

Git – 05

목차

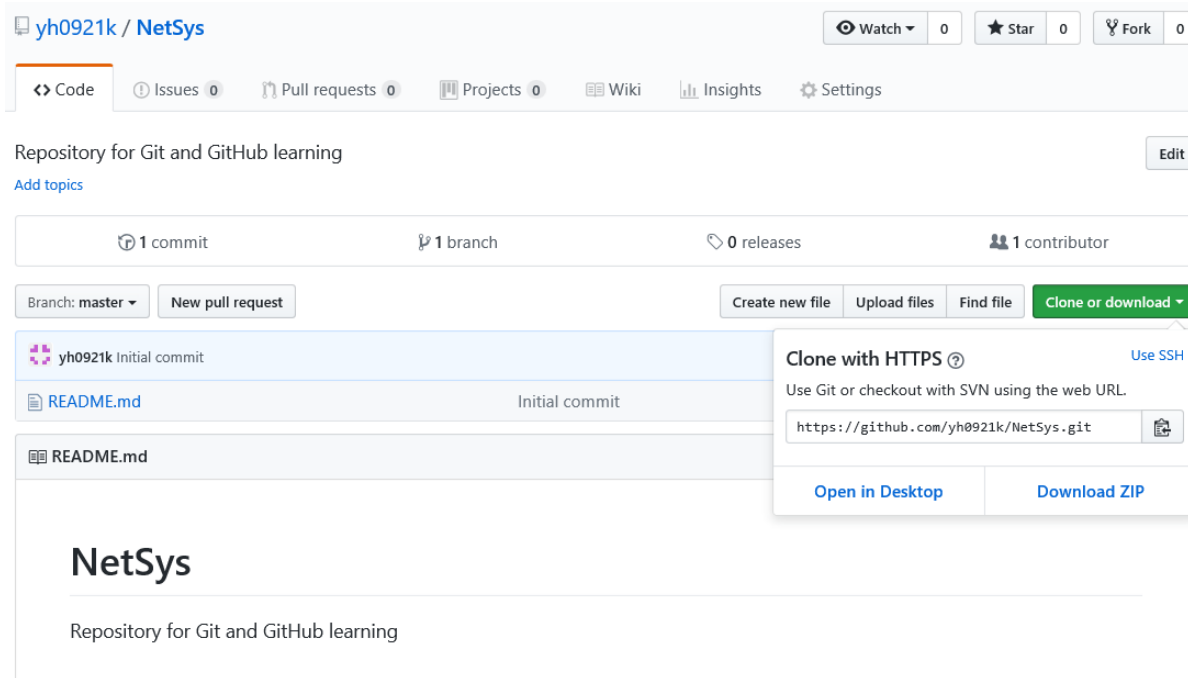
- Part 1. Git 기본과 원격 저장소
 - Chapter 1. 버전 관리 시스템과 Git
 - Chapter 2. Git 설치와 설정
 - Chapter 3. 로컬 저장소 사용을 위한 Git 기본
 - Chapter 4. 원격 저장소와 GitHub
 - Chapter 5. 원격 저장소와 Git

Chapter 5. 원격 저장소와 Git

- git clone : 원격 저장소의 내용을 로컬 저장소로 가져오기
 - 원격 저장소의 모든 내용을 로컬 저장소로 복사한다.
- git remote : 로컬 저장소와 원격 저장소를 연결하기
 - 로컬 저장소를 특정 원격 저장소와 연결한다.
- git push : 로컬 작업 내역을 원격 저장소에 올리기
 - 로컬 저장소의 내용을 보내거나 로컬 저장소의 변경 사항을 원격 저장소로 보낸다.
- git fetch & git pull : 원격 저장소와 로컬 저장소의 간격 메꾸기
 - fetch : 로컬 저장소와 원격 저장소의 변경 사항이 다를 때 이를 비교 대조하고 git merge 명령어와 함께 최신 데이터를 반영하거나 충돌 문제 등을 해결한다.
 - pull : git remote 명령을 통해 서로 연결된 원격 저장소의 최신 내용을 로컬 저장소로 가져오면서 병합한다.(push와 반대 성격의 명령어)

git clone

- 원격 저장소의 내용을 로컬 저장소로 가져오기
 - Clone
 - 내가 생성한 원격 저장소를 내 컴퓨터와 연결해서 데이터를 복사하는 작업
 - fork한 원격 저장소를 내 컴퓨터와 연결해서 데이터를 복사하는 작업
 - fork하지 않은 저장소를 가져올 수 있지만, 사용에 제한이 있다.
 - Git Bash나 리눅스 상에서 git clone 명령을 이용해 복제할 수 있고, 이때 네트워크 프로토콜로 SSH 또는 HTTPS를 이용할 수 있다.



git clone

- 원격 저장소의 내용을 로컬 저장소로 가져오기
 - git clone <https://github.com/yh0921k/NetSys.git>
 - 로컬 디렉터리로 GitHub에 있던 저장소가 복사된다.
 - 작업 디렉터리를 새로 만들어서 진행

```
kyh@DESKTOP-0UBU3AH MINGW64 ~/Desktop/LabSeminar/git_tutorial2
$ git clone https://github.com/yh0921k/NetSys
Cloning into 'NetSys'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

```
root@netsys:~/LabSeminar/git_tutorial2# git clone https://github.com/yh0921k/NetSys
'NetSys'에 복제합니다 ...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
오브젝트 묶음 푸는 중: 100% (3/3), 완료.
연결을 확인하는 중입니다 ... 완료.
```

- cd NetSys

git clone

- 원격 저장소의 내용을 로컬 저장소로 가져오기
 - git status

```
kyh@DESKTOP-0UBU3AH MINGW64 ~/Desktop/LabSeminar/git_tutorial2/NetSys (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

```
root@netsys:~/LabSeminar/git_tutorial2/NetSys# git status
현재 브랜치 master
브랜치가 'origin/master'에 맞게 업데이트된 상태입니다.
커밋할 사항 없음, 작업 폴더 깨끗함
```

- clone한 저장소의 이름은 origin으로 지어진다.
- clone을 이용하여 fork한 저장소도 옮겨올 수 있다.

git remote

- 로컬 저장소와 원격 저장소를 연결하기
 - 일반적인 프로젝트의 흐름
 - 빈 원격 저장소 생성 > 책임자가 기본 프로젝트 구조 생성 > 협업자들 모두가 빈 원격 저장소를 clone하여 작업 진행
 - 책임자가 기존에 작업 해놓은 로컬 저장소가 있고, 이를 원격 저장소와 연결 한다고 생각해보자.
 - 빈 원격 저장소를 clone하고 기존에 작업하던 파일들을 옮기는 것도 가능하지만, 번거롭다.
 - 실습을 위해 GitHub 웹사이트에서 빈 원격 저장소를 하나 생성한다.
 - README.md 파일은 생성하지 않는다.
 - git clone 실습 때 진행하던 디렉터리가 아니라, 이전에 진행하던 디렉터리에서 실습 진행

git remote

- 로컬 저장소와 원격 저장소를 연결하기
 - git remote add origin https://github.com/yh0921k/yh_github.git
 - 원격 저장소와 기존에 실습하던 로컬 저장소를 연결
 - origin : 저장소 별칭, 기본적으로 git 명령어만 아니면 사용자 임의로 정할 수 있으며 관례로 origin을 사용한다.
 - git remote -v
 - 원격 저장소와 연결되었는지 확인

```
kyh@DESKTOP-0UBU3AH MINGW64 ~/Desktop/LabSeminar/git_tutorial (master)
$ git remote add origin https://github.com/yh0921k/yh_github.git

kyh@DESKTOP-0UBU3AH MINGW64 ~/Desktop/LabSeminar/git_tutorial (master)
$ git remote -v
origin https://github.com/yh0921k/yh_github.git (fetch)
origin https://github.com/yh0921k/yh_github.git (push)
```

```
root@netsys:~/LabSeminar/git_tutorial# git remote add origin https://github.com/yh0921k/yh_github.git
root@netsys:~/LabSeminar/git_tutorial# git remote -v
origin https://github.com/yh0921k/yh_github.git (fetch)
origin https://github.com/yh0921k/yh_github.git (push)
```


git push

- 로컬 작업 내역을 원격 저장소에 올리기
 - git push 명령은 기본적으로 커밋들을 원격 저장소의 master branch에 업로드하고, 다양한 옵션을 통해서 특정 branch의 내용을 업데이트 하거나 tag를 push하는 등의 작업을 한다.
 - git push
 - 비어 있는 원격 저장소에 로컬 저장소의 내용을 push
 - 경고 메시지가 등장하며, 중간쯤에는 github에 로그인 하기 위한 사용자 이름과 비밀번호를 묻는다.
 - 하지만 로그인을 진행해도 거절당한다.
 - 이는 어느 원격 저장소로 발행할 것인지, 로컬의 어느 branch를 push할 것인지 명시하지 않았기 때문이다.
 - git push origin --all
 - 'git push' + '원격 저장소 별칭' + '로컬 branch 이름' 의 형태이다.
 - -all 옵션은 origin 저장소에 로컬의 모든 브랜치를 push하는 것이다.
 - Git은 원격 저장소에 로컬 저장소의 branch와 같은 이름의 branch가 있다면 해당 branch를 변경하고 없다면 새 branch를 저장소에 만든다.
 - 주의할 점은 같은 이름의 branch가 있는데, 서로의 내역이 다르다면 push를 거부한다.
 - 즉, 백지상태인 원격 저장소에 로컬 저장소에서 작업한 것을 push해야 하므로 README.md 파일을 생성해서는 안되는 것이다.

git push




- 로컬 작업 내역을 원격 저장소에 올리기
 - git push origin --all
 - 사용자 로그인 정보 입력

```
root@netsys:~/LabSeminar/git_tutorial# git push origin --all
Username for 'https://github.com': yh0921k
Password for 'https://yh0921k@github.com':
오브젝트 개수 세는 중 : 18, 완료 .
Delta compression using up to 8 threads.
오브젝트 압축하는 중 : 100% (12/12), 완료 .
오브젝트 쓰는 중 : 100% (18/18), 2.36 KiB | 0 bytes/s, 완료 .
Total 18 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/yh0921k/yh_github.git
 * [new branch]      hotfix -> hotfix
 * [new branch]      master -> master
```

- 이제, GitHub 웹 사이트에서 저장소에 로컬 저장소의 프로젝트가 push된 것을 확인할 수 있다.
 - hotfix branch까지 push 되어 있어야 한다.

git push

- 로컬 작업 내역을 원격 저장소에 올리기
 - 아래의 명령은 로컬에서 README.md 파일을 만들고 commit 한 뒤 원격 저장소로 push 하는 명령이다.
 - cat >> README.md
 - remote repository of git_tutorial
 - git add README.md
 - git commit -m "remote repository add a README.md"
 - git push origin master

yh0921k remote repository add a README.md		Latest commit 02cb9f1 Feb 5, 2018
 .gitignore	added '.gitignore' file	15 hours ago
 README.md	remote repository add a README.md	2 minutes ago
 hello.py	conflice resolved	14 hours ago

- hotfix branch에도 추가하고 싶다면?
 - 로컬 저장소에서 병합한 다음 git push origin hotfix 또는
 - git push origin master 명령을 실행하기 전에 먼저 로컬 저장소를 병합한 후 git push origin --all

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - 실제 협업 상황에서는 내가 로컬 저장소에서 작업하는 도중에 다른 협업자가 원격 저장소를 먼저 변경할 수 있다.
 - 이런 경우 Git은 push를 허용하지 않는다.
 - 로컬 저장소의 commit들을 원격 저장소와 맞춰야 한다.
 - 이 경우에 하는 것이 fetch이다.
 - fetch는 원격 저장소의 커밋들을 로컬 저장소로 가져온다.
 - 사용자는 로컬로 가져온 커밋들을 자신이 여태까지 한 로컬 저장소의 작업과 적절히 병합해서 원격 저장소에 제출해야 한다.
 - fetch & pull
 - pull 명령은 원격 저장소의 정보를 가져오면 자동으로 로컬 branch에 merge까지 수행한다.
 - 하지만 pull을 이용하여 fetch와 merge를 동시에 진행하면 어떤 내용이 병합되면서 바뀌게 되었는지 알 수 없다.
 - 물론 병합 시 어느 파일이 몇 줄 바뀌었고, 어떤 충돌이 발생했는지 표시해주지만, 프로젝트의 세세한 부분의 변화는 파악할 수 없다.
 - 때문에 pull을 이용한 원격 저장소의 commit 가져오기는 권고되지 않으며, 대신 fetch를 이용해 가져오고 로컬 저장소에서 이를 확인한 다음 수동으로 merge하는 방법을 권고한다.


git fetch & git pull



- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - 작업의 순서
 - GitHub 상에서 파일 수정
 - 로컬 저장소 내용 변경과 커밋
 - push 시도와 실패
 - fetch
 - merge
 - push
 - GitHub 상에서 확인
 - git remote -v
 - 로컬 저장소의 상태 확인

```
root@netsys:~/LabSeminar/git_tutorial# git remote -v
origin  https://github.com/yh0921k/yh_github.git (fetch)
origin  https://github.com/yh0921k/yh_github.git (push)
```

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - GitHub 상에서 hello.py 내용 변경

yh_github / hello.py  or [cancel](#)

 Edit file	 Preview changes
<pre>1 // For command line git tutorial 2 // GitHub modification 3 print("Hello World") 4 print("Branch Test") 5 print("Check point 1") 6 print("Check point 2") 7 print("hotfix check point")</pre>	

- commit message

Commit changes

hello.py modified on GitHub

Add an optional extended description

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - 로컬 상에서 hello.py 내용 변경

```
# for command line git tutorial
# Local repository modification
print("Hello World")
print("Branch Test")
print("Check point 1")
print("Check point 2")
print("hotfix check point")
```

- git commit -a -m "hello.py modified on Local repository"
- git status

```
root@netsys:~/LabSeminar/git_tutorial# git commit -a -m "hello.py modified on Local repository"
[master 023ca73] hello.py modified on Local repository
 1 file changed, 2 insertions(+)
root@netsys:~/LabSeminar/git_tutorial# git status
현재 브랜치 master
커밋할 사항 없음, 작업 폴더 깨끗함
```

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - 로컬 저장소의 작업을 원격 저장소로 push
 - git push origin master
 - push가 거절된다.
 - 원격 저장소와 로컬 저장소의 같은 브랜치가 다른 커밋을 가지고 있기 때문이다.
 - 힌트에서는 git pull 명령을 사용하라고 알려주지만, 권고되지 않는다.
 - fetch와 merge를 사용한다.

```
root@netsys:~/LabSeminar/git_tutorial# git push origin master
Username for 'https://github.com': yh0921k
Password for 'https://yh0921k@github.com':
To https://github.com/yh0921k/yh_github.git
 ! [rejected]        master -> master (fetch first)
error: 레퍼런스를 'https://github.com/yh0921k/yh_github.git'에 푸시하는데 실패했습니다
힌트: 리모트에 로컬에 없는 사항이 들어 있으므로 업데이트가
힌트: 거부되었습니다. 이 상황은 보통 또 다른 저장소에서 같은
힌트: 저장소로 푸시할 때 발생합니다. 푸시하기 전에
힌트: ('git pull ...' 등 명령으로) 리모트 변경 사항을 먼저
힌트: 포함해야 합니다.
힌트: 자세한 정보는 'git push --help'의 "Note about fast-forwards" 부분을
힌트: 참고하십시오.
```


git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - git fetch
 - fetch 명령을 실행해 원격 저장소의 commit 정보를 로컬 저장소에 가져온다.
 - git status

```
root@netsys:~/LabSeminar/git_tutorial# git fetch
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
오브젝트 묶음 푸는 중: 100% (3/3), 완료.
https://github.com/yh0921k/yh_github URL에서
02cb9f1..aa8dbf5 master -> origin/master
root@netsys:~/LabSeminar/git_tutorial# git status
현재 브랜치 master
커밋할 사항 없음, 작업 폴더 깨끗함
```

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - 원격 저장소의 master branch와 로컬 저장소의 master branch를 병합한 다음 다시 push해야 한다.
 - git branch -a
 - 현재 어떤 브랜치가 있는지 확인

```
root@netsys:~/LabSeminar/git_tutorial# git branch -a
hotfix
* master
remotes/origin/hotfix
remotes/origin/master
```

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - git merge origin/master
 - 로컬 저장소의 master branch에 원격 저장소의 origin/master branch를 병합한다.

```
root@netsys:~/LabSeminar/git_tutorial# git merge
FETCH HEAD      HEAD      ORIG_HEAD      hotfix      master      origin/hotfix  origin/master
root@netsys:~/LabSeminar/git_tutorial# git merge origin/master
자동 병합 : hello.py
충돌 ! (내용) : hello.py에 병합 충돌
자동 병합이 실패했습니다. 충돌을 바로잡고 결과물을 커밋하십시오.
```

- 같은 부분에서 다른 수정이 있으므로, 충돌 발생

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - git diff
 - 로컬 저장소의 branch와 원격 저장소의 branch 사이에 어떤 차이점이 있는지 확인
 - 만약 git pull 명령을 이용했다면 fetch와 merge를 자동으로 수행해버려서 어떤 변경 사항이 있는지 알기가 매우 어려워진다.

```
root@netsys:~/LabSeminar/git_tutorial# git diff
diff --cc hello.py
index 06cfe66,3d47c6a..0000000
--- a/hello.py
+++ b/hello.py
@@@ -1,5 -1,5 +1,10 @@@
++<<<<<< HEAD
+ # for command line git tutorial
+ # Local repository modification
++=====
+ // For command line git tutorial
+ // GitHub modification
++>>>>>> origin/master
+ print("Hello World")
+ print("Branch Test")
+ print("Check point 1")
```

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - 적절히 수정한다

```
<<<<<< HEAD
# for command line git tutorial
# Local repository modification
=====
// For command line git tutorial
// GitHub modification
>>>>>> origin/master
print("Hello World")
print("Branch Test")
print("Check point 1")
print("Check point 2")
print("hotfix check point")
```

```
# for command line git tutorial
# First : GitHub modification
# Second : Local repository modification
print("Hello World")
print("Branch Test")
print("Check point 1")
print("Check point 2")
print("hotfix check point")
```

git fetch & git pull


- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - `git commit -a -m "conflict resolved GitHub"`
 - `git status`
 - `git push origin master`

```
root@netsys:~/LabSeminar/git_tutorial# git commit -a -m "conflict resolved GitHub"
[master 241535d] conflict resolved GitHub
root@netsys:~/LabSeminar/git_tutorial# git status
현재 브랜치 master
커밋할 사항 없음, 작업 폴더 깨끗함
root@netsys:~/LabSeminar/git_tutorial# git push origin master
Username for 'https://github.com': yh0921k
Password for 'https://yh0921k@github.com':
오브젝트 개수 세는 중: 6, 완료.
Delta compression using up to 8 threads.
오브젝트 압축하는 중: 100% (6/6), 완료.
오브젝트 쓰는 중: 100% (6/6), 659 bytes | 0 bytes/s, 완료.
Total 6 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To https://github.com/yh0921k/yh_github.git
aa8dbf5..241535d master -> master
```




git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - GitHub 저장소 확인

Branch: master ▾ [yh_github](#) / hello.py Find file Copy path

 **yh0921k** conflict resolved GitHub 241535d Feb 5, 2018

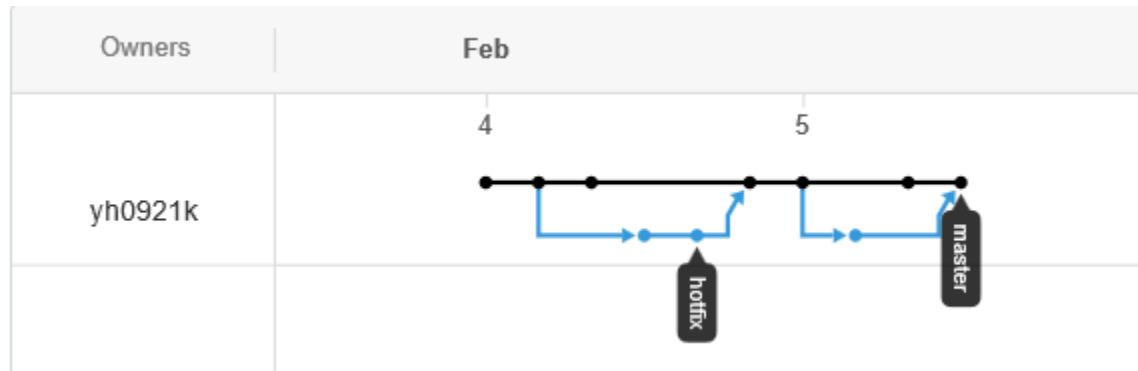
1 contributor

9 lines (8 sloc) | 219 Bytes Raw Blame History   

```
1 # for command line git tutorial
2 # First : GitHub modification
3 # Second : Local repository modification
4 print("Hello World")
5 print("Branch Test")
6 print("Check point 1")
7 print("Check point 2")
8 print("hotfix check point")
```

git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - Graph > Network
 - 아래와 같이 로컬 저장소에서 작업한 것이 반영되어 현재의 master branch를 만든 것을 확인할 수 있다.



git fetch & git pull

- 원격 저장소와 로컬 저장소의 간격 메꾸기
 - git pull 간단히 살펴보기
 - GitHub의 원격 저장소에서 출력 코드 하나를 추가한다.
 - 다음으로 로컬 저장소에서 git pull origin master 명령을 수행한다.
 - git commit, git merge, git status로 확인해보면 이미 작업할 것이 없다.

```
root@netsys:~/LabSeminar/git_tutorial# git pull origin master
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
오브젝트 묶음 푸는 중: 100% (3/3), 완료.
https://github.com/yh0921k/yh_github URL에서
* branch          master    -> FETCH_HEAD
  241535d..aa63053  master    -> origin/master
업데이트 중 241535d..aa63053
Fast-forward
 hello.py | 1 +
 1 file changed, 1 insertion(+)
root@netsys:~/LabSeminar/git_tutorial# git commit
현재 브랜치 master
커밋할 사항 없음, 작업 폴더 깨끗함
root@netsys:~/LabSeminar/git_tutorial# git merge
fatal: 현재 브랜치에 대한 리모트가 없습니다.
root@netsys:~/LabSeminar/git_tutorial# git status
현재 브랜치 master
커밋할 사항 없음, 작업 폴더 깨끗함
```