오픈소스 소프트웨어 개발 - 16

(Online) Git & Github - centralized

광운대학교 이윤상
2017년 2학기

이번 시간에 할 것

- Github 소개
- Centralized Workflow
- [실습] Github를 이용한 Centralized Workflow

GitHub

- https://github.com/
- 기본적으로 Git 호스팅 서비스 (Git 원격저장소를 제공하는 서비스).
- 뿐만 아니라, issue tracking, code review 등 프로젝트 진행에 필요한 많은 기능을 제공.
- 많은 수의 유명 오픈 소스 프로젝트가 github에서 호스팅되고 있으며, github를 이용하는 것이 현재 오픈 소스 개발의 대세라고 말할 수 있다.
- 공개 저장소는 무료, 비공개 저장소는 유료
- 편리한 웹 기반 인터페이스 - 파일 추가, 수정 후 commit등의 작업을 github 사이트에서 바로 할 수도 있다.

Github 프로젝트 사이트 구성

- Github에서 호스팅되는 프로젝트의 주소
  - https://github.com/(user id)/(project name)
  - https://github.com/(organization name)/(project name)
  - 사이트의 주소일 뿐만 아니라 Git 원격저장소의 주소이기도 하다.
  - 예) https://github.com/tensorflow/tensorflow

- Watch
  - 해당 프로젝트에 새로 만들어진 pull request와 issue에 대한 알림을 받는다 (email, github site).

Github 프로젝트 사이트 구성

- Star
  - '즐겨찾기' + '좋아요'
  - 프로젝트 운영자에 대한 감사의 표시
- Fork
  - 해당 저장소를 복제한 나의 저장소를 만든다.
- Code, Issues, Pull requests, Insights
- commits, branches, releases, contributors, license
- Programming languages

원격저장소를 이용하여 협업을 하는 방식

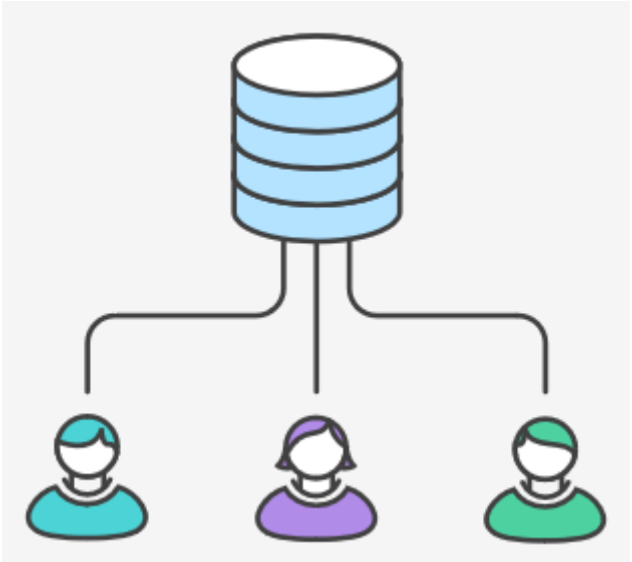
- 원격저장소를 이용해 여러 개발자가 함께 작업하는 방식에는 여러 가지가 있다.

- 그 중 본 강좌에서는 아래 두 가지 방식을 살펴보도록 하자.
  - Centralized workflow
  - Forking workflow

- Centralized workflow를 먼저 살펴보자.

Centralized Workflow

- 하나의 중앙 서버 (저장소)
- 모든 프로젝트 참여 개발자들이 중앙 저장소에 대한 read와 write권한을 가진다.
- 개발자들은 중앙 서버에 대한 pull / push를 하며 작업



Centralized Workflow

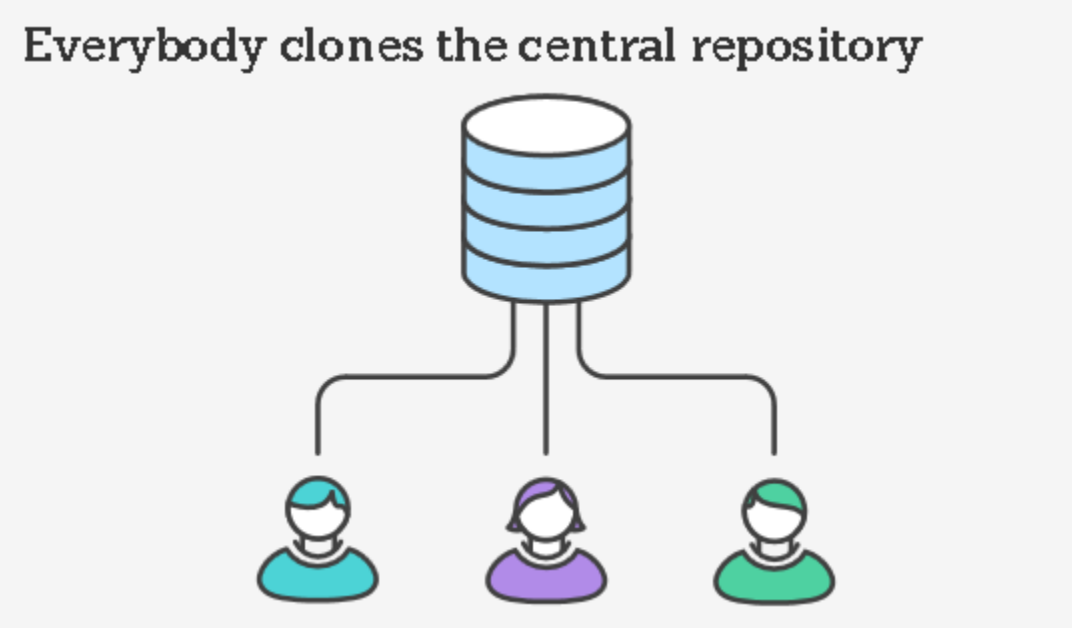
- 다양한 branch를 활용하며 작업할 수도 있다.
- 멤버가 정해져 있는 팀이 개발하는 프로젝트에 적절한 개발 방식.
- 예) 회사 내부의 프로젝트, 학교에서의 팀 프로젝트

Centralized Workflow Example

https://www.atlassian.com/git/tutorials/comparing-workflows

Centralized Workflow Example

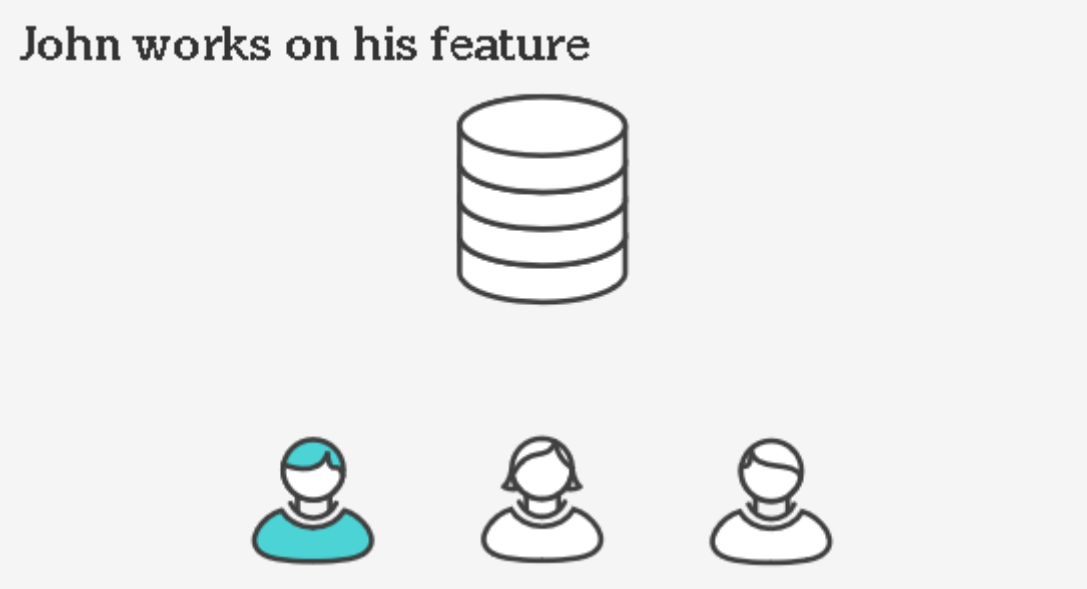
https://www.atlassian.com/git/tutorials/comparing-workflows




```
git clone ssh://user@host/path/to/repo.git
```

Centralized Workflow Example

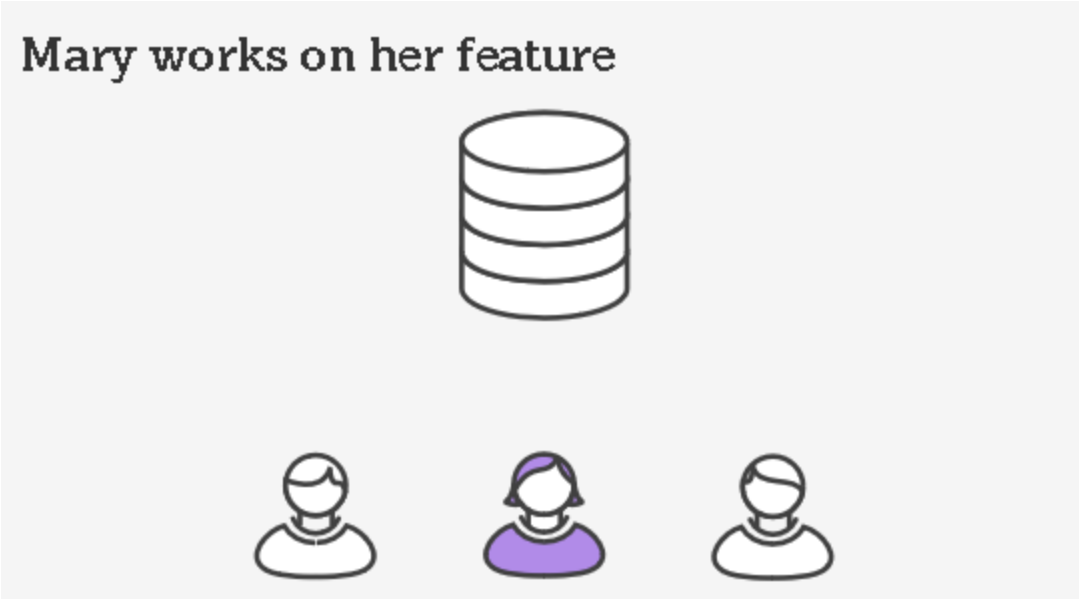
https://www.atlassian.com/git/tutorials/comparing-workflows




```
git status # View the state of the repo
git add <some-file> # Stage a file
git commit # Commit a file</some-file>
```

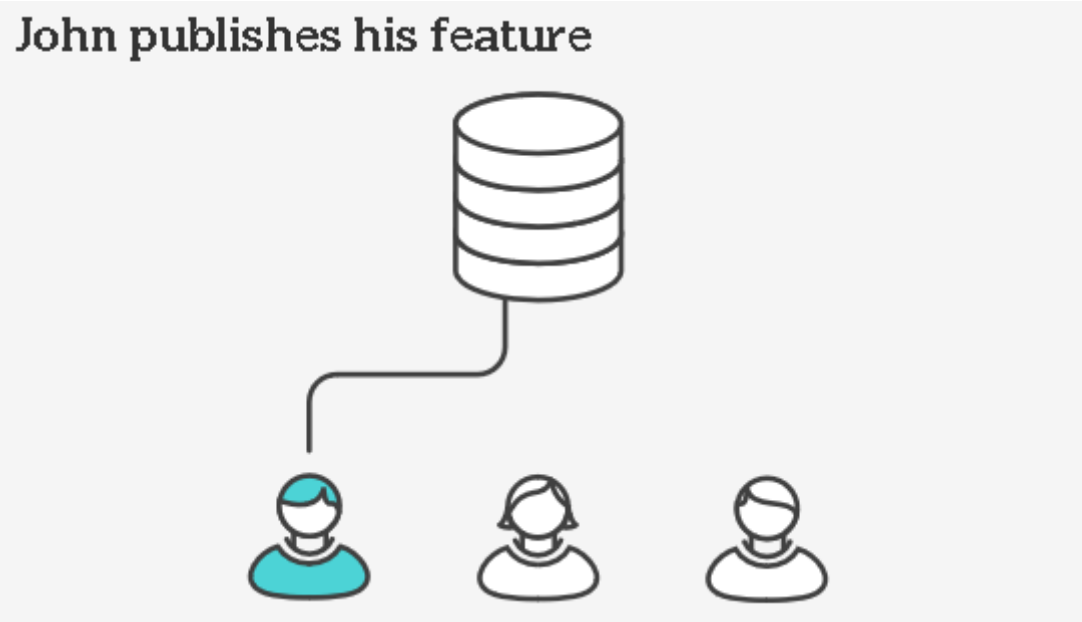
Centralized Workflow Example

https://www.atlassian.com/git/tutorials/comparing-workflows



Centralized Workflow Example

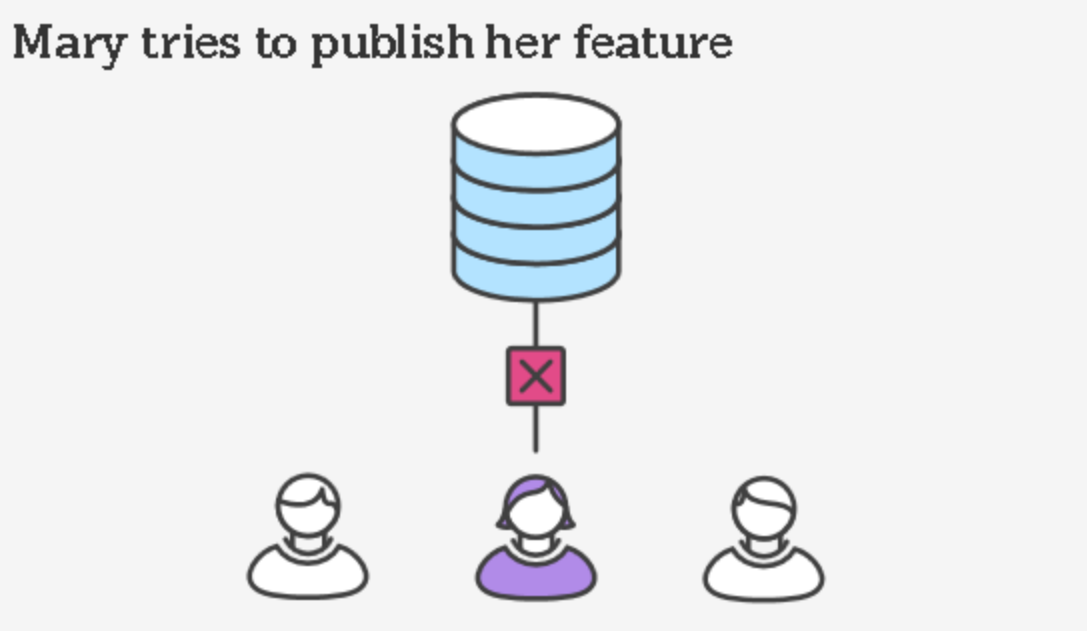
https://www.atlassian.com/git/tutorials/comparing-workflows




```
git push origin master
```

Centralized Workflow Example

https://www.atlassian.com/git/tutorials/comparing-workflows



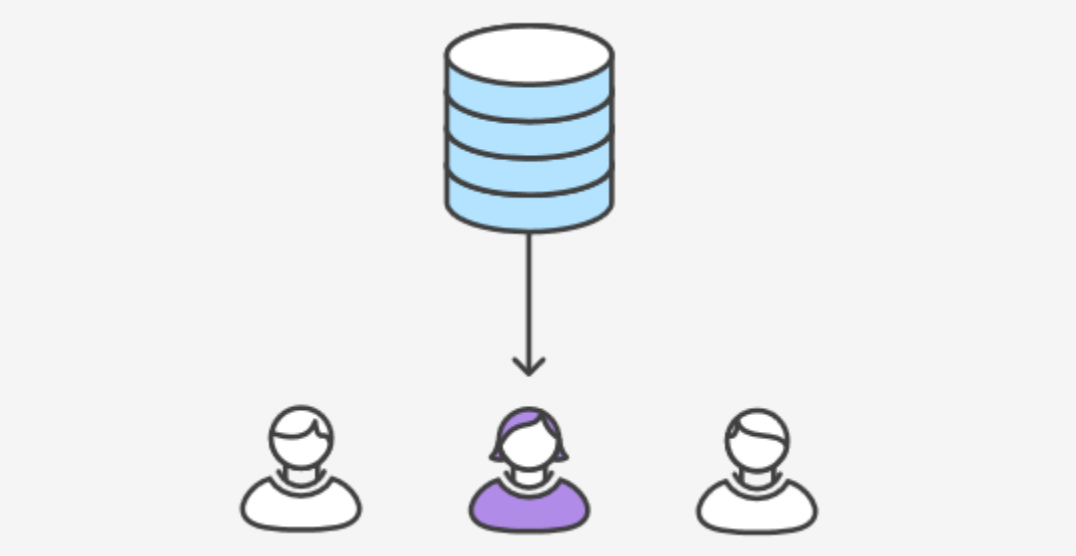

```
git push origin master
```

```
error: failed to push some refs to '/path/to/repo.git'
hint: Updates were rejected because the tip of your curre
hint: its remote counterpart. Merge the remote changes (e.
hint: before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --h
```

Centralized Workflow Example

https://www.atlassian.com/git/tutorials/comparing-workflows

Mary pulls the changes



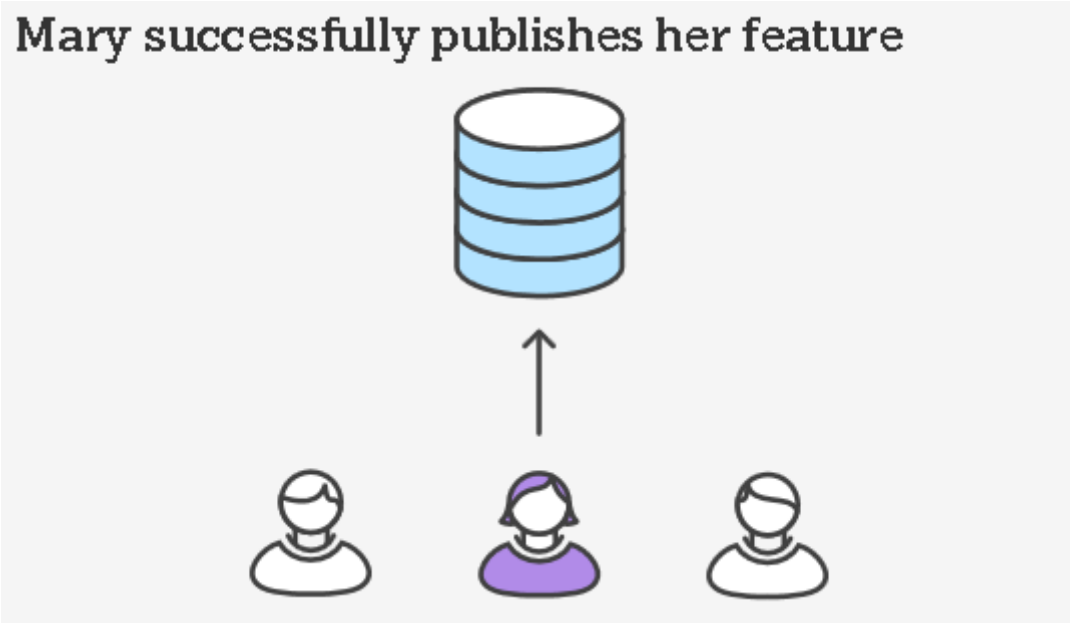
```
git pull origin master
```

```
CONFLICT (content): Merge conflict in <some-file>
```

만일 conflict이 발생한다면...

Centralized Workflow Example

https://www.atlassian.com/git/tutorials/comparing-workflows




```
git push origin master
```

고쳐서 다시 push

이번 강의의 실습에 대해...

- 이번 강의에서는 실제 협업 상황에서 여러 참여자의 역할을 직접 수행해보기 위해 ID를 하나 더 만들어 사용해 볼 것임.
  - Github에서 새로운 ID를 하나 더 만들자.

- 기존에 사용하던 ID를 id1, 새로 만든 ID를 id2라고 부르겠음.

[실습] Github를 이용한 Centralized Workflow

- id1이 'CentralizedTest'라는 저장소를 만든다.
  - Github에 id1으로 로그인해서 저장소를 새로 만든 후,

(Shell)

```
mkdir -p ~/github-2/id1/CentralizedTest  # 디렉터리 이름,위치는 변경가능
cd ~/github-2/id1/CentralizedTest

# Github에서 …or create a new repository on the command line에 나와있는 명령어를 copy & paste하여 파일추가, 지역저장소 생성, 원격저장소 추가를 하자
echo "# CentralizedTest" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:(id1)/CentralizedTest.git
git push -u origin master
```

[실습] Github를 이용한 Centralized Workflow

- id2가 작업에 참여하기 위해 CentralizedTest를 git clone 한다.

(Shell)

```
mkdir -p ~/github-2/id2  #디렉터리 이름,위치는 변경가능
cd ~/github-2/id2

git clone https://github.com/(id1)/CentralizedTest
# ssh url로 git clone하면 local machine에 저장된 ssh key를 이용하여 이후에 별도 로그인 없이 id1으로서 git pull & push를 하게 되기 때문에, ssh가 아닌 https url로 git clone을 하도록 하자.

cd CentralizedTest

git config user.name (id2)  # 이번 실습에서만, history상의 구분을 위해 이 저장소에서만 user name을 임시로 변경하자.
```

[실습] Github를 이용한 Centralized Workflow

- 두 종류의 다른 browser에서 각각 Github에 id1과 id2로 로그인 해보자.
- id1에서는 자신의 repository 목록에 CentralizedTest가 보일 것임.
- id2에서는 repository가 하나도 없는 것이 보일 것임.

[실습] Github를 이용한 Centralized Workflow

- id2가 파일을 추가하고 CentralizedTest에 push 하려고 시도해보자.

(Shell: id2/CentralizedTest)

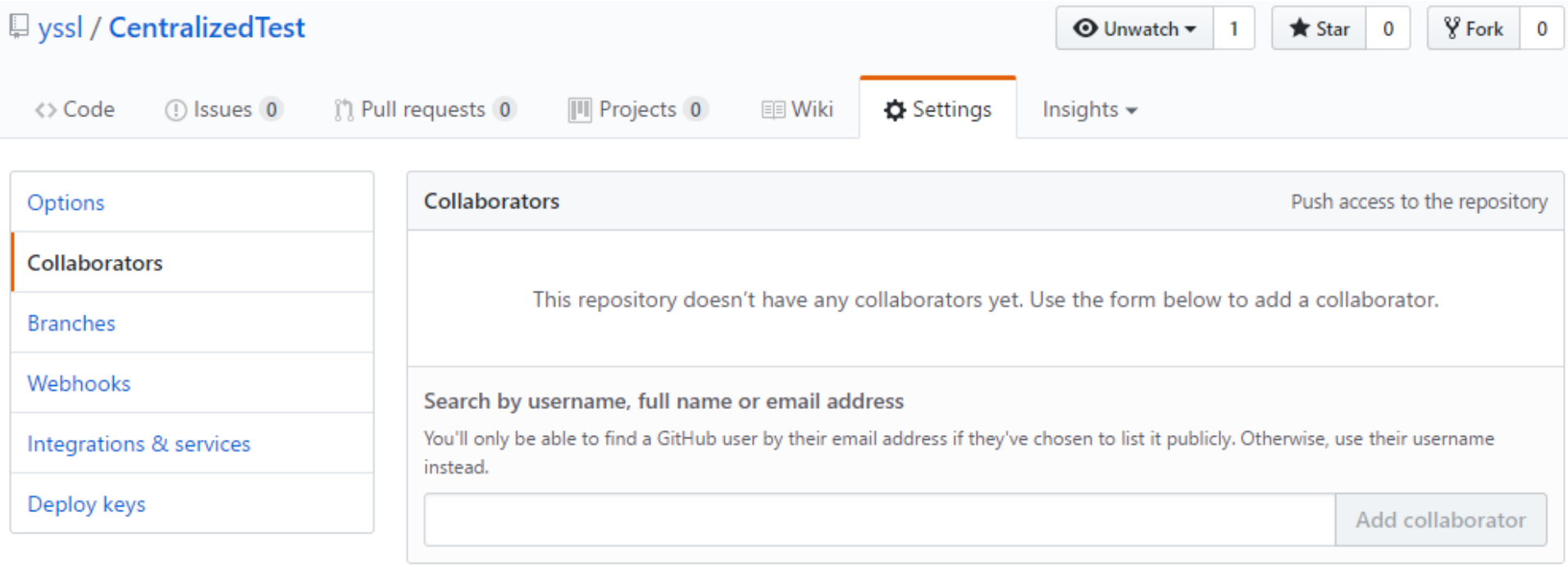
```
vi file1.txt  # Add 'hello'
git add .
git commit
git push origin master

(출력메시지)
Username for 'https://github.com': (id2)
Password for 'https://(id2)@github.com':
remote: Permission to (id1)/CentralizedTest.git denied to
(id2).
fatal: unable to access
'https://github.com/yssl/CentralizedTest/': The requested URL
returned error: 403
```

[실습] Github를 이용한 Centralized Workflow

- Why? - 기본적으로 Github 저장소는 owner만 write 권한을 가짐.
- read 권한은 공개저장소라면 누구나 가짐. (Github에서 무료로 만들어 사용하는 저장소는 공개저장소임)
- Centralized workflow로 작업하려면 다른 개발자에게도 write 권한을 줘야 함.

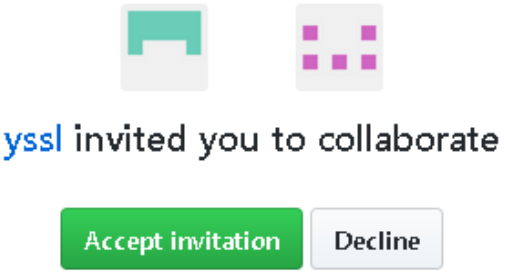
[실습] Github를 이용한 Centralized Workflow

- id1이 id2를 collaborator으로 초대
  - Github에 id1으로 로그인 후 CentralizedTest 저장소의 Settings - Collaborators에서 id2 입력 후 Add collaborator 누름

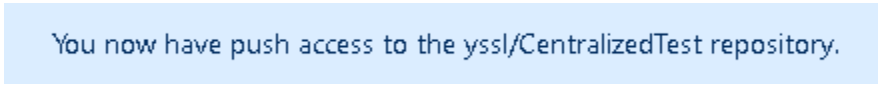


[실습] Github를 이용한 Centralized Workflow

- id2의 email로 온 invitation을 accept하면,



- 아래와 같이 id2가 이제 push access를 갖게 되었다는 메시지가 뜬다.



[실습] Github를 이용한 Centralized Workflow

- 이제 다시 id2로서 git push를 해보자.

(Shell: id2/CentralizedTest)

```
git push origin master

(출력메시지)
Username for 'https://github.com': (id2)
Password for 'https://(id2)@github.com':
Counting objects: 3, done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 281 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/(id1)/CentralizedTest
   0d863aa..b3f166e  master -> master
```

- Github의 CentralizedTest 페이지에서 commit이 반영된 기록을 살펴보자.

[실습] Github를 이용한 Centralized Workflow

- 이후에는 15-(Online)Git&Github-remote 강의에서처럼 id1과 id2가 push, pull을 반복하며 협업하면 된다.

- push 하기 전에는 항상 pull을 먼저 하는 습관!
- pull할 때 conflict발생하면 resolve

- 이미 push한 commit은 수정 / 삭제하지 말 것
- 새로운 작업을 시작하기 전에 항상 git pull을 먼저 할 것
- 작업이 끝나면 꼭 git push를 할 것

[실습] Github를 이용한 Centralized Workflow

- 만일 팀프로젝트를 Github로 진행하고자 한다면, 팀원 중 한 명이 Github 저장소를 만든 후 나머지 팀원들의 Github ID를 collaborator로 추가하면 된다.