

효율적인 프로젝트 협업을 위한 **Git & Github** 활용 방법

데이터베이스프로그래밍 / 소프트웨어시스템개발

목차

01 Git

1. 필요성
2. Git
3. Github
4. 분산버전관리

02 저장소

1. 저장소란
2. 실습-Github에 Repository생성

03 명령어

1. Add
2. Commit
3. Push
4. 실습

04 Git Flow

1. Branch란
2. Git Flow란

목차

05 Eclipse-Git

준비

1. Egit설치

팀장

1. 저장소 생성
2. Gitignore 꼭 추가
3. Manage access 설정
4. Branch 생성-develop

팀원

1. Clone
2. Fork (선택사항)
3. e-mail에서 팀 권한 수락

06 협업

1. 협업방식 적용 이유
2. 장점

협업(공통) -반복

1. Issue
2. Branch 생성 및 작업
3. Commit & push
4. Pull request (=PR)
5. 코드리뷰
6. Merge
7. Checkout branch develop
8. Pull
9. 작업이 끝난 Branch 삭제
10. 정리

07 주의사항

1. Pull 오류해결
2. Merge 충돌해결

Git PRESENTATION

01 Git

1.1 Git 필요성



어떤 파일이 최종이지??

최종제출 파일이라고 생각했지만

오류가 발생 또는 새로운 기능을 추가해야 할 경우가 발생한다.

계속해서 파일이름에 최종을 붙여 만들면

어떤 파일이 최종인지 찾는 것이 쉽지 않다.

누가 하나의 파일로 합칠래??

하나의 파일로 일일이 합치는 것도 힘들고

합치는 과정에서 오류가 나기 쉽다.

시간이 아깝다.

이 외에도 Git을 사용시 장점이 많다.

1.2 Git



Git이란?

- 코드 저장소
 - 이력을 관리하는 저장소
 - Version Control System
 - Code를 저장 및 저장지점으로 되돌아갈 수 있도록 해주는 시스템
 - 분산 버전 관리 시스템
- =>여러 개발 PC와 저장소에 분산해서 저장

1.3 GitHub



GitHub란?

-코드저장소(Git)을 웹에 옮겨 놓은 원격저장소

웹에 있을 때의 장점?

- 공유 가능(협업)
- 코드 복구 가능

1.4 분산버전관리

버전 관리란 ?

파일 변화를
시간에 따라 기록했다
특정 시점의 버전을 다시 가져올 수 있는 것

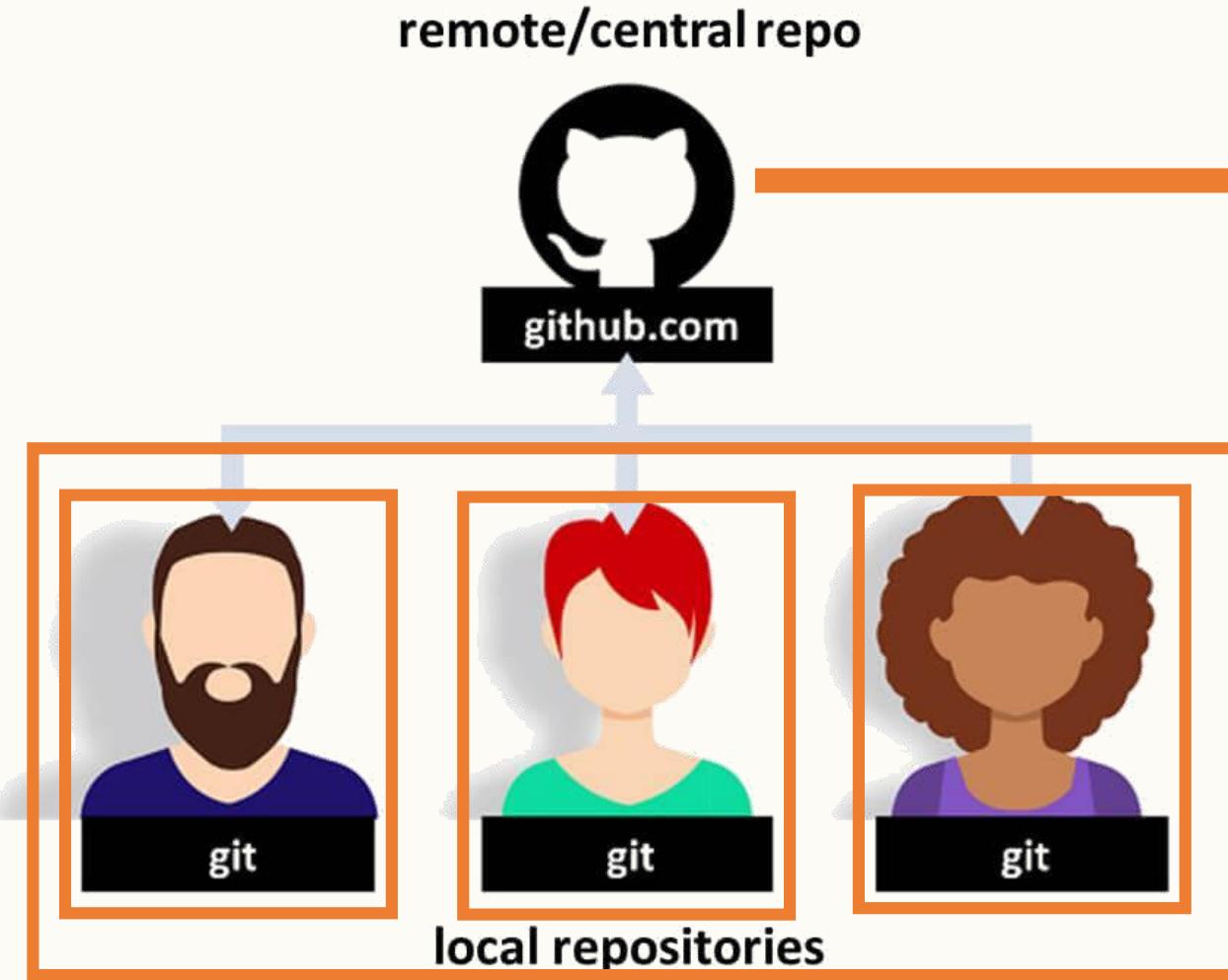
장점

1. 소스 코드가 변경된 이력 쉽게 확인 가능
2. 특정 시점에 저장된 버전과 비교 가능
3. 특정 시점으로 되돌아갈 수 있음
4. 내가 올리려는 파일이 누군가 편집한 내용과 충돌 시 덮어쓰기 위험에서 벗어나 해결 가능

Git PRESENTATION

02 Repository

2.1 저장소란?



원격 저장소 (Remote Repository)?

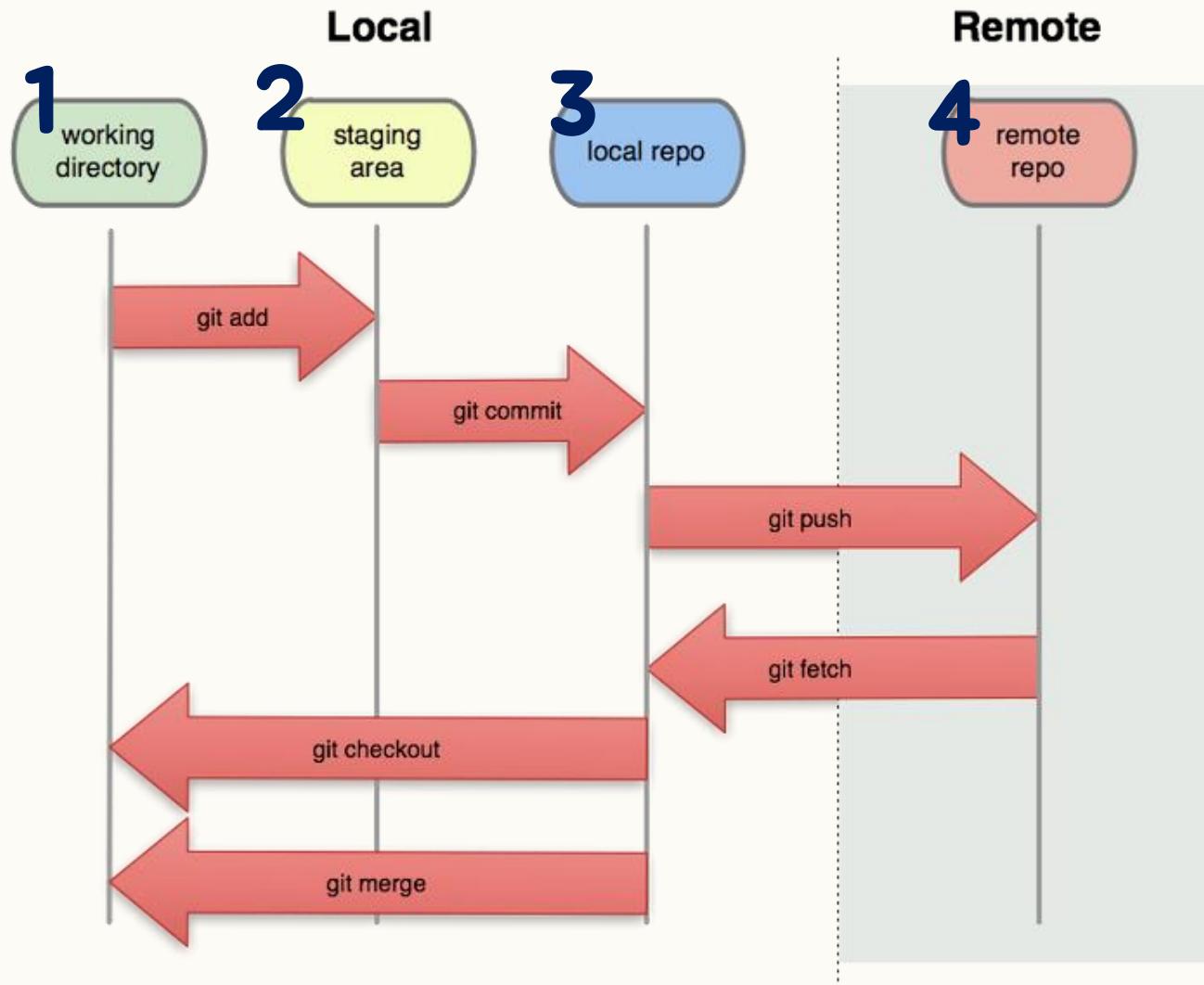
파일이 원격 저장소 전용 서버에서 관리 되며

여러 사람이 함께 공유하기 위한 저장소

로컬 저장소 (Local Repository)?

내 PC에 파일이 저장되는 개인 전용 저장소

2.1 저장소란?



1. Working directory?
로컬 작업 디렉토리

2. Staging area?
커밋 시 반영되는 파일 보관

3. Local repo?
로컬에 저장된 파일을
push할 경우 원격 저장소로 반영

4. Remote repo?
원격 저장소

2.2 저장소 생성

The screenshot shows a GitHub profile page for a user named Yewon Choi. On the left, there is a profile picture of a woman with long dark hair, a bio section with the name 'Yewon Choi' and handle 'devAon', and a 'Edit profile' button. Below that is an 'Organizations' section with four icons. At the top, there is a navigation bar with tabs: Overview, **Repositories 17** (highlighted with a red box), Projects 0, Packages 0, Stars 1, Followers 7, and Following 6. Below the navigation bar is a search bar with placeholder text 'Find a repository...', and dropdown menus for 'Type: All' and 'Language: All'. A green button labeled 'New' with a plus sign is also highlighted with a red box. The main content area features two cat images: a brown tabby cat on the left and an orange tabby cat on the right, both looking upwards.

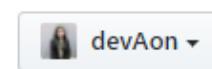
[Repositories]-[New]

2.2 저장소 생성

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner



Repository name *

Eclipse-GitHubTest

저장소(Repository) 이름

Great repository names are short and memorable. Need inspiration? How about `laughing-octo-lamp`?

Description (optional)

이클립스에서의 Git 사용법 - 테스트

저장소 설명(선택사항)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.



Initialize this repository with a README

This will let you immediately clone the repository to your computer.

Add .gitignore: None

Add a license: None

README 자동 생성 여부
.gitignore / license 자동 생성 여부

Create repository

2.2 저장소 생성

The screenshot shows a GitHub repository creation interface for a user named 'devAon' with the repository name 'Eclipse-GitHubTest'. The interface includes navigation tabs for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. A red box highlights the 'Code' tab.

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH <https://github.com/devAon/Eclipse-GitHubTest.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# Eclipse-GitHubTest" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/devAon/Eclipse-GitHubTest.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/devAon/Eclipse-GitHubTest.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

A red box highlights the '...or create a new repository on the command line' section. A red square highlights the copy icon in the top right corner of this section. To the right of the highlighted area, the text '생성된 저장소 주소 복사' (Copy repository address) is displayed.

저장소 사용 방법
원하는 것 선택하여 사용

2.2 저장소 생성 - config 초기 설정

```
MINGW64:/c/Users/choiyewon/Desktop/github_ppt/config_test  
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/config_test  
$ git config --global user.name "devAon" → '깃허브 이름'  
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/config_test  
$ git config --global user.email "yewon9742@gmail.com" → '깃허브 등록 이메일'  
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/config_test  
$ |
```

Git을 설치하고 나서 가장 먼저 해야 하는 것?

=> 사용자 이름과 이메일 주소를 설정

Git은 커밋할 때마다 이 정보를 사용한다. 한 번 커밋한 후에는 정보를 변경할 수 없다.

2.2 저장소 생성 - git 초기화

탐색 화면 > github_ppt > config_test

| 이름 | 수정한 날짜 | 유형 |
|------|--------------------|-------|
| .git | 2020-02-13 오전 1... | 파일 폴더 |

Git 설정 파일 (숨김파일)

저장소의 환경설정 정보 및 저장소 히스토리 정보 가짐

해당 파일 삭제 시 git 연결 설정 및 관련 정보가 사라짐. (원격x, 로컬에서만)

MINGW64:/c/Users/choiyewon/Desktop/github_ppt/config_test

1

```
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/config_test (master)
$ git init
Reinitialized existing Git repository in C:/Users/choiyewon/Desktop/github_ppt/config_test/.git/
```

2

```
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/config_test (master)
$ git remote add origin https://github.com/devAon/Eclipse-GitHubTest.git
```

3

```
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/config_test (master)
$ git remote -v
origin  https://github.com/devAon/Eclipse-GitHubTest.git (fetch)
origin  https://github.com/devAon/Eclipse-GitHubTest.git (push)
```

```
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/config_test (master)
$ |
```

깃 초기화

remote 연결

연결 확인

Git PRESENTATION

03 명령어

3.1 Add

Add 명령어?

저장소에 파일 추가 (Staged)

[Git bash0]용시]

경우1 - 디렉토리 안 모든 파일 추가)

: git add .

경우2 - 특정 파일만 추가)

: git add [경로/파일 이름]

3.2 Commit

Commit 명령어?

Staging한 파일 커밋.

커밋 메시지는 해당 작업에서 추가/변경된 사항 간략히 작성
타인이 알아볼 수 있도록 작성 및 영어로 작성하는 것이 좋다.

[Git bash0]용시]

`git commit -m '커밋 메시지'`

3.3 Push

Push 명령어?

원격 저장소로 Push

[Git bash 이용시]

`git push [원격 저장소 이름] [원격 저장소 브랜치 이름]`

ex) `git push origin master`

☞ 로컬 저장소의 변경사항을 원격 저장소(origin)의 브랜치(master)에 반영

3.4 실습

1

```
MINGW64:/c/Users/choiyewon/Desktop/github_ppt/github_eclipse_test/GitTest
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/github_eclipse_test/GitTest (master)          'Eclipse-GitHubTest' 내용의 README.md 파일 작성
$ echo "# Eclipse-GitHubTest" >> README.md

choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/github_eclipse_test/GitTest (master)
$ git status          깃 상태 조회
On branch master
Your branch is up to date with 'origin/master'.

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

    README.md
```

3.4 실습

```
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/github_eclipse_test/GitTest (master)
2 $ git add README.md      add
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory

choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/github_eclipse_test/GitTest (master)
$ git status               깃 상태조회(Staged)
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   README.md
```

3.4 실습

```
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/github_eclipse_test/GitTe  
st (master)  
3 $ git commit -m "first commit"      commit  
[master fd4e09c] first commit  
 1 file changed, 1 insertion(+)  
 create mode 100644 README.md  
  
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/github_eclipse_test/GitTe  
st (master)  
4 $ git push -u origin master      push  
Enumerating objects: 4, done.  
Counting objects: 100% (4/4), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 317 bytes | 158.00 KiB/s, done.  
Total 3 (delta 1), reused 0 (delta 0)  
remote: Resolving deltas: 100% (1/1), completed with 1 local object.  
To https://github.com/devAon/Eclipse-GitHubTest.git  
  1a1e79e..fd4e09c  master -> master  
Branch 'master' set up to track remote branch 'master' from 'origin'.  
  
choiyewon@DESKTOP-H3KQBP3 MINGW64 ~/Desktop/github_ppt/github_eclipse_test/GitTe  
st (master)  
$ |
```

3.4 실습

devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 1 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

이클립스에서의 Git 사용법 - 테스트 Edit

Manage topics

3 commits 3 branches 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

devAon first commit Latest commit fd4e09c 2 minutes ago

| | | |
|-------------|---|---------------|
| .settings | [예시]create project. create hello world. | 13 hours ago |
| src/gittest | git test2 | 13 hours ago |
| .classpath | [예시]create project. create hello world. | 13 hours ago |
| .gitignore | [예시]create project. create hello world. | 13 hours ago |
| .project | [예시]create project. create hello world. | 13 hours ago |
| README.md | first commit | 2 minutes ago |

README.md

Eclipse-GitHubTest

'Eclipse-GitHubTest' 내용으로 작성한 README.md 파일이 github에 반영된 것 확인

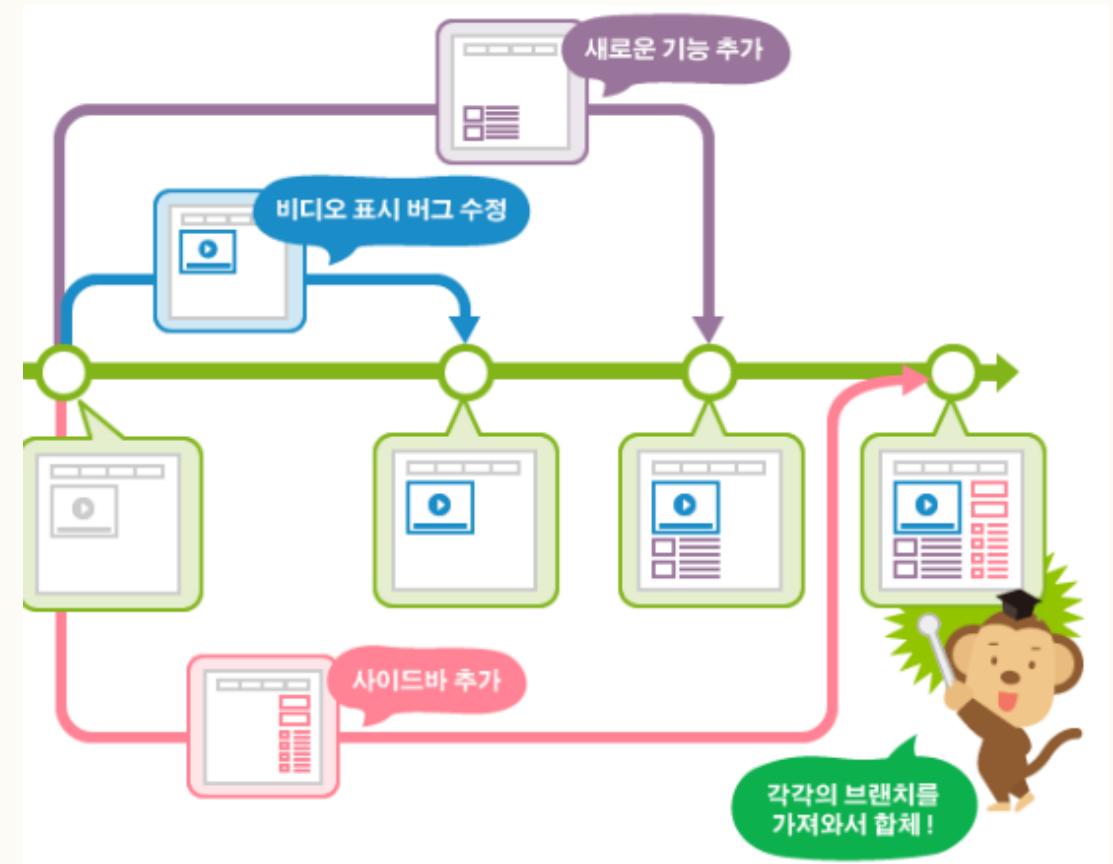
Git PRESENTATION

04 Git Flow

4.1 Branch

branch란?

- 각자 독립적인 작업 영역(저장소)
즉, 여러 사람이 동일한 소스코드를 기반으로
서로 다른 작업
- 분리된 작업 영역에서 마음대로 소스코드 변경
- 작업이 끝난 사람은 메인 브랜치에
자신의 브랜치의 변경 사항을 적용



4.2 GitFlow

GitFlow란?

Vincent Driessen의 브랜칭 모델을 적용하여 고수준으로 저장소를 관리하기 위한 브랜칭 기법

프로젝트를 진행하며 발생하는 많은 branch를 관리할 수 있도록 해주는 규칙 및 전략이다.
기본 전략이기 때문에 프로젝트 상황에 맞게 커스텀해서 사용하면 된다.

사용 이유?

1. feature branch를 이용하기 때문에 기능 개발의 책임 소재를 명확히 할 수 있다.
2. 개발 버전과 제품 버전을 개별 관리할 수 있다.
3. Pull Request를 이용하기 때문에 쉽게 코드 리뷰를 할 수 있다.
4. feature branch와 hotfix branch의 commit message를 취합하게 되면,
이전 버전과의 변경점 쉽게 파악 가능하다

4.2 GitFlow

Git-flow에는 5가지 종류의 브랜치가 존재한다.

master : 제품으로 출시될 수 있는 브랜치

develop : 다음 출시 버전을 개발하는 브랜치

feature : 기능을 개발하는 브랜치

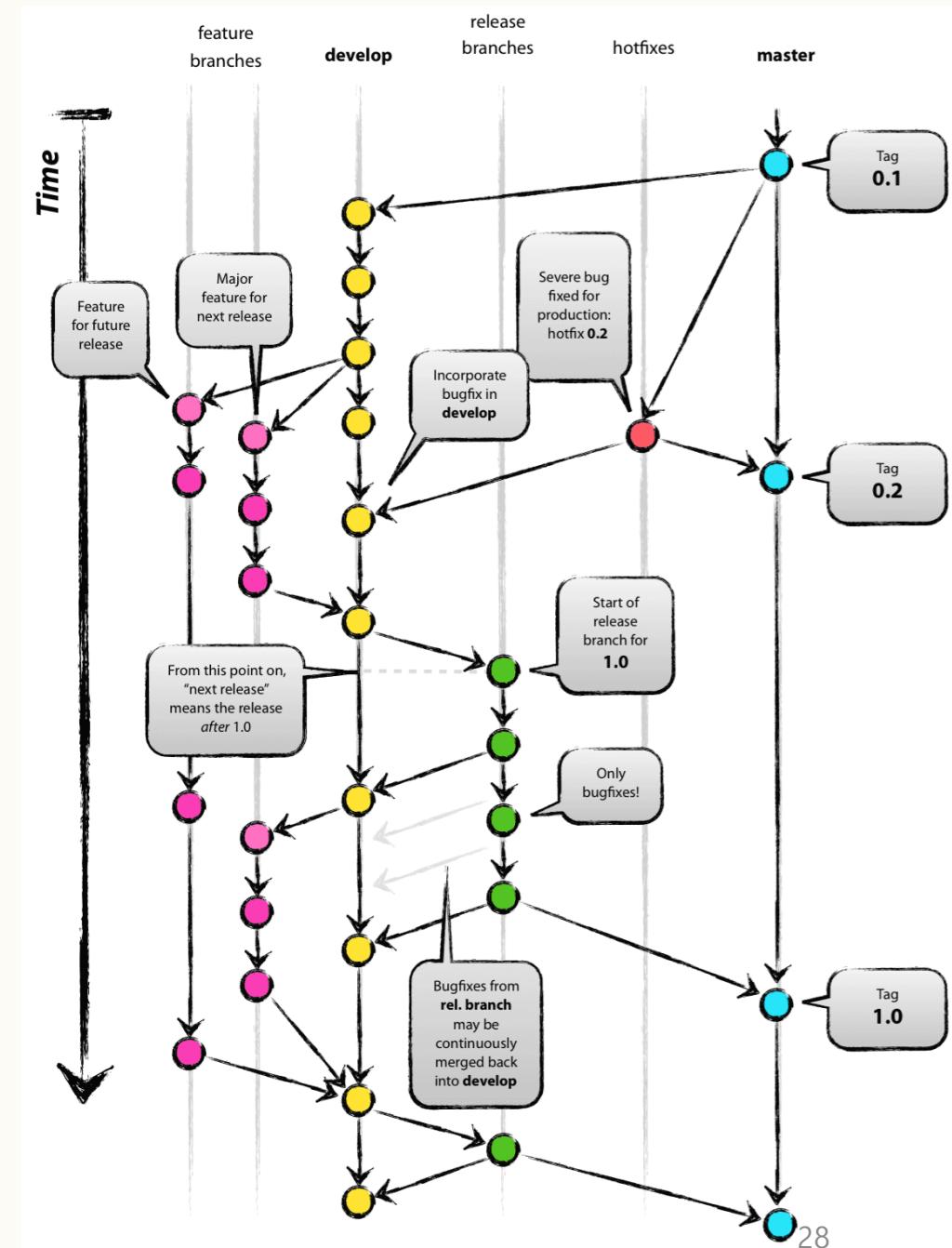
release : 이번 출시 버전을 준비하는 브랜치

hotfix : 출시 버전에서 발생한 버그를 수정 하는 브랜치

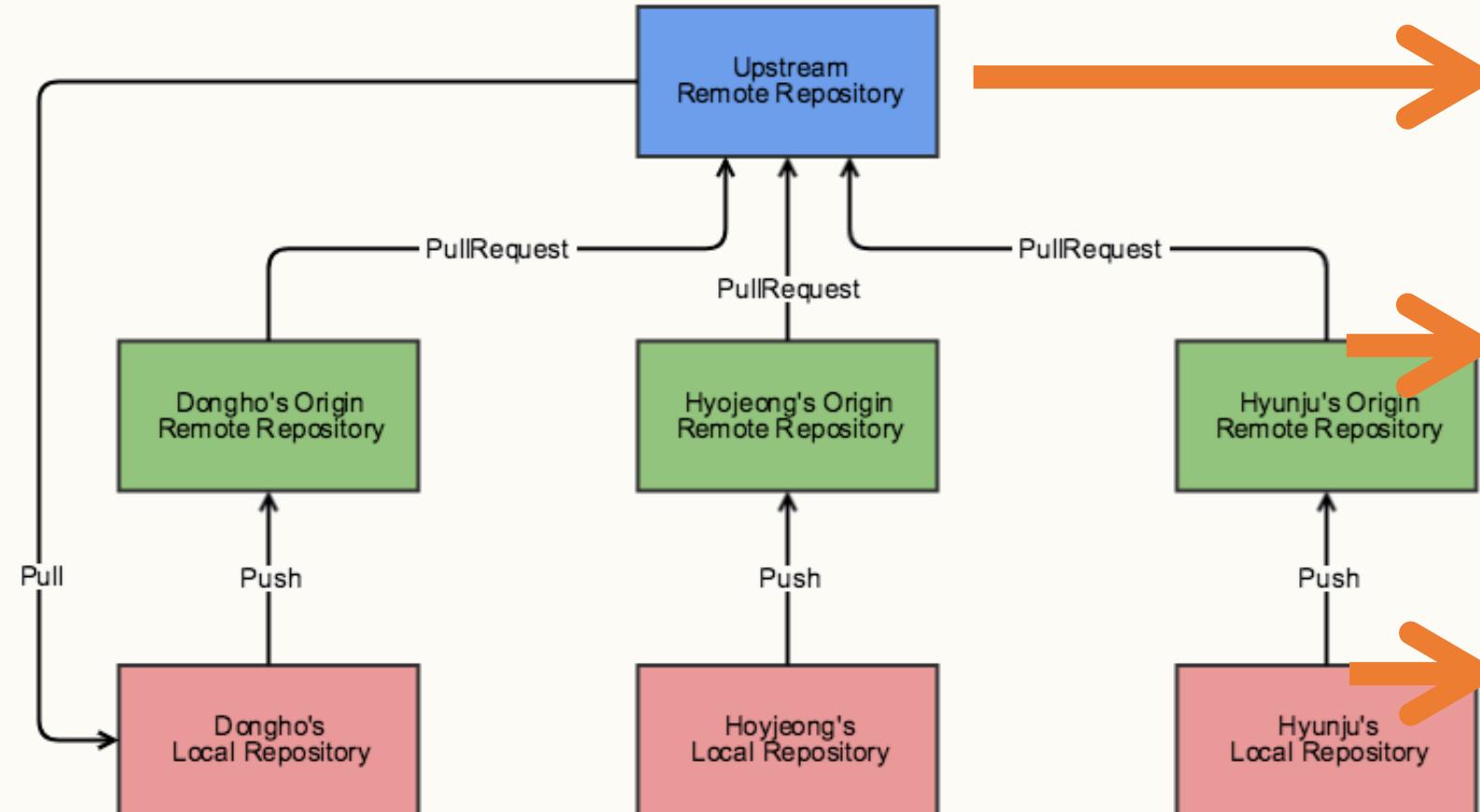
develop과 **master**은 중심이 되는 브랜치이다.

따라서, git flow에서 두 브랜치는 반드시 존재해야 한다.

(**develop** 브랜치는 **master**에서부터 시작된 브랜치)



4.2 GitFlow



Upstream Repository?

개발자들이 공유하는 저장소로
최신 소스코드가 저장되어 있는 원격 저장소

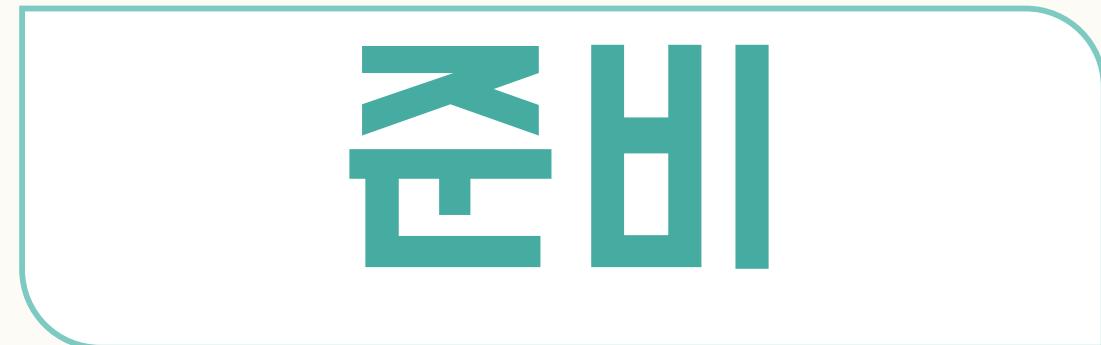
Origin Repository?
(fork하면 존재하는 저장소)

Upstream Repository를 Fork한
원격 개인 저장소

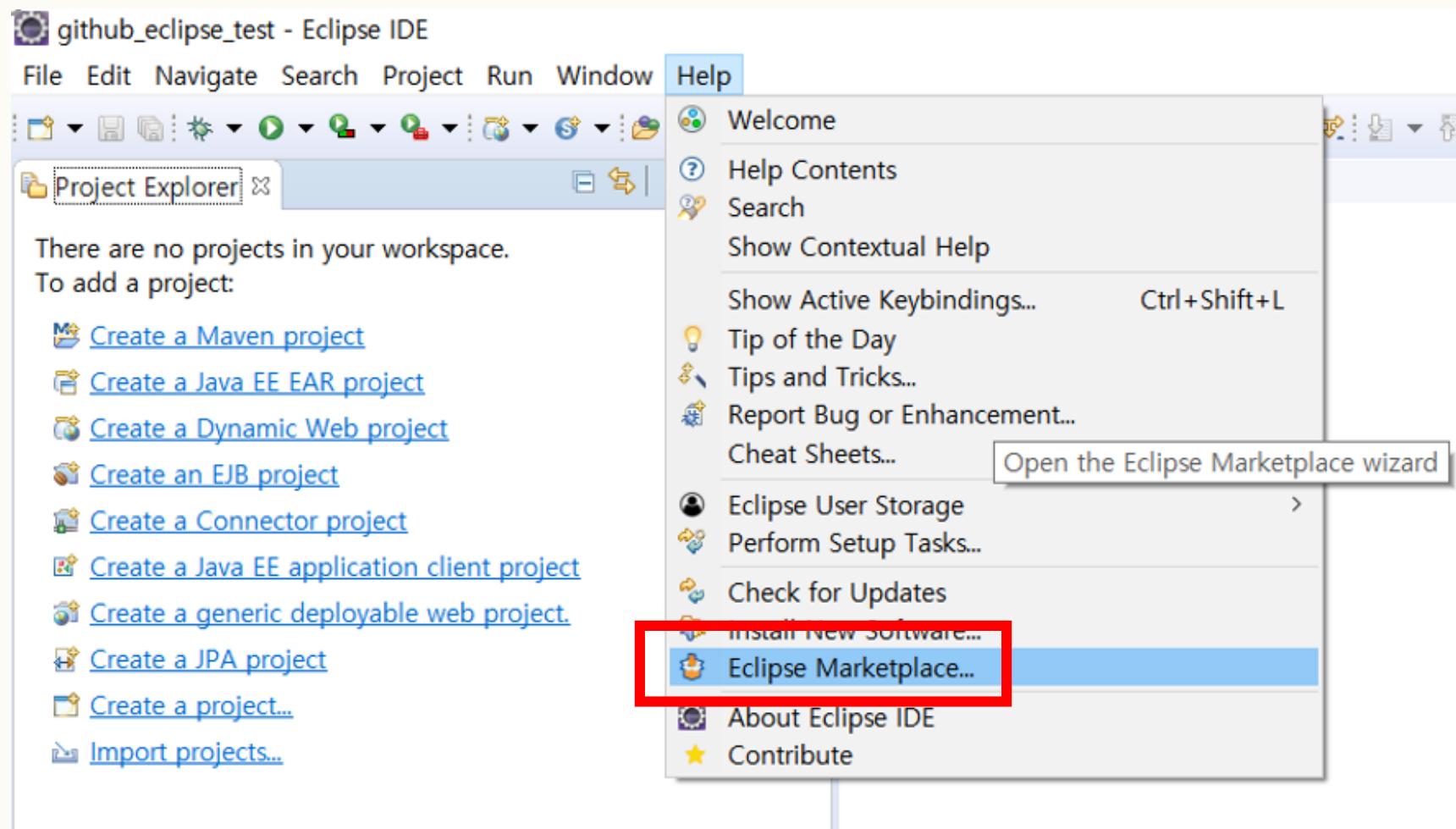
Local Repository?
내 컴퓨터에 저장되어 있는 개인 저장소

Git PRESENTATION

05 Eclipse-Git

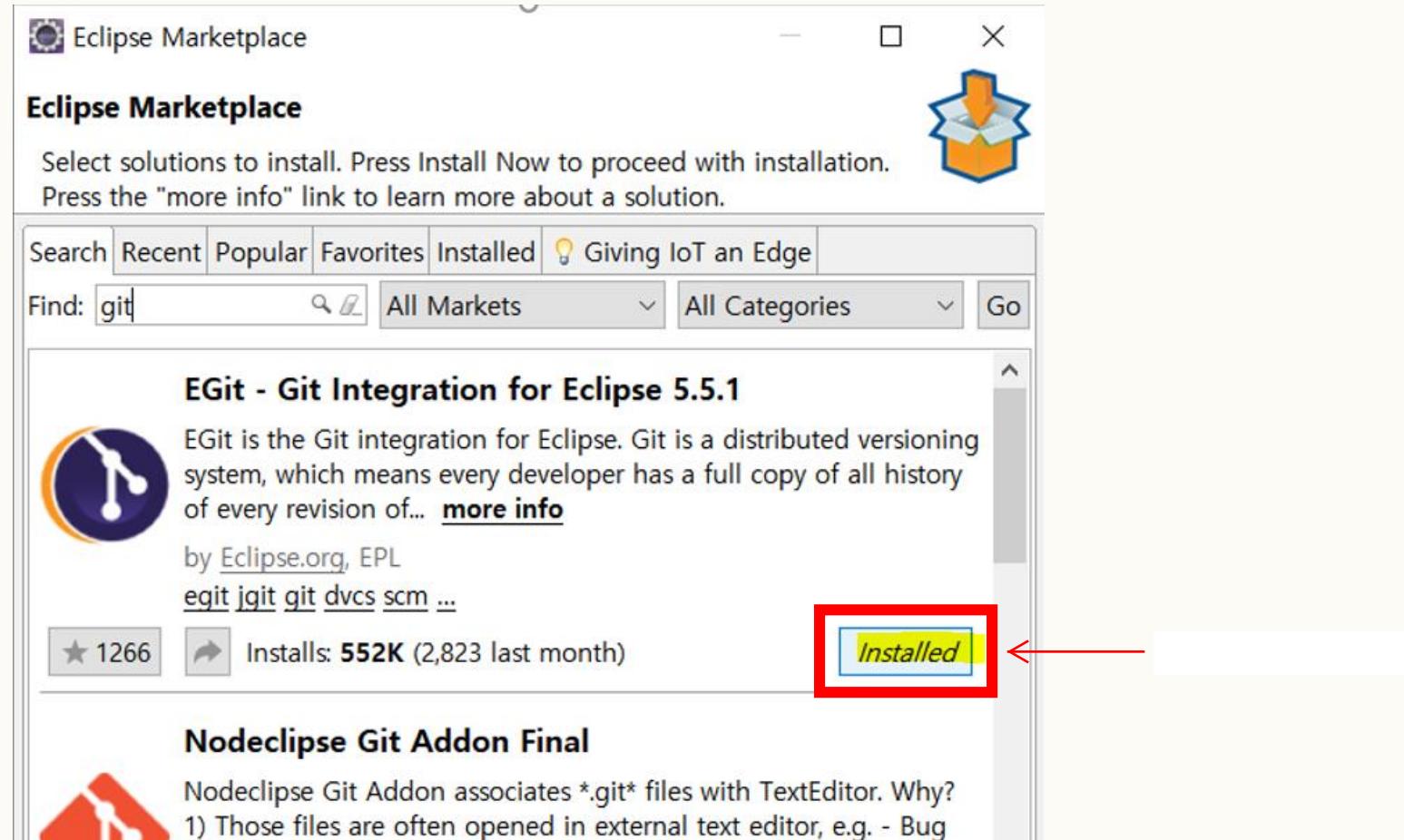


5.준비 - 1 Egit 설치



[Help]-[Eclipse Marketplace]

5.준비 - 1 Egit 설치

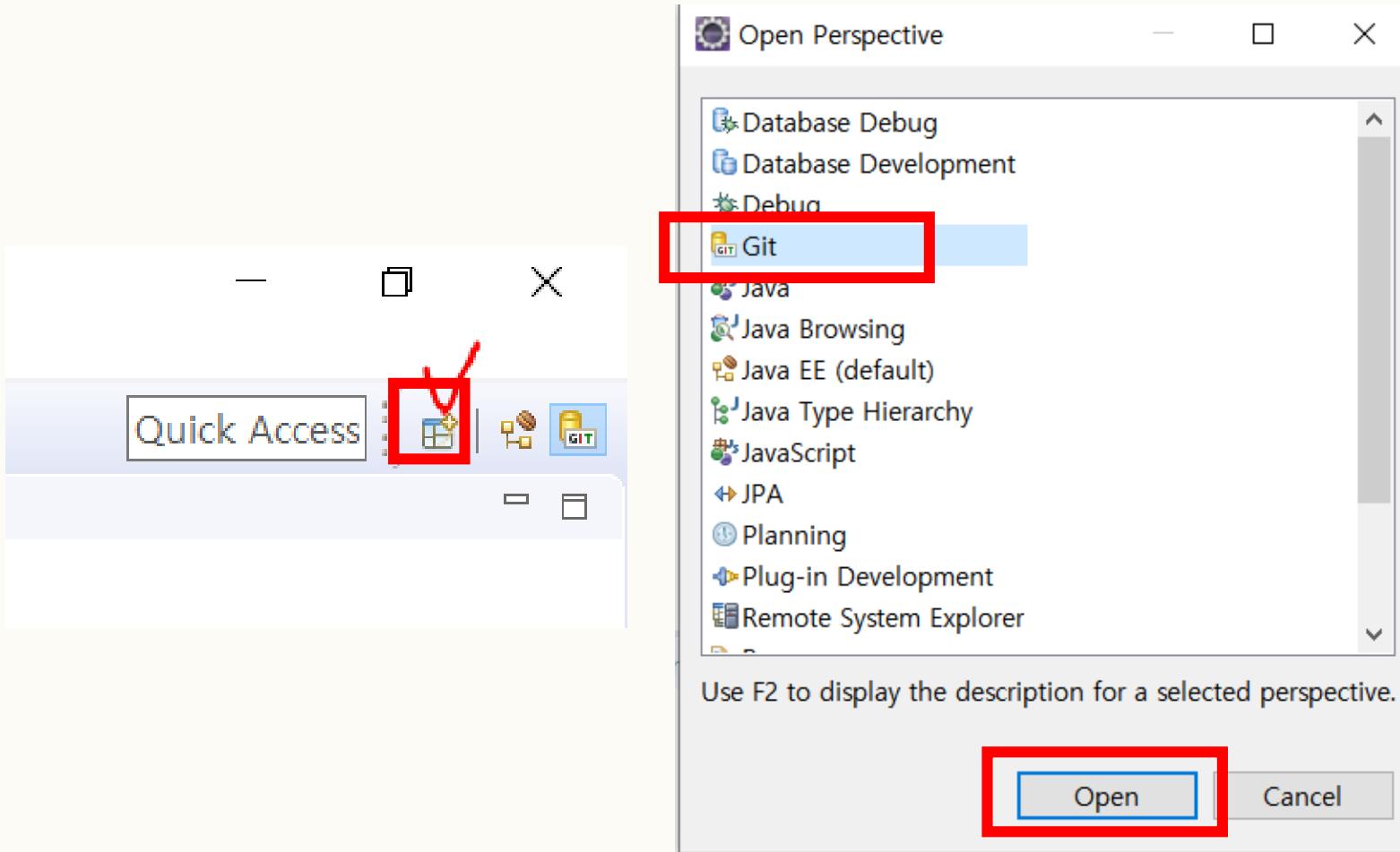


1) Find: git 검색 - Egit install or update

2) 모든 항목 선택 후 [confirm] 클릭

3) Review Licenses에서 'I accept the terms of the license agreement' 선택 후 [Finish] 클릭. - [Yes] 클릭

5.준비 - 1 Egit 설치



[Open Perspective]-[Git]-[Open]



팀장

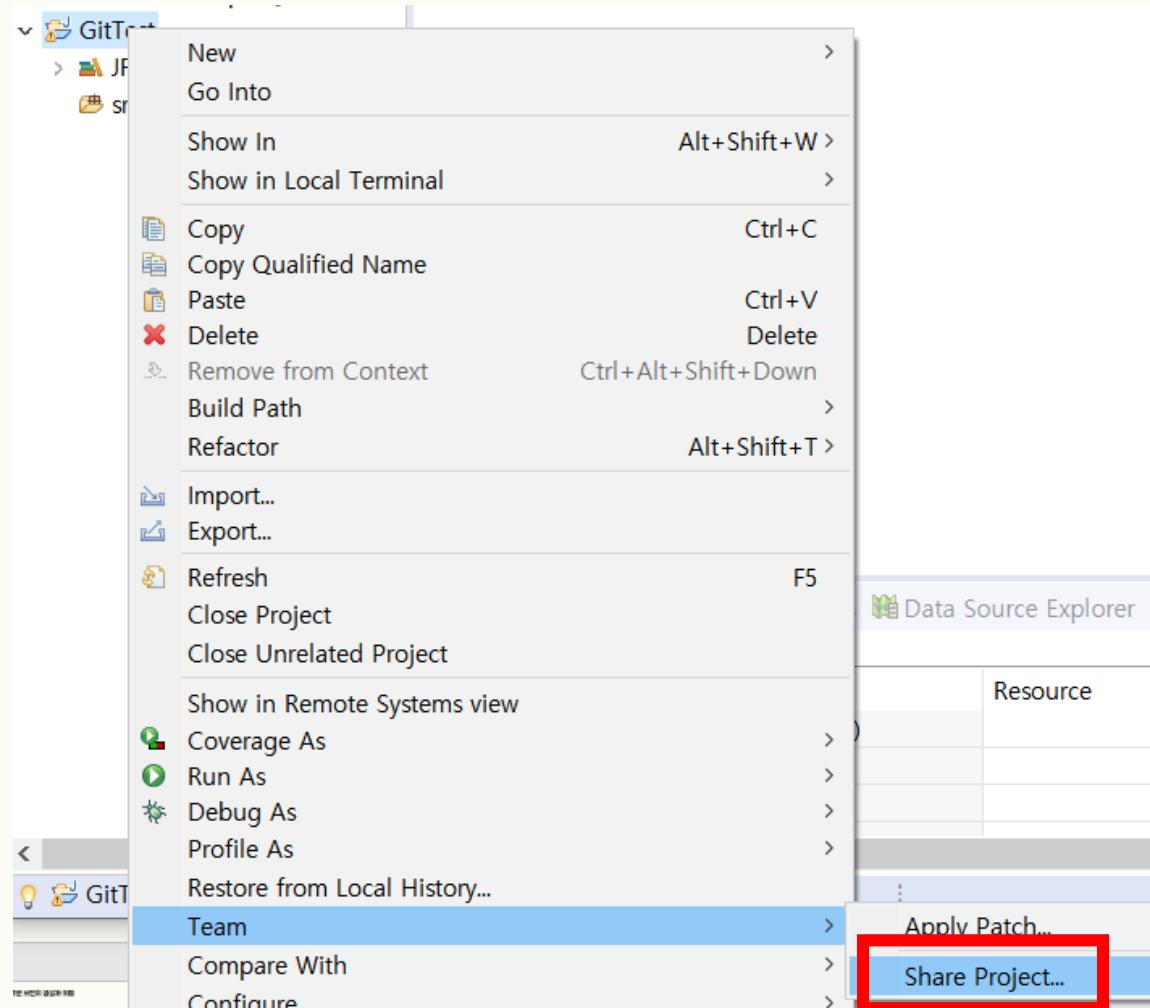
5. 팀장 - 1 의논

GitHub와 연동에 앞서 우선

1. 어떤 IDE, DB, 언어 사용할지,
프로젝트 구조 등 팀원들과 의논
2. 버전 및 환경설정을 맞춰 base 프로젝트를 준비

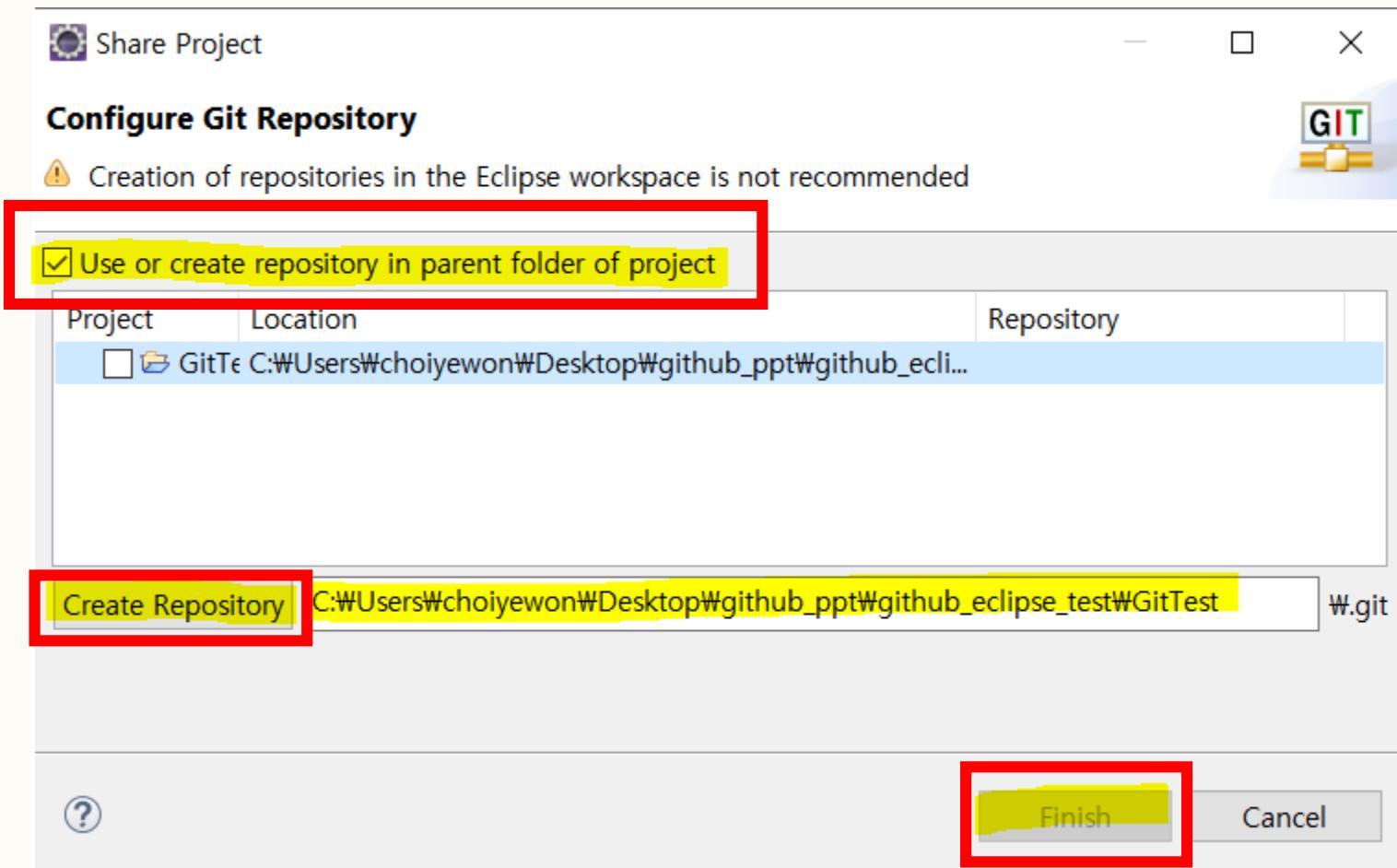


5. 팀장 - 1 저장소 생성



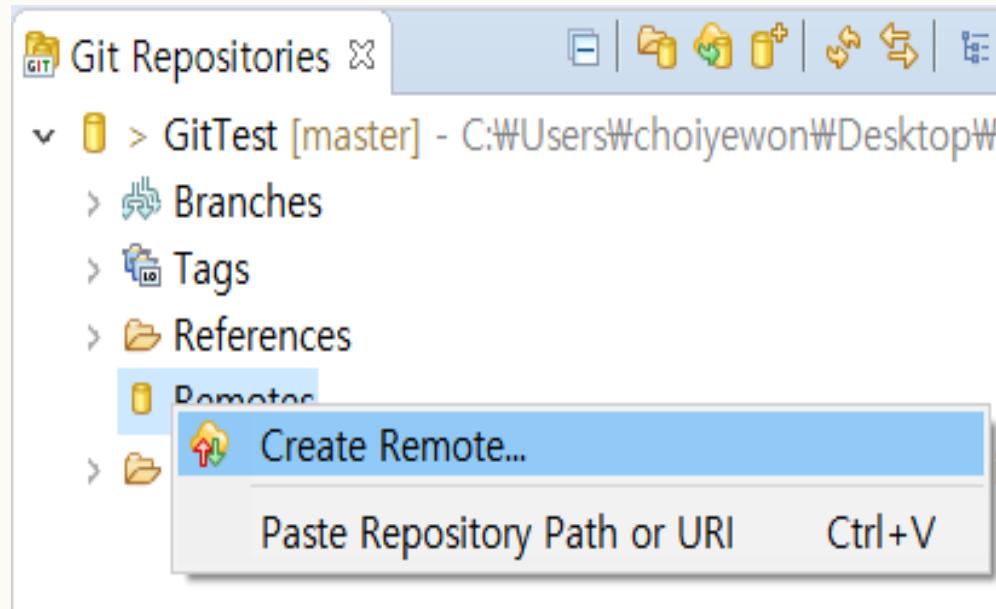
생성한 프로젝트에서 마우스 우클릭 - [Team]-[Share Project]

5. 팀장 - 1 저장소 생성

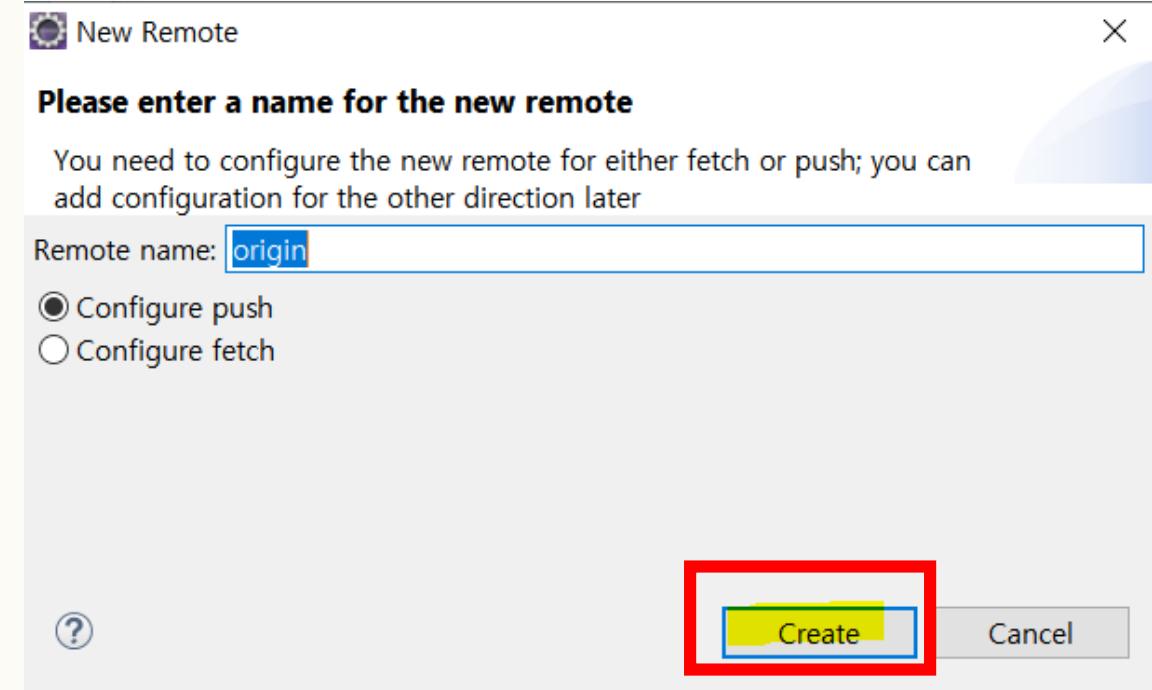


1. [User or create repository in parent folder of project] 체크
2. [Create Repository] 클릭 - .git 파일이 폴더에 생성됨
3. [Finish] 클릭

5. 팀장 - 1 저장소 생성

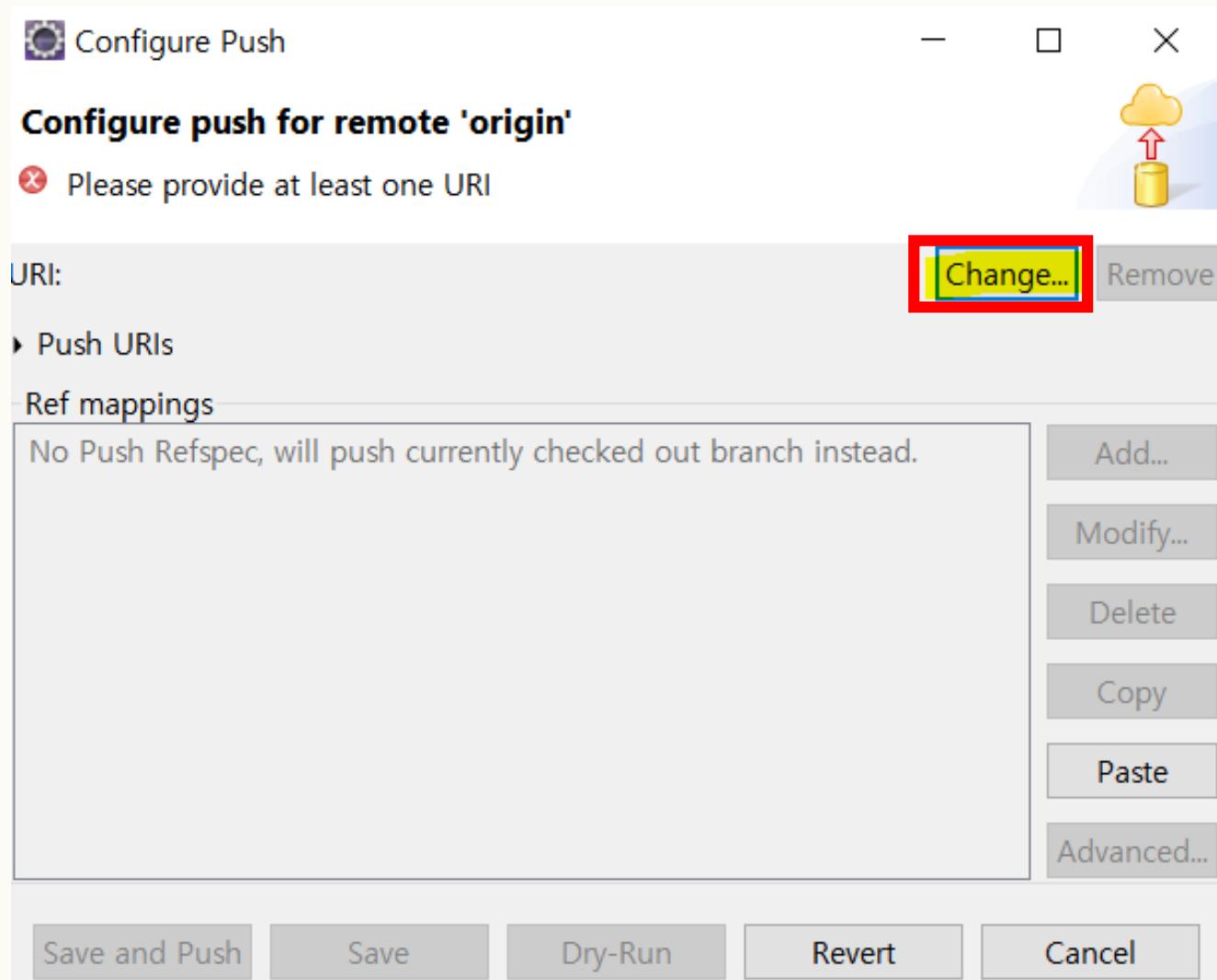


[Git Repositories]-[Remotes]-[Create Remote..]



[Create]

5. 팀장 - 1 저장소 생성



5. 팀장 - 1 저장소 생성

The screenshot shows the Eclipse GitHub Test interface on the left and a 'Select a URI' dialog box on the right.

Eclipse GitHub Test Interface:

- Repository name: devAon / Eclipse-GitHubTest
- Statistics: Unwatched (1), Starred (0), Forked (0)
- Navigation: Code, Issues (0), Pull requests (0), Actions, Projects (0), Wiki, Security, Insights, Settings
- Quick setup message: "Quick setup — if you've done this kind of thing before".
- Copy URL button: A yellow square highlights the copy icon next to the URL "https://github.com/devAon/Eclipse-GitHubTest.git".
- Text area: "Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore."

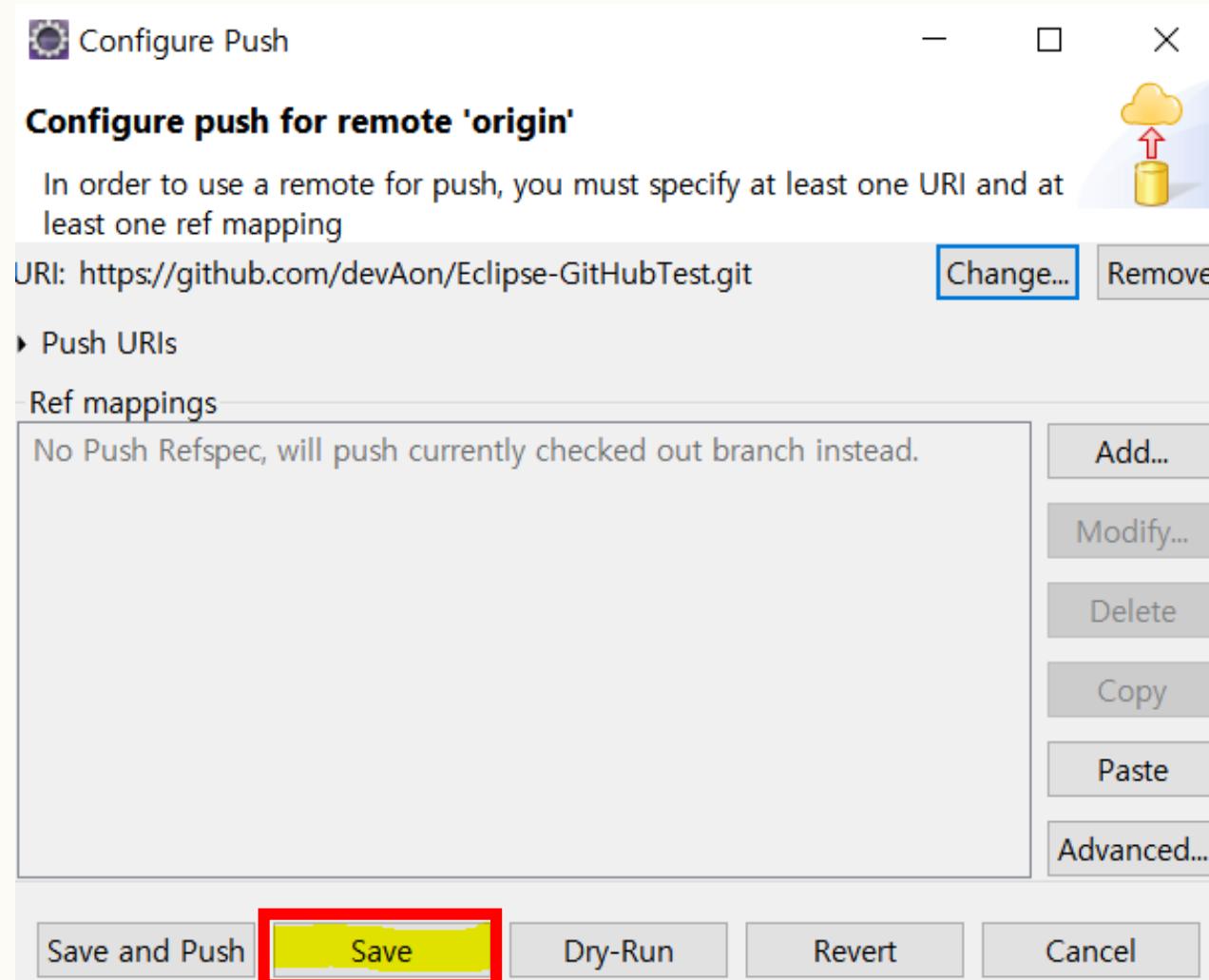
Select a URI Dialog Box:

- Source Git Repository:** Enter the location of the source repository.
- Location:**
 - URI: `https://github.com/devAon/Eclipse-GitHubTest.git` (highlighted with a yellow box and labeled **[Ctrl+V]**)
 - Host: `github.com`
 - Repository path: `/devAon/Eclipse-GitHubTest.git`
- Connection:**
 - Protocol: `https`
 - Port:
- Authentication:**
 - User: `devAon` (highlighted with a red box and labeled **내 깃허브 아이디**)
 - Password: (highlighted with a red box and labeled **내 깃허브 비밀번호**)
 - Store in Secure Store (highlighted with a blue arrow and labeled **체크시, 자동으로 아이디 비밀번호 입력됨**)
- Buttons:** Finish (highlighted with a red box), Cancel

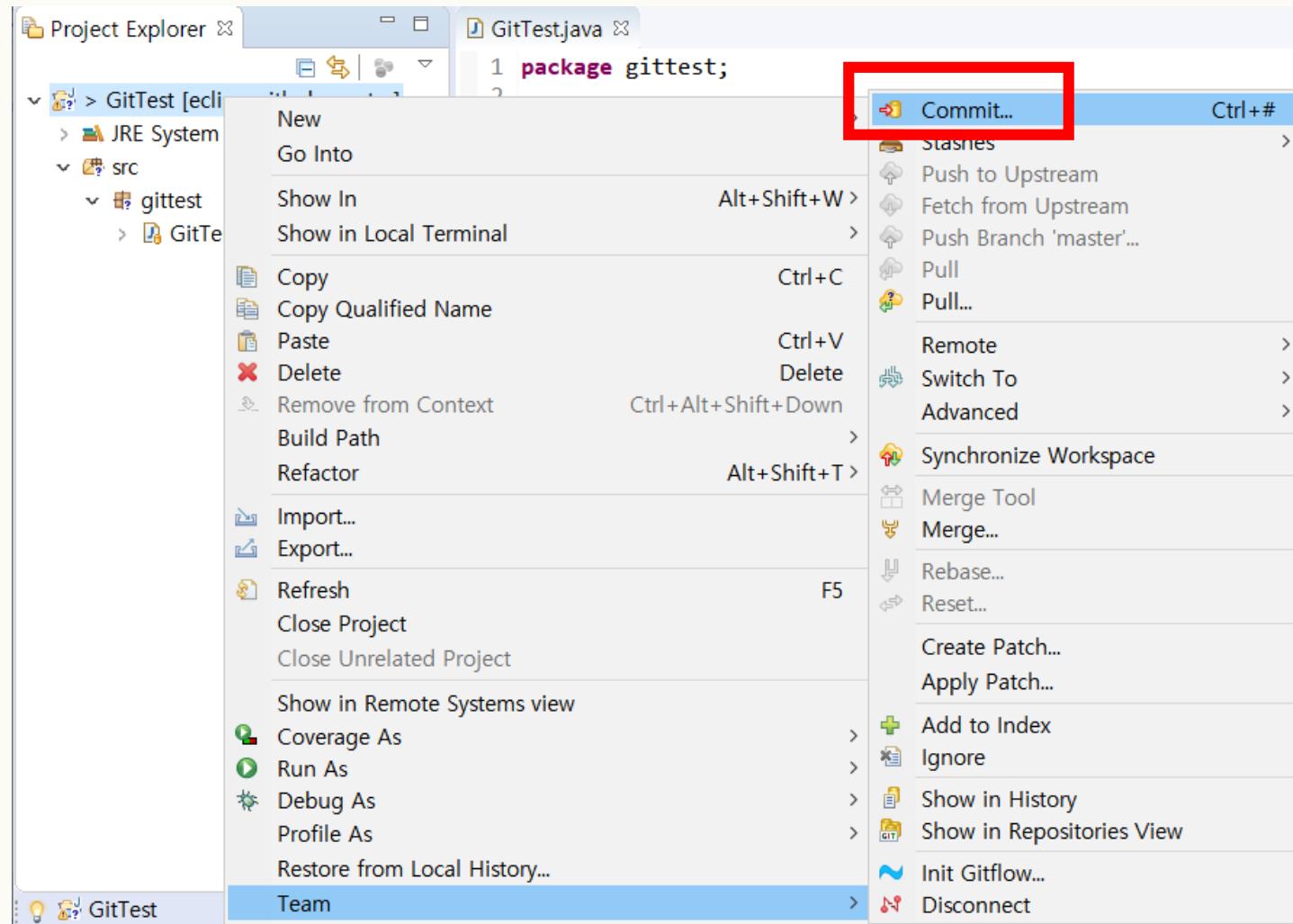
Bottom Left Note:

* : 2021 8 Github
Personal Access Token
Access Token
geno.tistory.com/89 (Personal
https://kitty -)

5. 팀장 - 1 저장소 생성

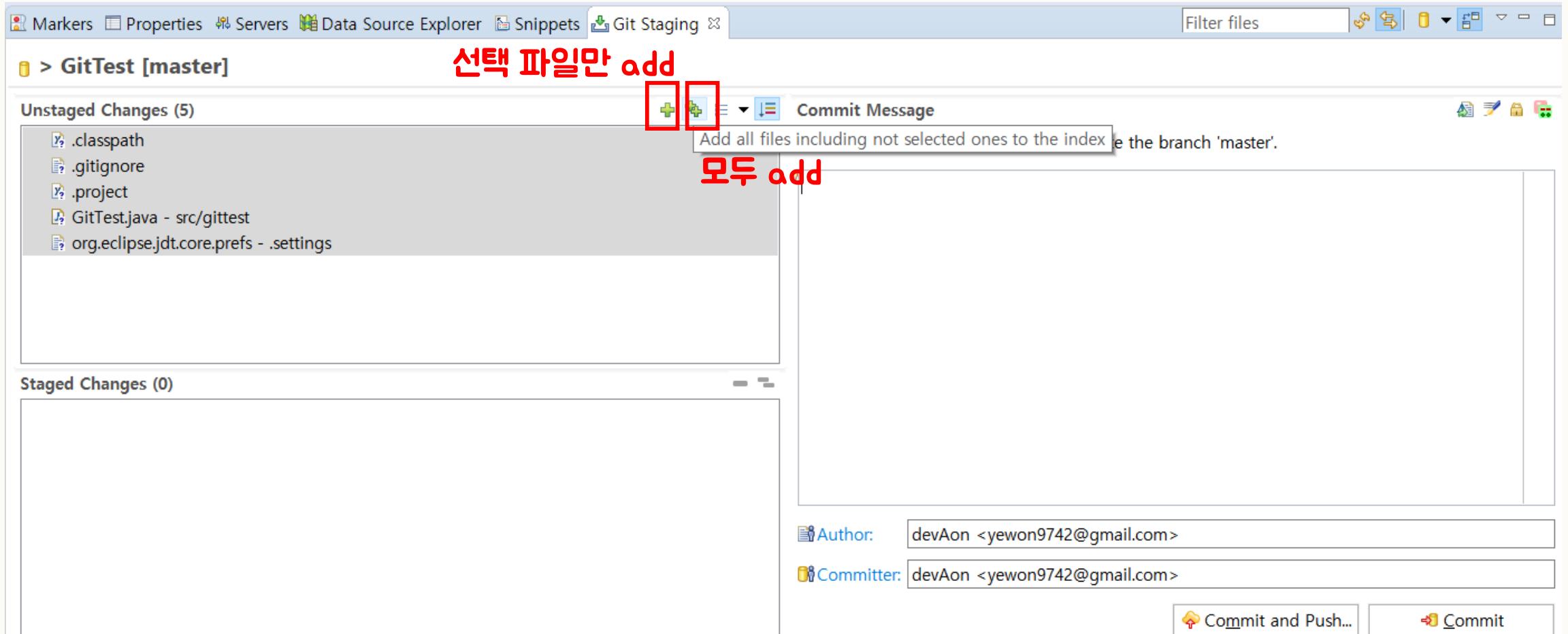


5. 팀장 - 1 저장소 생성



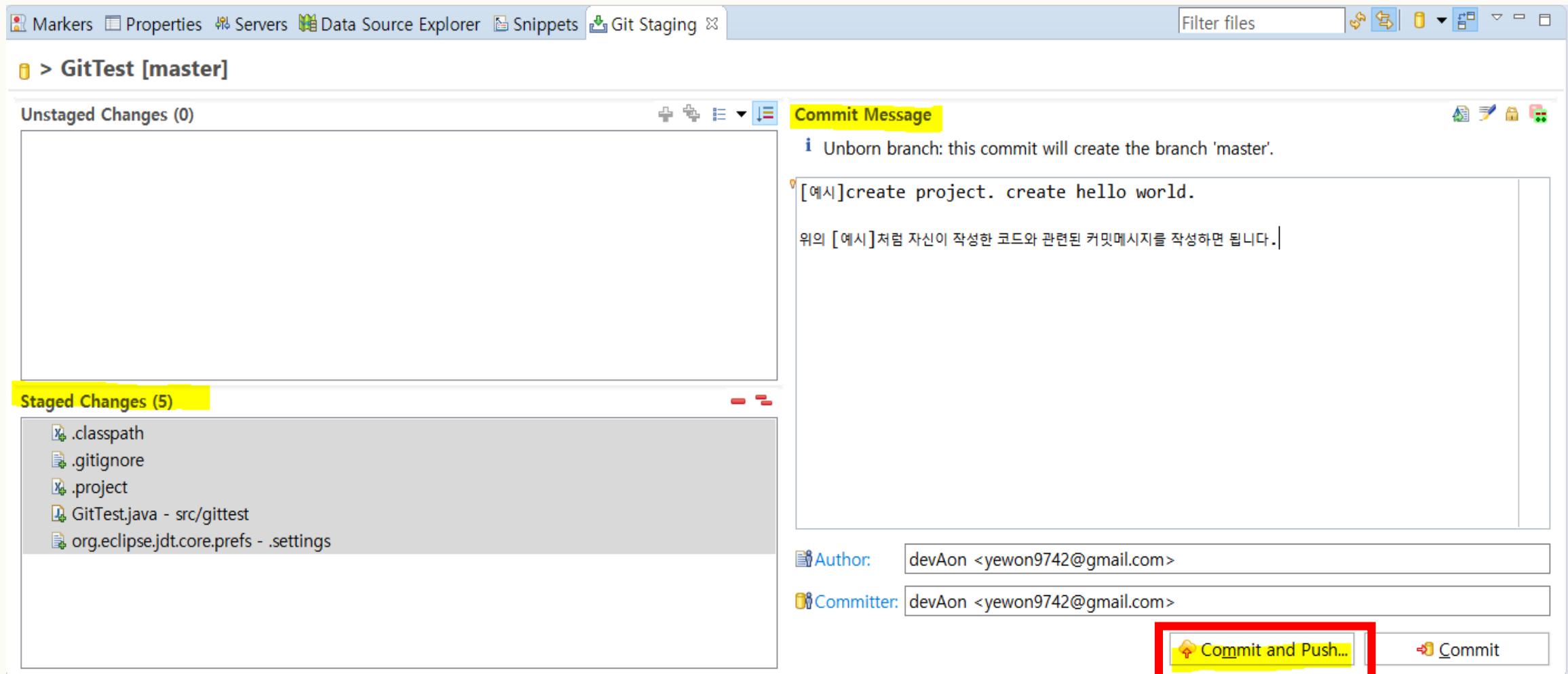
생성한 프로젝트에서 마우스 우클릭 - [Team]-[Commit]

5. 팀장 - 1 저장소 생성



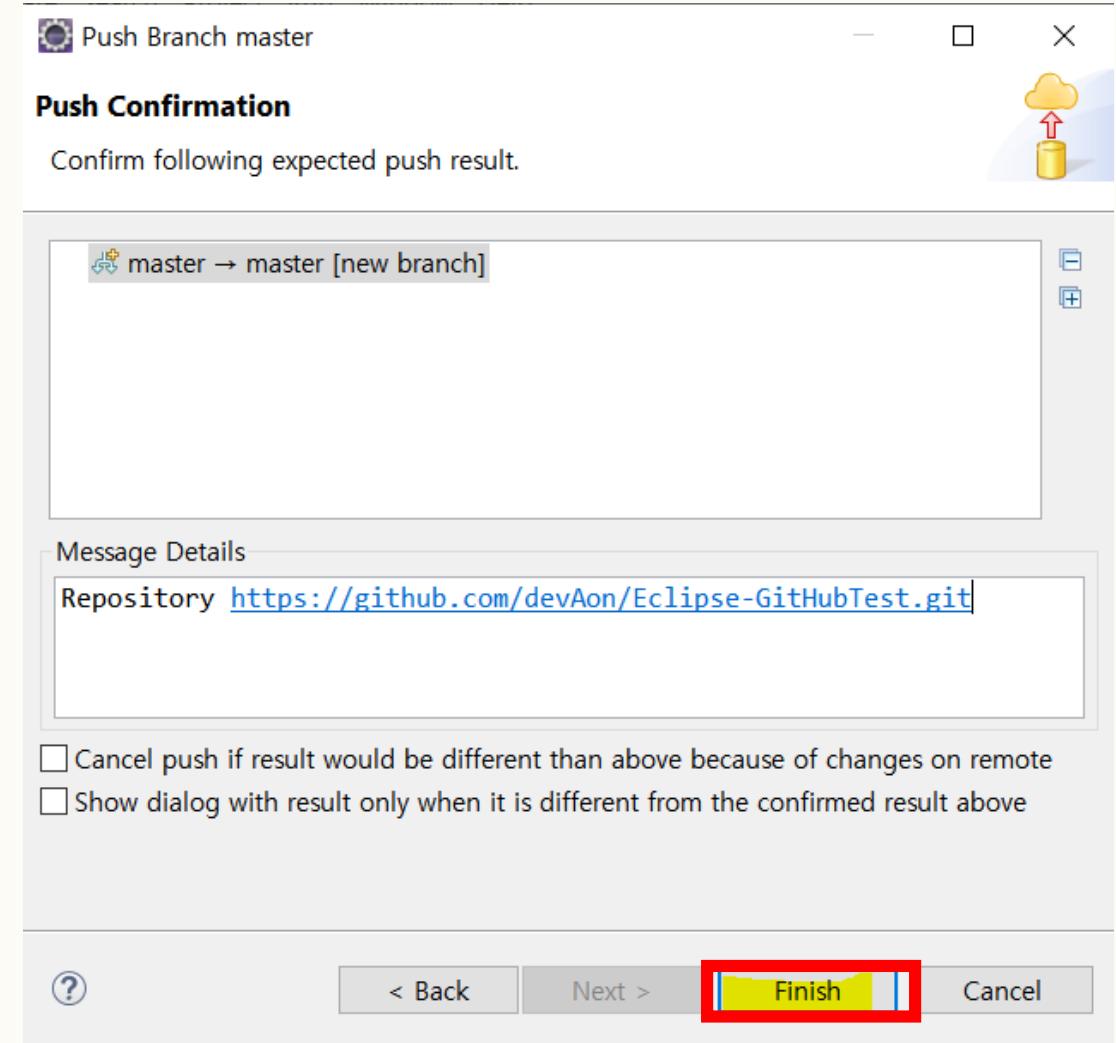
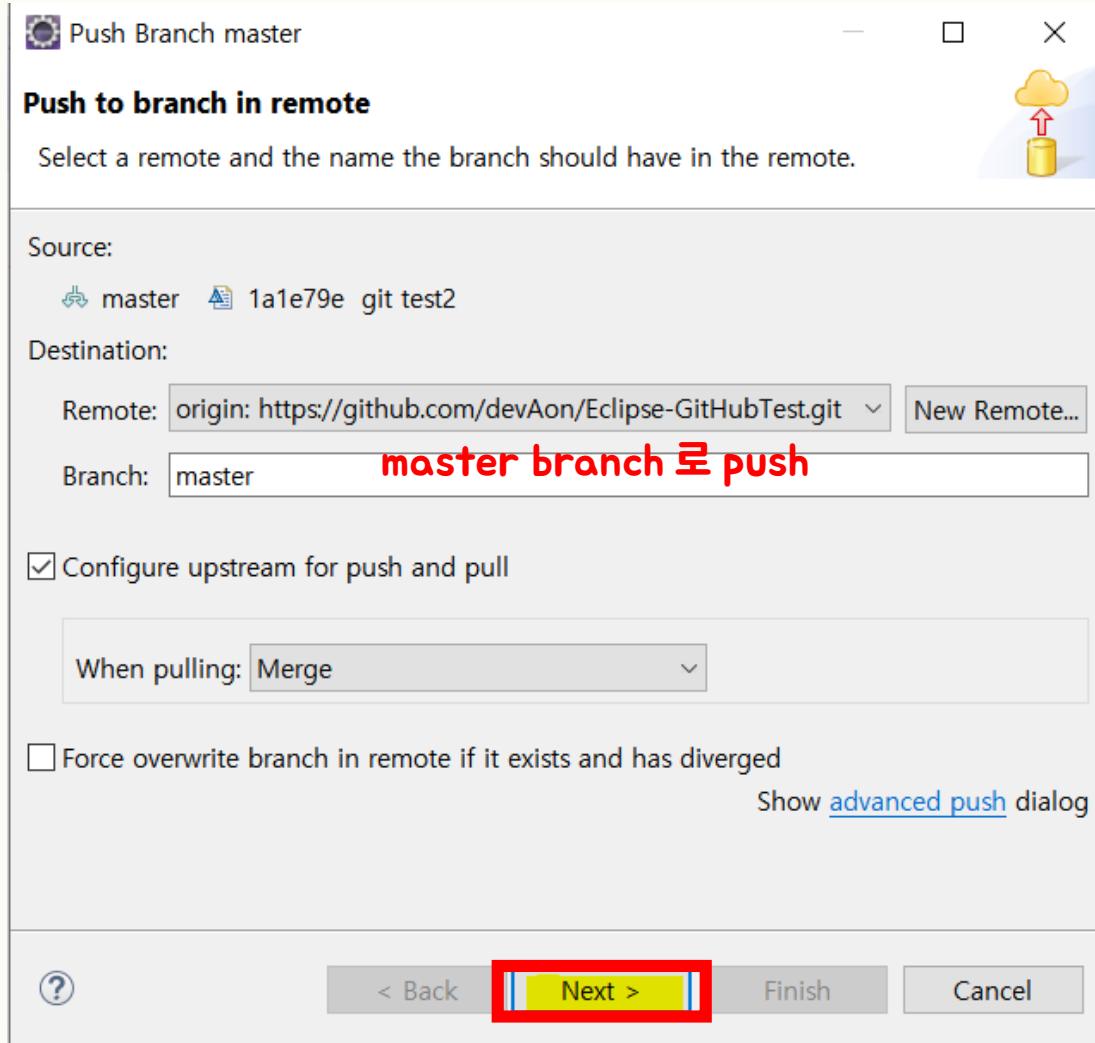
Unstaged Changes에 있는 파일 중 원격 저장소에 반영하고 싶은 파일을 선택해
Staged Changes로 add 해주면 된다.

5. 팀장 - 1 저장소 생성

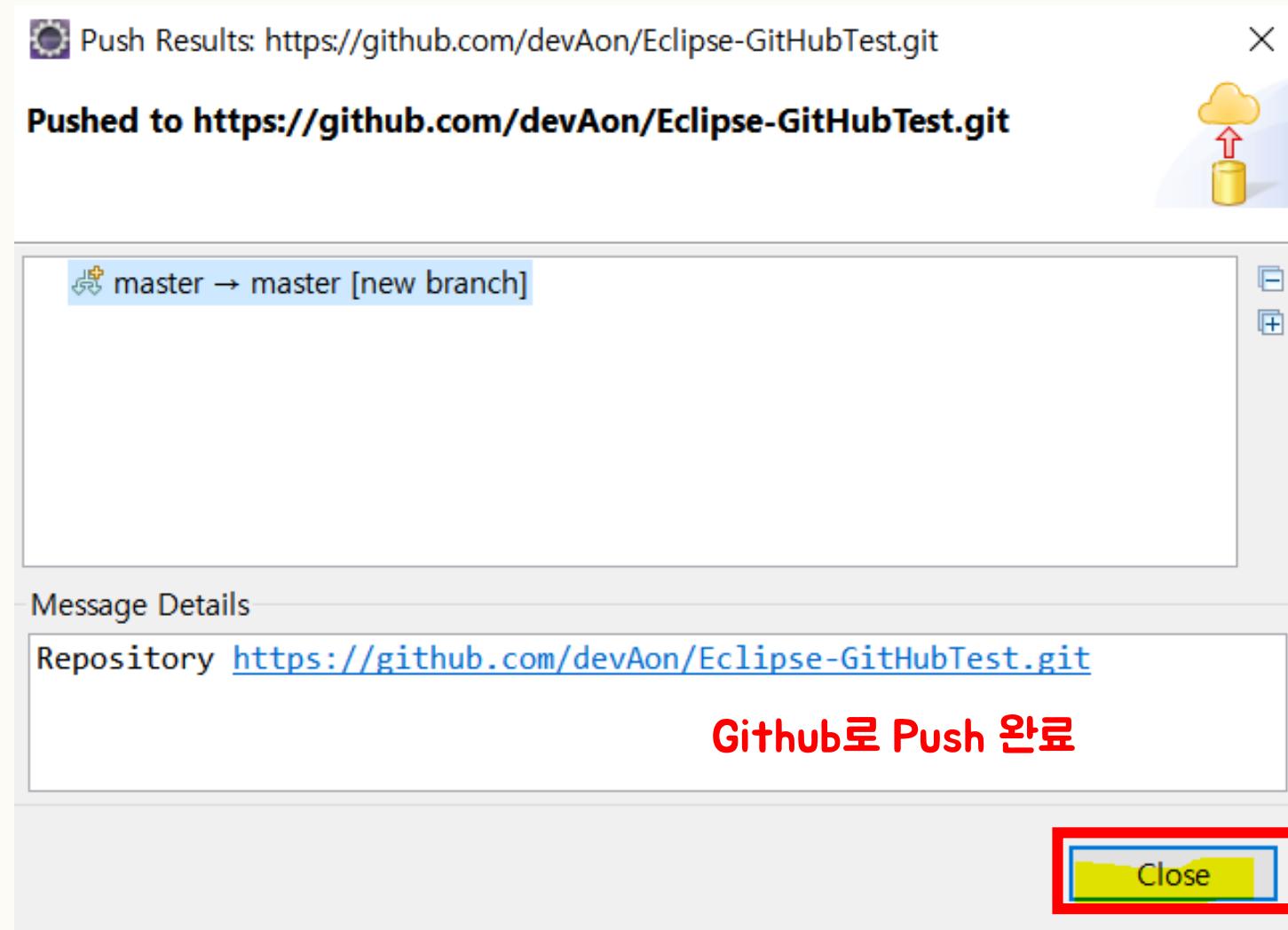


1. Staged Changes된 파일들에 관련된 커밋메시지 작성
2. [Commit and Push]클릭

5. 팀장 - 1 저장소 생성



5. 팀장 - 1 저장소 생성



5. 팀장 - 1 저장소 생성

팀장이 만든 저장소

devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

이클립스에서의 Git 사용법 - 테스트 Edit

Manage topics

2 commits 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

| devAon git test2 | Latest commit 1a1e79e 2 minutes ago |
|------------------|---|
| .settings | [예시]create project. create hello world. |
| src/gittest | git test2 |
| .classpath | [예시]create project. create hello world. |
| .gitignore | [예시]create project. create hello world. |
| .project | [예시]create project. create hello world. |

Help people interested in this repository understand your project by adding a README. Add a README

The screenshot shows a GitHub repository page for 'devAon / Eclipse-GitHubTest'. The repository has 2 commits, 1 branch, 0 packages, 0 releases, and 1 contributor. The latest commit was made by 'devAon' 2 minutes ago. The commit message is 'git test2'. Below the commit list, there is a note encouraging the user to add a README file to help others understand the project. At the bottom, there is a button to 'Add a README'.

Github에 들어가 정상적으로 Push된 것 확인

5. 팀장 - 2 Gitignore 꼭 추가

Gitignore란?

깃에서 특정 파일 혹은 디렉토리를 관리 대상에서 제외할 때 사용하는 파일.

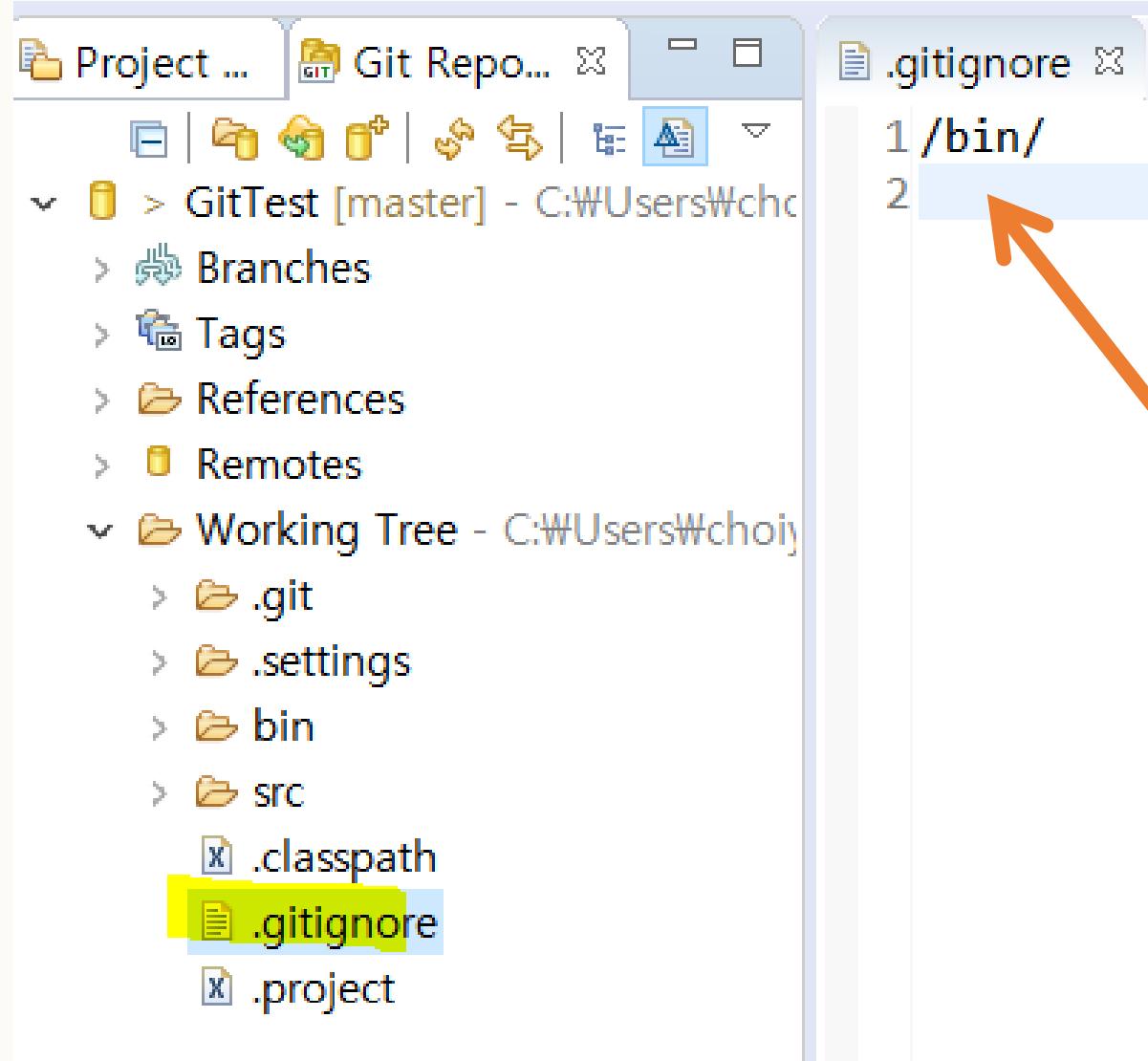
자동으로 생성되는 로그파일, 프로젝트 설정 파일, 중요 정보가 담긴 파일 등을
관리 대상에서 제외.

추가하지 않을 시?

자동으로 로그파일이 많이 생성된다. 추가하지 않을 경우 수정한 내용이 없어도 수정내용을 반영
하라는 등 원활하게 Git 협업하는데 문제점을 야기할 수 있다.

그러니 꼭 .gitignore를 추가하자.

5. 팀장 - 2 Gitignore 꼭 추가



[Git Repository]-[Working Tree]

-[src]-[.gitignore]가 있다.

Egit 플러그인은 기본적으로 저장소를 생성할 때
.gitignore 파일을 함께 생성해준다.

그렇지만, 기본으로 생성해준 파일은 비어있다.

그래서

반드시 ! 우리가 무시하고 싶은 불필요한 파일 및 폴더를
지정해줘야 한다.

5. 팀장 - 2 Gitignore 꼭 추가



.gitignore 파일을 추가하기 위해 <https://www.gitignore.io/> 에
접속해 사용중인 운영체제, 개발환경, 프로그래밍 언어를 선택하고 생성

5. 팀장 - 2 Gitignore 꼭 추가

```
# Created by https://www.gitignore.io/api/java,windows,eclipse
# Edit at https://www.gitignore.io/?templates=java,windows,eclipse

### Eclipse ####
.metadata
bin/
tmp/
*.tmp
*.bak
*.swp
*~.nib
local.properties
.settings/
.loadpath
.recommenders

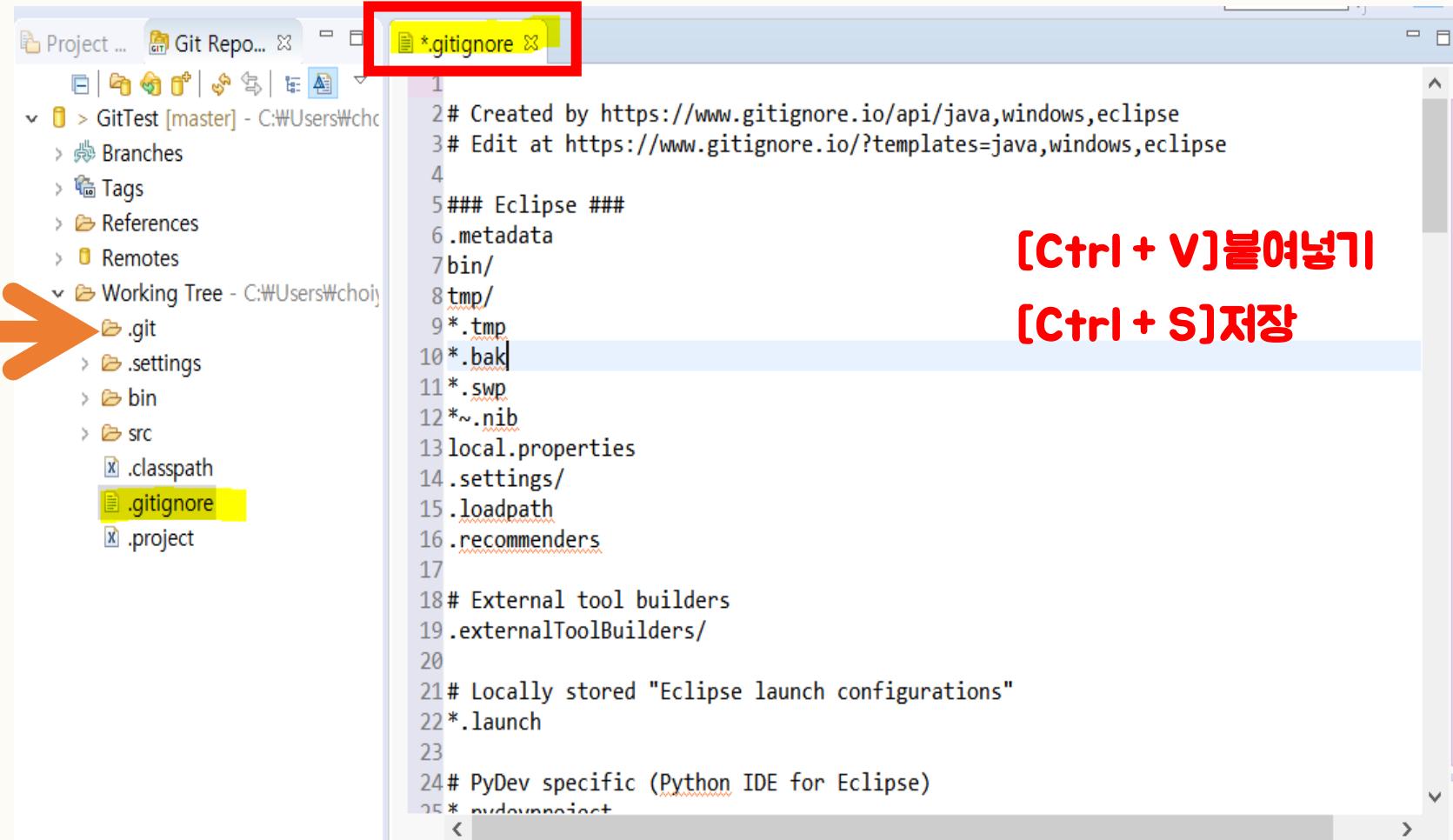
# External tool builders
.externalToolBuilders/

# Locally stored "Eclipse launch configurations"
.launch

# PyDev specific (Python IDE for Eclipse)
.pydevproject

# CDT-specific (C/C++ Development Tooling)
.cproject
```

[Ctrl + A] 전체선택
[Ctrl + C] 복사



```
*.gitignore
1
2# Created by https://www.gitignore.io/api/java,windows,eclipse
3# Edit at https://www.gitignore.io/?templates=java,windows,eclipse
4
5### Eclipse ####
6.metadata
7bin/
8tmp/
9*.tmp
10*.bak
11*.swp
12*~.nib
13local.properties
14.settings/
15.loadpath
16.recommenders
17
18# External tool builders
19.externalToolBuilders/
20
21# Locally stored "Eclipse launch configurations"
22.launch
23
24# PyDev specific (Python IDE for Eclipse)
25*.pydevproject
```

[Ctrl + V] 붙여넣기
[Ctrl + S] 저장

5. 팀장 - 3 Manage access 설정

The screenshot shows the GitHub repository settings for managing access. The left sidebar has a 'Manage access' section highlighted with a yellow box. The main area shows a 'Who has access' section with a 'DIRECT ACCESS' card indicating 3 collaborators. A large green button labeled 'Invite a collaborator' is highlighted with a red box and an orange arrow pointing to it. Below this, a modal window titled 'Invite a collaborator to DATABASE-PROGRAMMING' shows a search bar and a 'Select a collaborator above' button.

[Settings]-[Manage access]
-[invite a collaborator]-username입력 및 선택하여 초대

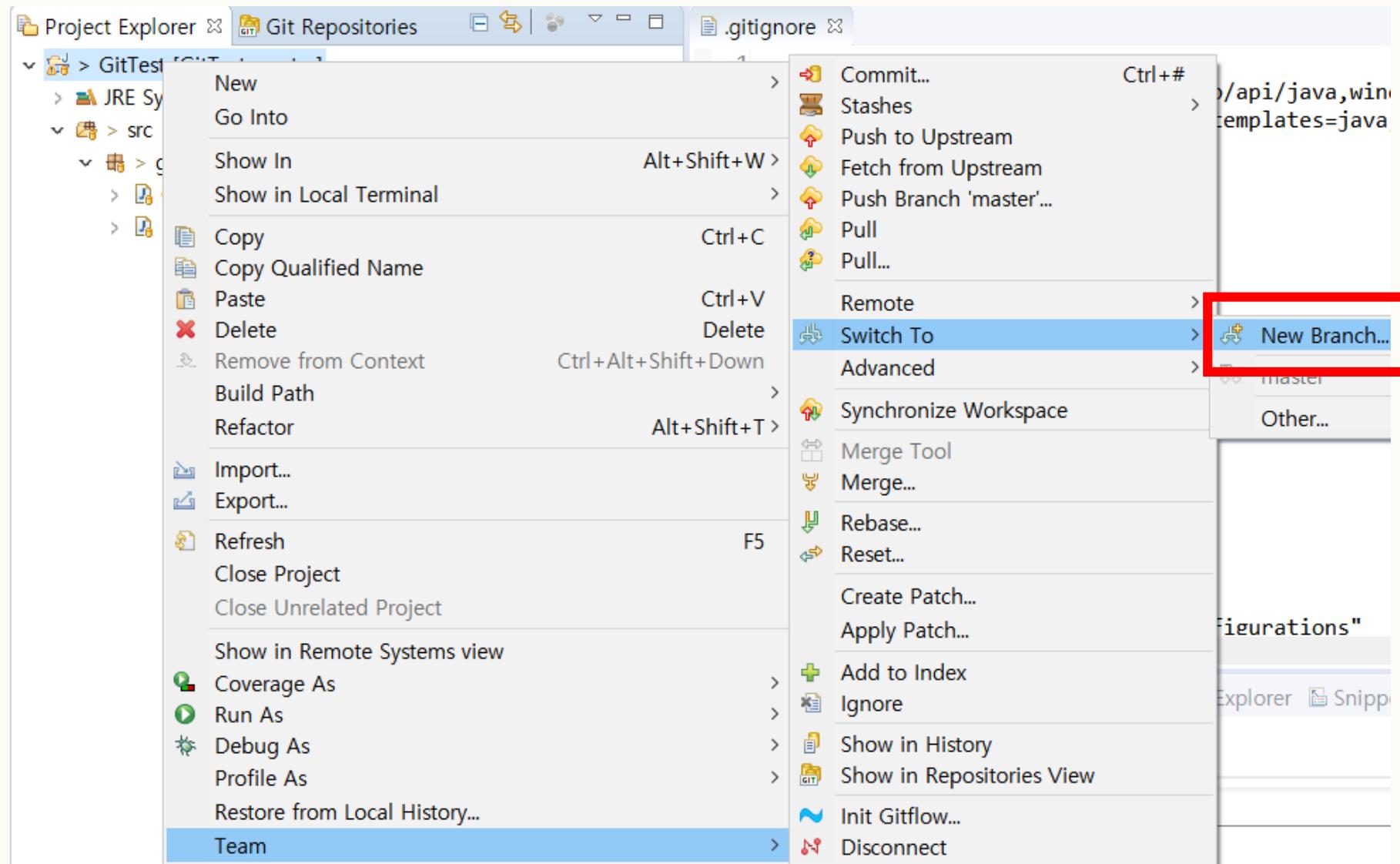
Invite a collaborator to DATABASE-PROGRAMMING

Search by username, full name, or email

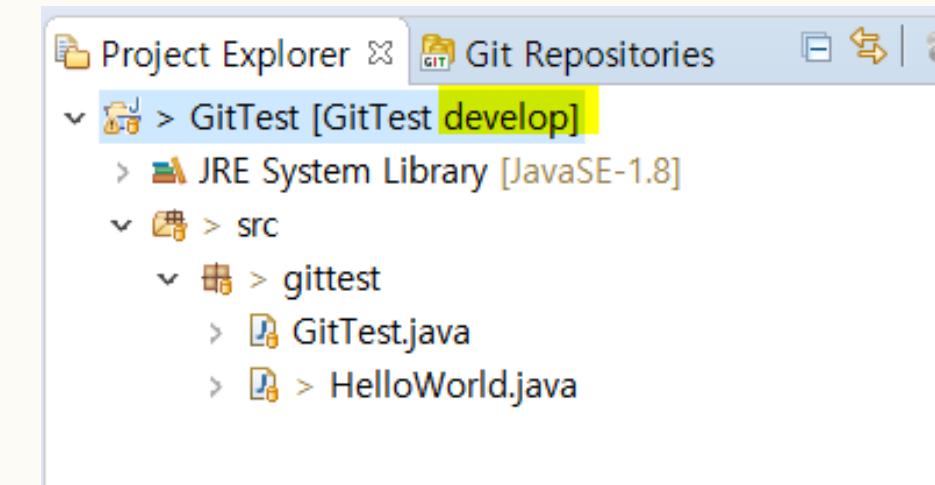
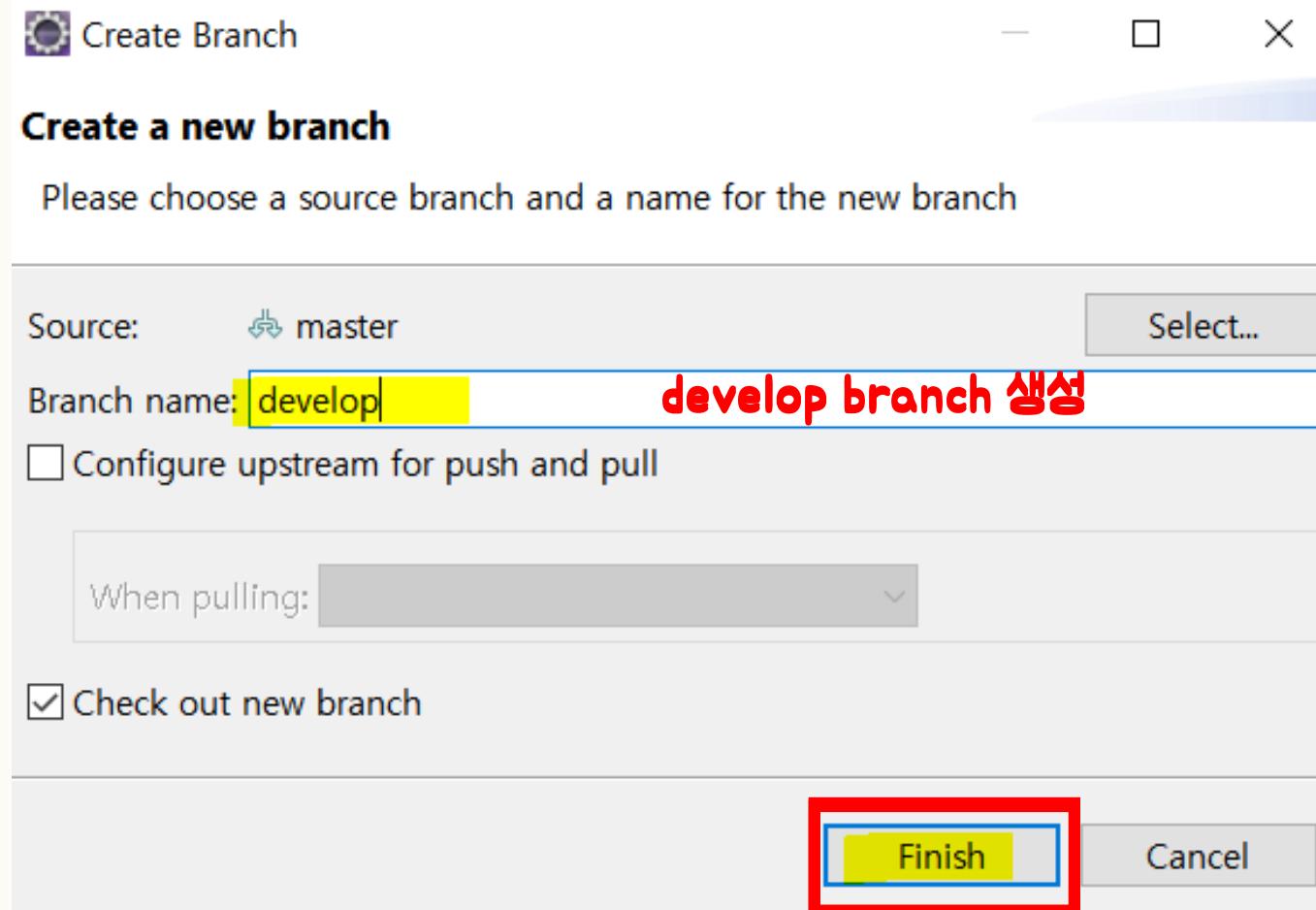
Select a collaborator above

53

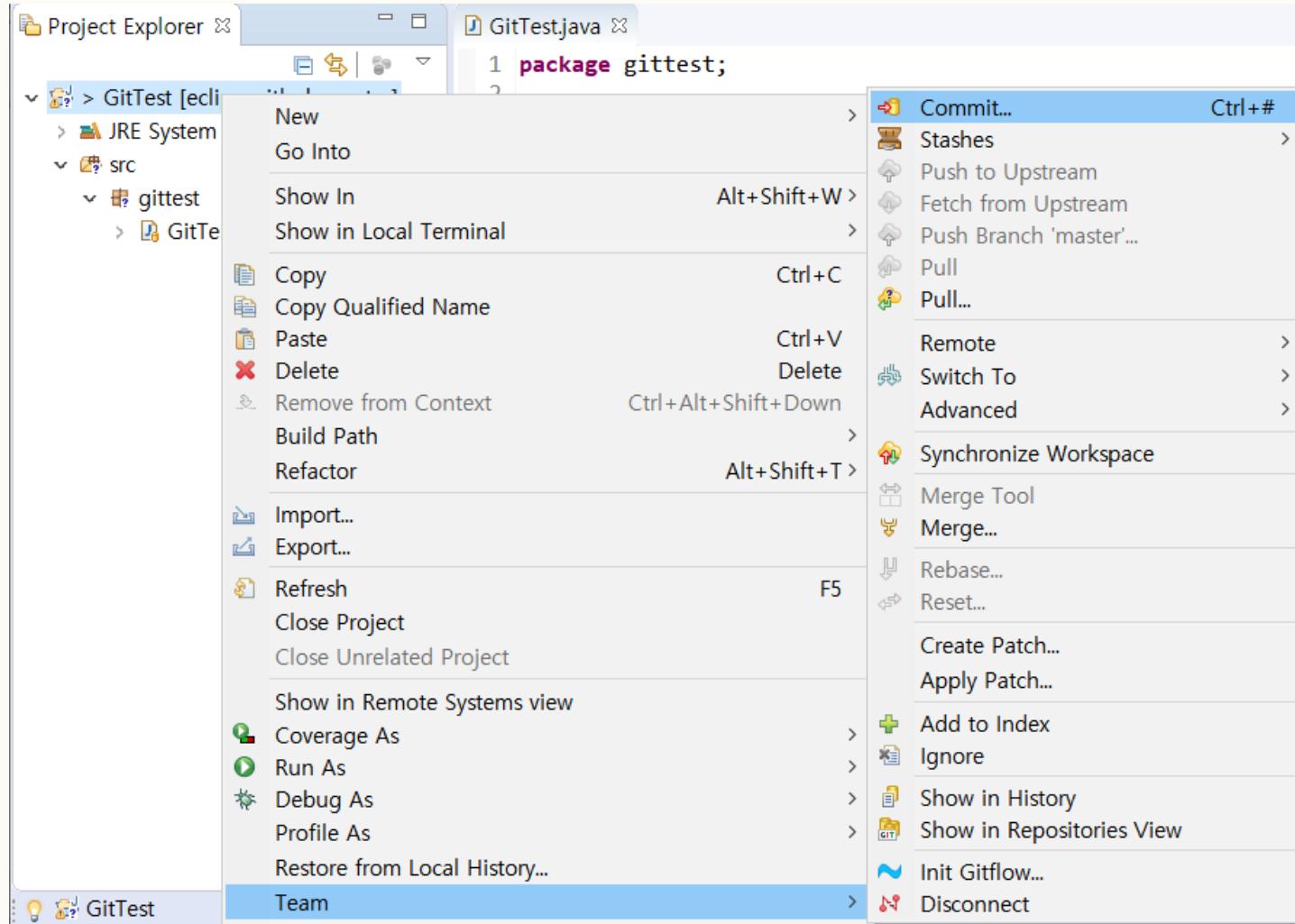
5. 팀장 - 4 Branch 생성 (develop)



5. 팀장 - 4 Branch 생성 (develop)

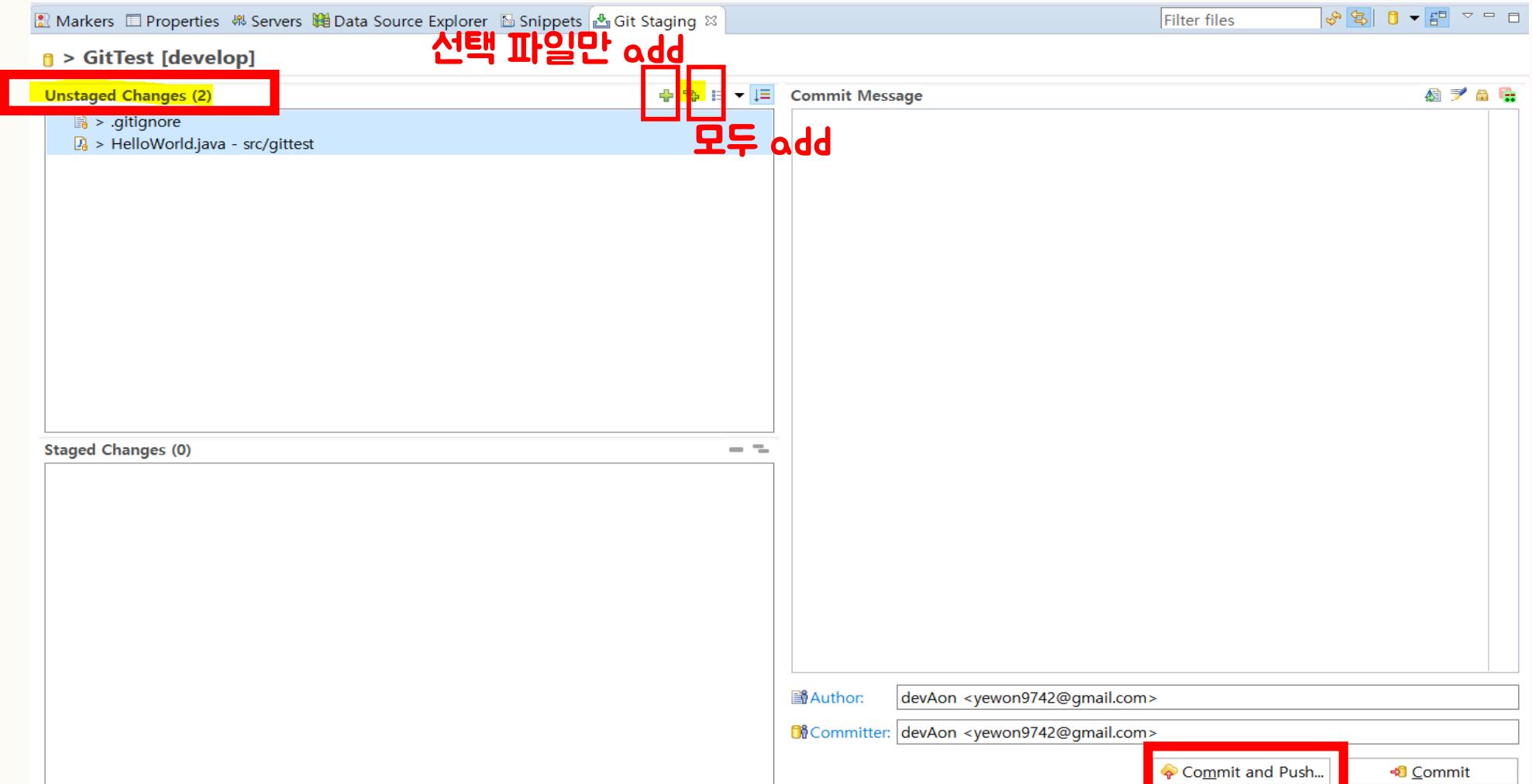


5. 팀장 - 5 Commit & Push



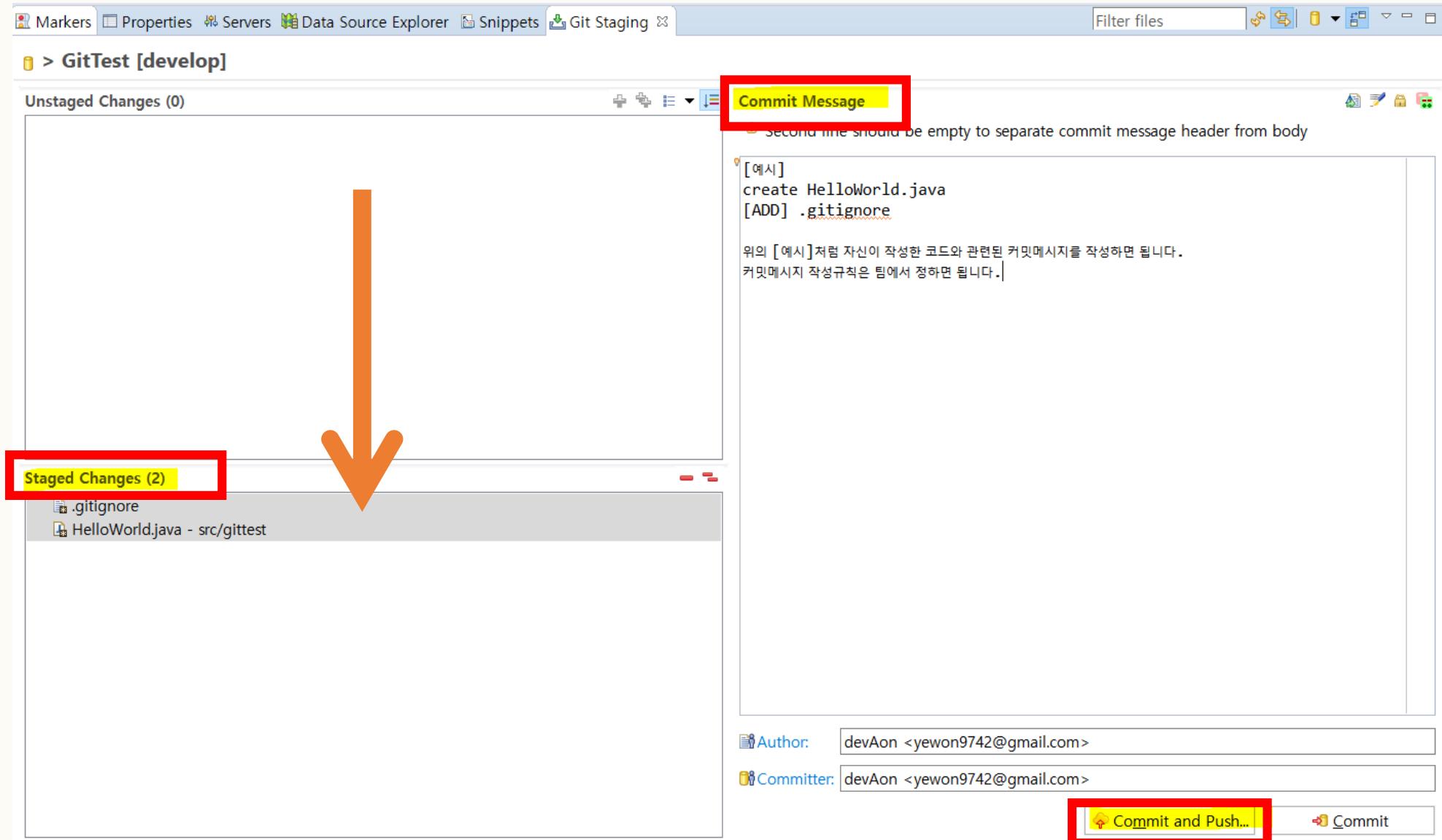
생성한 프로젝트에서 마우스 우클릭 - [Team]-[Commit]

5. 팀장 - 5 Commit & Push

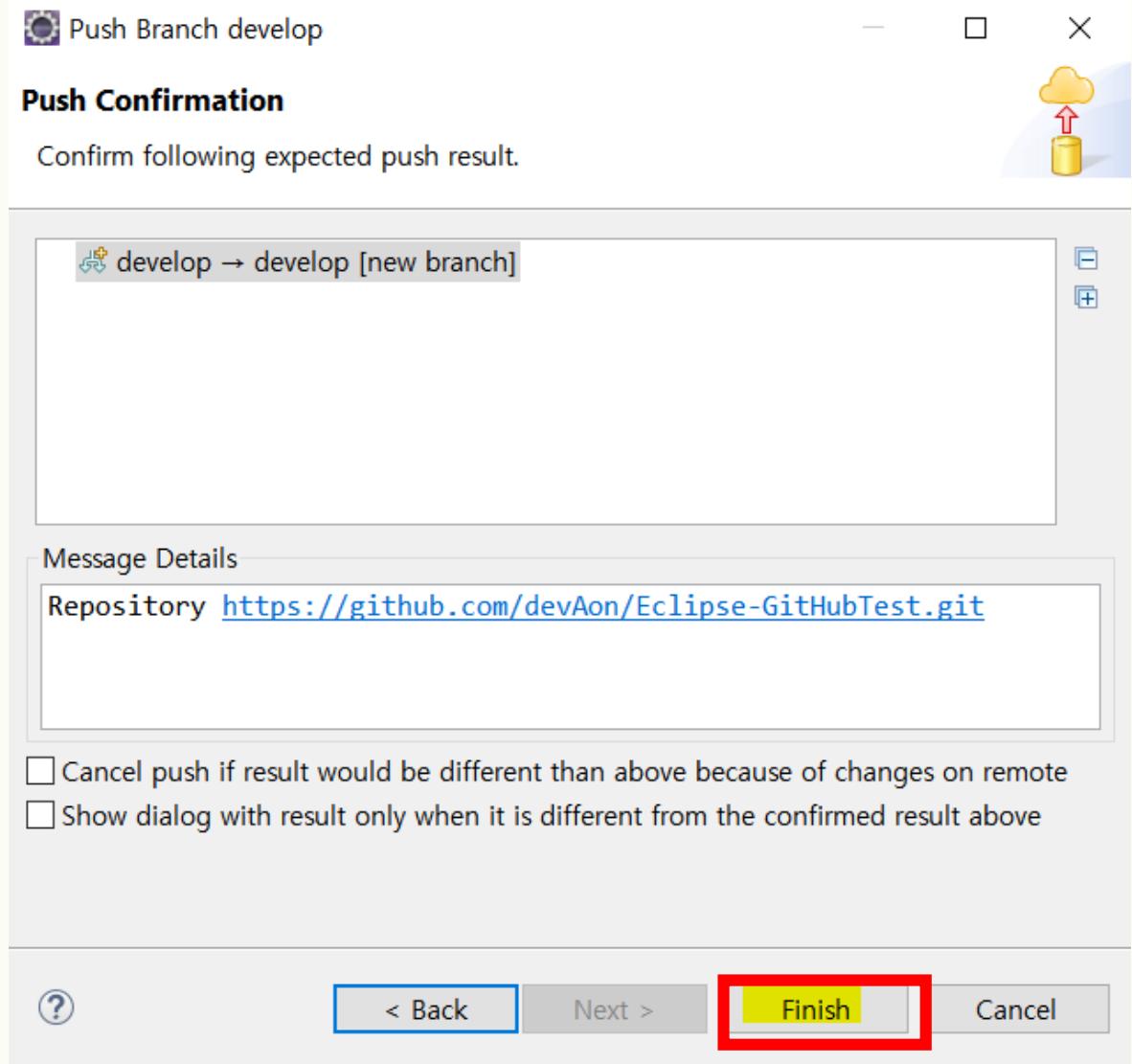
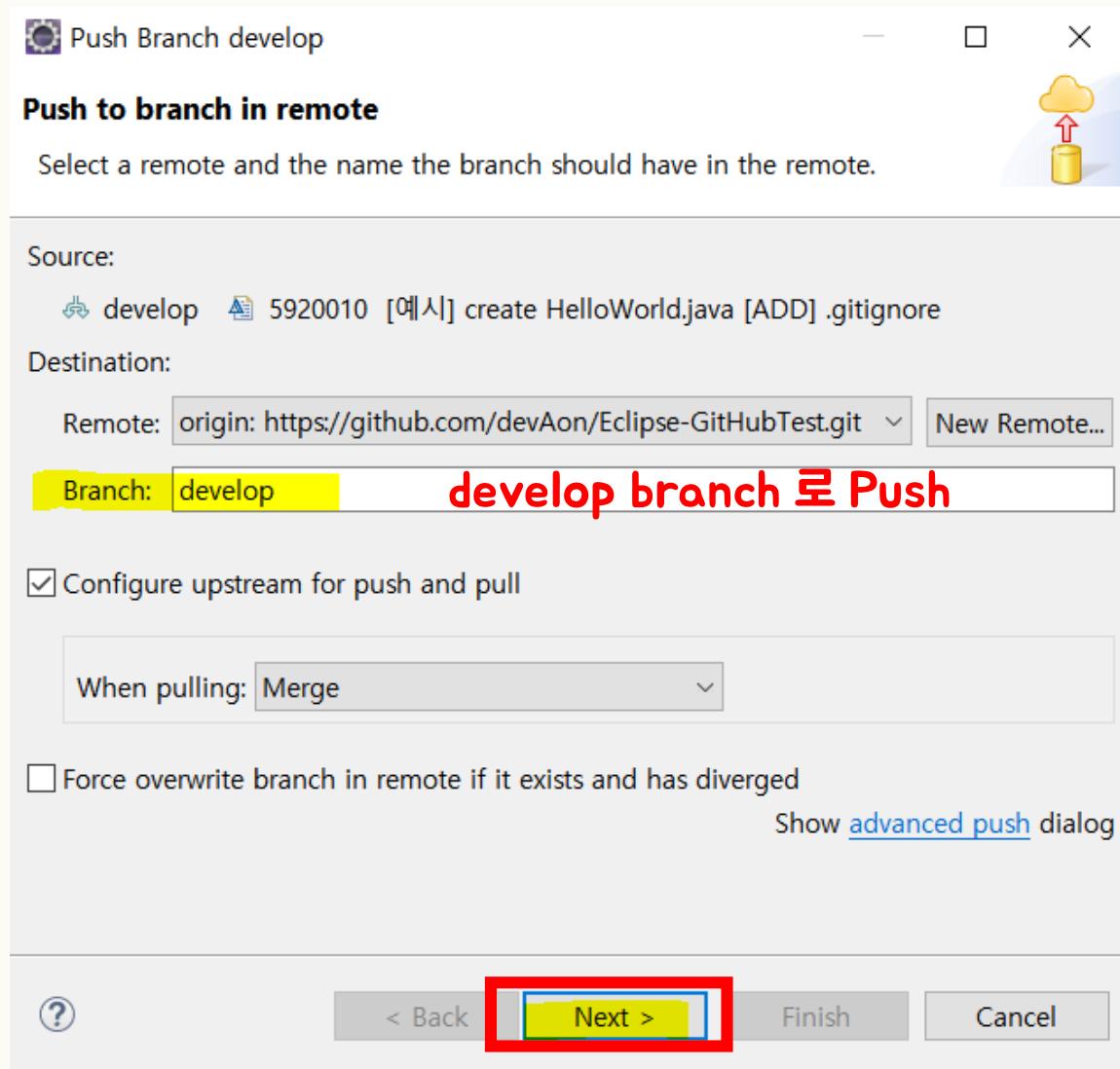


Unstaged Changes에 있는 파일 중 원격 저장소에 반영하고 싶은 파일을 선택해
Staged Changes로 add 해주면 된다.

5.팀장 - 5 Commit & Push



5. 팀장 - 5 Commit & Push



5.팀장 - 5 Commit & Push



5. 팀장 - 5 Commit & Push

팀장이 만든 저장소
devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

이클립스에서의 Git 사용법 - 테스트 Edit

Manage topics

3 commits 2 branches 0 packages 0 releases 1 contributor

Your recently pushed branches:

develop (5 minutes ago) Compare & pull request

Branch: develop New pull request Create new file Upload files Find file Clone or download

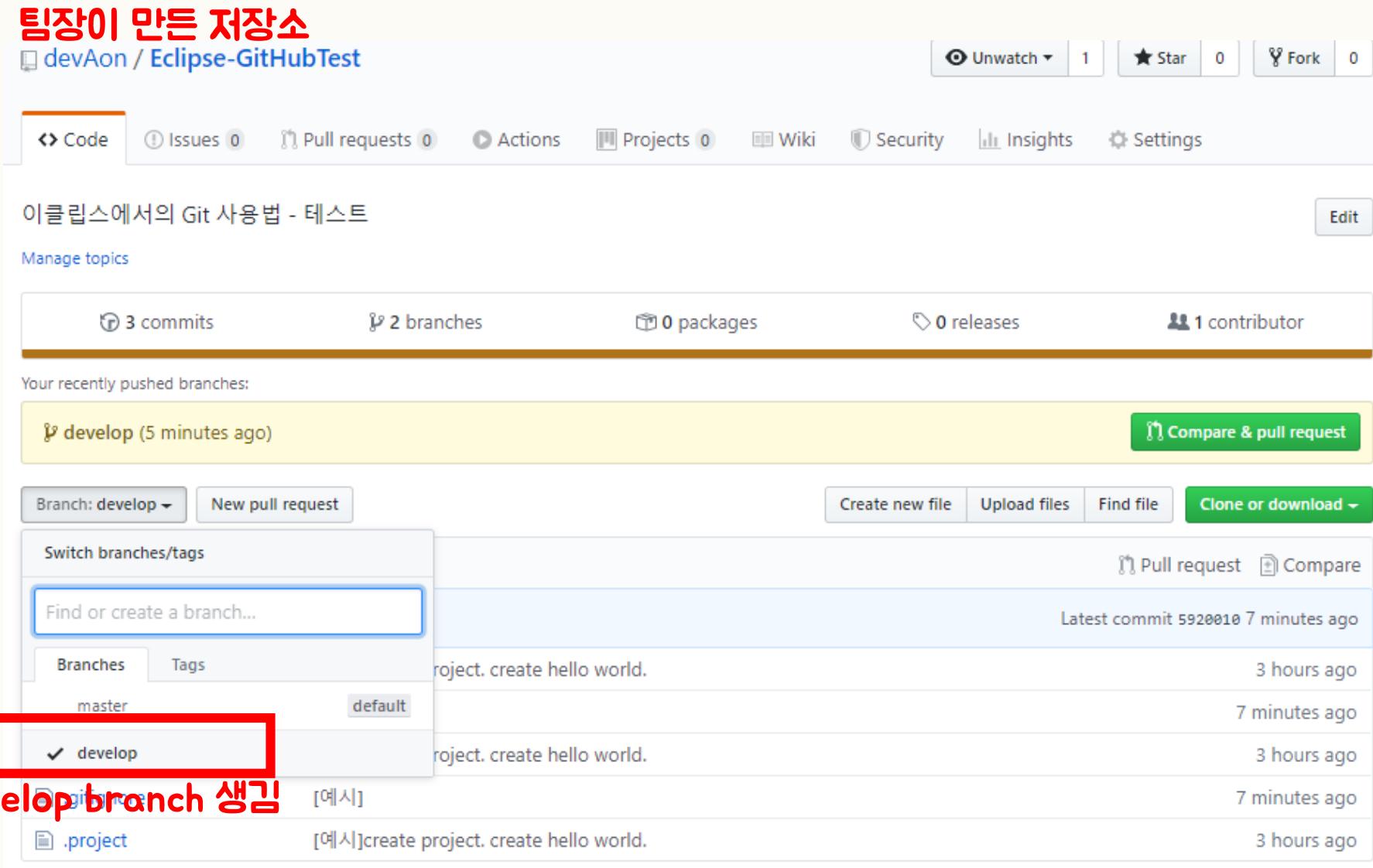
Switch branches/tags

Find or create a branch... Pull request Compare

Latest commit 5920010 7 minutes ago

| Commit Message | Time Ago |
|------------------------------------|---------------|
| project.create hello world. | 3 hours ago |
| project.create hello world. | 7 minutes ago |
| project.create hello world. | 3 hours ago |
| create project.create hello world. | 7 minutes ago |
| .project | 3 hours ago |

Github에 develop branch 생김



5. 팀장 - 5 Commit & Push

팀장이 만든 저장소

devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 1 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

Options

Manage access

Branches

Webhooks

Default branch

The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

develop

Update

팀장이 만든 repository에서 [Settings]-[Branches]

Default branch를 develop으로 변경 후 update

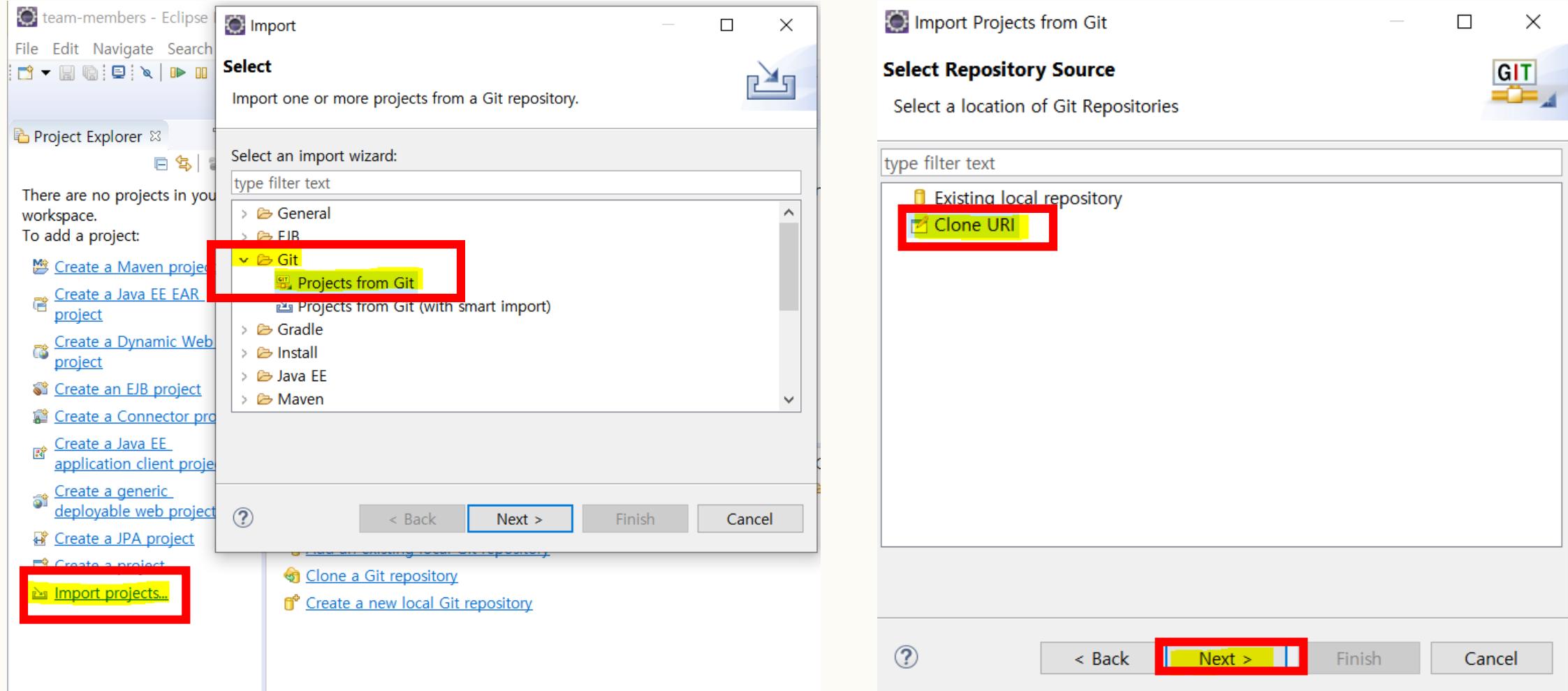
develop branch를 default로 설정하는 이유?

평소 develop branch 기반으로 개발을 진행하기 때문

팀원



5. 팀원 - 1 저장소 복제 - clone



[Import Projects]-[Git]-[Projects from Git]-[Clone URI]-[Next]

5.팀원 - 1 저장소 복제 - clone

팀장이 만든 저장소

devAon / Eclipse-GitHubTest

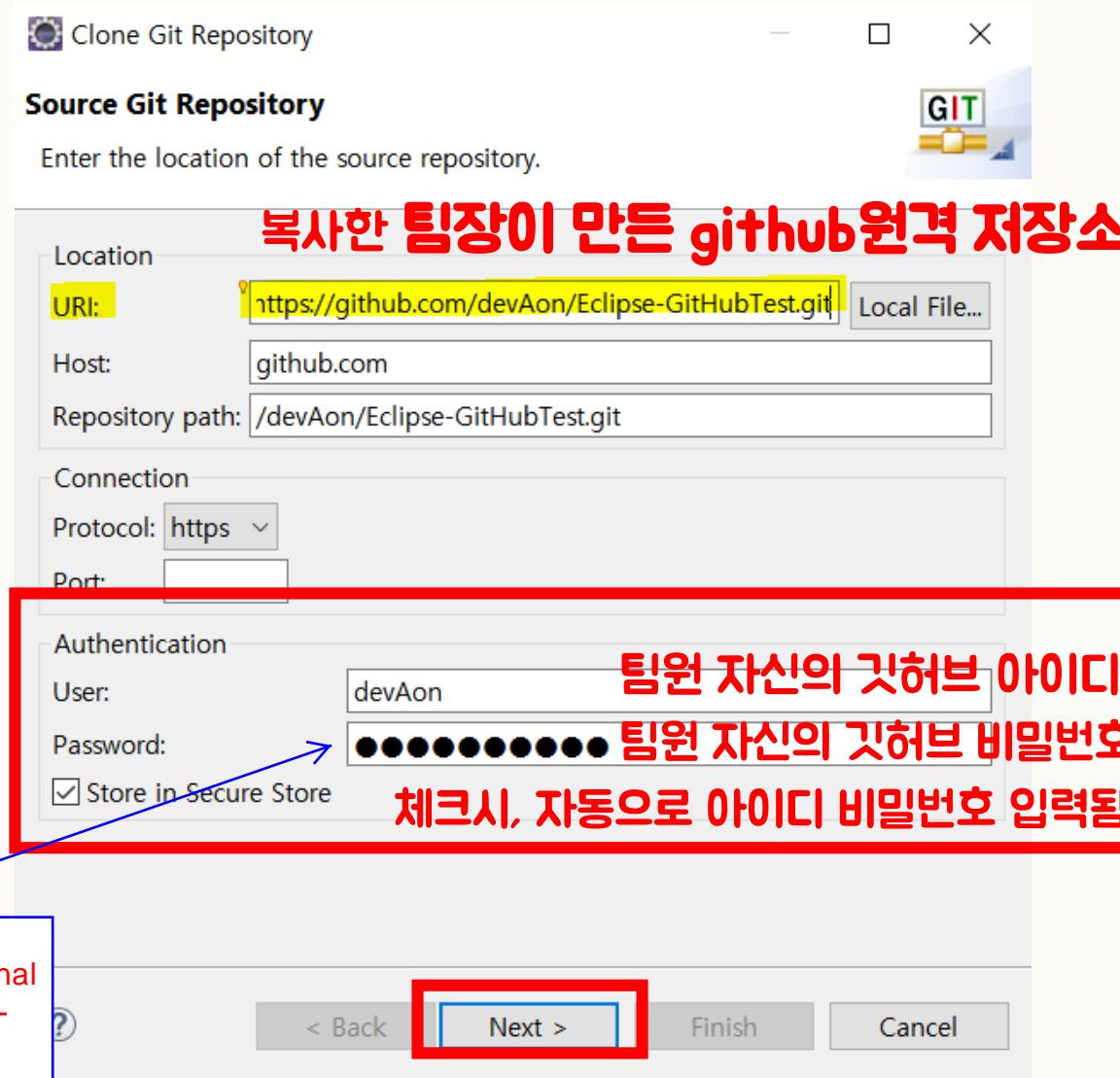
Unwatch 1 Star 0 Fork 0

The screenshot shows a GitHub repository page for 'devAon / Eclipse-GitHubTest'. At the top, there are navigation links for Code, Issues (0), Pull requests (0), Actions, Projects (0), Wiki, Security, Insights, and Settings. Below the header, the repository name is displayed with a 'Edit' button. A section titled 'Manage topics' follows. The repository statistics are shown: 2 commits, 1 branch, 0 packages, 0 releases, and 1 contributor. A red arrow points to the 'Clone or download' button, which is highlighted with a green border. A tooltip for 'Clone with HTTPS' is visible, showing the URL <https://github.com/devAon/Eclipse-GitHubTest>. The repository's contents are listed below, including .settings, src/gittest, .classpath, .gitignore, and .project files. A message at the bottom encourages adding a README, with a 'Add a README' button.

팀장이 만든 저장소 주소 Clone

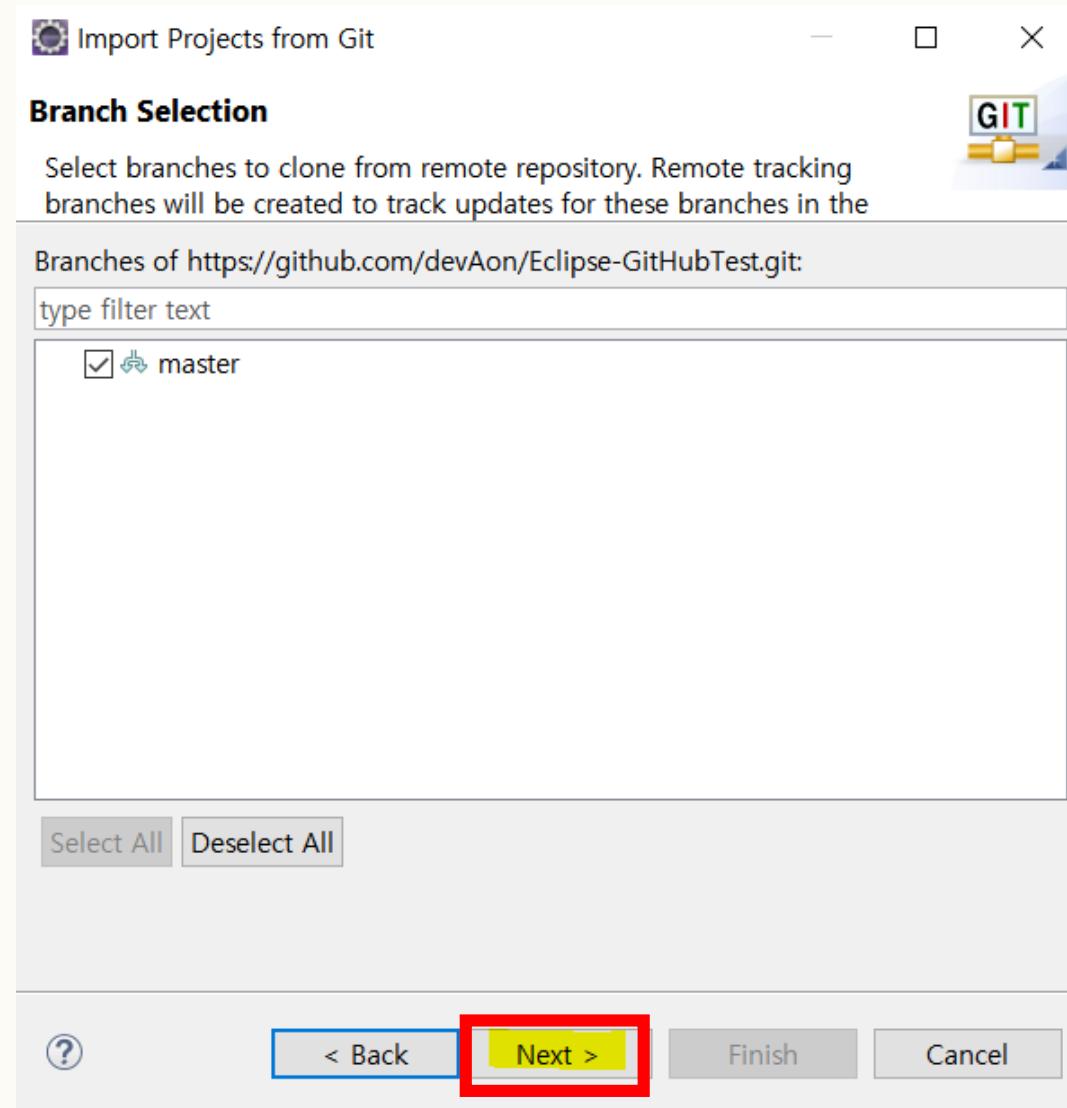
[Clone or download]-[copy] 저장소 주소 복사

5.팀원 - 1 저장소 복제 - clone

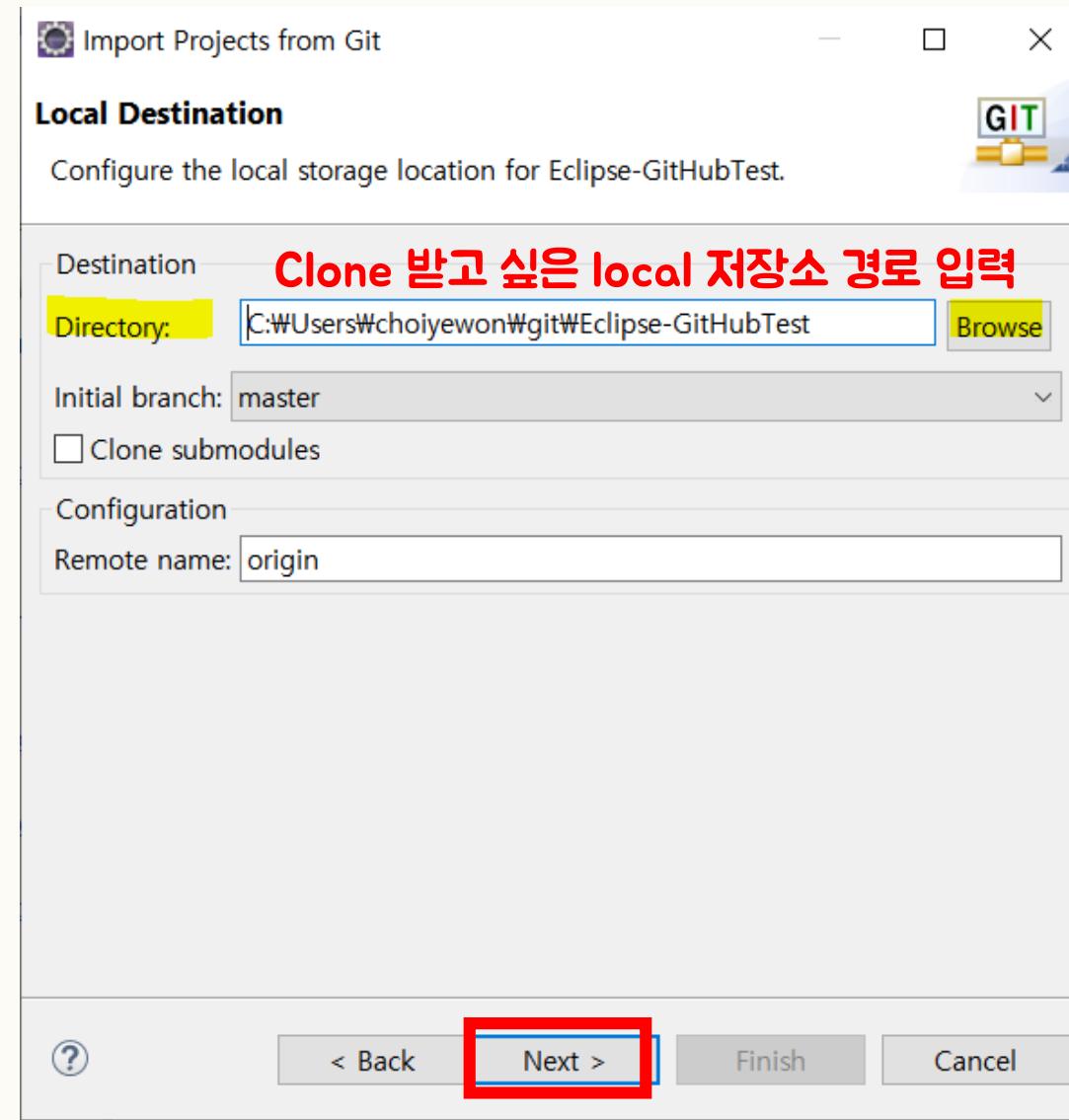


* : 2021 8 Github
Personal Access Token
Access Token
geno.tistory.com/89)
(Personal
https://kitty -
)

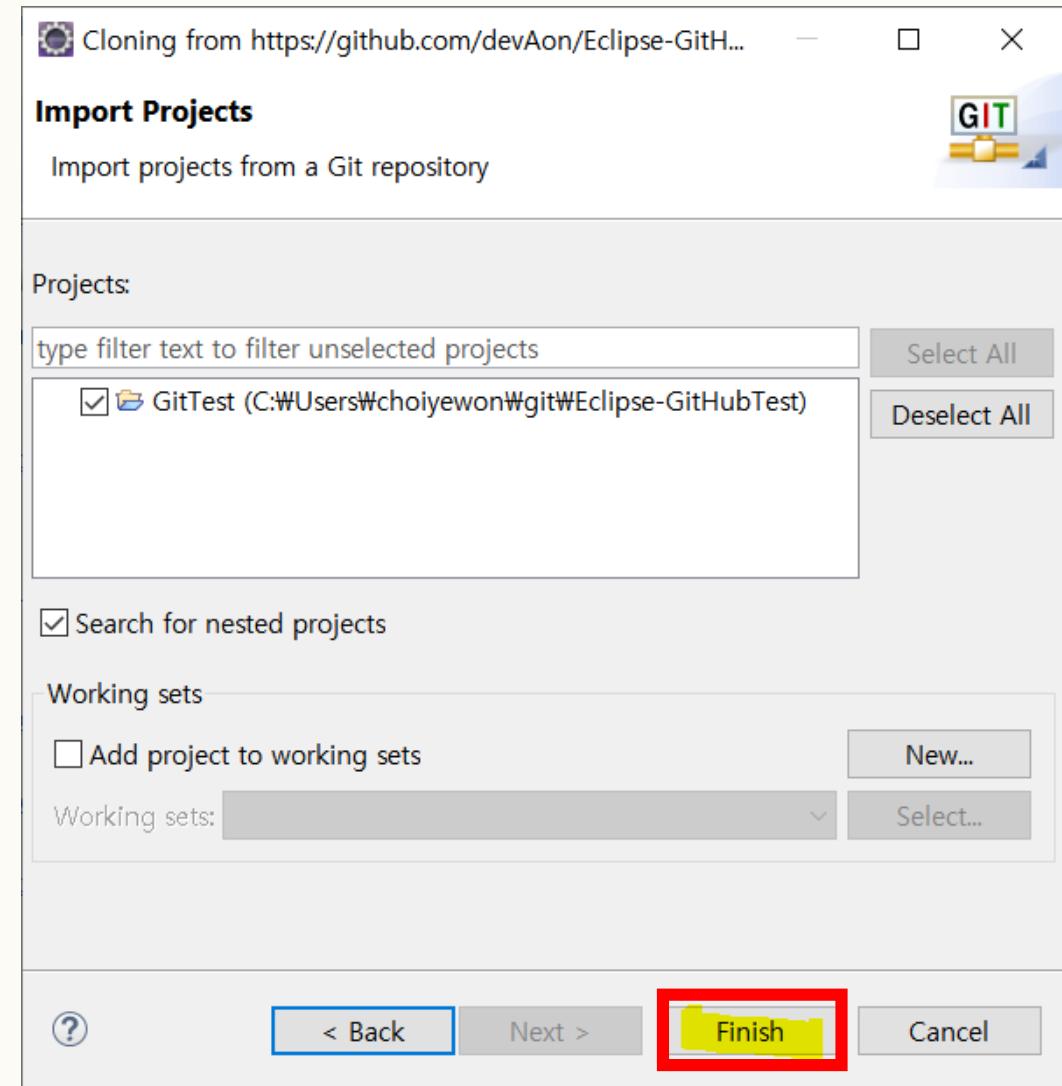
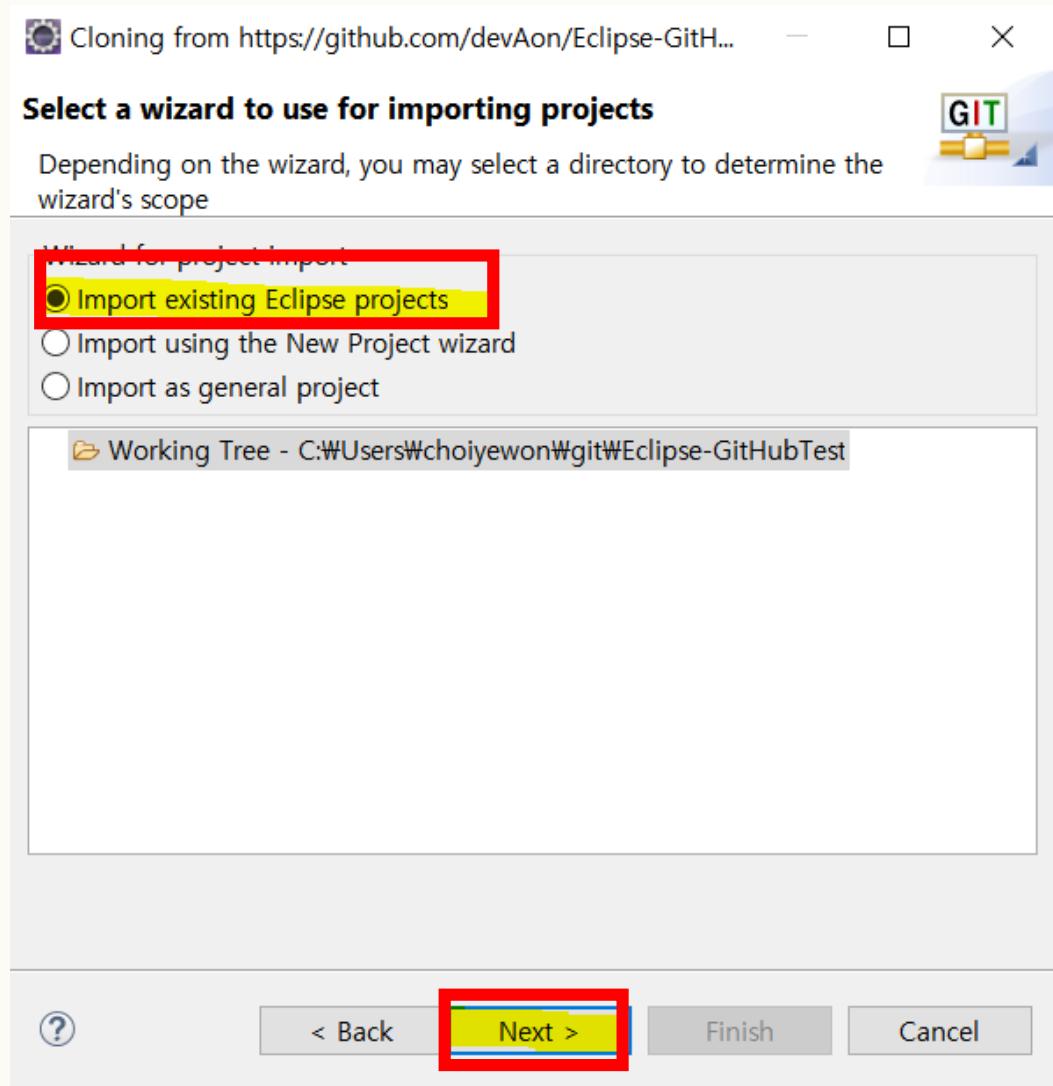
5.팀원 - 1 저장소 복제 - clone



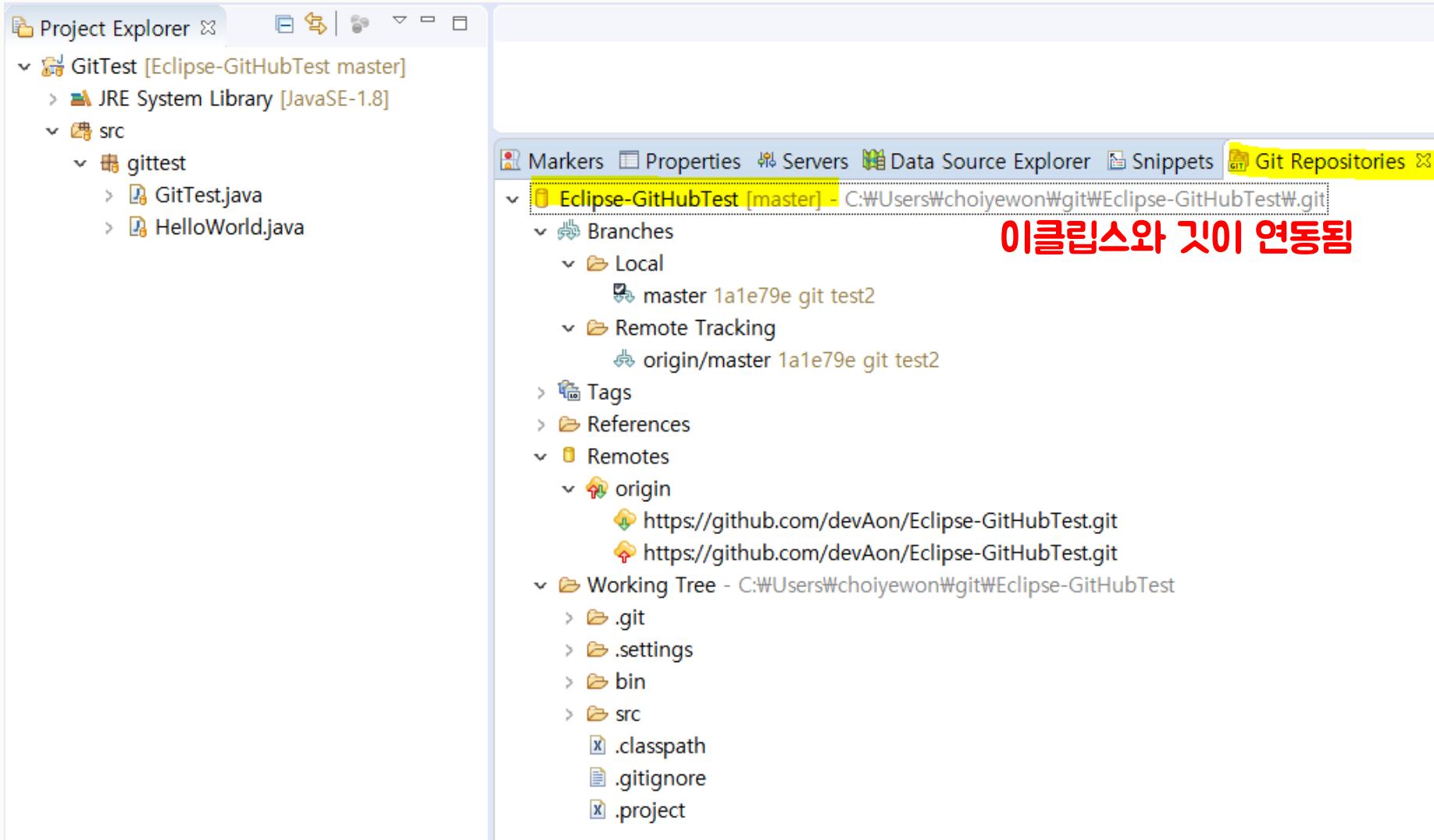
5.팀원 - 1 저장소 복제 - clone



5. 팀원 - 1 저장소 복제 - clone



5. 팀원 - 1 저장소 복제 - clone



5. 팀원 - 2 fork

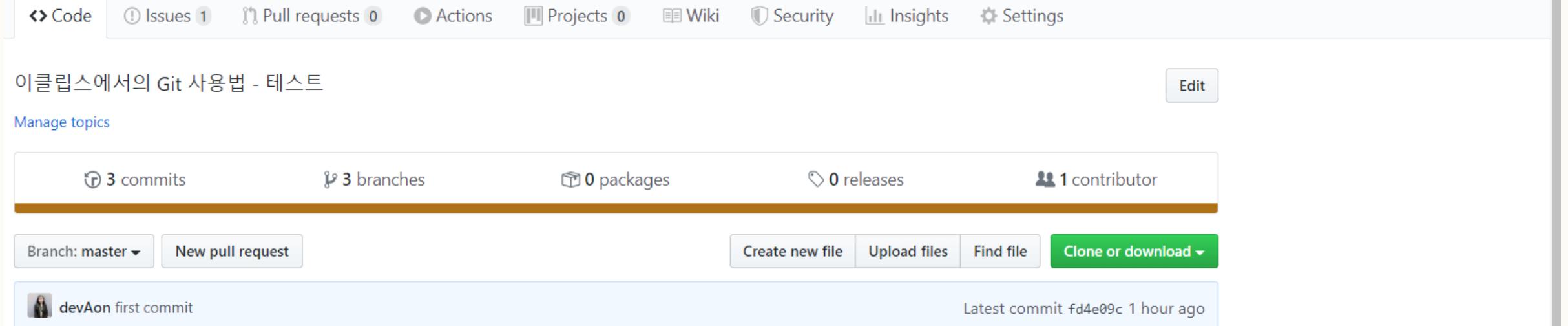
(fork는 선택사항)

이유? 팀장이 만든 저장소와 local저장소만 사용할 것이기 때문
팀장이 만든 저장소에 권한을 주면 fork하지 않아도 팀장이 만든 저장소에
직접적으로 코드를 반영할 수 있다.

그러나, 자신의 저장소에도 해당 저장소를 가져오고 싶다면
Fork 하여 자신의 저장소에 가져올 수 있다.

팀장이 만든 저장소

devAon / Eclipse-GitHubTest



The image shows a detailed view of the same GitHub repository page. It includes a navigation bar with 'Code', 'Issues 1', 'Pull requests 0', 'Actions', 'Projects 0', 'Wiki', 'Security', 'Insights', and 'Settings'. Below the navigation is a section titled '이클립스에서의 Git 사용법 - 테스트' with an 'Edit' button. A 'Manage topics' link is also present. Key statistics are listed: '3 commits', '3 branches', '0 packages', '0 releases', and '1 contributor'. A dropdown menu shows 'Branch: master' and a 'New pull request' button. At the bottom, there are buttons for 'Create new file', 'Upload files', 'Find file', and a green 'Clone or download' button. A commit history shows 'devAon first commit' and 'Latest commit fd4e09c 1 hour ago'.

[Fork]-팀장이 만든 repository를 나의 repositories로 fork

5. 팀원 - 2 fork (프로젝트 끝난 후 다시 읽어보기)

프로젝트가 끝난 후,

팀장이 develop의 코드를 master에 merge까지 완료된 상태

팀장의 저장소에 있는 프로젝트 코드를 내 저장소에도 반영하고 싶은 경우

\$git clone [자신의 원격 저장소 url]

\$cd [.git이 있는 파일] // .git이 있는 파일로 이동 ex)\$ cd Eclipse-GitHubTest

\$git remote -v // origin 자신의 저장소

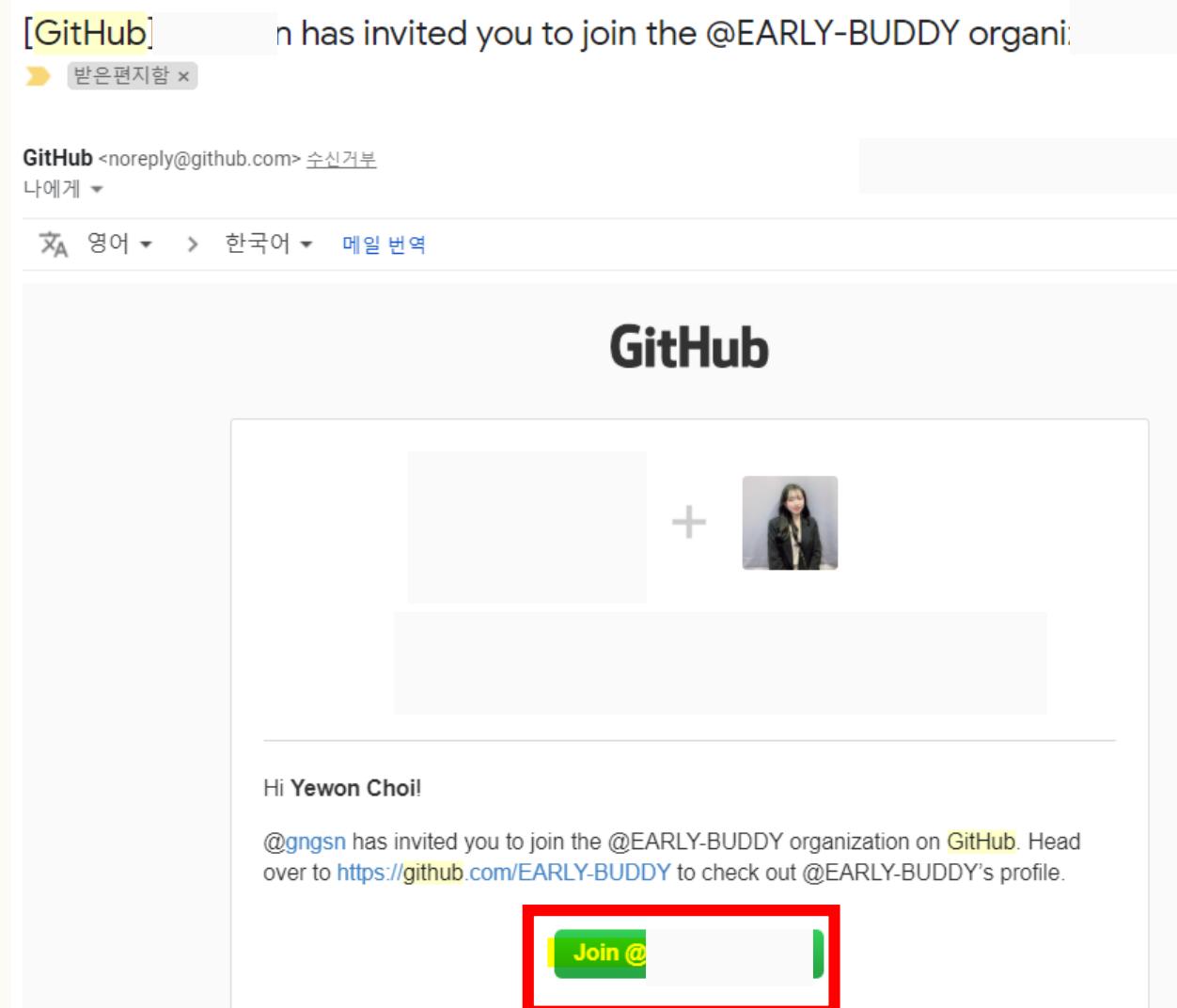
\$git remote add upstream [팀장의 원격 저장소 url]

\$git pull upstream master // 팀장이 develop의 코드를 master에 merge까지 완료된 상태

// master에 merge되지 않고 develop 코드가 최종 코드일 경우엔 master 대신 develop
// ex) git pull upstream develop

\$git push origin master // 내 저장소의 master branch에 코드 반영됨

5. 팀원 - 3 e-mail에서 팀 권한 수락



반드시 깃허브에 등록한 이메일에 들어가
팀 권한을 수락하기
그래야 권한을 부여 받을 수 있다.

Git PRESENTATION

06 협업(공통)

6.1 협업방식 적용 이유

4장에서 설명한 **Git Flow** 전략을 적용하여 협업 하는 방식을 설명할 것이다.

앞서 말했듯이 Git Flow 전략은 프로젝트 상황에 맞게 커스텀하여 사용할 수 있다.

5가지 브랜치 중 **master-develop-feature** 브랜치로만 프로젝트를 형상관리하려 한다.
(hotfix, release 사용 안함)

그 이유는?

단기 프로젝트 및 소규모 인원이기에 3가지 브랜치로 형상관리 하는 것이
가장 적합하다 판단했기 때문이다.

6.2 협업방식 적용 장점

장점에 대해 다시 말하면?

1. 다수의 팀 구성원이 메인 코드 베이스(master)를 중심으로 하기 때문에 안전하게 새로운 기능을 개발할 수 있다.
2. feature branch workflow와 Pull Request를 결합하면 팀 구성원간 변경 내용에 대한 소통을 촉진해 코드 품질을 높일 수 있다.
3. 유연성이 큰 협업 방법이다.

6.2 협업방식 적용 장점

5장을 수행했다면 협업하기 위한 기반이 준비됐다.

6장에서는 팀장, 팀원 모두에게 해당 되는 내용으로

맡은 기능을 구현하고 develop branch에 합치는 협업방식을 설명할 것이다.

1. 다수의 팀 구성원이 메인 코드 베이스(master)를 중심으로 하기 때문에 안전하게 새로운 기능을 개발할 수 있다.
2. feature branch workflow와 Pull Request를 결합하면 팀 구성원간 변경 내용에 대한 소통을 촉진해 코드 품질을 높일 수 있다.
3. 유연성이 큰 협업 방법이다.

6. 협업 - 1 Issue

The screenshot shows a GitHub repository page for 'devAon / Eclipse-GitHubTest'. The top navigation bar includes links for Code, Issues (0), Pull requests (0), Actions, Projects (0), Wiki, Security, Insights, and Settings. The 'Issues' link is highlighted with a yellow box and a red border. In the center, a modal window titled 'Label issues and pull requests for new contributors' informs users that GitHub will help potential first-time contributors discover issues labeled with 'good first issue'. Below the modal, there are filters ('Filters ▾ is:issue is:open'), a search bar, and buttons for Labels (9) and Milestones (0). A green 'New issue' button is highlighted with a red box. The main content area features a large exclamation mark icon and the text 'Welcome to Issues!'. It explains that issues are used to track todos, bugs, feature requests, and more, and encourages users to create an issue.

devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

Label issues and pull requests for new contributors

Now, GitHub will help potential first-time contributors discover issues labeled with good first issue Dismiss

Filters ▾ is:issue is:open

Labels 9 Milestones 0

New issue

!

Welcome to Issues!

Issues are used to track todos, bugs, feature requests, and more. As issues are created, they'll appear here in a searchable and filterable list. To get started, you should [create an issue](#).

6. 협업 - 1 Issue

The screenshot shows a GitHub repository page for 'devAon / Eclipse-GitHubTest'. The 'Issues' tab is selected, showing a count of 0. A red box highlights the title bar '회원가입 구현' (Implementation of membership) and the main title '구현할 기능 작성' (Write the feature to implement). Another red box highlights the rich text editor toolbar and the text input area containing 'issue test' and 'issue 작성방법 설명에 사용할 예시 이슈'. A third red box highlights the 'Assignees' section, which lists 'devAon (Optional)' and the 'Labels' section, which lists '(Optional)' and 'enhancement' (highlighted in blue). A fourth red box highlights the 'Submit new issue' button at the bottom right.

devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

회원가입 구현 구현할 기능 작성

Related Issues Beta Try it.

Assignees devAon (Optional)

Labels (Optional) enhancement

Projects None yet

Milestone No milestone

Linked pull requests Successfully merging a pull request may close this issue.

None yet

Helpful resources GitHub Community Guidelines

Write Preview AA B i “ ‘ < > @ ↵

issue test
issue 작성방법 설명에 사용할 예시 이슈

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

Styling with Markdown is supported

Submit new issue

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

6. 협업 - 1 Issue

devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 1 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

회원가입 구현 #1 **이슈번호를 feature 번호로 사용할 것이다.**

Open devAon opened this issue now · 0 comments

devAon commented now

issue test
issue 작성방법 설명에 사용할 예시 이슈

Owner + 😊 ...

Assignees devAon

Labels enhancement

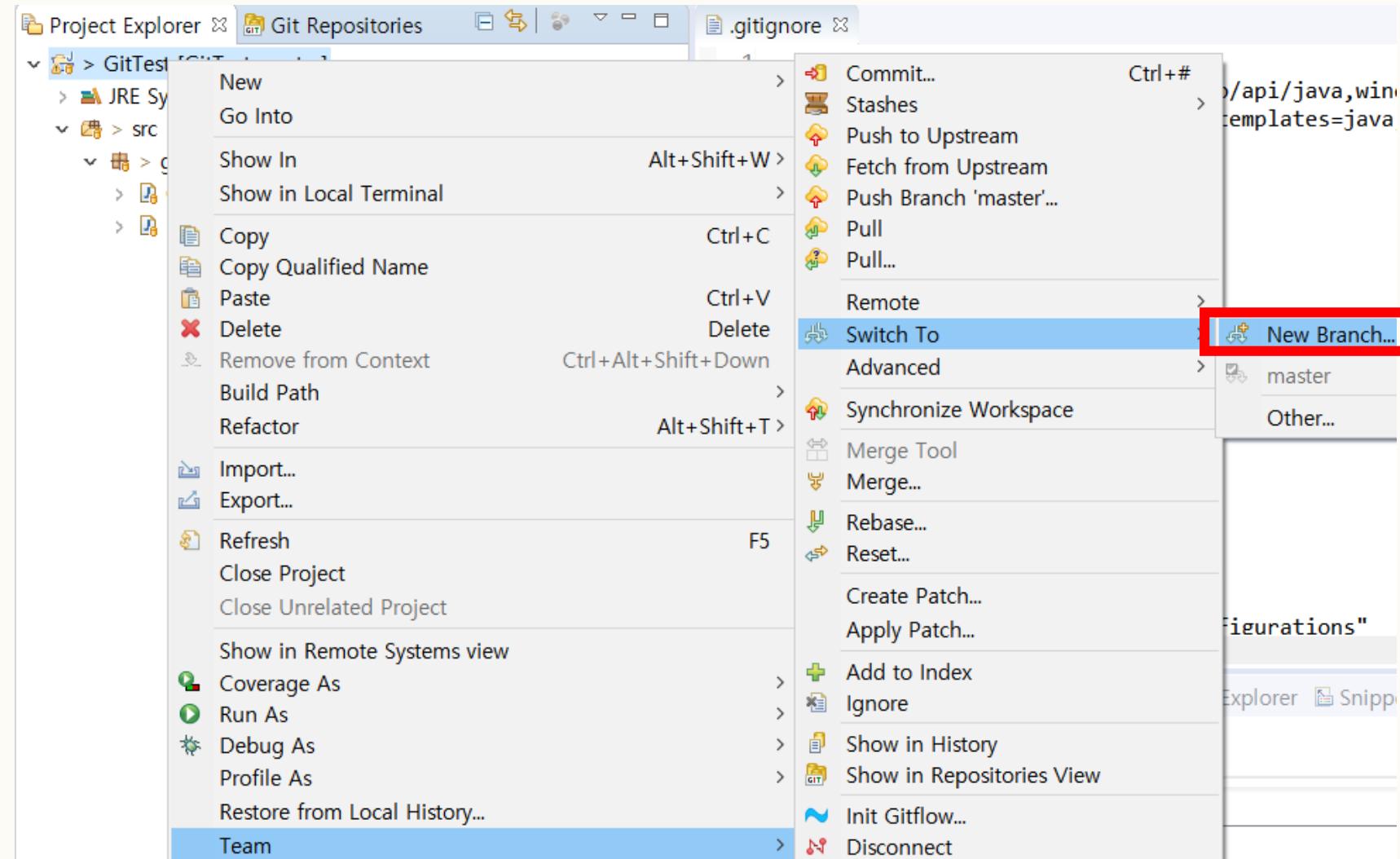
Projects None yet

Milestone

devAon added the enhancement label now

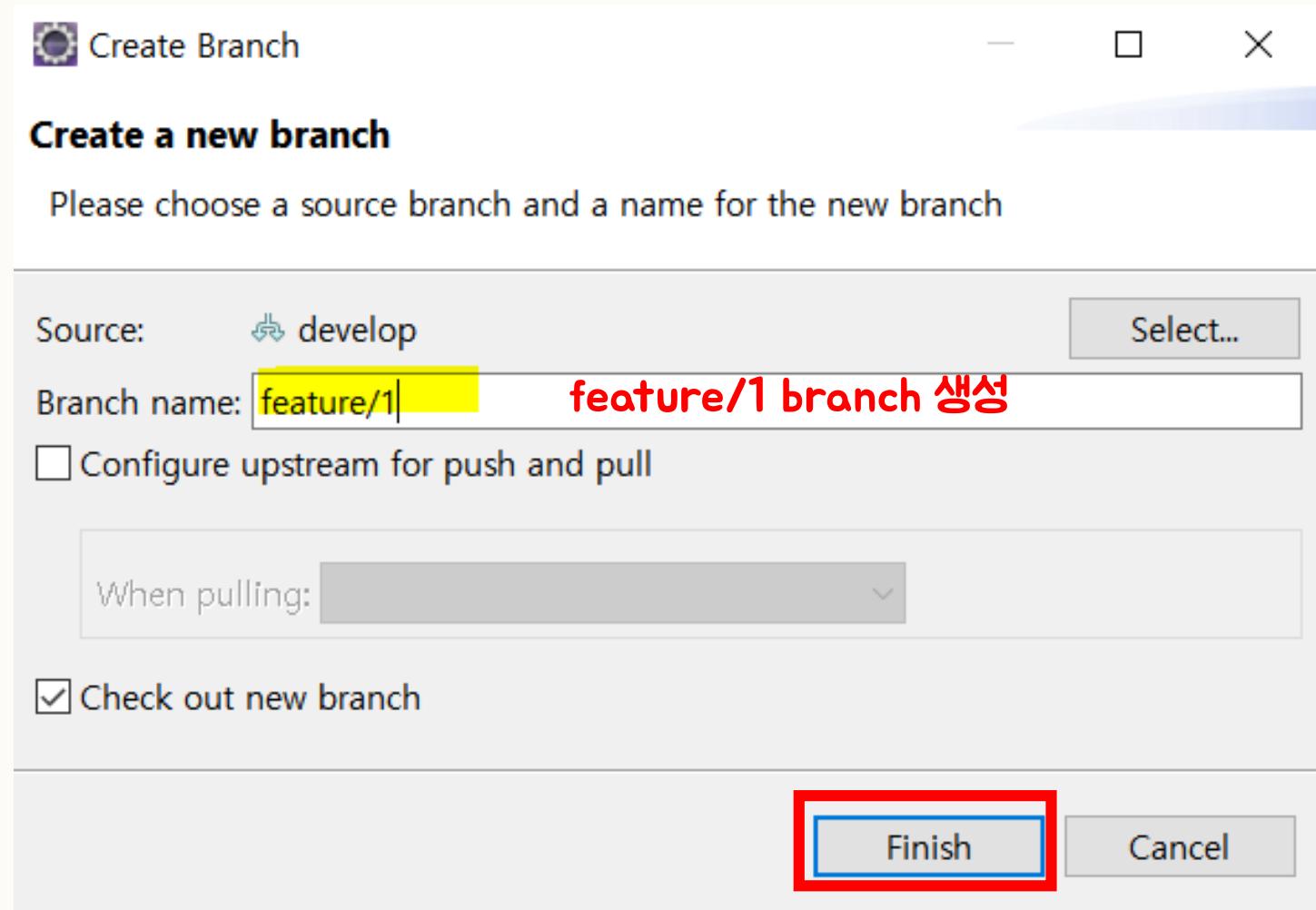
devAon self-assigned this now

6. 협업 - 2 Branch 생성 및 작업

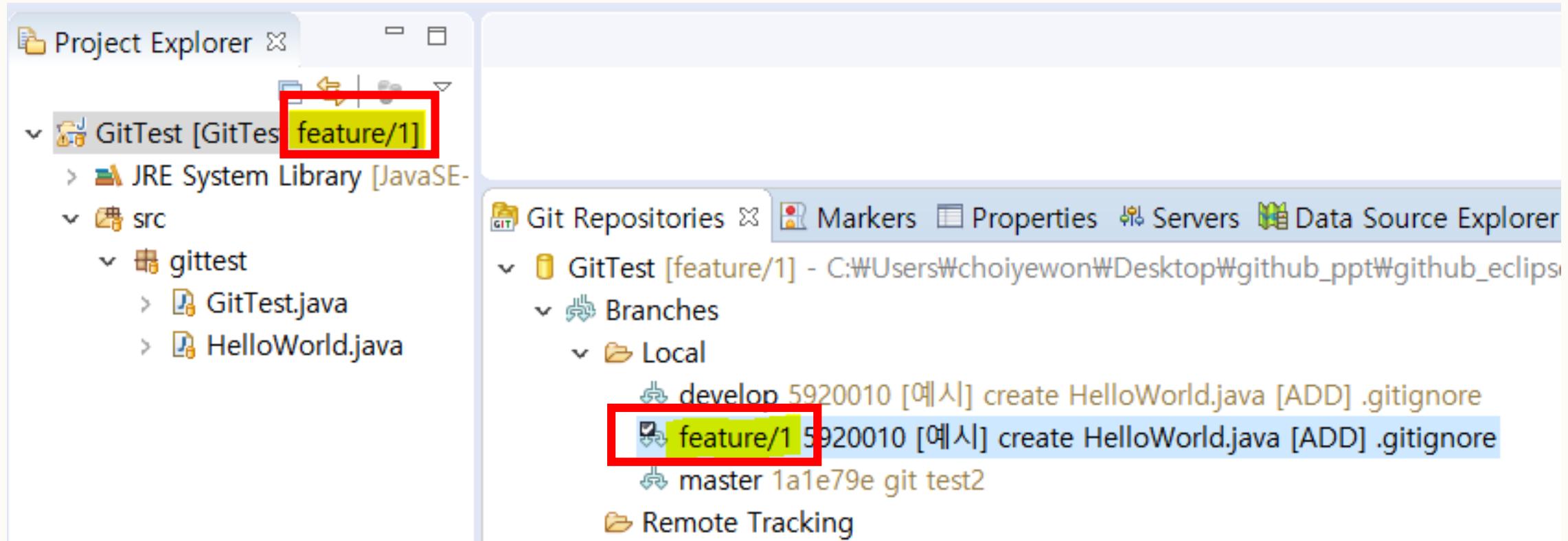


생성한 프로젝트에서 마우스 우클릭 - [Team]-[Switch To]-[New Branch]

6. 협업 - 2 Branch 생성 및 작업

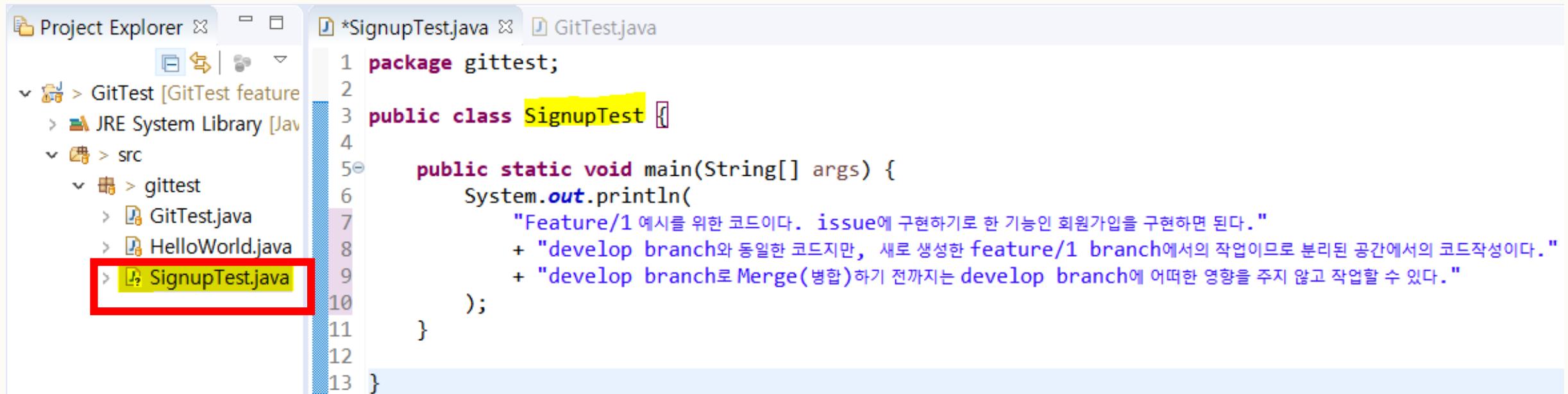


6. 협업 - 2 Branch 생성 및 작업



Feature/1 branch가 생성되고 해당 브랜치로 이동한 것을 확인할 수 있다.

6. 협업 - 2 Branch 생성 및 작업



The screenshot shows an IDE interface with two main panes. The left pane is the Project Explorer, displaying a project structure with a 'GitTest' feature branch under 'src/gittest'. The right pane is the Editor, showing the code for 'SignupTest.java'. The code is as follows:

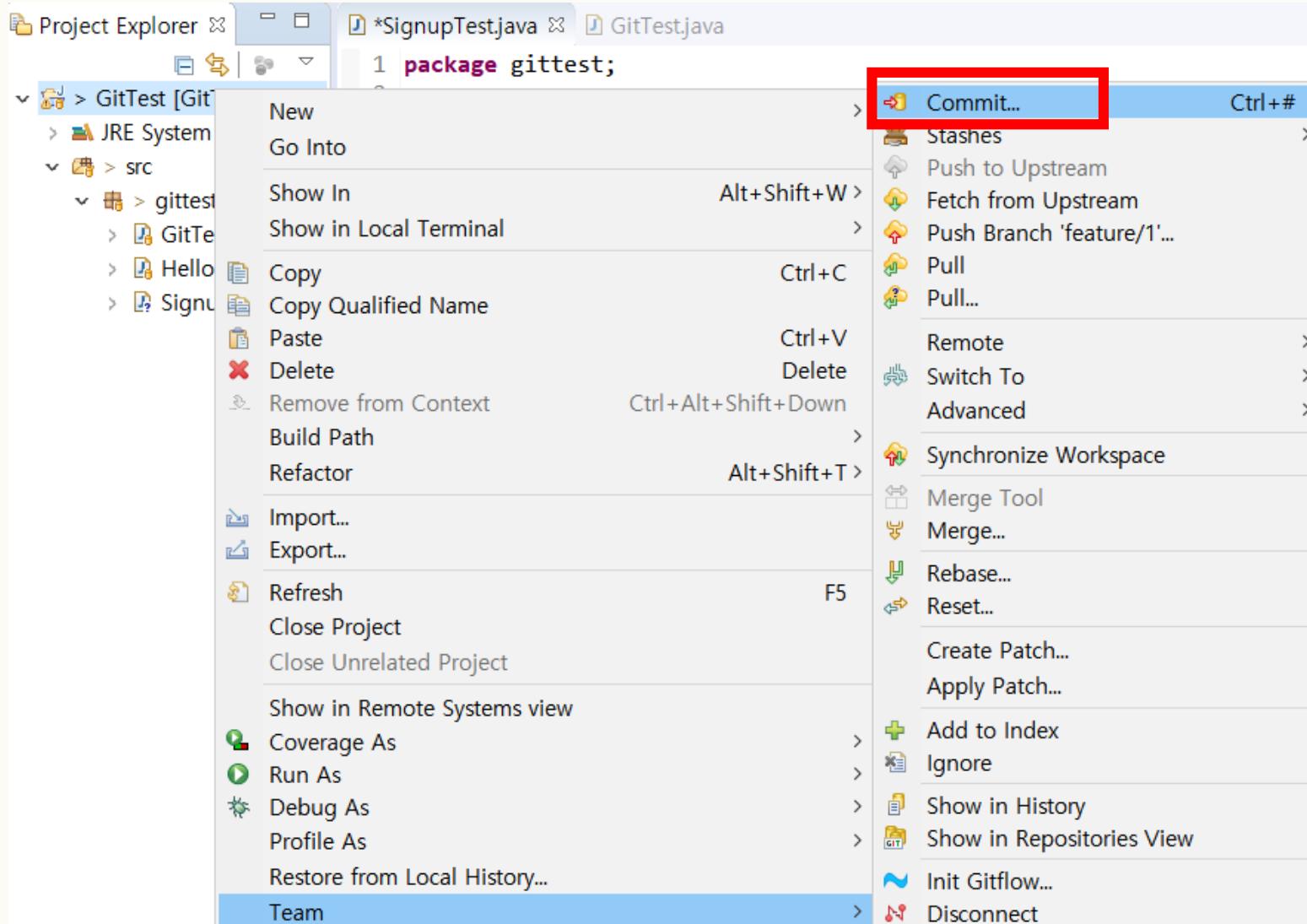
```
1 package gittest;
2
3 public class SignupTest {
4
5     public static void main(String[] args) {
6         System.out.println(
7             "Feature/1 예시를 위한 코드이다. issue에 구현하기로 한 기능인 회원가입을 구현하면 된다."
8             + "develop branch와 동일한 코드지만, 새로 생성한 feature/1 branch에서의 작업이므로 분리된 공간에서의 코드작성이다."
9             + "develop branch로 Merge(병합)하기 전까지는 develop branch에 어떠한 영향을 주지 않고 작업할 수 있다."
10        );
11    }
12
13 }
```

A red box highlights the file 'SignupTest.java' in the Project Explorer. Annotations in the margin of the Editor pane provide context about the code's purpose and its relationship to the 'develop' and 'feature/1' branches.

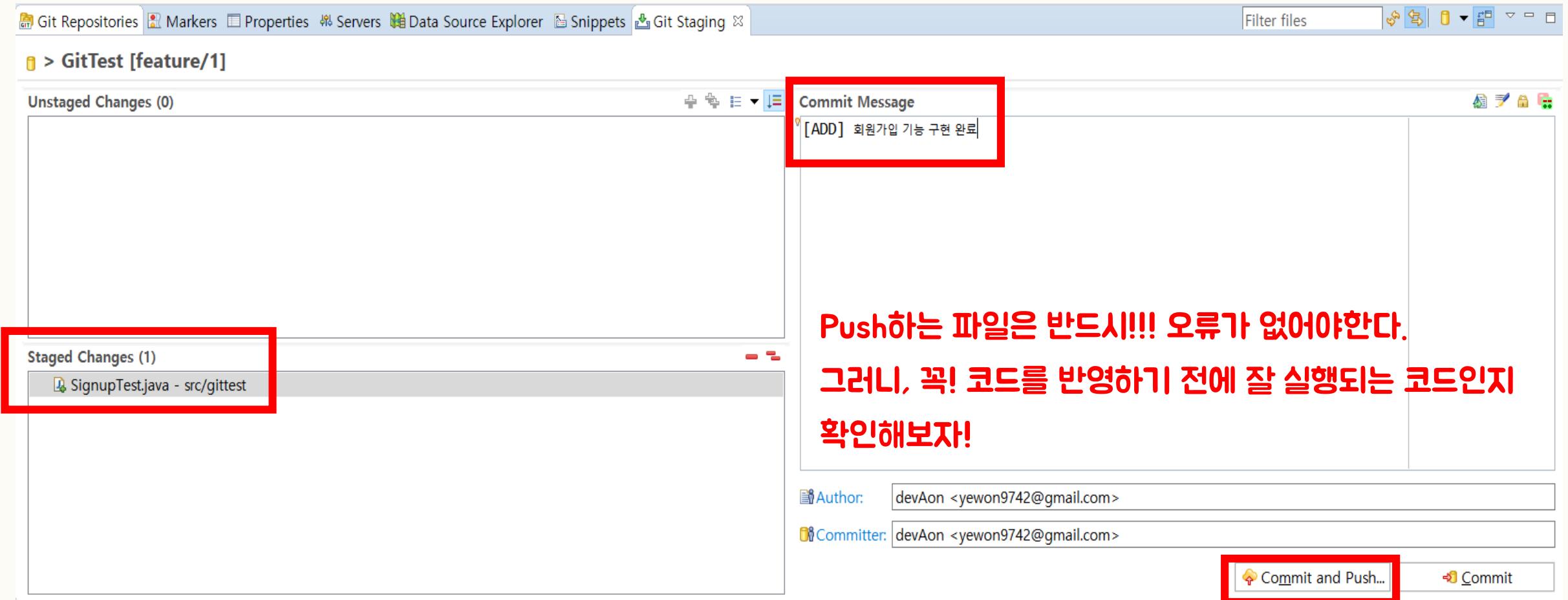
feature branch에서 내가 작업하기로 issue를 작성했던 기능을 구현하면 된다.

feature/1 에서는 회원가입을 구현을 위해 SignupTest.java 파일을 생성했다.

6. 협업 - 3 Commit & Push



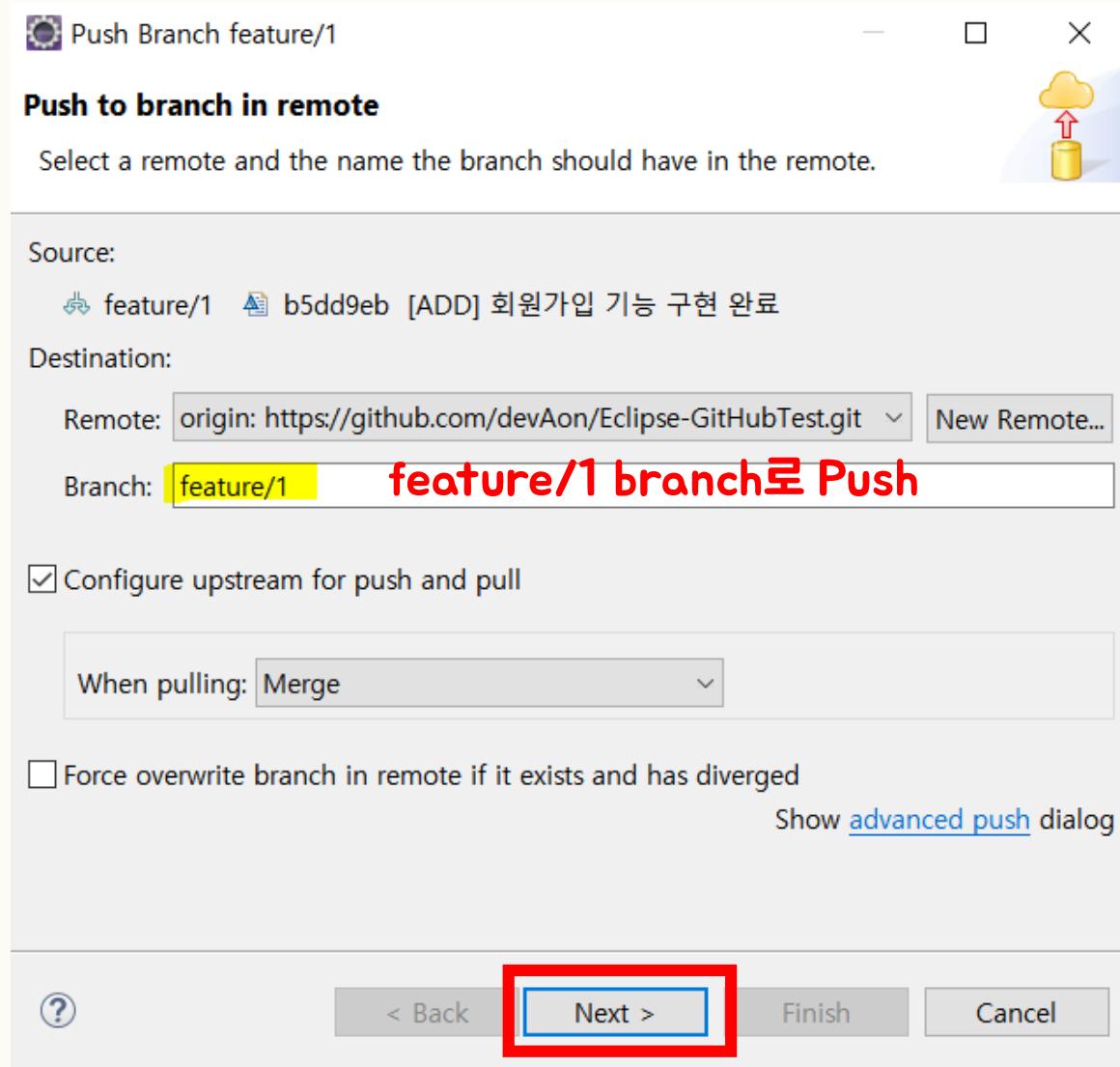
6. 협업 - 3 Commit & Push



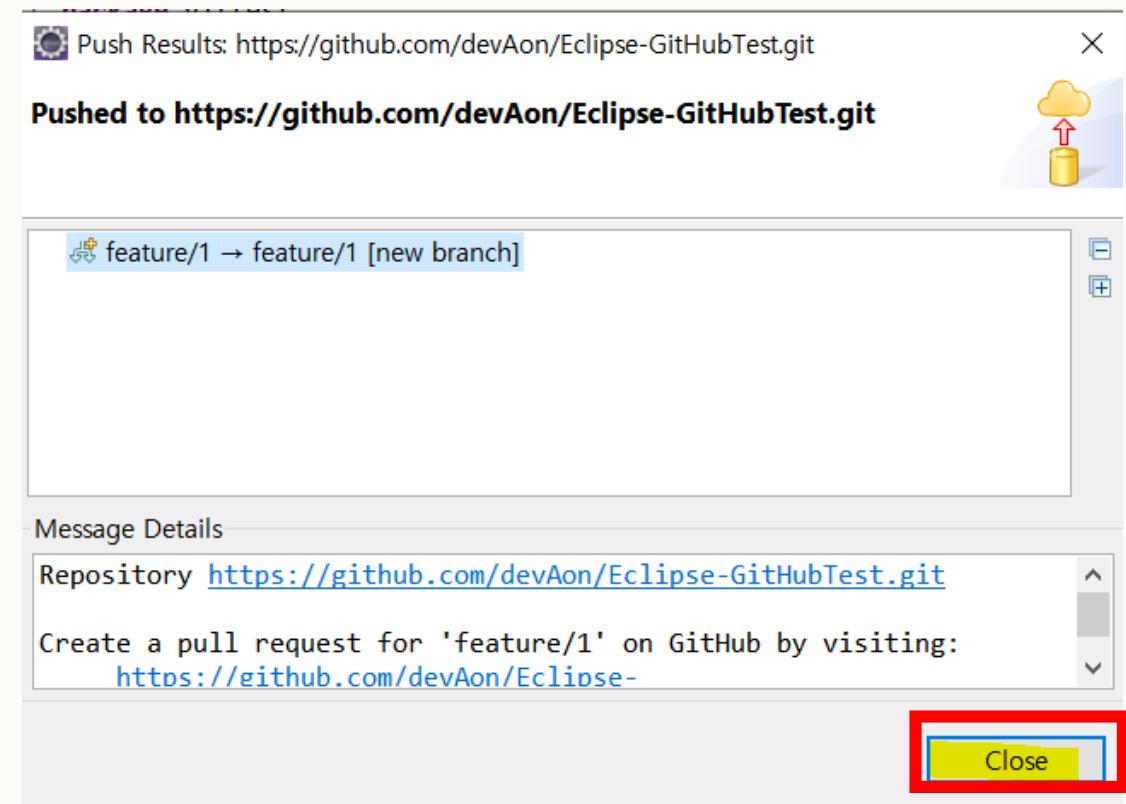
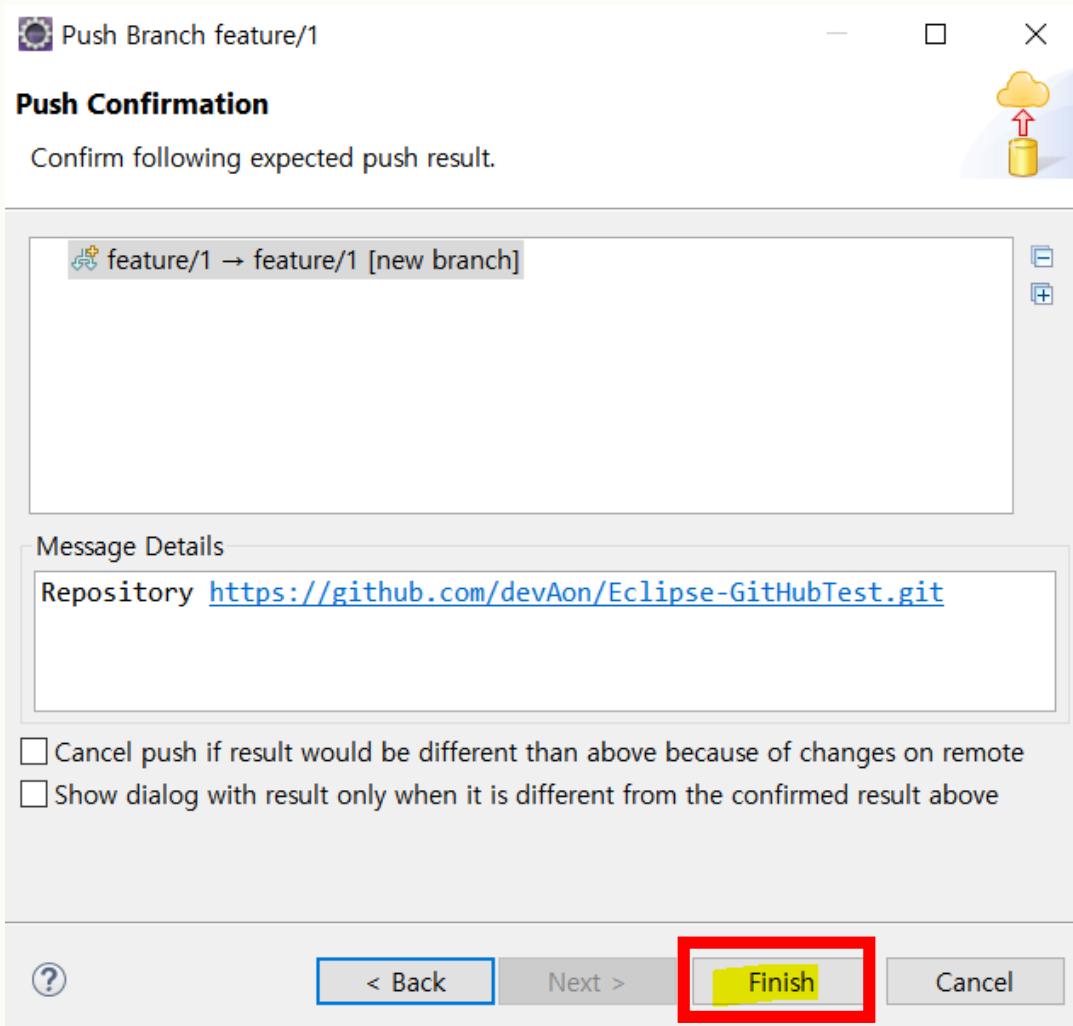
Push하는 파일은 반드시!!! 오류가 없어야한다.
그러니, 꼭! 코드를 반영하기 전에 잘 실행되는 코드인지
확인해보자!

[Unstaged Changes에서 Staged Changes로 파일을 옮긴다]
-[Commit Message]작성-[Commit and Push]

6. 협업 - 3 Commit & Push



6. 협업 - 3 Commit & Push



6. 협업 - 4 Pull Request (=PR)

팀장이 만든 저장소
devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

Code Issues 1 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

이클립스에서의 Git 사용법 - 테스트
Manage topics

2 commits 3 branches

팀장 계정의 ID/PW를 몰라도 된다.
코드를 작성한 본인의 계정 ID/PW로 Pull Request를 날릴 수 있기 때문이다.
즉, 코드를 작성한 본인이 직접 Pull Request를 날리면 된다.

Your recently pushed branches:

feature/1 (1 minute ago)

Branch: master New pull request Compare & pull request

Create new file Upload files Find file Clone or download

| | | |
|--|---|-------------|
|  devAon git test2 | Latest commit 1a1e79e 8 hours ago | |
| .settings | [예시]create project. create hello world. | 8 hours ago |
| src/gittest | git test2 | 8 hours ago |
| .classpath | [예시]create project. create hello world. | 8 hours ago |
| .gitignore | [예시]create project. create hello world. | 8 hours ago |
| .project | [예시]create project. create hello world. | 8 hours ago |



Push 후 깃허브 협업 저장소에 들어오면 Pull request가 뜬 것을 확인할 수 있다. Pull Request를 날리기 위해 클릭!

6. 협업 - 4 Pull Request (=PR)

Pull Request란?

기능 개발을 끝내고 master에 바로 병합(merge)하는 것이 아니라
브랜치를 중앙 원격 저장소에 올리고
master에 병합(merge)해달라고 요청하는 것

Pull Request를 하는 이유?

1. 자연스러운 코드 리뷰를 위해
2. Push 권한이 없는 오픈 소스 프로젝트에 기여할 때
3. Collaborator에 소속되어 있는 경우에는
그 저장소에서 Branch를 따고 Push하면 Pull Request 가능

6. 협업 - 5 코드리뷰 (선택사항)

팀원 각자의 github 계정에서 서로의 코드에 리뷰를 남길 수 있습니다.

방법1) 해당 코드에 직접 댓글을 달 수 있다.

[ADD] 회원가입 기능 구현 완료 #2

Open devAon wants to merge 1 commit into develop from feature/1

Conversation 0 Commits 1 Checks 0 Files changed 1

Changes from all commits ▾ File filter... ▾ Jump to... ▾ 0 / 1 files viewed

src/gittest SignupTest.java

```
@@ -0,0 +1,13 @@
+ package gittest;
+
+ public class SignupTest {
+
+     public static void main(String[] args) {
+         System.out.println(
+             "FeatureOneExample"
+             + "develop branch와 동일한 코드지만, 새로 생성한 feature/1 branch에서의 작업"
+             + "develop branch로 Merge(병합)하기 전까지는 develop branch에 대한 영향을
```

Write Preview

들여쓰기가 조금 이상해요.
위의 예시처럼 코드관련하여 글을 달 수 있다.

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

Cancel Add single comment Start a review

방법2) 코드확인했다 등 댓글을 달 수 있다.

Add more commits by pushing to the feature/1 branch on devAon/Eclipse-GitHubTest.

Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

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.

Write Preview

확인했습니다.

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

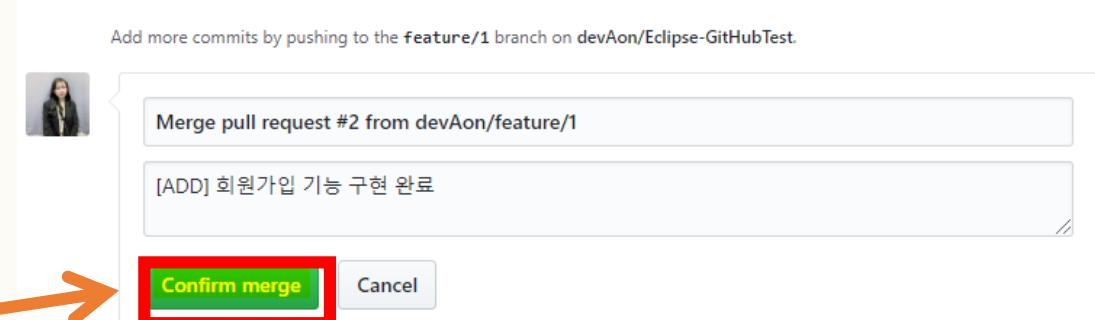
Close and comment Comment

6. 협업 - 6 Merge

팀장이 만든 저장소

The screenshot shows a GitHub pull request page for a repository named 'devAon / Eclipse-GitHubTest'. The pull request is titled '[ADD] 회원가입 기능 구현 완료 #2' and is from the 'feature/1' branch into the 'develop' branch. The status is 'Open' with 1 commit. The conversation tab shows a comment from 'devAon' 1 minute ago: '회원가입 기능 구현에 관련하여 팀원이 알아볼 수 있는 말을 남긴다. 선택사항이지만, 구체적으로 설명할 부분이 있거나 해당 코드에 대해 팀원들에게 원활하게 이해하는 것을 돋기 위해 남기는 것이 좋다.' Below the comment, there is a self-assigned note from 'devAon' 1 minute ago. A message at the bottom of the page says 'Add more commits by pushing to the feature/1 branch on devAon/Eclipse-GitHubTest.' On the left, there is a sidebar with a green button labeled 'Merge pull request' which is highlighted with a red box.

팀장의 저장소이지만
5-3 Manage access에서 팀원에게 협업권한을 주었기 때문에
팀장뿐만 아니라 팀원도 직접 merge 할 수 있다.
merge 방법은 다양하다.
Ex1) 모든 팀원에게 코드리뷰를 받고 merge 해도 되고,
Ex2) 팀장에게 확인을 거쳐 팀장이 모든 팀원의 코드를 merge 해도 되고,
Ex3) 코드를 작성한 팀원을 믿고 작성한 팀원이 직접 merge해도 된다.
이처럼 팀원끼리 merge 규칙을 정하면 된다.



6. 협업 - 6 Merge

[ADD] 회원가입 기능 구현 완료 #2

Merged devAon merged 1 commit into develop from feature/1 12 seconds ago

Conversation 1 Commits 1 Checks 0 Files changed 1

devAon commented 15 minutes ago

회원가입 기능 구현에 관련하여 팀원이 알아볼 수 있는 말을 남긴다.
선택사항이지만, 구체적으로 설명할 부분이 있거나 해당 코드에 대해 팀원들에게 원활하게 이해하는 것을 돋기 위해 남기는 것이 좋다.

Owner + 😊 ...

devAon [ADD] 회원가입 기능 구현 완료 b5dd9eb

devAon self-assigned this 15 minutes ago

devAon reviewed 7 minutes ago

View changes

src/gittest/SignupTest.java Show resolved

devAon merged commit 68109f3 into develop 12 seconds ago

Revert

Pull request successfully merged and closed

You're all set—the feature/1 branch can be safely deleted.

Delete branch

Merge 완료

6. 협업 - 6 Merge

팀장이 만든 저장소

devAon / Eclipse-GitHubTest

Unwatch 1 Star 0 Fork 0

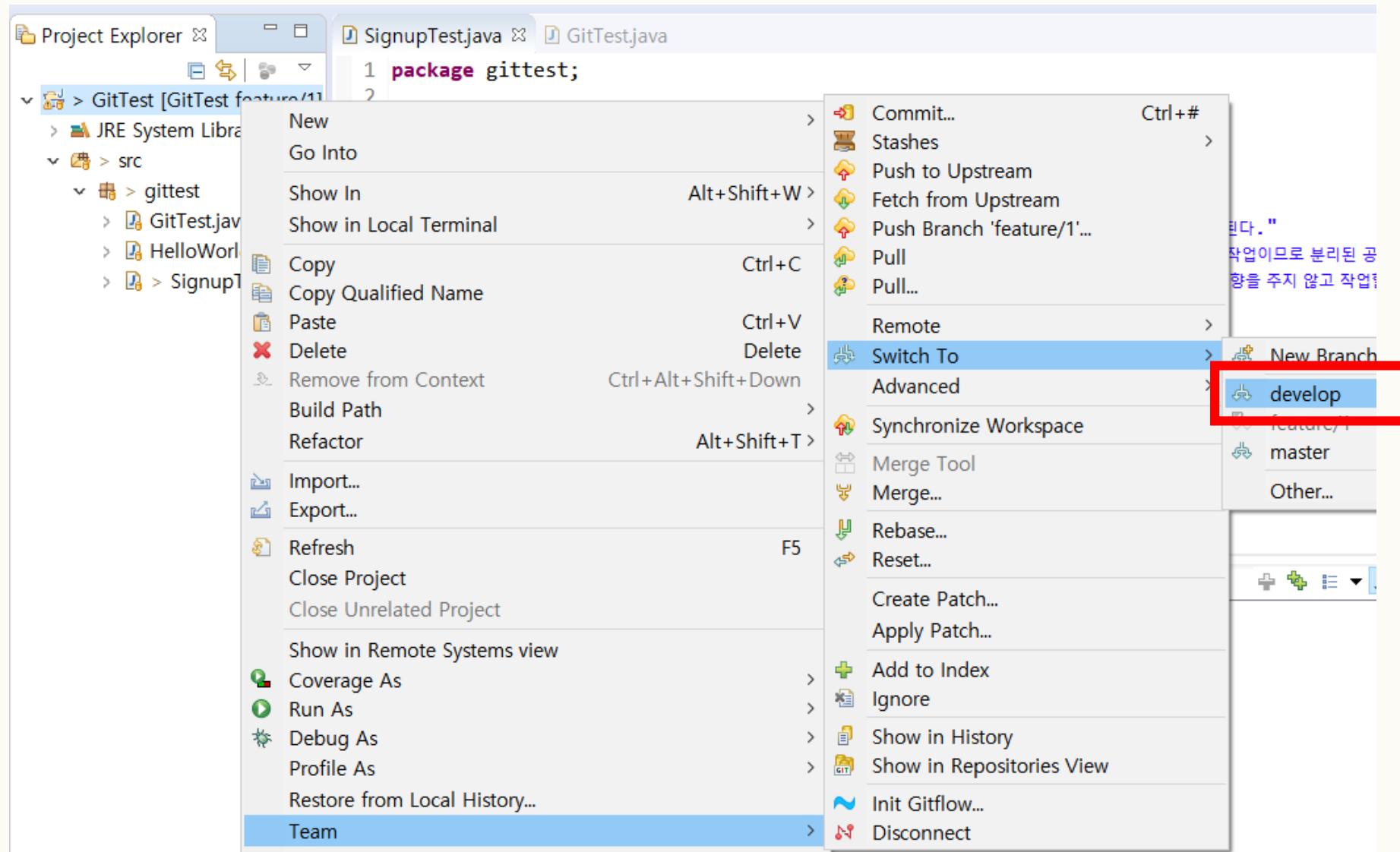
The screenshot shows a GitHub repository page for 'Eclipse-GitHubTest'. The 'Code' tab is selected. The 'Branch: develop' dropdown is highlighted with a yellow box. The repository path 'Eclipse-GitHubTest / src / gittest /' is shown. The main area displays a list of commits:

| Commit | Message | Time |
|----------------------------|---|----------------|
| devAon [ADD] 회원가입 기능 구현 완료 | Latest commit b5dd9eb | 34 minutes ago |
| .. | | |
| GitTest.java | [예시]create project. create hello world. | 9 hours ago |
| HelloWorld.java | [예시] | 6 hours ago |
| SignupTest.java | [ADD] 회원가입 기능 구현 완료 | 34 minutes ago |

GitHub 저장소에 들어가 develop branch로 바꿔 확인해보면

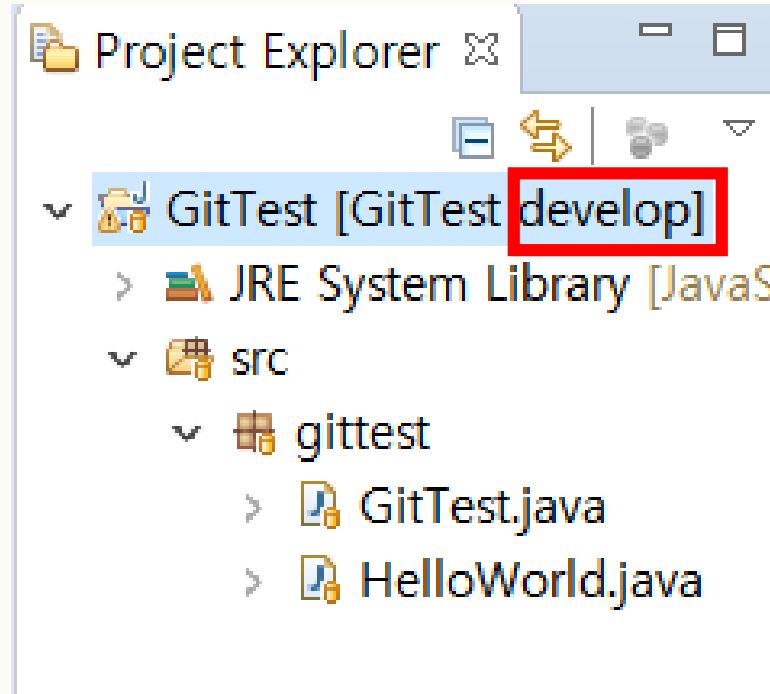
feature/1 branch에서 작업한 내용이 develop branch에 합쳐진 것을 확인할 수 있을 것이다.

6. 협업 - 7 Checkout branch develop



6. 협업 - 7 Checkout branch develop

checkout 명령어?
branch 이동



develop branch로 이동
이유?
중앙 원격 저장소와
자신의 로컬 저장소를
동기화하기 위해

Github 원격 저장소 develop branch에는 feature/1 branch에서 작업한 SignupTest.java 파일이 병합(merge)됐지만, local의 develop branch에는 반영되어 있지 않음을 확인할 수 있다.
Pull 명령어를 통해 local develop branch에도 반영시켜보자!

6. 협업 - 8 Pull

Pull 명령어?

원격 저장소 내용을 로컬 저장소에 반영

[Git bash 이용시]

git Pull [원격 저장소 이름] [원격 저장소 브랜치 이름]

ex) git pull origin master

ex) git pull origin develop

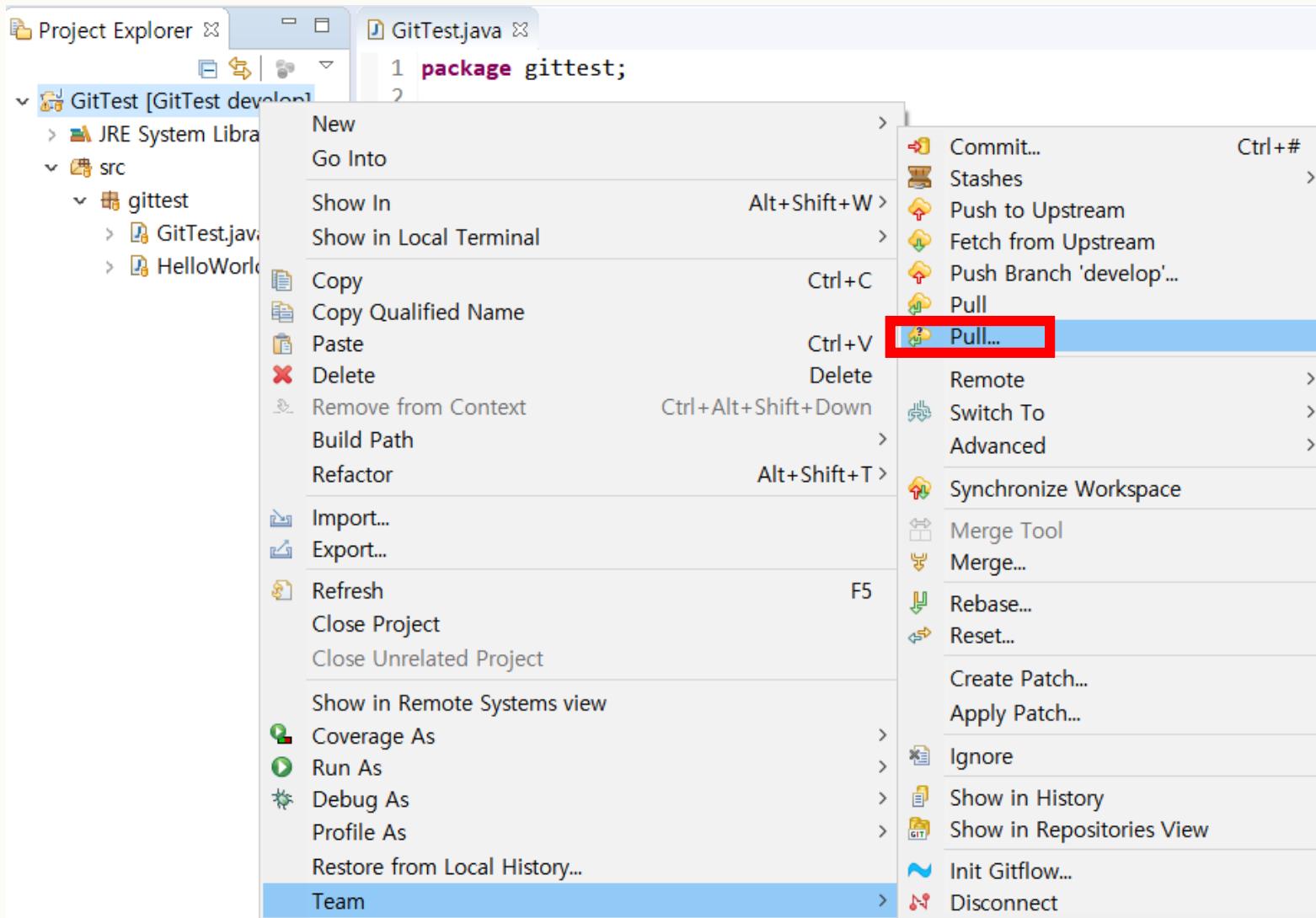
다른 작업자가 수정한 사항을 자신의 로컬 저장소에 반영

새로운 issue 및 feature branch를 따기 전

꼭!!! develop branch에 Pull 받기!!!

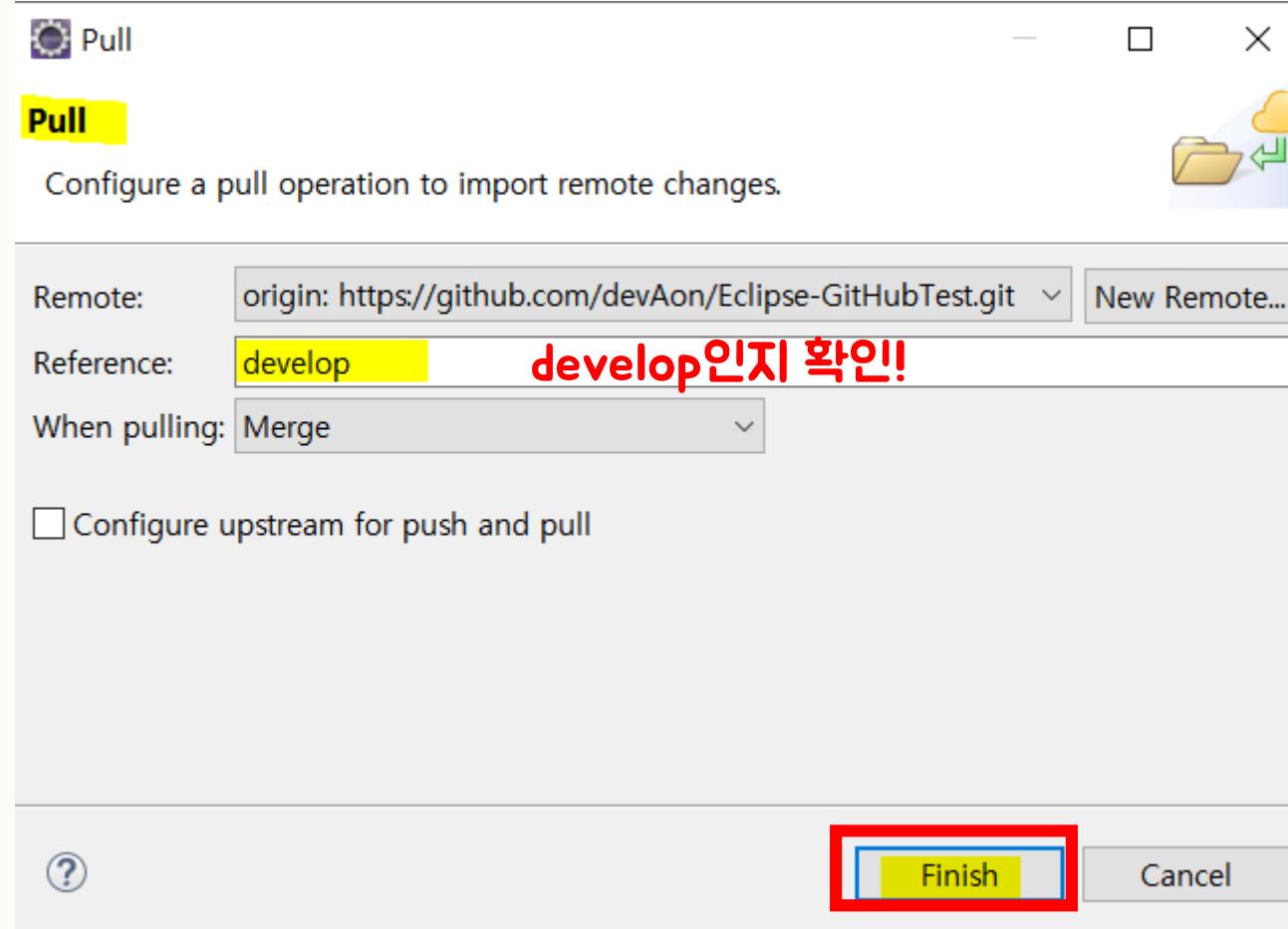
그래야 충돌 위험을 줄일 수 있다.

6. 협업 - 8 Pull

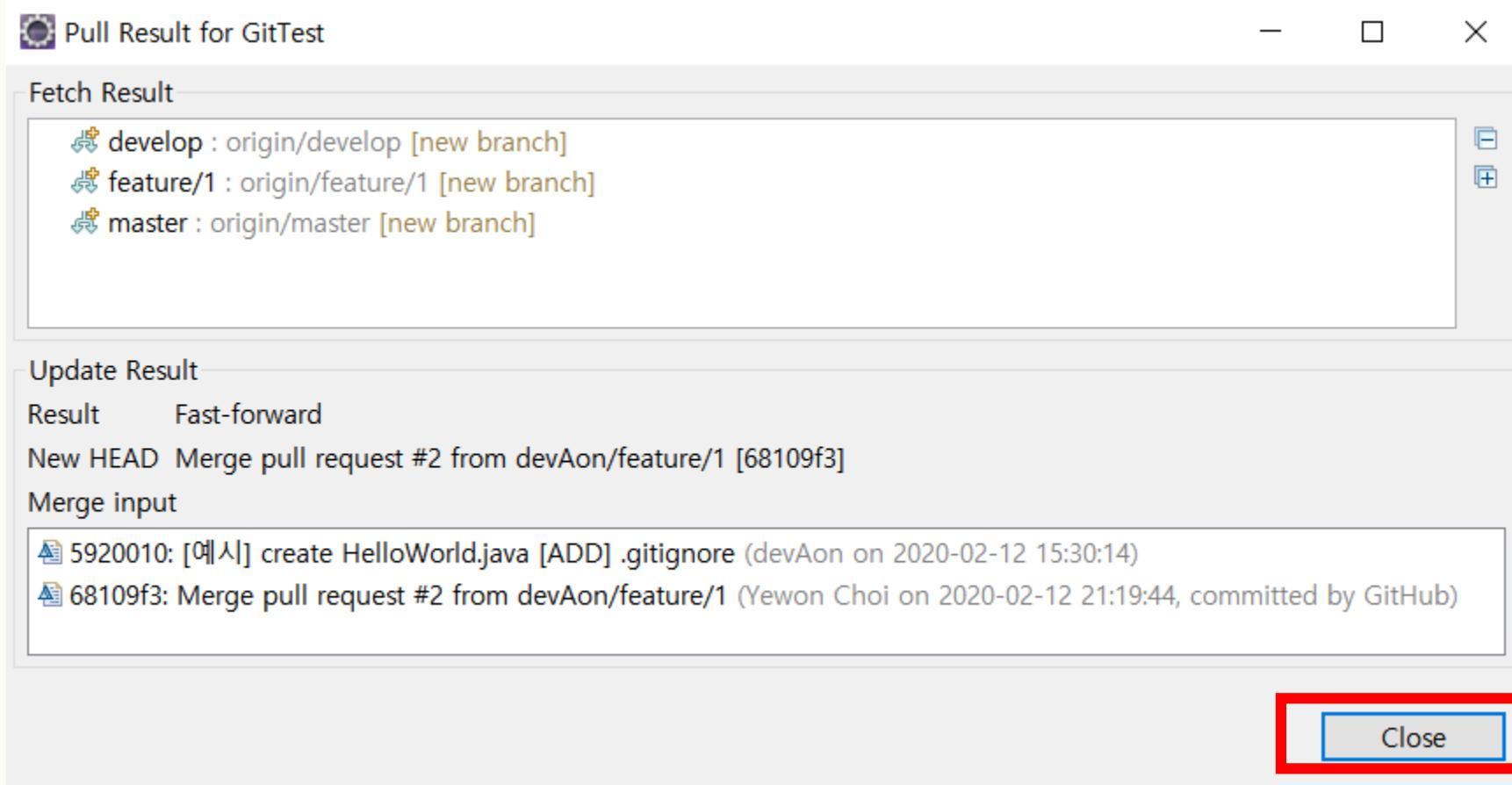


생성한 프로젝트에서 마우스 우클릭 - [Team]-[Pull..]

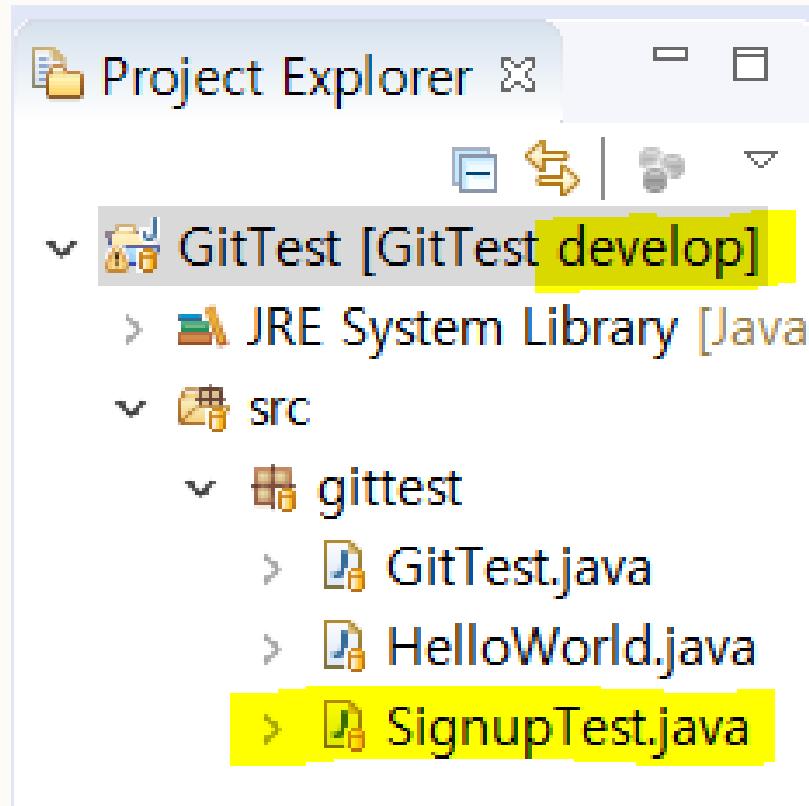
6. 협업 - 8 Pull



6. 협업 - 8 Pull

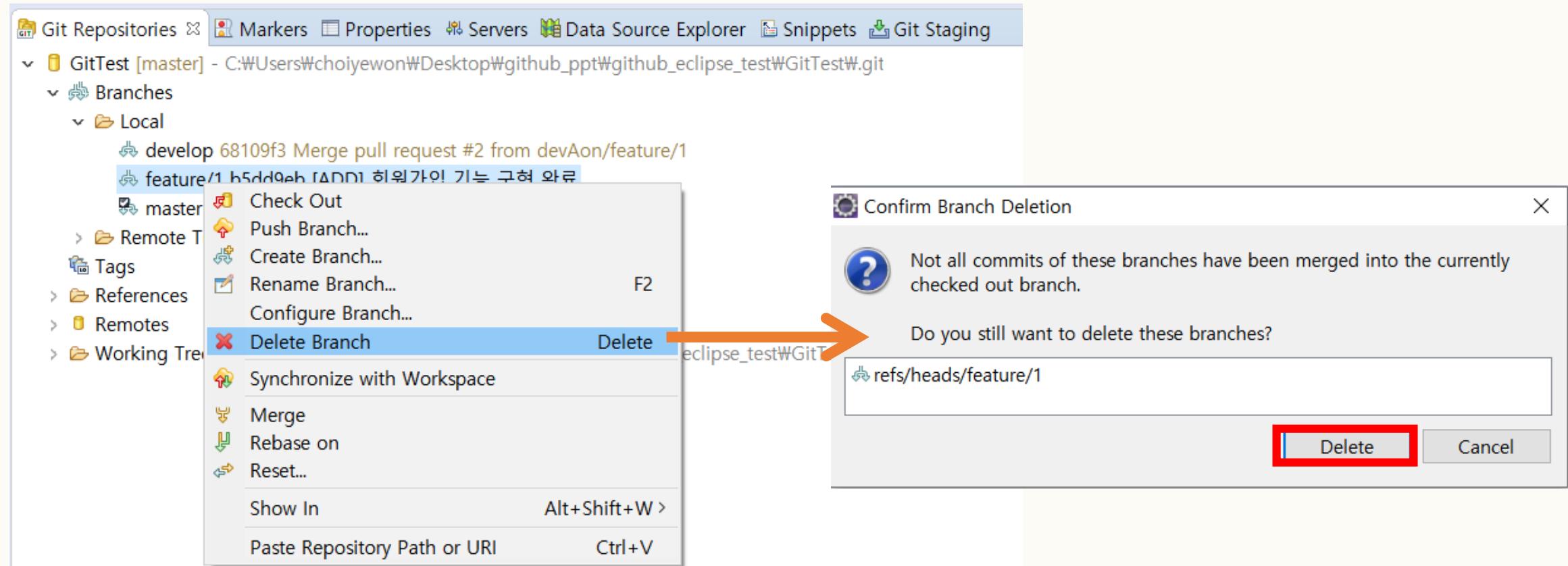


6. 협업 - 8 Pull



develop branch에 feature/1 branch에서 작업했던
SignupTest.java 파일이 pull 받아졌다.

6. 협업 - 9 delete feature branch



6. 협업 - 10 정리

master-develop-feature 방식으로 협업하는 방법을 설명했다.

순서 정리

1. 구현할 기능 Issue작성
2. Issue번호로 'feature/{Issue번호}'로 new branch 생성 및 작업
3. 기능 구현이 다 끝났고 오류가 없음을 확인했다면, Commit & push
4. 팀장의 GitHub 원격 저장소 홈페이지에 들어가 Pull Request
(이때 반드시!!!! **base:develop** 으로 변경해줘야 한다. 반드시!!!!!!)
5. 서로의 코드를 리뷰해줄 수 있다. (선택사항)
6. 코드에 오류가 없고 합쳐도 된다고 생각하면 Merge(병합)한다.
7. checkout 명령어를 사용해 develop 으로 이동시킨다. (local develop에도 변경된 코드 반영 목적)
8. 여러 팀원들의 코드가 반영된 develop branch 의 코드를 pull 받는다.
9. 구현이 끝난 'feature/{Issue번호}'는 삭제한다.

또 다른 기능을 구현하려 한다면, 다시 **1번 - 9번을 반복** 하면 된다. (자세한 설명은 다음 페이지에서)

6. 협업 - 10 정리 새로운 기능 구현하려면? 다시 6-1번~9번을 수행하면 된다!!

팀장이 만든 저장소

devAon / Eclipse-GitHubTest

The screenshot shows a GitHub repository named 'Eclipse-GitHubTest'. On the left, there's an 'Issues' tab with 2 issues. A red box highlights the number '3' next to an issue. On the right, there's a detailed view of issue #3. A red box highlights the '#3' in the URL. Below the issue view, there's a comment from 'devAon' and a 'Create Branch' dialog box. The dialog box has a red box around the 'Branch name:' field containing 'feature/3'. A large orange arrow points from the GitHub issue section to the Eclipse Git Staging interface.

① 6-1 issue작성

② 6-2 브랜치 생성

issue번호

로그인 구현 #3

Open devAon opened this issue now · 0 comments

devAon commented now

No description provided.

Create Branch

Create a new branch

Please choose a source branch and a name for the new branch

Source: master Select...

Branch name: feature/3

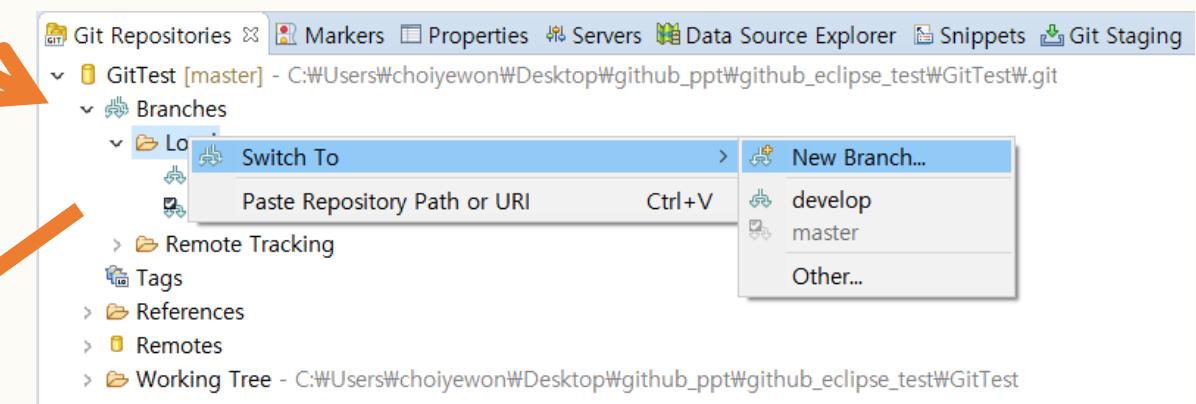
Configure upstream for push and pull

When pulling:

Check out new branch

Finish Cancel

branch 이름 : feature/{issue번호}



6. 협업 - 10 정리

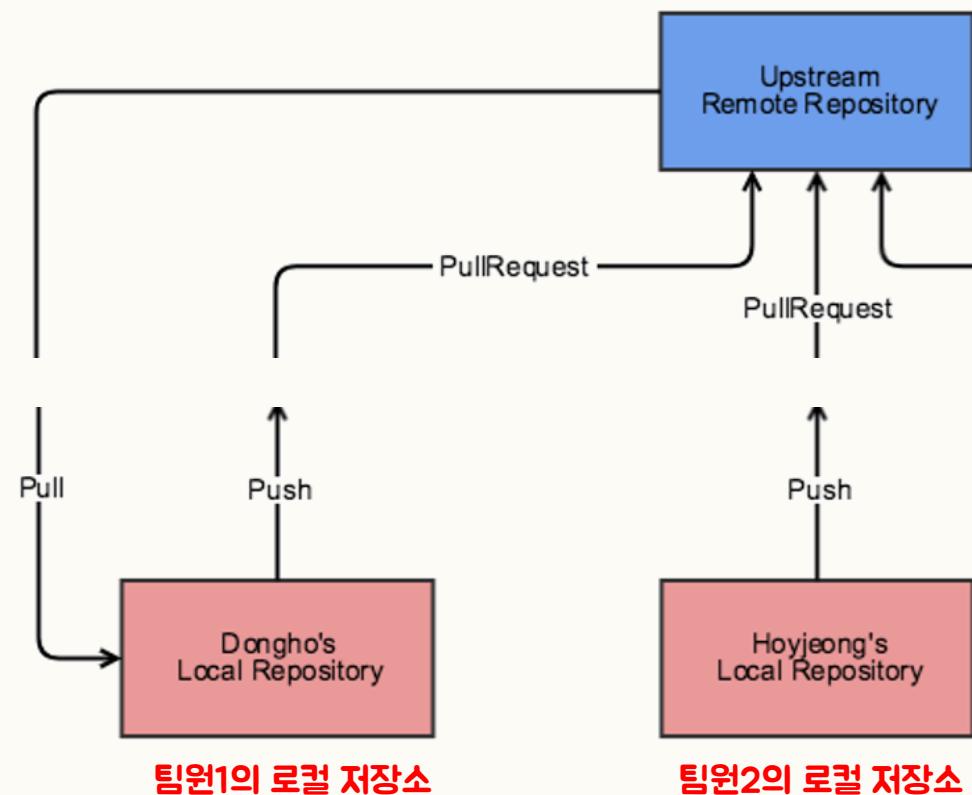
이렇게 협업을 한다면, 팀원들은 서로 다른 branch에서 작업을 하지만
최종적으로는 develop branch에 팀원 모두의 코드가 합쳐져 있을 것이다.

프로젝트 제출 시, develop branch에 오류가 없고 정상적으로 코드가 실행된다면
팀장은 develop branch를 master branch로 merge(합쳐)
master branch 파일을 제출하면 된다.

6. 협업 - 10 정리

지금까지 설명한 협업방식은
Upstream Repository - Local Repository만 사용하여
직접적으로 Pull Request를 날리는 방식이다.
즉, Origin Repository를 뺀 협업방식이다.

팀장의 저장소에 협업팀원에게 권한을 주어 fork하지 않아도 직접적인 권한을 갖기에
팀원의 local 저장소에서 팀장의 저장소로 직접적인 접근이 가능하다.



Upstream Repository?

개발자들이 공유하는 저장소로
최신 소스코드가 저장되어 있는 원격 저장소

~~Origin Repository?~~
~~(fork하면 존재하는 저장소)~~
~~Upstream Repository를 Fork한
원격 개인 저장소~~

Local Repository?

내 컴퓨터에 저장되어 있는 개인 저장소

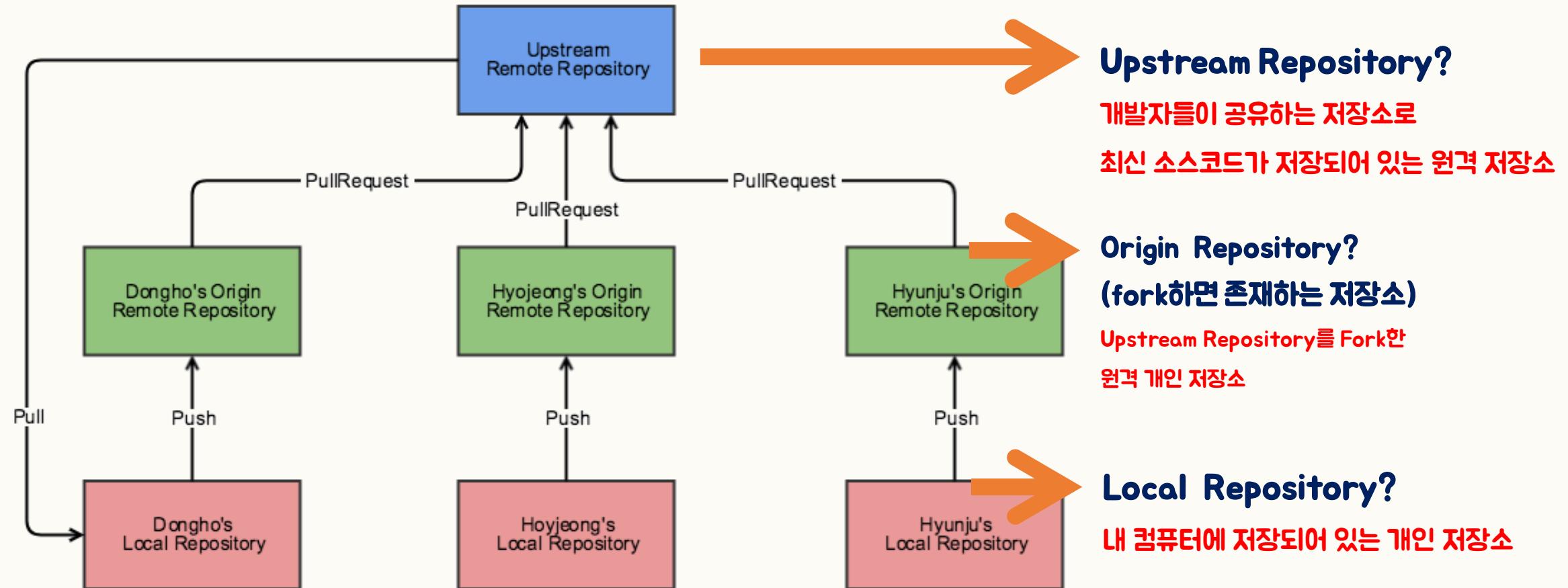
6. 협업 - 10 정리

[GitHub] GitHub로 협업하는 방법[2] - Forking Workflow

<https://gmlwj9405.github.io/2017/10/28/how-to-collaborate-on-GitHub-2.html>

[GitHub] GitHub로 협업하는 방법[3] - Gitflow Workflow

<https://gmlwj9405.github.io/2018/05/12/how-to-collaborate-on-GitHub-3.html>



6. 협업 - 10 정리

master-develop-feature 방식이 정답은 아니다.

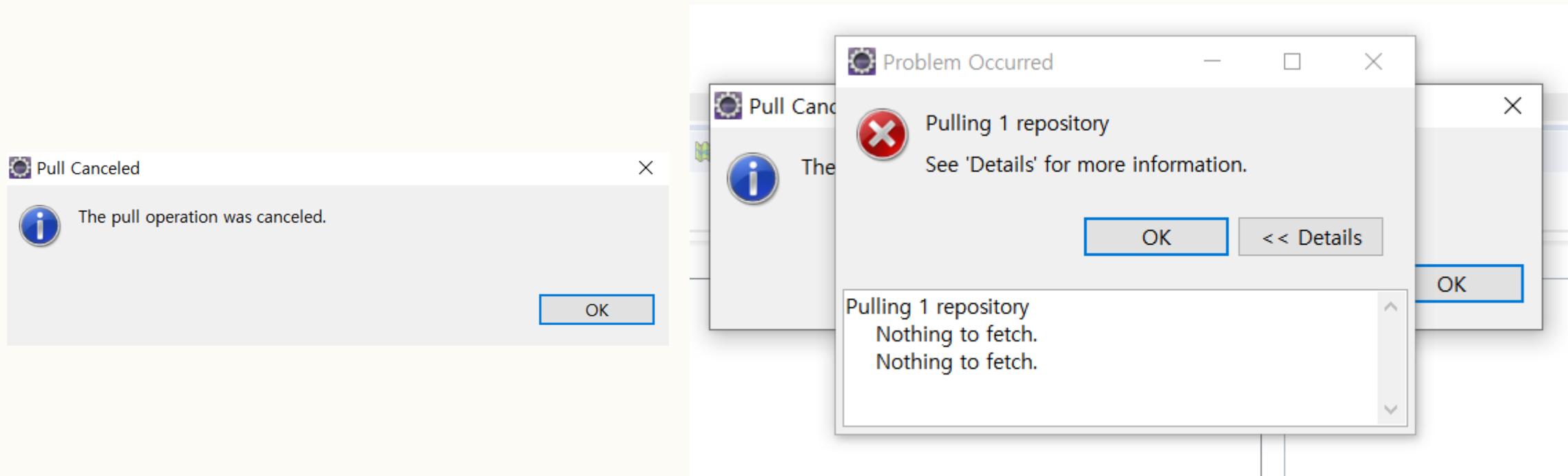
다른 방법도 존재하니 구글에 검색해

팀에 맞는 프로젝트 방식을 적용하면 된다.

Git PRESENTATION

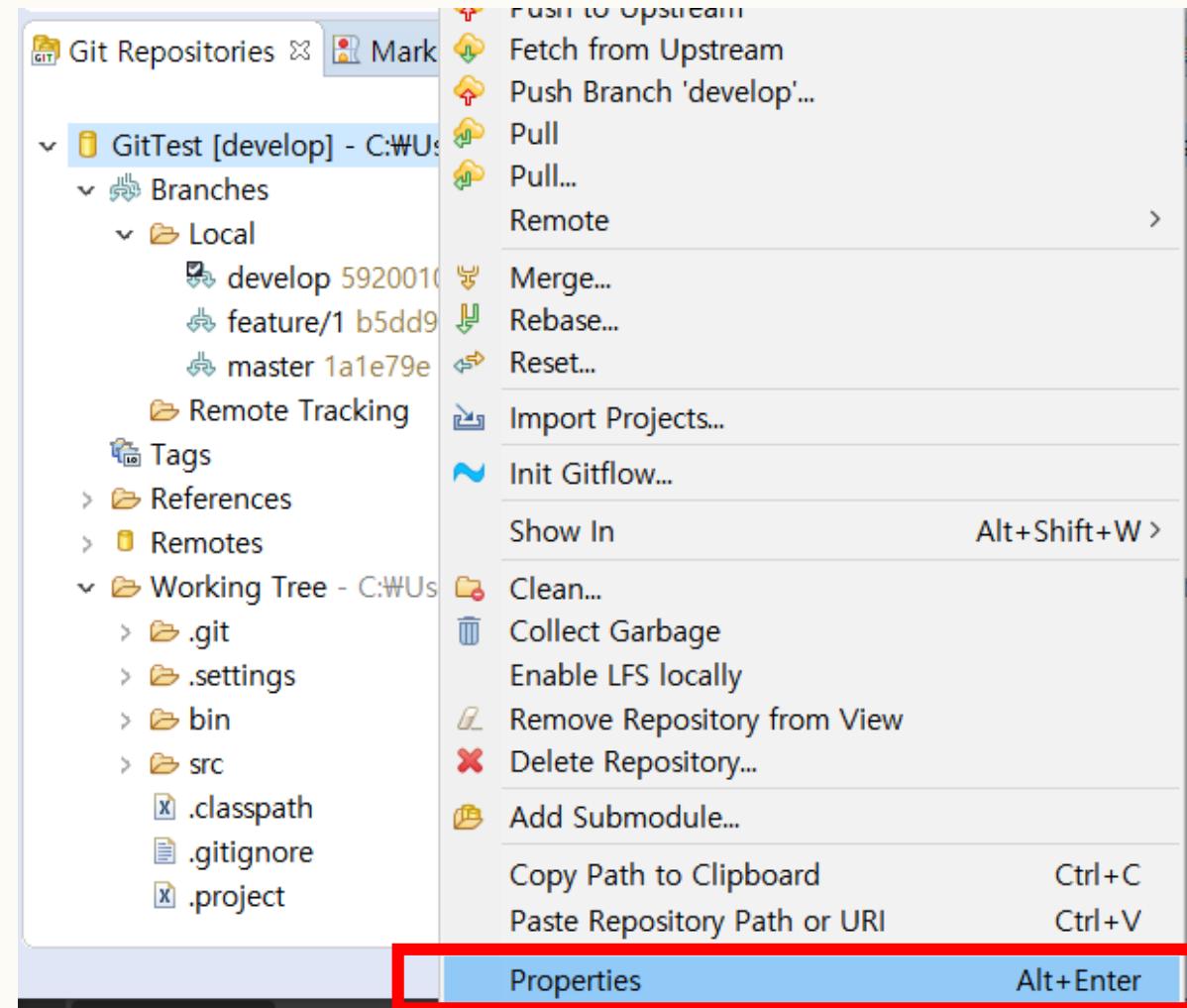
07 주의사항

7.1 Pull 문제 발생



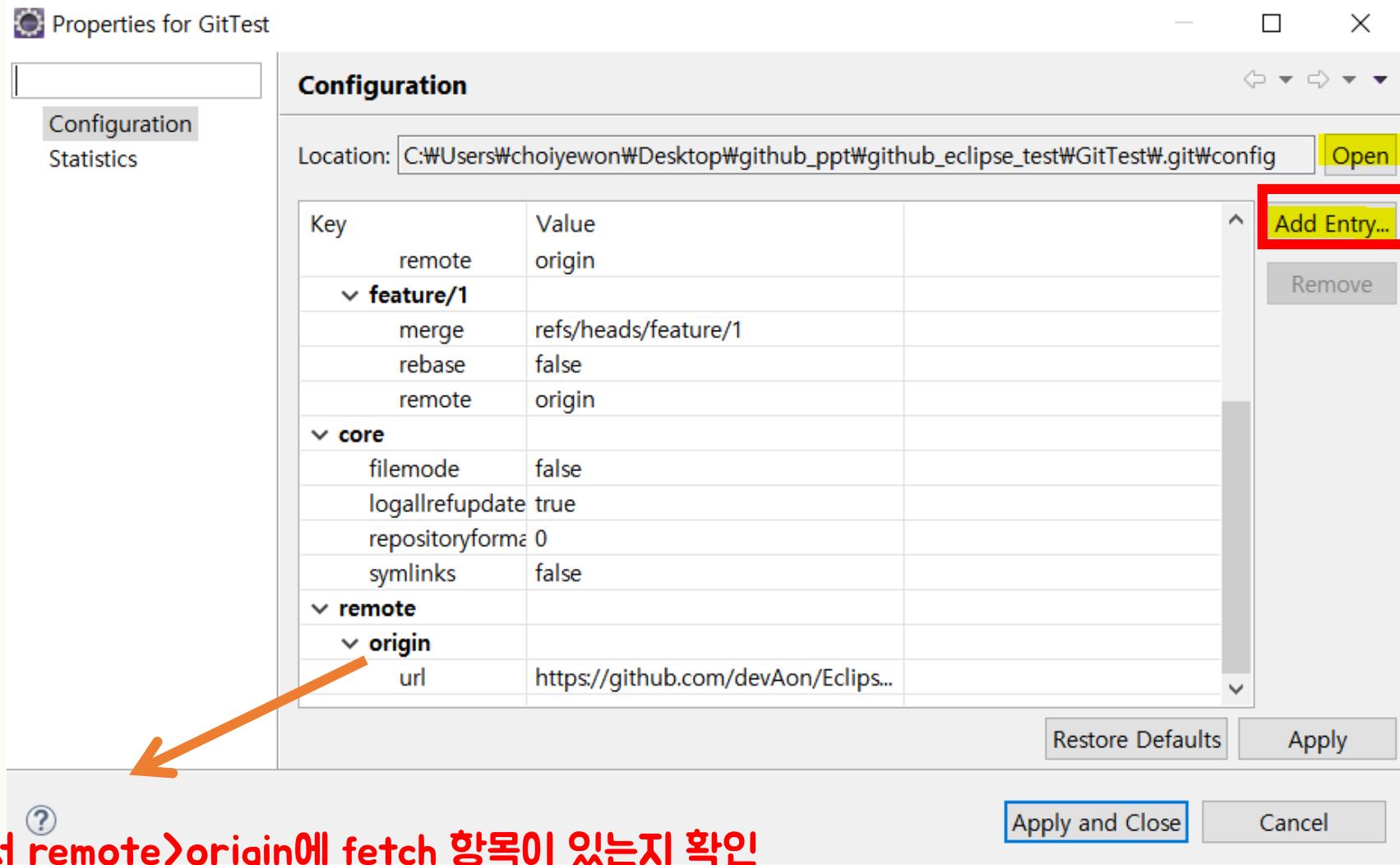
해당 프로젝트의 Remote에 fetch가 잡혀 있지 않아 발생하는 오류

7.1 Pull 오류 해결

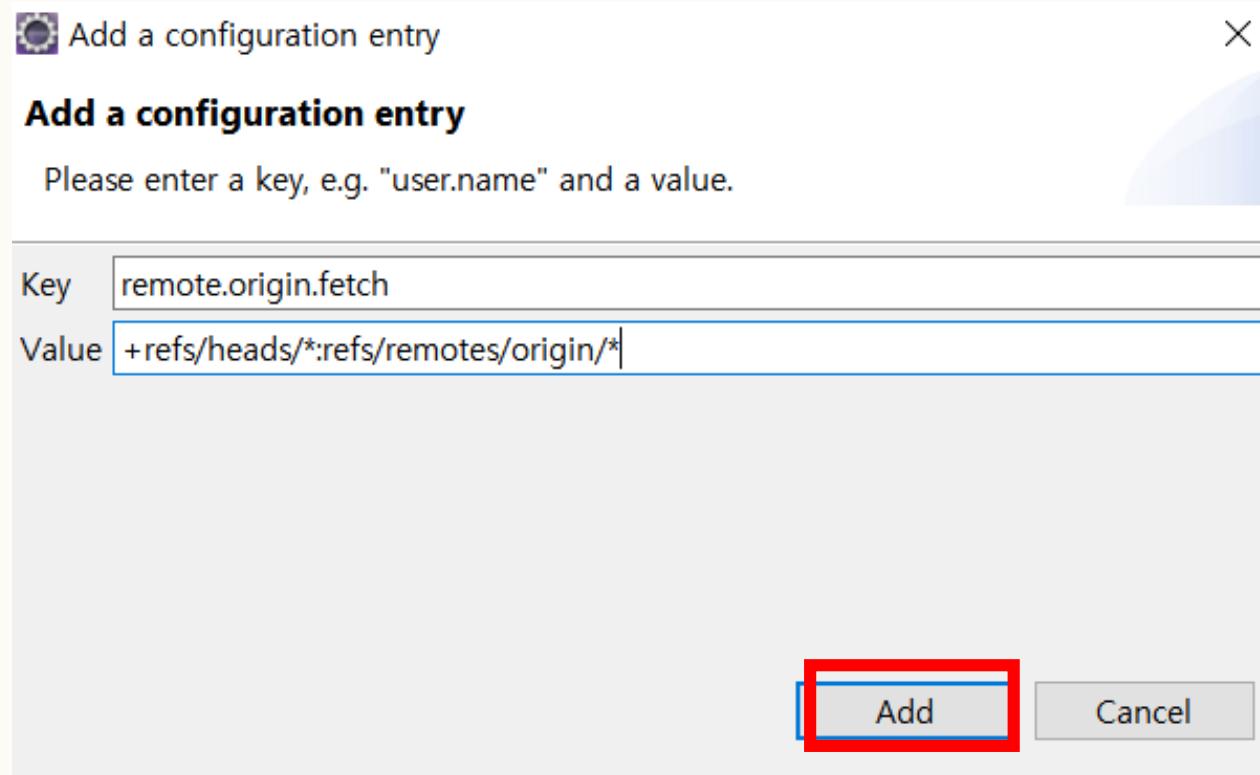


[Git Repositories]-[Properties]

7.1 Pull 오류 해결



7.1 Pull 오류 해결

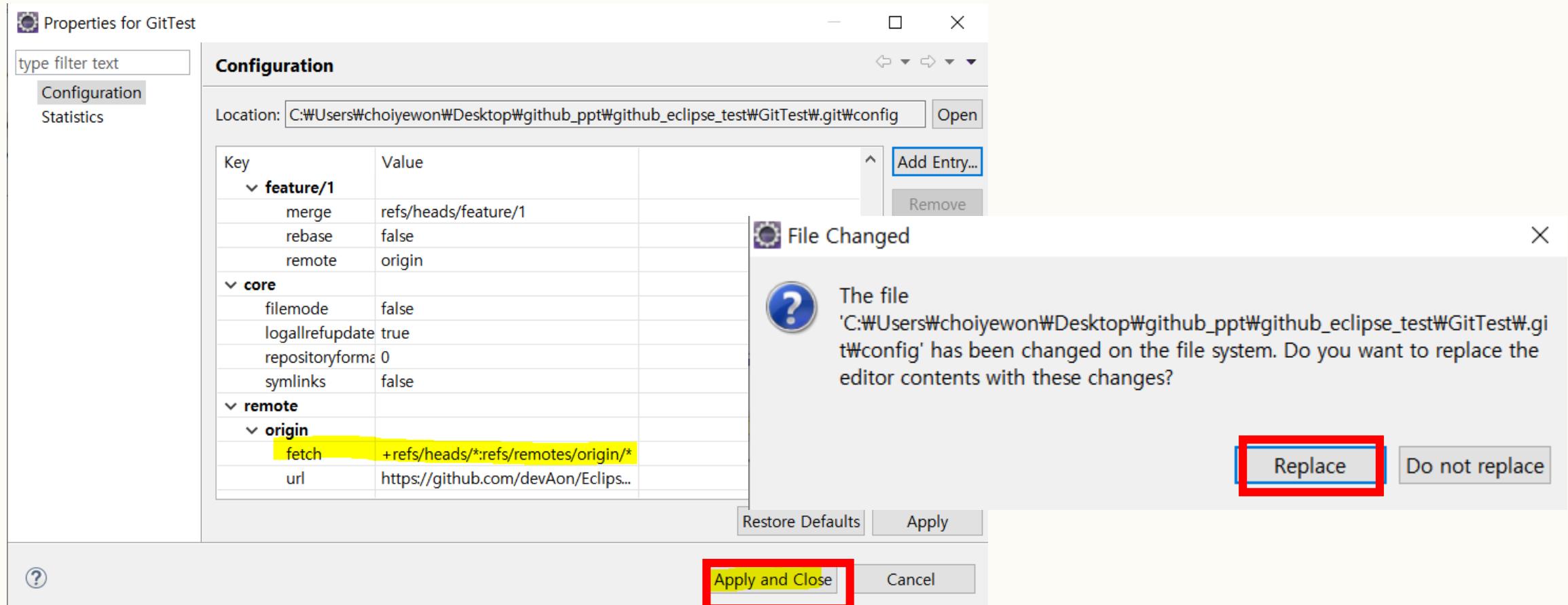


Add Entry ... 를 클릭

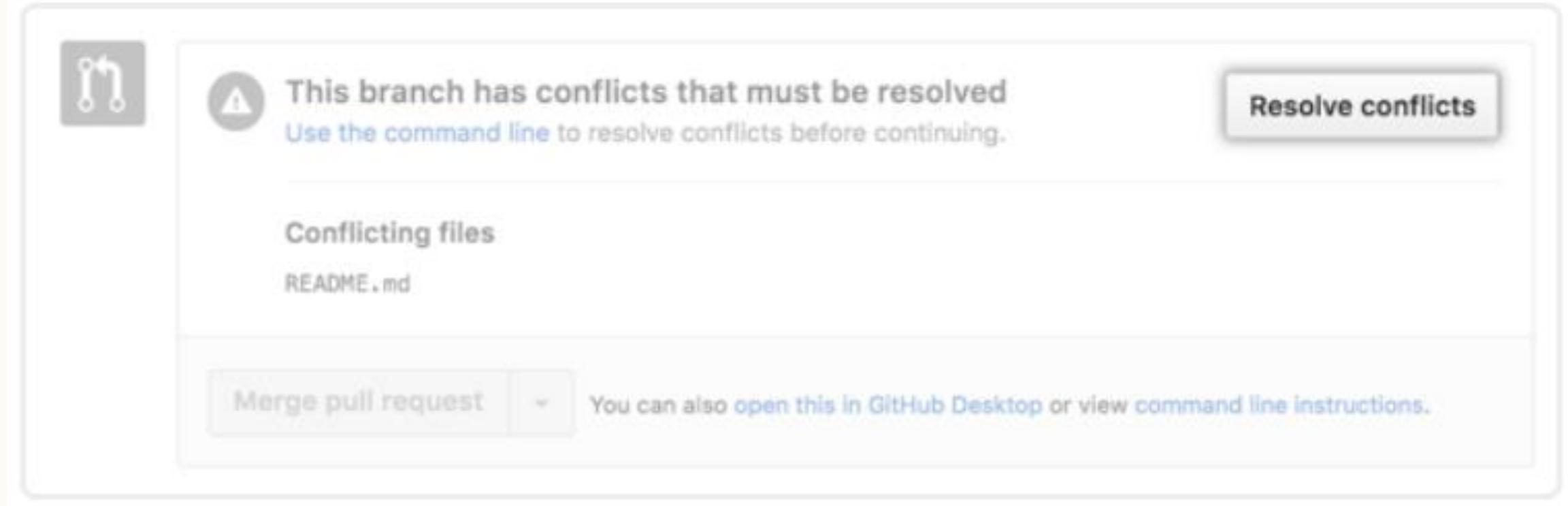
Key: remote.origin.fetch

Value: +refs/heads/*:refs/remotes/origin/*

7.1 Pull 오류 해결



7.2 충돌해결



merge를 하는 과정에서 충돌이 발생할 수 있다.
그러나, Resolve conflicts를 통해 해결할 수 있으니 걱정하지 말자.

7.2 충돌해결

Resolving conflicts between add-emoji and master and committing changes ➔ add-emoji

| 1 conflicting file | README.md | 1 conflict | Prev ⌂ | Next ⌂ | ⚙️ | Mark as resolved |
|--|---|------------|--------|--------|----|------------------|
|  README.md README.md | <pre>1 Octo-Repo 2 ===== 3 4 Contact @octo-org/core for questions about this repository. 5 6 ### Installing 7 8 Instructions for Installing! 9 10 ### Contributing 11 12 See the [CONTRIBUTING file](.../CONTRIBUTING) for guidance on contributing to this project. 13 14 ### Authors 15 16 <<<<< add-emoji 17 [We're all authors :star: :zap: :sparkles: 18 ===== 19 We're all authors :star: :zap: :heart: 20 >>>> master 21 22 This commit will be verified! 23</pre> | 1 conflict | Prev ⌂ | Next ⌂ | ⚙️ | Mark as resolved |

충돌 마커 <<<<<, =====, >>>>> 를 삭제
최종 병합에 원하는 코드 하나만 남긴다.

7.2 충돌해결

The screenshot shows a GitHub merge conflict resolution interface. At the top, it says "Resolving conflicts between add-emoji and master and committing changes → add-emoji". Below this, it indicates "1 conflicting file" and lists "README.md". The code editor shows the following content:

```
1 Octo-Repo
2 ======
3
4 Contact @octo-org/core for questions about this repository.
```

On the right side of the interface, there are buttons for "1 conflict", "Prev ⌂", "Next ⌂", "⚙️", and "Mark as resolved". An orange arrow points from the text below to the "Mark as resolved" button.

충돌 마커 <<<<<, =====, >>>>> 를 삭제하면 [Mark as resolved] 버튼이 활성화된다.

The screenshot shows the same GitHub merge conflict resolution interface, but now all conflicts have been resolved. The "Mark as resolved" button has turned green and is labeled "Commit merge". A checkmark icon is visible next to the "README.md" file name. An orange arrow points from the text below to the "Commit merge" button.

충돌이 발생한 모든 파일을 [Mark as resolved] 하면 [Commit merge]버튼이 활성화 된다.
버튼을 클릭하면 충돌해결 완료.

References

✓ Tutorial

<https://backlog.com/git-tutorial/kr/>

✓ GitFlow

<https://woowabros.github.io/experience/2017/10/30/baemin-mobile-git-branch-strategy.html>

<https://gmlwjd9405.github.io/2018/05/12/how-to-collaborate-on-GitHub-3.html>

✓ Pull Request

<https://www.youtube.com/watch?v=xlydJ53nnqY>

✓ 도서

- 소셜 코딩으로 이끄는 GitHub 실천 기술 :Git과 GitHub를 직접 따라하며 배운다 /오오츠카 히로키 지음 ;윤인성 옮김
- (만들면서 배우는) Git+GitHub 입문 /윤웅식 지음.