

수고하셨습니다~!

The screenshot shows a GitHub repository page for 'Git-Study'. The repository is public and has 1 branch and 0 tags. The main branch is 'main'. A search bar at the top right allows going to a file or adding a new file. The code tab is selected. The commit history is as follows:

Author	Commit Message	Time Ago
bigwave100	[FEAT] bigwave100 README 생성	10 hours ago
OneDrive/바탕 화면/Sogang/Release/Test	[MOD] 리드미 수정	2 days ago
bigwave100	[FEAT] bigwave100 README 생성	10 hours ago
docs	[ADD] Week 1 study PDF file	last week
hwlee	[FEAT] hwlee README 생성	17 hours ago
hyeonsang010716	[FEAT] hyeonsang010716 리드미 추가	last week
jungsb0415	[FEAT] jungsb0415 README 생성	14 hours ago
kyuhung	[FEAT] kyuhung README 생성	10 hours ago
pinkgorae	[FEAT] pinkgorae 리드미 추가	last week
wjm9765	[FEAT] wjm9765 README 생성	14 hours ago
xxxiv-34	[FEAT] xxxiv-34 리드미 추가	16 hours ago
yjs2673	[FEAT] yjs2673 README 생성	14 hours ago

At the bottom left, there is a link to 'README'.

About

2025 SGCS Release Git Study Repository

Activity, Custom properties, 0 stars, 0 watching, 0 forks, Report repository.

Releases

No releases published. Create a new release.

Packages

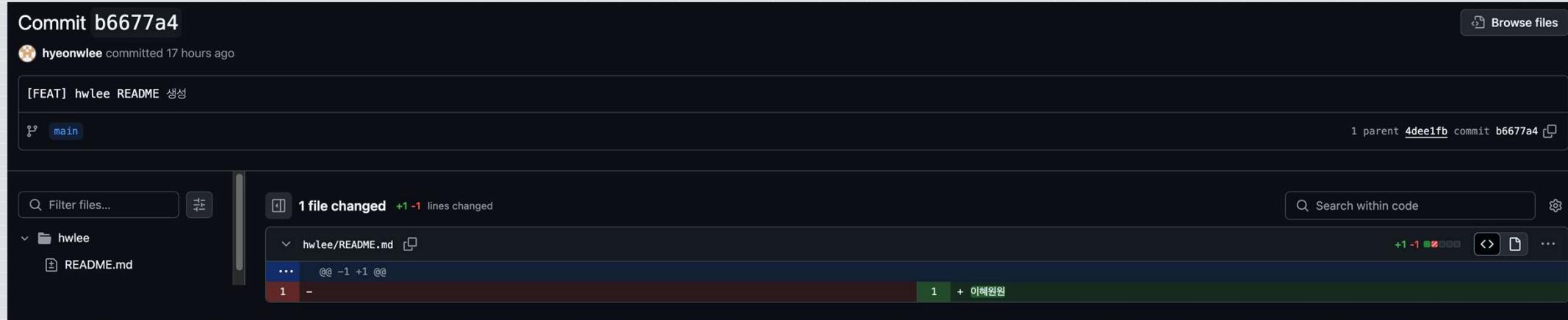
No packages published. Publish your first package.

Contributors 10

Icons for 10 contributors are shown.

모두 1주차 실습하시느라 고생하셨어요~!

₩/드 백



Commit b6677a4

hyeonwlee committed 17 hours ago

[FEAT] hwlee README 생성

main 1 parent 4dee1fb commit b6677a4

Filter files... 1 file changed +1 -1 lines changed Search within code

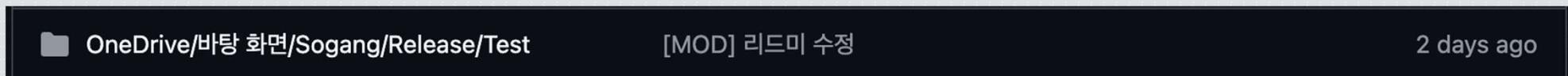
hwlee README.md @@ -1 +1 @@ 1 - 1 + 이해원원

This screenshot shows a GitHub commit page for a repository. The commit message is "[FEAT] hwlee README 생성". The commit was made by user "hyeonwlee" 17 hours ago. It has one parent commit, "4dee1fb". The commit is associated with branch "main". A merge conflict is shown in the file "hwlee/README.md", specifically in the line "1 -" which has been replaced by "+ 이해원원". The GitHub interface includes standard navigation and search tools.

윈도우 한글 두 번 입력 에러는 아래 방법으로 해결할 수 있어요!

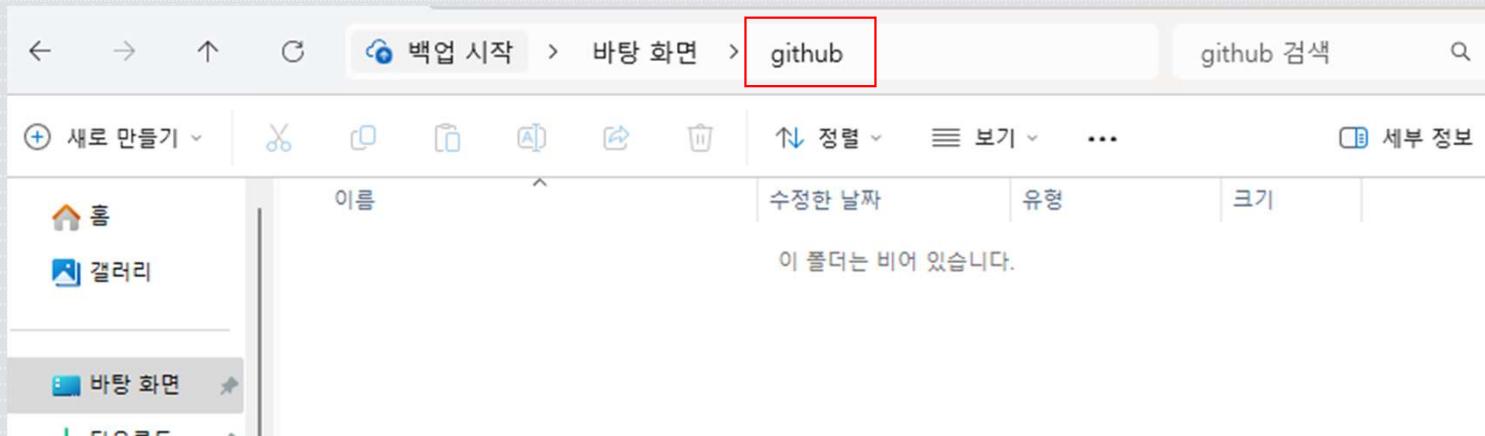
-> <https://blog.naver.com/yiyouya/223470446020>

파드백



어떤 일이 있었던 거죠!! 😅

깃을 사용할 때는 리프 디렉토리를 만들고 해당 위치에서 터미널을 사용해야 해요!



```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github
```

피드백

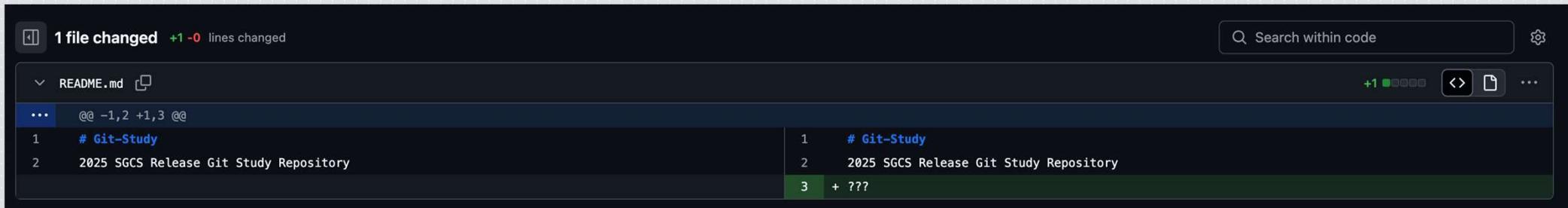
```
④ hyeonsang@johyeonsang-ui-MacBookPro git-study % git remote add origin https://github.com/SGCS-Release-Git-Project/Git-Study.git  
error: remote origin already exists.  
○ hyeonsang@johyeonsang-ui-MacBookPro git-study %
```

만약 git remote add origin 잘못된 URL 입력으로 꼬였을 경우!

git remote rm origin # origin 삭제함
또는
rm -rf .git # git을 삭제함

방법으로 해결할 수 있음!

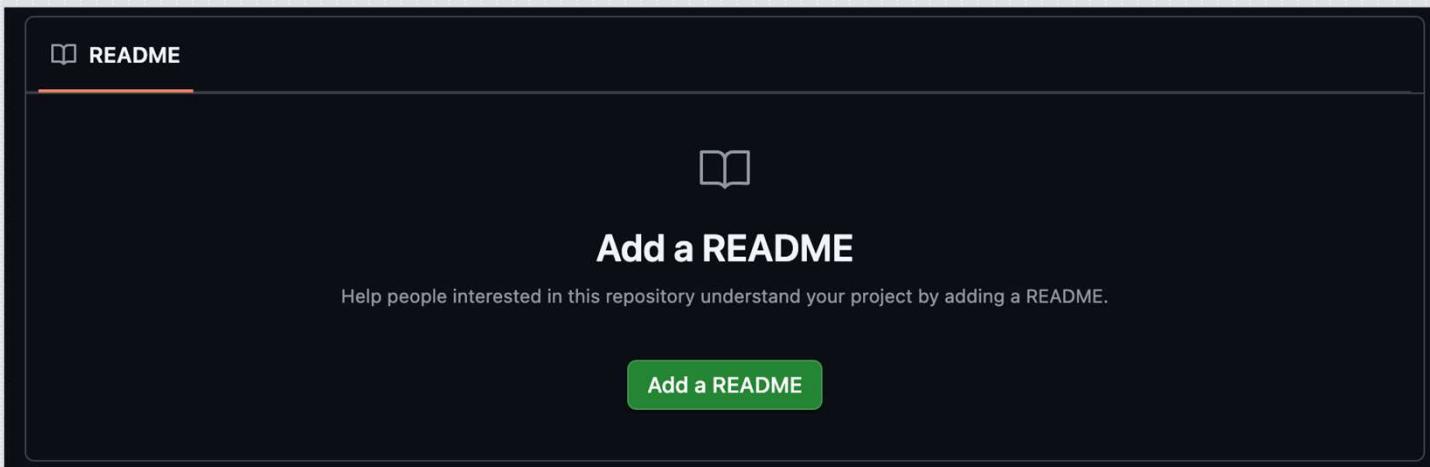
문제점



A screenshot of a GitHub commit history. At the top, it says "1 file changed +1 -0 lines changed". Below that, under "README.md", there is a diff view:

```
@@ -1,2 +1,3 @@
1 # Git-Study
2 2025 SGCS Release Git Study Repository
3 + ???
```

The commit message is "# Git-Study" and the date is "2025 SGCS Release Git Study Repository". A green bar at the bottom indicates the addition of three new lines.



건드리면 안 되는 파일 수정



0/오/에도..

- 왜 git pull origin main 을 사용해야 했을까!!

컨트롤 + s 를 안 해서, 에러를 겪으신 분들은 아래 VSCode 자동 저장 참고하세요!

(<https://hianna.tistory.com/354>)

깃협 히스토리

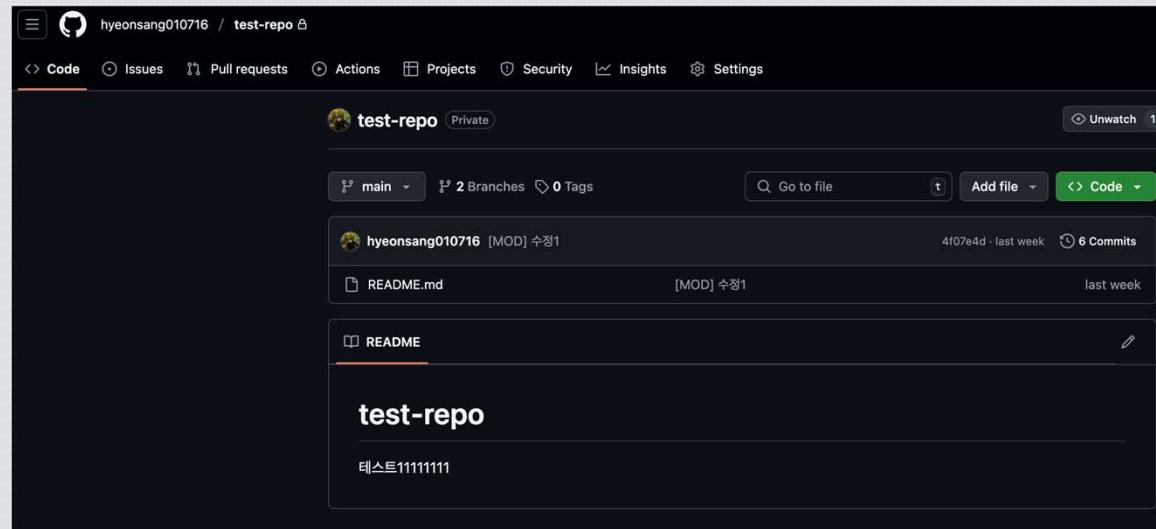
A screenshot of a GitHub repository named "Git-Study" (Public). The repository has 1 branch and 0 tags. A recent commit by "bigwave100" titled "[FEAT] bigwave100 README 생성" was made 18 hours ago with commit hash f1aa421. A red box highlights the "21 Commits" link. The interface includes standard GitHub navigation buttons like "Edit Pins", "Watch", "Go to file", "Add file", and "Code".

깃협에는 히스토리가 존재하고 매우 중요합니다!

A detailed view of the commit history for March 30, 2025. The commits are listed in chronological order:

- [FEAT] bigwave100 README 생성 (Verified) - authored by bigwave100 18 hours ago, commit hash f1aa421.
- [FEAT] bigwave100 리드미 추가 (Verified) - authored by bigwave100 18 hours ago, commit hash ec42d3b.
- [FEAT] kyuhyung README 생성 - committed by kyuhyunglee 18 hours ago, commit hash e4f6030.
- [MOD] 리드미 수정 - committed by bigwave100 20 hours ago, commit hash 843072b.
- [FEAT] bigwave100 리드미 생성 - committed by bigwave100 20 hours ago, commit hash a2ce65c.
- Merge latest changes - committed by jungsbo415 yesterday, commit hash 4ebdb06.
- [FEAT] wjm9765 README 생성 - committed by wjm9765 yesterday, commit hash 8e7bccef.
- [MOD] 리드미 테스트 - committed by wjm9765 yesterday, commit hash 8121bf6.
- [FEAT] jungsbo415 README 생성 - committed by jungsbo415 yesterday, commit hash 05706ee.

깃헙 히스토리 예제



Code Issues Pull requests Actions Projects Security Insights Settings

test-repo Private

main 2 Branches 0 Tags

Go to file Add file Code

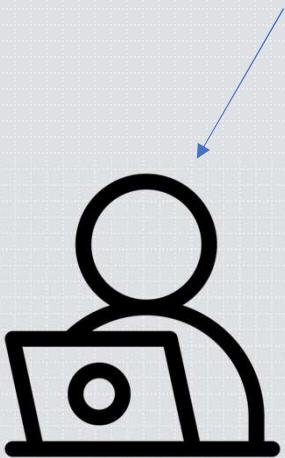
hyeonsang010716 [MOD] 수정1 4f07e4d · last week 6 Commits

README.md [MOD] 수정1 last week

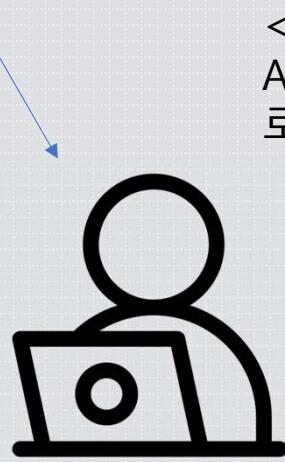
README

test-repo

테스트111111111



A개발자



B개발자

히스토리

[MOD] 수정1

A

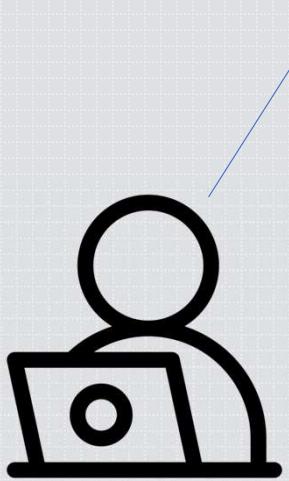
B

<시나리오>

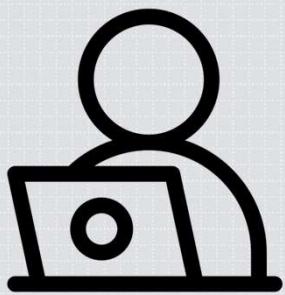
A개발자와 B개발자가 리모트 main 브ranche를
로컬에 가져와서 개발할 예정!

깃헙 히스토리 예제

The screenshot shows a GitHub repository named 'test-repo'. The main branch is 'main'. There are 2 branches and 0 tags. A commit from user 'hyeonsang010716' titled '[MOD] README 수정' was made at 6da7db2 · now, with 7 commits. The file 'README.md' was updated. The commit message '[MOD] README 수정' is also present in the commit history. The repository description is 'test-repo' and the owner is 'A개발자'.



A개발자



B개발자

히스토리

[MOD] 수정1

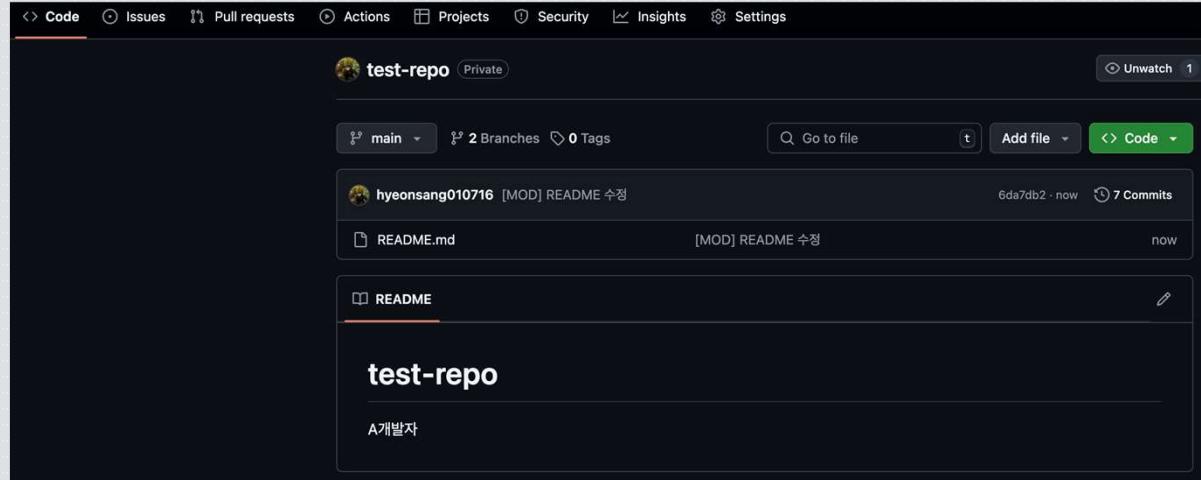
[MOD] README
수정

B

A

<시나리오>
A 개발자가 리모트를 수정해서 푸쉬 함.

깃헙 히스토리 예제



히스토리

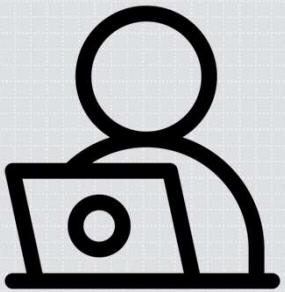
[MOD] 수정1

[MOD] README
수정

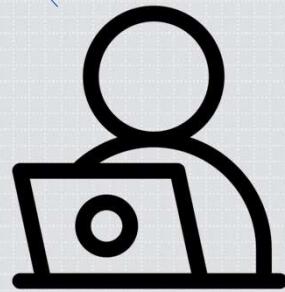
B

A

<시나리오>
그 다음 B개발자가 개발을 하고, 푸쉬를 할 거임



A개발자



B개발자

깃헙 히스토리 예제

```
main.py
1  print("hello world")
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
• $ git add main.py

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
• $ git commit -m "[FEAT] hello world 구현"
[main a8d7809] [FEAT] hello world 구현
 1 file changed, 1 insertion(+)
  create mode 100644 main.py
```

```
1 # test-repo
2 테스트111111111111
3
```

```
EXPLORER OPEN EDITORS main.py ...
1  print("hello world")
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

1 file changed, 1 insertion(+)
create mode 100644 main.py
```

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
• $ git push
To https://github.com/hyeonsang010716/test-repo.git
 ! [rejected]      main -> main (fetch first)
error: failed to push some refs to 'https://github.com/hyeonsang010716/test-repo.git'
hint: Updates were rejected because the remote contains work that you do not
hint: have locally. This is usually caused by another repository pushing to
hint: the same ref. If you want to integrate the remote changes, use
hint: 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
• $
```

하지만 충돌 발생!

깃헙 히스토리 예제

A developer (A) pushes a commit to the 'main' branch of the 'test-repo' repository.

히스토리

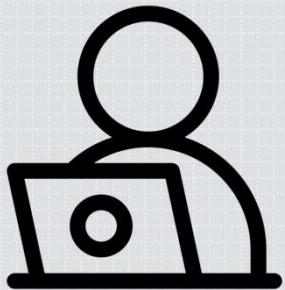
[MOD] 수정1

[MOD] README
수정

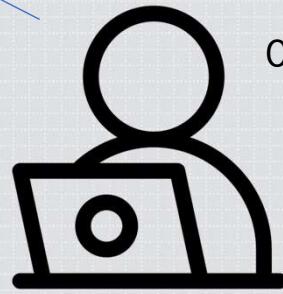
B

A

충돌



A개발자



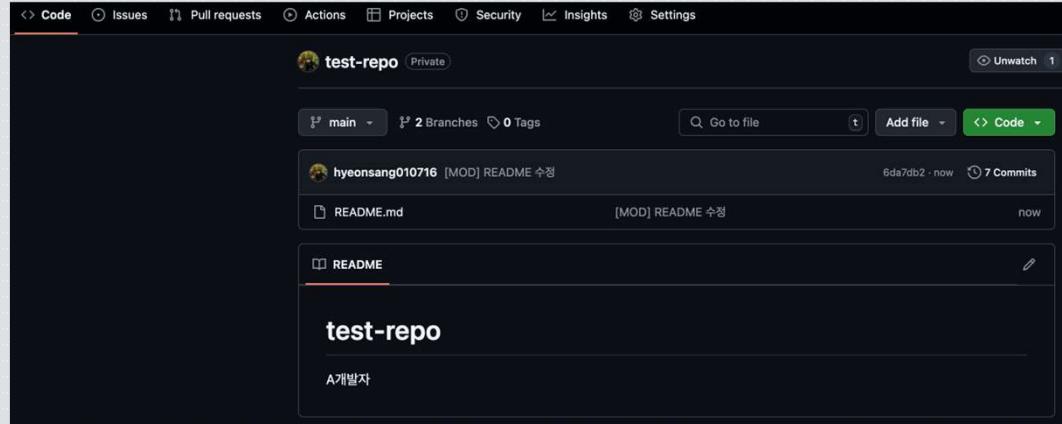
B개발자

<시나리오>

B 개발자가 리모트를 수정해서 푸쉬 함.
하지만 충돌!

이유) 최신 히스토리가 아니라서!

깃헙 히스토리 예제



히스토리

[MOD] 수정1

[MOD] README
수정

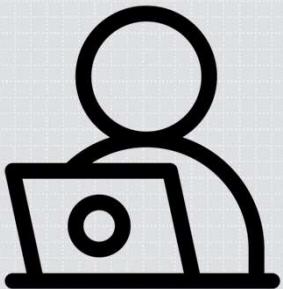
B

A

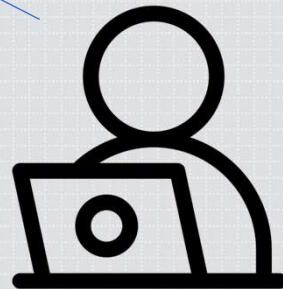
충돌

충돌을 해결하기 위해선, 히스토리 재정리가 필요함!

히스토리 충돌이 일어날 때, 필요한 명령어 : **git pull**



A개발자



B개발자

git pull origin main

리모트 main 브랜치에 있는 변경 사항을 가져옴
즉 업데이트 된 히스토리를 가져 온다.

만약 B개발자가 push를 하기 전, pull을 먼저 했다면
어땠을까?

깃헙 히스토리 예제

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ git pull origin main
From https://github.com/hyeonsang010716/test-repo
 * branch            main      -> FETCH_HEAD
Updating 3d8b3b5..3208c0c
Fast-forward
 README.md | 1 +
 1 file changed, 1 insertion(+)

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ 
```

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ git add .
$ git commit -m "[FEAT] hello world 구현"
[main 72b1eaa] [FEAT] hello world 구현
 1 file changed, 1 insertion(+)
  create mode 100644 main.py

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 22 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 326 bytes | 163.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/hyeonsang010716/test-repo.git
 3208c0c..72b1eaa main -> main

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ 
```

깃헙 히스토리 예제

Code Issues Pull requests Actions Projects Security Insights Settings

Unwatch 1

test-repo Private

main 2 Branches 0 Tags

Go to file Add file Code

hyeongsang010716 [MOD] README 수정 6da7db2 · now 7 Commits

README.md [MOD] README 수정 now

README

test-repo

A개발자

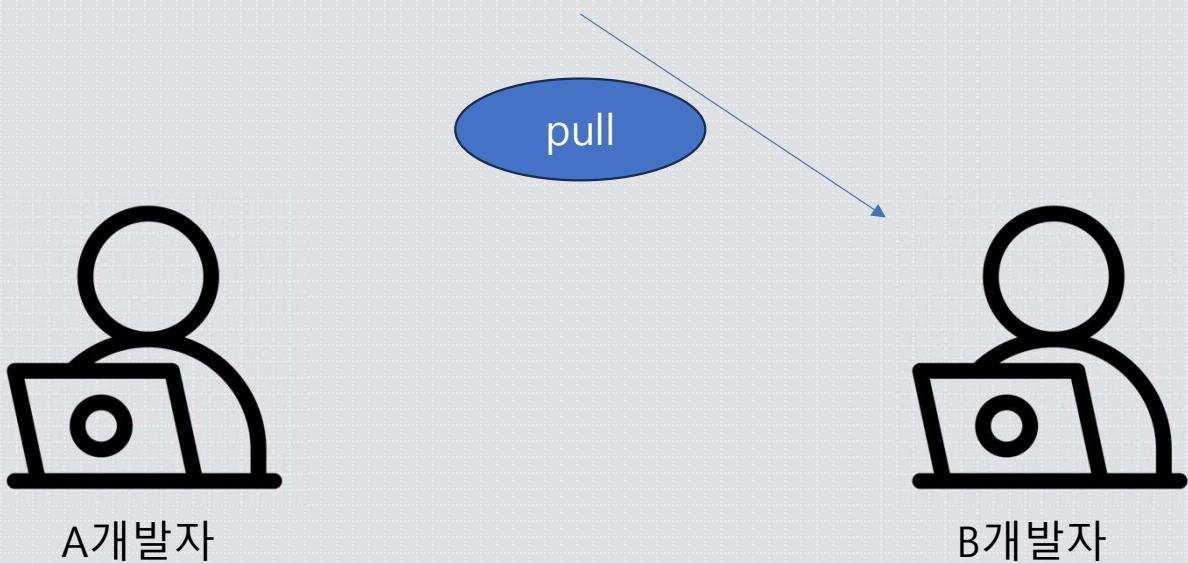
히스토리

[MOD] 수정1

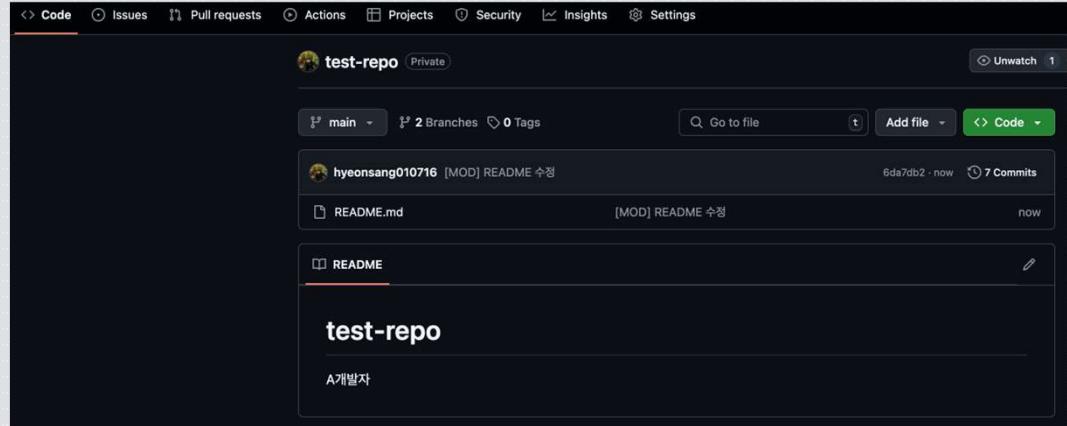
[MOD] README
수정

A

B



깃헙 히스토리 예제



히스토리

[MOD] 수정1

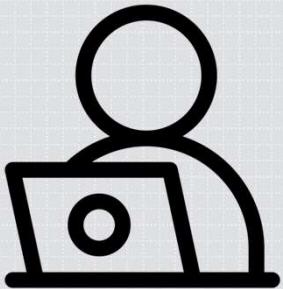
[MOD] README
수정

[FEAT] hello ~

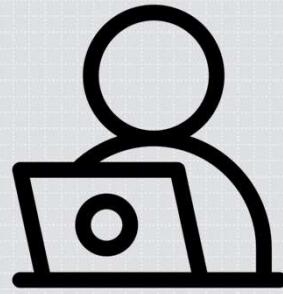
A

B

push

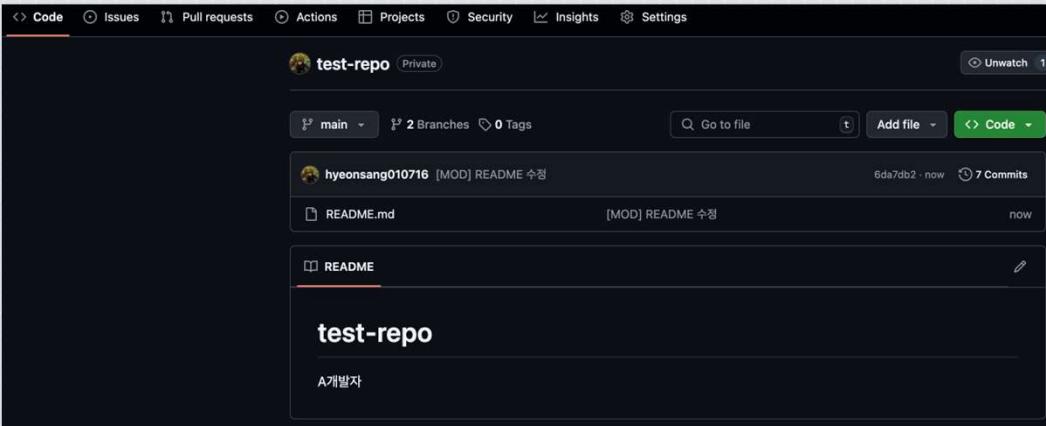


A개발자

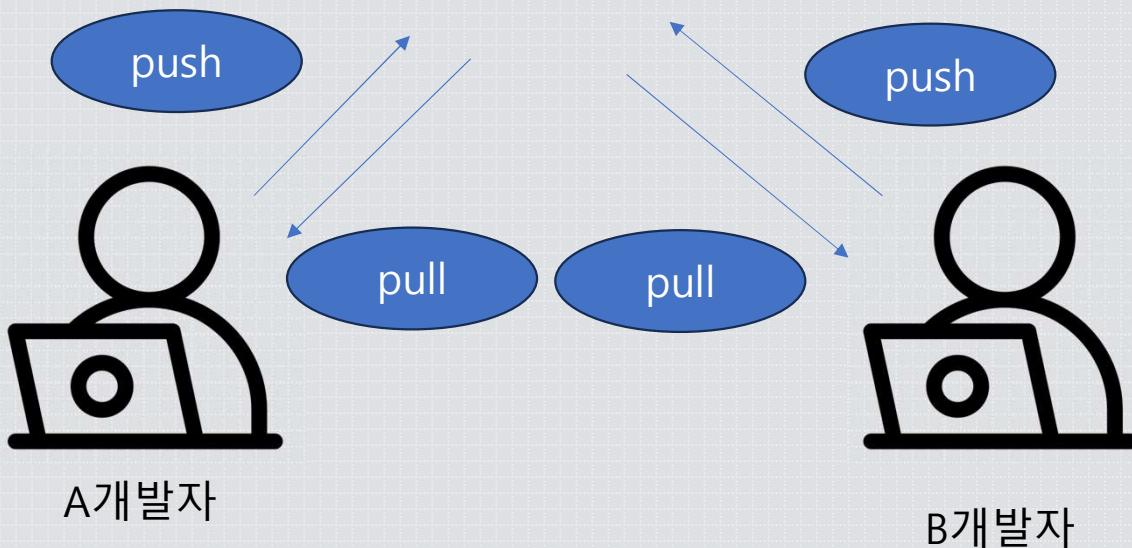


B개발자

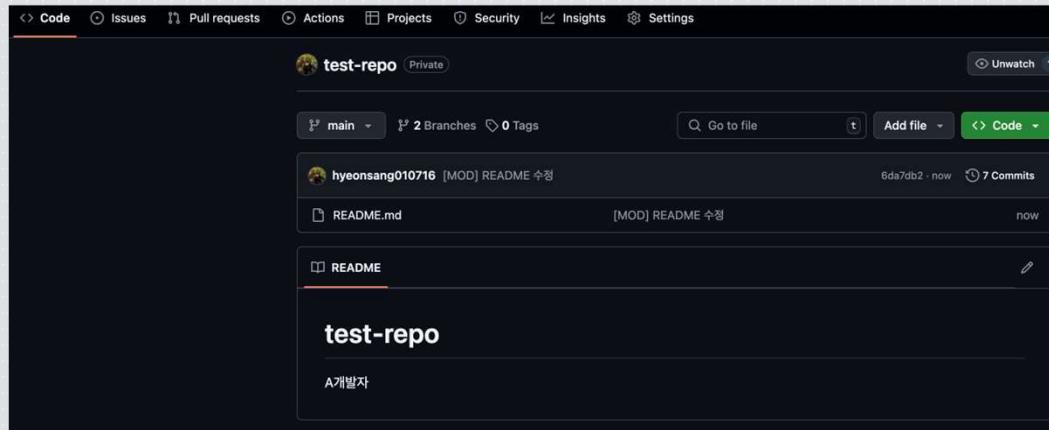
깃헙 히스토리 예제



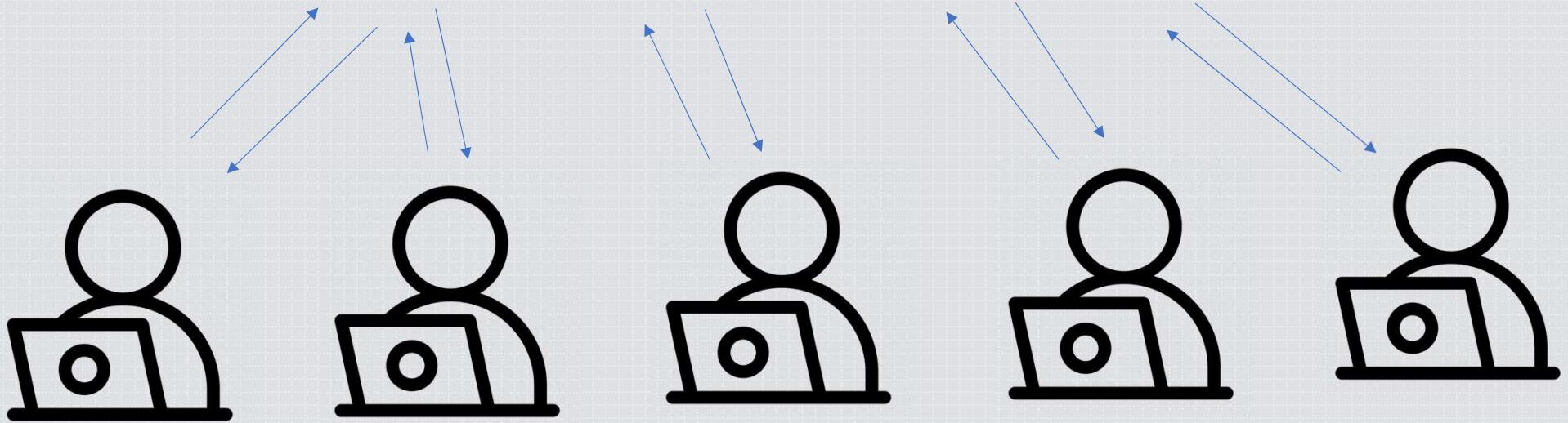
그렇다면 계속 pull, push ,
pull, push... 해야할까..?



깃헙 히스토리 예제

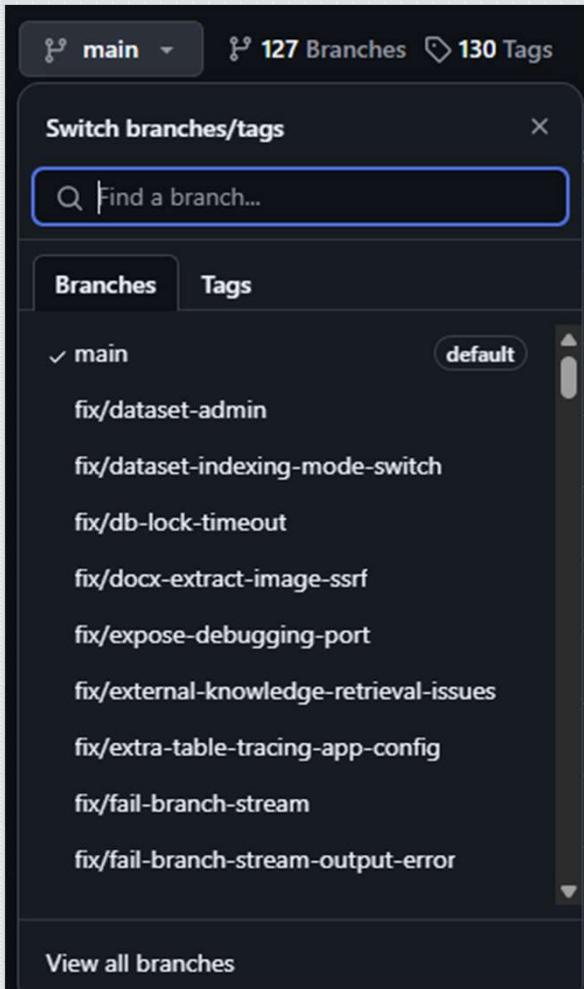


N명이라면..?



깃헙 브런치 예제

해당 문제 때문에, 하나의 브런치로 협업을 하지 않음!

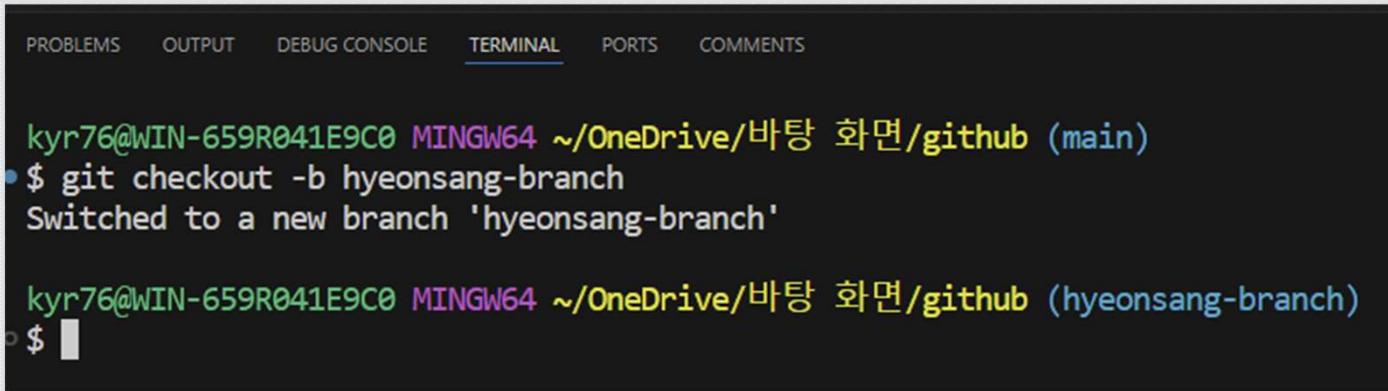


기능마다 브런치를 나누고, 최대한 효율적인 작업을 진행 함.

그러면 왜 여러 브런치로 나눠서 작업을 하면 효과 적일까?

깃헙 브런치 예제

브런치가 효과적인 이유를 보기 전, 브런치를 어떻게 만드는지 알아봅시다.



The screenshot shows a terminal window with the following interface elements at the top:

- PROBLEMS
- OUTPUT
- DEBUG CONSOLE
- TERMINAL** (underlined)
- PORTS
- COMMENTS

The terminal content is as follows:

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ git checkout -b hyeonsang-branch
Switched to a new branch 'hyeonsang-branch'

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (hyeonsang-branch)
$
```

git checkout -b 브런치이름 : 새로운 브런치를 만드는 명령어!

(주의할 점 로컬에만 브런치가 만들어진 것! 리모트에도 만들어졌다고 생각하면 안 됩니다.)

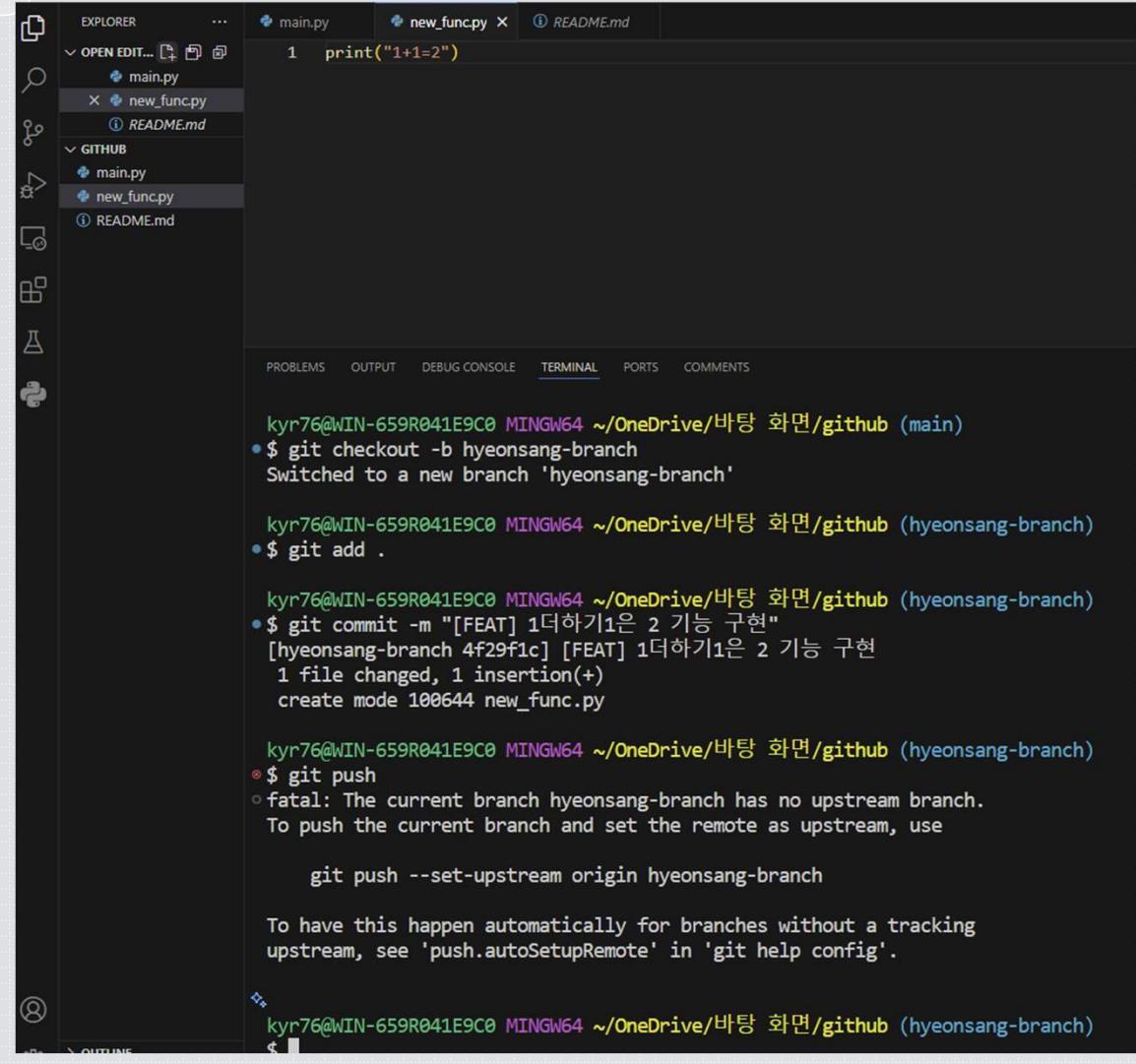
깃헙 브랜치 예제

The screenshot shows a VS Code interface with the following details:

- Explorer View:** Shows files in the current workspace:
 - OPEN EDITORS:** main.py, new_func.py (highlighted), README.md
 - GITHUB:** main.py, new_func.py (highlighted), README.md
- Terminal Tab:** TERMINAL is selected.
- Terminal Output:**
 - Switched to a new branch 'hyeonsang-branch'
 - Added all files in the current directory.
 - Committed changes with a message "[FEAT] 1더하기1은 2 가능 구현".
 - Created a new file named new_func.py.

이전과 동일하게, add -> commit 까지 진행합니다.

깃헙 브런치 예제



```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ git checkout -b hyeonsang-branch
Switched to a new branch 'hyeonsang-branch'

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (hyeonsang-branch)
$ git add .

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (hyeonsang-branch)
$ git commit -m "[FEAT] 1더하기1은 2 기능 구현"
[hyeonsang-branch 4f29f1c] [FEAT] 1더하기1은 2 기능 구현
 1 file changed, 1 insertion(+)
 create mode 100644 new_func.py

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (hyeonsang-branch)
$ git push
fatal: The current branch hyeonsang-branch has no upstream branch.
To push the current branch and set the remote as upstream, use

  git push --set-upstream origin hyeonsang-branch

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.
```

한 가지 다른 점은
git push를 했을 때
git push --set-upstream origin 브런치이름
이 등장합니다.

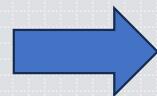
이 명령어는 해당 로컬에서 작업한 브런치를 리모트에 생성하고, 리모트와 로컬을 연결 시킬까요?

라는 의미입니다.

git push --set-upstream origin 브런치이름
를 한 번 하게 된다면
그 다음부터 git push만 해도 푸쉬 됨!

깃헙 브랜치 예제

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (hyeonsang-branch)
$ git push --set-upstream origin hyeonsang-branch
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 22 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 373 bytes | 373.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'hyeonsang-branch' on GitHub by visiting:
remote:     https://github.com/hyeonsang010716/test-repo/pull/new/hyeonsang-branch
remote:
To https://github.com/hyeonsang010716/test-repo.git
 * [new branch]      hyeonsang-branch -> hyeonsang-branch
branch 'hyeonsang-branch' set up to track 'origin/hyeonsang-branch'.
```



The screenshot shows the GitHub interface for switching branches. At the top, it says "main" (selected) and "3 Branches 0 Tags". Below that is a search bar with "Find or create a branch...". Under "Branches", there are three entries: "main" (selected), "hyeonsang-branch" (highlighted with a red border), and "test". A "Tags" tab is also visible. At the bottom, there's a "View all branches" link.

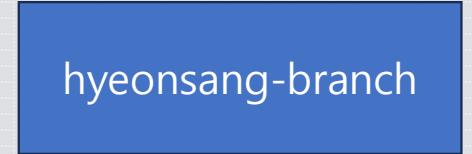
The screenshot shows the GitHub repository page for the "hyeonsang-branch". The top navigation bar includes a dropdown for "hyeonsang-branch" (highlighted with a red border), "3 Branches 0 Tags", a search bar, "Add file", and "Code". Below the header, it says "This branch is 1 commit ahead of main". A list of commits is shown:

Author	Commit Message	Time	Commits
hyeonsang010716	[FEAT] 1더하기1은 2 기능 구현	4f29f1c · 6 minutes ago	17 Commits
	README.md	Update README.md	22 minutes ago
	main.py	[FEAT] hello world 구현	18 minutes ago
	new_func.py	[FEAT] 1더하기1은 2 기능 구현	6 minutes ago

깃헙 브런치 예제

현재 $1+1=2$ 기능을 hyeonsang-branch로 구현한 상태.

그 다음 해줘야 할 작업은 hyeonsang-branch를 main에 합쳐야 함.
(pull request (풀리퀘스트) : 브런치를 합친다)



hyeonsang-branch

깃헙 브랜치 예제

The screenshot shows a GitHub interface for a repository. At the top, there are navigation links: Code, Issues, Pull requests (which is highlighted with a red box), Actions, Projects, Security, Insights, and Settings. Below the navigation bar, a notification box says "hyeonsang-branch had recent pushes 7 minutes ago". To the right of the notification is a green button labeled "Compare & pull request". Underneath the notification is a search bar with the query "is:pr is:open" and filter options. To the right of the search bar are buttons for Labels (9) and Milestones (0). A prominent green button labeled "New pull request" is located on the far right, also highlighted with a red box. In the center of the page, there is a large white box containing a "Welcome to pull requests!" message, a small icon of two people, and a paragraph explaining how pull requests help collaborate on code. At the bottom of the page, there is a "ProTip!" section with a note about using the 'g' and 'i' keyboard shortcuts.

Code Issues Pull requests Actions Projects Security Insights Settings

hyeonsang-branch had recent pushes 7 minutes ago

Compare & pull request

Filters is:pr is:open Labels 9 Milestones 0 New pull request

Welcome to pull requests!

Pull requests help you collaborate on code with other people. As pull requests are created, they'll appear here in a searchable and filterable list. To get started, you should [create a pull request](#).

ProTip! Type `g` `i` on any issue or pull request to go back to the issue listing page.

깃헙 브랜치 예제

Compare changes

Compare changes across branches, commits, tags, and more below. If you need to, you can also [compare across forks](#).

base: main ← compare: main ▾

Choose different branch

Choose a head ref

Find a branch

Learn about pull requests

Create pull request

Branches Tags

✓ main default

hyeonsang-branch

test

Compare changes across branches, commits, tags, and more below. In the same repository and across forks.

Example comparisons

hyeonsang-branch	13 minutes ago
test	last week
main@{1day}...main	24 hours ago

깃헙 브랜치 예제

The screenshot shows a GitHub pull request page for a repository. The title of the pull request is "1더하기1 기능 구현 #1". The status is "Open" and it shows "hyeonsang010716 wants to merge 1 commit into main from hyeonsang-branch". The pull request has 0 conversations, 1 commit, 0 checks, and 1 file changed. The commit message is "[FEAT] 1더하기1은 2 기능 구현". The commit hash is 4f29f1c. The pull request is automatically mergeable with no conflicts with the base branch. There is a "Merge pull request" button and a note that you can also merge it with the command line. The pull request is still in progress, with no reviews or assignees. It has no labels, projects, or milestones. Notifications are customized, and there is an "Unsubscribe" button.

1더하기1 기능 구현 #1

Open hyeonsang010716 wants to merge 1 commit into main from hyeonsang-branch

Conversation 0 Commits 1 Checks 0 Files changed 1 +1 -0

hyeonsang010716 commented now

No description provided.

[FEAT] 1더하기1은 2 기능 구현 4f29f1c

No conflicts with base branch

Merging can be performed automatically.

Merge pull request You can also merge this with the command line. View command line instructions.

Add a comment

Write Preview

Add your comment here...

Markdown is supported Paste, drop, or click to add files

Close pull request Comment

Remember, contributions to this repository should follow our GitHub Community Guidelines.

Reviewers No reviews Still in progress? Learn about draft PRs

Assignees No one Assign yourself

Labels None yet

Projects None yet

Milestone No milestone

Development Successfully merging this pull request may close these issues.

None yet

Notifications Customize Unsubscribe

You're receiving notifications because you're watching this repository.

여기까지 뜨면 성공!
(merge pull request는 팀장 권한이어서 위처럼 안 나올 겁니다!)

깃헙 브런치 예제

The screenshot shows a GitHub repository named "test-repo". The repository is private, has 3 branches, and 0 tags. There is one merge pull request from "hyeonsang010716/hyeonsang-branch" into the main branch. The most recent commit is a merge pull request from "hyeonsang010716/hyeonsang-branch" at 7ebfa54 · now, containing 18 commits. The commit history includes:

- Update README.md (34 minutes ago)
- [FEAT] hello world 구현 (30 minutes ago)
- [FEAT] 1더하기1은 2 가능 구현 (17 minutes ago)

The README file contains the text "test-repo" and "A개발자".

팀장이 수락을 했다면 main에 위치럼 반영 됨!

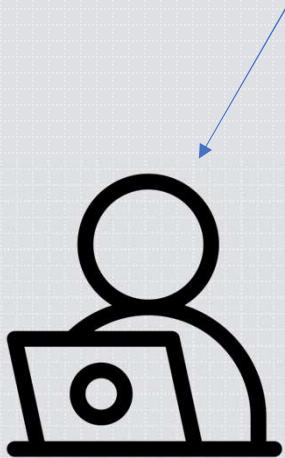
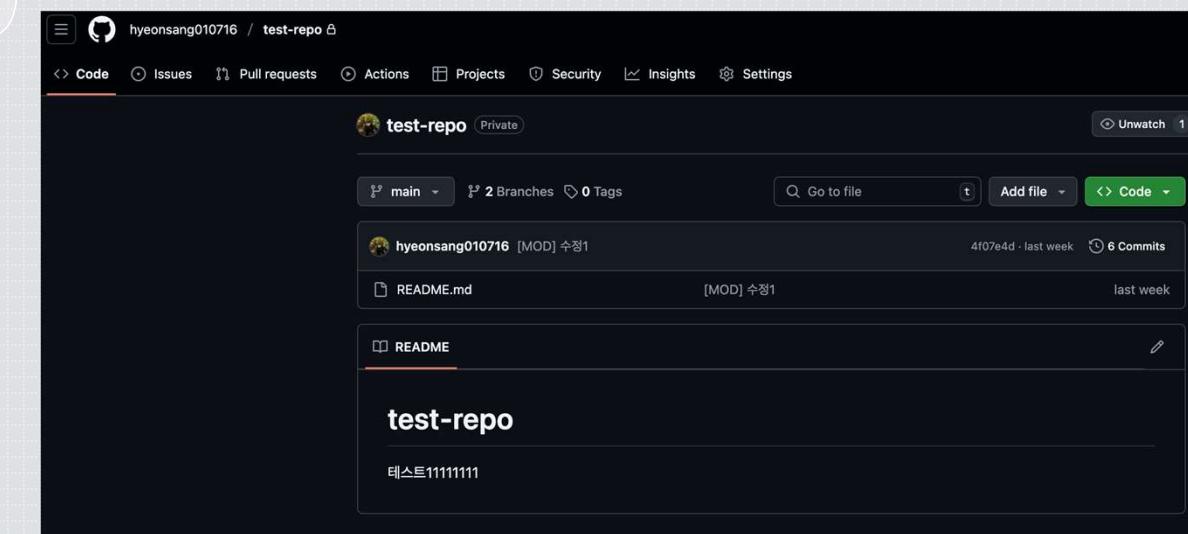
깃헙 브랜치 예제

자.. 그래서 왜 branch를 나눠서 관리를 하면 좋을까..

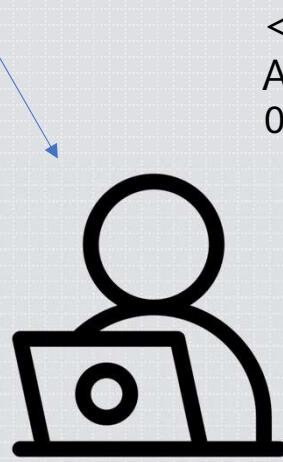


1. main에 합치기 전, 코드 리뷰를 할 수 있음.
 1. 어떤 파일을 수정했는지
 2. 커밋은 잘했는지
 3. 누가 무슨 코드를 수정했고, 언제 수정했는지 다 알 수 있음!
 4. 팀장 개발자가 마음에 안 들면 무한 수정 시작..
2. 하지만 제일 좋은 장점은 히스토리 충돌이 사라짐

깃헙 브런치 예제



A개발자



B개발자

히스토리

[MOD] 수정1

main

A

B

<시나리오>

A개발자와 B개발자가 동시에 리모트 main 브런치에서 각자 브런치를 생성하고 기능 구현을 시작함

깃헙 브런치 예제

```
READER  
OPEN EDIT...  
X README.md  
X README.md  
GITHUB  
README.md  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS  
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (a/readme)  
• $ git add .  
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (a/readme)  
• $ git commit -m "[MOD] 리드미 수정"  
[a/readme 00ab336] [MOD] 리드미 수정  
1 file changed, 1 insertion(+), 1 deletion(-)  
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (a/readme)  
• $ git push  
fatal: The current branch a/readme has no upstream branch.  
To push the current branch and set the remote as upstream, use  
git push --set-upstream origin a/readme  
To have this happen automatically for branches without a tracking  
upstream, see 'push.autoSetupRemote' in 'git help config'.  
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (a/readme)  
• $ git push --set-upstream origin a/readme  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Writing objects: 100% (3/3), 303 bytes | 303.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
remote:  
remote: Create a pull request for 'a/readme' on GitHub by visiting:  
remote: https://github.com/hyeonsang010716/test-repo/pull/new/a/readme  
remote:  
To https://github.com/hyeonsang010716/test-repo.git  
 * [new branch]      a/readme -> a/readme  
branch 'a/readme' set up to track 'origin/a/readme'.  
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (a/readme)
```

test-repo Private

a/readme had recent pushes 45 seconds ago

Compare & pull request

a/readme 2 Branches 0 Tags

Find or create a branch...

Switch branches/tags

Branches Tags

main (default)

a/readme

00ab336 · 1 minute ago 20 Commits

[MOD] 리드미 수정 · 1 minute ago

Contribute

A개발자

Pull requests 1 Actions Projects Security Insights Settings

Filters is:pr is:open

1 Open ✓ 1 Closed

A개발자 리드미 수정 #2 opened now by hyeonsang010716

깃헙 브랜치 예제

The screenshot shows a dark-themed code editor interface. On the left, the Explorer sidebar lists files: README.md, main.py, and a GitHub folder containing main.py and README.md. The main area displays the content of main.py:

```
1 print("hello world")
```

Below the code editor is a terminal window showing the following git session:

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (main)
$ git checkout -b b/hello-word
Switched to a new branch 'b/hello-word'

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (b/hello-word)
$ git add main.py

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (b/hello-word)
$ git commit -m "[FEAT] hello world 기능 구현"
[b/hello-word f22f848] [FEAT] hello world 기능 구현
 1 file changed, 1 insertion(+)
 create mode 100644 main.py

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/github (b/hello-word)
$ git push
fatal: The current branch b/hello-word has no upstream branch.
To push the current branch and set the remote as upstream, use

  git push --set-upstream origin b/hello-word

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.
```

At the bottom, an outline panel is visible.

The screenshot shows a GitHub repository named "test-repo". The repository is private and has 3 branches and 0 tags. A modal dialog titled "Switch branches/tags" is open, showing the current branch "b/hello-word" as the default. Other branches listed are "main" and "a/readme".

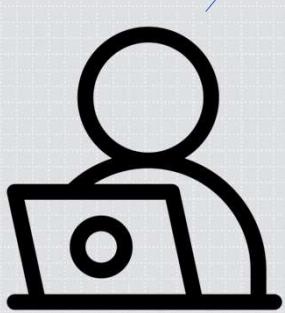
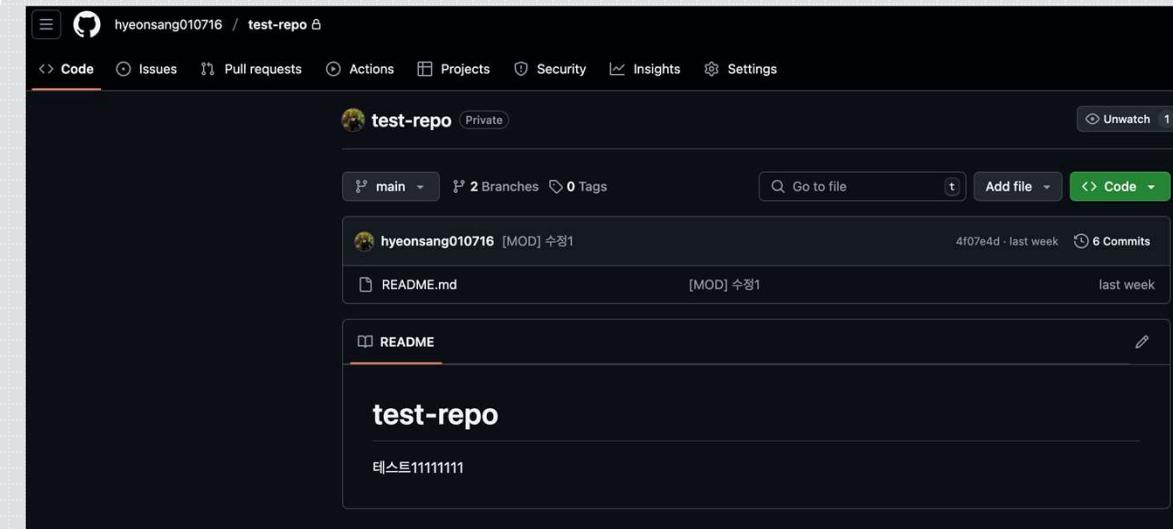
The repository page shows the following details:

- Recent activity: "b/hello-word" had recent pushes 38 seconds ago.
- Branches: 3 Branches (main, a/readme, b/hello-word)
- Tags: 0 Tags
- Last commit: f22f848 · now (20 Commits)
- Recent commits:
 - [MOD] README 수정 (6 minutes ago)
 - [FEAT] hello world 기능 구현 (now)
- Test message: 테스트11111111

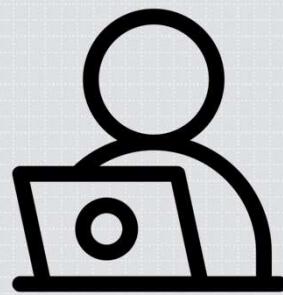
At the bottom, the "Pull requests" tab is active, showing 2 open pull requests:

- #3: b 개발자 hello world 기능 구현 완료~ (opened now by hyeonsang010716)
- #2: A개발자 리드미 수정 (opened 2 minutes ago by hyeonsang010716)

깃헙 브랜치 예제



A개발자



B개발자

<시나리오>
동시에 기능 구현 완료를 해서, 풀리퀘스트를 보냄

히스토리



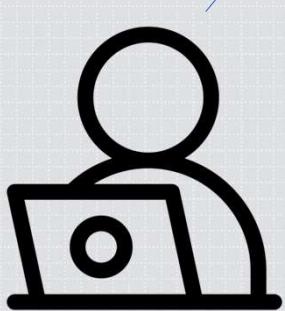
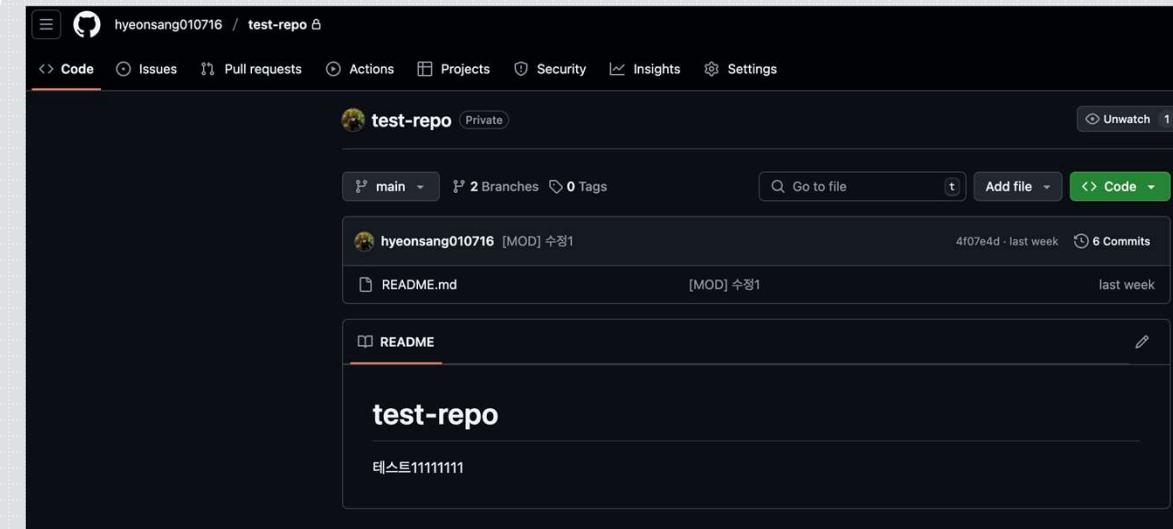
[MOD] 수정1

main

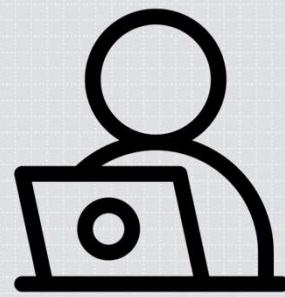
A

B

깃헙 브랜치 예제



A개발자



B개발자

<시나리오>
동시에 기능 구현 완료를 해서, 풀리퀘스트를 보냄

히스토리



[MOD] 수정1

main

A

B

깃헙 브런치 예제

The screenshot shows a GitHub repository named 'test-repo' owned by 'hyeonsang010716'. The repository is private and has 3 branches and 0 tags. A merge pull request from 'hyeonsang010716/a/readme' was merged at commit ce5f02b · now, containing 23 commits. Two files were updated: 'README.md' (MOD) and 'main.py' (FEAT). The README file contains the text 'test-repo' and 'A개발자'.

Code Issues Pull requests Actions Projects Security Insights Settings

Unwatch 1

main 3 Branches 0 Tags Go to file Add file Code

hyeonsang010716 Merge pull request #2 from hyeonsang010716/a/readme ce5f02b · now 23 Commits

README.md [MOD] 리드미 수정 8 minutes ago

main.py [FEAT] hello world 기능 구현 4 minutes ago

README

test-repo

A개발자

동시에 풀리퀘스트를 보냈지만,
팀장 개발자가 merge만 하면 충돌X 반영 완료!

깃협 실습

이번 주 실습!

1. **feat/깃협아이디-main** 으로 브런치 만들기 ex) feat/hyeonsang010716-main
2. 자신의 디렉토리 (이전 실습으로 만든 디렉토리)에 **main.py** 생성
3. git add 한 후, git commit -m “[FEAT] main 생성” 으로 커밋하고 푸쉬하기
4. 풀리퀘스트까지 보내면 실습 끝!



깃협 실습

이번 주 실습!

(먼저 git pull origin main을 해서 업데이트 된 main을 가져와주세요!)

1. **feat/깃협아이디-main** 으로 브런치 만들기 ex) feat/hyeonsang010716-main
2. 자신의 디렉토리 (이전 실습으로 만든 디렉토리)에 **main.py** 생성
3. git add 한 후, git commit -m “[FEAT] main 생성” 으로 커밋하고 푸쉬하기
4. 풀리퀘스트까지 보내면 실습 끝!



깃협 실습

이번 주 실습!

(먼저 git pull origin main을 해서 업데이트 된 main을 가져와주세요!)

1. **feat/깃협아이디-main** 으로 브런치 만들기 ex) feat/hyeonsang010716-main
2. 자신의 디렉토리 (이전 실습으로 만든 디렉토리)에 **main.py** 생성
3. git add 한 후, git commit -m “[FEAT] main 생성” 으로 커밋하고 푸쉬하기
4. 풀리퀘스트까지 보내면 실습 끝!

깃헙 실습

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows a tree structure of a Git repository named "GIT-STUDY".
 - bigwave100
 - docs
 - week1.pdf
 - HwangInseok
 - README.md
 - hwlee
 - hyeonsang010716
 - README.md (selected)
 - jungsb0415
 - kyuhyung
 - pinkgorae
 - wjm9765
 - xxiv-34
 - yjs2673

Terminal Tab: Active tab, showing the command-line interface output.

Terminal Output:

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (main)
• $ git checkout -b feat/hyeonsang010716-main
Switched to a new branch 'feat/hyeonsang010716-main'

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (feat/hyeonsang010716-main)
◦ $ █
```

깃헙 실습

The screenshot shows a dark-themed instance of Visual Studio Code (VS Code) with the following interface elements:

- Explorer View:** On the left, it lists several projects and files. A folder named "GIT-STUDY" contains subfolders "bigwave100", "docs" (which includes "week1.pdf"), "HwangInseok" (containing "README.md"), "hwlee", and "hyeonsang010716-main" (containing "main.py"). Other listed items include "jungsb0415", "kyuhyoung", "pinkgorae", "wjm9765", "xxxiv-34", and "yjs2673".
- Terminal Tab:** The active tab at the bottom is "TERMINAL". It displays the following terminal session:

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (main)
• $ git checkout -b feat/hyeonsang010716-main
Switched to a new branch 'feat/hyeonsang010716-main'

kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (feat/hyeonsang010716-main)
$
```
- Code Editor:** The main area shows a file named "main.py" with the following content:

```
1  Press Ctrl+I to ask Code to do something. Start typing to dismiss.
```

깃헙 실습

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (feat/hyeonsang010716-main)
```

- \$ git add hyeonsang010716/main.py

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (feat/hyeonsang010716-main)
```

- \$ git commit -m "[FEAT] main 생성"
[feat/hyeonsang010716-main 4a946ab] [FEAT] main 생성
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 hyeonsang010716/main.py

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (feat/hyeonsang010716-main)
```

- \$ git push
fatal: The current branch feat/hyeonsang010716-main has no upstream branch.
To push the current branch and set the remote as upstream, use

git push --set-upstream origin feat/hyeonsang010716-main

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

```
kyr76@WIN-659R041E9C0 MINGW64 ~/OneDrive/바탕 화면/git-study (feat/hyeonsang010716-main)
```

- \$ git push --set-upstream origin feat/hyeonsang010716-main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 22 threads

깃헙 실습

The screenshot shows the GitHub interface for a repository named "SGCS-Release-Git-Project / Git-Study". The "Pull requests" tab is selected, indicating there is 1 open pull request. The search bar contains the query "is:pr is:open". The pull request listed is titled "[FEAT] main 생성", which was opened by "hyeonsang010716" and requires review. The GitHub footer includes links to Terms, Privacy, Security, Status, Docs, Contact, Manage cookies, and a note about not sharing personal information.

SGCS-Release-Git-Project / Git-Study

Code Issues Pull requests 1 Actions Projects Wiki Security Insights Settings

Filters is:pr is:open Labels 9 Milestones 0 New pull request

1 Open 0 Closed

[FEAT] main 생성 #1 opened now by hyeonsang010716 • Review required

ProTip! Exclude your own issues with [-author:hyeonsang010716](#).

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

언제든지 자유롭게 물어보기!

A screenshot of a GitHub Issues page. The top navigation bar includes links for Code, Issues (which is highlighted with a red box), Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the navigation is a search bar containing the query "is:issue state:open". To the right of the search bar are buttons for Labels, Milestones, and a green "New issue" button, which is also highlighted with a red box. A "Preview" link is located above the Milestones button. The main content area shows filters for Open (0) and Closed (0) issues, along with dropdown menus for Author, Labels, Projects, Milestones, Assignees, Types, and sorting by Newest. The central message reads "No results" with the sub-instruction "Try adjusting your search filters."

질문이 생기면 주저 말고 GitHub Issue에 남겨주세요 ;)