

ขั้นตอนที่ 1 เปิด Terminal หรือ Command Prompt

ตรงจุดนี้นักศึกษาสามารถใช้ Git Bash Terminal บน vscode หรือ ใช้ Git Bash หรือ Command Prompt ก็ได้

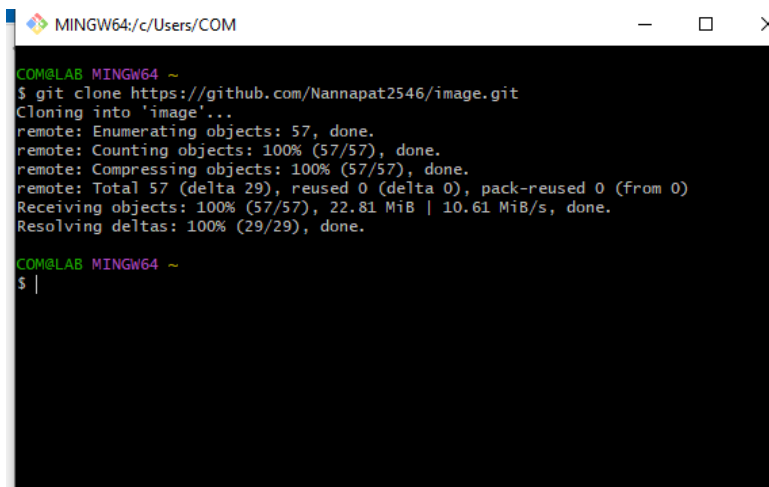
ขั้นตอนที่ 2 Clone repository ลงในเครื่อง

1. เข้าไปที่หน้า GitHub repository ของตนเอง
2. คลิกที่ปุ่ม Code (ปุ่มสีเขียว) แล้วคัดลอก URL ของ repository (เลือก HTTPS หรือ SSH)

ตัวอย่าง URL

`https://github.com/username/repository-name.git`

3. เปิด Terminal หรือ Command Prompt แล้วใช้คำสั่ง `git clone` เพื่อคัดลอก repository มายังเครื่องใหม่



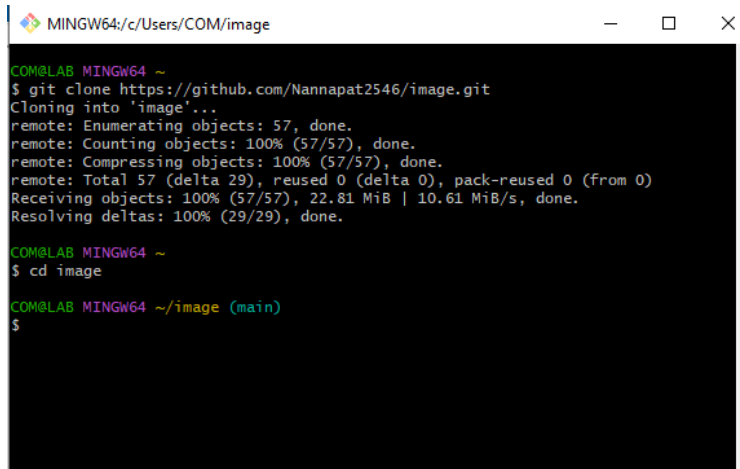
```
MINGW64: c:/Users/COM
COM@LAB MINGW64 ~
$ git clone https://github.com/Nannapat2546/image.git
Cloning into 'image'...
remote: Enumerating objects: 57, done.
remote: Counting objects: 100% (57/57), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 57 (delta 29), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (57/57), 22.81 MiB | 10.61 MiB/s, done.
Resolving deltas: 100% (29/29), done.
COM@LAB MINGW64 ~
$ |
```

ขั้นตอนที่ 3 เข้าไปในโฟลเดอร์ของ repository

1. ใช้คำสั่ง `cd` เพื่อเข้าไปในโฟลเดอร์ของ repository

`cd repository-name`

2. ตอนนี้คุณสามารถเริ่มต้นแก้ไขและทำงานใน repository นี้ได้

A screenshot of a terminal window titled 'MINGW64: c:/Users/COM/image'. The terminal shows the following commands and output:

```
COM@LAB MINGW64 ~  
$ git clone https://github.com/Nannapat2546/image.git  
Cloning into 'image'...  
remote: Enumerating objects: 57, done.  
remote: Counting objects: 100% (57/57), done.  
remote: Compressing objects: 100% (57/57), done.  
remote: Total 57 (delta 29), reused 0 (delta 0), pack-reused 0 (from 0)  
Receiving objects: 100% (57/57), 22.81 MiB | 10.61 MiB/s, done.  
Resolving deltas: 100% (29/29), done.  
  
COM@LAB MINGW64 ~  
$ cd image  
  
COM@LAB MINGW64 ~/image (main)  
$
```

กรณีที่ 2 มี repository ในเครื่องแล้ว (ใช้คำสั่ง `git pull`)

ขั้นตอนที่ 1 เปิด Terminal หรือ Command Prompt

1. เข้าไปที่โฟลเดอร์ที่มี repository ที่คุณ clone ลงมาแล้ว โดยใช้คำสั่ง `cd`

`cd repository-name`

ขั้นตอนที่ 2 ใช้คำสั่ง `git pull`

1. ใช้คำสั่ง `git pull` เพื่อดึงข้อมูลล่าสุดจาก GitHub

`git pull origin master`

```
MINGW64: c:/Users/COM/image
COM@LAB MINGW64 ~
$ git clone https://github.com/Nannapat2546/image.git
Cloning into 'image'...
remote: Enumerating objects: 57, done.
remote: Counting objects: 100% (57/57), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 57 (delta 29), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (57/57), 22.81 MiB | 10.61 MiB/s, done.
Resolving deltas: 100% (29/29), done.

COM@LAB MINGW64 ~
$ cd image

COM@LAB MINGW64 ~/image (main)
$ git pull origin main
From https://github.com/Nannapat2546/image
* branch      main      -> FETCH_HEAD
Already up to date.

COM@LAB MINGW64 ~/image (main)
$
```

ส่วนที่ 2 การดูแลประวัติการ Commit และเรียกคืน Commit ก่อนหน้า

ขั้นตอนที่ 1 ดูแลประวัติการ Commit

1. ใช้คำสั่ง git log เพื่อดูประวัติการ commit

git log

```
MINGW64: c:/Users/COM/image
Already up to date.

COM@LAB MINGW64 ~/image (main)
$ git log
commit e4ce621c00bc56fbd1514c24f441261a93420718 (HEAD -> main, origin/main, origin/HEAD)
Author: Nannapat2546 <655021000089@mail.rmutk.ac.th>
Date:   Mon Jul 22 16:36:08 2024 +0700

    Update img2.html

commit 82a254a439039bcf0afc1ec693ff31b0830015ec
Author: Nannapat2546 <655021000089@mail.rmutk.ac.th>
Date:   Mon Jul 22 16:26:20 2024 +0700

    Update img9.html

commit 5016d62ccc8b97d12a4be13ea9b17b681b63bd59
Author: Nannapat2546 <655021000089@mail.rmutk.ac.th>
Date:   Mon Jul 22 16:26:01 2024 +0700

    Update img8.html

commit 66267a1f74d3c623c459ffffcd1b53ed04716dd
Author: Nannapat2546 <655021000089@mail.rmutk.ac.th>
Date:   Mon Jul 22 16:25:46 2024 +0700

    ...skipping...
commit e4ce621c00bc56fbd1514c24f441261a93420718 (HEAD -> main, origin/main, origin/HEAD)
Author: Nannapat2546 <655021000089@mail.rmutk.ac.th>
Date:   Mon Jul 22 16:36:08 2024 +0700

    Update img2.html

commit 82a254a439039bcf0afc1ec693ff31b0830015ec
Author: Nannapat2546 <655021000089@mail.rmutk.ac.th>
Date:   Mon Jul 22 16:26:20 2024 +0700

    Update img9.html
```

3. คัดลอก commit hash ของ commit ที่คุณต้องการเรียกคืนหรือตรวจสอบ

ขั้นตอนที่ 2 การเรียกคืนไปยัง commit ก่อนหน้า

-การเรียกคืนค่าก่อนหน้า `git checkout 5016d6` (5016d6 คือ commit-hash)

```
MINGW64/c/Users/COM/image
COM@LAB MINGW64 ~
$ cd image

COM@LAB MINGW64 ~/image (main)
$ git checkout c46ae520
error: pathspec 'c46ae520' did not match any file(s) known to git

COM@LAB MINGW64 ~/image (main)
$ git checkout 5016d6
Note: switching to '5016d6'.

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 5016d62 Update img8.html
COM@LAB MINGW64 ~/image ((5016d62...))
$ |
```

-การเรียกคืนค่าอย่างถาวร `git reset --hard 5016d6`

```
COM@LAB MINGW64 ~/image ((5016d62...))
$ git reset --hard 5016d6
HEAD is now at 5016d62 Update img8.html
COM@LAB MINGW64 ~/image ((5016d62...))
$ |
```

- **git revert** ยกเลิก **commit** โดยไม่ลบประวัติ

```
MINGW64/c/Users/COM/image
COM@LAB MINGW64 ~
$ cd image
COM@LAB MINGW64 ~/image ((4ccc77b...))
$ git revert 5016d6
HEAD detached from 5016d62
nothing to commit, working tree clean
COM@LAB MINGW64 ~/image ((4ccc77b...))
$ |
```

คำถามท้ายใบงาน

1. อธิบายความแตกต่างระหว่างคำสั่ง **git clone** และ **git pull**

- **git clone** ใช้เพื่อทำสำเนา repository ที่มีอยู่ไปยังเครื่องของผู้ใช้
- **git pull** ใช้เพื่อดึงข้อมูลการเปลี่ยนแปลงจาก remote repository มายัง local repository ที่คุณได้ clone มาแล้ว

2. การใช้คำสั่ง **git revert** และ **git reset** มีความแตกต่างกันอย่างไร?

- **git revert** ใช้สำหรับย้อนกลับการเปลี่ยนแปลงที่เกิดจาก commit ที่มีอยู่
- **git reset** ใช้เพื่อรีเซ็ต pointer ของ branch ไปยัง commit ที่ระบุ

3. หากคุณทำงานร่วมกันในทีมและต้องการแก้ไขโค้ดที่ส่งผลกระทบต่อหลายคน คุณควรเลือกใช้คำสั่งใดระหว่าง **git reset** หรือ **git revert**?

- **git revert** เพราะจะช่วยให้ทุกคนในทีมสามารถทำงานได้อย่างราบรื่นและไม่เกิดความสับสนใน history ของ repository

สรุปได้ว่า หลังจากโหลด git แล้วทำการพิมพ์ใน command git bash ตามคำสั่งที่กำหนดมาให้ตามขั้นตอน จากนั้นทำการโคลน Repository ลงในเครื่อง และ ใช้คำสั่ง git clone ตามด้วยลิงค์งาน github ของเรา การเปิดโฟลเดอร์ใช้คำสั่ง cd ตามด้วยชื่อ repository จากนั้นใช้คำสั่ง git pull origin main จะเป็นการดึงข้อมูลจาก github ของงานนั้นมา การใช้คำสั่ง git log เป็นการดูประวัติที่เรา commit ส่งขึ้นบน github ในแต่ละครั้ง การใช้คำสั่ง git reset –hard ตามด้วยเลข 6 ตัวแรก ของเลข commit งานเรา และเป็นการรีเซ็ต pointer ของ branch ไปยัง commit ที่ระบุ คำสั่ง git revert ตามด้วยเลข 6 ตัวแรก ของเลข commit งานเรา จะเป็นการยกเลิกการเปลี่ยนแปลง