

3/21/2019

# GIT / GITHUB

이현아

1

# CONTENTS

---

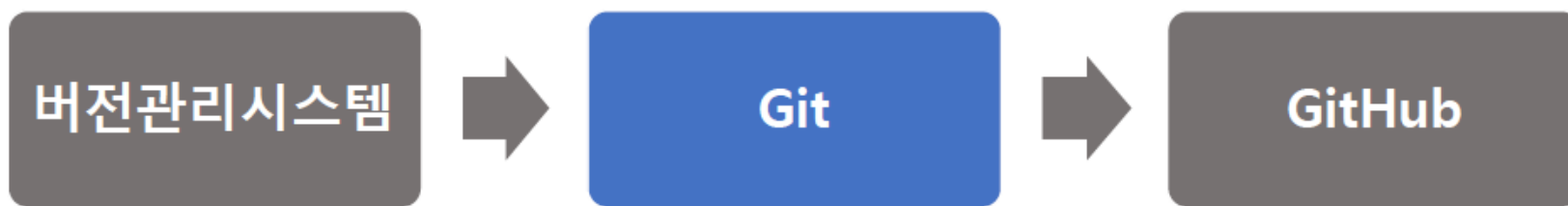
1. Git / GitHub 란?
2. Git - 로컬저장소
3. GitHub - 원격저장소
4. 로컬 & 원격저장소

3/21/2019

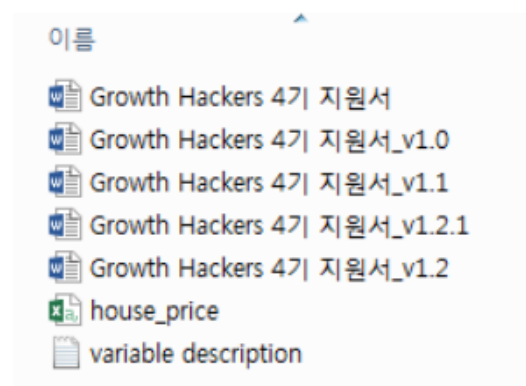
# 1. GIT / GITHUB란?

3

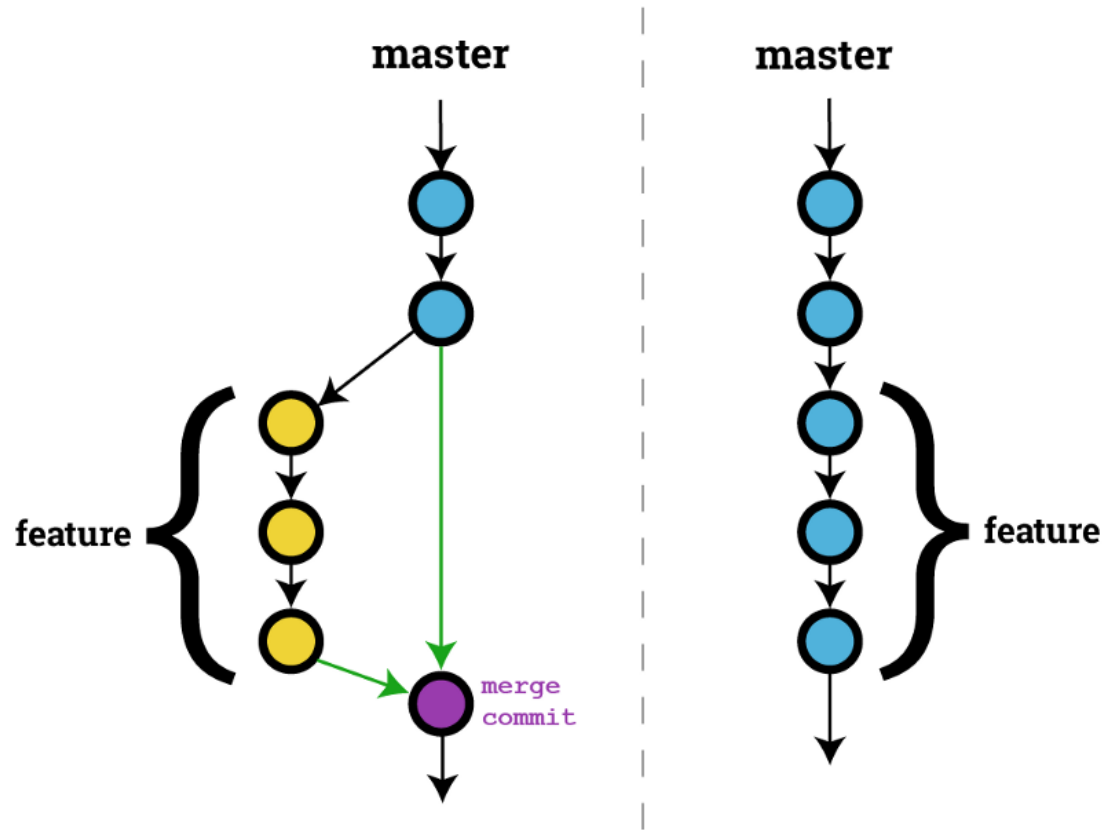
# Git / GitHub란?



- 다양한 버전관리 시스템 중 하나
  - Git, Subversion, Mercurial 등
- 분산형 버전관리 시스템: 개발자가 중앙 서버에 접속하지 않은 상태에서 코드 작업을 할 수 있음
- 가장 최신의, 그리고 가장 상용화된 버전관리 시스템



# Git - Branch의 개념



Git의 중요한 기능  
★ 버전관리

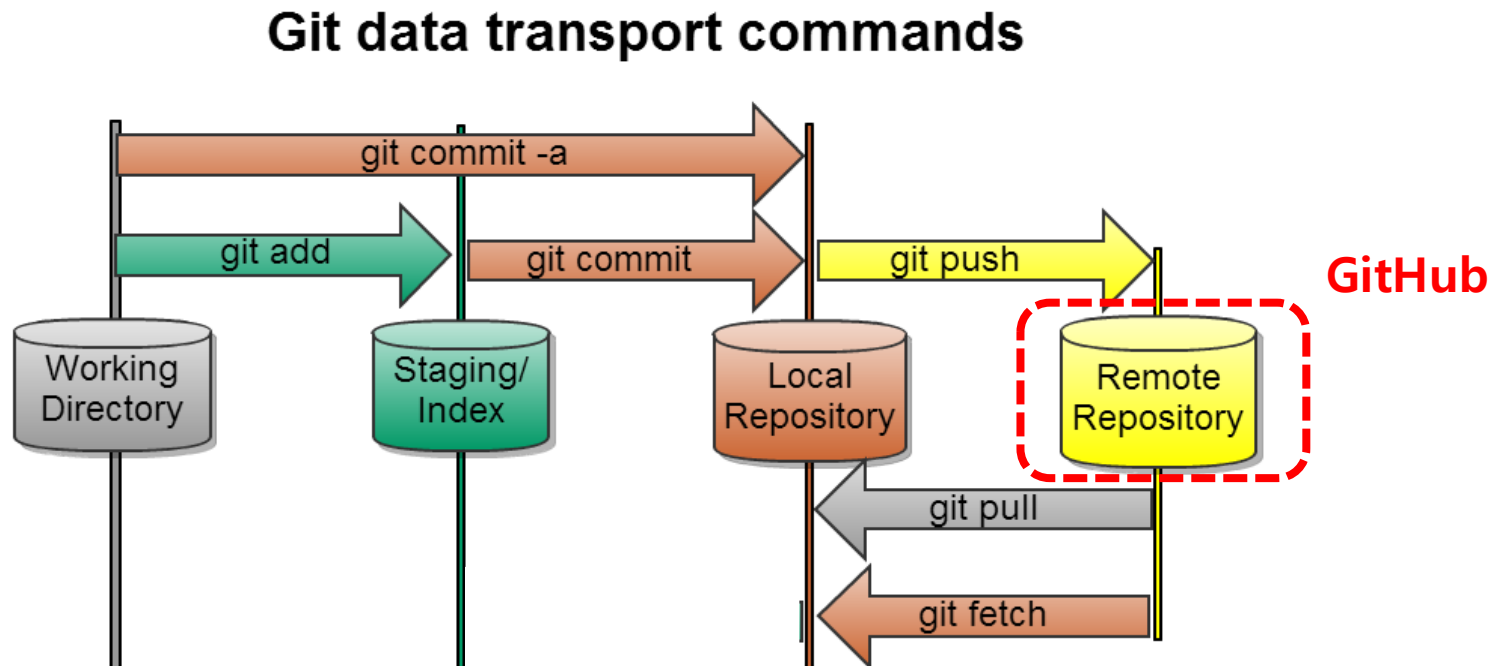
# Git / GitHub란?



- 버전관리 시스템인 Git을 이용한 프로젝트들을 위한 **원격저장소**를 제공하는 웹호스팅 서비스
- 원격저장소
  - 파일을 작성, 변경하면 변경사항들이 저장소에 차곡차곡 저장됨
  - 로컬이 아닌 원격으로 공유하여 협업 용이
- 가장 인기있는 오픈소스 코드 저장소(오픈소스는 free, 미공개는 유료)
- +소셜 기능 제공(피드백, 팔로우, 스타 등)



# Git – 작업 흐름



Git의 중요한 기능  
★ 협업

3/21/2019

## 2. GIT 로컬저장소

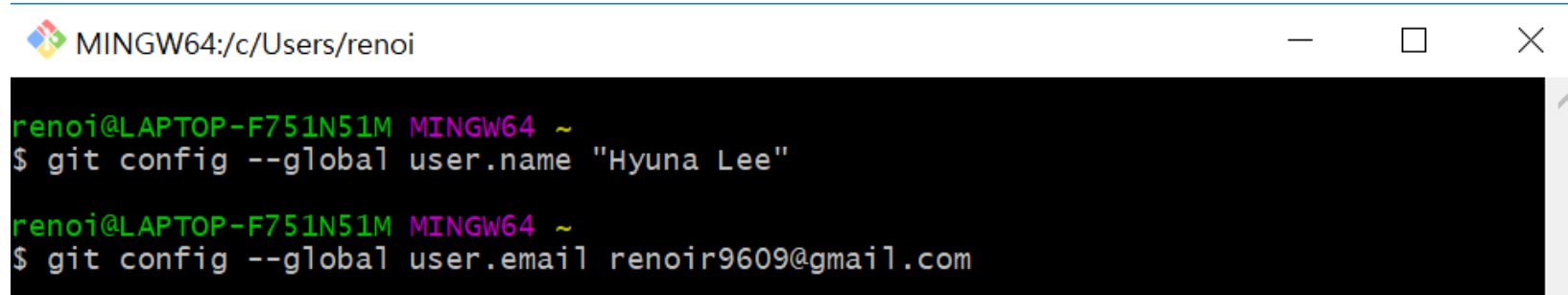
(Mission1) 로컬저장소를 만들고, 이 로컬저장소에 파일을 추가하자!

8



# Git 사용 준비 – Git bash 열기

- 사용자 이름 및 이메일 입력
  - `git config --global user.name "사용자이름"`
  - `git config --global user.email 이메일주소`



A screenshot of a Windows command prompt window titled "MINGW64:/c/Users/renoi". The window shows two lines of commands being entered and executed. The first line is `renoi@LAPTOP-F751N51M MINGW64 ~` followed by `$ git config --global user.name "Hyuna Lee"`. The second line is `renoi@LAPTOP-F751N51M MINGW64 ~` followed by `$ git config --global user.email renoir9609@gmail.com`. The prompt is green, and the commands are white on a black background.

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ git config --global user.name "Hyuna Lee"  
  
renoi@LAPTOP-F751N51M MINGW64 ~  
$ git config --global user.email renoir9609@gmail.com
```

# 로컬저장소 생성

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ mkdir git_tutorial
```

mkdir  
: 디렉토리 생성

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ cd git_tutorial
```

cd  
: 디렉토리로 이동

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial  
$ git init  
Initialized empty Git repository in C:/Users/renoi/git_tutorial/.git/  
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ |
```

git init  
: 실행위치를  
저장소로 만들기

# 로컬저장소 생성

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ mkdir git_tutorial
```

mkdir  
: 디렉토리 생성

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ cd git_tutorial
```

cd  
: 디렉토리로 이동

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial  
$ git init  
Initialized empty Git repository in C:/Users/renoi/git_tutorial/.git/  
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ |
```

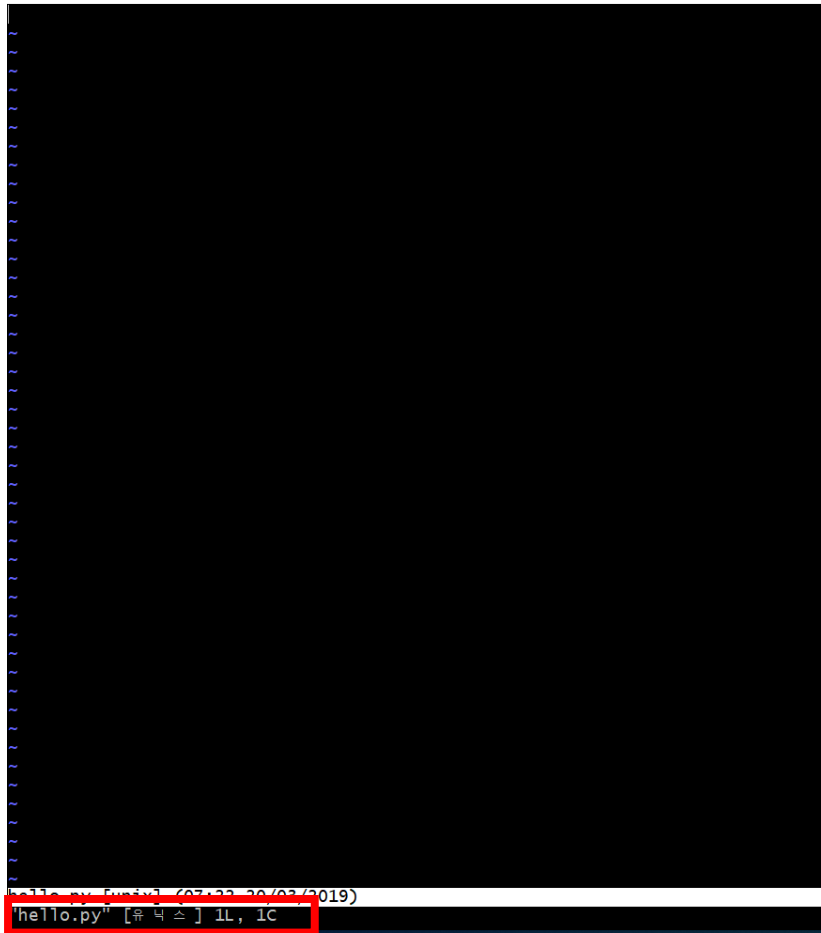
git init  
: 실행위치를  
저장소로 만들기

# 파일 생성 (vim)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ vim hello.py|
```

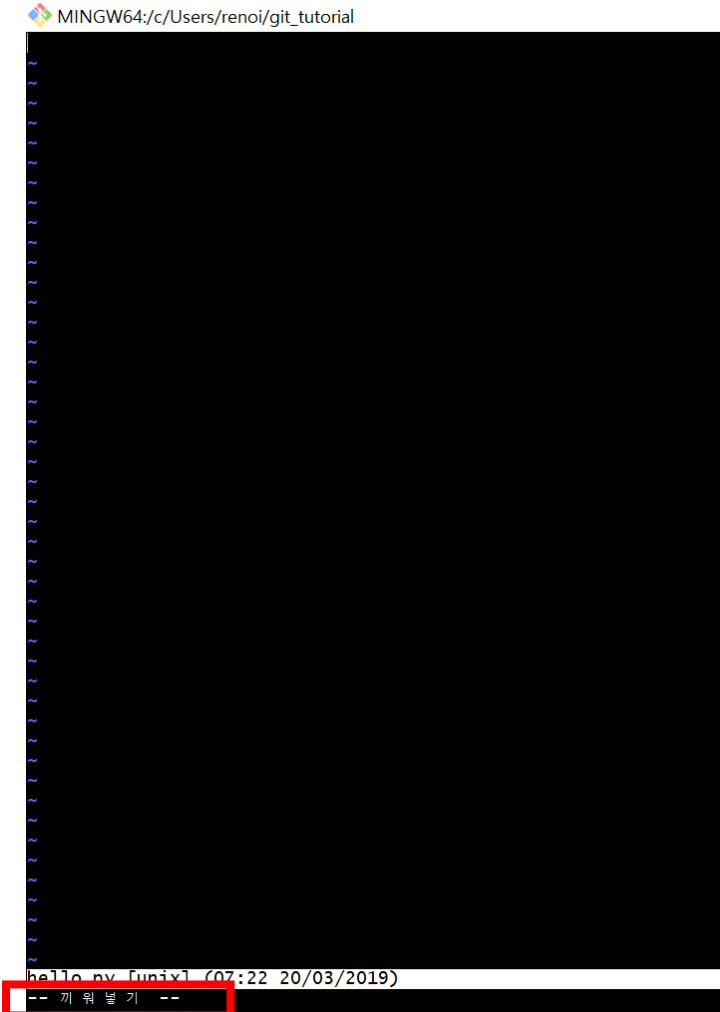
# 파일 생성 (vim)

MINGW64:/c:/Users/renoi/git\_tutorial



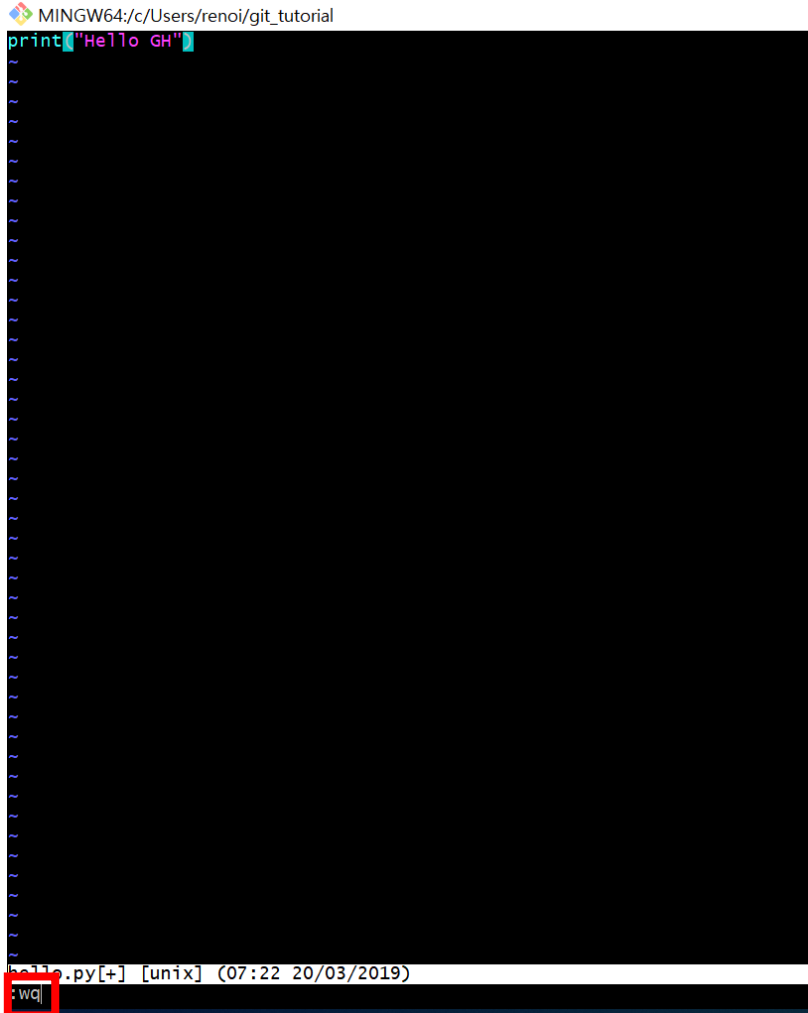
- **Vim이란?**
  - 리눅스나 Unix에서 사용되는 텍스트 편집기
- **Vim 사용법**
  - Vim에서 작성을 시작하려면 키보드의 [i]키를 눌러 작성모드로 바꾸기
  - 작성 후에 [ESC]키를 누르면 일반모드로 돌아옴
  - :를 입력하면 명령모드로 바뀌고, 명령모드에서 wq를 입력하면 저장(w) 후 종료(q)됨

# 파일 생성 (vim)



- Vim이란?
  - 리눅스나 Unix에서 사용되는 텍스트 편집기
- Vim 사용법
  - Vim에서 작성을 시작하려면 키보드의 [i]키를 눌러 작성모드로 바꾸기
  - 작성 후에 [ESC]키를 누르면 일반모드로 돌아옴
  - :를 입력하면 명령모드로 바뀌고, 명령모드에서 wq를 입력하면 저장(w) 후 종료(q)됨

# 파일 생성 (vim)



```
MINGW64:/c/Users/renoi/git_tutorial
print('Hello GH')
:wq
```

- **Vim이란?**
  - 리눅스나 Unix에서 사용되는 텍스트 편집기
- **Vim 사용법**
  - Vim에서 작성을 시작하려면 키보드의 [i]키를 눌러 작성모드로 바꾸기
  - 작성 후에 [ESC]키를 누르면 일반모드로 돌아옴
  - :를 입력하면 명령모드로 바뀌고, 명령모드에서 wq를 입력하면 저장(w) 후 종료(q)됨

# 파일 내용 확인 (cat)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ cat hello.py  
print("Hello GH")
```

(참고) 내 컴퓨터의 디렉토리에 직접 .py 파일을 만들어도 됨



# 저장소 상태 확인 (git status)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git status
On branch master

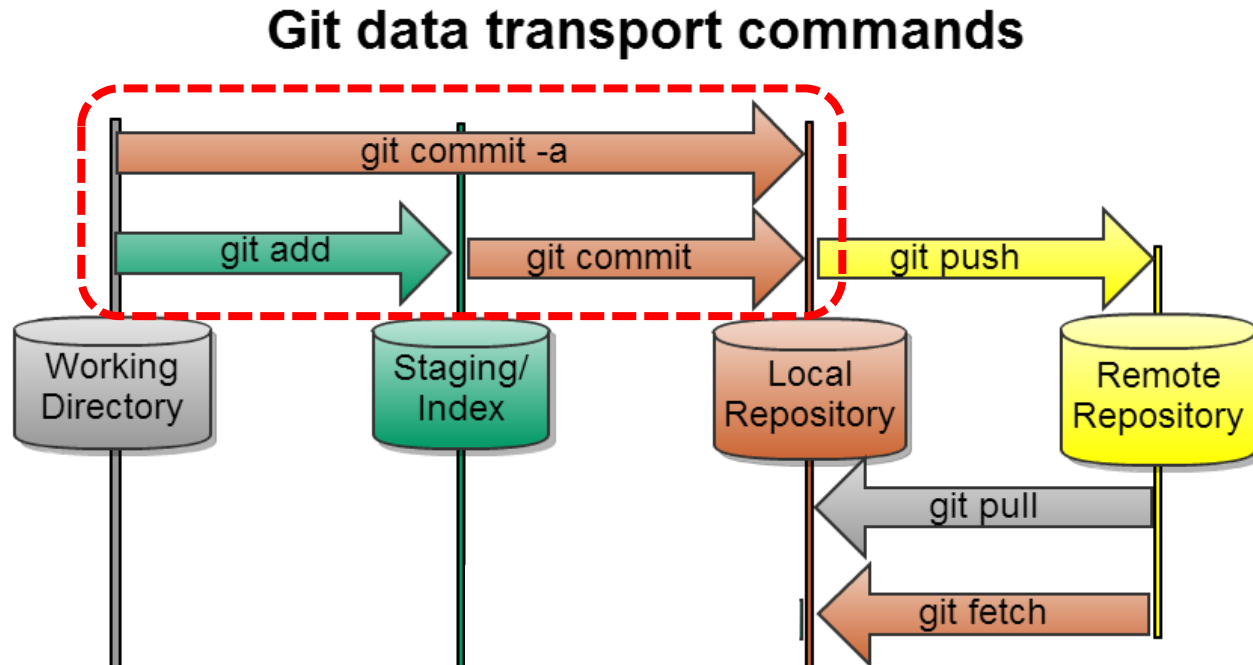
No commits yet

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

        hello.py

nothing added to commit but untracked files present (use "git add" to track)
```

# 로컬저장소의 기본 작업 과정



# 저장소에 파일 추가 (git add)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git add hello.py
```

git add

: 해당 파일을 git이 추적할 수 있게 저장소에 추가하기

# 저장소 상태 다시 확인 (git status)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   hello.py
```

# 커밋 (git commit)

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ git commit|
```

git commit

: 변경된 파일을 저장소에 제출하기

# 커밋 메시지 입력 (vim)

[illegible]

- **Vim이란?**
  - 리눅스나 Unix에서 사용되는 텍스트 편집기
- **Vim 사용법**
  - Vim에서 작성을 시작하려면 키보드의 [i]키를 눌러 작성모드로 바꾸기
  - 작성 후에 [ESC]키를 누르면 일반모드로 돌아옴
  - :를 입력하면 명령모드로 바뀌고, 명령모드에서 wq를 입력하면 저장(w) 후 종료(q)됨

# Mission1 성공!

(Mission1) 로컬저장소를 만들고, 이 로컬저장소에 파일을 추가하자!

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git commit
[master (root-commit) 7da5211] create "hello world" program
1 file changed, 1 insertion(+)
create mode 100644 hello.py
```

3/21/2019

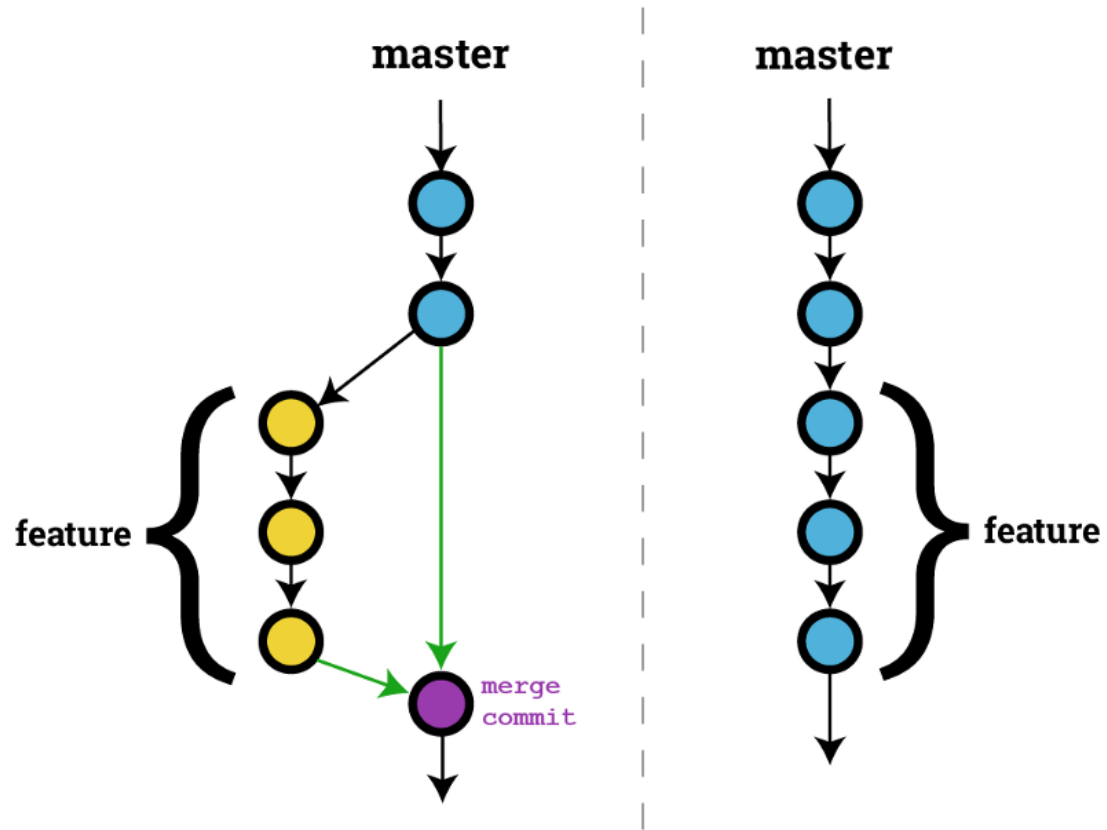
## 2. GIT 로컬저장소

(Mission2) 새로운 branch를 만들고,  
이 branch에서 파일을 수정해서 다시 master branch에 반영시키자!

24



# Git - Branch의 개념



# Branch 확인하기 (git branch)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git branch  
* master
```

# Branch 만들기 (git branch 이름)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git branch hotfix
```

(참고) hotfix

: 프로그램 사용 중 발견된 버그(bug)의 수정이나 취약점 보완, 성능 향상을 위해 긴급 배포되는 응급 패치 프로그램.

# Branch 다시 확인하기 (git branch)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git branch
  hotfix
* master
```

# 다른 branch로 이동 (git checkout)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git checkout hotfix
Switched to branch 'hotfix'

renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (hotfix)
$ |
```

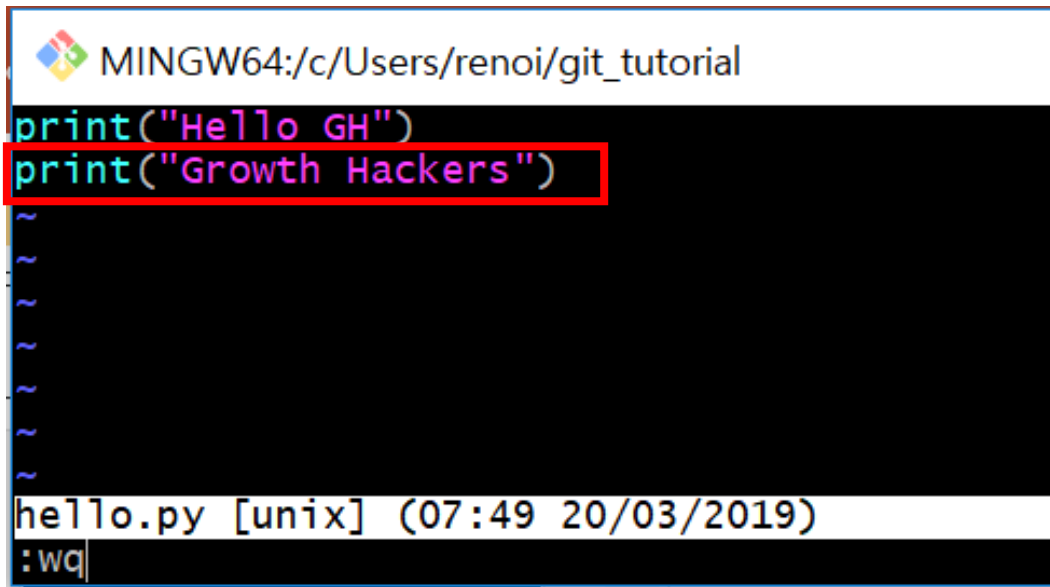
# 파일 확인 (ls)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (hotfix)
$ ls
hello.py
```

# 파일 수정 (vim)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (hotfix)  
$ vim hello.py|
```

# 파일 수정 (vim)



```
MINGW64:/c/Users/renoi/git_tutorial
print("Hello GH")
print("Growth Hackers")
~
~
~
~
~
~
hello.py [unix] (07:49 20/03/2019)
:wq
```

## • Vim 사용법

- Vim에서 작성을 시작하려면 키보드의 [i]키를 눌러 작성모드로 바꾸기
- 작성 후에 [ESC]키를 누르면 일반모드로 돌아옴
- :를 입력하면 명령모드로 바뀌고, 명령모드에서 wq를 입력하면 저장(w) 후 종료(q)됨



# 파일내용 확인 (cat)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (hotfix)
$ cat hello.py
print("Hello GH")
print("Growth Hackers")
```

# 저장소 상태 확인 (git status)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (hotfix)
$ git status
On branch hotfix
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   hello.py

no changes added to commit (use "git add" and/or "git commit -a")
```

# 커밋 (git commit -a)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (hotfix)  
$ git commit -a|
```

git commit -a

: 변경된 저장소 파일을 모두 저장소에 제출(커밋)

# 커밋 메시지 작성 (vim)

```
MINGW64:/c/Users/renoi/git_tutorial
added output "Growth Hackers"
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch hotfix
# Changes to be committed:
#   modified:   hello.py
#
~
<rs/renoi/git_tutorial/.git/COMMIT_EDITMSG[+] [unix] (07:54 20/03/2019):
:wq|
```

## • Vim 사용법

- Vim에서 작성을 시작하려면 키보드의 [i] 키를 눌러 작성모드로 바꾸기
- 작성 후에 [ESC]키를 누르면 일반모드로 돌아옴
- :를 입력하면 명령모드로 바뀌고, 명령모드에서 wq를 입력하면 저장(w) 후 종료(q) 됨

# 다른 branch로 이동 (git checkout)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (hotfix)  
$ git checkout master  
Switched to branch 'master'
```

# 저장소 파일 상태 확인 (ls / cat)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ ls
hello.py

renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ cat hello.py
print("Hello GH")
```

hotfix 브랜치에서 수정한 내용이 master 브랜치에는 반영되지 않음!  
➔ 반영시키고 싶다면?

# Branch 병합 (git merge)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git merge hotfix
Updating 7da5211..443df5b
Fast-forward
 hello.py | 1 +
 1 file changed, 1 insertion(+)
```

(주의) master 브랜치에서 "git merge hotfix"를 하는 것과  
hotfix 브랜치에서 "git merge master"를 하는 것은 다르다  
→ 현재 위치한 브랜치가 수정하고 싶은 브랜치여야 함!

# Mission2 성공!

- (Mission2) 새로운 branch를 만들고,  
이 branch에서 파일을 수정해서 다시 master branch에 반영시키자!

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ ls
hello.py

renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ cat hello.py
print("Hello GH")
print("Growth Hackers")
```



# 커밋 내역 확인하기 (git log)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git log
commit 4ccb158870338a77e1715e36325c464235a22c9f (HEAD -> master)
Merge: e96095d e93f43a
Author: Hyuna Lee <renoir9609@gmail.com>
Date:   Wed Mar 20 23:40:25 2019 +0900

    conflict resolved GitHub

commit e96095dbfcf97d4a4e7303528c98e6ba53c8aef9
Author: Hyuna Lee <renoir9609@gmail.com>
Date:   Wed Mar 20 23:34:18 2019 +0900

    hello.py modified on Local repository

commit e93f43a0dce56baa1179034a12d18d0317eb849b
Author: Hyuna Lee <43376842+hysophie@users.noreply.github.com>
Date:   Wed Mar 20 23:33:46 2019 +0900

    hello.py modified on GitHub
```

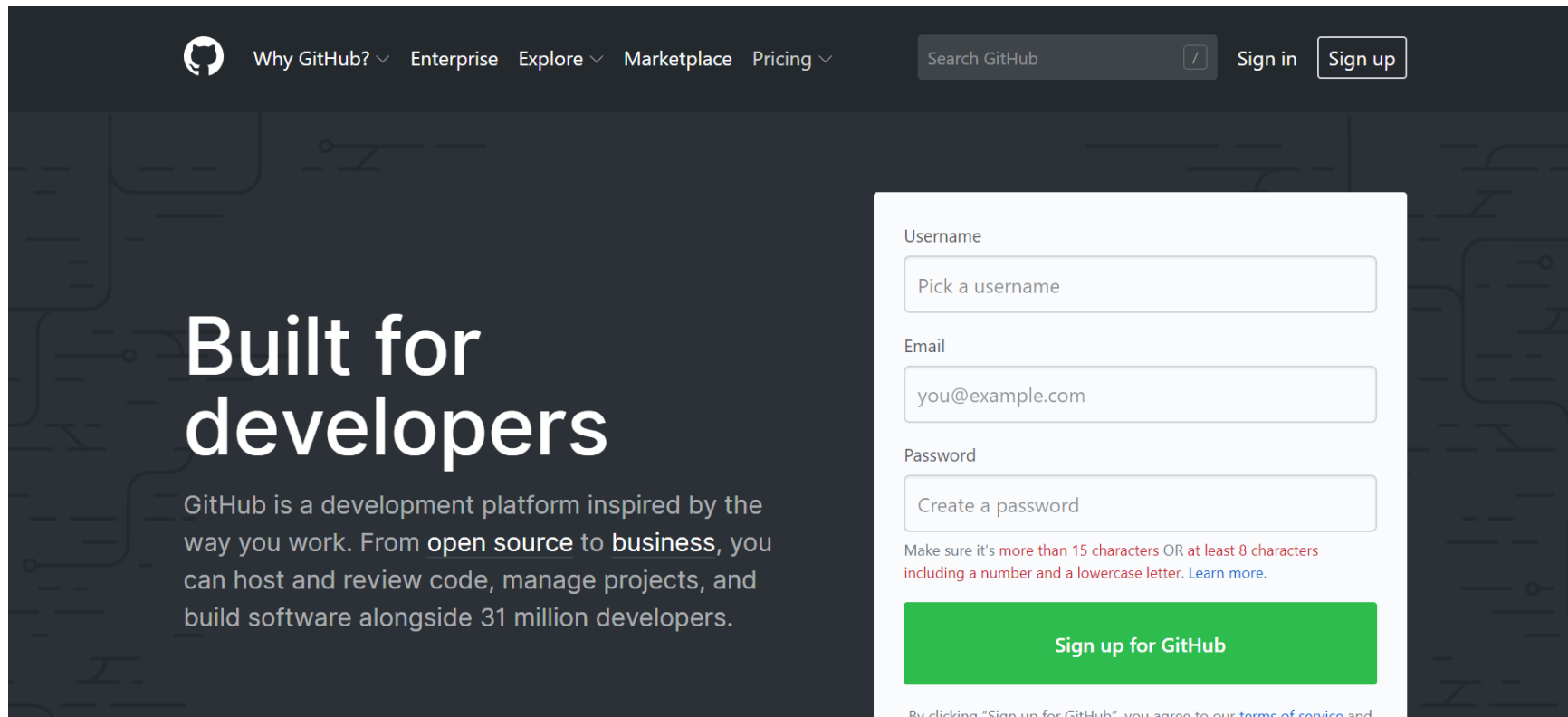
3/21/2019

# 3. GITHUB - 원격저장소

(Mission3) GitHub에 원격저장소를 만들고, 원격저장소를 로컬저장소에 가져오자!

42

# GitHub repository(저장소) 만들기



The image shows the GitHub sign-up page. On the left, there's a dark section with the GitHub logo and the text "Built for developers". Below this, it says "GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 31 million developers." On the right, there's a white sign-up form. The form has three input fields: "Username" with the placeholder "Pick a username", "Email" with the placeholder "you@example.com", and "Password" with the placeholder "Create a password". Below the password field, there's a note: "Make sure it's more than 15 characters OR at least 8 characters including a number and a lowercase letter. Learn more." At the bottom of the form is a green button that says "Sign up for GitHub". Above the form, there's a navigation bar with links: "Why GitHub?", "Enterprise", "Explore", "Marketplace", "Pricing", a search bar, and "Sign in" and "Sign up" buttons.

Why GitHub? ▾ Enterprise Explore ▾ Marketplace Pricing ▾ Search GitHub / Sign in Sign up

## Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 31 million developers.

Username  
Pick a username

Email  
you@example.com

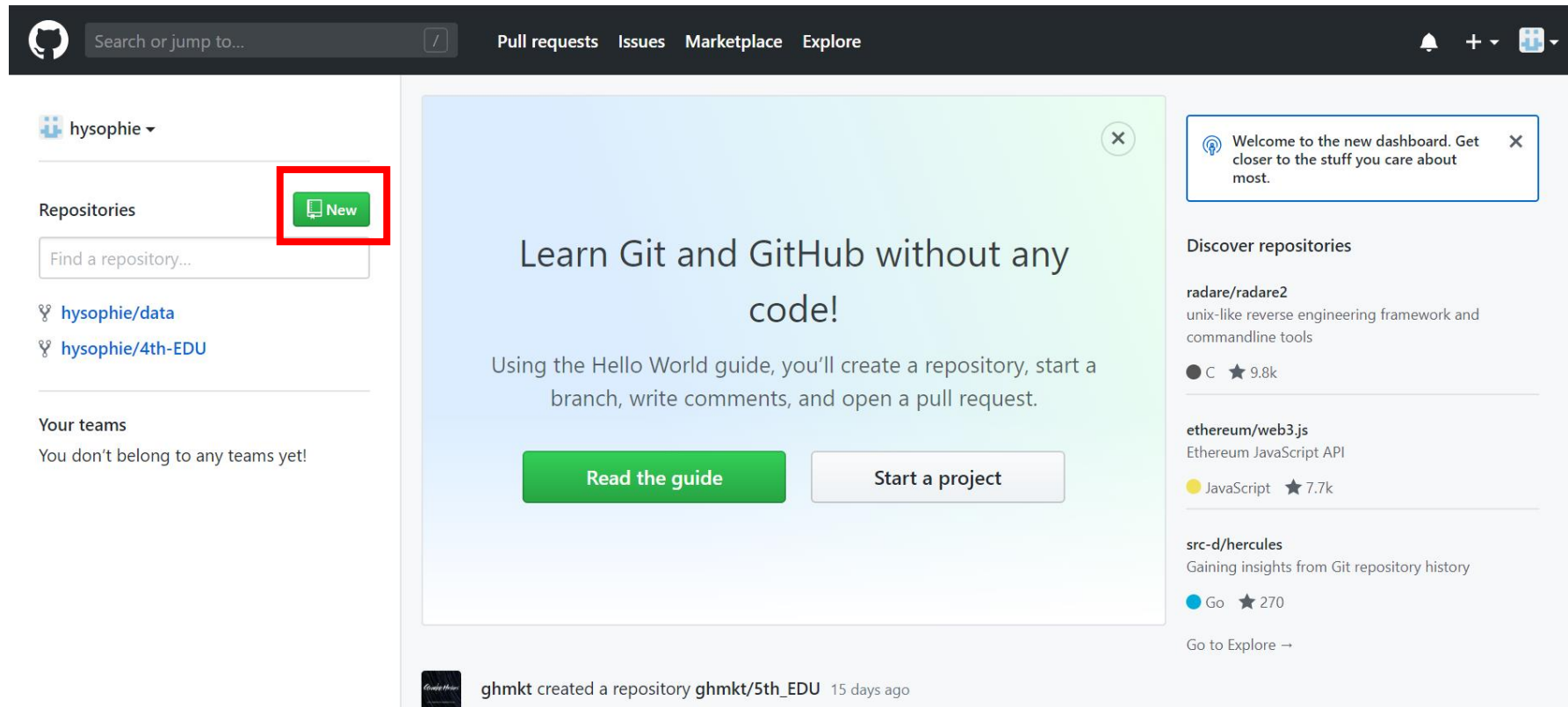
Password  
Create a password

Make sure it's more than 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

**Sign up for GitHub**

By clicking "Sign up for GitHub" you agree to our [terms of service](#) and

# GitHub repository 만들기




# GitHub repository 만들기


## Create a new repository

A repository contains all project files, including the revision history.

Owner

Repository name \*

 hysophie ▾

/ study 

Great repository names are short and memorable. Need inspiration? How about [cautious-fiesta](#)?

Description (optional)

☒  **Public**

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


☐  **Private**

You choose who can see and commit to this repository.

☒ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾ 

Create repository

# GitHub repository 만들기 (완성~)

The screenshot shows the GitHub interface for a new repository named 'study' by user 'hysophie'. At the top, the repository name and user are displayed, along with buttons for 'Watch', 'Star', and 'Fork', each with a count of 0. Below this is a navigation bar with tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The 'Code' tab is selected. A message states 'No description, website, or topics provided.' with an 'Edit' button. Below this, statistics show '1 commit', '1 branch', '0 releases', and '1 contributor'. A row of buttons includes 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find File', and 'Clone or download'. The commit history shows 'hysophie Initial commit' as the latest commit. Below the history, a file named 'README.md' is listed as the 'Initial commit'. The file content area shows the text 'study'.

hysophie / study

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

No description, website, or topics provided. Edit

Manage topics

1 commit 1 branch 0 releases 1 contributor

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

hysophie Initial commit Latest commit 3821284 just now

README.md Initial commit just now

README.md

study

# 원격저장소를 로컬저장소로 복사 (git clone)

The screenshot shows the GitHub interface for a repository named 'study' by user 'hysophie'. The repository has 1 commit, 1 branch, 0 releases, and 1 contributor. The 'Clone or download' button is highlighted with a red box. A dropdown menu is open, showing the 'Clone with HTTPS' option, which is also highlighted with a red box. The clone command is displayed as 'https://github.com/hysophie/study.git'. Below the clone command, there are buttons for 'Open in Desktop' and 'Download ZIP'.

hysophie / study

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

No description, website, or topics provided. Edit

Manage topics

1 commit 1 branch 0 releases 1 contributor

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

hysophie Initial commit

README.md Initial commit

README.md

study

Clone with HTTPS Use SSH

Use Git or checkout with SVN using the web URL

https://github.com/hysophie/study.git

Open in Desktop Download ZIP

# 원격저장소를 로컬저장소로 복사 (git clone)

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ mkdir github_tutorial
```

```
renoi@LAPTOP-F751N51M MINGW64 ~  
$ cd github_tutorial
```

**mkdir**

: 디렉토리 생성

**cd**

: 디렉토리로 이동



# 원격저장소를 로컬저장소로 복사 (git clone)

```
renoi@LAPTOP-F751N51M MINGW64 ~/github_tutorial  
$ git clone https://github.com/hysophie/study.git
```

## **git clone**

: 원격저장소의 모든 내용을 로컬저장소로 복사하기

# Mission3 성공!

- (Mission3) GitHub에 원격저장소를 만들고, 원격저장소를 로컬저장소에 가져오자!

```
renoi@LAPTOP-F751N51M MINGW64 ~/github_tutorial
$ cd study

renoi@LAPTOP-F751N51M MINGW64 ~/github_tutorial/study (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

3/21/2019

# 3. GITHUB - 원격저장소

(Mission4) GitHub에 원격저장소를 만들고, 원격저장소와 로컬저장소를 연결하자!


51

# Github repository 만들기

## Create a new repository

A repository contains all project files, including the revision history.

Owner

 hysophie ▾

Repository name \*

command\_hello ✓

Great repository names are short and memorable. Need inspiration? How about **fictional-invention**?

Description (optional)

☒  **Public**

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

☐  **Private**

You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

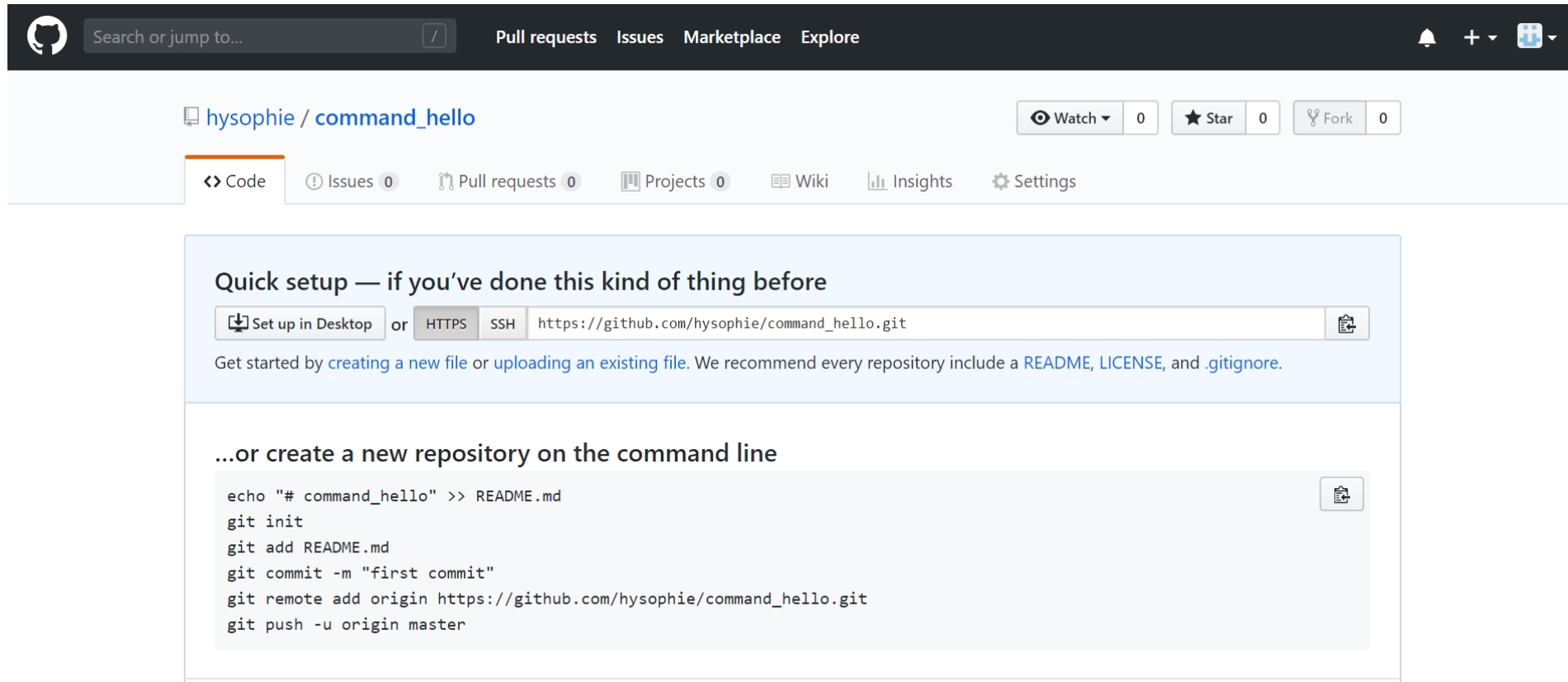
Add .gitignore: **None** ▾

Add a license: **None** ▾



Create repository

# Github repository 만들기



The screenshot shows the GitHub interface for a repository named 'command\_hello' by user 'hysophie'. The repository has 0 stars, 0 forks, and 0 watches. The 'Code' tab is selected, showing options to clone the repository via Desktop, HTTPS, or SSH. The SSH URL is 'https://github.com/hysophie/command\_hello.git'. Below the cloning options, there is a section titled 'Quick setup — if you've done this kind of thing before' with instructions to create a new file or upload an existing file. Further down, there is a section titled '...or create a new repository on the command line' with a list of terminal commands to initialize a new repository and push it to GitHub.

hysophie / command\_hello

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

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

Set up in Desktop or HTTPS SSH https://github.com/hysophie/command\_hello.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 "# command_hello" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/hysophie/command_hello.git
git push -u origin master
```

# 다른 로컬저장소로 이동하기

```
renoi@LAPTOP-F751N51M MINGW64 ~/github_tutorial/study (master)
$ cd ..

renoi@LAPTOP-F751N51M MINGW64 ~/github_tutorial
$ cd ..

renoi@LAPTOP-F751N51M MINGW64 ~
$ cd git_tutorial
```

**cd**

: 디렉토리로 이동

**cd ..**

: 한 단계 위의 디렉토리로 이동

# 로컬저장소를 원격저장소에 연결 (git remote)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git remote add origin https://github.com/hysophie/command_hello.git
```

`git remote add` 원격저장소별칭 `https://github.com/사용자이름/원격저장소이름.git`

➔ 저장소별칭은 어떤 이름이라도 상관 없으나, 관례로 origin을 사용하는 경우 많음.

# Mission4 성공!

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git remote -v
origin https://github.com/hysophie/command_hello.git (fetch)
origin https://github.com/hysophie/command_hello.git (push)
```

**git remote -v**

: 어떤 원격저장소와 연결 되어 있는지 확인



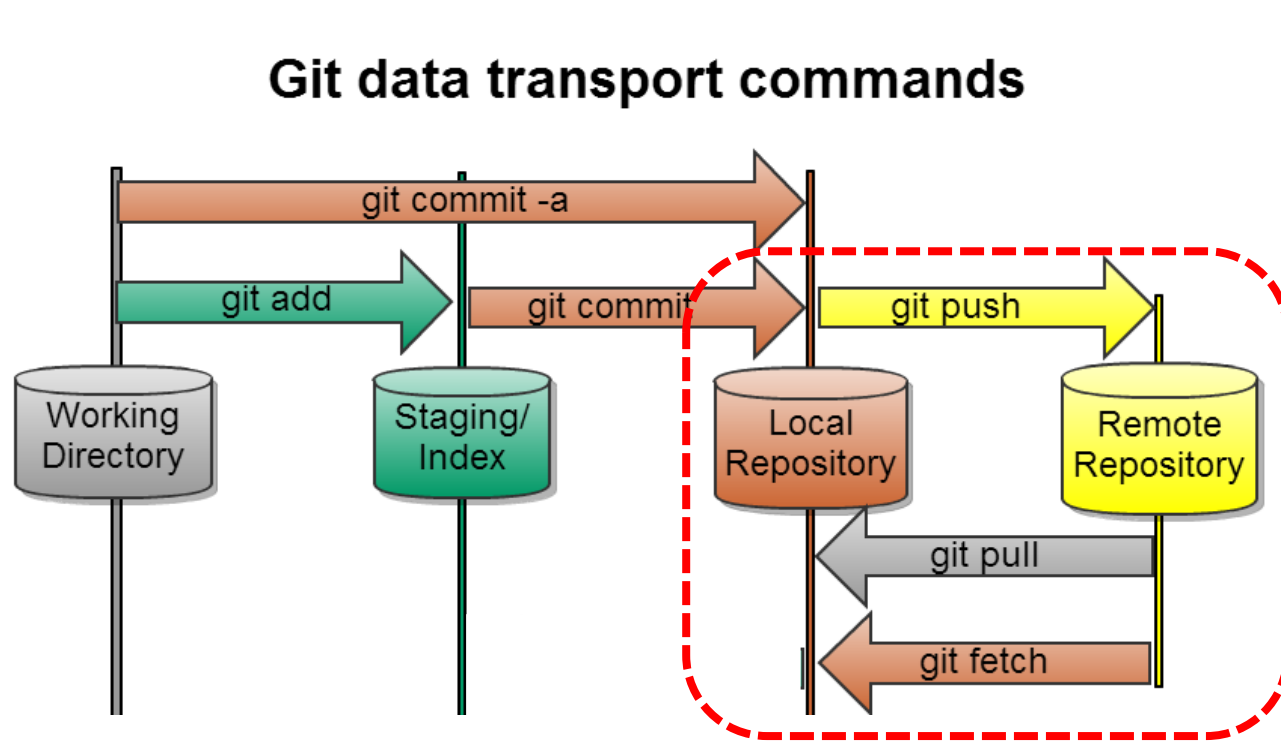
3/21/2019

## 3. 원격저장소 & 로컬저장소

(Mission5) 로컬저장소의 변경사항을 원격저장소로 공유해보자!

57

# 원격&로컬저장소 간의 변경사항 공유



Git의 중요한 기능  
★ 협업

# 로컬 작업내역을 원격 저장소에 올리기 (git push)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git push origin --all
```

**git push 원격저장소별칭 로컬브랜치이름**

: 로컬브랜치의 내용 or 수정사항을 현재 연결된 원격저장소에 보내는 것

➔ 로컬브랜치이름에 "--all"을 쓰는 것은 로컬의 "모든 브랜치"를 의미

# 로컬 작업내역을 원격 저장소에 올리기 (git push)

GitHub Login



**GitHub**  
Login



Login



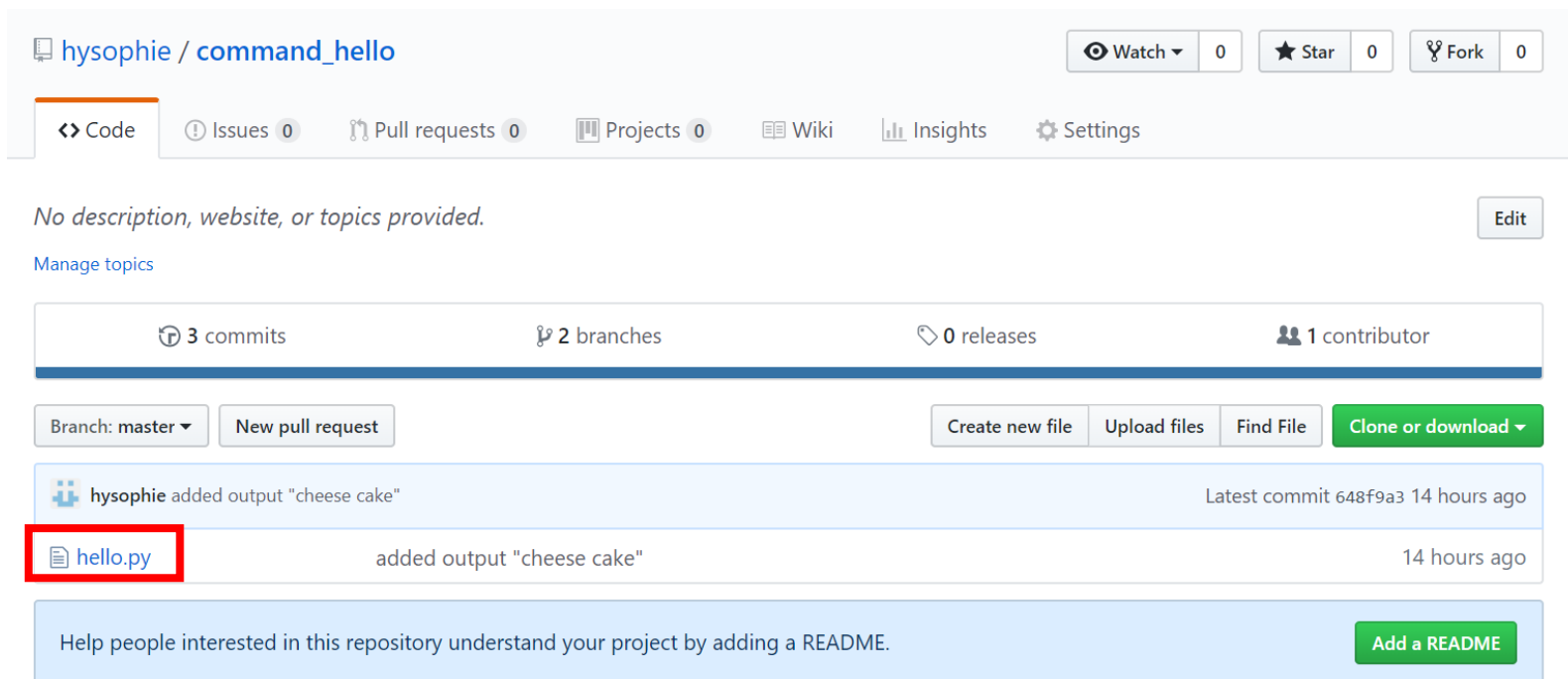
Cancel

Don't have an account? [Sign up](#)

[Forgot your password?](#)

# Mission5 성공!

- (Mission5) 로컬저장소의 변경사항을 원격저장소로 공유해보자! (협업)



# Mission5 완료!

hysophie / command\_hello

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Overview Yours Active Stale All branches Search branches...

**Default branch**

master	Updated 14 hours ago by hysophie	Default	Change default branch
--------	----------------------------------	---------	-----------------------

**Your branches**

hotfix	Updated 14 hours ago by hysophie	1   0	New pull request
--------	----------------------------------	-------	------------------

**Active branches**

hotfix	Updated 14 hours ago by hysophie	1   0	New pull request
--------	----------------------------------	-------	------------------

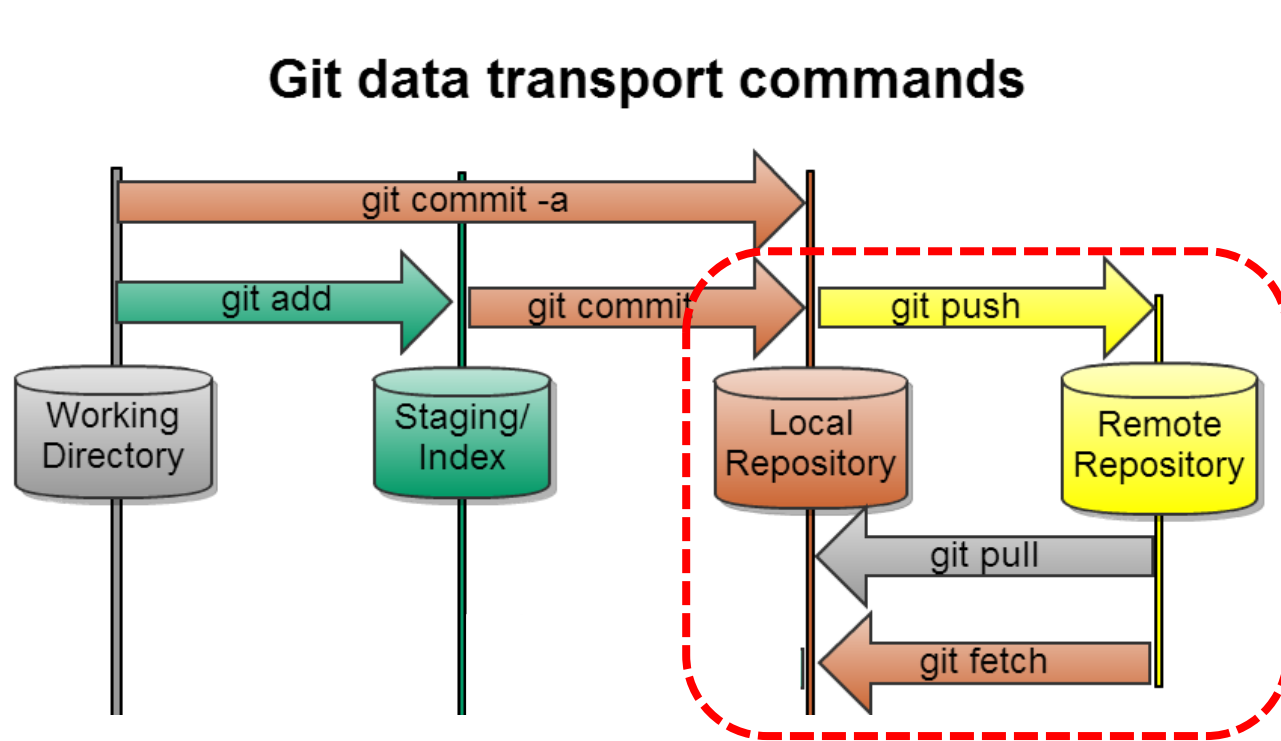
3/21/2019

## 3. 원격저장소 & 로컬저장소

(Mission6) 원격저장소의 변경사항을 로컬저장소에 가져오자!

63

# 원격&로컬저장소 간의 변경사항 공유



Git의 중요한 기능  
★ 협업



# git pull vs. git fetch

**Git pull = git fetch + git merge**

- ➔ Git fetch는 원격저장소의 커밋들을 로컬저장소로 가져오는 것
- ➔ Git pull은 원격저장소의 커밋들을 로컬저장소로 가져오는(git fetch) 동시에, 로컬 브랜치에 병합(git merge)까지 시키는 것

**Git pull의 장단점**

- ➔ 장점: git fetch와 merge를 한꺼번에 해주는 편리함 (그래서 많이 사용됨)
- ➔ 단점: fetch와 merge를 함께 하다 보니 변경사항을 세세하게 파악하기 어렵다

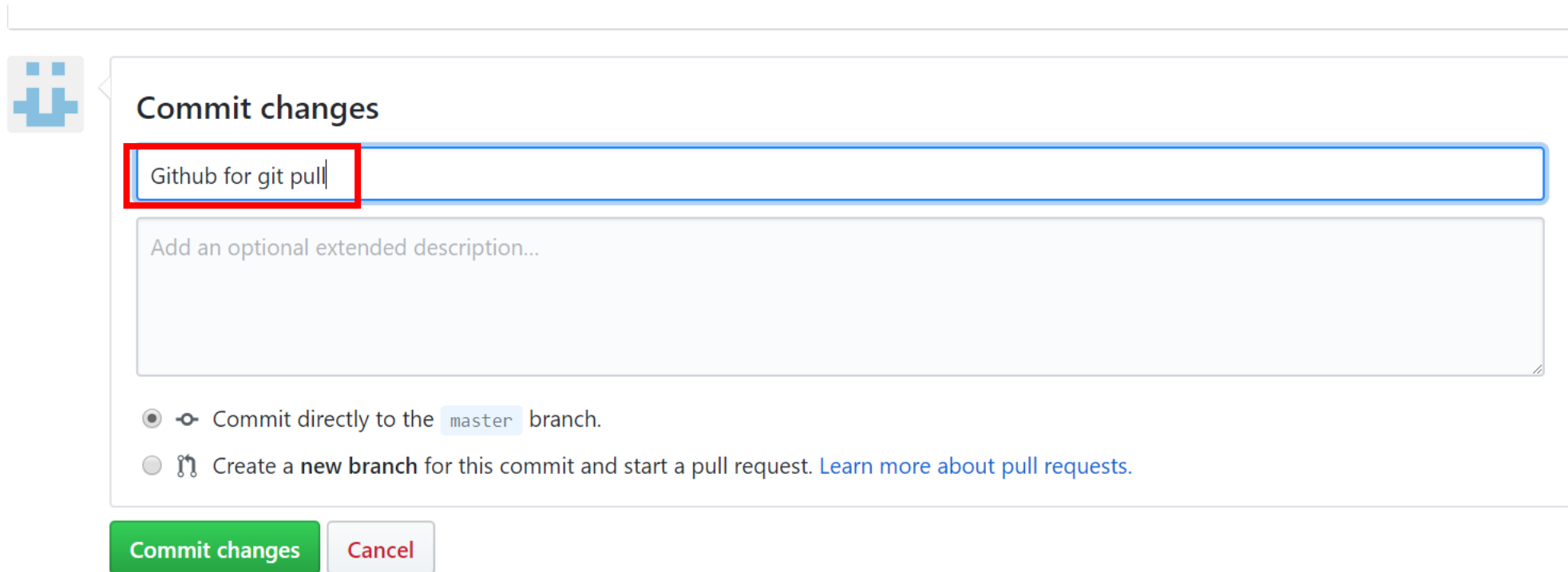
# Git pull

The screenshot shows the GitHub interface for the repository 'hysophie / command\_hello'. At the top, there are buttons for 'Watch', 'Star', and 'Fork', each with a count of 0. Below this is a navigation bar with links for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The 'Code' tab is selected. Below the navigation bar, the file path 'command\_hello / hello.py' is shown, followed by a clipboard icon and the text 'or cancel'. The main area shows the 'Edit file' tab selected, with a 'Preview changes' button. On the right, there are settings for 'Spaces' (set to 2) and 'No wrap'. The code editor displays three lines of Python code: 


```
1 print("Hello GH")  
2 print("Growth Hackers")  
3 print("cheese cake")  
4
```

 The third line, `print("cheese cake")`, is highlighted with a red rectangular box.

# Git pull



A screenshot of the Git commit dialog box. The dialog has a title bar with the Git logo and the text "Commit changes". Below the title bar is a text input field containing "Github for git pull", which is highlighted with a red rectangular border. Below the input field is a larger text area with the placeholder text "Add an optional extended description...". At the bottom of the dialog, there are two radio button options: the first is selected and labeled "Commit directly to the master branch.", and the second is labeled "Create a new branch for this commit and start a pull request. Learn more about pull requests." Below the options are two buttons: a green "Commit changes" button and a grey "Cancel" button.

 **Commit changes**

Github for git pull

Add an optional extended description...

☒ Commit directly to the `master` branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

**Commit changes** Cancel

# Git pull

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git pull origin master
```

**git pull** 원격저장소별칭 로컬브랜치이름

: 원격저장소의 내용을 로컬저장소로 불러오고, 로컬브랜치에 병합  
(git fetch + git merge)

# Mission6 성공!

- (Mission6) 원격저장소의 변경사항을 로컬저장소에 가져오자!

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master
$ ls
hello.py

renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master
$ cat hello.py
print("Hello GH")
print("Growth Hackers")
print("cheese cake")
```

2019.03.16

# ANACONDA & JUPYTER NOTEBOOK

이현아

# CONTENTS

---

1. Anaconda
2. Jupyter Notebook

3/21/2019

# 1. ANACONDA

가상환경 생성 / 패키지 관리

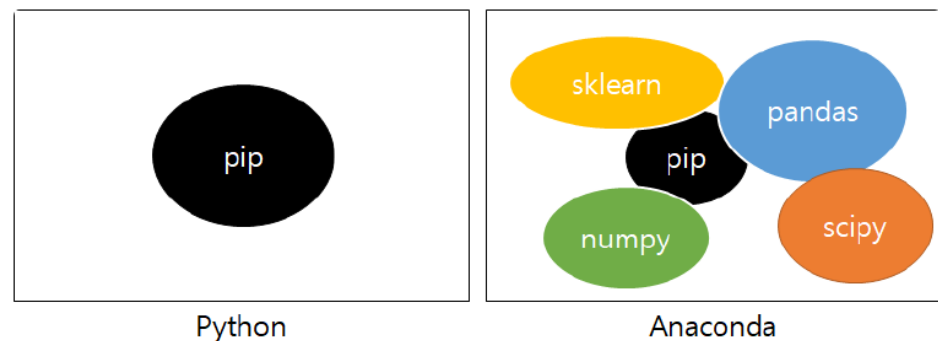
72



# Why Anaconda?

- **Anaconda를 사용하는 이유**

- Python 기반 데이터 분석에 필요한 주요 패키지 포함
- 주요 IDE 포함 (ex. Jupyter notebook)
- 가상환경 및 패키지 관리자 제공
- Tensorflow의 문서가 conda 기준으로 작성됨



출처: <http://snowdeer.github.io/python/2017/11/07/python-vs-anaconda/>

# Conda – 기본 명령

- 아나콘다 버전 확인 (conda --version)

```
Anaconda Prompt
(base) C:\Users\reno1>conda --version
conda 4.6.8
```

- 아나콘다 업데이트 (conda update conda)

```
(base) C:\Users\reno1>conda update conda
Collecting package metadata: done
Solving environment: done

# All requested packages already installed.
```

# Conda – 패키지 관리

- 설치된 패키지 목록 확인 (conda list)

```
(base) C:\Users\reno1>conda list
# packages in environment at C:\Anaconda3:
#
# Name                        Version      Build    Channel
ipyw_jlab_nb_ext_conf        0.1.0        py37_0
alabaster                     0.7.11       py37_0
anaconda                      5.3.1        py37_0
anaconda-client               1.7.2        py37_0
anaconda-navigator            1.9.2        py37_0
anaconda-project              0.8.2        py37_0
appdirs                       1.4.3        py37h28b3542_0
asn1crypto                    0.24.0       py37_0
```

- 패키지 설치 (conda install 패키지이름)

```
(growthhackers) C:\Users\reno1>conda install somepackage
```

(참고) conda 기본 서버에 존재하지 않는 패키지들은  
"pip install 패키지이름"으로 설치하기

# Conda – 가상환경 관리

- 새로운 가상환경 생성하기 (conda create -n 가상환경이름 python=버전)

```
(base) C:\Users\reno>conda create -n growthhackers python=3.7.0  
Collecting package metadata: done  
Solving environment: done
```

```
Proceed ([y]/n)? y
```

## (참고) 파이썬 버전 확인 방법

- Windows: 명령 프롬프트에 들어가서 python --version
- Mac: 터미널에 들어가서 python --version

명령 프롬프트

```
Microsoft Windows [Version 10.0.17134.648]  
(c) 2018 Microsoft Corporation. All rights reserved.  
  
C:\Users\reno>python --version  
Python 3.7.0
```

# Conda – 가상환경 관리

- 가상환경 목록 확인 (conda info --envs)

```
(base) C:\Users\reno1>conda info --envs
# conda environments:
#
base                * C:\Anaconda3
growthhackers        C:\Anaconda3\envs\growthhackers
```

# Conda – 가상환경 관리

- 가상환경 활성화 (activate 가상환경명)

```
(base) C:\Users\reno1>activate growthhackers  
(growthhackers) C:\Users\reno1>
```

- 가상환경에 설치된 패키지 목록 확인해보기 (conda list)

```
(growthhackers) C:\Users\reno1>conda list  
# packages in environment at C:\Anaconda3\envs\growthhackers:  
#  
# Name                        Version                Build    Channel  
certifi                       2019.3.9               py37_0  
pip                           19.0.3                 py37_0  
python                        3.7.0                  hea74fb7_0  
setuptools                    40.8.0                 py37_0  
vc                             14.1                   h0510ff6_4  
vs2015_runtime                14.15.26706            h3a45250_0  
wheel                         0.33.1                 py37_0  
wincertstore                   0.2                    py37_0
```

# Conda – 가상환경 관리

- 가상환경 비활성화 (conda deactivate)

```
(growthhackers) C:\Users\reno>conda deactivate  
(base) C:\Users\reno>
```

3/21/2019

# 2. JUPYTER NOTEBOOK

IDE란? / Jupyter notebook

80



# IDE란?

- 파이썬을 위한 통합개발환경 (Integrated Development Environment)
- 종류:



PyCharm



Spyder



Jupyter  
notebook

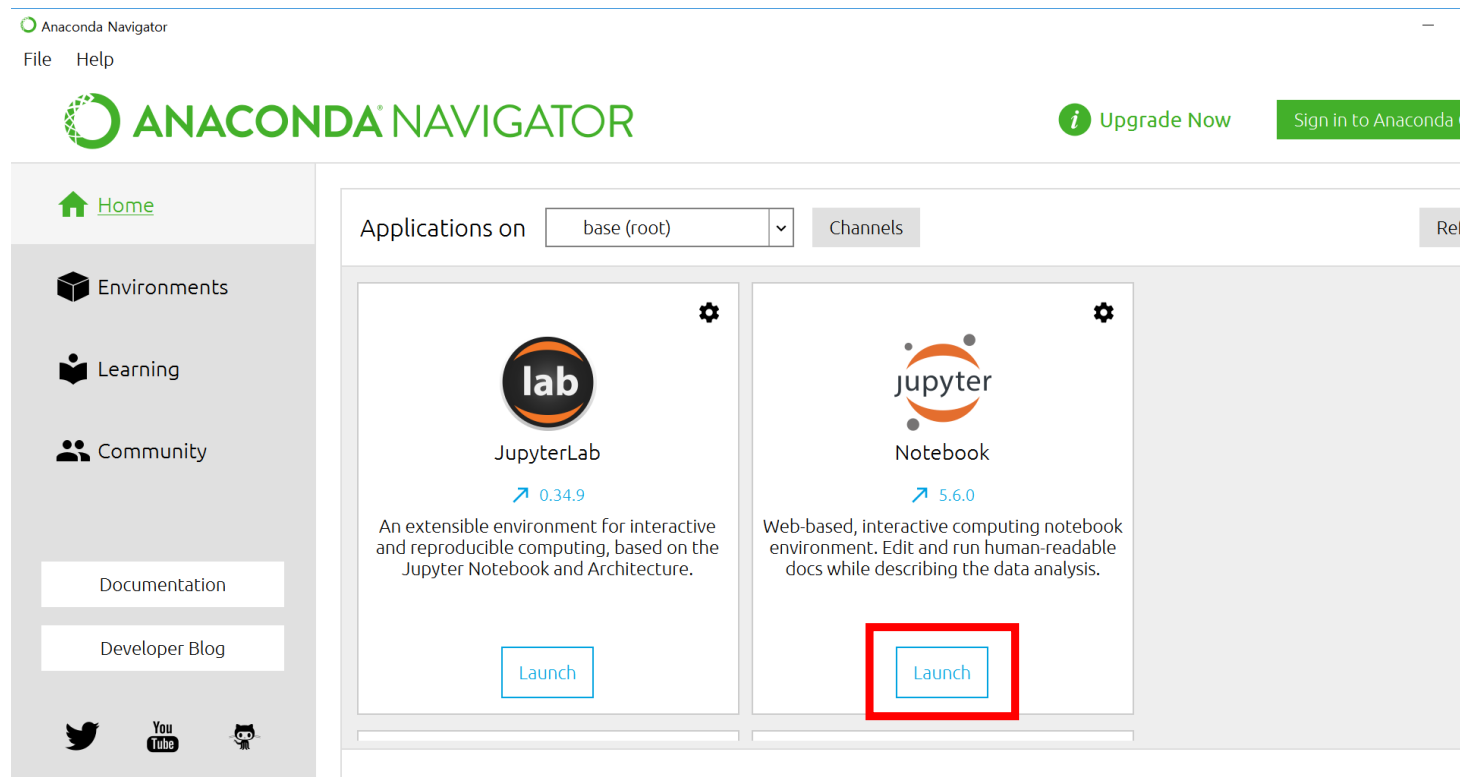
# Why Jupyter notebook?

- Html이나 pdf로 변환해서 공유하기 쉬움
- 셀 단위의 순차적인 실행 가능
- Github과 연동 가능



# Jupyter notebook 실행

## 1-1. Anaconda Navigator > 'Jupyter notebook'



# Jupyter notebook 실행

## 1-2. Anaconda Prompt > 'jupyter notebook' 입력

```
Anaconda Prompt - jupyter notebook
(base) C:\Users\poo0> jupyter notebook
[I 21:02:58.496 NotebookApp] JupyterLab beta preview extension loaded from C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab
[I 21:02:58.496 NotebookApp] JupyterLab application directory is C:\ProgramData\Anaconda3\share\jupyter\lab
[W 21:02:58.539 NotebookApp] Error loading server extension jupyterlab
Traceback (most recent call last):
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\commands.py", line 321, in __init__
    self._run(['node', 'node-version-check.js'], cwd=HERE, quiet=True)
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\commands.py", line 1165, in _run
    proc = Process(cmd, **kwargs)
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\process.py", line 73, in __init__
    self.proc = self._create_process(cwd=cwd, env=env)
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\process.py", line 131, in _create_process
    cmd[0] = which(cmd[0], kwargs.get('env'))
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\jupyterlab.py", line 59, in which
    raise ValueError(msg)
ValueError: Please install nodejs 5+ and npm before continuing installation. nodejs may be installed using conda or directly from the nodejs website.

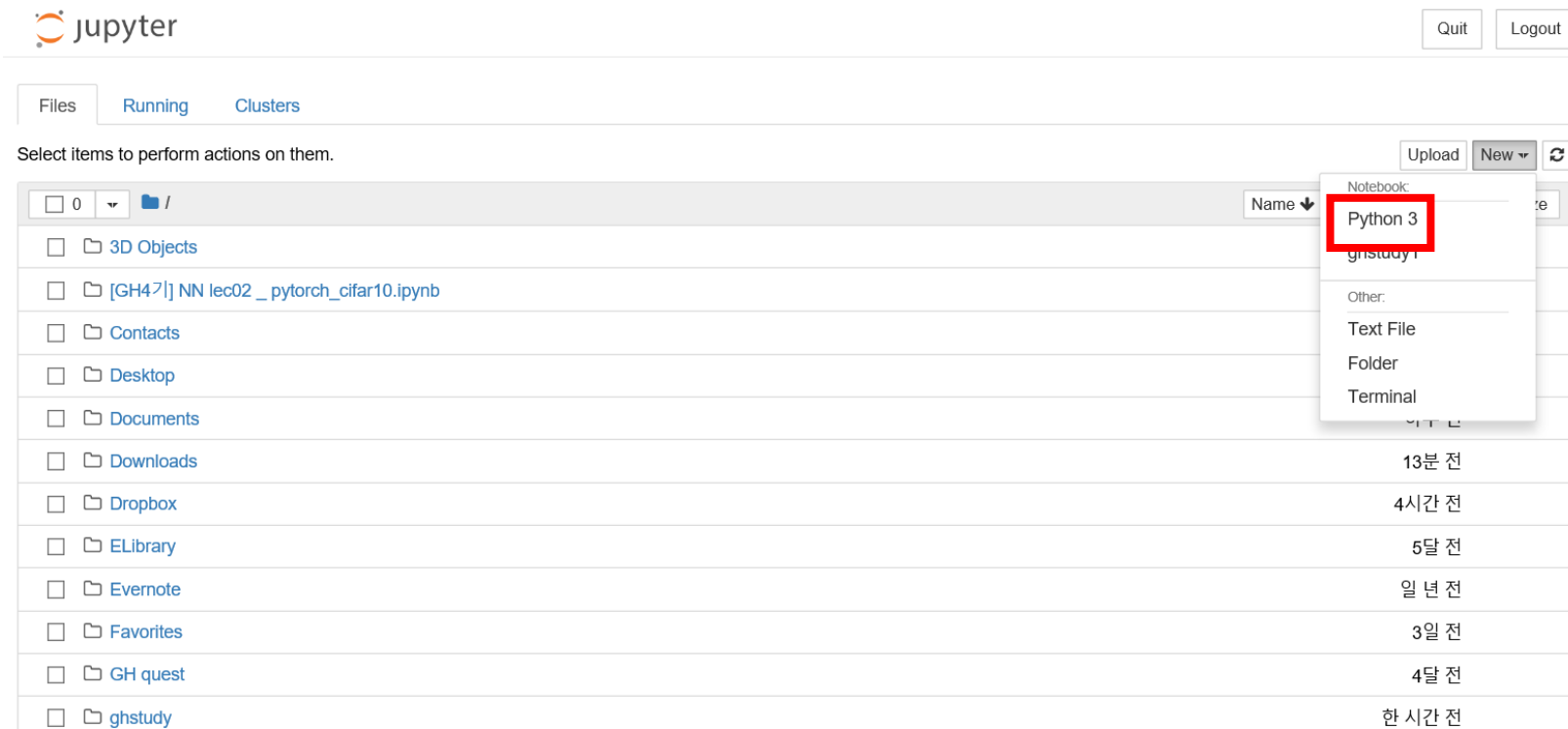
During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\ProgramData\Anaconda3\lib\site-packages\notebook\notebookapp.py", line 1454, in init_server_extensions
    func(self)
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\extension.py", line 111, in load_jupyter_server_extension
    info = get_app_info(app_dir)
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\commands.py", line 244, in get_app_info
    handler = _AppHandler(app_dir, logger)
  File "C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab\commands.py", line 324, in __init__
    raise ValueError(msg)
ValueError: Please install nodejs 5+ and npm before continuing installation. nodejs may be installed using conda or directly from the nodejs website.
[I 21:02:59.324 NotebookApp] Serving notebooks from local directory: C:\Users\poo0\
[I 21:02:59.325 NotebookApp] 0 active kernels
[I 21:02:59.326 NotebookApp] The Jupyter Notebook is running at:
[I 21:02:59.329 NotebookApp] http://localhost:8888/?token=8cbe525b622230149bba5eea9c43c48a2b7779154a0ddbc
[I 21:02:59.330 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 21:02:59.333 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:8888/?token=8cbe525b622230149bba5eea9c43c48a2b7779154a0ddbc
[I 21:03:00.018 NotebookApp] Accepting one-time-token-authenticated connection from ::1
```

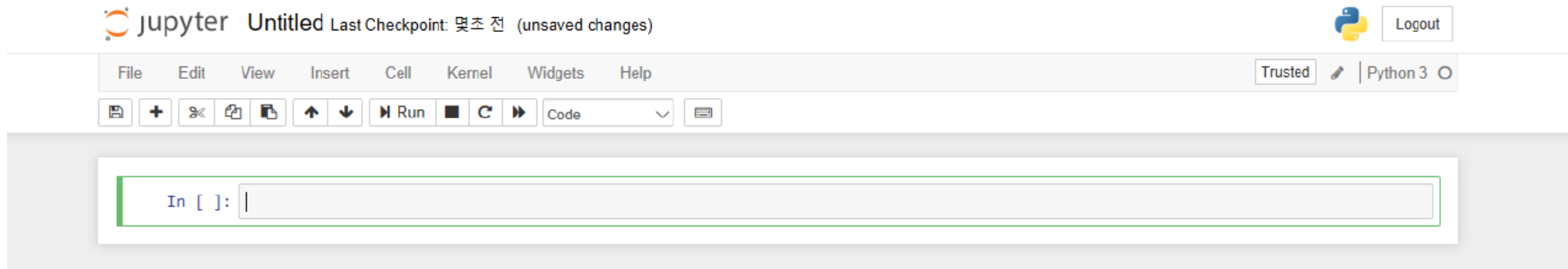
# Jupyter notebook 실행

## 2. 오른쪽 상단의 New > Python3




# Jupyter notebook 실행


## 3. Jupyter Notebook 실행 완료





# Jupyter notebook 사용법





 Save: 저장 [S]


 Add: 새로운 셀 추가 [A], [B]


 Cut: 잘라내기 [X]


 Copy: 복사 [C]


 Paste: 붙여넣기 [V]

 위의 셀로 이동 [Up], [K]

 아래의 셀로 이동 [Down], [J]

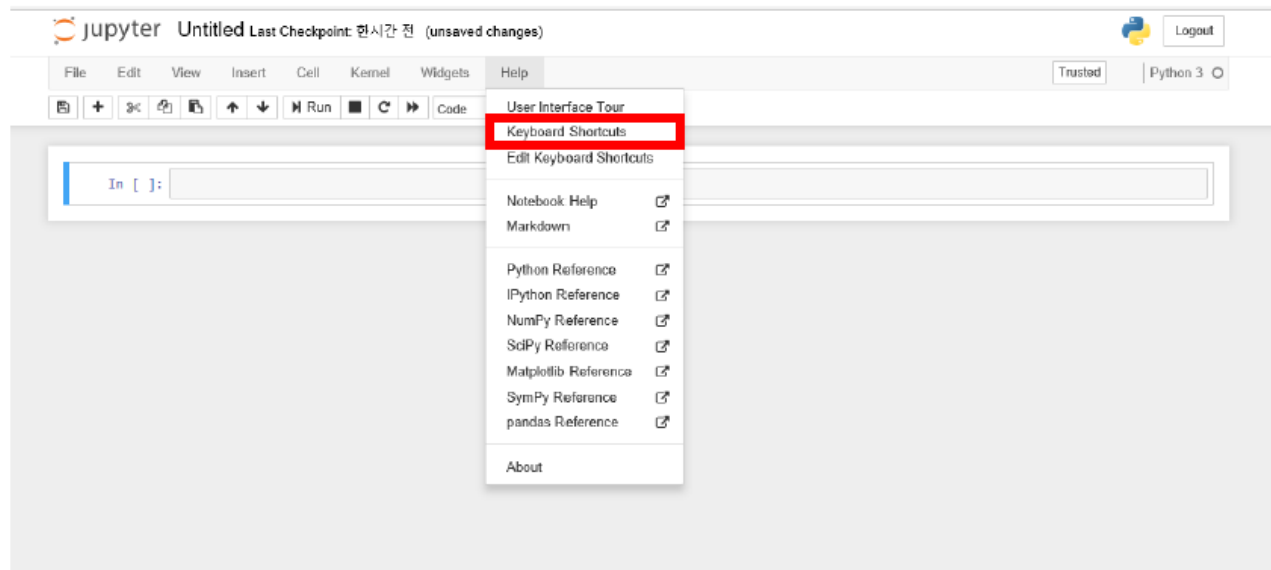
 Run: 입력한 코드 실행 [Shift] + [Enter]

 Code [Y], Markdown [M] 등 모드 변경

 Command Palette: 다양한 명령을 검색할 수 있는 검색창 패널 [P]

# Jupyter notebook 사용법

- 단축키 (Help > Keyboard Shortcuts)



단축키에 대해 더 알고 싶다면?

<https://www.dataquest.io/blog/jupyter-notebook-tips-tricks-shortcuts>



# Jupyter notebook 사용법

## • 기본 단축키

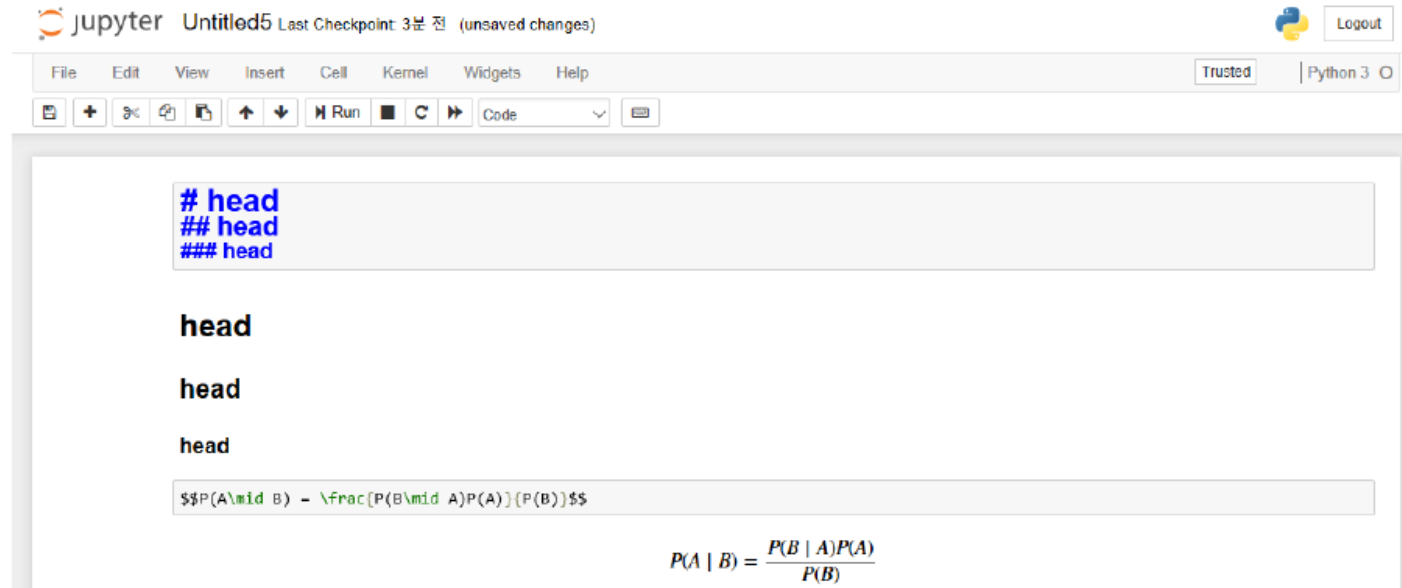
- Shift + Enter : 해당 Cell 실행, 그 아래 Cell 선택
- Ctrl + Enter : 해당 Cell 실행
- Alt + Enter: 해당 Cell 실행, 그 아래에 새로운 Cell 하나를 추가
- ➔ 작성된 코드를 실행할 때는 주로 Shift + Enter / 자신이 코드를 작성할 때는 주로 Alt + Enter
- a : 현재 cell 바로 위에 cell 하나 추가
- b : 현재 cell 바로 아래에 cell 하나 추가
- c : cell 복사하기
- v : 현재 cell의 아래에 붙여넣기
- shift + v : 현재 cell의 위에 붙여넣기
- d : 셀 삭제
- ctrl + z : 되돌리기

```
In [1]: print("Hello World!")
```

Hello World!

# Jupyter notebook 사용법

- Code & Markdown



Markdown에 대해 더 알고 싶다면?

<https://kevinthegrey.tistory.com/74?category=793117>:

# ★Quest★ 퀘스트 제출 방법 익히기

- Jupyter notebook에서 "Hello, I'm 본인이름"을 출력하는 코드 작성하고, 이를 .ipynb 파일로 저장하기  
➔파일 이름: "본인이름\_Quest1"
- Github에 퀘스트를 제출할 repository 만들기  
➔Repository 이름: "본인이름\_GH\_Quest"
- Jupyter notebook에서 실습한 파일을 git을 통해 github repository에 push하기

# 참고자료

- 만들면서 배우는 Git, GitHub 입문 (윤웅식)
- 생활코딩 "지옥에서 온 git" 강의

(링크: <https://www.youtube.com/watch?v=hFJZwOfme6w&list=PLuHgQVnccGMA8iwZwrGyNXCGy2LAAsTXk>)

- 구글링...

3/21/2019

감사합니다 😊

93

3/21/2019

# 부록

Git fetch의 사용법 & 충돌 해결

94

# 원격저장소에서 파일 수정

hysophie / command\_hello

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Branch: master command\_hello / hello.py Find file Copy path

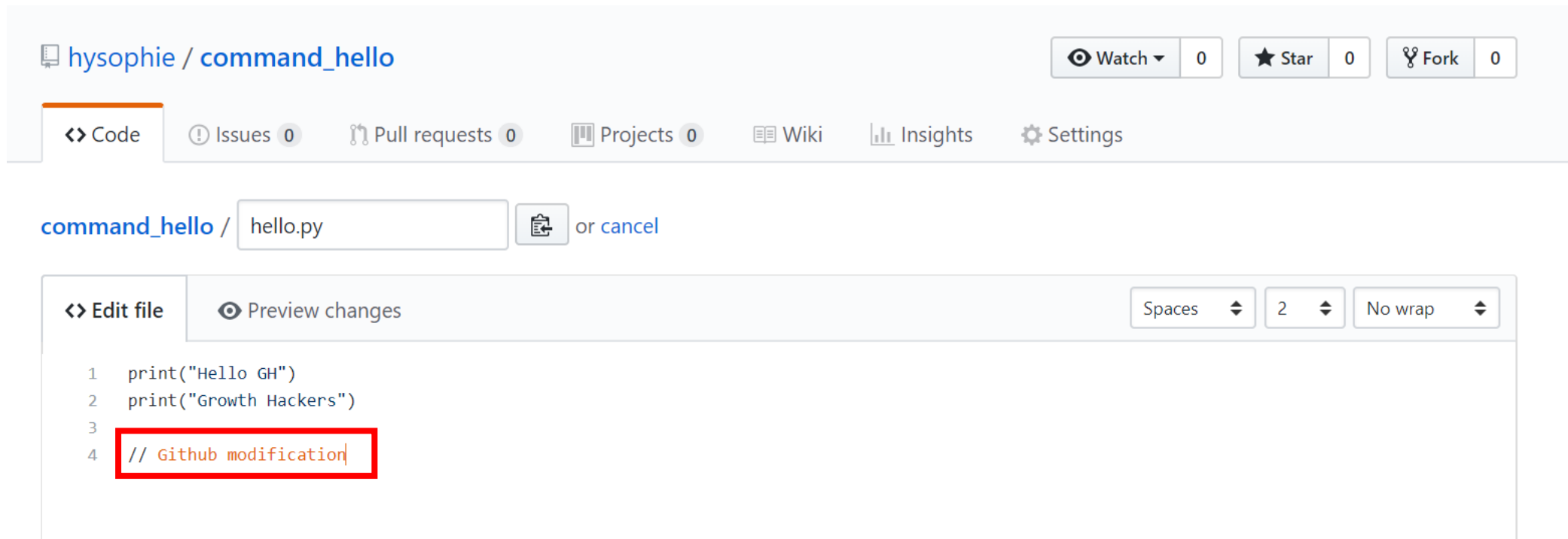
hysophie Update hello.py 1c9265c 10 seconds ago

1 contributor

3 lines (2 sloc) 42 Bytes Raw Blame History

```
1 print("Hello GH")
2 print("Growth Hackers")
```

# 원격저장소에서 파일 수정



The screenshot shows the GitHub interface for a repository named 'command\_hello' by user 'hysophie'. The repository has 0 Watchers, 0 Stars, and 0 Forks. The 'Code' tab is selected, showing a file named 'hello.py'. The file content is as follows:

```
1 print("Hello GH")
2 print("Growth Hackers")
3
4 // Github modification
```

The fourth line of code, `// Github modification`, is highlighted with a red rectangular box. The interface also shows options to 'Edit file' or 'Preview changes', and settings for 'Spaces' (2) and 'No wrap'.



# 원격저장소에서 파일 수정



## Commit changes

hello.py modified on GitHub

Add an optional extended description...

- ☒ Commit directly to the `master` branch.
- ☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel

# 원격저장소에서 파일 수정 (완료~)

The screenshot shows a GitHub repository page for 'hysophie / command\_hello'. At the top, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork' (0). Below these are tabs for 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Insights', and 'Settings'. The 'Code' tab is selected, showing the file 'hello.py' on the 'master' branch. A commit by 'hysophie' is shown with the message 'Update hello.py' and hash '95b8306'. Below the commit, it says '1 contributor'. The file details show '5 lines (3 sloc)' and '66 Bytes'. On the right, there are buttons for 'Raw', 'Blame', and 'History', along with icons for a monitor, edit, and delete. The code content is as follows:

```
1 print("Hello GH")
2 print("Growth Hackers")
3
4 // Github modification
```

The fourth line of code, `// Github modification`, is highlighted with a red rectangular box.

# 로컬저장소에서 파일 수정

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ vim hello.py
```

# 로컬저장소에서 파일 수정

A screenshot of a Windows command prompt window titled "MINGW64:/c/Users/renoi/git\_tutorial". The terminal shows three lines of Python code being executed:

```
print("Hello GH")  
print("Growth Hackers")  
// Local repository modification
```

The third line is highlighted with a red rectangular box. Below the code, there are several tilde (~) symbols indicating continuation or scrolling. At the bottom, the file path "hello.py[+]" is shown along with the date and time "(08:07 20/03/2019)". The cursor is positioned at the end of the first line of code.

- **Vim 사용법**
  - Vim에서 작성을 시작하려면 키보드의 [i]키를 눌러 작성모드로 바꾸기
  - 작성 후에 [ESC]키를 누르면 일반모드로 돌아옴
  - :를 입력하면 명령모드로 바뀌고, 명령모드에서 wq를 입력하면 저장(w) 후 종료(q)됨

# 로컬저장소에서 파일 수정

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git commit -a
```

git commit -a

: 변경된 로컬저장소 파일을 모두 로컬저장소에 제출(커밋)

# Git push → BUT 실패

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git push origin master
To https://github.com/hysophie/command_hello.git
! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/hysophie/command_hello.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'note about fast-forwards' in 'git push --help' for details.
```

왜 실패?

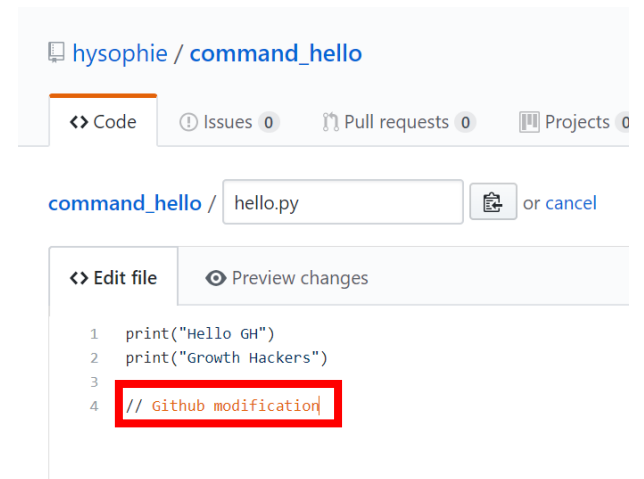
- 로컬저장소의 hello.py와 원격저장소의 hello.py의 내용이 다르기 때문
- Push하려는 원격저장소에 같은 이름의 브랜치가 있는데 서로의 내용이 다르면, push가 거부됨

# git fetch & git merge로 해결하기!

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git fetch
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 11 (delta 1), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (11/11), done.
From https://github.com/hysophie/command_hello
   3b4ae5e..e93f43a  master    -> origin/master
```

git fetch

: 원격저장소의 커밋들을 로컬저장소로 가져옴



# git fetch & git merge로 해결하기!

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)
$ git branch -a
hotfix
* master
remotes/origin/hotfix
remotes/origin/master
```

**git branch -a**

: 모든 로컬저장소, 원격저장소의 브랜치 정보를 볼 수 있음



# Git merge → BUT 충돌(conflict) 발생!

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git merge origin/master
```



```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git merge origin/master  
Auto-merging hello.py  
CONFLICT (content): Merge conflict in hello.py  
Automatic merge failed; fix conflicts and then commit the result.
```

# 충돌 해결 (git diff)

```
reno@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master | MERGING)
$ git diff
diff --cc hello.py
index 9651424,88b4c95..0000000
--- a/hello.py
+++ b/hello.py
@@@ -1,3 -1,4 +1,8 @@@
    print("Hello GH")
    print("Growth Hackers")
++<<<<<< HEAD
+// Local repository modification
+=====
+
+ // Github modificationn
++>>>>>> origin/master
```

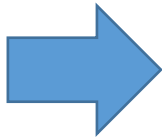
hello.py의 오류내용

## git diff

: 로컬저장소의 브랜치와  
원격저장소의 브랜치 사이에  
어떤 차이점이 있는지 확인

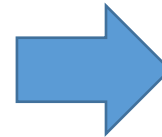
# 충돌 해결 (vim)

```
renoi@LAPTOP-F75  
$ vim hello.py
```



```
MINGW64:/c/Users/renoi/git_tutorial  
print("Hello GH")  
print("Growth Hackers")  
<<<<<< HEAD  
// Local repository modification  
=====  
  
// Github modificationnn  
>>>>>> origin/master  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
hello.py [dos] (23:37 20/03/2019)  
"hello.py" [ⓧ Ⓢ ] 8L, 151C
```

## 해결 전



```
MINGW64:/c/Users/renoi/git_tutorial  
print("Hello GH")  
print("Growth Hackers")  
// Local repository modification  
// Github modification
```

hello.py[+] [dos] (23:37 20/03/2019)  
:wq|

## 해결 후

# 충돌 해결 (git commit -a)

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master|MERGING)  
$ git commit -a
```

git commit -a  
: 변경된 저장소 파일을 모두 저장소에 제출(커밋)


# 커밋 메시지 작성

```
MINGW64:/c/Users/renoi/git_tutorial
conflict resolved GitHub
# Conflicts:
#   hello.py
#
# It looks like you may be committing a merge.
# If this is not correct, please remove the file
#   .git/MERGE_HEAD
# and try again.
#
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
# All conflicts fixed but you are still merging.
#
# Changes to be committed:
#   modified:   hello.py
#
~
~
~
C:/Users/renoi/git_tutorial/.git/COMMIT_EDITMSG[+] [unix] (23:40 20/0
:Wq|
```

# 충돌 해결 후 git push 시도

```
renoi@LAPTOP-F751N51M MINGW64 ~/git_tutorial (master)  
$ git push origin master
```

# Git push (성공~)

 [hysophie](#) / [command\\_hello](#)

Watch 0

Star 0

Fork 0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights


Settings

Branch: master ▾

[command\\_hello](#) / [hello.py](#)

Find file

Copy path




 [hysophie](#) [hello.py](#) modified on GitHub

e93f43a 10 minutes ago

1 contributor

6 lines (4 sloc) | 100 Bytes

RawBlameHistory



```
1 print("Hello GH")
2 print("Growth Hackers")
3
4 // Local repository modification
5 // Github modification
```