

# Lab : Git 與 GitHub 的入門與使用

## 一、本節目的：

- 了解什麼是版本控制行為、版本控制系統 (如: git, svn, ...等)
- 用 Git 解決對於程式碼的版本控制的困擾
- 了解常用 Git 的指令以及其功能
- 實際練習 Git 與 GitHub 的基本使用情境

## 二、觀念說明：

### 1. 使用 Git 的好處

- 如下圖，為我們平時在編輯檔案時，經常會有許多版本修改更新，造成一堆亂七八糟的檔案命名，如原始檔案為 120525\_文檔.txt，過程中可能變為如下：

- A. 120604\_文檔.txt
- B. 120604\_文檔\_更新版.txt
- C. 120604\_文檔\_最新.txt
- D. 120604-OAO\_文檔.txt

Name
120525_文檔_更新版.txt
120604_文檔.txt
120605_文檔_yourname.txt
120605_文檔_最新.txt
120605_文檔_最新複製.txt
120605_文檔_修改版.txt
120605_文檔.txt
120602_文檔.txt
文檔_會議用.txt



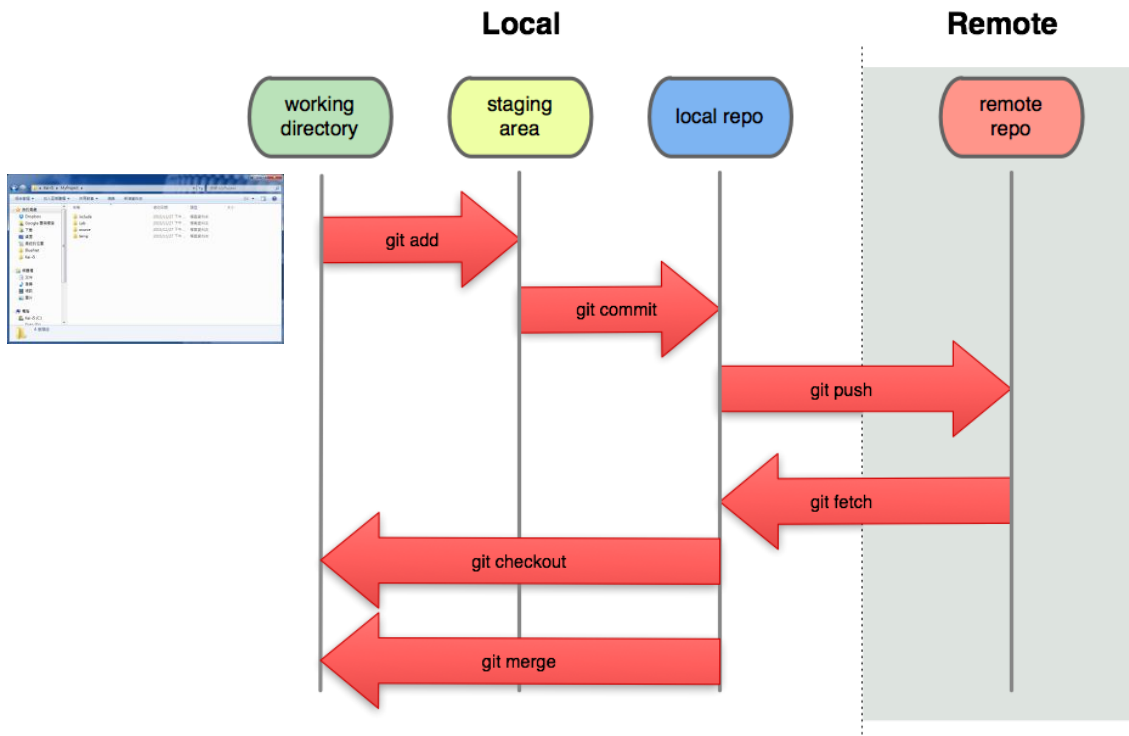
- 甚至會對於哪一個版本，修改哪些細節內容完全不記得，因此這時候 Git 來救你了！

### 2. Git (Local) 與 GitHub (Remote)

- Git: 版本控制程式，安裝於電腦端進行 git 指令實際操作。
- GitHub: 雲端程式專案發布平台，可將位於在電腦端進行 git 操作後的結果(commit 點)上傳至網路上進行分享亦可稱為備份，因此實現了讓多位程式開發人員可以一同處理開發同一程式專案。

如下圖，說明了程式專案在 Git 控管下與 GitHub 上的不同階段變化，可分為如下：

- working directory (工作目錄): 即為檔案系統下的一個資料夾目錄，但是受到 git 版本控管
- staging area (準備提交區): 在 working directory 下檔案的改變，如這些改變希望接下來可被提交至 local repository，會被放在此區域
- local repository (本地端程式庫): 即為自己電腦端上的程式庫(repository)，記錄了每個程式檔案以及修改變動 (commit 點)
- remote repository (遠端的程式庫): 同樣是紀錄了每個程式檔案以及修改變動 (commit 點)，但通常是來自於 local repository 的紀錄



- git add 將選定的程式檔案加入到 staging area 進行追蹤
- git commit 將 staging area 的檔案，提交成一個改變紀錄 (commit 點)
- git push 將 local repository 推到 remote repository 進程式上的同步
- git fetch 將 remote repository 的最新狀態擷取下來
- git checkout 用來切換程式庫下的分支
- git merge 用來合併不同分支上的檔案及內容

※本 Lab 主要是針對 A-C 三項 git 指令的使用進行練習

### 三、設計重點：

- 安裝 Git 使用環境 Git Bash
- 註冊 GitHub 帳號與建立一個 Remote Repository
- 實際練習 Git 與 GitHub 的基本使用情境

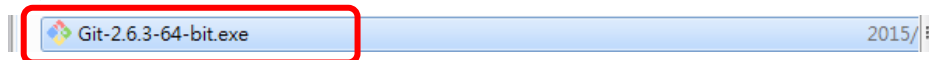
#### 四、設計說明：

##### ■ 安裝 Git 使用環境 Git Bash

1. 請先至 <https://git-for-windows.github.io/> 下載 Git



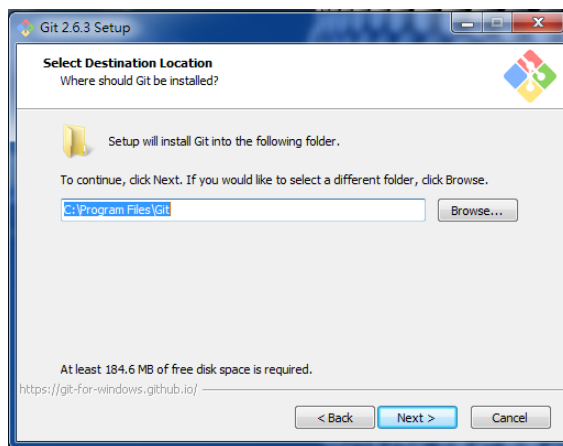
2. 點擊 Git-2.6.3-64-bit.exe 開始安裝 Git Bash



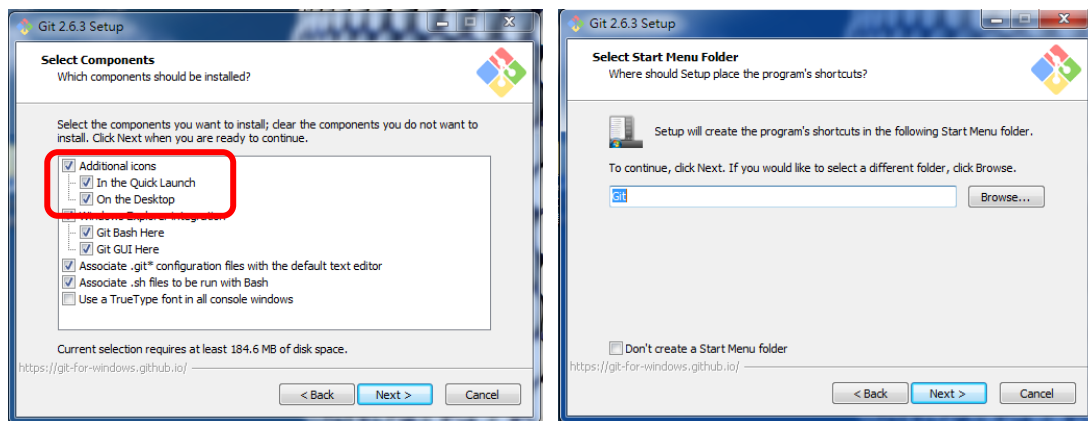
3. 點擊 Next，再點擊 Next



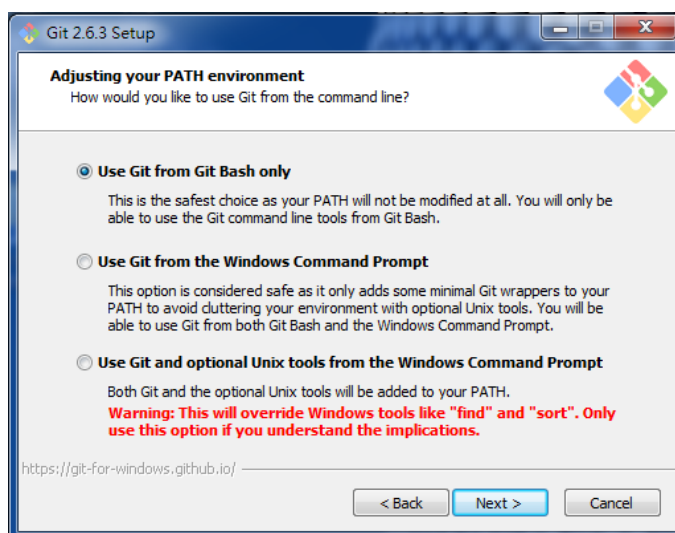
#### 4. 點擊 Next



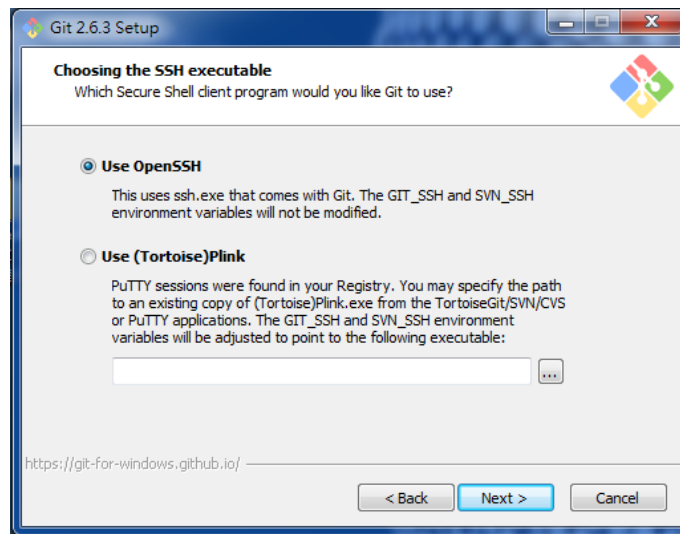
#### 5. 勾選 In the Quick Launch 和 On the Desktop，點擊 Next，再點擊 Next



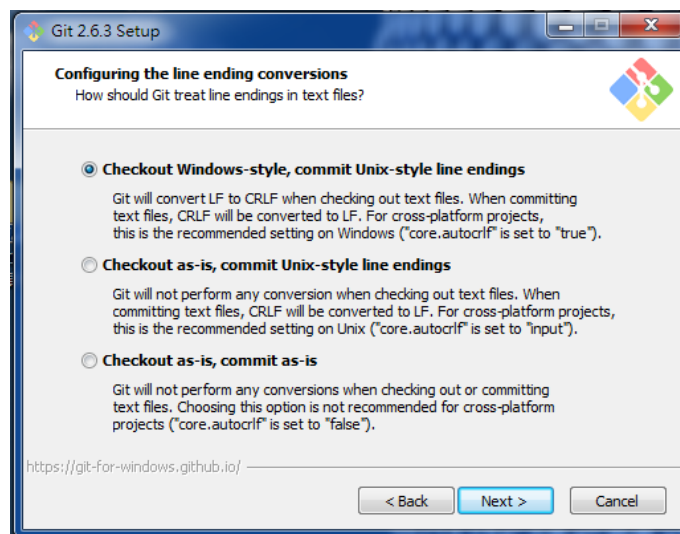
#### 6. 預設選擇 Use Git from Git Bash only，點擊 Next (註：設定環境變數 PATH)



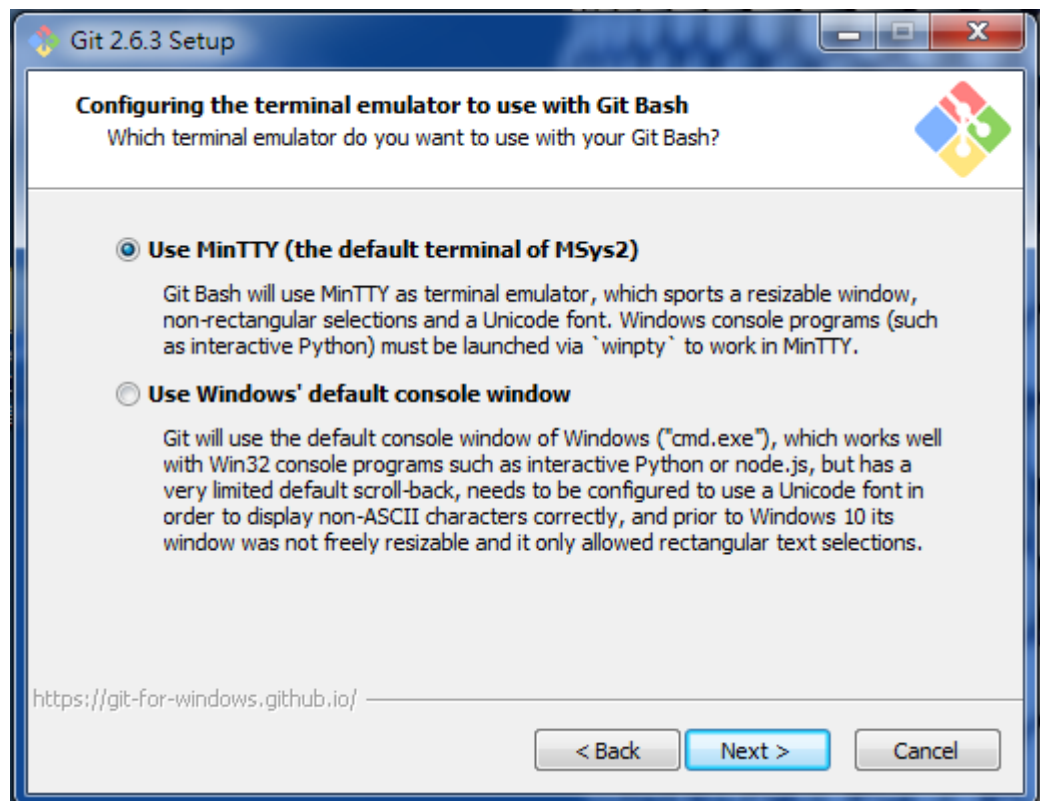
7. 預設選擇 Use OpenSSH，點擊 Next



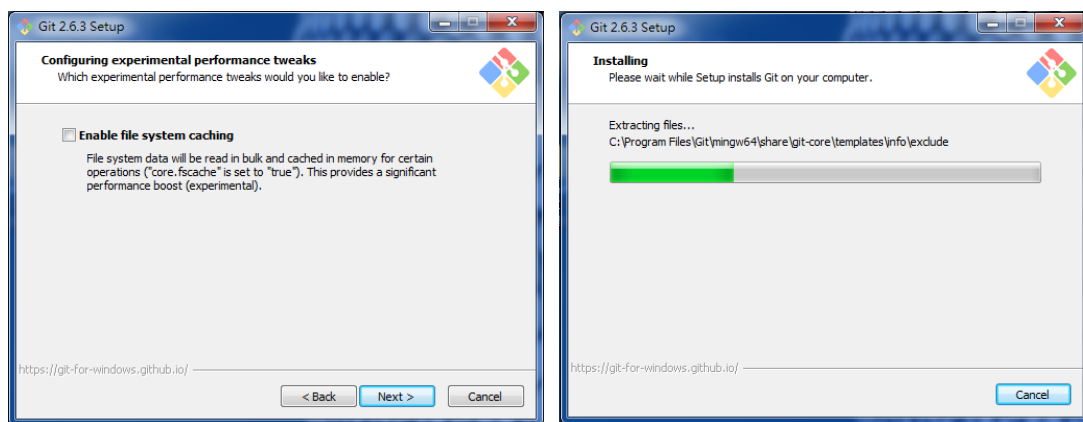
8. 預設選擇 Checkout Windows-style, commit Unix-style line endings，點擊 Next  
(註：設定結束符號類型)



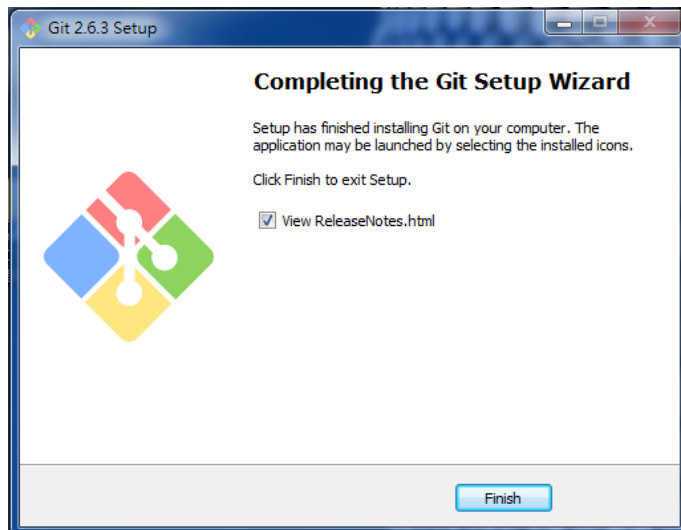
9. 預設選擇 Use MinTTY，點擊 Next



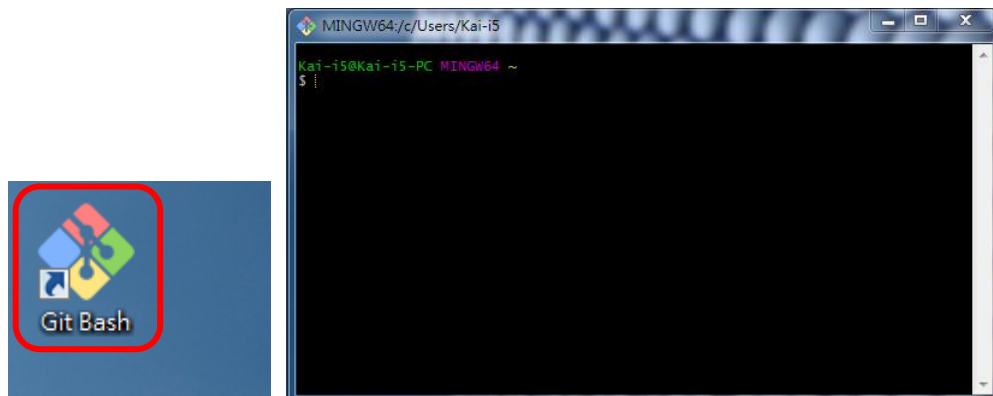
#### 10. 直接點擊 Next



#### 11. 完成安裝點擊 Finish

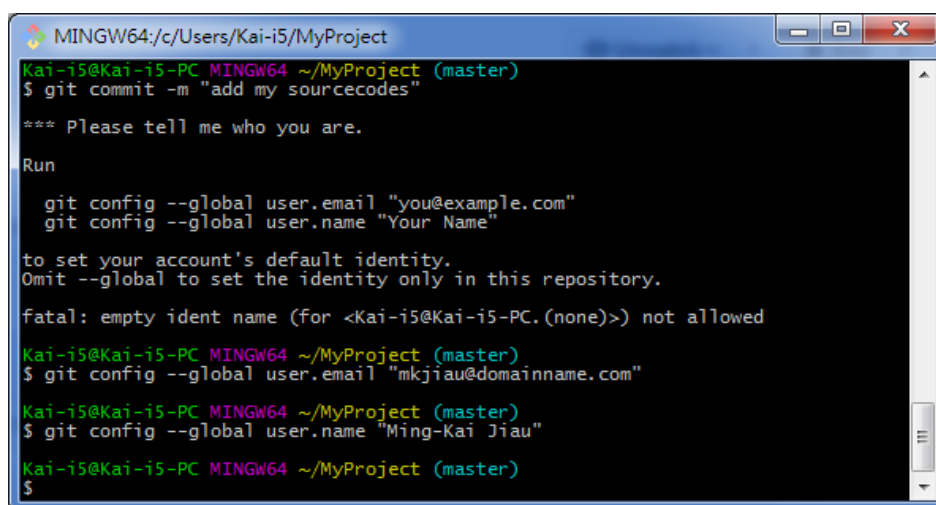


12. 到桌面點擊如下圖示，執行 Git Bash，如下黑色畫面



13. 在 git bash 內，設定自己的 user.email 和 user.name，使用如下指令：

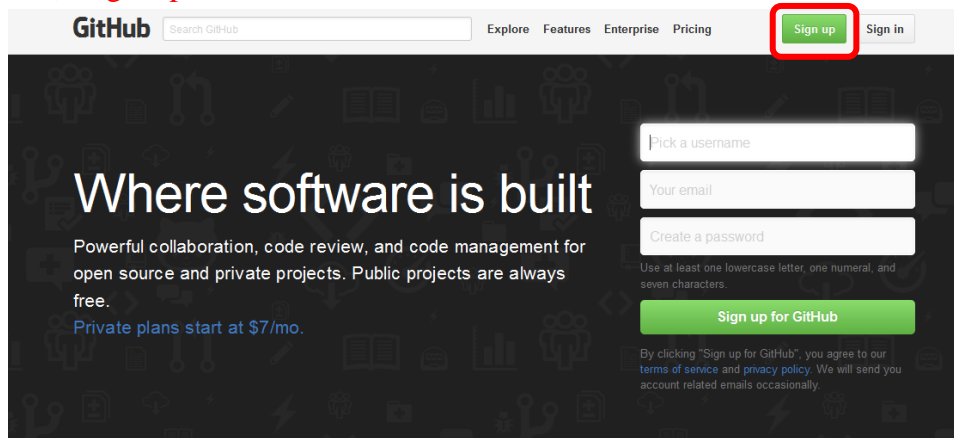
```
$ git config --global user.email "mkjiau@domainname.com"
$ git config --global user.name "Ming-Kai Jiau"
```



## ■ 註冊 GitHub 帳號與建立一個 Remote Repository

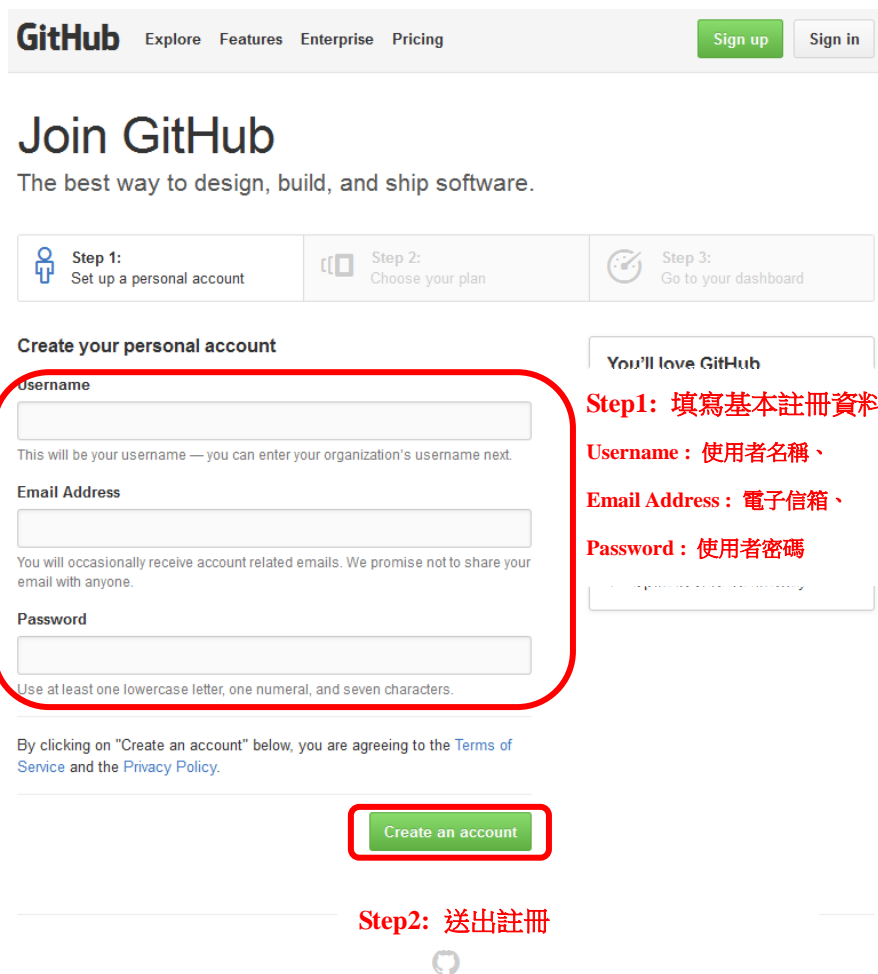
1. 至 GitHub 官方網站(<https://github.com/>) 註冊帳號，  
點擊 **Sign Up** 註冊 GitHub 帳戶

點擊此按鈕



The image shows the GitHub homepage. At the top right, there is a navigation bar with links for 'Explore', 'Features', 'Enterprise', and 'Pricing'. Next to these links is a green 'Sign up' button, which is highlighted with a red rectangle. Below the navigation bar, the main content area has a dark background with the text 'Where software is built' and 'Powerful collaboration, code review, and code management for open source and private projects. Public projects are always free.' To the right of this text is a registration form with fields for 'Pick a username', 'Your email', and 'Create a password'. Below these fields is a green 'Sign up for GitHub' button. At the bottom right, there is a small disclaimer: 'By clicking "Sign up for GitHub", you agree to our terms of service and privacy policy. We will send you account related emails occasionally.'

2. 填寫註冊資料，並點擊 **Create an account**



The image shows the 'Join GitHub' registration page. At the top, there is a navigation bar with links for 'Explore', 'Features', 'Enterprise', and 'Pricing'. Next to these links is a green 'Sign up' button. Below the navigation bar, the main content area has a white background with the text 'Join GitHub' and 'The best way to design, build, and ship software.' Below this text is a three-step process: 'Step 1: Set up a personal account', 'Step 2: Choose your plan', and 'Step 3: Go to your dashboard'. The 'Step 1' section is highlighted with a red rectangle. It contains the heading 'Create your personal account' and three input fields: 'Username', 'Email Address', and 'Password'. Below the 'Password' field is a note: 'Use at least one lowercase letter, one numeral, and seven characters.' To the right of the input fields is a section titled 'You'll love GitHub'. Below the input fields is a green 'Create an account' button, which is highlighted with a red rectangle. At the bottom, there is a disclaimer: 'By clicking on "Create an account" below, you are agreeing to the Terms of Service and the Privacy Policy.'

**Step1: 填寫基本註冊資料**  
**Username :** 使用者名稱、  
**Email Address :** 電子信箱、  
**Password :** 使用者密碼

**Step2: 送出註冊**

3. 選擇 GitHub 註冊帳號的方案，此處保持 **Free Plan** 的方案，直接點擊 **Finish sign up** 完成註冊



# Welcome to GitHub

You've taken your first step into a larger world, @chienhua7243.

✓ Completed  
Set up a personal account

📁 Step 2:  
Choose your plan

🕒 Step 3:  
Go to your dashboard

## Choose your personal plan

Plan	Cost (view in TWD)	Private repositories	
Large	\$50/month	50	Choose
Medium	\$22/month	20	Choose
Small	\$12/month	10	Choose
Micro	\$7/month	5	Choose
Free	\$0/month	0	Chosen

### Each plan includes:

Unlimited collaborators  
Unlimited public repositories

- ✓ Free setup
- ✓ HTTPS Protection
- ✓ Email support
- ✓ Wikis, Issues, Pages, & more

Charges to your account will be made in US Dollars. Converted prices are provided as a convenience and are only an estimate based on current exchange rates. Local prices will change as the exchange rate fluctuates.  
Don't worry, you can cancel or upgrade at any time.

☐ Help me set up an organization next  
Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.  
[Learn more about organizations.](#)

Finish sign up

4. 創立一個遠端 Repository 在 GitHub 上，  
點選綠色按鈕 **+New repository**

## GitHub Bootcamp

1



**Set up Git**  
A quick guide to help you get started with Git.

2



**Create repositories**  
Repositories are where you'll work and collaborate on projects.

3



**Fork repositories**  
Forking creates a new, unique project from an existing one.

4



**Work together**  
Send pull requests, follow friends. Star and watch projects.

chienhua7243

Welcome to GitHub! What's next? (2 minutes ago)  
[Create a repository](#)  
[Tell us about yourself](#)  
[Browse interesting repositories](#)  
[Follow @github on Twitter](#)

💡 ProTip! Edit your feed by updating the users you follow and repositories you watch.

Your repositories **+ New repository**

You don't have any repositories yet!  
[Create your first repository](#) or [learn more about Git and GitHub.](#)

ProTip! The [git-scm.com](#) site offers official downloads of Git installers.

📡 Subscribe to your news feed



## 5. 輸入及設定 Repository 的資料及屬性，之後點擊 Create repository

### Create a new repository

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

Owner

Repository name

kaikyle7997 / MyProject

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

Description (optional)

An awesome project to solve OO problem

☒ 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

### Step 1

Repository name : [程式庫名稱 (=專案名稱)]

Description : [程式庫描述]

Public/Private : [程式庫是否別人看得到]

Initialize this ... a README : [是否加入初始的 README.md 檔案]

Add .gitignore : [加入要忽略的檔案設定]

Add a license : [加入 sourcecode 想使用的

license 方式，如：MIT 最為自由]

### Step 2

點擊進行建立

## 6. 完成在 GitHub 讓的 remote repository 的建立，名為「MyProject」

This repository Search

Pull requests Issues Gist

Introducing a new repository design

Hey there! We're [rolling out](#) a faster, more streamlined repository experience and would love to give you early access.

Switch to new design

kaikyle7997 / MyProject

Unwatch 1 Star 0 Fork 0

1 commit 1 branch 0 releases 1 contributor

Branch: master MyProject / +

kaikyle7997 Initial commit Latest commit 6703a75 just now

README.md Initial commit just now

README.md

# MyProject

An awesome project to solve OO problem

Code

Issues 0

Pull requests 0

Wiki

Pulse

Graphs

Settings

HTTPS clone URL

https://github.com

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

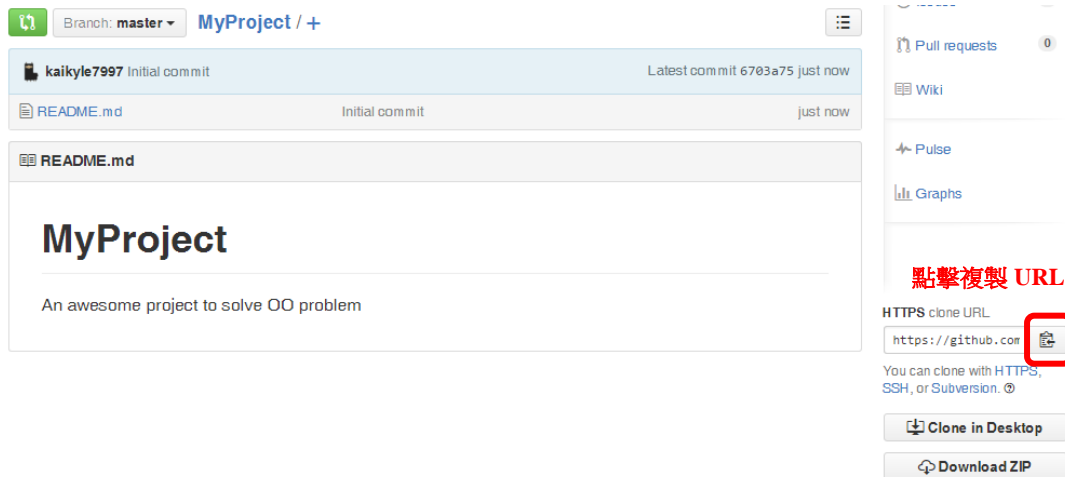
Clone in Desktop

Download ZIP

## ■ 實際練習 Git 與 GitHub 的基本使用情境

1. 從 GitHub 下載剛剛在上面建立的 **remote repository (MyProject)**，首先將 MyProject 的專屬 URL 複製，點擊如下圖紅色框框內按鈕

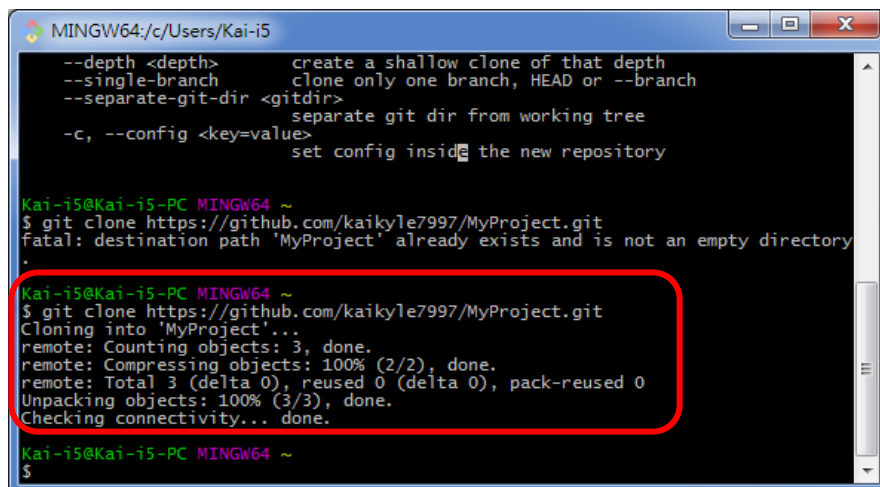
註：此 <https://github.com/kaikyle7997/MyProject.git> 已經存入在剪貼簿內



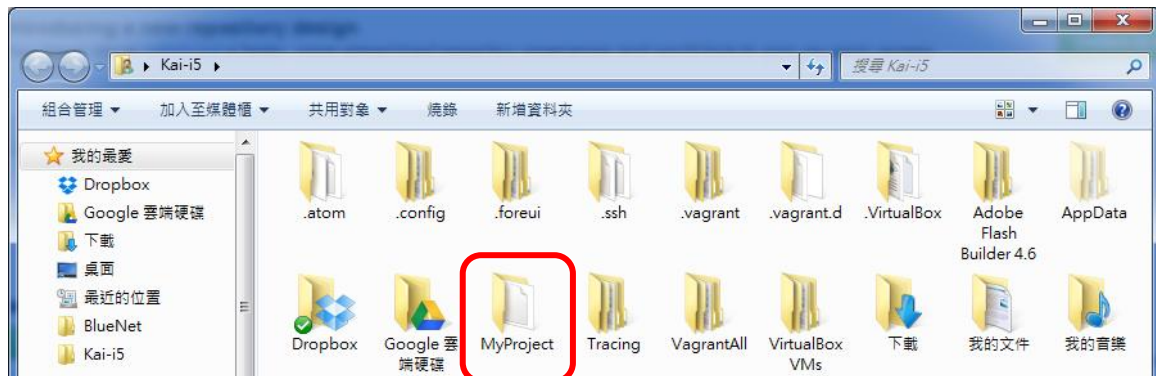
2. 回到剛剛已經執行的 git bash 視窗，執行如下指令，進行 Remote Repository (MyProject) 下載：

```
$ git clone https://github.com/kaikyle7997/MyProject.git
```

註：前一步驟已經將 Remote Repository 的 URL 存在剪貼簿內，所以打完 git clone 可以直接「右鍵 → past 貼上 URL」



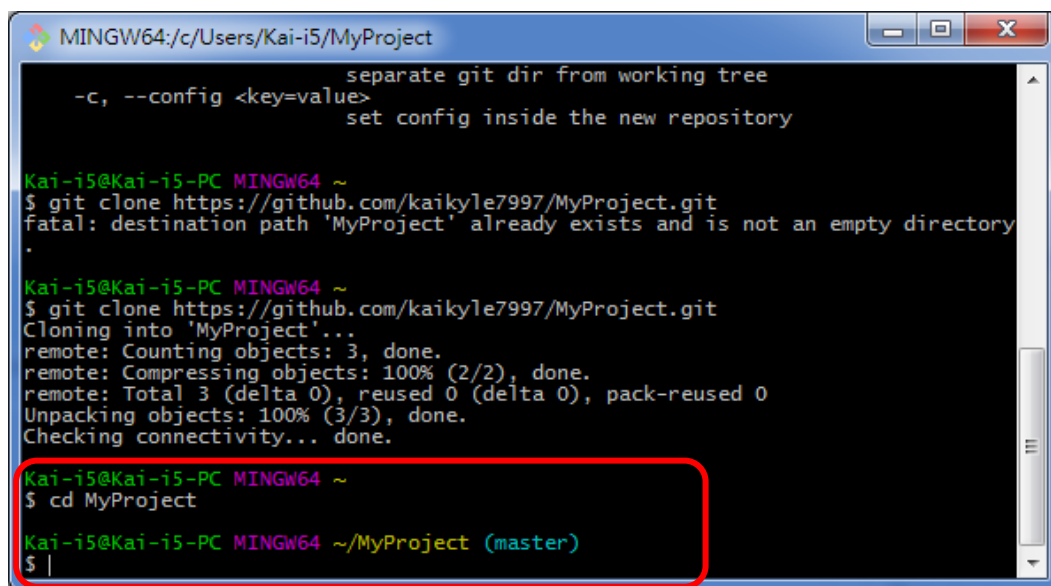
註：程式庫將下載到電腦的 C:\Users\[你的名稱]，會有一個資料夾較 MyProject



3. 執行下面指令，切換當前目錄到 local repository 的目錄位置，如下：

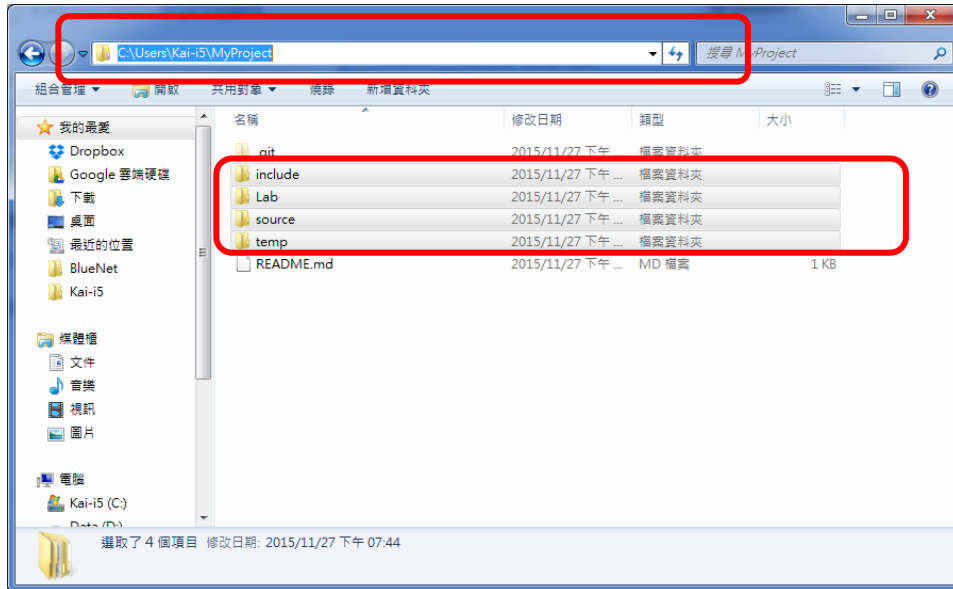
```
$ cd MyProject
```

註：~ 這符號代表使用者的主目錄，為 C:\Users\[你的名稱]，所以 cd MyProject 後，當前目錄變為 C:\Users\[你的名稱]\ MyProject\



4. 在檔案總管瀏覽路徑 C:\Users\[你的名稱]\ MyProject\，並加入自己的程式專案及檔案到此 local repository 目錄下，如下圖

註：可以滑鼠右鍵，加入一個文字檔案在 local repository 的目錄



5. 查看 local repository 目錄下的變更狀況，使用指令，如下：

```
$ git status
```

註：有 Lab、include、source、temp 四個目錄（紅色字體）及底下的程式檔案的增加，但未被追蹤

```
MINGW64/c/Users/Kai-i5/MyProject
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
Checking connectivity... done.
Kai-i5@Kai-i5-PC MINGW64
$ cd MyProject

Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    Lab/
    include/
    source/
    temp/

nothing added to commit but untracked files present (use "git add" to track)
Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$
```

6. 將此次的檔案變更動作加入（加入到 staging area），指令如下：

```
$ git add .
```

註：執行 git add 後可以，執行 git status 看一下詳細資訊，有哪些檔案被追蹤/新增了

```
MINGW64:/c/Users/Kai-i5/MyProject
temp/
nothing added to commit but untracked files present (use "git add" to track)
Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$ git add .
Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   Lab/Lab.sdf
        new file:   Lab/Lab.sln
        new file:   Lab/Lab/Ch1_Lab2.vcxproj
        new file:   Lab/Lab/Ch1_Lab2.vcxproj.filters
        new file:   include/User_defined.h
        new file:   source/Main.c
        new file:   source/User_defined.c
        new file:   temp/debug/Ch1_Lab2.exe
        new file:   temp/debug/Ch1_Lab2.ilc
        new file:   temp/debug/Ch1_Lab2.log
        new file:   temp/debug/Ch1_Lab2.pdb
        new file:   temp/debug/Ch1_Lab2.tlog/CL.read.1.tlog
        new file:   temp/debug/Ch1_Lab2.tlog/CL.write.1.tlog
        new file:   temp/debug/Ch1_Lab2.tlog/Ch1_Lab2.lastbuildstate
        new file:   temp/debug/Ch1_Lab2.tlog/cl.command.1.tlog
        new file:   temp/debug/Ch1_Lab2.tlog/link.command.1.tlog
        new file:   temp/debug/Ch1_Lab2.tlog/link.read.1.tlog
        new file:   temp/debug/Ch1_Lab2.tlog/link.write.1.tlog
        new file:   temp/debug/Main.obj
        new file:   temp/debug/User_defined.obj
        new file:   temp/debug/vc120.idb
        new file:   temp/debug/vc120.pdb

Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$
```

7. 提交此次 local repository 的目錄檔案變更(位於 staging area 的變更)，使用如下指令：

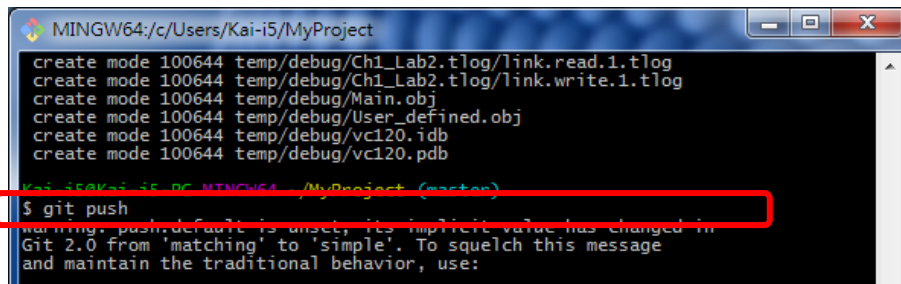
```
$ git commit -m "add my sourcecode"
```

```
MINGW64:/c/Users/Kai-i5/MyProject
Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$ git commit -m "add my sourcecodes"
[master 1ef0a8f] add my sourcecodes
22 files changed, 168 insertions(+)
create mode 100644 Lab/Lab.sdf
create mode 100644 Lab/Lab.sln
create mode 100644 Lab/Lab/Ch1_Lab2.vcxproj
create mode 100644 Lab/Lab/Ch1_Lab2.vcxproj.filters
create mode 100644 include/User_defined.h
create mode 100644 source/Main.c
create mode 100644 source/User_defined.c
create mode 100644 temp/debug/Ch1_Lab2.exe
create mode 100644 temp/debug/Ch1_Lab2.ilc
create mode 100644 temp/debug/Ch1_Lab2.log
create mode 100644 temp/debug/Ch1_Lab2.pdb
create mode 100644 temp/debug/Ch1_Lab2.tlog/CL.read.1.tlog
create mode 100644 temp/debug/Ch1_Lab2.tlog/CL.write.1.tlog
create mode 100644 temp/debug/Ch1_Lab2.tlog/Ch1_Lab2.lastbuildstate
create mode 100644 temp/debug/Ch1_Lab2.tlog/cl.command.1.tlog
create mode 100644 temp/debug/Ch1_Lab2.tlog/link.command.1.tlog
create mode 100644 temp/debug/Ch1_Lab2.tlog/link.read.1.tlog
create mode 100644 temp/debug/Ch1_Lab2.tlog/link.write.1.tlog
create mode 100644 temp/debug/Main.obj
create mode 100644 temp/debug/User_defined.obj
create mode 100644 temp/debug/vc120.idb
create mode 100644 temp/debug/vc120.pdb

Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$
```

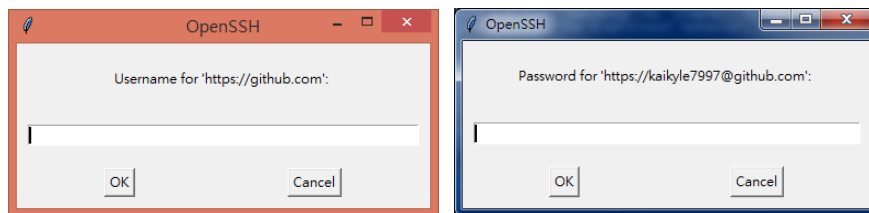
8. 將 local repository 變更紀錄提交到 remote repository (GitHub)上，指令如下：

```
$ git push
```

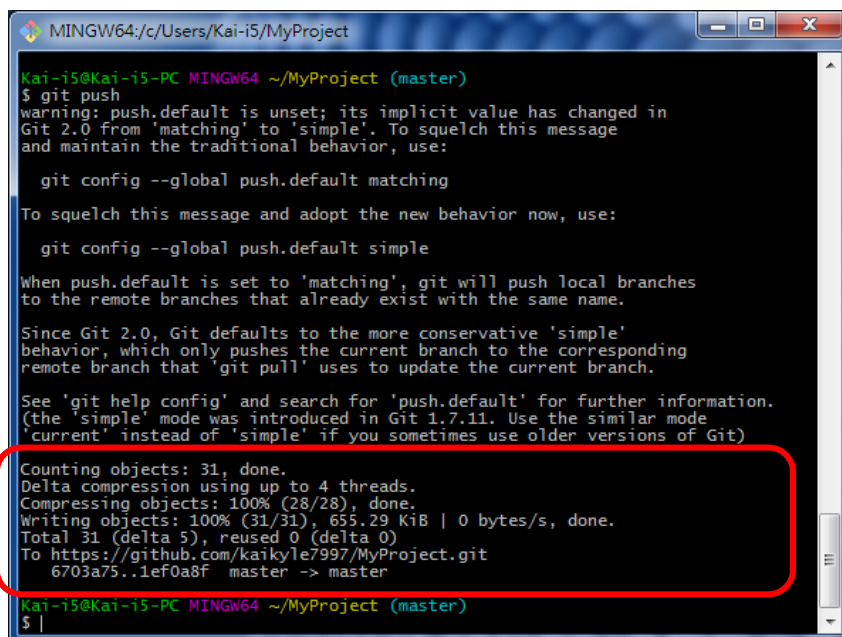


```
MINGW64:/c/Users/Kai-i5/MyProject
create mode 100644 temp/debug/Ch1_Lab2.tlog/link.read.1.tlog
create mode 100644 temp/debug/Ch1_Lab2.tlog/link.write.1.tlog
create mode 100644 temp/debug/Main.obj
create mode 100644 temp/debug/User_defined.obj
create mode 100644 temp/debug/vc120.idb
create mode 100644 temp/debug/vc120.pdb
Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$ git push
warning: push.default is unset; its implicit value has changed in
Git 2.0 from 'matching' to 'simple'. To squelch this message
and maintain the traditional behavior, use:
```

9. 此時須輸入 GitHub 的使用者名稱及使用者密碼，完成 git push 提交時需要的身份驗證，點選 OK



10. 成功推至遠端的 GitHub，如下畫面



```
MINGW64:/c/Users/Kai-i5/MyProject
Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$ git push
warning: push.default is unset; its implicit value has changed in
Git 2.0 from 'matching' to 'simple'. To squelch this message
and maintain the traditional behavior, use:

  git config --global push.default matching

To squelch this message and adopt the new behavior now, use:

  git config --global push.default simple


When push.default is set to 'matching', git will push local branches
to the remote branches that already exist with the same name.

Since Git 2.0, Git defaults to the more conservative 'simple'
behavior, which only pushes the current branch to the corresponding
remote branch that 'git pull' uses to update the current branch.

See 'git help config' and search for 'push.default' for further information.
(the 'simple' mode was introduced in Git 1.7.11. Use the similar mode
'current' instead of 'simple' if you sometimes use older versions of Git)




Counting objects: 31, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (28/28), done.
Writing objects: 100% (31/31), 655.29 KiB | 0 bytes/s, done.
Total 31 (delta 5), reused 0 (delta 0)
To https://github.com/kaikyle7997/MyProject.git
  6703a75..1ef0a8f master -> master
Kai-i5@Kai-i5-PC MINGW64 ~/MyProject (master)
$ |
```

11. 回瀏覽器，F5 重新整理 GitHub 的 remote repository 頁面，可以看到剛剛上傳的檔案



This repository    Search


[Pull requests](#)   [Issues](#)   [Gist](#)

Introducing a new repository design

Hey there! We're [rolling out](#) a faster, more streamlined repository experience and would love to give you early access.

Switch to new design

 [kaikyle799](#)

Unwatch

1

Star

0


Fork

0


An awesome project to solve OO problem — Edit

2 commits   1 branch   0 releases   1 contributor

Branch: master   **MyProject** / +


 Ming-Kai Jiau add my sourcecodes

Latest commit 1ef0a8f 7 minutes ago

 Lab


add my sourcecodes

7 minutes ago

 include


add my sourcecodes

7 minutes ago

 source


add my sourcecodes

7 minutes ago

 temp/debug


add my sourcecodes

7 minutes ago

 README.md

Initial commit

42 minutes ago

 README.md

<> Code

Issues0

Pull requests0

Wiki

Pulse

Graphs

Settings

HTTPS clone URL

<https://github.com>

You can clone with HTTPS, SSH, or Subversion. @

Clone in Desktop

Download ZIP