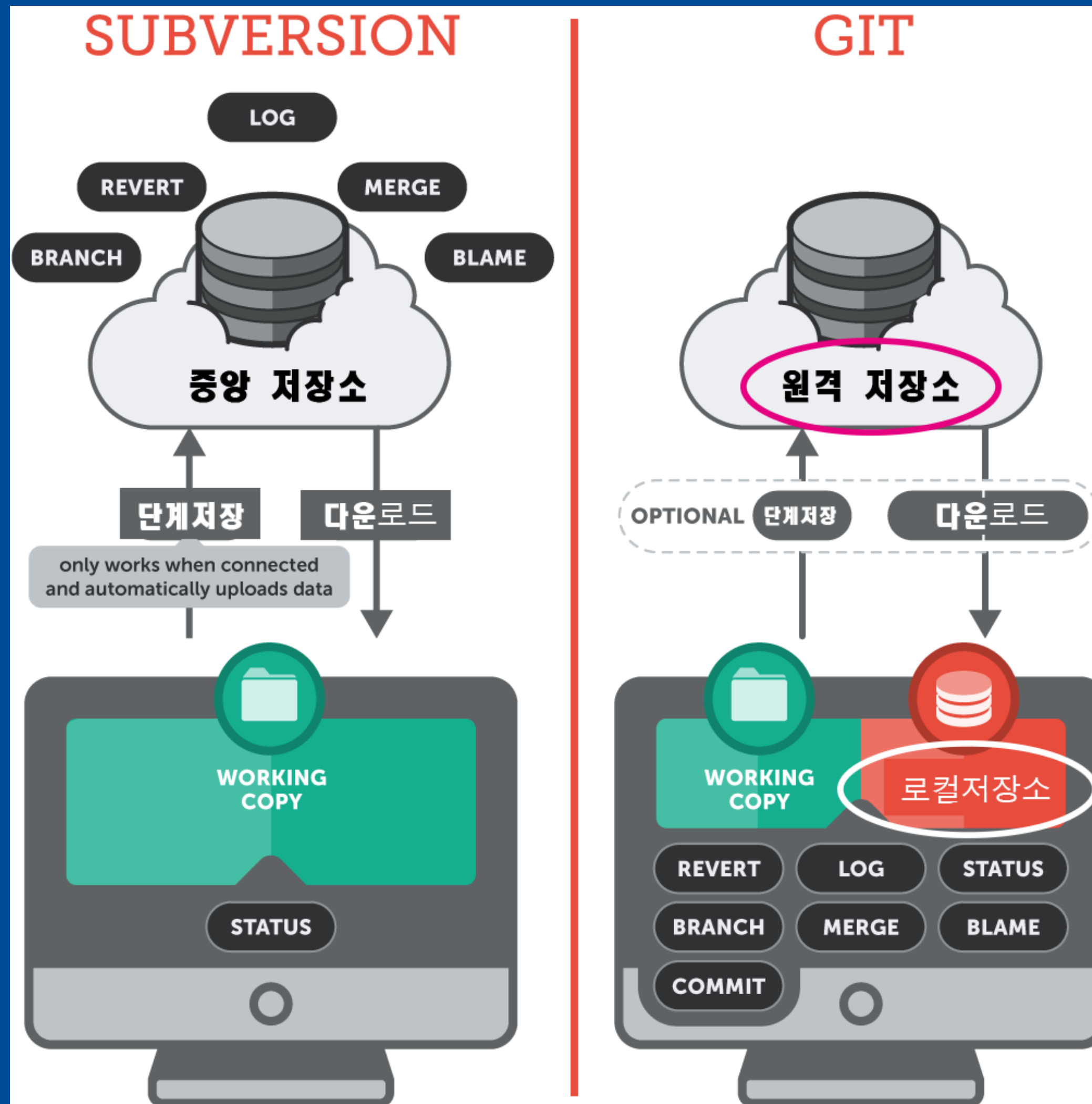




# git 기초




9월 19일

# 분산



- 분산 저장으로 안정적

# 버전

이름	수정일
 최종	오늘 오후 5:23
 찌막	오늘 오후 5:24
 찌찌막	오늘 오후 5:24

- 우리가 과제할 때 자주 보는 모습

# 버전

## 2.81. 9.8.5

2022년 6월 30일 릴리즈<sup>[16]</sup><sup>[17]</sup>

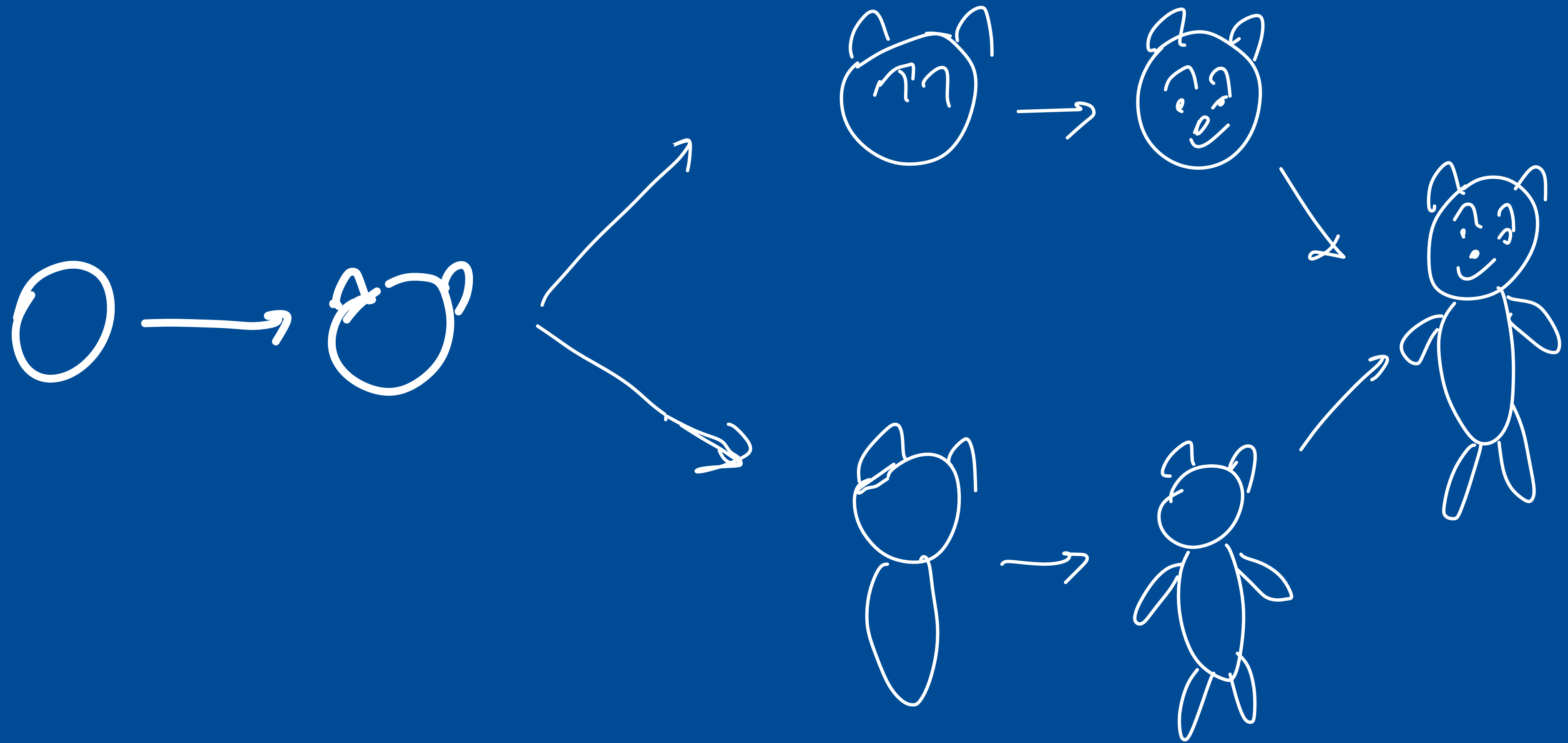
- 실험실에 그룹채팅방 참여 설정 기능 추가
  - 친구가 아닌 상대방이 나를 그룹채팅방에 초대하면 참여 여부를 선택할 수 있어요.
- 보이스룸 서비스 개선
  - 이제 누구나 오픈채팅에서 보이스룸을 만들 수 있어요.
  - 음성필터를 활용해 재미있는 목소리로 대화해보세요.
- 복사 기능 개선
  - 말풍선을 길게 눌러 대화내용을 선택 복사 할 수 있어요.
- 샵검색 추천 검색어 바로 공유 기능 추가
  - 추천 검색어의 검색 결과를 채팅방에 바로 공유해보세요.

## 2.82. 9.9.0 <sup>[18]</sup>

2022.8.8 릴리즈

- 오픈채팅 홈 아이콘 추가
  - 채팅목록 상단에 추가된 아이콘으로 오픈채팅 홈에 바로 진입할 수 있어요.
- 내 프로필 QR 개선
  - 친구추가 화면에서 내 프로필 QR을 확인하고 카카오프렌즈로 꾸며보세요.
- 오픈채팅 송금기능 추가
  - 오픈채팅방에서 상대방의 프로필을 선택해 송금할 수 있어요.
- 이미지 편집기 개선
  - 편집 도구에서 화살표를 그리거나 글자를 입력할 수 있어요,
  - 친구에게 받을 사진을 편집 후 공유해 보세요.
- 보이스톡 공감 아이템 추가
  - 공감 아이템 설정하여 감정을 표현해 보세요.
- 이모티콘 즐겨찾기 수 확대
  - 최대 120개의 이모티콘을 즐겨찾기 할 수 있어요.
- 채팅방 상단 공지 아이콘이 파란색 라인아트로 바뀌었으며, 노란색 투표 아이콘도 추
- 설정 메뉴 레이아웃 변경

# 버전

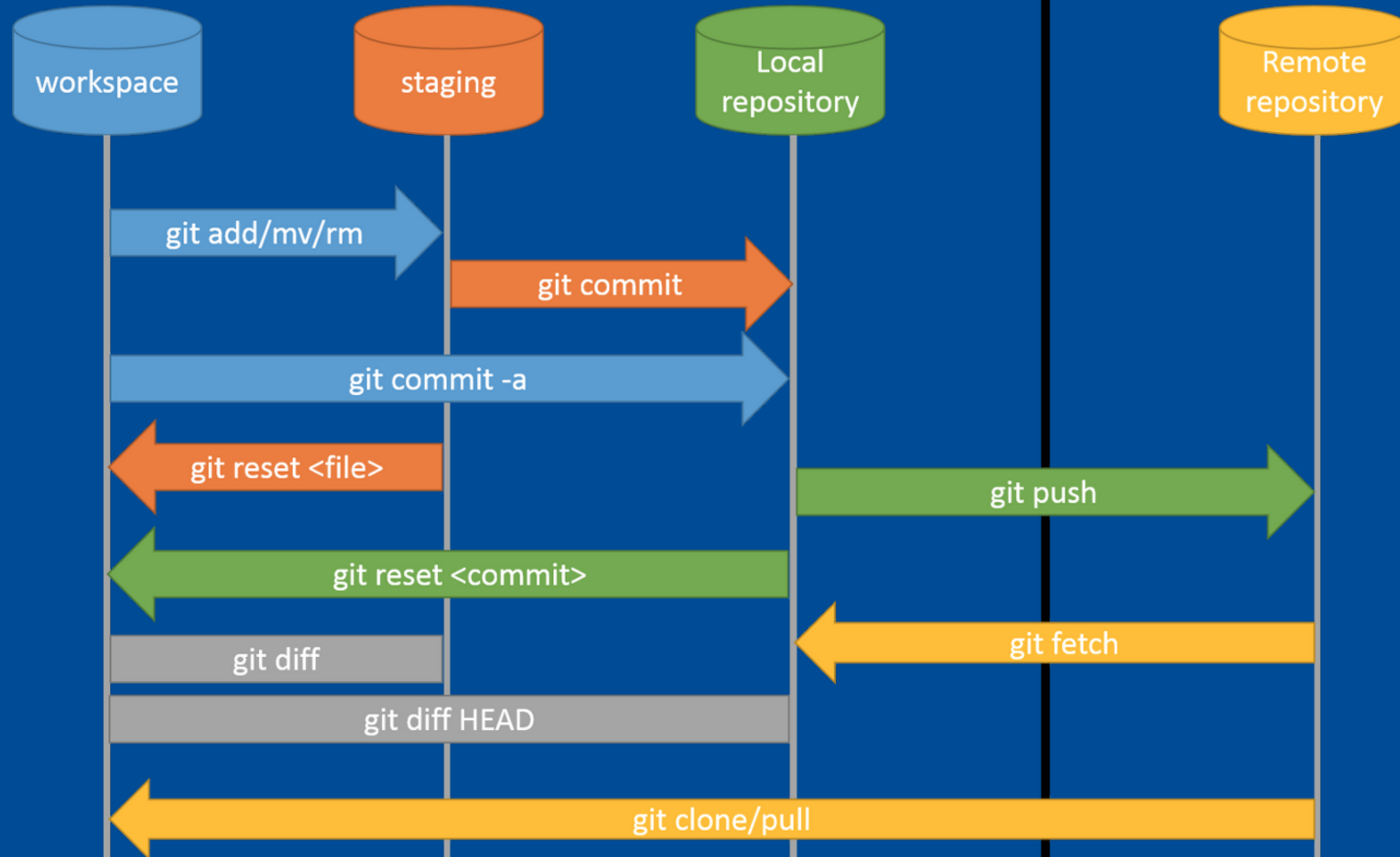


# Git 원격 저장소 (github)



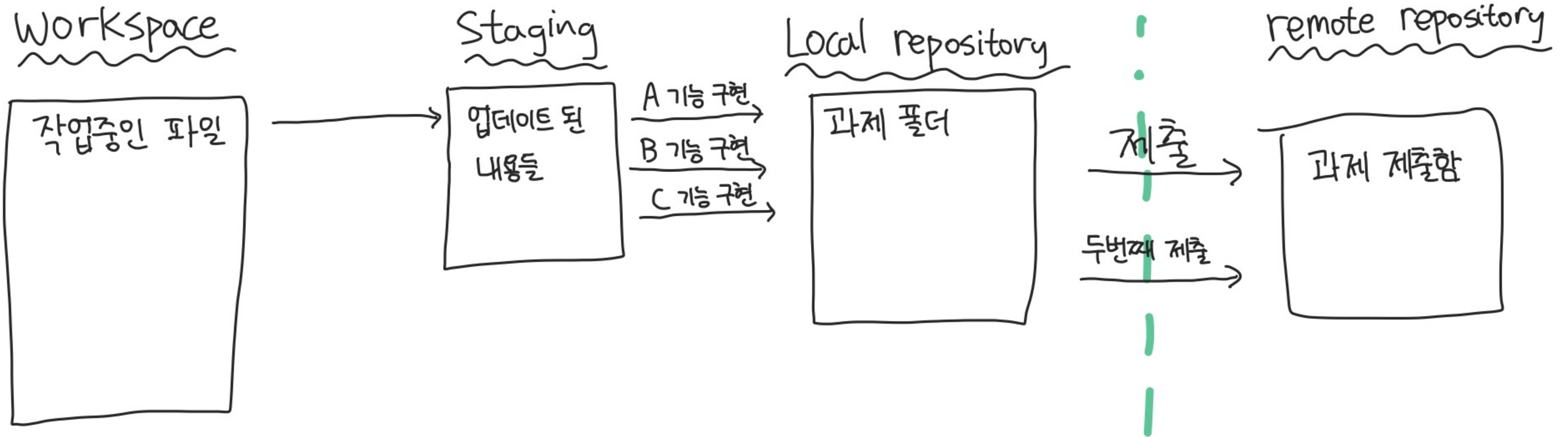
- 분산 저장의 장점과 중앙 저장의 장점을 모두 활용

# Git 상태





# 쉬운 예로 보여드리겠습니다.





# git 명령어 정리

- git init : 로컬 디렉토리에 git 저장소 생성
- git status : 현재 브랜치의 커밋 상태, 파일 상태를 확인
- git add <파일명> : 해당 파일 staging
- git add . : 작업한 모든 파일 staging
- git commit -m “메시지 내용” : 메시지와 함께 커밋 (변경사항 보고)
- git commit -a -m “메시지 내용” : staging과 커밋을 동시에 진행
- git log : 커밋 내역 확인
- git push : 지금까지 커밋된 내용들 remote에 저장 (== git push origin master)
- git branch : 브랜치 목록 조회
- git branch <브랜치 이름> : 브랜치 생성
- git branch -d <브랜치 이름> : 브랜치 삭제
- git checkout <브랜치 이름> : 해당 브랜치로 이동
- git pull : remote에 있는 프로젝트 상태를 현재 내 local과 병합

# git add

- 현재 작업중인 파일을 스테이징
- `git add .` : 업데이트된 모든 사항을 스테이징
- `git add <파일명>` : 해당 파일의 모든 변경사항을 스테이징

# git add .

```
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar | jeonghoon ± vim minhyuk.py
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar | jeonghoon ± git add .
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar | jeonghoon ±+ git status
```

On branch jeonghoon

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: minhyuk.py

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: ../.DS\_Store

Untracked files:

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

../Linux/2022-Linux-Git-seminar/

../a

../ex01/

```
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar | jeonghoon ±+ 
```

# git add <파일 이름>

```
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar |> jeonghoon ±+ vim jeonghoon.py
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar |> jeonghoon ±+ git add jeonghoon.py
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar |> jeonghoon ±+ git status

On branch jeonghoon
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   jeonghoon.py
    new file:   minhyuk.py

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:   ../.DS_Store

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ../Linux/2022-Linux-Git-seminar/
    ../a
    ../ex01/
```

# git restore

- 복구를 위한 명령어
- `git restore --staged <파일 이름>` : 스테이징된 파일을 스테이징에서 해제
- `git restore <파일 이름>` : 해당 파일을 수정 전으로 초기화

# git restore <파일 이름>

- 수정 사항을 초기화
- 말 그대로 수정된거 다 리셋!



# git restore --staged <파일 이름>

```
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ⌵ jeonghoon ±+ git restore --staged jeongho
on.py
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ⌵ jeonghoon ±+ git status
On branch jeonghoon
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   minhyuk.py

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:   ../.DS_Store

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    jeonghoon.py
    ../Linux/2022-Linux-Git-seminar/
    ../a
    ../ex01/

hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ⌵ jeonghoon ±+ 
```



# git commit

- 스테이징된 파일들을 로컬 저장소에 저장
- `git commit -m "<메시지 내용>"` : 커밋 메시지와 함께 저장
- `git commit -a -m "<메시지 내용>"` : 스테이징 + 커밋 동시에

# git commit -m

```
hooni 🌸 ➤ ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ➤ ⌵ jeonghoon ±+ ➤ git commit -m
error: switch `m' requires a value
x hooni 🌸 ➤ ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ➤ ⌵ jeonghoon ±+ ➤ git commit -m "[Feat] 장민
혁 추가"
[jeonghoon 5edabd2] [Feat] 장민혁 추가
1 file changed, 4 insertions(+)
create mode 100644 GitSeminar/minhyuk.py
```

# git push

- 로컬의 내용들을 remote에 저장

```
✖ hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ➤ ⌵ jeonghoon ± ➤ git push
```

```
fatal: The current branch jeonghoon has no upstream branch.  
To push the current branch and set the remote as upstream, use
```

```
git push --set-upstream origin jeonghoon
```

```
✖ hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ➤ ⌵ jeonghoon ± ➤ git push --set-upstream ori
```

```
gin jeonghoon
```

```
Enumerating objects: 9, done.
```

```
Counting objects: 100% (9/9), done.
```

```
Delta compression using up to 10 threads
```

```
Compressing objects: 100% (7/7), done.
```

```
Writing objects: 100% (8/8), 938 bytes | 938.00 KiB/s, done.
```

```
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
```

```
remote: Resolving deltas: 100% (1/1), done.
```

```
remote:
```

```
remote: Create a pull request for 'jeonghoon' on GitHub by visiting:
```

```
remote:      https://github.com/CLUG-kr/2022-Linux-Git-seminar/pull/new/jeonghoon
```

```
remote:
```

```
To https://github.com/CLUG-kr/2022-Linux-Git-seminar.git
```

```
* [new branch]      jeonghoon -> jeonghoon
```

```
Branch 'jeonghoon' set up to track remote branch 'jeonghoon' from 'origin'.
```

# ex01

- <본인 이름>.txt 파일을 만들고 add, commit, push를 해보자.

# git branch

- 브랜치 관련 명령어
- git branch : 모든 브랜치를 보여줌
- git branch <브랜치 명> : 브랜치 생성
- git branch -d <브랜치 명> : 해당 브랜치 삭제
- git branch -r : 원격 저장소의 브랜치 확인
- git checkout <브랜치 명> : 해당 브랜치로 이동

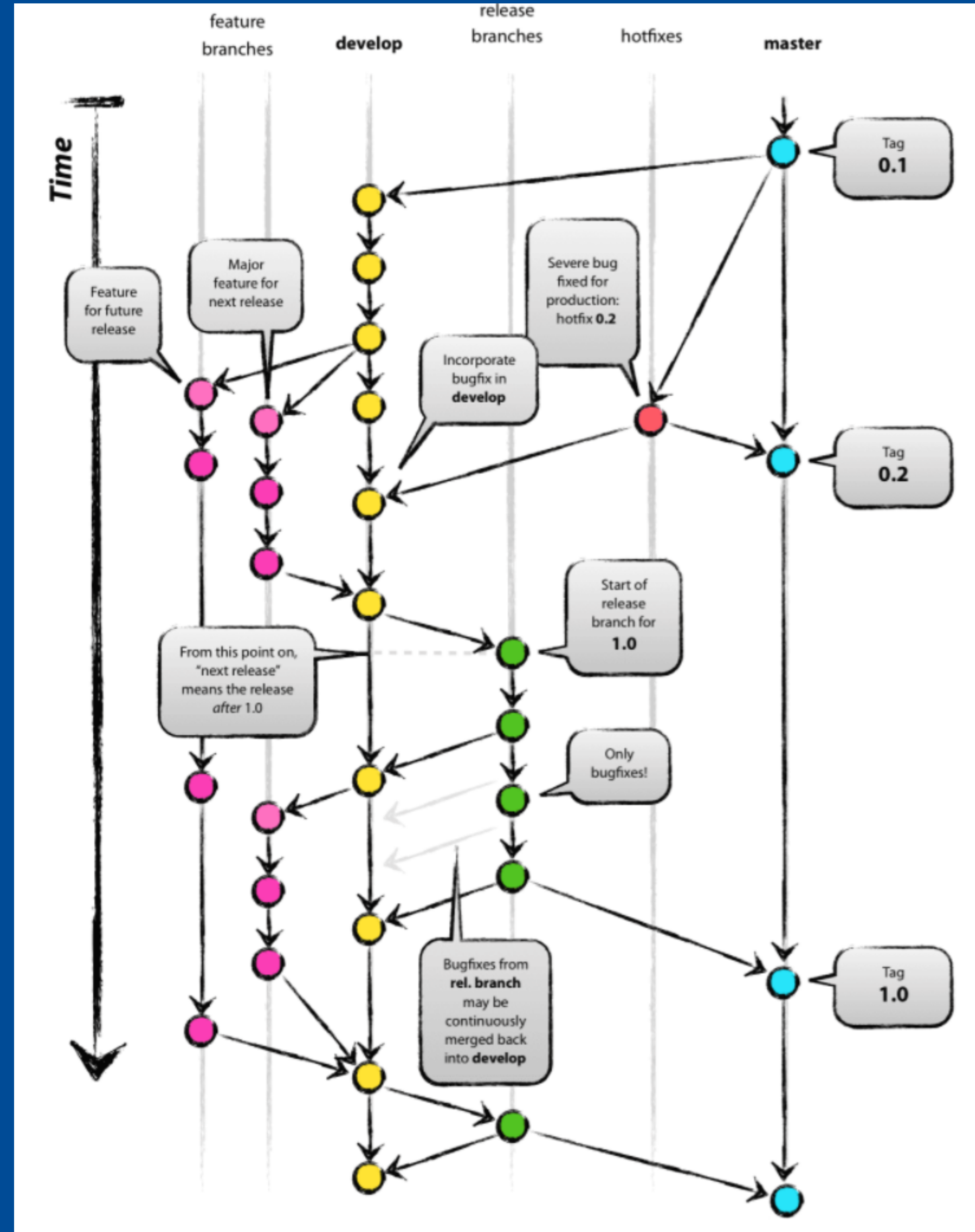
# git branch

- 실습으로 보여드리겠습니다.



# git branch 전략

- 저는 feature/<issue 번호> 를 선호합니다.
- develop의 경우 develop/1.0.0





# git merge

- 브랜치 합체
- git merge <합칠 브랜치 명>
- 현재 브랜치에 <합칠 브랜치 명> 을 합침
- 충돌이 날 수도 있습니다.

# git merge

```
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar ➤ main ➤ git merge jeonghoon
Updating 514d302..7eec20d
Fast-forward
 GitSeminar/jeonghoon.py | 3 +++
 GitSeminar/minhyuk.py   | 4 ++++
 2 files changed, 7 insertions(+)
 create mode 100644 GitSeminar/jeonghoon.py
 create mode 100644 GitSeminar/minhyuk.py
hooni 🌸 ~/Documents/clug/2022-Linux-Git-seminar ➤ main ➤ ls
GitSeminar Linux      README.md  ex00      ex01      ex02      _
```

# git merge 충돌이 나는 경우

```
hooni 🌸 ➤ ~/Documents/clug/2022-Linux-Git-seminar/GitSeminar ➤ main ➤ git merge jeonghoon
Auto-merging GitSeminar/jeonghoon.py
CONFLICT (content): Merge conflict in GitSeminar/jeonghoon.py
Automatic merge failed; fix conflicts and then commit the result.
```

# 해결 방법

- 강제 푸시로 내 코드로 덮어 씌움
- 충돌나는 부분을 하나하나 살펴보고 선택하며 충돌 해결
- 내 코드 말고 상대방의 코드로 덮어 씌움

# 깃크라켄



- git의 gui 버전
- GUI이기 때문에 상태 확인이 편함
- 명령어를 덜 칩
- 저는 충돌해결은 무조건 크라켄으로 합니다.

# clone

Repositories

Access local repos, clone remote repos, or start new repos on your favorite hosting services. If you work on multiple repos, try Workspaces.

Open a repo

Clone a repo

Start a local repo

Recent Repos

2022-Linux-Git-seminar

~/Documents/clug/...

jpa\_begin

~/Documents/jpa\_begin

Deartoday-Server

~/Documents/Deartoday...

BeMyPlan-Android

~/Documents/BeMyPla...

CAUSW\_backend

~/Documents/CAUSW\_ba...

Oisireoyo-Backend

~/Documents/Oisireoy...

JPA-shop

~/Documents/JPA-shop

spring-practice

~/Documents/spring-practice

Start a hosted repo:

on GitHub

on GitLab

Workspaces

New

Group repos you use the most into Workspaces that aggregate PR and issue data, allow you to perform bulk actions, and can be shared with your team.

Open Workspaces

New Workspace

Integrations

Connect integrations to git hosting services and issue trackers for quick access to remote repos and streamlined workflows with PRs, issues, and more.

GitHub

GitHub Enterprise

GitLab

GitLab Self-Managed

Bitbucket

Bitbucket Server

Azure DevOps

Jira Cloud

Jira Server

Trello

Other tools

GitKraken CLI

New Terminal Tab

GitKraken Boards

Open GitKraken Boards

New GitKraken Board

GitKraken Timelines

Open GitKraken Timelines

28

add

main

jeonghoon

master

Initial commit

답안지

문제

Merge pull request #1 from CLUG-kr/master

Add files via upload

pdf is too big...

[Feat] 장민혁 추가

[Feat] 박정훈 추가

[Fix] jeonghoon.py 저 -> 나 로 수정

[Fix] jeonghoon.py 장민혁 -> 여러분 으로 수정

Merge branch 'jeonghoon' into main

// WIP

+ 1

Path

Tree

Unstaged Files (1)

Stage all changes

GitSeminar/kraken.py

Staged Files (0)

Commit Message

Amend



Summary



Description




# commit

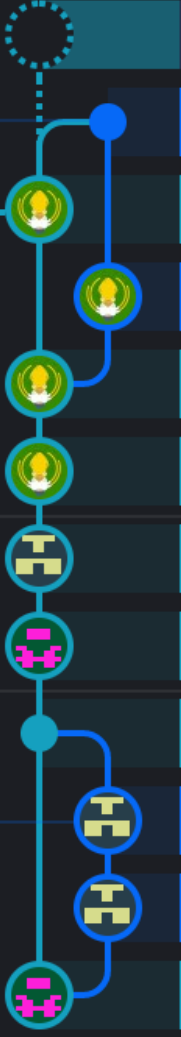
BRANCH / TAG

main  

✓ jeonghoon  

master 

GRAPH



COMMIT MESSAGE

[Feat] add kraken.py + 1

Merge branch 'jeonghoon' into main

[Fix] jeonghoon.py 장민혁 -> 여러분 으로 수정

[Fix] jeonghoon.py 저 -> 나 로 수정

[Feat] 박정훈 추가

[Feat] 장민혁 추가

pdf is too big... 12 hours ago

Add files via upload yesterday

Merge pull request #1 from CLUG-kr/master

문제

답안지

Initial commit

1 file change on jeonghoon

↑ ↕

Path Tree

Unstaged Files (0)

Staged Files (1)

Unstage all changes

+ GitSeminar/kraken.py

Commit Message

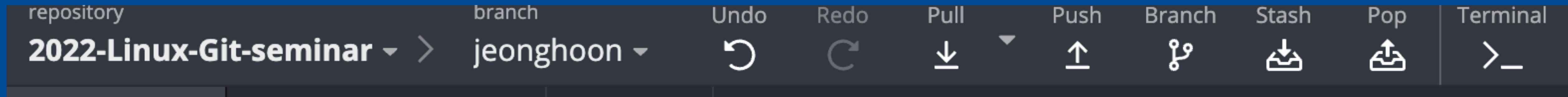
Amend

[Feat] add kraken.py 52

Description

Commit changes to 1 file

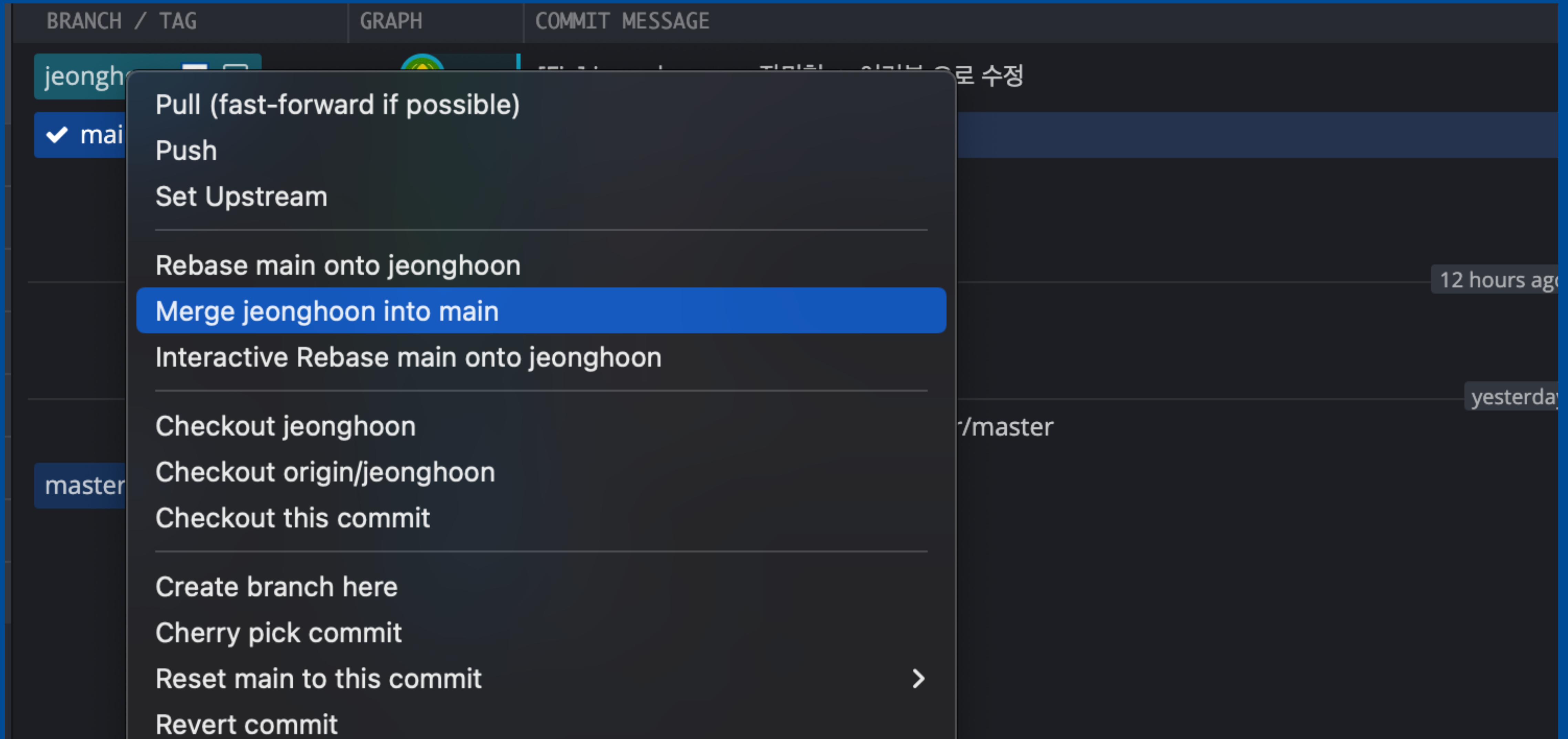
# push



# checkout

- 더블클릭

# merge



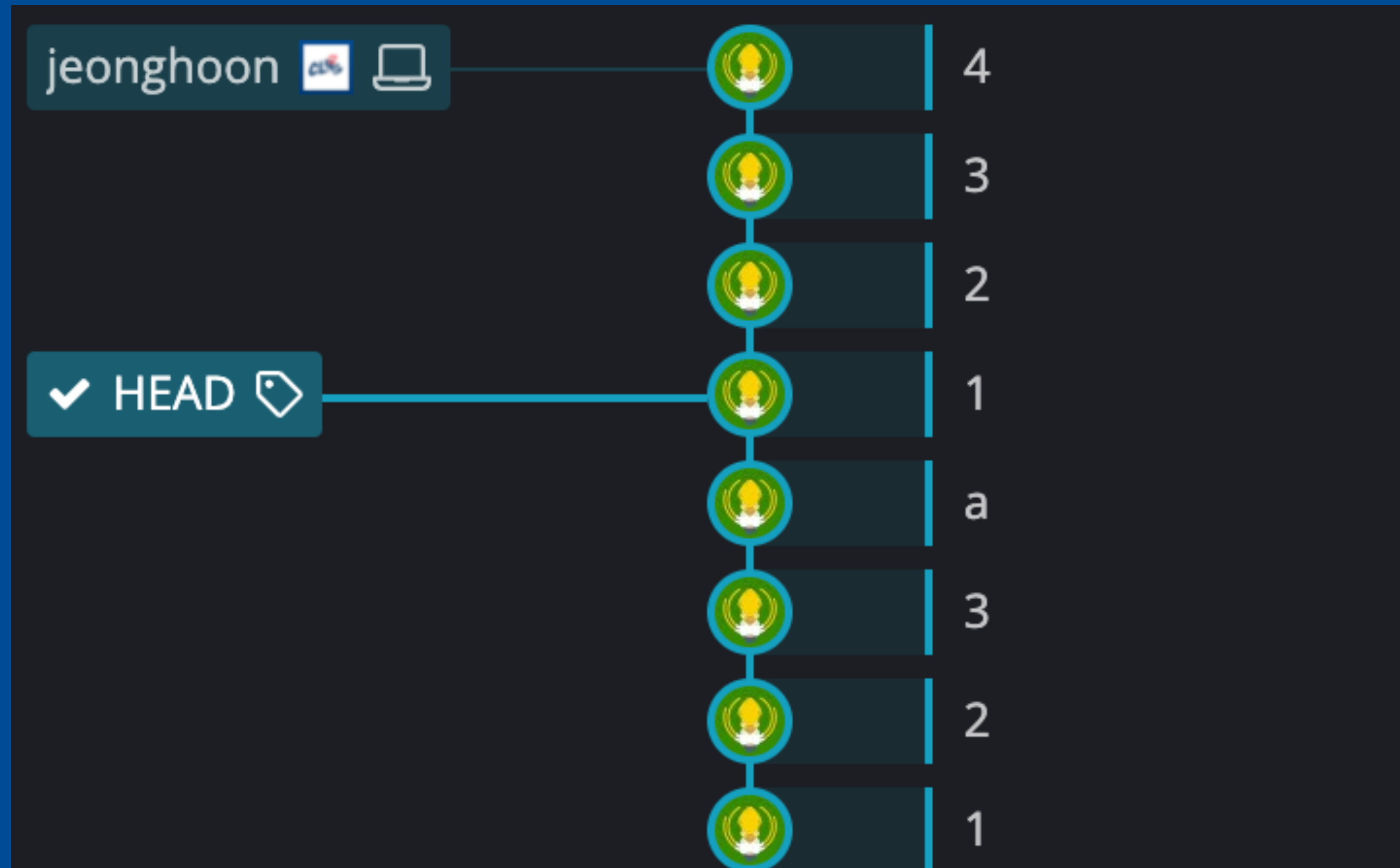
# merge

# 이전 커밋으로 가기

- `git reset <옵션> <돌아가고 싶은 커밋 id>`
- 옵션 : `soft`, `mixed`, `hard`
- `git reset —hard <커밋id>` : 돌아가려는 이력 이후의 모든 내용을 지움
- `git reset —soft <커밋id>` : 돌아가려 했던 이후의 내용 보존, 스테이징이 된 상태
- `git reset —mixed <커밋id>` : 돌아가려 했던 이후의 내용 보존, 스테이징이 안된 상태
- `git reset HEAD~<숫자>` : 몇 개 이전 커밋으로 이동. `mixed` 옵션으로.

# 이전 커밋으로 가기 (커밋 손실 없이)

- git checkout <커밋id>





# ex02

- 본인이름의 브랜치를 만드세요.
- 본인 브랜치로 이동하세요.
- GitSeminar 폴더에 a.txt를 만들어 1을 쓰고 커밋, 옆에 2를 쓰고 커밋 ... 3까지 하세요.  
(마지막 커밋할 때 a.txt에는 123가 적혀있어야 합니다.)
- git reset을 이용하여 자유롭게 이전 커밋들로 돌아가봅시다!

# pull request

- merge하기 전에 다른 사람들에게 허락받기
- 코드리뷰를 받을 수 있음
- 남의 것은 코드리뷰 가능
- 다른 사람 approve 없이 merge 못하게 막을 수도 있음
- 라벨 설정 가능
- 칸반보드 설정 가능
- 등등등... 굉장히 다양하게 설정 가능하다

# pull request

Feature/91 agenda #93

Merged

shb03323 merged 6 commits into develop/1.0.0 from feature/91-agenda on 22 Jul

Conversation 0

Commits 6

Checks 0

Files changed 4

+4,448 -4,370

shb03323 commented on 22 Jul

Member

PR 한 줄 요약

푸시알림 스케줄링

PR 세부 내용

푸시알림 스케줄링

shb03323 added 6 commits 2 months ago

[Chore] #91 - change order of responseMessage

70543ca

[Chore] #91 - add agenda dependency

a1ac0a1

[Feat] #91 - add agenda in pushAlarmService

037a7c3

[Add] #91 - agend로 3초 후 시간 스케줄링

122f569

[Fix] #91 - 스케줄링 성공

d0a6767

[Feat] #91 - delete data when schedule is done

4c18f22

shb03323 added 정훈 feat labels on 22 Jul

shb03323 self-assigned this on 22 Jul

shb03323 added this to In progress in Dear today (Server) via automation on 22 Jul

Reviewers

No reviews

Assignees

shb03323

Labels

feat 정훈

Projects

Dear today (Server)

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

Notifications

Unsubscribe

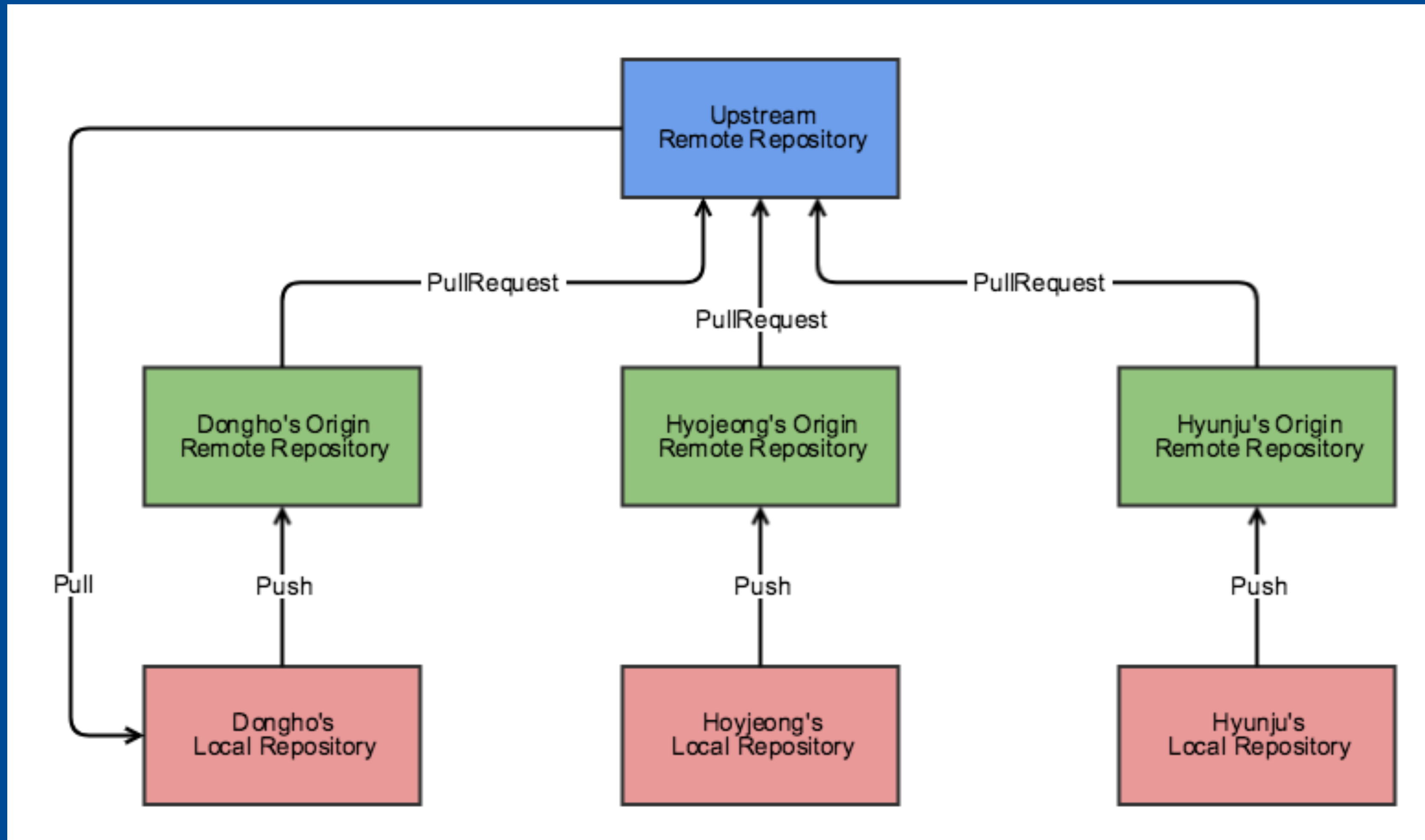
You're receiving notifications because you modified the open/close state.

59

# pull request

- 한 번 보여드리겠습니다.

# 보통의 pull request

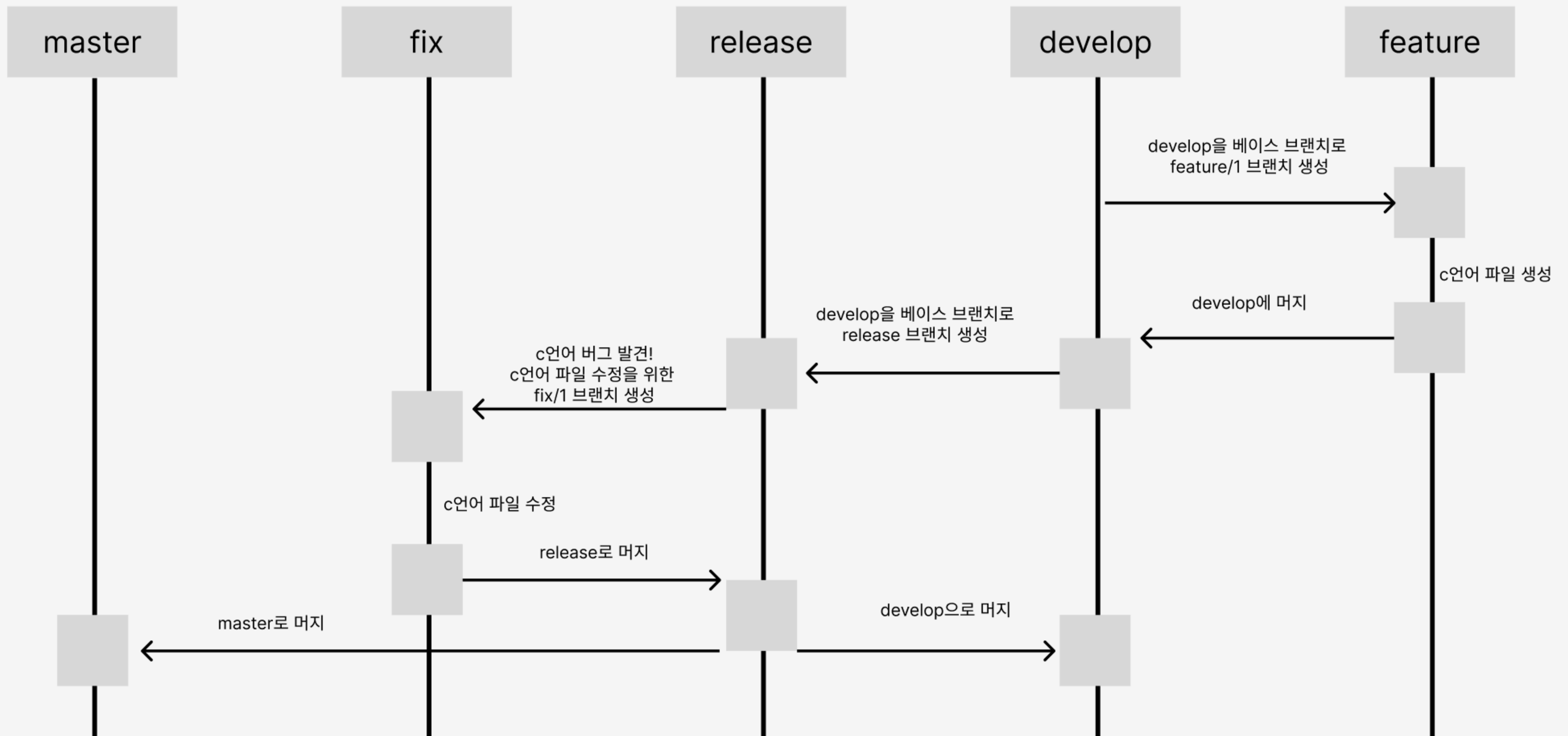


# ex03

- GitSeminar 폴더에 a.txt를 만들고 아무 글자나 쓰세요.
- 작업한 것을 pull request 올려보세요.

# ex04

- 아래의 그림처럼 만들어주세요.





# gitignore

- git에 올리면 안되는 것을 git에 올라가지 않도록 막아줌
- 보안적인 부분을 도와줌
- 보통 프로젝트 생성할 때 같이 생성
- gitignore로 인해 너는 되는데 나는 안되네와 같은 경우가 많이 발생합니다.
- 숨김파일은 카톡이나 슬랙을 통해 공유하세요. (아니면 도커)

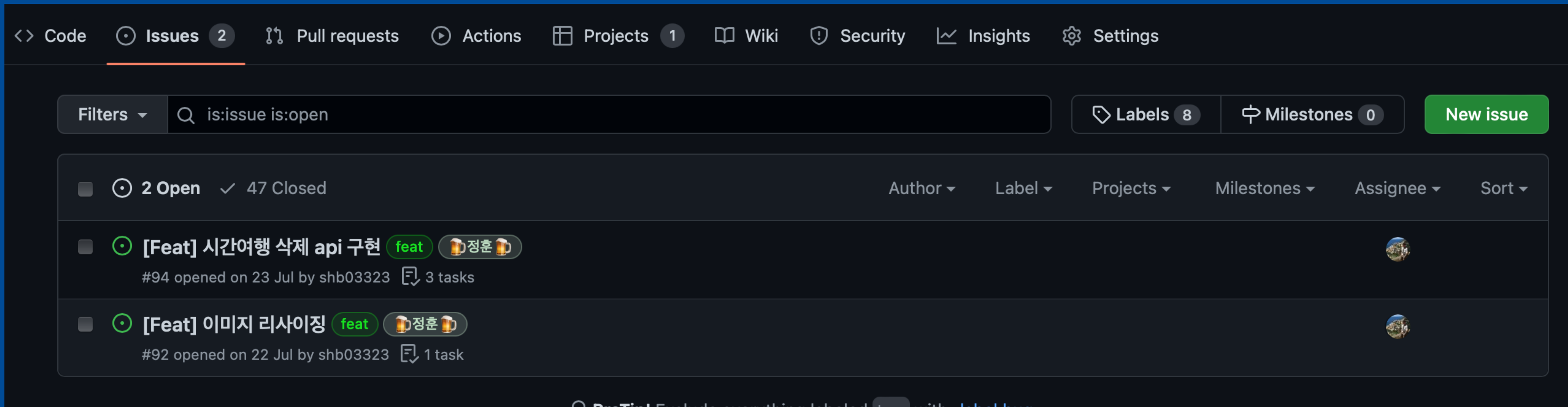
# gitignore.io

- gitignore 파일을 생성해주는 사이트
- 프로젝트 스택에 맞게 키워드를 입력하고 생성을 누르시면 됩니다.



# issue

- 작업할 것을 등록하는 것
- 보통 issue에 따라 작업을 진행하고 pull request를 함



The screenshot shows the GitHub Issues interface. At the top, there's a navigation bar with tabs: Code, Issues (2), Pull requests, Actions, Projects (1), Wiki, Security, Insights, and Settings. Below this, there's a search bar with the filter 'is:issue is:open'. To the right of the search bar, there are buttons for 'Labels 8' and 'Milestones 0', and a green 'New issue' button. The main content area displays a list of issues. The first issue is '[Feat] 시간여행 삭제 api 구현' (feat) by shb03323, opened on 23 Jul, with 3 tasks. The second issue is '[Feat] 이미지 리사이징' (feat) by shb03323, opened on 22 Jul, with 1 task. Both issues are assigned to a user named '정훈' (Jeonghun). The interface is in dark mode.

<> Code Issues 2 Pull requests Actions Projects 1 Wiki Security Insights Settings

Filters  Labels 8 Milestones 0 New issue

☐ 2 Open ✓ 47 Closed Author Label Projects Milestones Assignee Sort

☐ [Feat] 시간여행 삭제 api 구현 feat 정훈  
#94 opened on 23 Jul by shb03323 3 tasks

☐ [Feat] 이미지 리사이징 feat 정훈  
#92 opened on 22 Jul by shb03323 1 task

ProTip! Exclude everything labeled bug with label:bug

# action

- 프로젝트 관리를 도와주는 추가적인 툴
- CI/CD, security 등등 다양
- 어려워요...