

GitHub를 이용하여 협업하기

1. Repository간 상호 작용 종류

- 1) 원격저장소 조회(추가)하기 : **git remote**
- 2) 원격저장소에 밀어넣기 : **git push**
- 3) 원격저장소 갖고 와서 합치기 : **git pull**
- 4) 원격저장소 일단 갖고만 오기 : **git fetch**
- 5) 원격저장소 복사하기 : **git clone**

2. 원격저장소 조회(추가)하기

1) git remote [-v]


- 2) **Local Repository**와 상호작용하는 또는 상호작용할 수 있는 원격 저장소들의 목록을 조회
- 3) **-v** : 단축이름과 URL같이 보기
- 4) 예제를 위해 GitHub에 새로운 Repository를 생성한다.
- 5) 이번에는 README File 등 체크박스를 체크하지 않고 생성했다.

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 *

 gitinstructor ▾


 /

github-demo ✓


Great repository names are short and memorable. Need inspiration? How about **solid-disco**?

Description (optional)

GitHub 연습

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

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


☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾



Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▾

 You are creating a public repository in your personal account.

Create repository

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

 Set up in Desktop or [HTTPS](#) [SSH](#) <https://github.com/gitinstructor/github-demo.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 "# github-demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/gitinstructor/github-demo.git
git push -u origin main
```

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

```
git remote add origin https://github.com/gitinstructor/github-demo.git
git branch -M main
git push -u origin main
```

...or import code from another repository

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

[Import code](#)

6) git remote add origin <url>

- <url>에 있는 원격 저장소를 *origin*이라는 별명으로 추가하기

7) GitWork 폴더를 Git으로 초기화하고 README.md 파일 생성해서 Push하고, 방금 생성한 GitHub의 원격저장소를 추가한다.

```
henry-MacBook-Pro:GitWork henry$ git init
Initialized empty Git repository in /Users/henry/GitWork/.git/
henry-MacBook-Pro:GitWork henry$ git config user.name henry
henry-MacBook-Pro:GitWork henry$ git config user.email javaexpert@nate.com
henry-MacBook-Pro:GitWork henry$
```

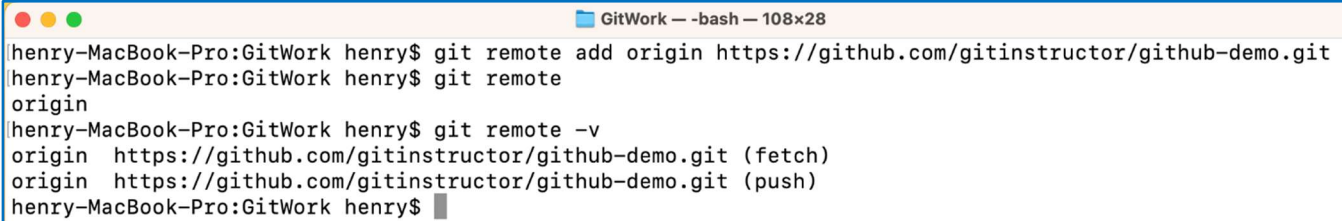
```
henry-MacBook-Pro:GitWork henry$ touch README.md
henry-MacBook-Pro:GitWork henry$ echo '#GitHub Tutorial' > README.md
henry-MacBook-Pro:GitWork henry$ git add README.md
henry-MacBook-Pro:GitWork henry$ git commit -m 'Add README.md File'
[main (root-commit) a378f37] Add README.md File
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
henry-MacBook-Pro:GitWork henry$
```

8) 원격 저장소를 추가했으면 -v 옵션으로 조회도 가능하다.

```
$ git remote add origin https://github.com/gitinstructor/github-demo.git
```

```
$ git remote
```

```
$ git remote -v
```



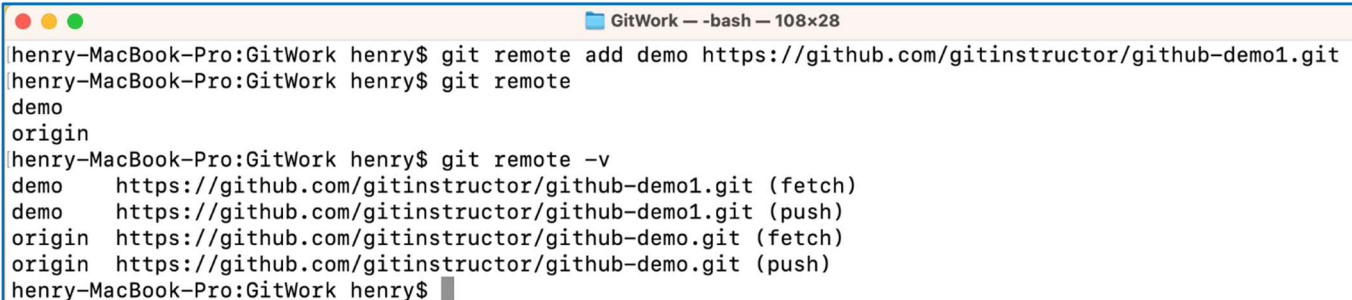
```
GitWork -- -bash -- 108x28
henry-MacBook-Pro:GitWork henry$ git remote add origin https://github.com/gitinstructor/github-demo.git
henry-MacBook-Pro:GitWork henry$ git remote
origin
henry-MacBook-Pro:GitWork henry$ git remote -v
origin https://github.com/gitinstructor/github-demo.git (fetch)
origin https://github.com/gitinstructor/github-demo.git (push)
henry-MacBook-Pro:GitWork henry$
```

9) GitHub에 또 하나의 github-demo1이라는 원격저장소를 추가하고 연결해보자.

```
$ git remote add demo https://github.com/gitinstructor/github-demo1.git
```

```
$ git remote
```

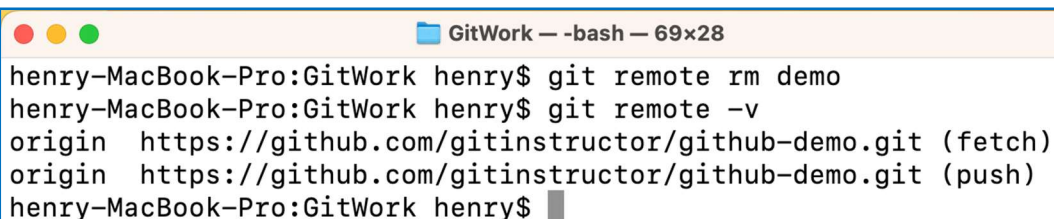
```
$ git remote -v
```



```
GitWork -- -bash -- 108x28
henry-MacBook-Pro:GitWork henry$ git remote add demo https://github.com/gitinstructor/github-demo1.git
henry-MacBook-Pro:GitWork henry$ git remote
demo
origin
henry-MacBook-Pro:GitWork henry$ git remote -v
demo https://github.com/gitinstructor/github-demo1.git (fetch)
demo https://github.com/gitinstructor/github-demo1.git (push)
origin https://github.com/gitinstructor/github-demo.git (fetch)
origin https://github.com/gitinstructor/github-demo.git (push)
henry-MacBook-Pro:GitWork henry$
```

10) 방금 추가한 demo라는 별칭을 갖는 원격저장소를 삭제해보자.

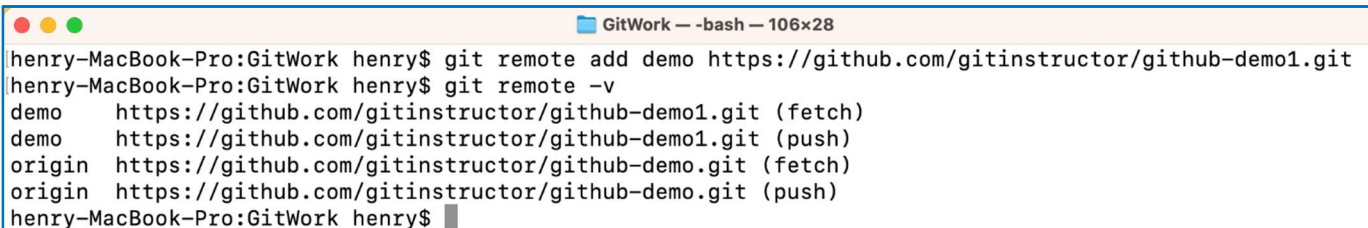
```
$ git remote rm demo
```



```
GitWork -- -bash -- 69x28
henry-MacBook-Pro:GitWork henry$ git remote rm demo
henry-MacBook-Pro:GitWork henry$ git remote -v
origin https://github.com/gitinstructor/github-demo.git (fetch)
origin https://github.com/gitinstructor/github-demo.git (push)
henry-MacBook-Pro:GitWork henry$
```

11) 삭제한 demo라는 원격저장소를 다시 추가해보자.

```
$ git remote add demo https://github.com/gitinstructor/github-demo1.git
```



```
GitWork -- -bash -- 106x28
henry-MacBook-Pro:GitWork henry$ git remote add demo https://github.com/gitinstructor/github-demo1.git
henry-MacBook-Pro:GitWork henry$ git remote -v
demo https://github.com/gitinstructor/github-demo1.git (fetch)
demo https://github.com/gitinstructor/github-demo1.git (push)
origin https://github.com/gitinstructor/github-demo.git (fetch)
origin https://github.com/gitinstructor/github-demo.git (push)
henry-MacBook-Pro:GitWork henry$
```

3. 원격저장소에 밀어넣기

1) `git push -u origin main`

2) **Remote Repository**의 [main] Branch를 *origin*의 [main] Branch로 Push

3) `-u` : Default 설정

4) 앞으로 추가 동작없이 **git push**했을 때 자동으로 나의 [main] Branch와 원격의 *origin* Branch사이에 연결의 상호작용이 일어나도록 하는 옵션이다.

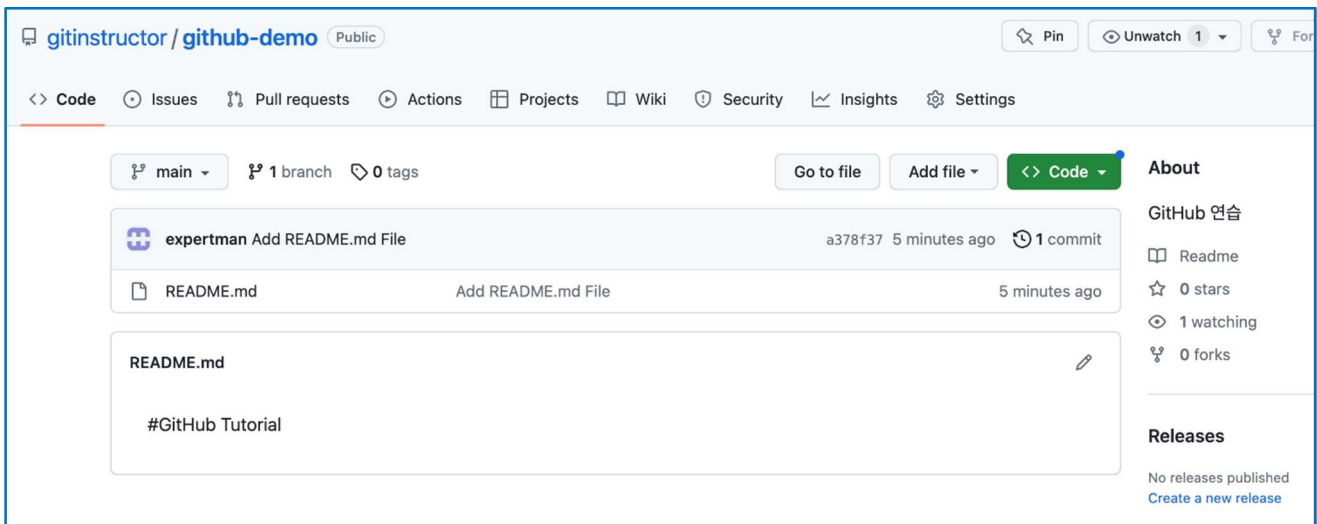
5) 즉, 다음부터는 [main] Branch에서 Push하면 자동으로 *origin*의 [main] Branch으로 밀어넣기가 되는 것을 의미한다.

6) 위에서 생성한 README.md 파일을 원격저장소에 Push한다.

\$ git push -u origin main

```
henry-MacBook-Pro:GitWork henry$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 238 bytes | 238.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/gitinstructor/github-demo.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
henry-MacBook-Pro:GitWork henry$
```

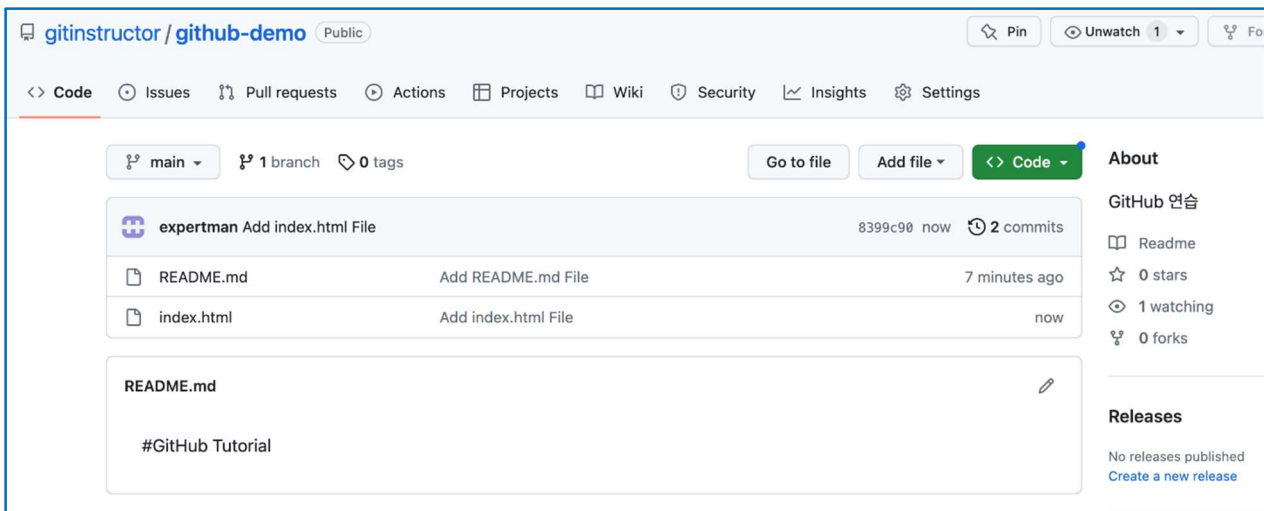
7) 그럼, 원격저장소의 README.md가 성공적으로 Push 되었음을 확인할 수 있다.



- 8) 이미 **-u** 옵션을 이용해서 Default 원격 저장소를 **origin**으로 설정했기 때문에 다음과 같이 index.html을 생성하고 그냥 Push해도 자동으로 **origin**으로 Push된다.

```
henry-MacBook-Pro:GitWork henry$ touch index.html
henry-MacBook-Pro:GitWork henry$ git add index.html
[henry-MacBook-Pro:GitWork henry$ git commit -m 'Add index.html File'
[main 8399c90] Add index.html File
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html
[henry-MacBook-Pro:GitWork henry$ git 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), 280 bytes | 280.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/gitinstructor/github-demo.git
 a378f37..8399c90  main -> main
henry-MacBook-Pro:GitWork henry$
```

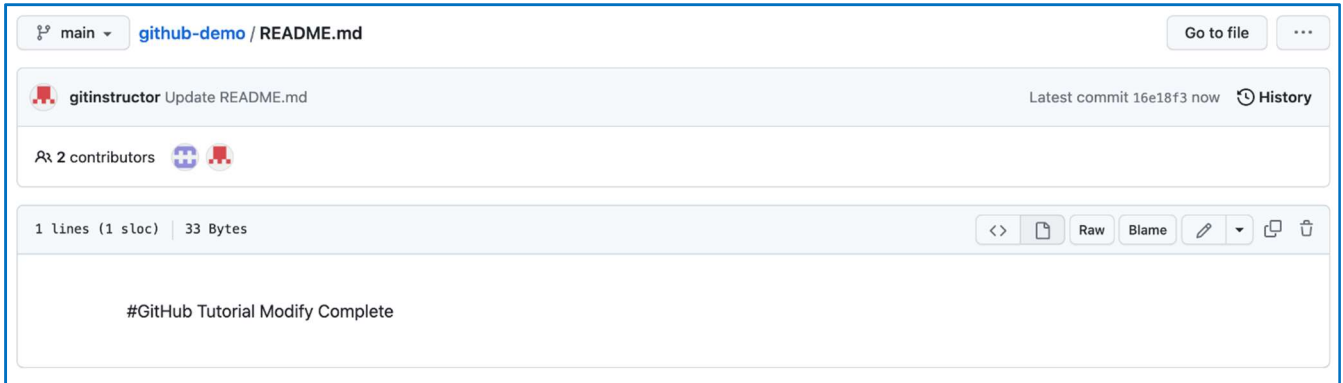
- 9) **origin**으로 설정된 github-demo Repository에 가면 index.html이 성공적으로 Push 되었음을 확인할 수 있다.



4. 원격저장소 갖고 와서 합치기

1) `git pull (origin main)`

- 2) **origin**을 **Local Repository**의 [main] Branch로 갖고 오라는 명령(Merge)
- 3) 원격저장소의 파일이 **Local Repository**에 Merge가 되기 때문에 나의 파일을 덮어쓰게 된다.
- 4) GitHub의 README.md 파일을 다음과 같이 수정한다.



- 5) 원격저장소와 Pull하고 README.md 파일을 확인해 보자.

\$ git pull origin main

```
GitWork -- -bash -- 72x28
henry-MacBook-Pro:GitWork henry$ git pull origin main
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 695 bytes | 231.00 KiB/s, done.
From https://github.com/gitinstructor/github-demo
 * branch                main                -> FETCH_HEAD
    8399c90..16e18f3      main                -> origin/main
Updating 8399c90..16e18f3
Fast-forward
 README.md | 2 + -
 1 file changed, 1 insertion(+), 1 deletion(-)
henry-MacBook-Pro:GitWork henry$ cat README.md
#GitHub Tutorial Modify Complete
henry-MacBook-Pro:GitWork henry$
```

- 6) GitHub에서 수정한 내용으로 변경되었음을 확인할 수 있다.

7) **git status**나 **git log**를 통해서도 확인이 가능하다.

```
GitWork — -bash — 76x28

henry-MacBook-Pro:GitWork henry$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
henry-MacBook-Pro:GitWork henry$ git log
commit 16e18f3b57b2c3c88a688f42769e802b8d60f32d (HEAD -> main, origin/main)
Author: gitinstructor <45567479+gitinstructor@users.noreply.github.com>
Date: Thu Apr 13 09:47:48 2023 +0900

    Update README.md

commit 8399c9018ff664c77ab8fed143bec5c42be3c649
Author: henry <javaexpert@nate.com>
Date: Thu Apr 13 09:46:25 2023 +0900

    Add index.html File

commit a378f37ad3bccb1df371a041826b2d59d4a234c0
Author: henry <javaexpert@nate.com>
Date: Thu Apr 13 09:40:07 2023 +0900

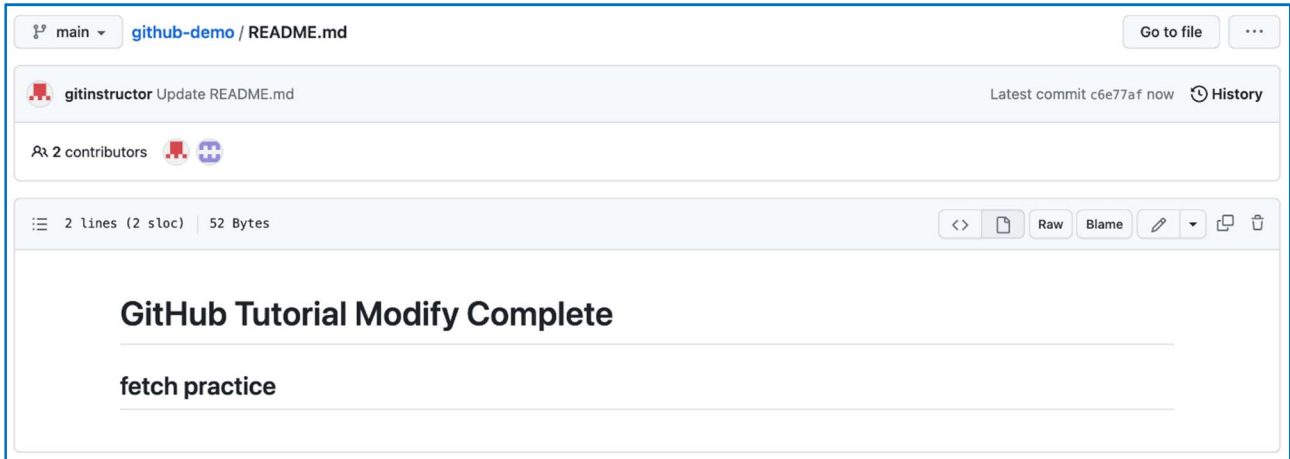
    Add README.md File
henry-MacBook-Pro:GitWork henry$
```


5. 원격저장소 일단 갖고만 오기

1) git fetch (origin main)

2) 동기화(덮어쓰기)시키지 않고(Merge하지 않고), **origin**을 **Local Repository**의 [main] Branch로 일단 갖고 오기

3) 예제를 위해서 github-demo Repository의 README.md를 다시 수정한다.



4) 그 다음, **Fetch** 를 수행한다.

\$ git fetch origin

```
henry-MacBook-Pro:GitWork henry$ git fetch origin
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 711 bytes | 355.00 KiB/s, done.
From https://github.com/gitinstructor/github-demo
  16e18f3..c6e77af  main      -> origin/main
henry-MacBook-Pro:GitWork henry$
```

5) **Local Repository**의 README.md를 확인해 보면, 변경되지 않음을 알 수 있다.

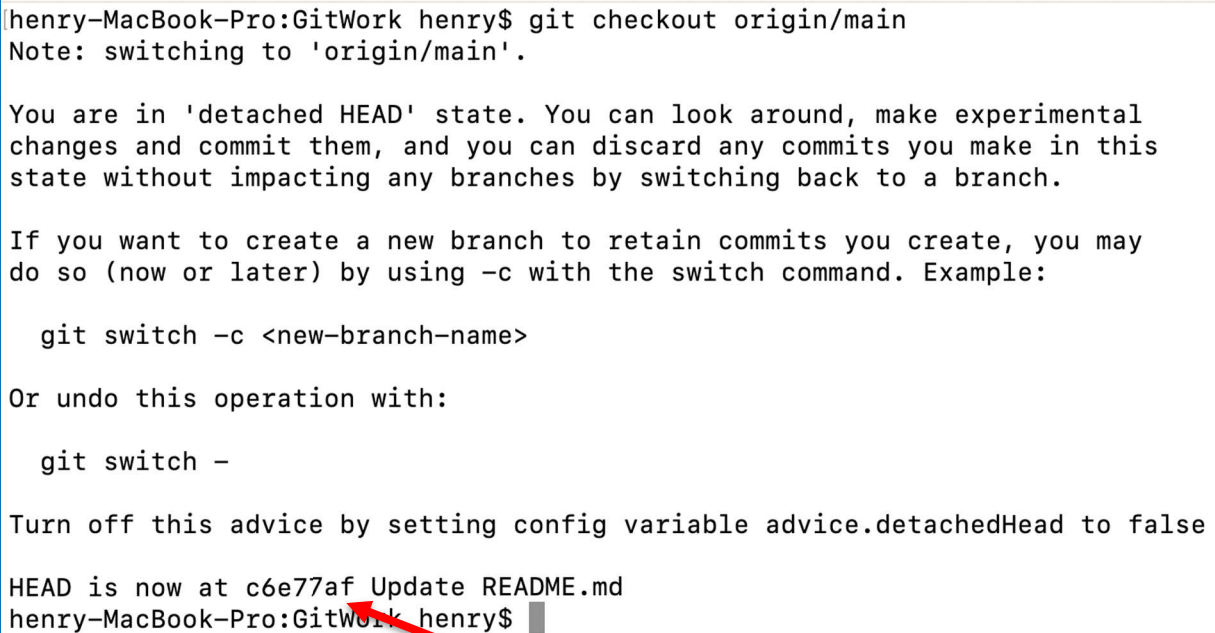
```
henry-MacBook-Pro:GitWork henry$ cat README.md
#GitHub Tutorial Modify Complete
henry-MacBook-Pro:GitWork henry$
```

6) 그렇다면, 방금 원격저장소와 Fetch한 정보는 어디에 저장되어 있는가?

7) 그것은 위의 그림에서 보면 [origin/main]라는 Branch에서 확인할 수 있다.

8) 해당 Branch로 이동해서 확인해 보자.

\$ git checkout origin/main



```
henry-MacBook-Pro:GitWork henry$ git checkout origin/main
Note: switching to 'origin/main'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

    git switch -c <new-branch-name>

Or undo this operation with:

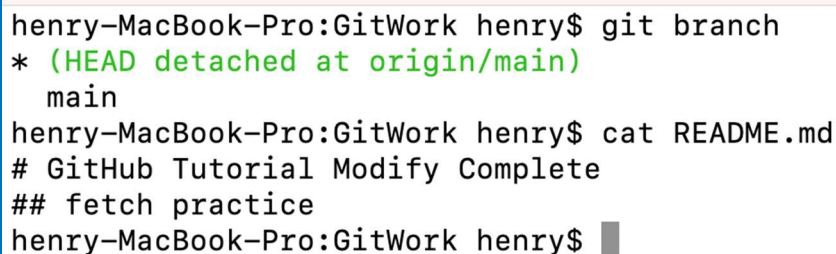
    git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at c6e77af Update README.md
henry-MacBook-Pro:GitWork henry$
```

9) 위의 그림에 보면 해당 Branch의 HEAD는 현재 c6e77af으로 되어 있다.

10) 이 Branch는 Fetch한 정보를 저장하기 위한 Branch라고 보면 된다. 이 Branch의 README.md의 내용을 보면 GitHub에서 수정한 내용임을 확인할 수 있다.



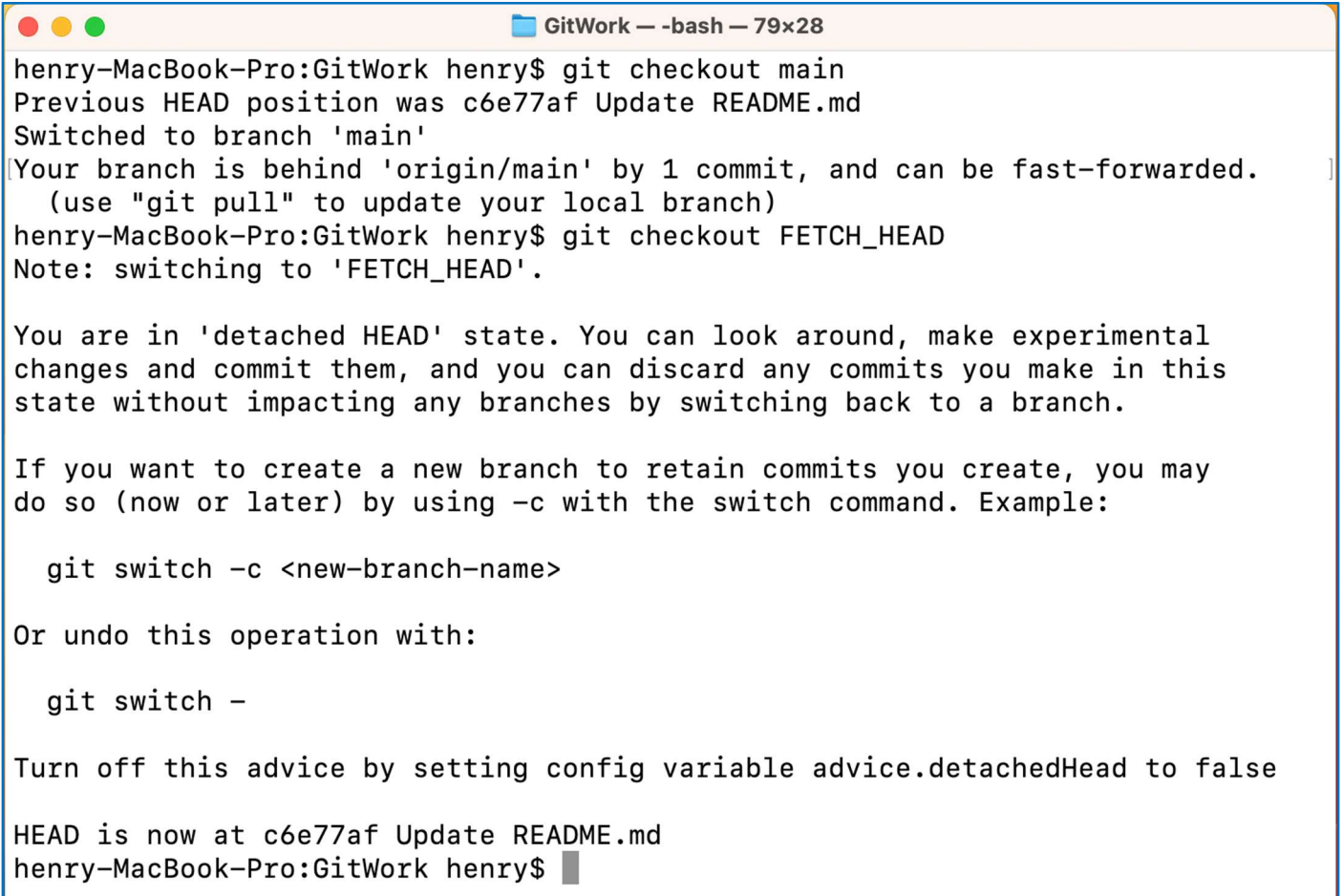
```
henry-MacBook-Pro:GitWork henry$ git branch
* (HEAD detached at origin/main)
  main
henry-MacBook-Pro:GitWork henry$ cat README.md
# GitHub Tutorial Modify Complete
## fetch practice
henry-MacBook-Pro:GitWork henry$
```

11) 이렇게 변경된 내용을 확인하기 위한 Branch는 [origin/main]만 있는 것이 아니다. 다음과 같은 Branch로도 확인이 가능하다.

12) 이 Branch의 이름은 ***FETCH_HEAD***이다. 일단 [main]로 Checkout한 후, [FETCH_HEAD] Branch로 Checkout해보자.

\$ git checkout main

\$ git checkout FETCH_HEAD

A terminal window titled "GitWork -- -bash -- 79x28" showing the execution of git checkout commands. The user switches from an unnamed branch to 'main', then to 'FETCH_HEAD', entering a 'detached HEAD' state. The terminal displays messages about the previous HEAD position, commit history, and instructions for using git pull, git switch, and git switch - to manage branches and the detached state. The session ends with the HEAD position confirmed at c6e77af.

```
henry-MacBook-Pro:GitWork henry$ git checkout main
Previous HEAD position was c6e77af Update README.md
Switched to branch 'main'
[Your branch is behind 'origin/main' by 1 commit, and can be fast-forwarded.
 (use "git pull" to update your local branch)
henry-MacBook-Pro:GitWork henry$ git checkout FETCH_HEAD
Note: switching to 'FETCH_HEAD'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

    git switch -c <new-branch-name>

Or undo this operation with:

    git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at c6e77af Update README.md
henry-MacBook-Pro:GitWork henry$
```

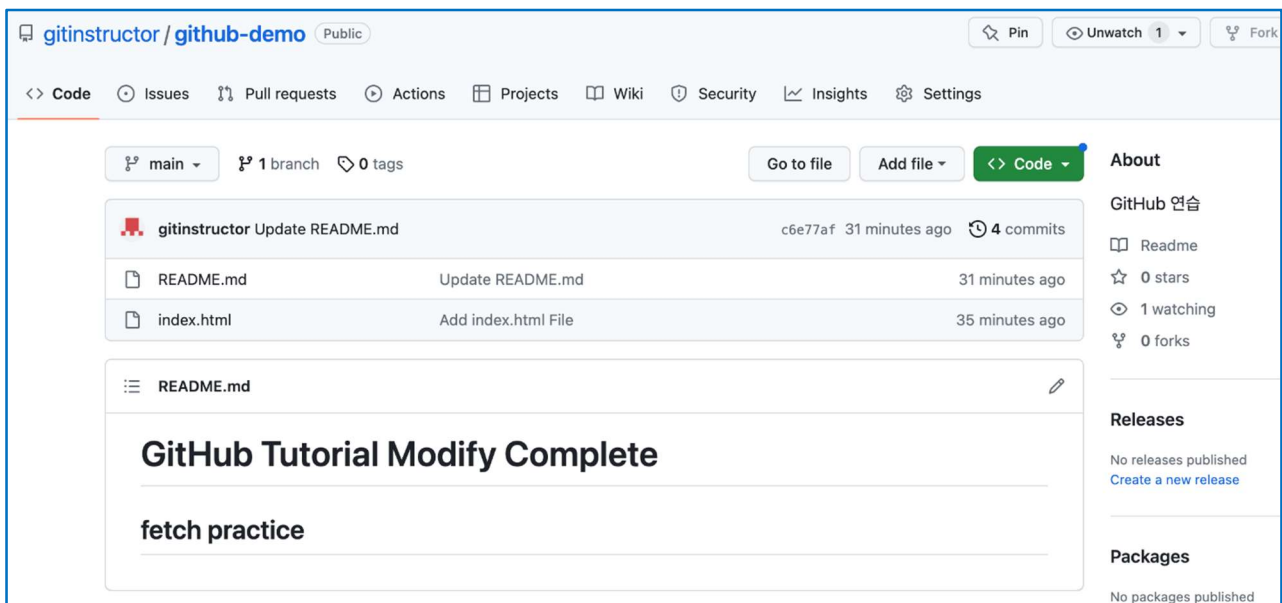
6. 원격저장소 복사하기

1) `git clone <url>`

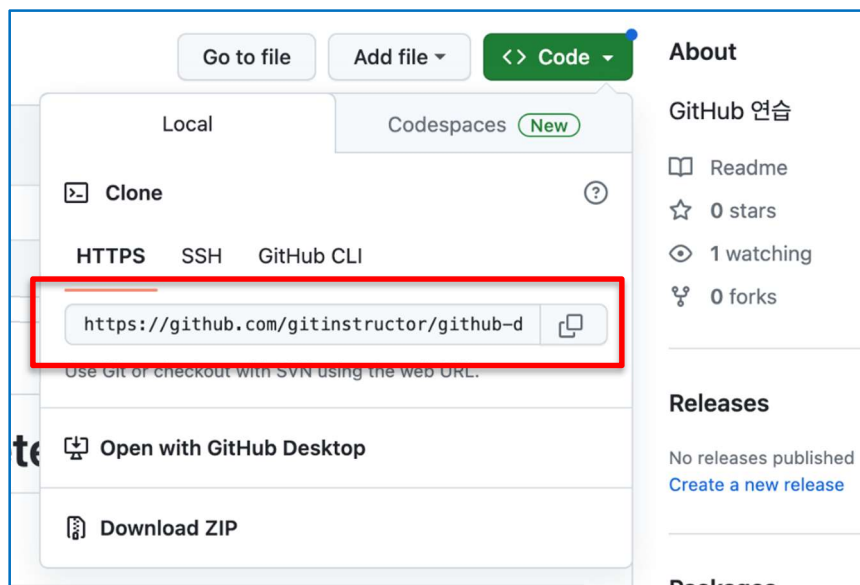
- 2) `<url>`에 있는 원격저장소 내용을 현재 디렉토리에 복사해오기
- 3) **origin** 자동 생성
- 4) 예제를 위해 GitWork를 다시 Git으로 초기화한다.

```
henry-MacBook-Pro:GitWork henry$ git init
Initialized empty Git repository in /Users/henry/GitWork/.git/
henry-MacBook-Pro:GitWork henry$ git config user.name henry
henry-MacBook-Pro:GitWork henry$ git config user.email javaexpert@nate.com
henry-MacBook-Pro:GitWork henry$
```

- 5) 디렉토리를 `git init`으로 초기화 한 후, github-demo 원격저장소를 Clone 해보자.

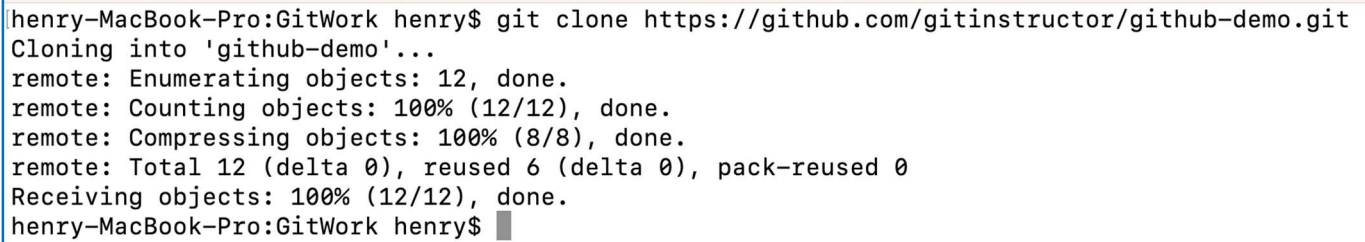


- 6) 좌측 상단의 **[Code]**라는 초록색 버튼을 클릭하여 URL을 복사한다.



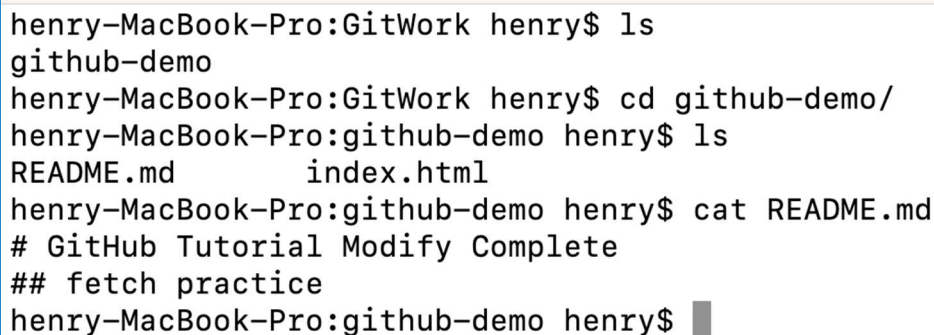
7) 그리고 GitWork에 Clone한다.

\$ git clone <url>



```
henry-MacBook-Pro:GitWork henry$ git clone https://github.com/gitinstructor/github-demo.git
Cloning into 'github-demo'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 12 (delta 0), reused 6 (delta 0), pack-reused 0
Receiving objects: 100% (12/12), done.
henry-MacBook-Pro:GitWork henry$
```

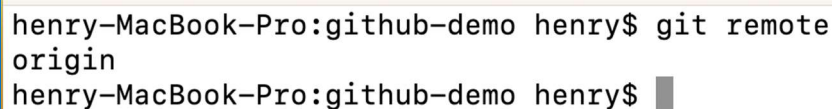
8) 해당 디렉토리의 파일과 그 파일의 내용을 확인해보자.



```
henry-MacBook-Pro:GitWork henry$ ls
github-demo
henry-MacBook-Pro:GitWork henry$ cd github-demo/
henry-MacBook-Pro:github-demo henry$ ls
README.md      index.html
henry-MacBook-Pro:github-demo henry$ cat README.md
# GitHub Tutorial Modify Complete
## fetch practice
henry-MacBook-Pro:github-demo henry$
```

9) 참고로 **git remote**로 확인해 보자. 자동으로 **origin**이 추가된 것을 알 수 있다.

\$ git remote



```
henry-MacBook-Pro:github-demo henry$ git remote
origin
henry-MacBook-Pro:github-demo henry$
```

7. 원격저장소를 이용하여 협업하기


- 1) **Local Repository**는 변경되었는데, 원격 저장소는 변경이 없는 경우
- 2) **Local Repository**는 변경이 없는데, 원격 저장소가 변경이 있는 경우 : **pull & push**
- 3) **Local Repository**도 변경이 있고, 원격저장소도 변경이 있는 경우 : **pull request**

8. Local Repository는 변경이 없는데, 원격 저장소가 변경이 있는 경우 : **pull & push**

1) 예제를 위해 GitHub에는 github-demo Repository를 생성하고 GitWork를 **git init**으로 초기화했다.

```
henry-MacBook-Pro:GitWork henry$ git init
Initialized empty Git repository in /Users/henry/GitWork/.git/
henry-MacBook-Pro:GitWork henry$ git config user.name henry
henry-MacBook-Pro:GitWork henry$ git config user.email javaexpert@nate.com
henry-MacBook-Pro:GitWork henry$
```

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

 Set up in Desktop or ☐ HTTPS ☐ SSH <https://github.com/gitinstructor/github-demo.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 "# github-demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/gitinstructor/github-demo.git
git push -u origin main
```

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

```
git remote add origin https://github.com/gitinstructor/github-demo.git
git branch -M main
git push -u origin main
```

...or import code from another repository

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

[Import code](#)

2) 다음과 같이 README.md 파일을 생성하고 **add, commit**까지 마친다.

```
henry-MacBook-Pro:github-demo henry$ echo '# Github Demo' > README.md
henry-MacBook-Pro:github-demo henry$ git add README.md
henry-MacBook-Pro:github-demo henry$ git commit -m 'Add README.md File'
[main bccea74] Add README.md File
1 file changed, 1 insertion(+), 2 deletions(-)
henry-MacBook-Pro:github-demo henry$
```

3) 원격 저장소를 **origin**이라는 이름으로 등록한다.

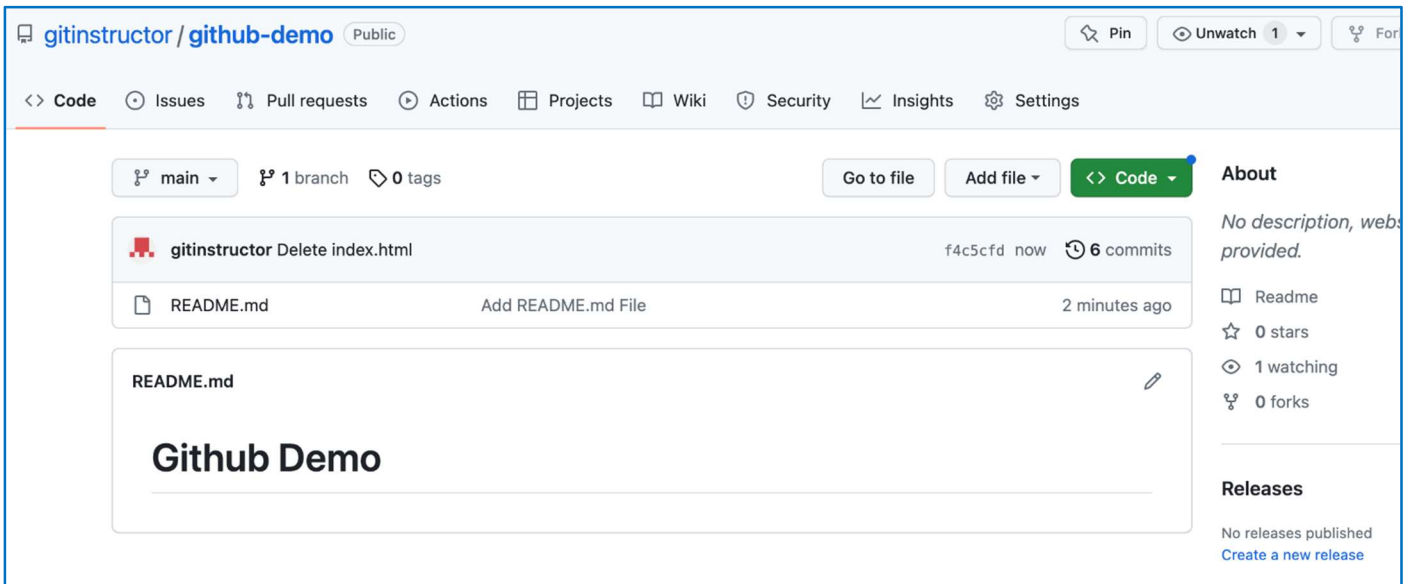
```
henry-MacBook-Pro:github-demo henry$ git remote add origin https://github.com/gitinstructor/github-demo.git
henry-MacBook-Pro:github-demo henry$ git remote -v
origin https://github.com/gitinstructor/github-demo.git (fetch)
origin https://github.com/gitinstructor/github-demo.git (push)
henry-MacBook-Pro:github-demo henry$
```


4) 이제 원격저장소와 연결되었기 때문에 README.md 파일을 Push한다.

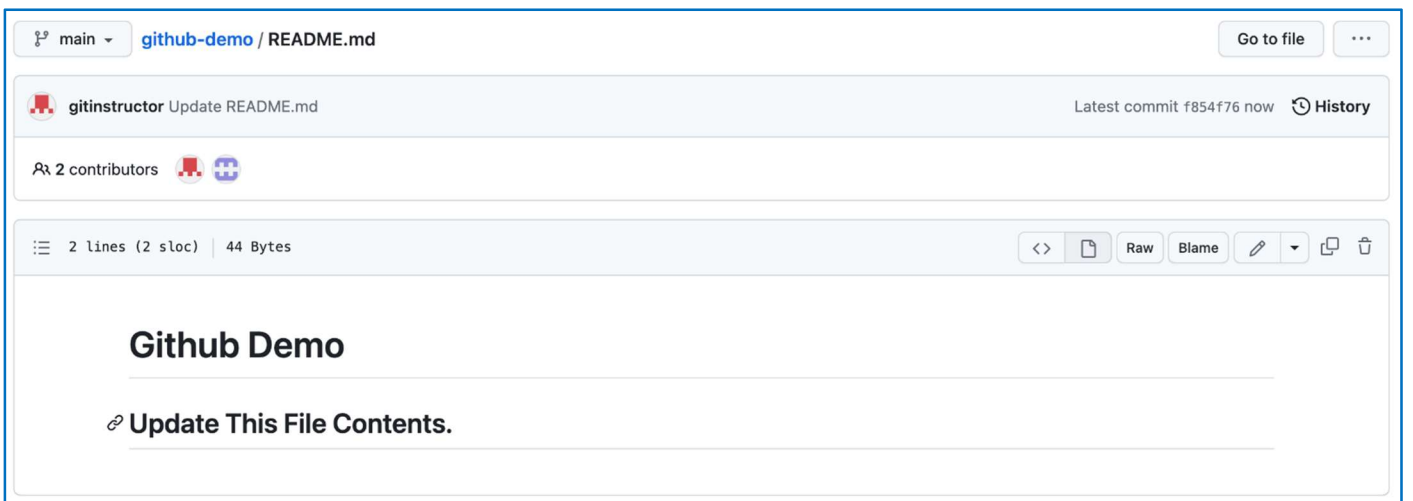
\$ git push -u origin main

```
henry-MacBook-Pro:github-demo henry$ git push -u origin main
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 4 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (15/15), 2.08 KiB | 2.08 MiB/s, done.
Total 15 (delta 0), reused 12 (delta 0), pack-reused 0
To https://github.com/gitinstructor/github-demo.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
henry-MacBook-Pro:github-demo henry$
```

5) 아래와 같이 원격저장소에서도 README.md를 확인할 수 있다.



6) 이번에는 원격 저장소에서 README.md 파일을 다음과 같이 수정하였다.



- 7) 이번 시나리오대로 **Local Repository**는 변경이 없는데, 원격에 변경사항이 발생한 경우가 된다.
이럴 경우에는 동기화(덮어쓰기)를 하면 된다.
- 8) 동기화를 위해 **git pull**를 수행한다. 그리고 Push할 때 **-u** 옵션으로 **origin**을 [main] Branch와 기본적으로 연결해 놓았기 때문에 그냥 **git pull**만 수행하면 된다.

\$ git pull

```
henry-MacBook-Pro:github-demo henry$ git pull
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 1.24 KiB | 317.00 KiB/s, done.
From https://github.com/gitinstructor/github-demo
   bccea74..f854f76  main       -> origin/main
Updating bccea74..f854f76
Fast-forward
 README.md | 1 +
 index.html | 0
2 files changed, 1 insertion(+)
delete mode 100644 index.html
henry-MacBook-Pro:github-demo henry$
```

- 9) 다음과 같이 원격에서 수정한 README.md가 **Local Repository**에 반영되었는지 확인해보자.

\$ cat README.md

\$ git log

```
henry-MacBook-Pro:github-demo henry$ cat README.md
# Github Demo
## Update This File Contents.
henry-MacBook-Pro:github-demo henry$
```

```
henry-MacBook-Pro:github-demo henry$ git log
commit 017ec6cd5dd5f79c7634b05e1fec72d8dd56843 (HEAD -> main, origin/main)
Author: gitinstructor <45567479+gitinstructor@users.noreply.github.com>
Date: Thu Apr 13 10:41:36 2023 +0900

    Update README.md

commit 0a4f27f3903ea9b0b0f323bb3cab1a994b8bb4cd
Author: henry <javaexpert@nate.com>
Date: Thu Apr 13 10:40:37 2023 +0900

    Add README.md File
henry-MacBook-Pro:github-demo henry$
```