

# Git (2)

발표자: 양유진

링크: <https://youtu.be/Aj0rQ5uoxWs>

Git Branch와 merge

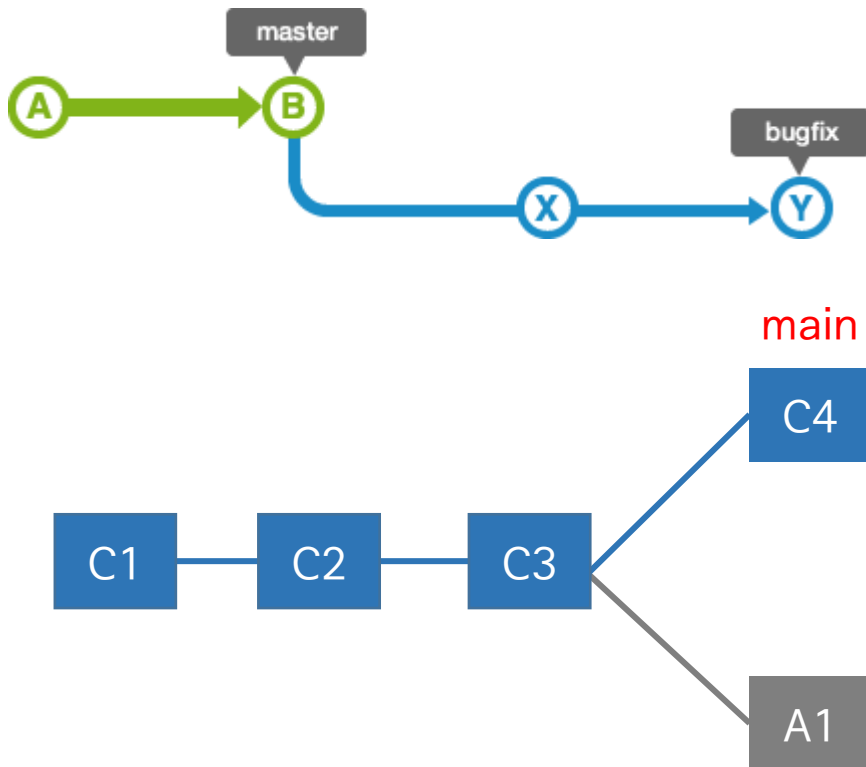
Git 협업 - (1)~(8)

# Git Branch와 merge

## 브랜치(Branch)

: 특정 commit을 가리키는 일종의 포인터

- branch에 담긴 commit 중 가장 마지막 commit을 가리킴



## 브랜치 병합(Merge)

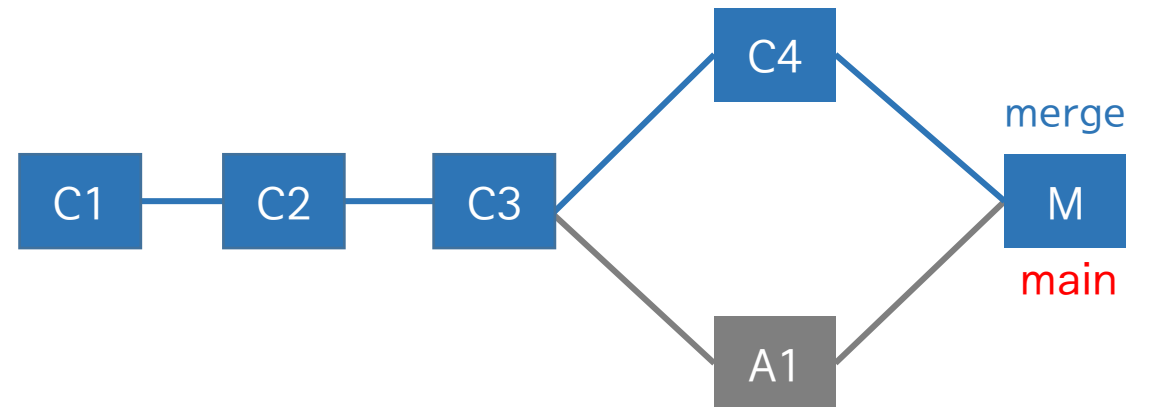
: branch를 하나로 합치는 방법

- 1) fast-forward(빨리 감기) 병합

- 브랜치가 main 브랜치의 이력을 모두 포함하고 있어 단순히 이동하기만 하면 되는 병합

- 2) non fast-forward 병합

- 각 브랜치의 변경 내용을 하나로 통합하는 병합
- 브랜치를 삭제하지 않는 한, 통합하더라도 그대로 남음.



# Git 협업 (1) 초대하기

Settings > Manage access > Invite a collaborator > 초대할 계정 이메일 입력

The screenshot displays the GitHub repository settings interface. On the left, a sidebar menu lists various settings: Options, Manage access (highlighted with a red box), Security & analysis, Branches, Webhooks, Notifications, Integrations, Deploy keys, Actions, Environments, and Secrets. The main content area is titled 'Who has access' and contains two panels: 'PUBLIC REPOSITORY' (with an eye icon) stating 'This repository is public and visible to anyone.' with a 'Manage' link, and 'DIRECT ACCESS' (with a person icon) stating '0 collaborators have access to this repository. Only you can contribute to this repository.' Below these panels is the 'Manage access' section, which shows a message 'You haven't invited any collaborators yet' accompanied by an icon of a person with a lock. A red box highlights the 'Invite a collaborator' button. On the right side, a modal window titled 'Invite a collaborator to git\_test' is open, featuring a search bar with the email 'yujin.yang34@gmail.com' and a blue button with an envelope icon and the text 'yujin.yang34@gmail.com Invite to git\_test'.

Options

Manage access

Security & analysis

Branches

Webhooks

Notifications

Integrations

Deploy keys

Actions

Environments

Secrets

### Who has access

**PUBLIC REPOSITORY**

This repository is public and visible to anyone.

[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

[Invite a collaborator](#)

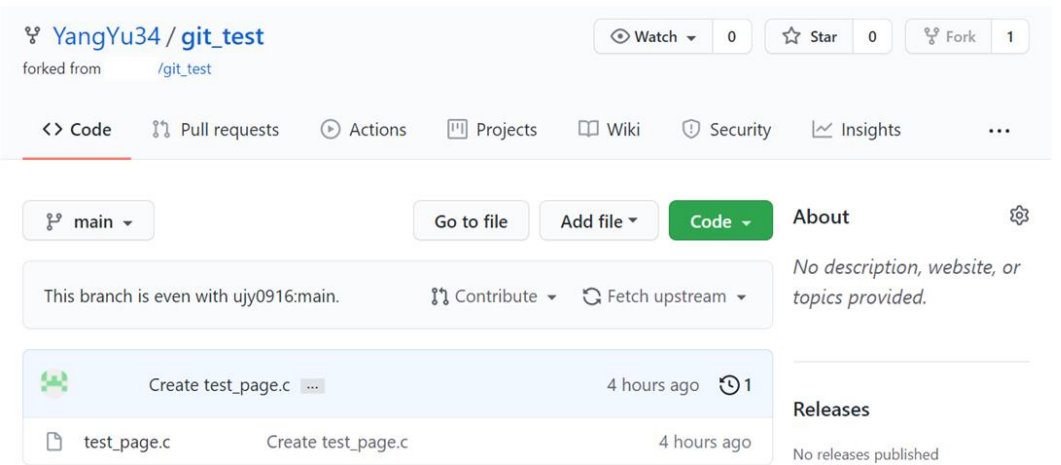
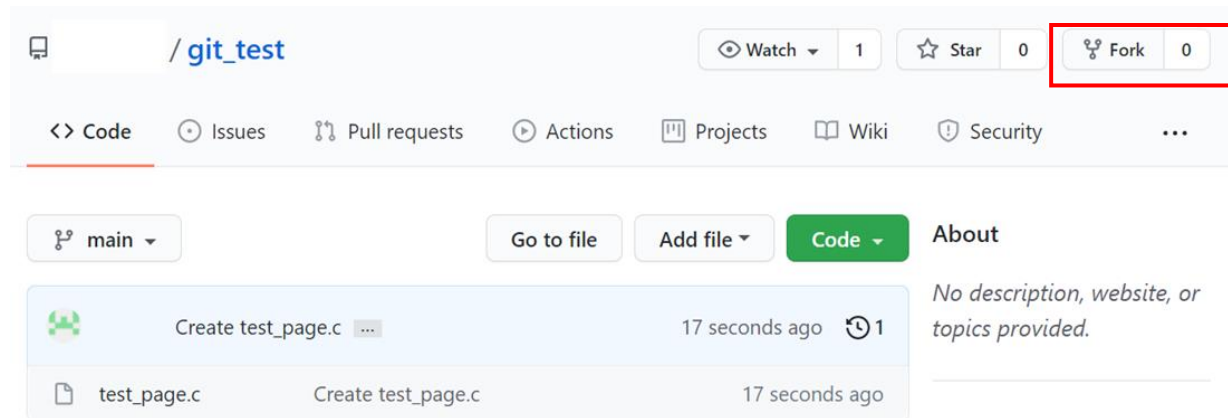
### Invite a collaborator to git\_test

yujin.yang34@gmail.com

yujin.yang34@gmail.com  
Invite to git\_test

# Git 협업 (2) Fork

원본 Project를 본인의 Github로 복제해 오는 것.



# Git 협업 (3) Clone

`git clone` [본인의 git repository주소]

:특정 원격 저장소와 로컬 PC의 저장소를 연결하고 데이터를 복사하여 가져옴.

```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git (master)
$ git clone https://github.com/YangYu34/git_test.git
Cloning into 'git_test'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

# Git 협업 (4) upstream 저장소 추가

**upstream** - [상대적개념]Fork해온 Github 저장소

Fork해온 Github 저장소 이름=복제한 Github 저장소 이름=origin 경우,  
둘을 구분하기 위하여 Fork해온 Github 저장소를 upstream이라 부름.

**git remote add upstream [협업하는 git repository주소]**

: upstream 저장소 추가

```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (main)
$ git remote add upstream https://github.com/      /git_test.git
```

**git remote -v**

: remote 저장소 목록 확인

```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (main)
$ git remote -v
origin  https://github.com/YangYu34/git_test.git (fetch)
origin  https://github.com/YangYu34/git_test.git (push)
upstream      https://github.com/      /git_test.git (fetch)
upstream      https://github.com/      /git_test.git (push)
```

# Git 협업 (5) branch 생성 & remote repository에 반영

**git checkout -b [branch명]**

: 새로운 branch 생성

- 생성과 동시에 새로 생성된 branch로 변경됨

```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (main)
$ git checkout -b yujin
Switched to a new branch 'yujin'
```

add > commit > push

**git add .**

: untracked 파일 전부 working directory → staging area 추가

**git commit -m "[commit message]"**

: staging area에 있는 정보 commit하여 local repository에 변경사항 적용

**git push origin [branch명]**

: remote repository(origin)에 파일 업로드

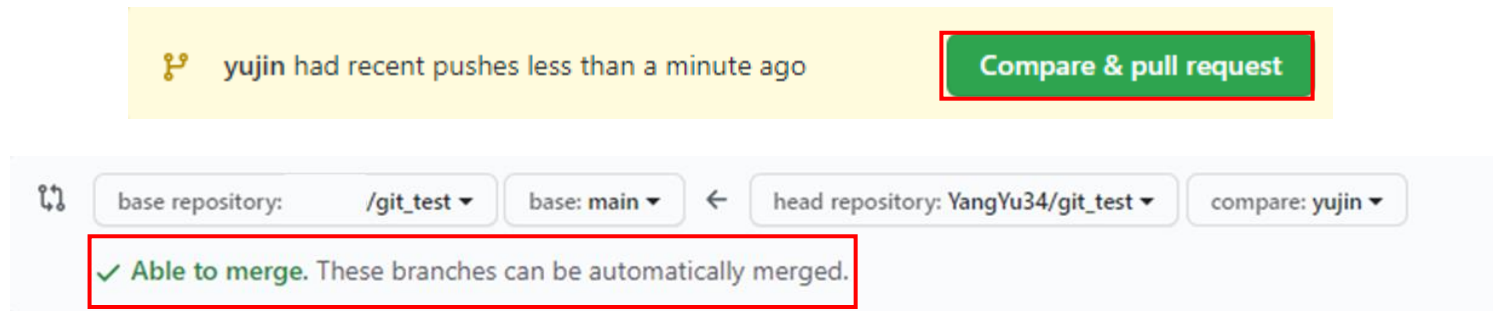
```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (yujin)
$ git push origin yujin
```



# Git 협업 (6) PR(Pull Request) 생성

**Pull Request** - 원본 Repository에 commit 반영할 것을 요청하는 작업.

- 협업에서 가장 중요한 기능
- 팀원들에게 확인을 받을 수도 있고(PR review), 자동으로 확인을 받을 수도 있다.(Action)
- 리뷰할 팀원을 지정할 수도 있음.
- Reviewer로 지정된 팀원은 Add your review 버튼을 눌러 의견을 남길 수 있음.



× **Can't automatically merge.** Don't worry, you can still create the pull request.

Conflict가 발생한 경우 해당 메시지가 뜸.

- 여러 사람이 같은 파일을 작업하여 merge하면 conflict 발생.

# \*\*Pull Request template 만들기

Create new file > pull\_request\_template.md 작성

main [git\\_test / pull\\_request\\_template.ma](#)

Create pull\_request\_template.ma ...

1 contributor

7 lines (4 sloc) | 113 Bytes

```
1 ## 수정사항
2 - 여기에 작성해 주세요
3
4
5 ## 대략적인 코드 설명
6 - 여기에 작성해 주세요
7
```

Modified error, Add print

Write Preview

H B I

## 수정사항  
- 여기에 작성해 주세요

## 대략적인 코드 설명  
- 여기에 작성해 주세요

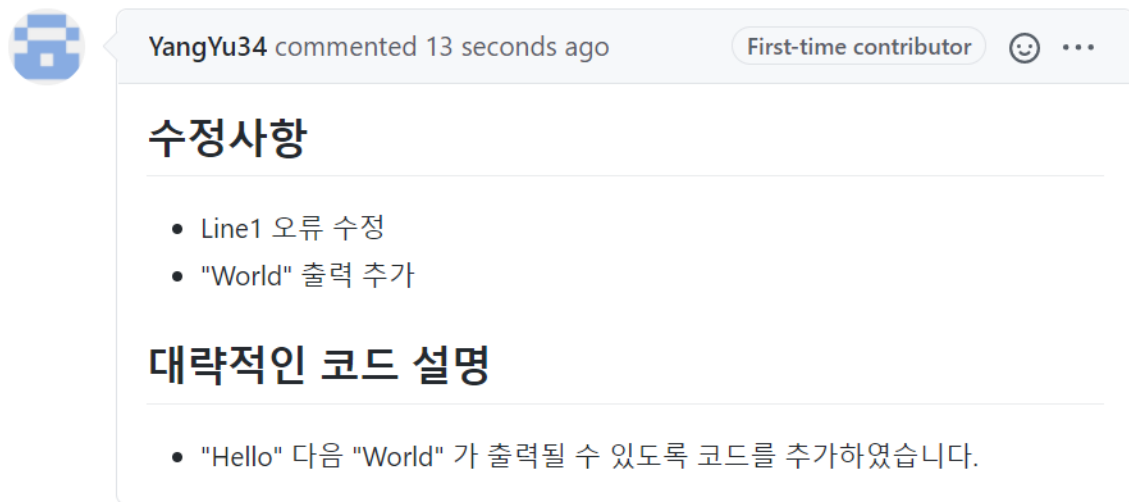
Attach files by dragging & dropping, selecting or pasting them.

☒ Allow edits by maintainers

Create pull request

# Git 협업 (7) 검토 및 merge

## 1. Pull requests



YangYu34 commented 13 seconds ago First-time contributor

### 수정사항

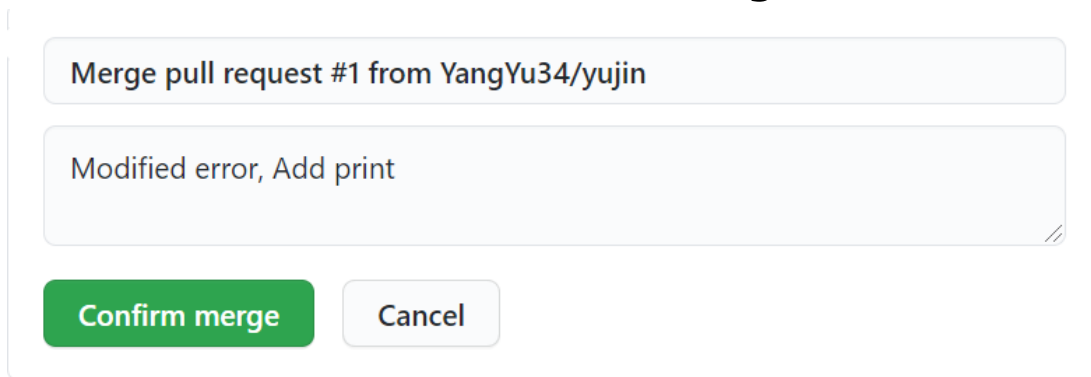
- Line1 오류 수정
- "World" 출력 추가

### 대략적인 코드 설명

- "Hello" 다음 "World" 가 출력될 수 있도록 코드를 추가하였습니다.

## ※ merge는 관리자만 가능한 작업

### 3. Click Conform merge

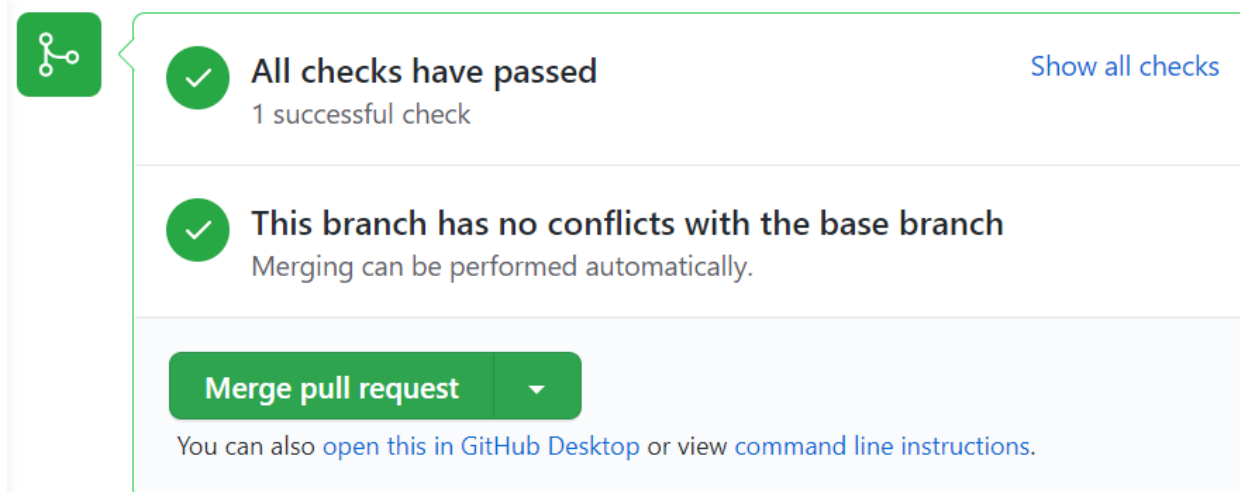


Merge pull request #1 from YangYu34/yujin

Modified error, Add print

**Confirm merge** Cancel

## 2. Click Merge pull request



**All checks have passed** 1 successful check [Show all checks](#)

**This branch has no conflicts with the base branch**  
Merging can be performed automatically.

**Merge pull request** ▼

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

## Modified error, Add print #1

**Merged** merged 1 commit into :main from YangYu34:yujin now

# Git 협업 (8) repository 동기화-1

## git fetch upstream

: upstream(협업 github 저장소)에서 원본 소스코드를 로컬로 가져옴.

```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (yujin)
$ git fetch upstream
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 1.41 KiB | 25.00 KiB/s, done.
From https://github.com/ujy0916/git_test
* [new branch]      main      -> upstream/main
```

## git checkout main

```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (yujin)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

## fetch

원격 저장소에서 코드를 받아오는 작업만 함.  
(자동 병합X)

## pull

연결된 remote repository에서 코드를 받아와(fetch) 자동으로 병합하여 줌. (merge)

## clone

지정한 remote repository에서 코드를 받아와 자동으로 병합하여 줌.

# Git 협업 (8) repository 동기화-2

**git merge upstream/main**

: upstream을 local repository(main)에 merge 시켜줌.

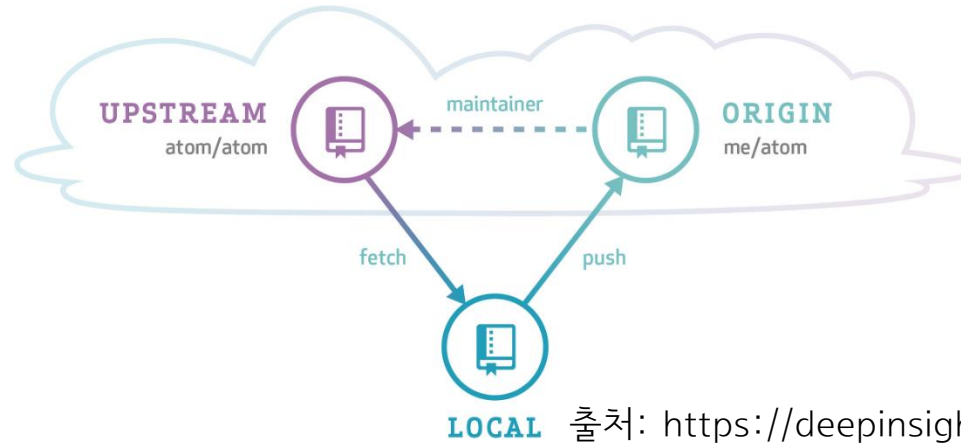
```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (main)
$ git merge upstream/main
Updating 1c4c1f4..da647f3
Fast-forward
 pull_request_template.ma | 7 ++++++
 test_page.c               | 3 ++-
 2 files changed, 9 insertions(+), 1 deletion(-)
 create mode 100644 pull_request_template.ma
```

**git push**

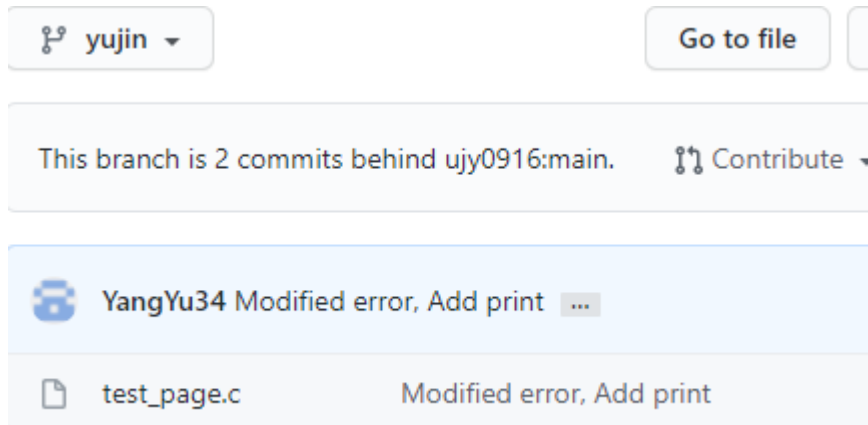
: 원격저장소 origin repository에 반영.

```
yujin.yang@DESKTOP-9EVEH5F MINGW64 ~/Desktop/Git/git_test (main)
$ git push
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/YangYu34/git_test.git
 1c4c1f4..da647f3  main -> main
```

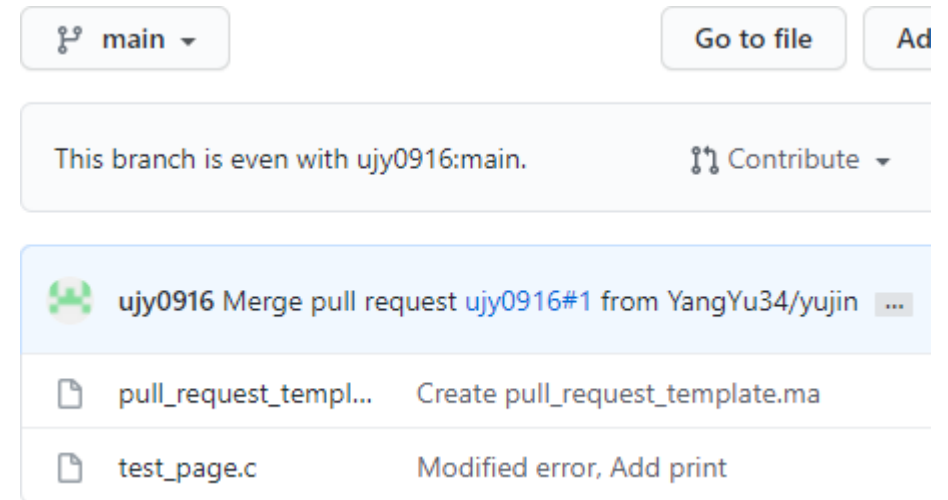
# Git 협업 (8) repository 동기화



출처: <https://deepinsight.tistory.com/167>



branch 'yujin'



branch 'main' (병합됨)

감사합니다