

# GITHUB 사용법

---

By 박찬희

## 목차


- I. Github 가입하기
- II. Gitbash 설치하기
- III. Repository 만들기
- IV. Git 명령어 익히기
  - I. Git clone
  - II. Git pull
  - III. Git add .
  - IV. Git commit -m "message"
  - V. Git push


# I. GITHUB 가입하기


- Github.com에 들어가서 sign in을 해줍니다.

## Join GitHub

The best way to design, build, and ship software.

**Step 1:**  
Set up your account

**Step 2:**  
Choose your subscription

**Step 3:**  
Tailor your experience

### Create your personal account

**Username \***

This will be your username. You can add the name of your organization later.

**Email address \***

We'll occasionally send updates about your account to this inbox. We'll never share your email address with anyone.

**Password \***

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

#### You'll love GitHub

- Unlimited public repositories
- Unlimited private repositories
- ✓ Limitless collaboration
- ✓ Frictionless development
- ✓ Open source community

## II. GITBASH 설치하기

- Gitforwindows.org에 들어가서 git bash를 설치해줍니다. (OS은 따로 있습니다.)

## II. 왜 GIT을 사용하는가?

- 설치하는 데 시간이 꽤 걸리니, 그 동안 설명을 해보겠습니다.
- Github : 쉽게 말해서 업로드/다운로드가 가능한 클라우드 서비스를 제공해주는 서버입니다. 한 파일당 최대 300mb 까지 업로드를 허용해줍니다. 일반 클라우드와 다르게 개인 소장보다 공유와 협업의 목적이 큼니다.
  - 1) 여러 가지 프로젝트를 관리하기가 용이하다
  - 2) 따라서 협업의 여지가 넓어진다
  - 3) 프로그램을 유지/보수하기 용이하다 -> 업데이트 로그를 계속해서 확인할 수 있다.
- 3번이 조금 중요한데, 자신/팀원이 코드의 어떤 부분을 추가/제거했는지 확인할 수 있습니다.

## II. 왜 GIT을 사용하는가?

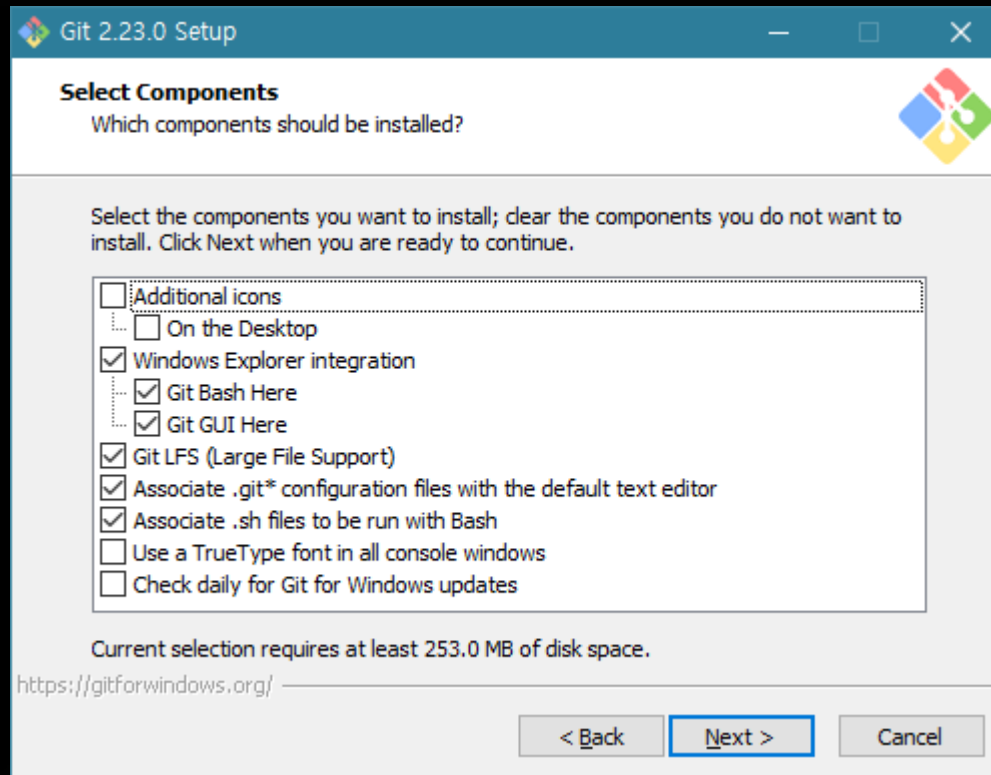
- 우리가 알고 있는 대부분의 오픈소스 라이브러리가 github를 통해서 배포되고 있습니다.
- 이처럼 자신이 만든 코드, 혹은 다른 사람이 만든 코드를 쉽게 공유할 수 있는 네트워크 역할을 합니다.
- 또한 \*개발자로서 자신의 성과를 꾸준히 증명할 수 있는 포트폴리오 기능을 합니다\*
- Github는 자체적으로 ~~.io/?? 와 같은 형태의 도메인을 만들 수 있도록 해주는데, 이를 통해서 자신의 웹 사이트를 만드는 거 역시 가능합니다.

## II. 왜 GIT을 사용하는가?

- Gitbash : 일종의 cmd창입니다. Github가 저장소라면, gitbash는 그 저장소에 접근할 수 있게 해주는 프로그램입니다.
- 여러 가지 명령어를 통해 github에 공유되고 있는 창작물을 관리할 수 있습니다.
- 명령어가 상당히 많지만, 기본적으로 pull/add/commit/push로 중요한 기능을 사용할 수 있습니다.

## II. GITBASH 설치 과정

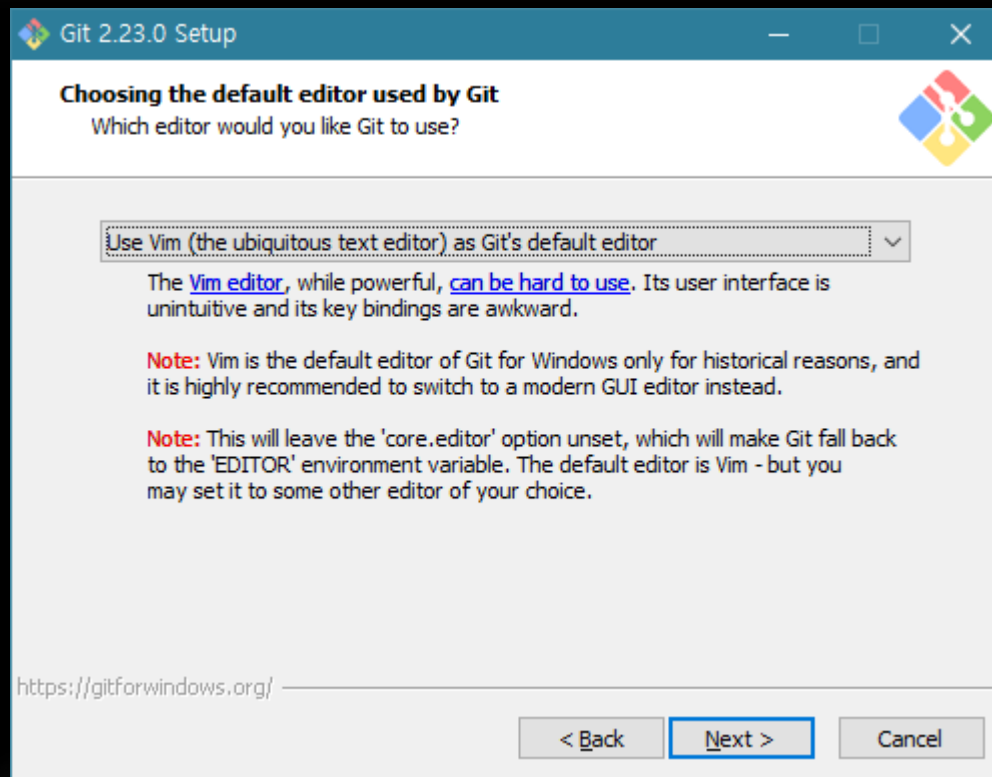
- 아무 것도 건드리지 맙시다.





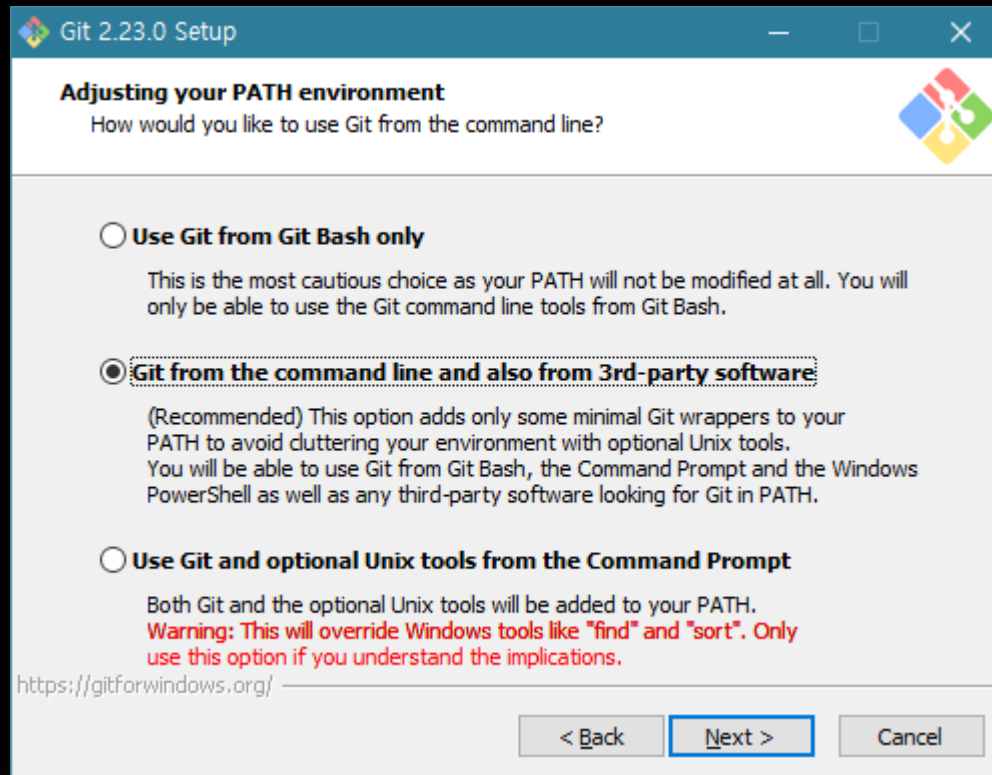
## II. GITBASH 설치 과정

- 별다른 이유가 없다면 비를 사용해주시는 게 좋다고 들어 더 거 같습니다.



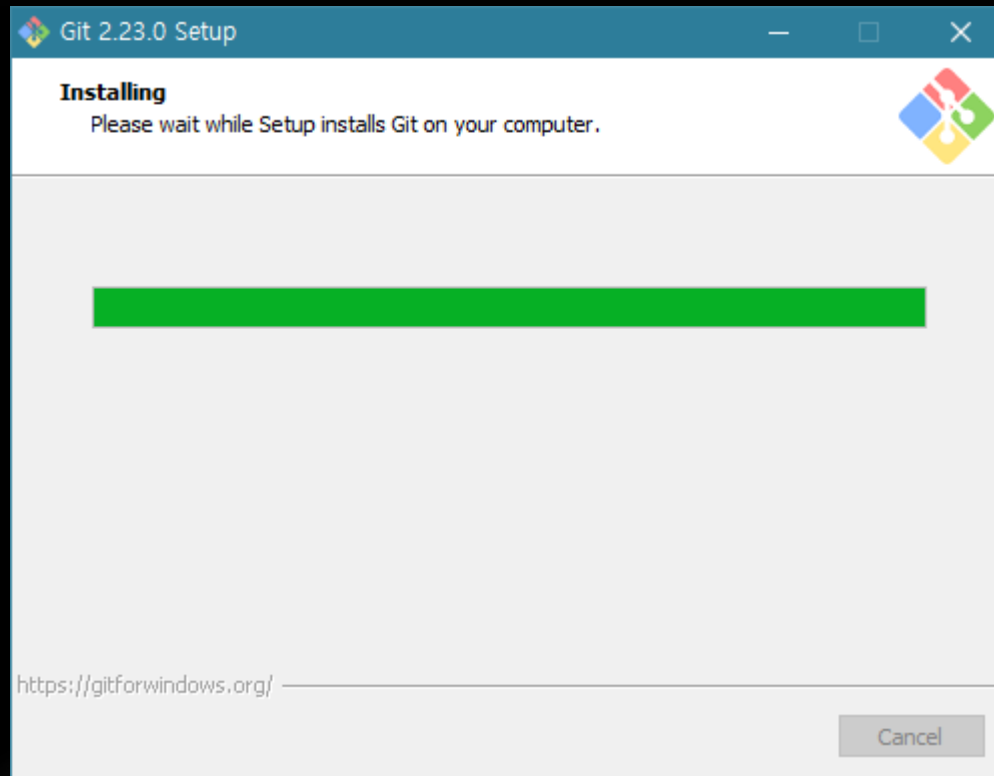
## II. GITBASH 설치 과정

- 이것도 딱히..?



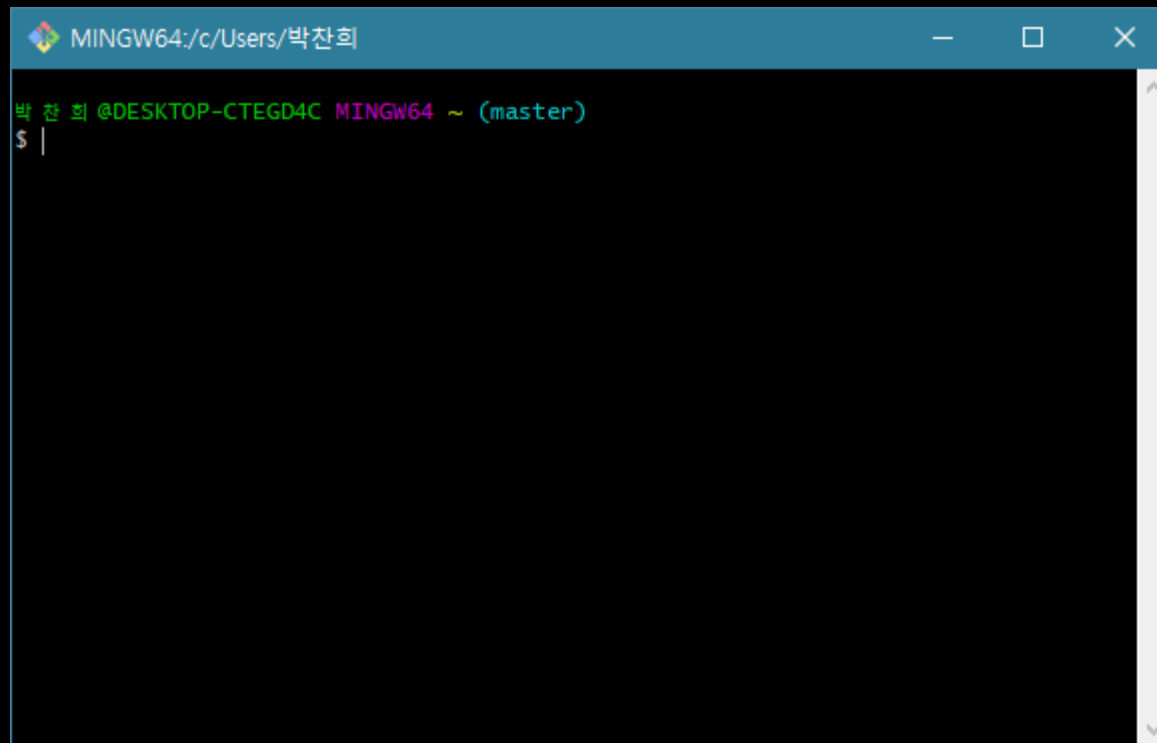
## II. GITBASH 설치 과정

- 그냥 기본 설정으로 두고 쓱쓱 넘어가 줍니다.



## II. GITBASH 설치 과정

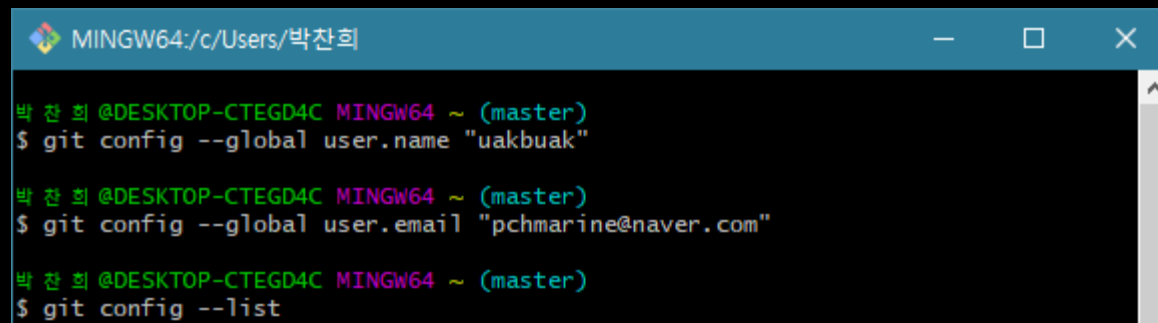
- 그러면 이런 기본 화면이 실행됩니다.



A screenshot of a MINGW64 terminal window. The title bar is blue and contains the text "MINGW64:/c/Users/박찬희" along with standard window control buttons. The terminal area has a black background with white text. The first line shows the prompt "박 찬 희 @DESKTOP-CTEGD4C MINGW64 ~ (master)" in a monospaced font. The second line shows the prompt "\$ |" with a vertical cursor. A vertical scrollbar is visible on the right side of the terminal window.

## II. GITBASH 설치 과정

- `Git config --global user.name` “자신의 github 닉네임”
- `Git config --global user.email` “자신의 github 이메일”
- 위 두 명령어로 사용자를 인식시켜줍니다.
- (비밀번호를 입력하는 부분도 있는데, 어느 부분인지 기억이 잘 나지 않습니다..)
- 위 작업은 한번만 수행합니다.



```
MINGW64:/c/Users/박찬희

박 찬 희 @DESKTOP-CTEGD4C MINGW64 ~ (master)
$ git config --global user.name "uakbuak"

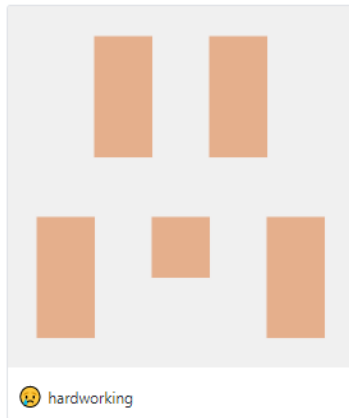
박 찬 희 @DESKTOP-CTEGD4C MINGW64 ~ (master)
$ git config --global user.email "pchmarine@naver.com"

박 찬 희 @DESKTOP-CTEGD4C MINGW64 ~ (master)
$ git config --list
```

### III. REPOSITORY 만들기

- Repository란 하나의 프로그램 혹은 하나의 프로젝트를 담은 폴더입니다.
- 각 repository는 프로그램/프로젝트의 사용법/개요 등을 설명해주는 README.md라는 파일이 꼭 들어가게 되어 있습니다. (이종의 약속 같은 것)
- 이는 처음에 얘기한 것처럼, '공유'와 '협업'의 모토와도 이어집니다.

# III. REPOSITORY 만들기



uakbuak

Edit profile

ProTip! Updating your profile with your name, location, and a profile picture helps other GitHub users get to know you.

Edit profile



Overview

Repositories 8

Projects 0

Stars 3

Followers 0

Following 0

Find a repository...

Type: All

Language: All

New

TIL

오늘 하루 공부한 내용을 올려보자.

Jupyter Notebook Updated 11 days ago

Star



writings Private

창작물 관리 레파지토리

Updated 13 days ago

Star



notes

대학교 수업 필기들을 전산화하여 보관하자

HTML Updated on 25 Jun

Star



### III. REPOSITORY 만들기

- 아래에 보여지고 있는 각각의 항목들이 저의 repository입니다.
- 오른쪽 상단에 초록색 버튼인 new를 눌러 새로운 저장소를 생성해줍니다.
- 참고로 TIL은 Today I learned라는 의미로, 주로 자신이 그날 학습한 내용들을 정리하는 용도로 사용하는 저장소의 이름입니다.



# III. REPOSITORY 만들기

## 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 \*

uakbuak ▾

/

Great repository names are short and memorable. Need inspiration? How about **didactic-broccoli**?

Description (optional)



**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** ▾



Create repository

### III. REPOSITORY 만들기

- Repository name에 저장소 이름을 적어 줍니다.
- Description은 짧막하게 저장소의 기능을 설명해줍니다.
- 공유 여부에 따라 public과 private을 선택해줍니다.
- Readme파일을 initialize하면 md 편집 툴이 없어도 github에서 간단한 readme 작성을 도와줍니다.

# III. REPOSITORY 만들기

The screenshot shows a GitHub repository page for 'uakbuak / test'. 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', 'Security', 'Insights', and 'Settings'. The main content area has a placeholder text 'No description, website, or topics provided.' with an 'Edit' button. Below this is a 'Manage topics' link. A summary bar shows '3 commits', '1 branch', '0 releases', and '1 contributor'. Below the summary bar are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find File', and a green 'Clone or download' button. A dropdown menu is open from the 'Clone or download' button, showing 'Clone with HTTPS' (with a help icon) and 'Use SSH'. The 'Clone with HTTPS' option includes the text 'Use Git or checkout with SVN using the web URL.' and the URL 'https://github.com/uakbuak/test.git' with a copy icon. At the bottom of the dropdown are 'Open in Desktop' and 'Download ZIP' buttons. The repository file list shows a file named 'text' with a commit message '3: modified'. A blue box at the bottom of the repository area contains the text 'Help people interested in this repository understand your project by adding a README.'

uakbuak / test

Watch 0 Star 0 Fork 0

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

No description, website, or topics provided. Edit

Manage topics

3 commits 1 branch 0 releases 1 contributor

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

Clone with HTTPS ? Use SSH

Use Git or checkout with SVN using the web URL.

https://github.com/uakbuak/test.git

Open in Desktop Download ZIP

uakbuak 3: modified

text 3: modified

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

### III. REPOSITORY 만들기

- Clone or download 버튼을 누르면 url이 나오는데, 오른쪽에 아이콘을 누르거나 ctrl+c 해서 url을 복사하도록 합니다.
- 그다음 화면을 내리고, 자신이 git폴더로 사용하고 싶은 폴더를 하나 만듭니다.
- 저 같은 경우에는 git이라는 이름의 폴더를 만들어서 사용하고 있는데, 이름을 정하고 싶은 대로 정하며 됩니다.
- 폴더를 만들어 다면, 그 폴더에 들어가 빈 공간에 우클릭, git bash here 버튼을 누릅니다.

### III. REPOSITORY 만들기

- Clone or download 버튼을 누르면 url이 나오는데, 오른쪽에 아이콘을 누르거나 ctrl+c 해서 url을 복사하도록 합니다.
- 그다음 화면을 내리고, 자신이 git폴더로 사용하고 싶은 폴더를 하나 만듭니다.
- 폴더 이름을 정하고 싶은 대로 정하면 됩니다. 이 폴더는 github와 관련된 여러 저장소(책)를 다는 일종의 가방이라고 보시면 됩니다.
- 폴더를 만들어 다면, 그 폴더에 들어가 빈 공간에 우클릭, git bash here 버튼을 누릅니다.
- 이후 '-git init' 명령어를 입력해 해당 폴더를 github와 연동시킬 용도로 쓰게다는 내용을 컴퓨터에게 인식시킵니다. (.git 폴더가 생성됩니다.)
- 그런 다음 '-git clone 해당url' 을 입력해줍니다.

### III. REPOSITORY 만들기

```
MINGW64:/c/Users/박찬희/Desktop/Git
박 찬 희 @DESKTOP-CTEGD4C MINGW64 ~/Desktop/Git (master)
$ git clone https://github.com/uakbuak/test.git
Cloning into 'test'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 9 (delta 0), reused 9 (delta 0), pack-reused 0
Unpacking objects: 100% (9/9), done.

박 찬 희 @DESKTOP-CTEGD4C MINGW64 ~/Desktop/Git (master)
$ |
```

### III. REPOSITORY 만들기

- 성공적으로 불러와 다면 해당 폴더에 아까 만든 저장소 이름의 폴더가 생겨 있을 것입니다.
- 이제 github의 저장소와 gitbash가 성공적으로 연동되었습니다.
- github와 로컬 컴퓨터에 저장되어 있는 내용은 gitbash로 수정 사항을 갱신시켜주기 전까지는 결코 연동되지 않습니다 (\*클라우드 저장소와 다른점)
- 나 혼자만 해당 저장소를 사용하며 상관이 없지만, 특히 협업할 때는 내가 수정한 사항과 팀원이 수정한 내용이 맞지 않을 때 오류가 발생하게 됩니다. 따라서 팀 프로젝트를 진행할 때는 적절한 조율을 통해 누가 먼저 업데이트할 것인지 정하는 것이 중요합니다.

## IV. GITBASH 명령어 익히기

- 우클릭으로 텍스트 파일을 하나 생성하고, 그 안에 아무 내용이나 입력해봅시다.
- 그러면 현재 로컬 저장소에는 텍스트 파일이 하나 형성되어있는 상태지만, github는 아직 최초로 clone했을 때의 상태입니다.
- Gitbash에서 `-git status`를 입력하면 붉은 글씨로 텍스트 파일의 이름이 나오게 되는데, 이는 '수정되어으나 적용되지 않았다'는 의미입니다.
- '`-git add .`' 명령어는 해당 폴더에 있는 모든 수정 상황을 github에 적용하겠다는 명령어입니다. 무언가 바뀐 내용이 있다면 꼭 이 명령어를 써주어야 합니다.
- 만약 폴더 전체의 수정상황이 아니라, 개별 파일의 수정상황만 적용하고 싶다면 '`- git add 파일명 확장자`'와 같이 입력해줍니다.
- 성공적으로 add 했다면 `git status`를 입력했을 때 파일명이 초록색으로 뜨게 됩니다.



## IV. GITBASH 명령어 익히기

- ‘-git add .’ 명령어를 마쳤다면, gitbash는 컴퓨터의 변경사항을 ‘임시적으로’ 기억하고 있습니다. 그러나 아직 아무 것도 기록되지 않은 상태이고, 수정 작업을 계속해서 진행할 수 있습니다. 일종의 필수적 임시 저장 기능이라고 생각할 수 있습니다.
- ‘-git commit -m 메시지’ 명령어는 내가 add로 수정했던 모든 파일을 하나로 묶어서 서버로 보낼 준비가 되었다는 명령어입니다. 이때 메시지에 하고 싶은 메모를 기록하면 (ex) 8월 18일 스터디 일지) github 로그에 해당 내용이 일종의 헤드라인으로 입력됩니다.
- 여기까지 진행 되어 다면 일차적인 수정이 마쳐진 상태입니다. 그러나 이때 바뀐 내용은 우체국으로 보내기 전 택배 상태와 같습니다. 보내지 않으면 아무 일도 일어나지 않죠.

## IV. GITBASH 명령어 익히기

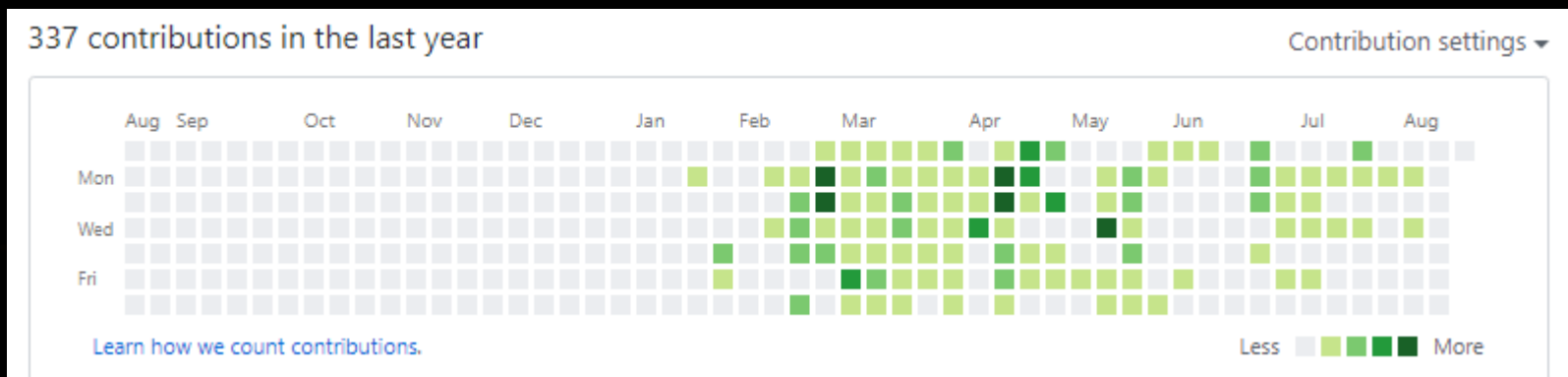
- ‘-git push’ 명령어를 입력하면 최종적으로 add로 기록한 모든 변동상황이 commit에 입력한 메시지로 포장되어 서버로 보내집니다.
- 여기까지 성공적으로 이루어져 다면 저장소에 해당 파일이 업로드되고, 변동 로그가 기록됩니다.
- 그렇다면 내가 만든 저장소를 다른 사람과, 혹은 다른 컴퓨터와 공유하고 싶다면 어떻게 할까요?
- 먼저 해당 저장소를 clone합니다. (방법은 위와 동일-1회만 적용)
- 그런데 해당 저장소 주인이 아주 성실해서 매일 업데이트가 되며 어떻게요? 마찬가지로 로컬 컴퓨터에서는 클라우드와 다르게 변동 상황을 내려받지 않으면 github에서 업데이트가 일어나는지 알 수 없습니다.

## IV. GITBASH 명령어 익히기

- ‘-git pull’ 명령어는 우리가 clone한 저장소에서 변동 내용이 있는지 살펴보고, 최신 버전으로 업데이트 시켜줍니다.
- 그런데 내가 업데이트를 하기도 전에 어떤 파일을 수정했는데, 그게 최신본과 다르며 충돌이 발생합니다.
- 이러한 상황을 해결하려면 굉장히 복잡하므로, 최초 1회 저장소를 clone했다면 이후 습관처럼 ‘-git pull’ 명령어를 가장 먼저 써서 서버에 변동사항이 있는지 확인하는 게 좋습니다.
- 즉, gitbash를 이용한 저장소 관리 메커니즘은 다음과 같습니다.  
‘pull -> 수정 -> add -> 수정 -> add -> ... -> commit -m -> push’

## IV. GITBASH 명령어 익히기

- 앞서 말한 것처럼 각 폴더는 하나의 저장소, 즉 하나의 프로그램이나 프로젝트를 의미합니다.
- 그러므로 옹도에 맞게 여러 가지 저장소들을 효율적으로 관리하고 이를 주기적으로 업데이트하는 것이 좋은 개발자로 나아가길 수 있는 방법이라고 할 수 있습니다.
- 그렇게 부지런하게 업데이트를 하면 자신의 github에 이와 같은 잔디가 생기게 되는데, 잔디가 뻗뻗하고 지을수록 부지런하게 코딩을 했다는 증거가 됩니다.
- 저런 고품격 바운 지후 한두 달 열심히 하다가 그 후로 망했네요..



감사합니다.