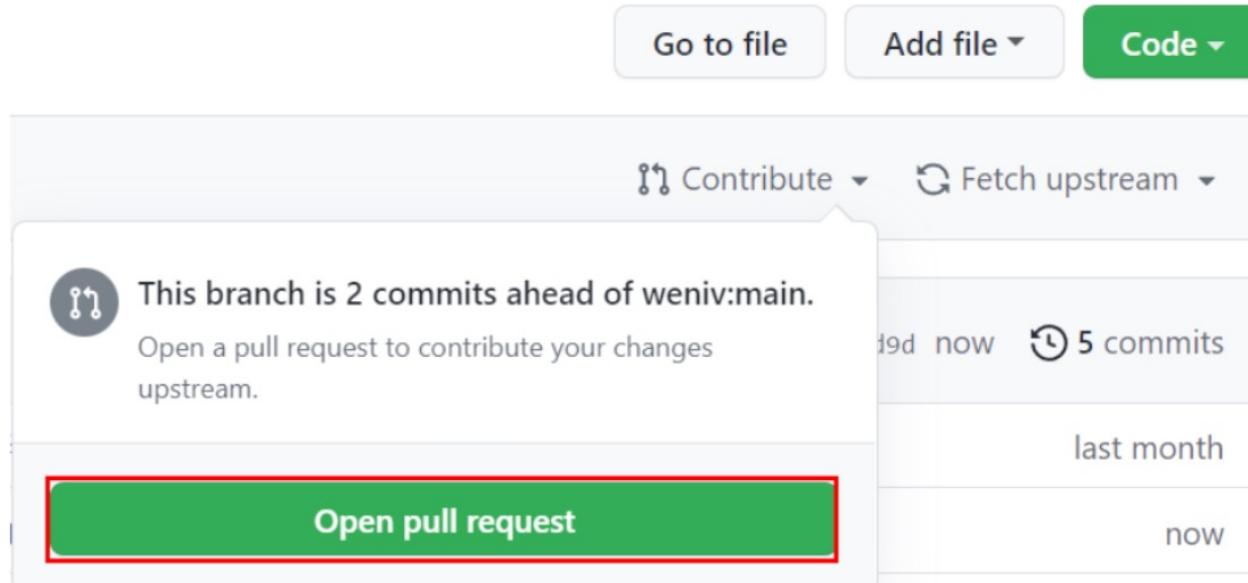
github에서 파일 수정 후 pull request 보내기

<https://github.com/beomjae/githubTest>



- git clone <fork한 새 저장소>
- 내ID.html 파일 생성후 커밋 & 푸시

# Pull Request 보내기



# Pull Request Merge

This branch has conflicts that must be resolved

Use the [web editor](#) or the [command line](#) to resolve conflicts.

**Conflicting files**

routes/users.js

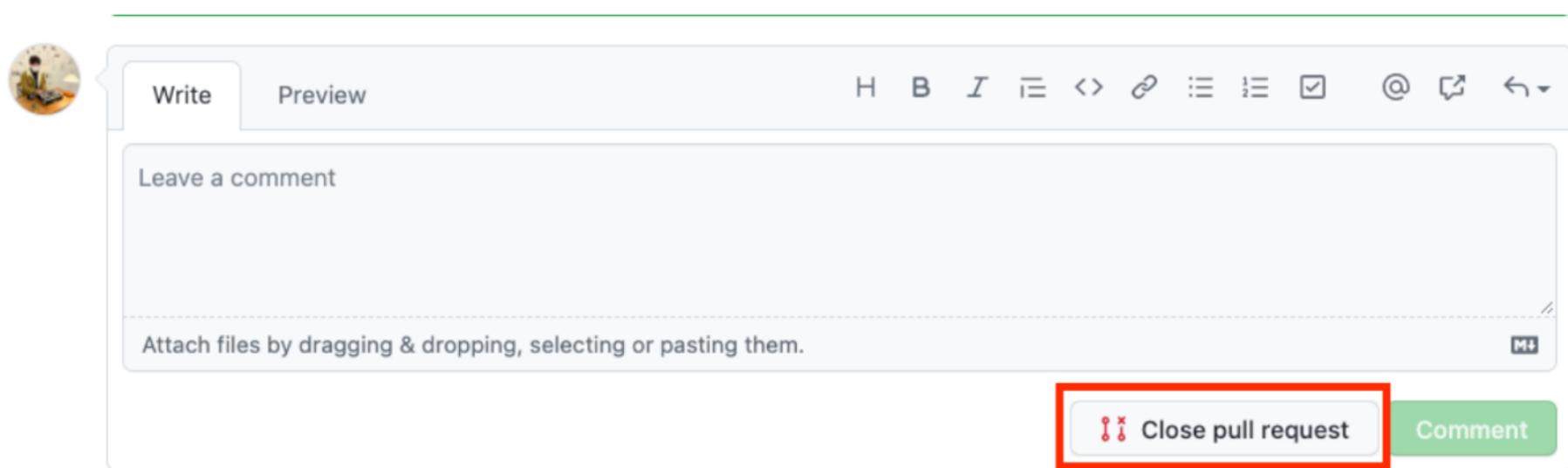
**Merge pull request**

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

**Resolve conflicts**

The screenshot shows a GitHub pull request merge interface. At the top left is a circular icon with a gear and two lines. To its right, a warning icon (a triangle with an exclamation mark) is followed by the text "This branch has conflicts that must be resolved". Below this, instructions say "Use the [web editor](#) or the [command line](#) to resolve conflicts.". A section titled "Conflicting files" lists "routes/users.js". At the bottom left is a red-bordered button labeled "Merge pull request". To its right is a dropdown arrow. At the bottom right is a link to "GitHub Desktop" and another link for "command line instructions". A "Resolve conflicts" button is located in the top right corner of the main message area.

# Pull Request 취소하기



# GitHub Repository 권한 부여

Code Issues Pull requests Actions Projects Security Insights **Settings**

Options

**Collaborators**

Security & analysis  
Branches  
Webhooks  
Notifications  
Integrations  
Deploy keys  
Actions  
Secrets  
Pages

Who has access

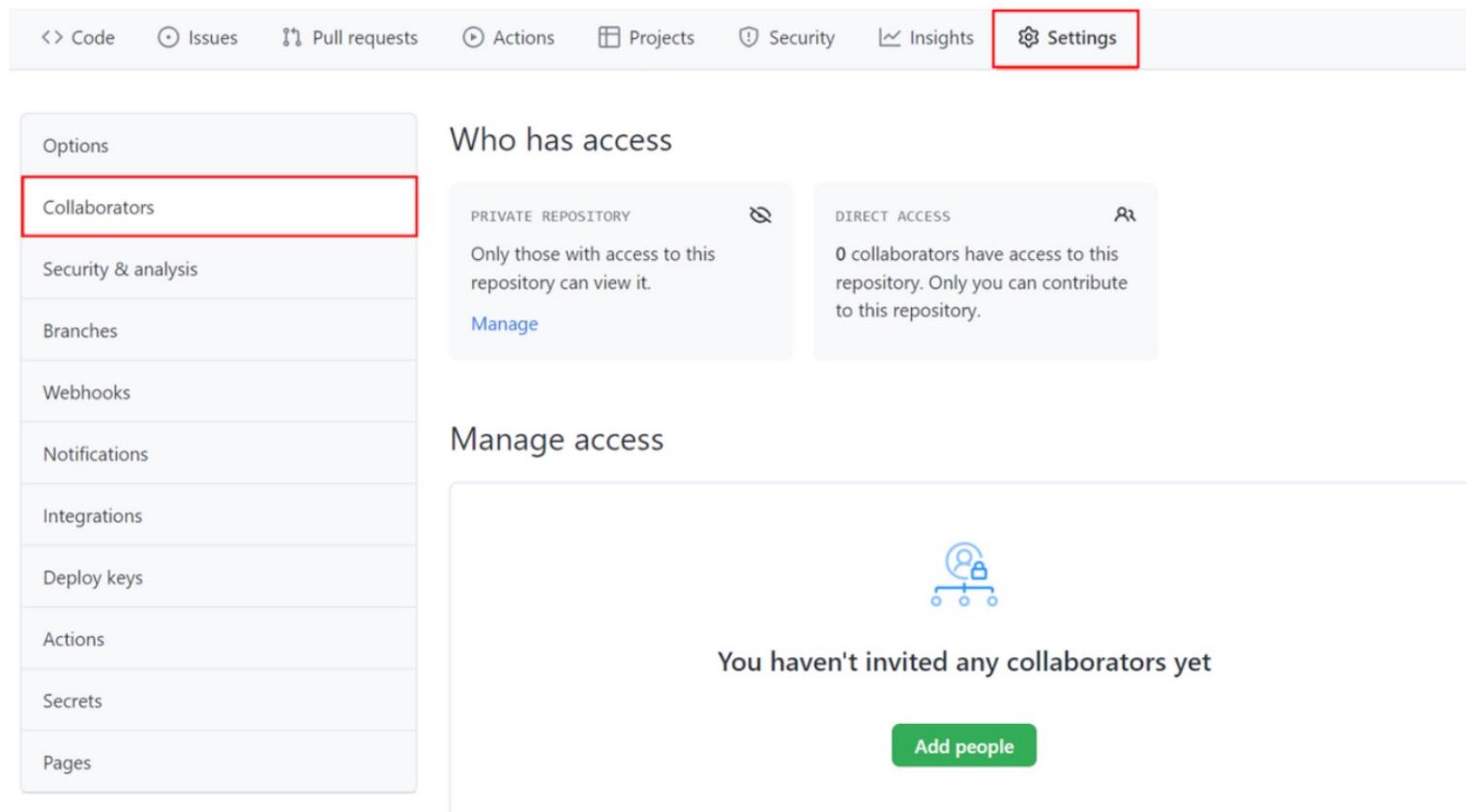
PRIVATE REPOSITORY 🔒 Only those with access to this repository can view it. [Manage](#)

DIRECT ACCESS 🌐 0 collaborators have access to this repository. Only you can contribute to this repository.

Manage access

You haven't invited any collaborators yet

Add people

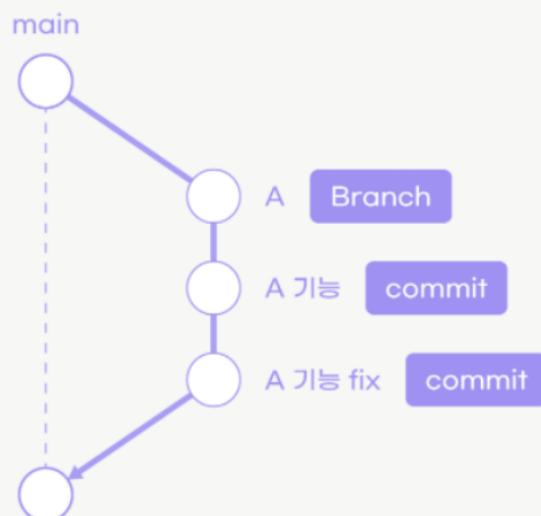
A screenshot of the GitHub repository settings page. The 'Settings' tab is selected. On the left, a sidebar lists options: Collaborators (highlighted with a red box), Security & analysis, Branches, Webhooks, Notifications, Integrations, Deploy keys, Actions, Secrets, and Pages. The main area shows 'Who has access' with two sections: 'PRIVATE REPOSITORY' (locked padlock icon) and 'DIRECT ACCESS' (people icon). Below this is a 'Manage access' section with a network icon and the message 'You haven't invited any collaborators yet'. A green 'Add people' button is at the bottom. The entire screenshot is framed by a thin gray border.

# 실습 3 - fork 후 Pull Request

- <https://github.com/beomjae/githubTest> 를 fork 한다
- git clone <fork한 새 저장소>
- 내ID.html 파일 생성후 커밋 & 푸시 (main branch)
- Pull Request 생성하기

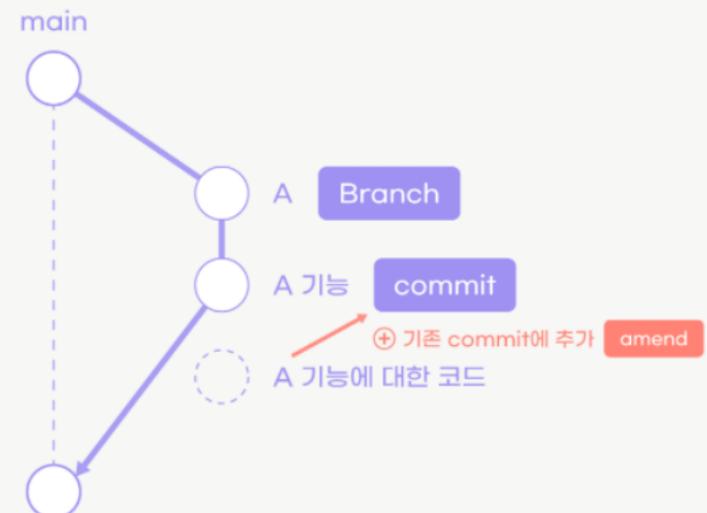
# amend - 마지막 커밋 수정하기

## ■ amend



commit 한 경우

## ■ amend



amend한 경우

# amend - 마지막 커밋 수정하기

- git commit --amend

# stash - 변경사항 임시 보관

- git stash
- git stash pop

## stash --all : 모든 파일 변경사항 임시 보관

- touch test.txt
- git stash --all
- git stash pop

# reset

- 이전 커밋으로 브랜치를 되돌릴 때 사용

## reset --hard

- 지정한 커밋 이력 이후 변경사항 다 버리고 지정한 커밋으로 리셋
- git reset --hard commit-id

## reset --mixed

- 지정한 커밋 이력 이후 변경사항은 로컬에 unstaged 상태로 유지하고 커밋은 리셋
- git reset --mixed commit-id

## reset --soft

- 지정한 커밋 이력 이후 변경사항은 로컬에 stage 상태로 유지하고 커밋은 리셋
- git reset --soft commit-id

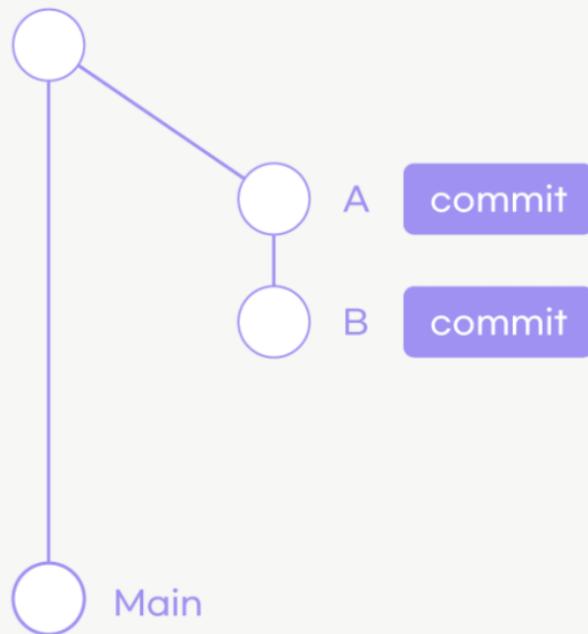
# revert

- 지정한 커밋을 되돌리는 커밋을 새로 생성
- `git revert <commit-id>`

# cherry-pick

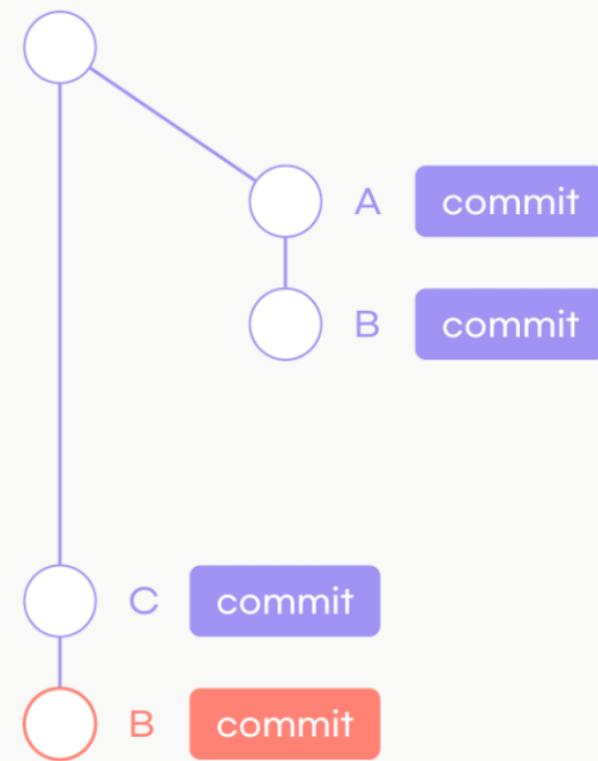
- 내가 원하는 커밋만 가져와서 현재 브랜치에 붙이기
- git cherry-pick <commit-id>

## ■ cherry-pick



cherry-pick 하기 전

## ■ cherry-pick



cherry-pick 한 후

# GitHub로 협업하기

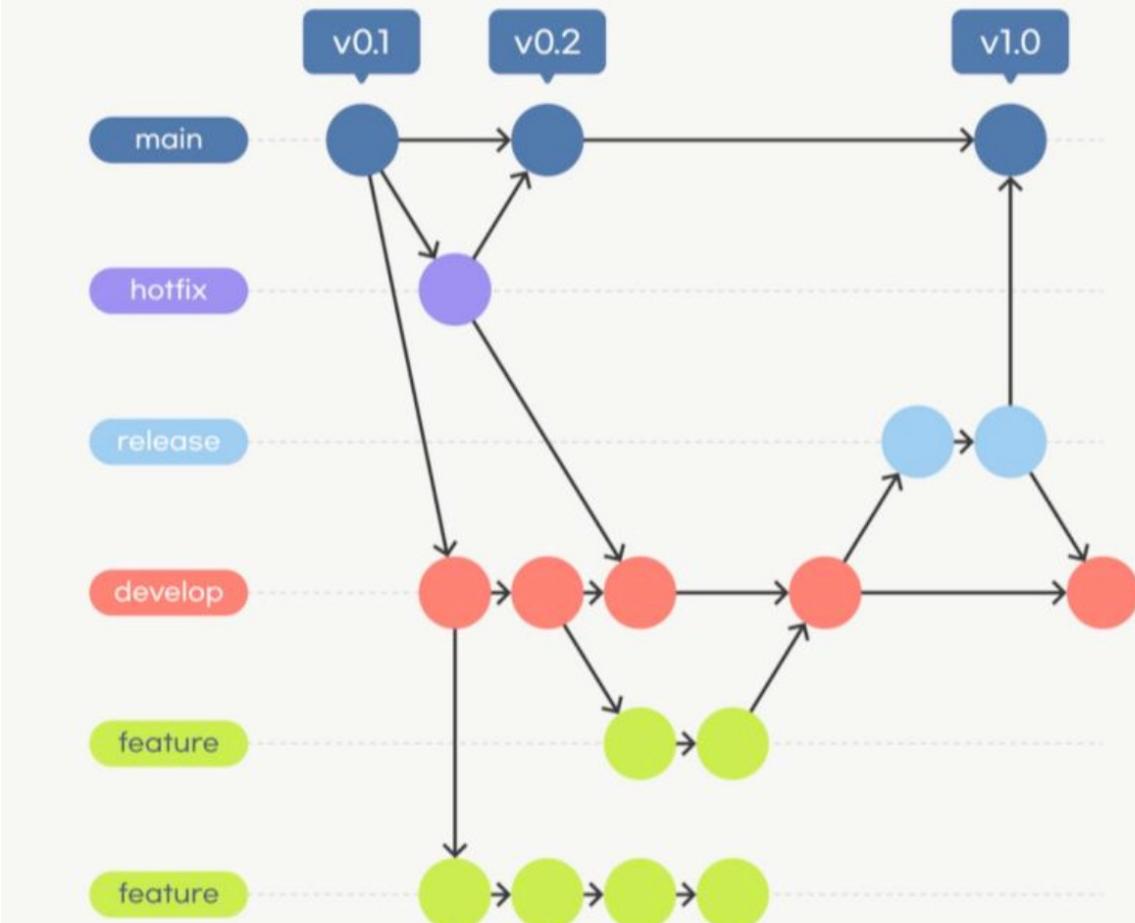
# Branch 전략

- 프로젝트 진행 시 효율적인 버전 관리를 위해 사용하는 워크플로우(work-flow)

# Git-flow

- 대규모 프로젝트에서 사용할 때 큰 장점
- feature, develop, release, main(master), hotfix

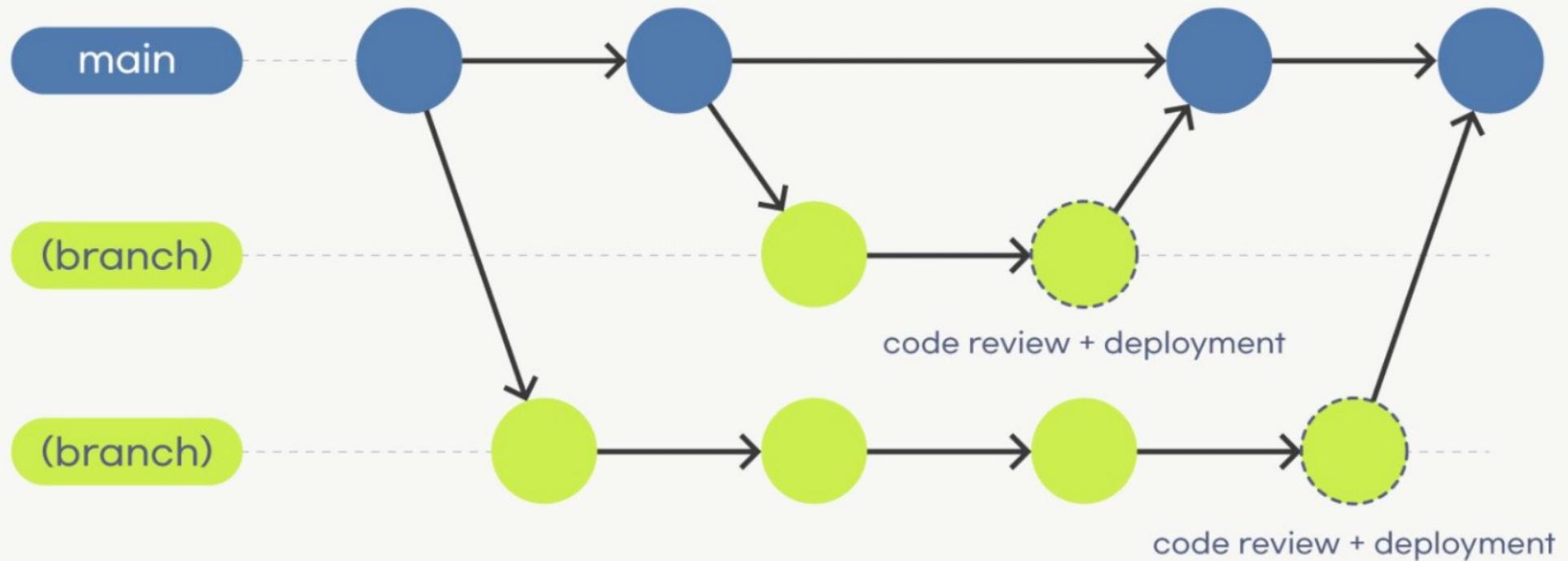
# Git Flow



# GitHub-flow

- main 브랜치를 중심으로 기능 개발을 위한 브랜치를 생성하여 진행합니다.

# GitHub Flow



# GitHub Projects

- 팀프로젝트의 여러 작업의 진행상황을 한번에 관리 할수 있는 곳

# GitHub Projects

- 팀프로젝트의 여러 작업의 진행상황을 한번에 관리 할수 있는 곳
- Github Project 생성해보기

# Github Issues

- Github에서 작업하는 프로젝트의 다양한 이슈 관리 기능
- 이슈
  - 버그
  - 개선사항
  - 새로운 기능
  - 아이디어

# 이슈 템플릿 활용하기

- Features - Issues - Set up templates

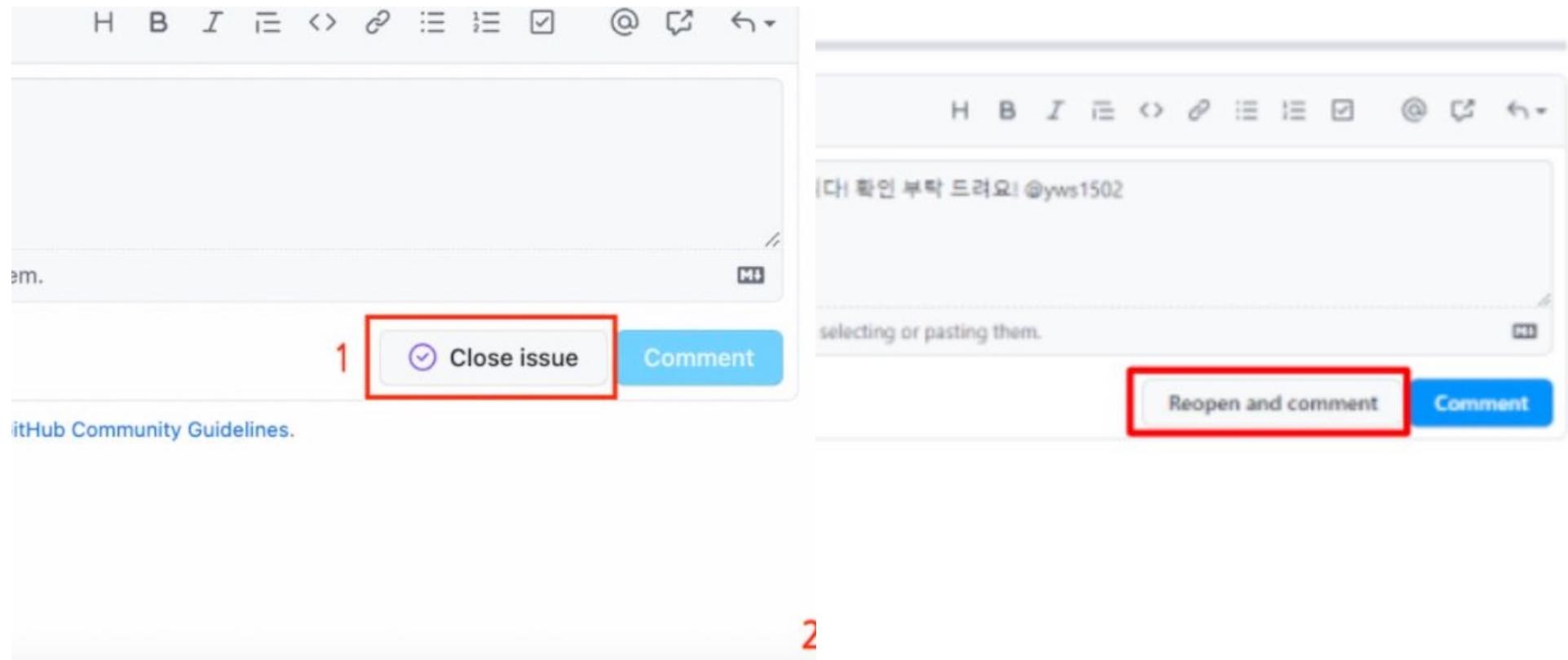
# 이슈 활용하기

- assignees : 이슈 담당자
- label : 이슈 구분
- project : 이슈의 프로젝트
- milestones : 마일스톤

## 커밋과 Issue 연결

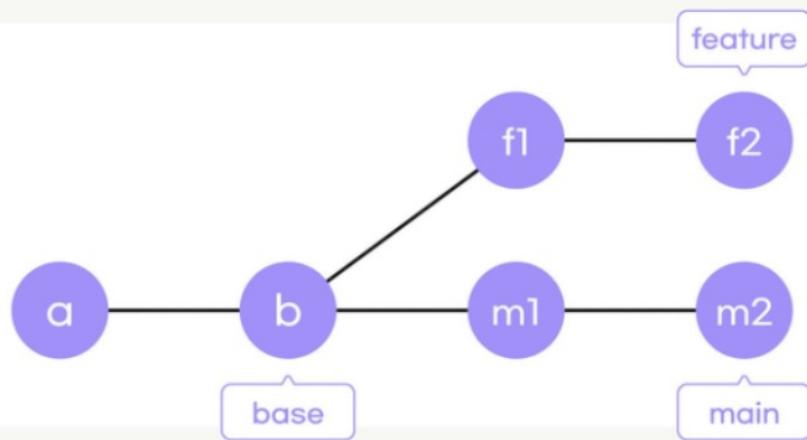
- 커밋 메시지 입력시 이슈번호를 입력하면 자동으로 연결되어 표시가 됩니다.

# 이슈 종료하기, 삭제하기, 재오픈 하기

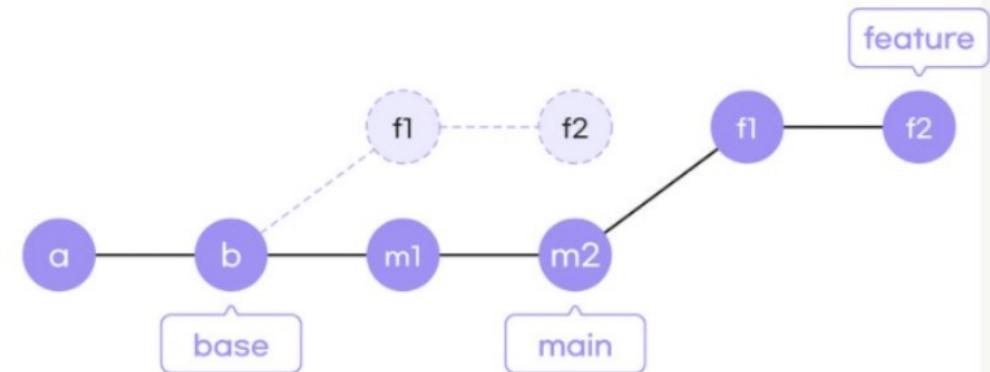


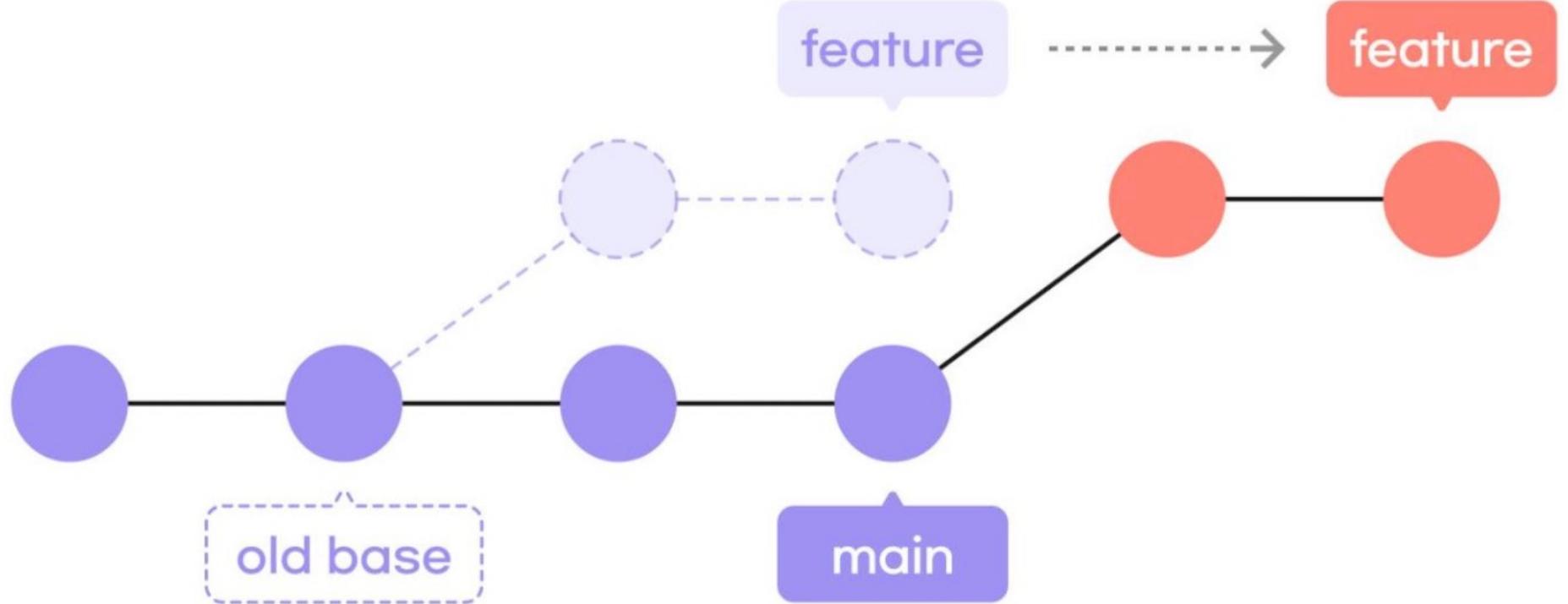
# git rebase

## ■ rebase 하기 전



## ■ rebase 된 후





# git rebase

- git checkout feature
- \$(feature) git rebase main
- git checkout main
- \$(main) git merge feature

# 부록

Cloudflare pages에  
배포하기

# github codespace 사용하기

github copilot

# git 연습사이트

<https://learngitbranching.js.org/?locale=ko>