



01076105, 01076106

Object Oriented Programming

Object Oriented Programming Project

Git & Github



Git Installation

- ไปที่ <https://git-scm.com/>
- โหลดโปรแกรม Git (ถ้ายังไม่ได้ติดตั้ง)

Downloads



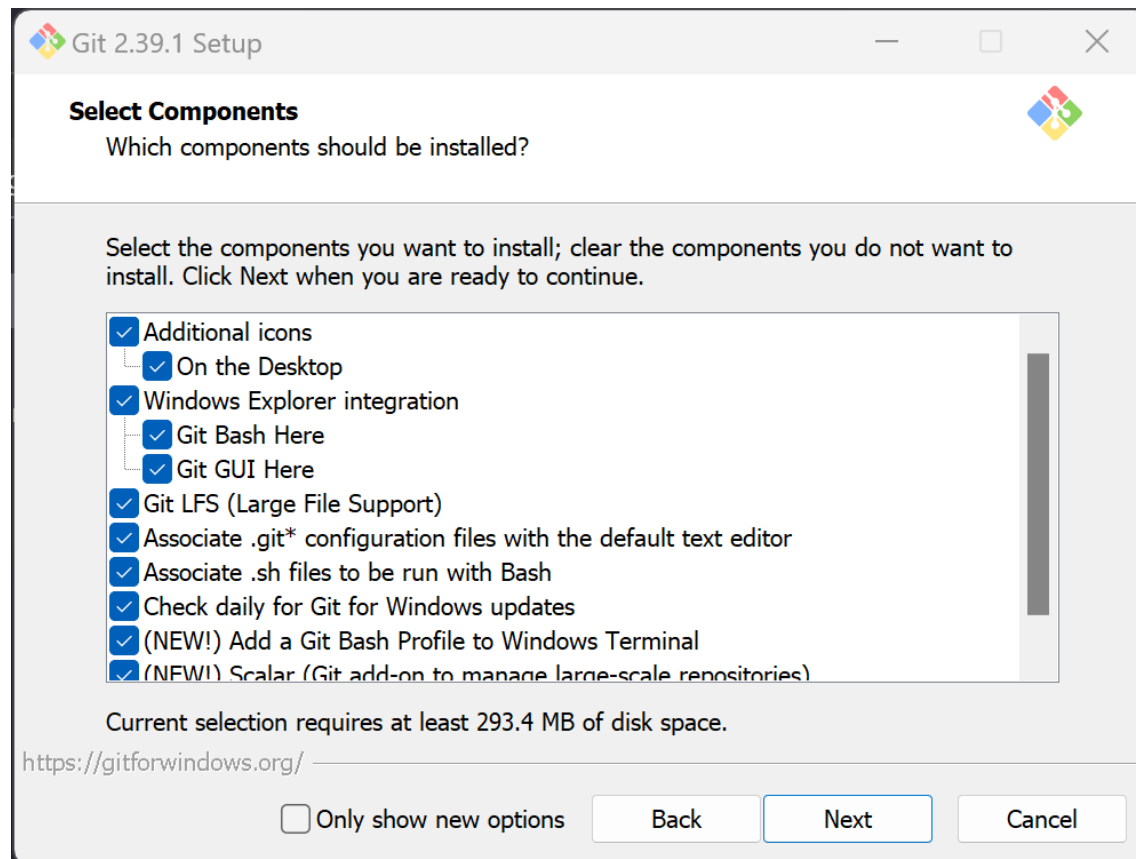
Older releases are available and the [Git source repository](#) is on GitHub.





Git Installation

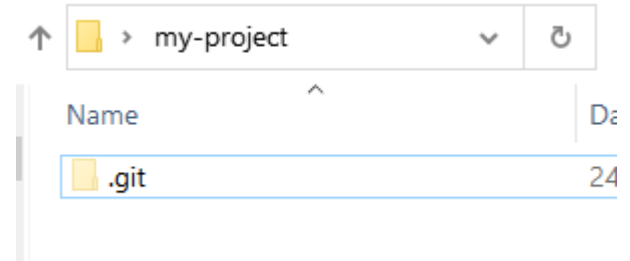
- ติดตั้งตามขั้นตอน โดยใช้ default option





Basic git operation

- Git เป็นโปรแกรม version control ที่นิยมใช้กันมาก หน้าที่ของ Git คือ เก็บ snapshot ของไฟล์ใน working folder ไปไว้ที่ repository
- ไปที่ desktop สร้าง folder ชื่อ my-project
- คลิกขวาที่ folder และเลือก Git Bash Here จะเปิดหน้าต่าง Git Bash
- ใช้คำสั่ง `git init` เพื่อเริ่มใช้ git ใน folder
- ให้ตั้ง config ของ folder ให้สามารถมองเห็น hidden
- จะพบว่ามี การสร้าง folder .git ขึ้นมา



```
khthana@PC-Terry MINGW64 ~/Desktop/my-project
$ git init
Initialized empty Git repository in C:/Users/khtha/Desktop/my-project/.git/

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ ls -al
total 68
drwxr-xr-x 1 khthana 197121 0 Feb 24 16:43 ./
drwxr-xr-x 1 khthana 197121 0 Feb 24 16:30 ../
drwxr-xr-x 1 khthana 197121 0 Feb 24 16:43 .git/

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Basic git operation

- เริ่มด้วยการกำหนดผู้ใช้และ email โดยใช้คำสั่ง

- `git config --global user.name <Name>`
- `git config --global user.email <Email>`

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git config --global user.name "Thana Hongsuwan"

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git config --global user.email khthana@hotmail.com

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```

- สามารถใช้คำสั่ง `git config --list` เพื่อตรวจสอบได้ (มีเครื่องหมาย – จำนวน 2 ตัว)



Basic git operation

- ให้สร้างไฟล์ขึ้นมา 2 ไฟล์ใน folder ชื่อ file1.txt กับ file2.txt

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ touch file1.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ echo "file2" > file2.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ ls
file1.txt  file2.txt
```

- คำสั่ง touch เป็นคำสั่งที่ใช้เปลี่ยน วัน-เวลา ของไฟล์ ถ้าใช้กับไฟล์ใหม่ จะเป็นการสร้างไฟล์ที่มีขนาด 0 ไบต์
- คำสั่ง echo เป็นคำสั่งที่เอาข้อความที่อยู่ในเครื่องหมาย “ ” ไปสร้างไฟล์และใส่ในไฟล์ที่สร้างขึ้น



Basic git operation

- ใช้คำสั่ง git status ซึ่งจะแสดงสถานะของไฟล์ใน git

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file1.txt
        file2.txt

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

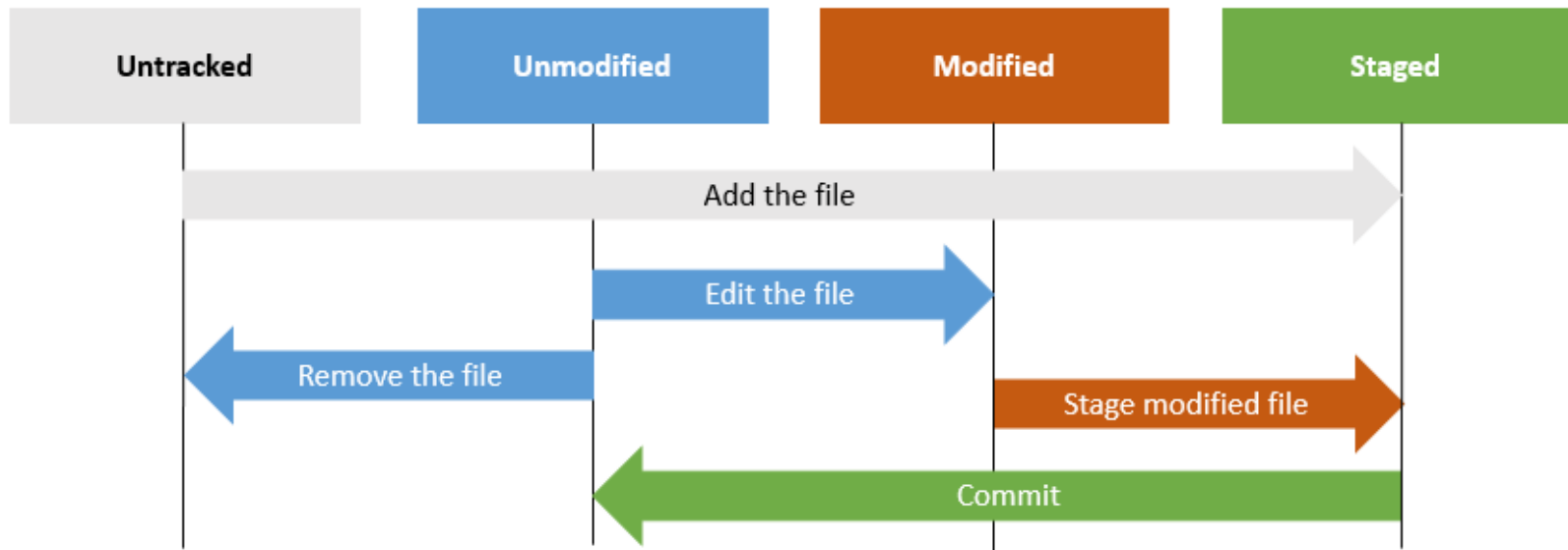
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```

- จะแสดงให้เห็นว่าไฟล์ทั้ง 2 ไฟล์อยู่ในสถานะ untracked



Basic git operation

- ไฟล์ใน git จะมี 4 สถานะ
 - untracked ยังไม่อยู่ใน “การติดตาม” ของ git
 - ถ้า add file เข้าสู่ git จะเข้าสู่สถานะ staged (“ติดตาม” แต่ยังไม่ทำ snapshot)
 - ถ้า commit จะทำ snapshot และเข้าสู่สถานะ unmodified
 - ถ้าไฟล์ที่ commit แล้ว มีการแก้ไขจะเข้าสู่สถานะ modified





Basic git operation

- ใช้คำสั่ง git add เพื่อนำไฟล์เข้าสู่สถานะ staged
- สัญลักษณ์ . มีความหมายว่าทุกไฟล์

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git add .
warning: LF will be replaced by CRLF in file2.txt.
The file will have its original line endings in your working directory

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   file1.txt
    new file:   file2.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Basic git operation

- ในขณะที่ไฟล์อยู่ในสถานะ staged หากมีการแก้ไข จะไปอยู่ในสถานะ modified
- ถ้าต้องการให้ไฟล์มาอยู่ในสถานะ staged ใหม่ ก็ให้ git add อีกครั้ง

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ echo "file1" >> file1.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   file1.txt
    new file:   file2.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   file1.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Basic git operation

- จะใช้คำสั่ง git commit เพื่อนำไฟล์เข้าสู่ repository โดยคำสั่ง commit จะต้องใส่ข้อความกำกับด้วย เมื่อ commit แล้วจะเกิด snapshot ของไฟล์ที่ commit

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git add .
warning: LF will be replaced by CRLF in file1.txt.
The file will have its original line endings in your working directory

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git commit -m "first commit"
[master (root-commit) 2b9040a] first commit
2 files changed, 3 insertions(+)
create mode 100644 file1.txt
create mode 100644 file2.txt

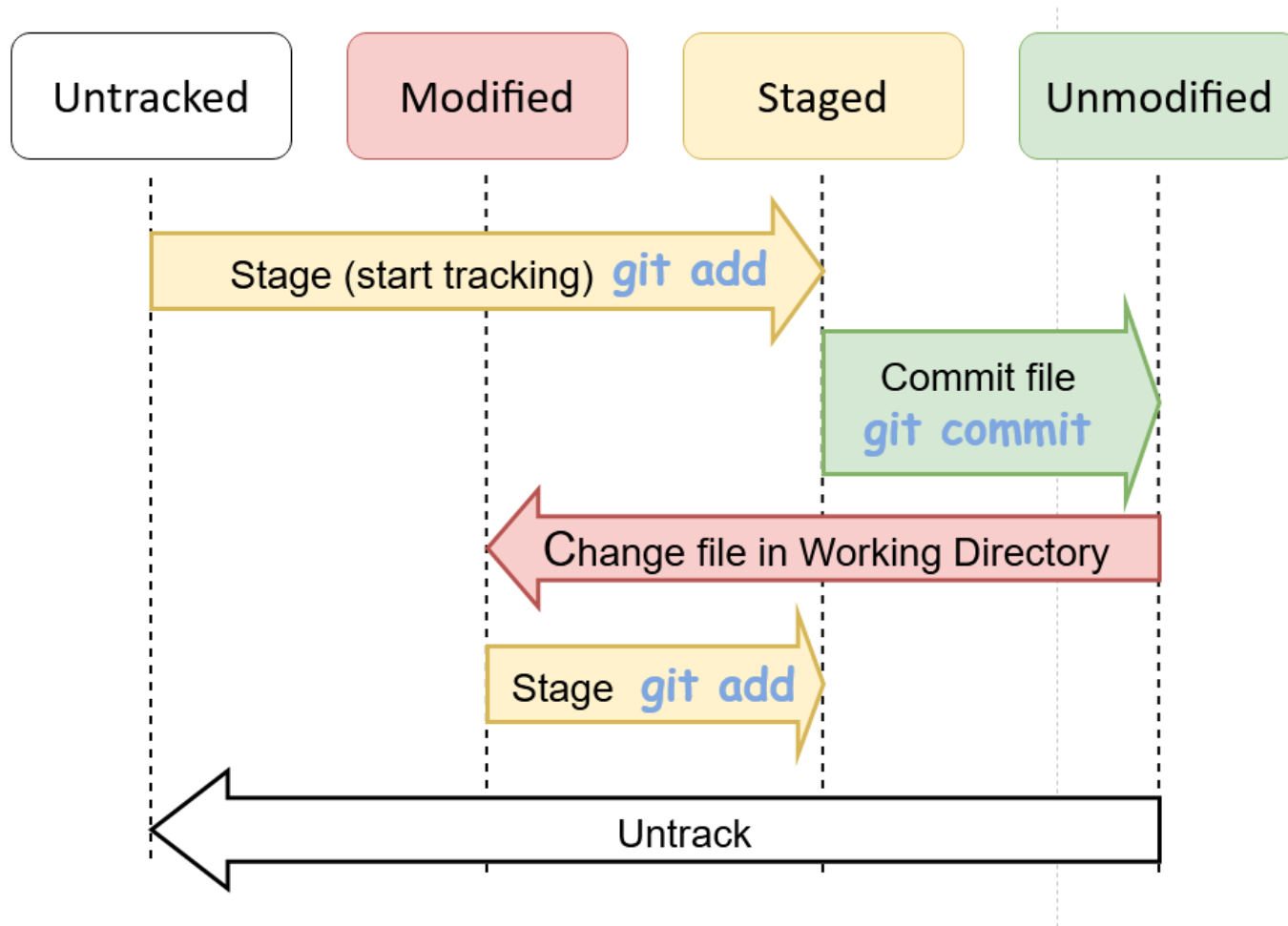
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git status
On branch master
nothing to commit, working tree clean

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Basic git operation

- สรุปสถานะและคำสั่งที่ใช้ในการเปลี่ยนสถานะไฟล์





Basic git operation

- เราสามารถใช้คำสั่ง git log ในการตรวจสอบ snapshot
- สมมติว่าเราสร้างไฟล์ที่ 3 ชื่อ file3.txt ไฟล์นี้จะอยู่ในสถานะ untracked

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git log
commit 2b9040aca03c2a64015b1810ad408fd97d386fc7 (HEAD -> master)
Author: khthana@hotmail.com <khthana@hotmail.com>
Date:   Fri Feb 24 18:26:42 2023 +0700

    first commit

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ touch file3.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file3.txt

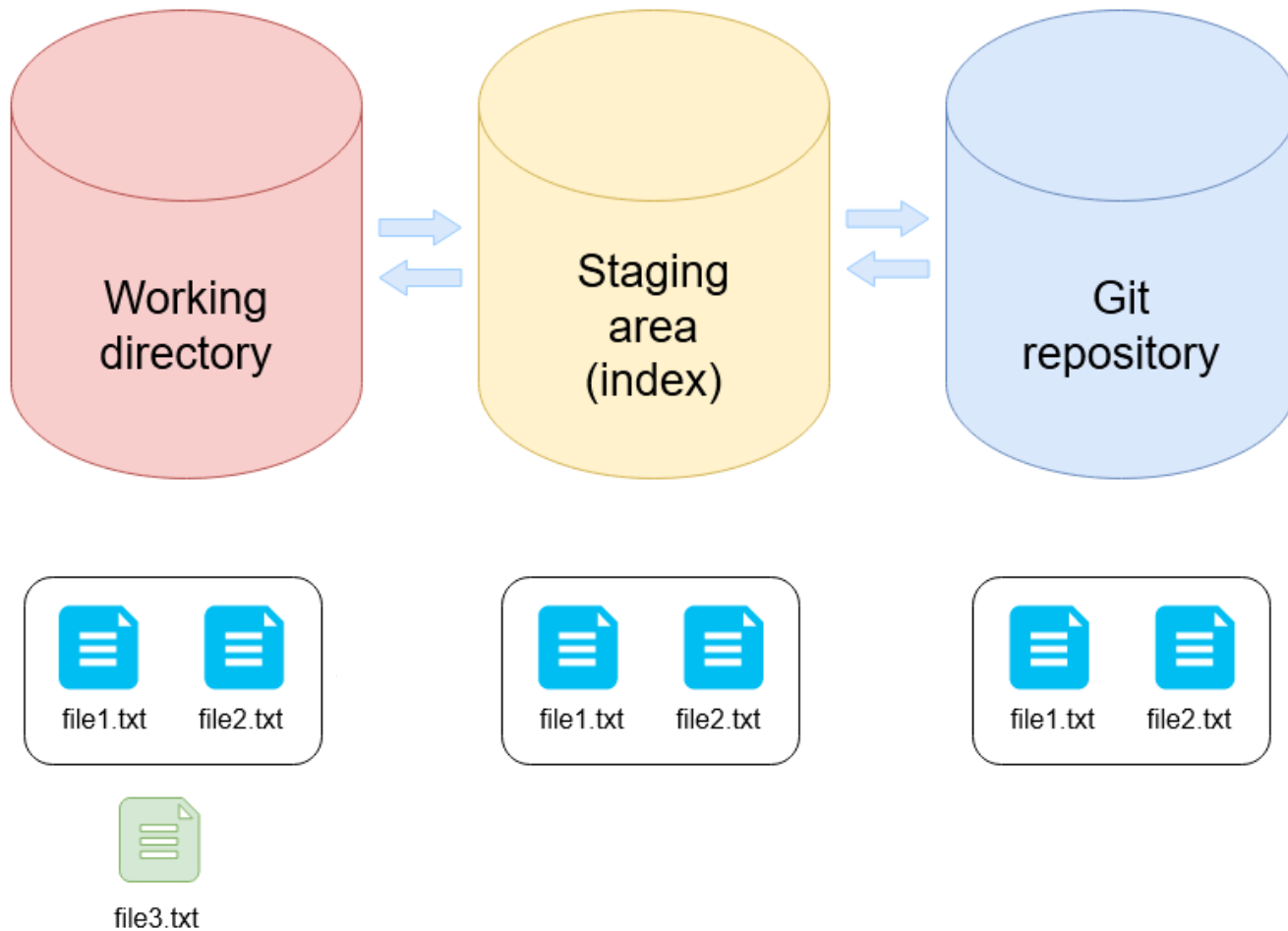
nothing added to commit but untracked files present (use "git add" to track)

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Basic git operation

- สถานะของไฟล์ ณ จุดนี้





Basic git operation

- จะสมมติสถานการณ์การทำงานเพิ่มเติม ดังนี้
 - แก้ไขไฟล์ file2.txt โดยเพิ่มข้อความ edited ต่อท้ายเข้าไป
 - ใช้ git add . เพื่อนำไฟล์ file3.txt และ file2.txt เข้าสู่สถานะ stage
 - จากนั้นให้ commit ครั้งที่ 2

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ echo "edited" >> file2.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git add .
warning: LF will be replaced by CRLF in file2.txt.
The file will have its original line endings in your working directory

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git commit -m "second commit"
On branch master
nothing to commit, working tree clean

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Basic git operation

- เมื่อใช้คำสั่ง git log จะพบว่ามี 2 commit ตามรูป
- ข้อมูลในกรอบสีแดง คือ ค่า hash ที่เป็นตัวแทนของแต่ละ commit

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git log
commit fee25f9f1b6cc3ccb8bf06a380e7b4c94eed6722 (HEAD -> master)
Author: khthana@hotmail.com <khthana@hotmail.com>
Date:   Fri Feb 24 20:14:12 2023 +0700
```

second commit

```
commit 2b9040aca03c2a64015b1810ad408fd97d386fc7
Author: khthana@hotmail.com <khthana@hotmail.com>
Date:   Fri Feb 24 18:26:42 2023 +0700
```

first commit

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```




Basic git operation

- เมื่อตรวจสอบไฟล์และข้อมูลในไฟล์ ก็จะได้พบดังนี้
- จะเห็นว่า มี 3 ไฟล์ และ ไฟล์ file2.txt มีการแก้ไข ซึ่งตรงกับการทำงานล่าสุด

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ ls -l
total 2
-rw-r--r-- 1 khthana 197121 12 Feb 24 18:21 file1.txt
-rw-r--r-- 1 khthana 197121 13 Feb 24 20:15 file2.txt
-rw-r--r-- 1 khthana 197121 0 Feb 24 18:31 file3.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ cat file2.txt
file2
edited

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Basic git operation

- สมมติว่าเรานึกได้ว่า มีความผิดพลาดบางอย่าง และ ต้องการจะกลับไปกลับที่สถานะ first commit สามารถใช้คำสั่ง git checkout เพื่อย้อนกลับไป first commit ได้

```
khthana@PC-Terry-MINGW64 ~/Desktop/my-project (master)
$ git checkout 2b9040
Note: switching to '2b9040'.

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 2b9040a first commit
```



Basic git operation

- เมื่อเราตรวจสอบไฟล์จะพบว่าไฟล์กลับมาอยู่ที่สถานะ first commit

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project ((2b9040a...))
$ ls -l
total 2
-rw-r--r-- 1 khthana 197121 12 Feb 24 18:21 file1.txt
-rw-r--r-- 1 khthana 197121 7 Feb 24 20:29 file2.txt

khthana@PC-Terry MINGW64 ~/Desktop/my-project ((2b9040a...))
$ cat file2.txt
file2

khthana@PC-Terry MINGW64 ~/Desktop/my-project ((2b9040a...))
$ |
```

- จะเห็นได้ว่า สามารถจะย้ายสถานะไปตาม snapshot หรือ commit ได้
- หากต้องการกลับมาที่ commit ล่าสุด ให้ใช้ git checkout master
- หากต้องการอยู่ที่สถานะก่อนหน้าถาวรให้ใช้คำสั่ง git reset [commit ก่อนหน้า]



Basic git operation

- git จะสร้างไฟล์ชื่อ master ซึ่งมีหน้าที่เป็น pointer ที่ชี้ที่ commit ล่าสุด ซึ่งจะเก็บไว้ที่ folder .git/refs/heads

```
khthana@PC-Terry MINGW64 ~/desktop/my-project (master)
$ cd .git/refs/heads

khthana@PC-Terry MINGW64 ~/desktop/my-project/.git/refs/heads (GIT_DIR!)
$ ls -l
total 1
-rw-r--r-- 1 khthana 197121 41 Jun 13 19:41 master

khthana@PC-Terry MINGW64 ~/desktop/my-project/.git/refs/heads (GIT_DIR!)
$ cat master
a3e1b9f4ee6806634b1599e62ffb2ee4f5886ac9
```



Basic git operation

- หน้าชี้ของ HEAD คือ ชี้ที่ commit ล่าสุด และ เป็น snapshot ที่จะแสดงผล

```
khthana@PC-Terry MINGW64 ~/desktop/my-project/.git (GIT_DIR!)
$ ls -l
total 13
-rw-r--r-- 1 khthana 197121  14 Jun 13 19:41 COMMIT_EDITMSG
-rw-r--r-- 1 khthana 197121  23 Jun 10 10:20 HEAD
-rw-r--r-- 1 khthana 197121 112 Jun 10 10:20 config
-rw-r--r-- 1 khthana 197121  73 Jun 10 10:20 description
drwxr-xr-x 1 khthana 197121   0 Jun 10 10:20 hooks/
-rw-r--r-- 1 khthana 197121 281 Jun 13 19:41 index
drwxr-xr-x 1 khthana 197121   0 Jun 10 10:20 info/
drwxr-xr-x 1 khthana 197121   0 Jun 13 14:25 logs/
drwxr-xr-x 1 khthana 197121   0 Jun 13 19:41 objects/
drwxr-xr-x 1 khthana 197121   0 Jun 10 10:20 refs/

khthana@PC-Terry MINGW64 ~/desktop/my-project/.git (GIT_DIR!)
$ cat HEAD
ref: refs/heads/master

khthana@PC-Terry MINGW64 ~/desktop/my-project/.git (GIT_DIR!)
$ cat refs/heads/master
a3e1b9f4ee6806634b1599e62ffb2ee4f5886ac9
```



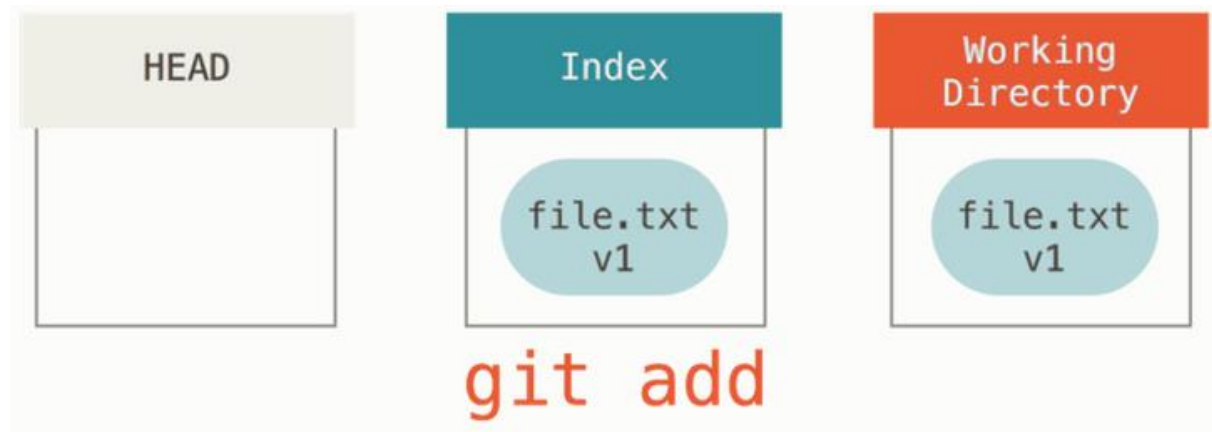
Git reset

- เพื่อให้เข้าใจการทำงานของ git reset จะสมมติเหตุการณ์ สร้างไฟล์ และ git add

สร้างไฟล์
file.txt



git add นำเข้า
Staging area

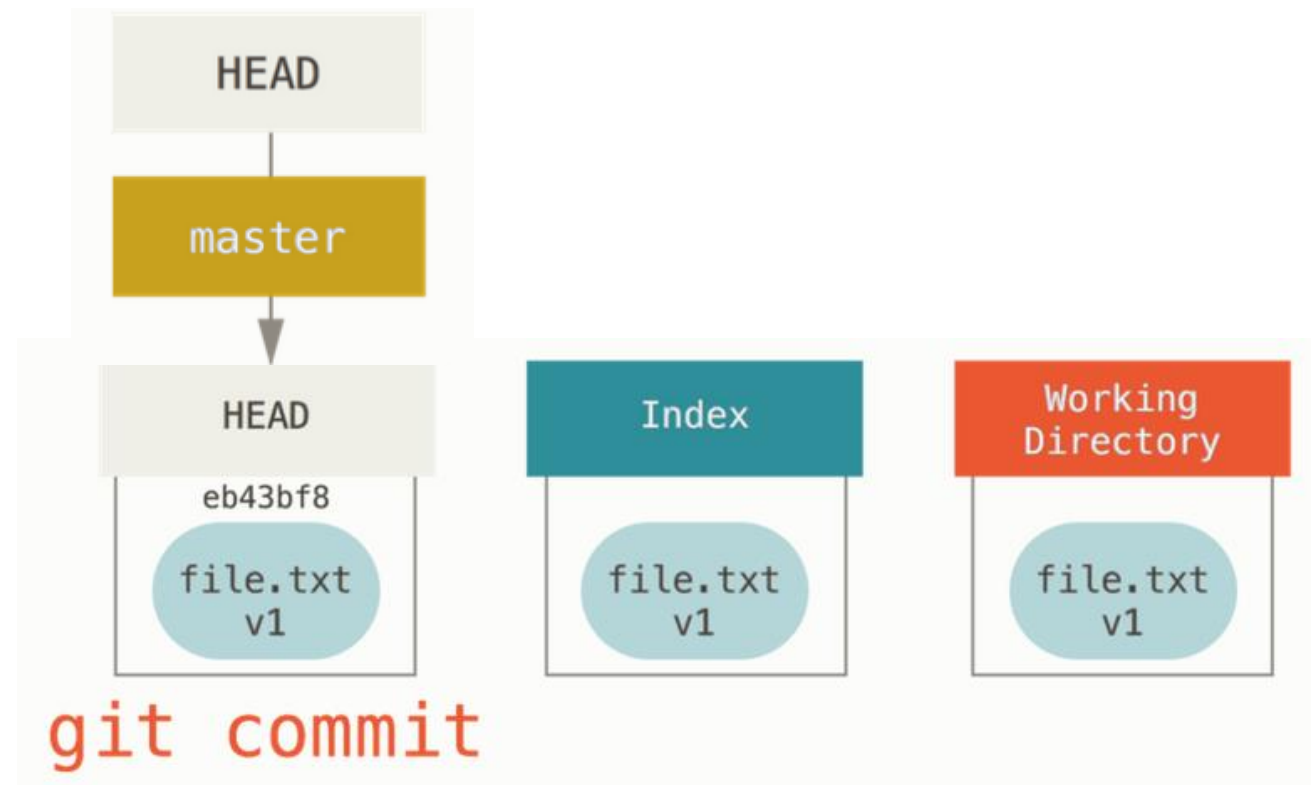




Git reset

- จากนั้น commit จะ copy ไฟล์เข้าสู่ repository และสร้าง commit (eb43bf8)

Commit
file.txt



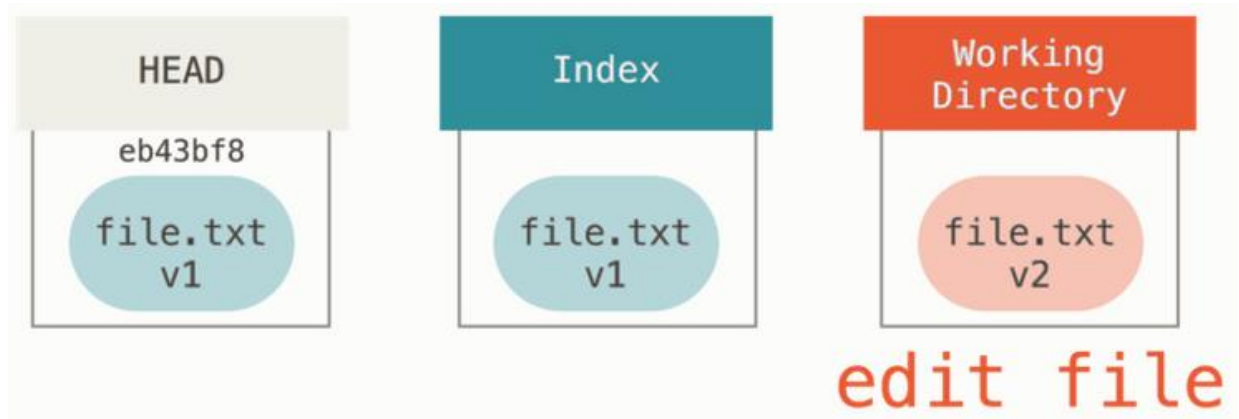


Git reset

- หากแก้ไขไฟล์ใน working directory จะเป็นไฟล์ v2 เมื่อใช้ git add ใน staged (index) ก็จะเป็น v2 ด้วย

แก้ไขไฟล์

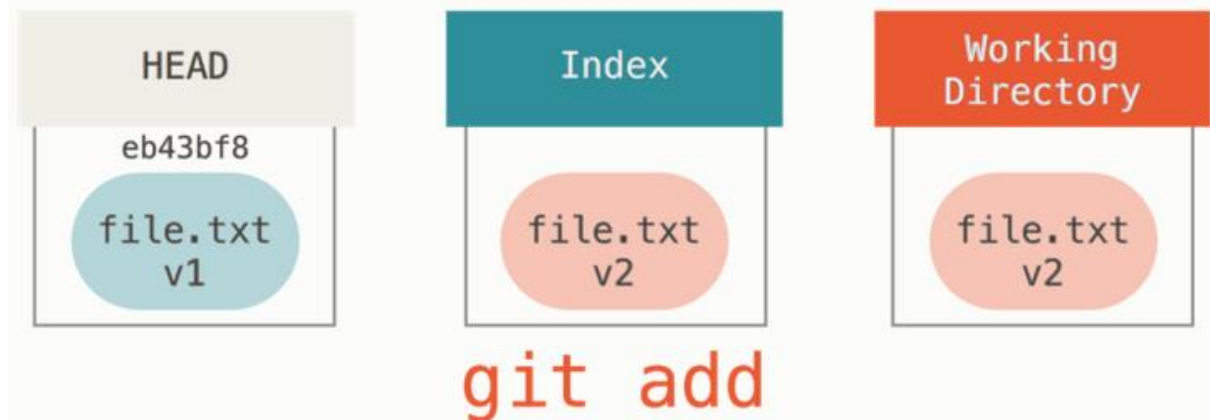
file.txt



git add

นำเข้าสู่

Staging area



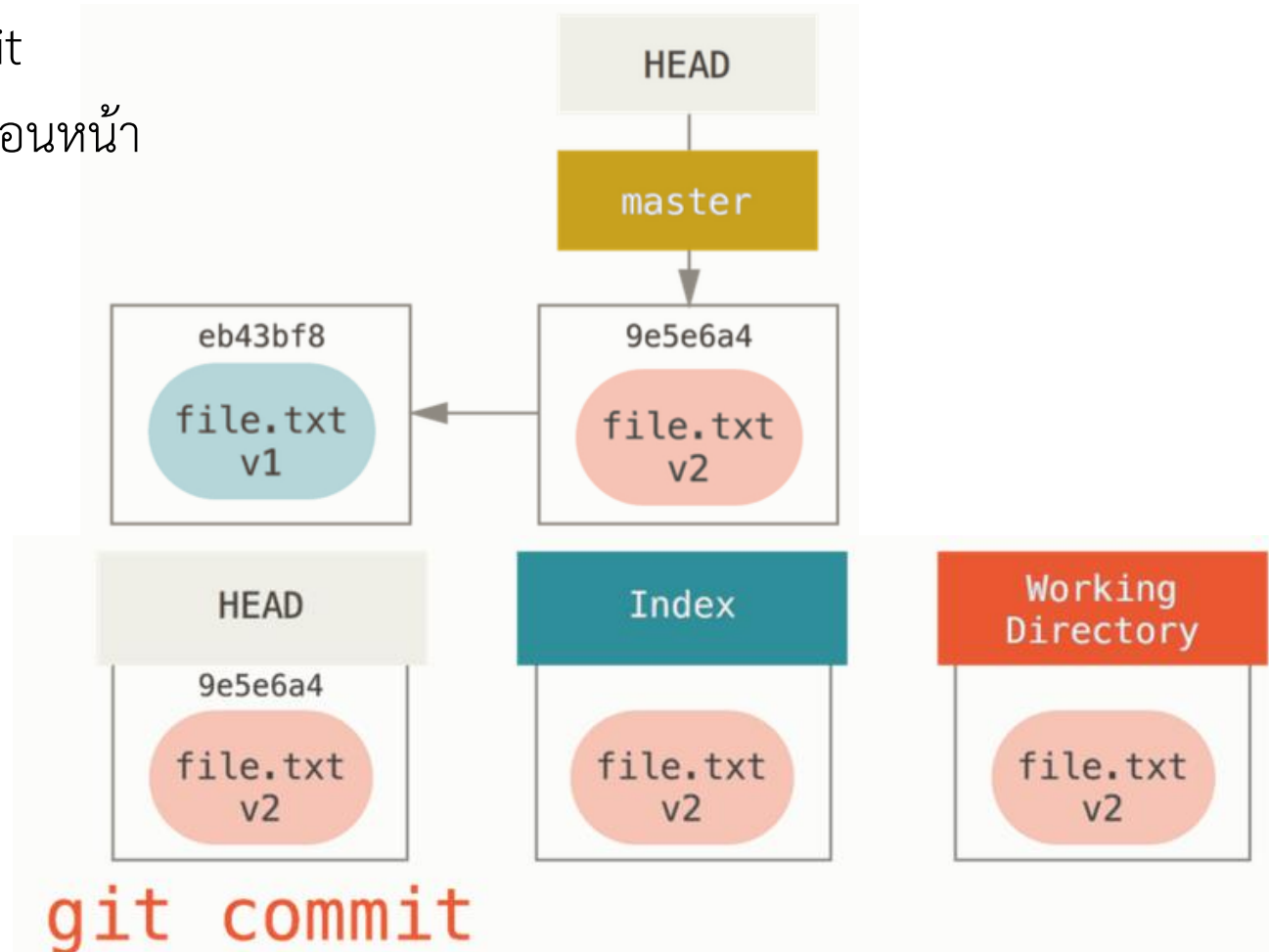


Git reset

- เมื่อ commit อีกครั้ง ก็จะสร้าง snapshot ใน repository (9e5e6a4)
- จะเห็นว่า commit ชี้ไปที่ commit ก่อนหน้า

Commit

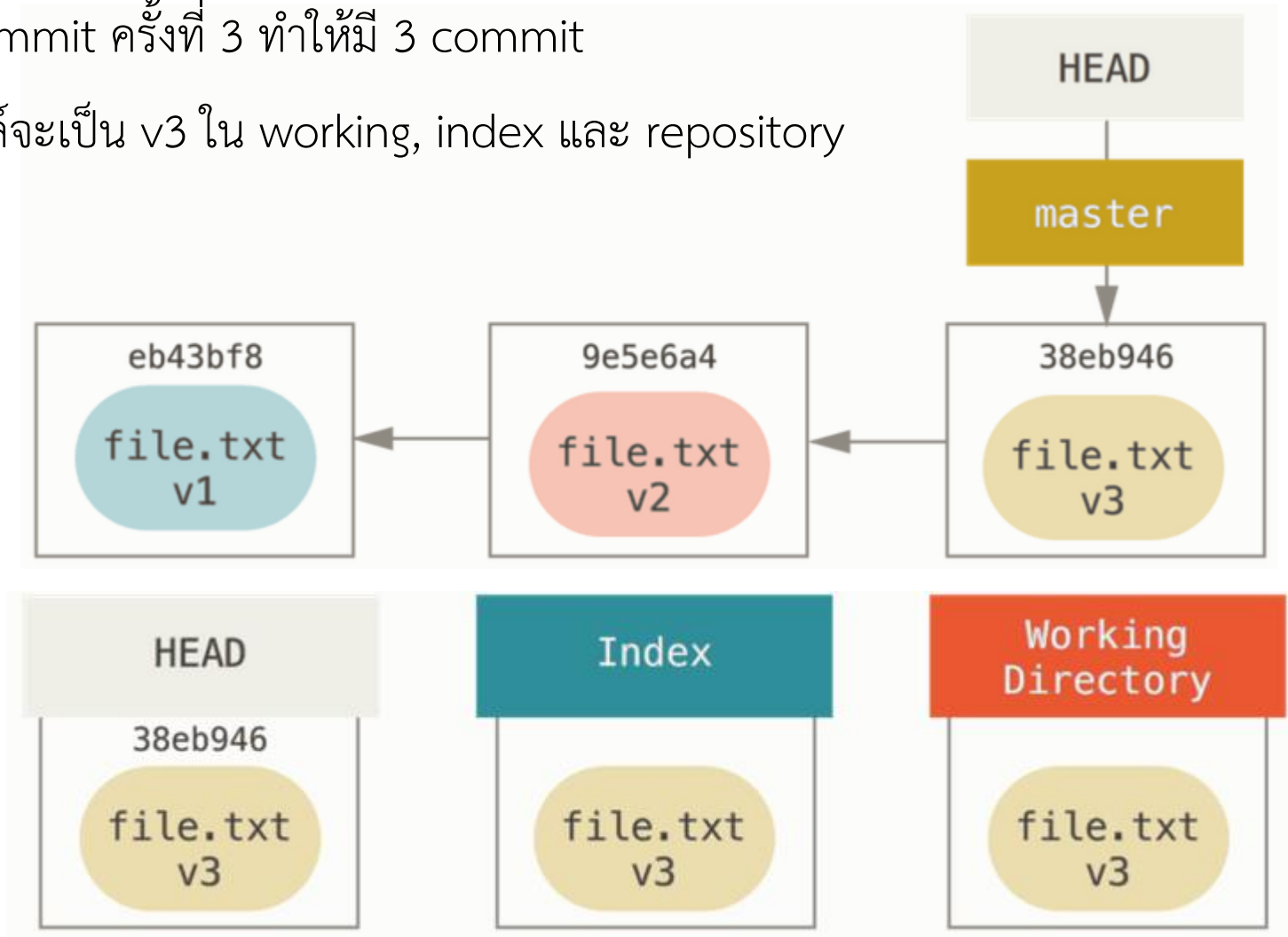
file.txt





Git reset

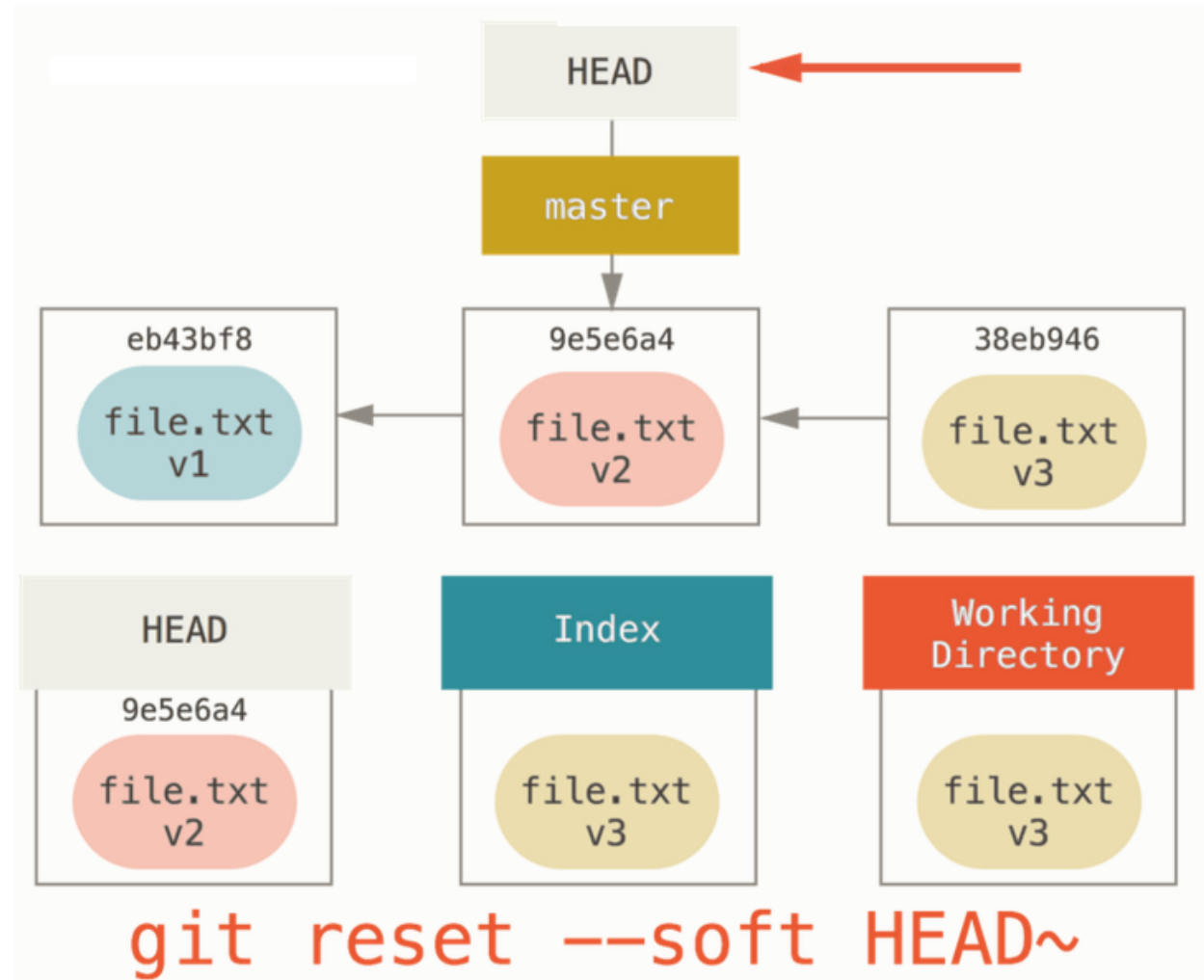
- Commit ครั้งที่ 3 ทำให้มี 3 commit
- ไฟล์จะเป็น v3 ใน working, index และ repository





Git reset

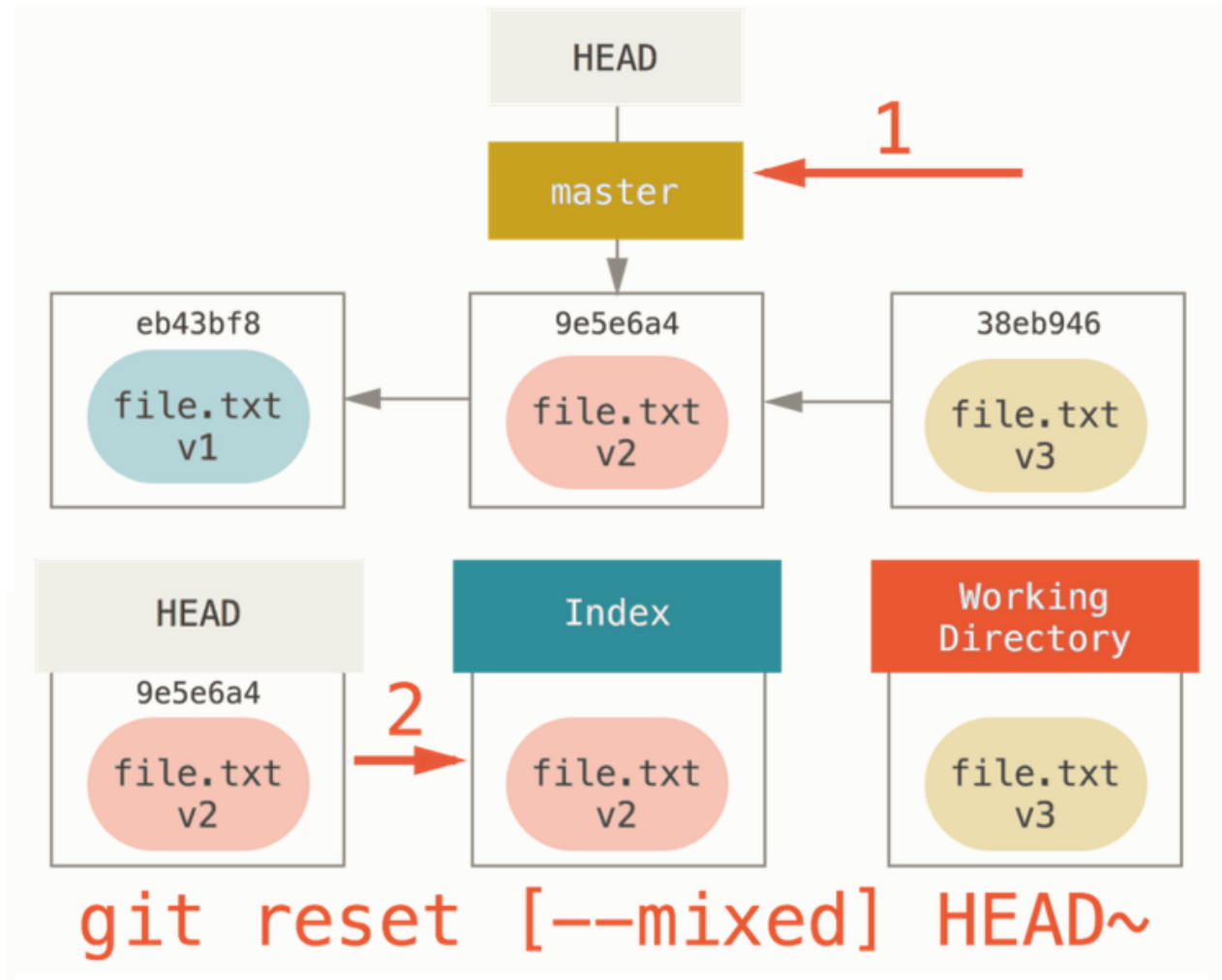
- หากต้องการย้ายไป version 2
- คำสั่ง `git reset --soft` จะเป็นการย้าย master และ HEAD ไปที่ commit ที่ระบุ
- เมื่อมีการ commit ใหม่ commit 38eb946 ก็จะไม่อยู่ในสายการพัฒนาอีก
- และจะถูกลบภายหลัง





Git reset

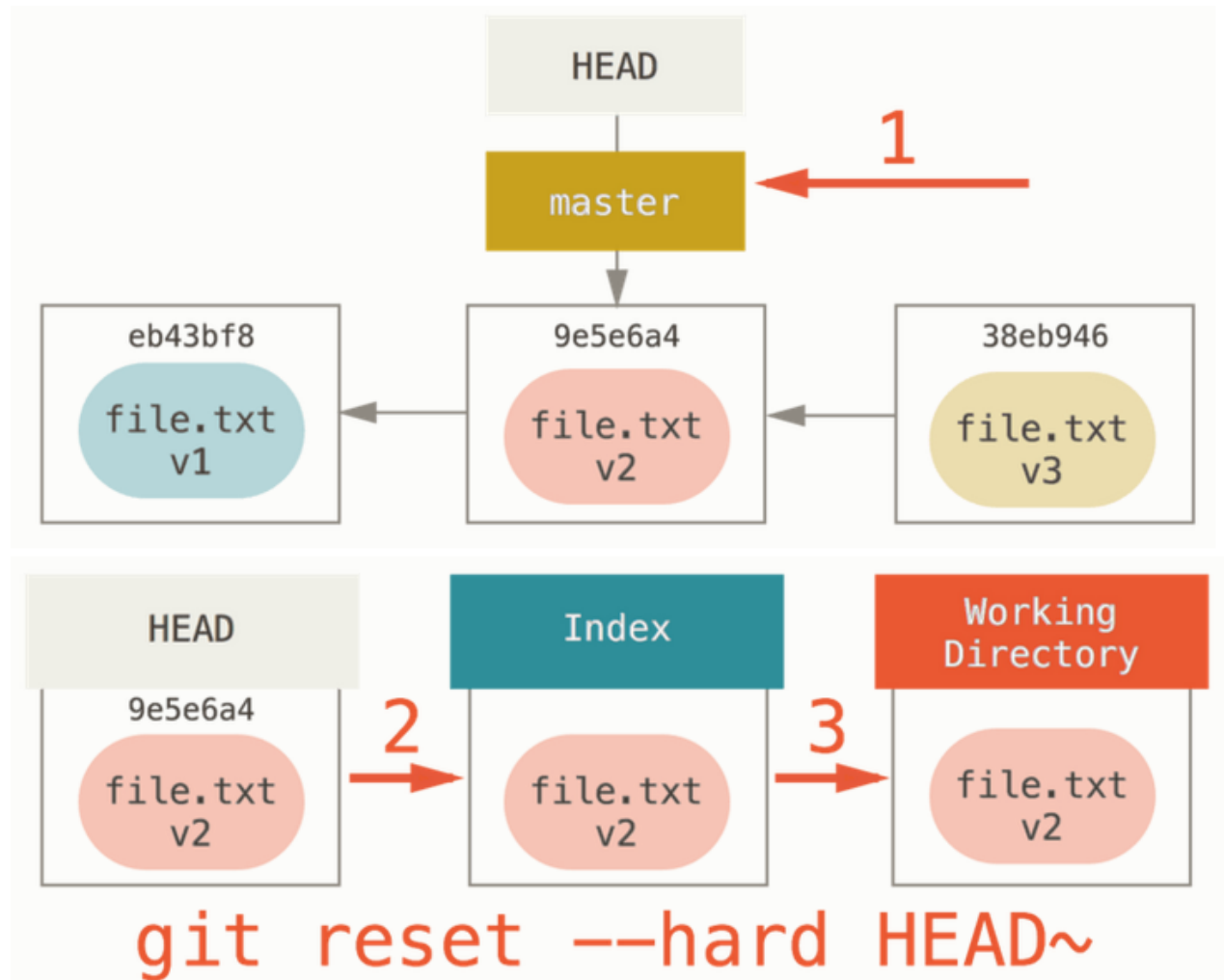
- สำหรับคำสั่ง
git reset --mixed
นอกจากจะย้าย HEAD
และ master แล้ว (1)
- ยังจะลบไฟล์ล่าสุดออก
จาก staging area
ด้วย





Git reset

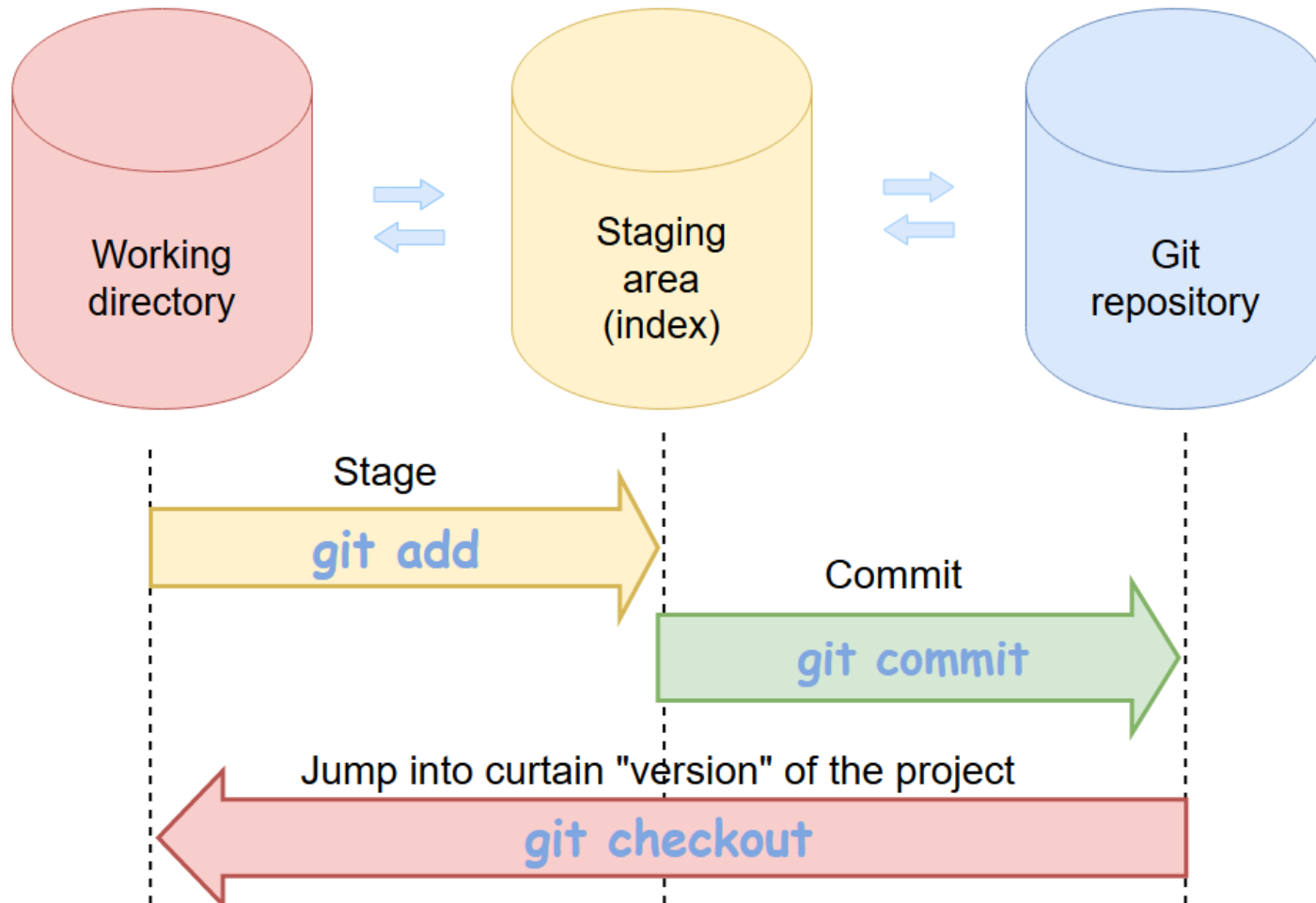
- สำหรับ
git reset –hard
นอกจากย้าย HEAD
และ master แล้ว ยัง
ลบไฟล์ออกจาก
staging area และ
working directory
อีกด้วย





Basic git operation

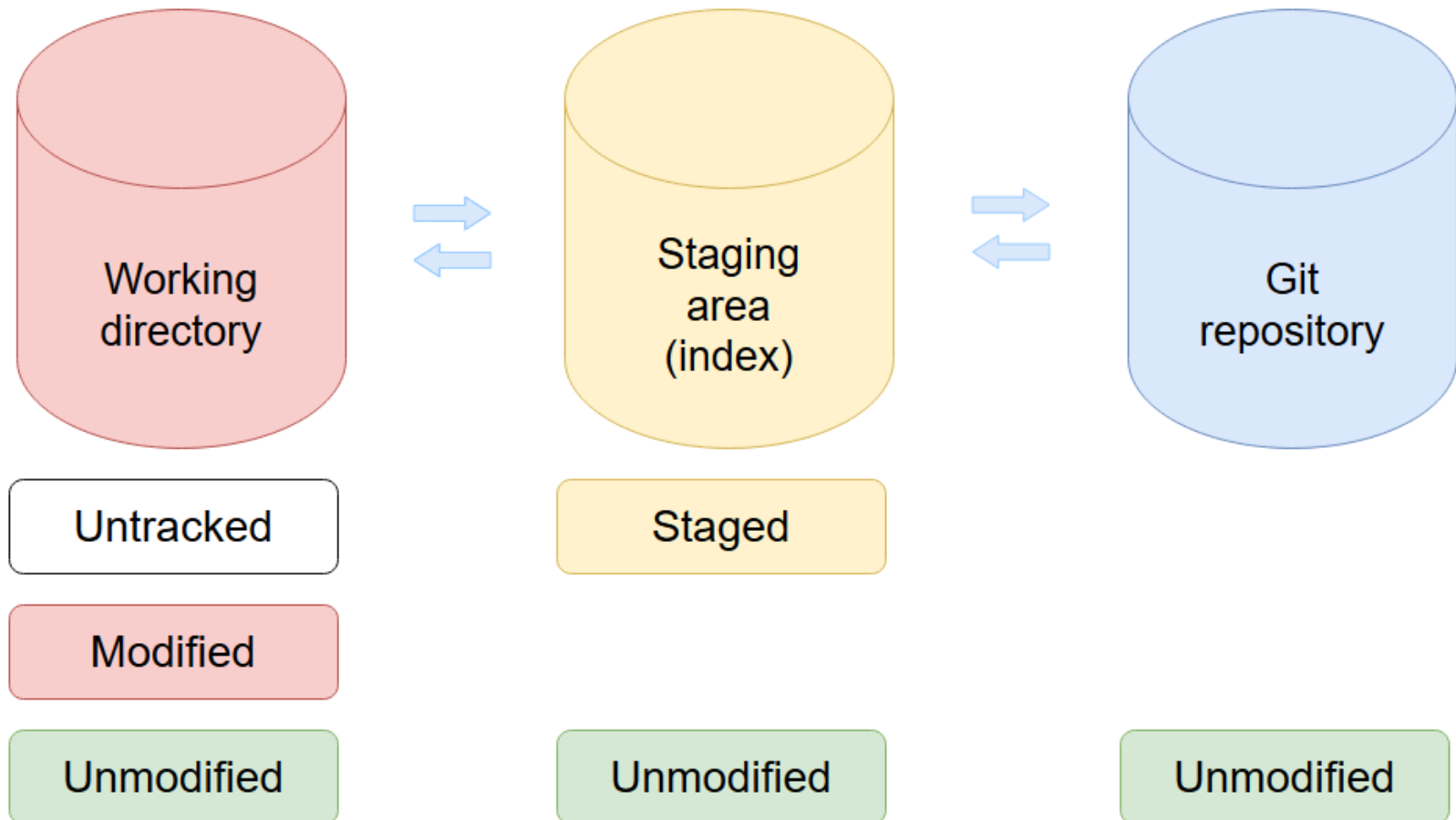
- สรุปคำสั่งที่ใช้เปลี่ยนสถานะ





Basic git operation

- สรุปสถานะ เทียบกับ ที่เก็บข้อมูลในแต่ละสถานะ





Basic git operation

- สรุปคำสั่ง
 - `git config` กำหนดค่าต่างๆ เช่น ชื่อ, email
 - `git add` เพิ่มไฟล์ลงใน Staging Area
 - `git status` แสดงสถานะปัจจุบันของ Git repository
 - `git commit` เขียนการเปลี่ยนแปลงลงใน Git repository
 - `git log` ประวัติการเปลี่ยนแปลง
 - `git checkout` เปลี่ยนไปยัง commit ต่างๆ
 - `git reset` ย้อนกลับการ commit



How to commit

- สิ่งที่ต้องพิจารณาในการ commit
 - **Make clean, single-purpose commits** ในการ commit แต่ละครั้ง ให้ commit ในแต่ละ unit of work โดยควรเป็นงานที่มีขอบเขตชัดเจน และเบ็ดเสร็จ (atomic) เช่น เพิ่มส่วนงาน หรือ แก้ bug หรืออื่นๆ และเป็นงานเดียว ไม่ควรเอาหลายๆ งานมารวมกันเป็น commit เดียวกัน
 - การทำเช่นนี้มีข้อดี คือ ง่ายกับคนในทีมจะตรวจสอบความเปลี่ยนแปลงของ code และทำให้ code review มีประสิทธิภาพดีกว่า นอกจากนั้นยังสะดวกต่อการย้อนกลับไป commit ก่อนหน้า



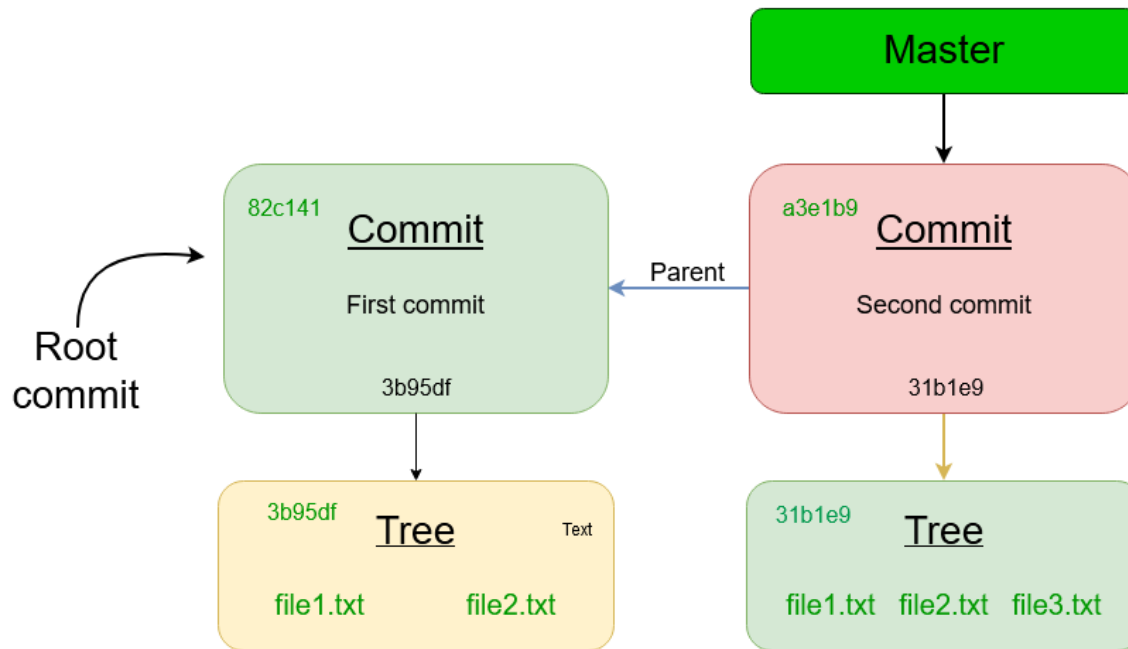
How to commit

- สิ่งที่ต้องพิจารณาในการ commit
 - **Write meaningful commit messages** การเขียน commit message ที่กระชับ มีความชัดเจนว่า commit นี้ มีเป้าหมายอะไร มีอะไรเปลี่ยนแปลง และเป็นระบบจะช่วยให้การทำงานมีประสิทธิภาพขึ้น
การเขียน commit message ที่ดี ควรประกอบด้วย
<type>**[optional scope]:** **<description>**
type หมายถึง ประเภทของการ commit เช่น feat (feature), fix (fix bug), chore, refactor, docs, style, test, perf (performance), build, revert
scope ทำหน้าที่บอกว่า commit นี้ ไปเกี่ยวกับอะไรบ้าง (เป็น option)
descriptive เป็นคำอธิบาย commit
 - **Commit early, commit often** ให้ commit ให้อยู่บ่อยๆ แม้ส่วนเล็กๆ เสร็จก็ commit ได้



Git Branch

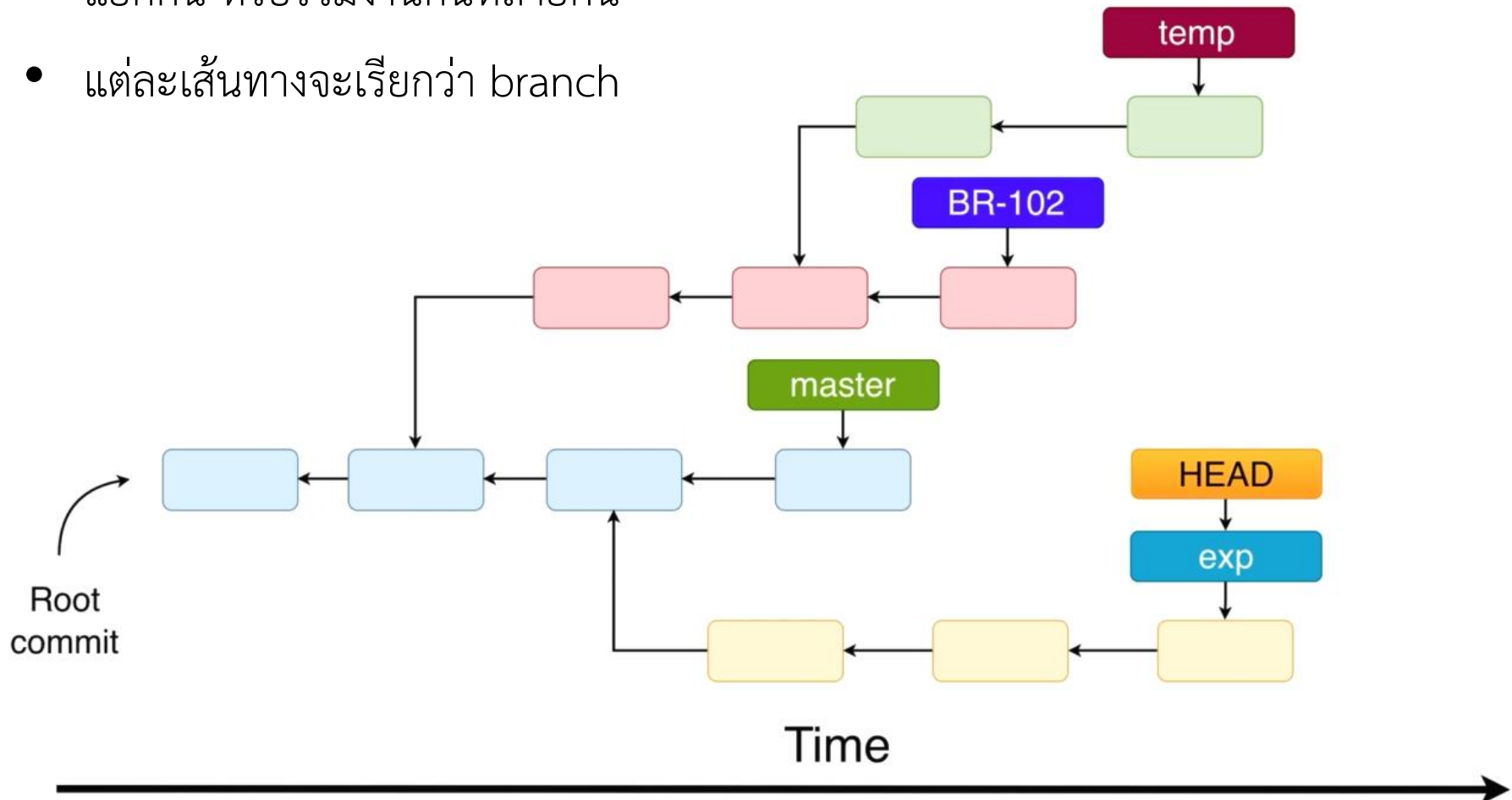
- โครงสร้างของ git ใน repository นั้น commit แต่ละอัน จะเชื่อมโยงไปยัง commit ก่อนหน้า โดยแต่ละ commit จะเก็บ tree ของไฟล์ที่อยู่ในแต่ละ commit อีกที
- จากรูปจะเห็นว่า ณ จุดนี้จะมี 2 commit โดย commit แรกมี file1.txt และ file2.txt และ commit ที่สองเพิ่มไฟล์ file3.txt เข้ามา





Git Branch

- ในบางครั้งเราสามารถแยกเส้นทางการพัฒนาโปรแกรม เพื่อพัฒนาแต่ละ feature แยกกัน หรือร่วมงานกันหลายคน
- แต่ละเส้นทางจะเรียกว่า branch





Git Branch

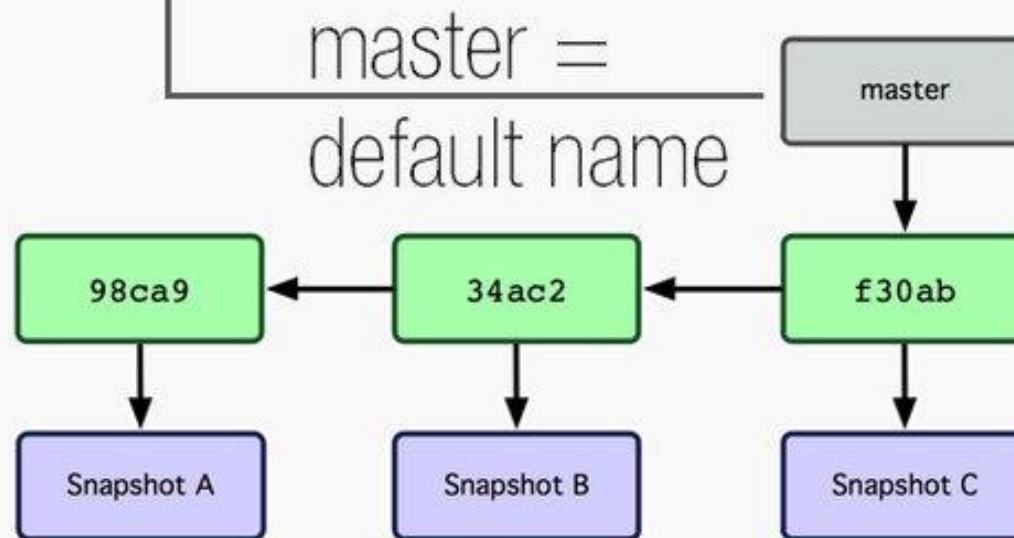
- Branch เป็นเพียง pointer ที่ชี้ไปยัง commit ล่าสุดของแต่ละเส้นทางเท่านั้น โดยแต่ละ commit จะมี parent pointer ที่บอกว่า commit ก่อนหน้าคือ commit ใด
- Default branch มีชื่อว่า master (แต่ใน Github จะเปลี่ยนไปเรียก default branch ว่า main ตั้งแต่ปี 2563)
- ในแต่ละ repository สามารถมีได้หลาย branch ตามความต้องการของผู้พัฒนา
- Pointer ที่จะชี้ไปแต่ละ branch จะเก็บอยู่ที่ `.git/refs/heads` ทั้งหมด
- เมื่อมีการ commit จะถือว่าทำบน branch ปัจจุบัน ซึ่ง branch pointer จะเลื่อนไปชี้ที่ commit ล่าสุด โดยไฟล์ชื่อ HEAD ทำหน้าที่เป็นบอกว่า branch ใด branch ปัจจุบัน โดยไฟล์ HEAD จะอยู่ที่ `.git/HEAD` ปกติค่า Default จะอยู่ที่ `.git/refs/heads`
- สามารถเปลี่ยน branch โดยใช้คำสั่ง `git checkout <branch>`



Git Branch

- จากรูปจะเห็นว่า branch คือ pointer ที่จะใช้ในการสร้าง commit อันต่อไป

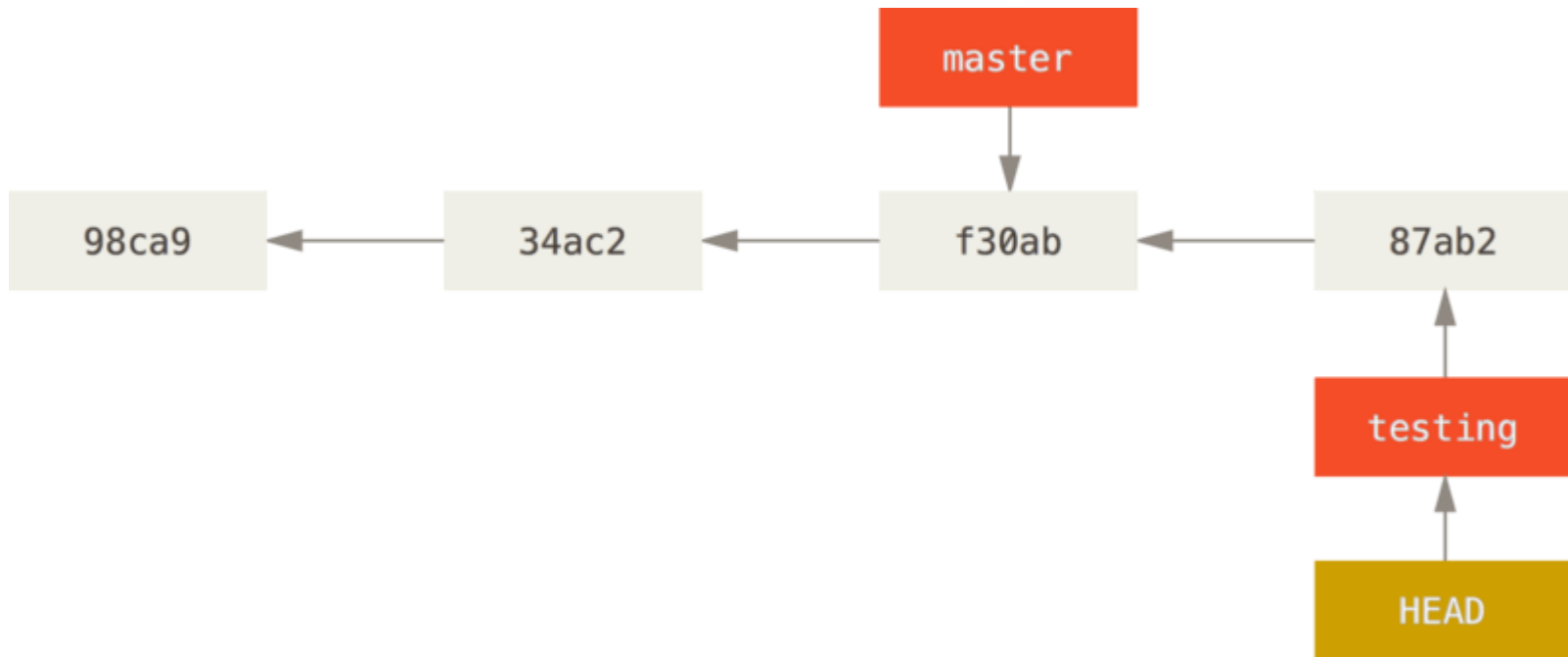
Branch = Pointer to a Commit





Git Branch

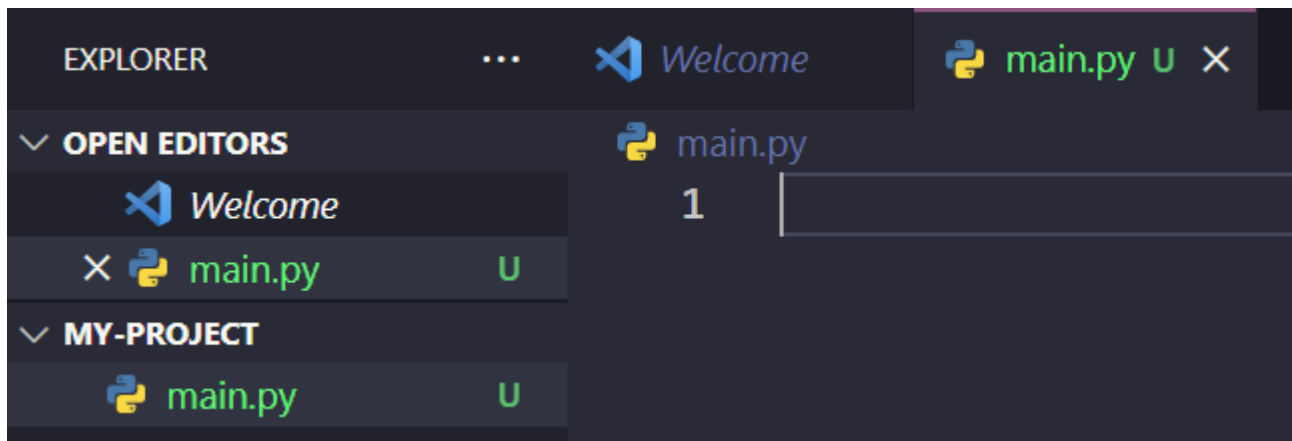
- จากรูปจะเห็นว่าการสร้าง branch ใหม่ที่ commit f30ab โดยชื่อว่า testing และ เป็น default branch เนื่องจากไฟล์ HEAD ชี้ที่ testing ดังนั้นเมื่อมีการ commit จะ เป็นการ commit ต่อจาก 87ab2






การใช้ Git ใน Visual Studio Code

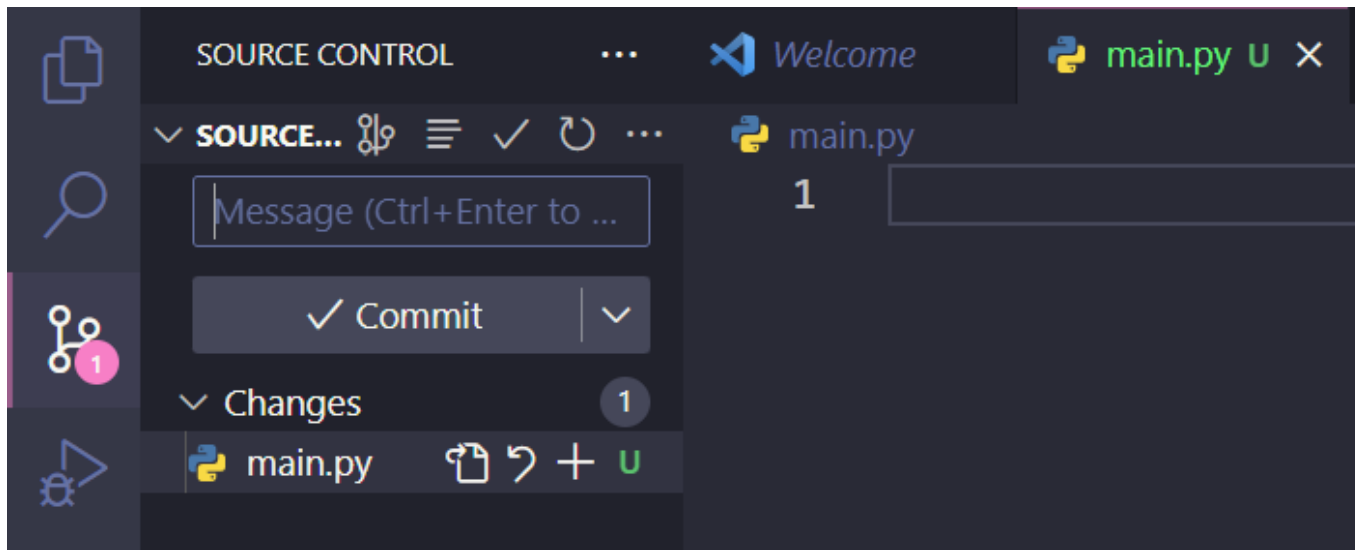
- ใน Editor หรือ IDE รุ่นใหม่ๆ มักจะรวมการทำงานของ git เข้ารวมไว้ในโปรแกรม
- จะยกตัวอย่าง VS Code ให้สร้าง folder ใหม่ ชื่อ my-project จากนั้นรันคำสั่ง git init ใน folder แล้วใช้ VS Code เปิด folder แล้วสร้างไฟล์ main.py จะแสดงหน้าจอดังรูป
- จะเห็นว่ามีย่อ U มีความหมายว่า Untracked






การใช้ Git ใน Visual Studio Code

- เมื่อคลิกที่สัญลักษณ์  จะแสดงหน้าจอ (ให้ลง GitLens เพิ่ม)



- หากกดเครื่องหมาย + จะเทียบเท่ากับรันคำสั่ง `git add` กับไฟล์นั้น คือนำไฟล์นั้นเข้าสู่สถานะ stage
- หลังจาก `git add` หากป้อนข้อความลงในช่องด้านบน และ กดเครื่องหมาย  จะเทียบเท่ากับคำสั่ง `git commit -m "message"`



Git ignore

- เนื่องจากจะมีไฟล์ของ Editor หรือ IDE อยู่ใน folder ของเราด้วย ซึ่งเราไม่ต้องการให้ไฟล์เหล่านั้นอยู่ใน repository เนื่องจากไม่ใช่ code ที่เขียนขึ้น
- ใน git จะมีไฟล์ .gitignore ทำหน้าที่บอกว่า File หรือ Folder ไດ ที่ไม่สนใจ
- ในการใช้งานให้ไปที่ <https://www.toptal.com/developers/gitignore> แล้วป้อนข้อมูลภาษา และ Editor/IDE ที่ใช้ จะสร้าง gitignore มาให้

gitignore.io

สร้างไฟล์ .gitignore ที่มีประโยชน์สำหรับโปรเจกต์ของคุณ

Python x

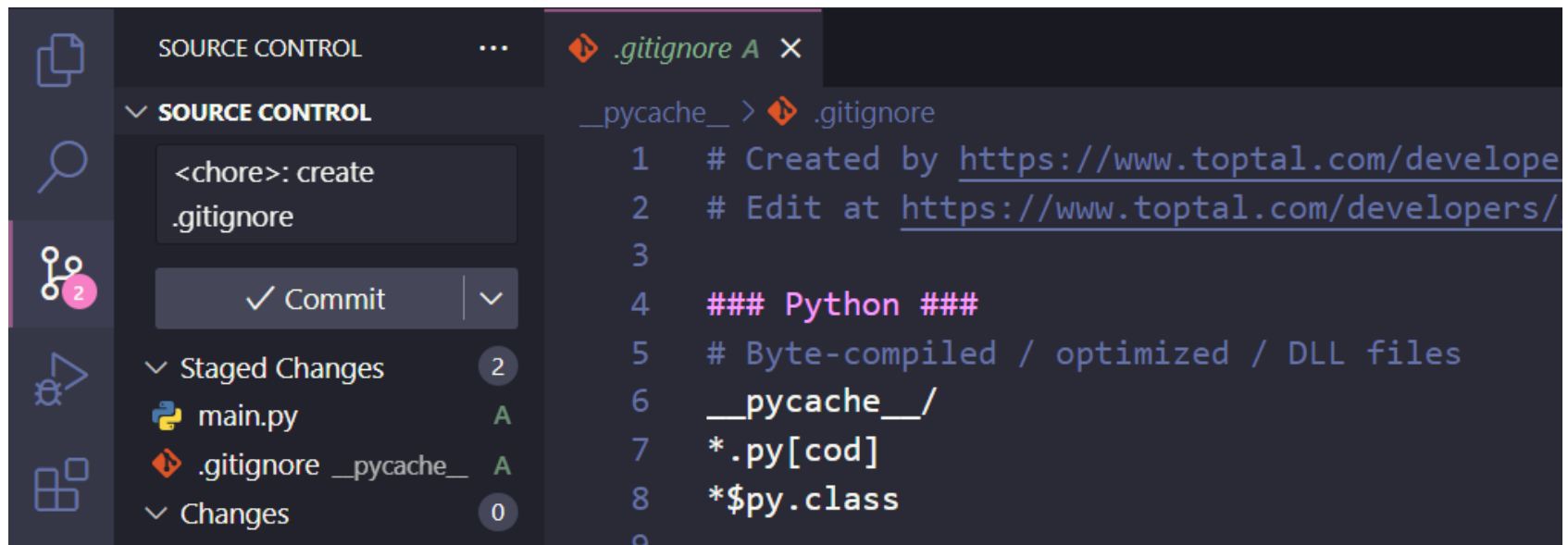
VisualStudioCode x

สร้าง



Git ignore

- จาก folder my-project ที่ว่างอยู่ ให้สร้างไฟล์ .gitignore
- จากนั้นให้เอาข้อมูลที่สร้างจากการทำงานใน slide ก่อนหน้ามาใส่ในไฟล์ .gitignore
- จากนั้นให้ add ไฟล์ .gitignore เข้าสู่ stage
- และ commit ครั้งที่ 1 ใช้ message เป็น <chore>: create .gitignore



The screenshot shows the VS Code Source Control panel on the left and the .gitignore file content on the right.

Source Control Panel:

- Panel title: SOURCE CONTROL
- Section: SOURCE CONTROL
- Commit message input: <chore>: create .gitignore
- Commit button: ✓ Commit
- Staged Changes (2 items):
 - main.py (A)
 - .gitignore __pycache__ (A)
- Changes (0 items)

.gitignore File Content:

```
__pycache__ > .gitignore
1 # Created by https://www.toptal.com/develop
2 # Edit at https://www.toptal.com/developers/
3
4 ### Python ###
5 # Byte-compiled / optimized / DLL files
6 __pycache__/
7 *.py[cod]
8 *$py.class
```



Git Branch

- จะขอสมมติสถานการณ์ ดังนี้
 - สร้าง class CinemaSystem เพื่อเป็นคลาสใหญ่สุดของโรงภาพยนตร์ ไว้ในไฟล์ cinema.py ทำหน้าที่เก็บโรงภาพยนตร์ โดยมี method add_cinema และ show_cinema

```
class CinemaSystem:
    def __init__(self):
        self.__cinema = []

    def add_cinema(self, cinema):
        if isinstance(cinema, Cinema):
            self.__cinema.append(cinema)

    def show_cinema(self):
        if self.__cinema is not None:
            for c in self.__cinema:
                print(c)
```



Git Branch

- จากนั้น
 - สร้าง class Cinema สำหรับเก็บข้อมูลโรงภาพยนตร์ในแต่ละสาขา

```
class Cinema:
    def __init__(self, name, location, total_cinema_hall=0):
        self.__name = name
        self.__location = location
        self.__total_cinema_hall = total_cinema_hall
        self.__halls = [] # List of CinemaHall

    def add_halls(self, halls):
        if isinstance(halls, list):
            self.__halls.extend(halls)
        else:
            self.__halls.append(halls)

    def __str__(self):
        return self.__name+" in location " \
            + self.__location+" has total cinema hall " \
            + str(self.__total_cinema_hall)
```



Git Branch

- เมื่อเสร็จแล้ว ลองสร้างโรงภาพยนตร์ 1 สาขา โดยเขียนใน main.py
- รันแล้วพบว่าทำงานได้ ถือว่าเป็น unit of work ก็จะได้ commit โดยใช้ commit message เป็น <feat> : add/test class CinemaSystem and Cinema

The screenshot shows a code editor with a commit message box on the left and a Python file named main.py on the right. The commit message box contains the text "<feat> : add/test class CinemaSystem and Cinema" and a "Commit" button. The Python code in main.py is as follows:

```
1 from cinema import CinemaSystem, Cinema
2
3 def main():
4     cinema_system = CinemaSystem()
5     cinema_central = Cinema("Central World", "Central World")
6     cinema_system.add_cinema(cinema_central)
7     cinema_system.show_cinema()
8
9
10 if __name__ == "__main__":
11     main()
```



Git Branch (Extension GitLens)

- ตอนนี้ มี 2 commit ตามรูป

BRANCH / TAG	GRAPH	COMMIT MESSAGE	AUTHOR	COMMIT DATE / TI...	SHA
✓ master		<feat> : add/test class CinemaSystem and Cine...	You	40 seconds ago	2f04b1e
		<chore>: create .gitignore	You	16 hours ago	52c1525


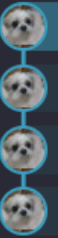
- จะพัฒนาโปรแกรมต่อ โดยเพิ่มคลาส CinemaHall และปรับปรุง Code ใน main.py ให้เพิ่ม Cinema Hall ของ Central World จำนวน 15 โรงฉาย จากนั้น commit อีกครั้ง จะได้ 3 commit ตามรูป

BRANCH / TAG	GRAPH	COMMIT MESSAGE	AUTHOR	COMMIT DATE / TI...	SHA
✓ master		<feat>: add/test class CinemaHall and data	You	28 seconds ago	c710d5f
		<feat> : add/test class CinemaSystem and Cine...	You	4 hours ago	2f04b1e
		<chore>: create .gitignore	You	21 hours ago	52c1525



Git Branch

- จะทำงานต่อไป โดยเพิ่ม class CinemaHallSeat และสร้างเก้าอี้ ในทุก CinemaHall โดยกำหนดให้ทุกโรงมี 252 เก้าอี้ โดยเป็นเก้าอี้แบบ Premium จำนวน 120 ที่นั่ง (G1-M20) เก้าอี้แบบ Prime จำนวน 120 ที่นั่ง (A1-F20) และเก้าอี้แบบ Suite จำนวน 12 ที่นั่ง (AA1-AA6)
- เมื่อเรียบร้อยแล้วก็ commit อีกครั้ง จะได้ 4 commit ตามรูป

BRANCH / TAG	GRAPH	COMMIT MESSAGE	AUTHOR	COMMIT DATE / TI...	SHA
✓  master		<feat>: add/test class CinemaHallSeat and data	You	17 seconds ago	4392d23
		<feat>: add/test class CinemaHall and data	You	2 hours ago	c710d5f
		<feat> : add/test class CinemaSystem and Cine...	You	6 hours ago	2f04b1e
		<chore>: create .gitignore	You	22 hours ago	52c1525



Git Branch

- คลาส CinemaHall

```
class CinemaHall:
    def __init__(self, name, projection_system, total_seat):
        self.__name = name
        self.__projection_system = projection_system
        self.__total_seat = total_seat
        self.__shows = [] # List of Show
        self.__seats = []

        # Add A1 to F20
        for i in [chr(x) for x in range(ord('A'), ord('F')+1)]:
            for j in range(1,21):
                self.add_seat(CinemaHallSeat(str(i),str(j),SeatType.PRIME))
        # Add G1 to M20
        for i in [chr(x) for x in range(ord('G'), ord('M')+1)]:
            for j in range(1,21):
                self.add_seat(CinemaHallSeat(str(i),str(j),SeatType.PREMIUM))

        for i in range(7):
            self.add_seat(CinemaHallSeat('AA',str(i),SeatType.SUITE))

    def add_seat(self, seat):
        self.__seats.append(seat)
```



Git Branch

- คลาส CinemaHallSeat

```
class CinemaHallSeat:  
    def __init__(self, seat_row, seat_col, seat_type):  
        self.__seat_row = seat_row  
        self.__seat_col = seat_col  
        self.__seat_type = seat_type
```

- เพิ่ม Hall ในโรงภาพยนตร์

```
cinema_central.add_halls(CinemaHall("Hall1", "nf First Class Cinema", 90))  
cinema_central.add_halls(CinemaHall("Hall2", "nf First Class Cinema", 90))  
cinema_central.add_halls(CinemaHall("Hall3", "MX4D", 90))  
cinema_central.add_halls(CinemaHall("Hall4", "ZIGMA CINESTADIUM", 90))  
cinema_central.add_halls(CinemaHall("Hall5", "Bed Cinema", 90))  
cinema_central.add_halls(CinemaHall("Hall6", "Mastercard Cinema", 90))  
cinema_central.add_halls(CinemaHall("Hall7", "Standard Cinema", 90))  
cinema_central.add_halls(CinemaHall("Hall8", "Standard Cinema", 90))  
cinema_central.add_halls(CinemaHall("Hall9", "Standard Cinema", 90))
```



Git Branch

- เมื่อถึงตรงนี้ เราก็มียโปรแกรมส่วนของโรงภาพยนตร์ และข้อมูลโรงภาพยนตร์แล้ว
- จากนั้นเราจะสร้างเส้นทางการพัฒนาใหม่นี้
 - ใน branch เดิม (master branch) จะพัฒนาในส่วนของคลาส Movie
 - ใน branch ที่แยกออกมาใหม่ จะพัฒนาในส่วนของคลาส User ต่างๆ
- คำสั่ง git ที่เกี่ยวกับ branch มีดังนี้
 - git branch แสดง branch ทั้งหมดใน repository
 - จากรูปจะเห็นได้ว่ามีเพียง branch เดียว โดย * จะเป็นตัวบอก default branch

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git branch
* master

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Git Branch

- ในการสร้าง branch ใหม่จะมีทางเลือกดังนี้
 - คำสั่ง `git branch <name>` จะสร้าง branch ใหม่
 - คำสั่ง `git checkout -b <name>` จะสร้าง branch ใหม่ และ เปลี่ยน default ไปยัง branch ที่สร้างขึ้นใหม่

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git checkout -b feature_user
Switched to a new branch 'feature_user'

khthana@PC-Terry MINGW64 ~/Desktop/my-project (feature_user)
$ git branch
* feature_user
  master

khthana@PC-Terry MINGW64 ~/Desktop/my-project (feature_user)
$ |
```



Git Branch

- คำสั่งอื่นๆ เกี่ยวกับ branch
 - `git checkout <name>` สลับไปยัง branch ที่เลือก
 - `git branch -d <name>` ลบ branch
 - `git branch -m <old> <new>` เปลี่ยนชื่อ branch
- เราจะลองสร้าง branch ชื่อ `feature_1` แล้วลบออก


```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git branch feature_1

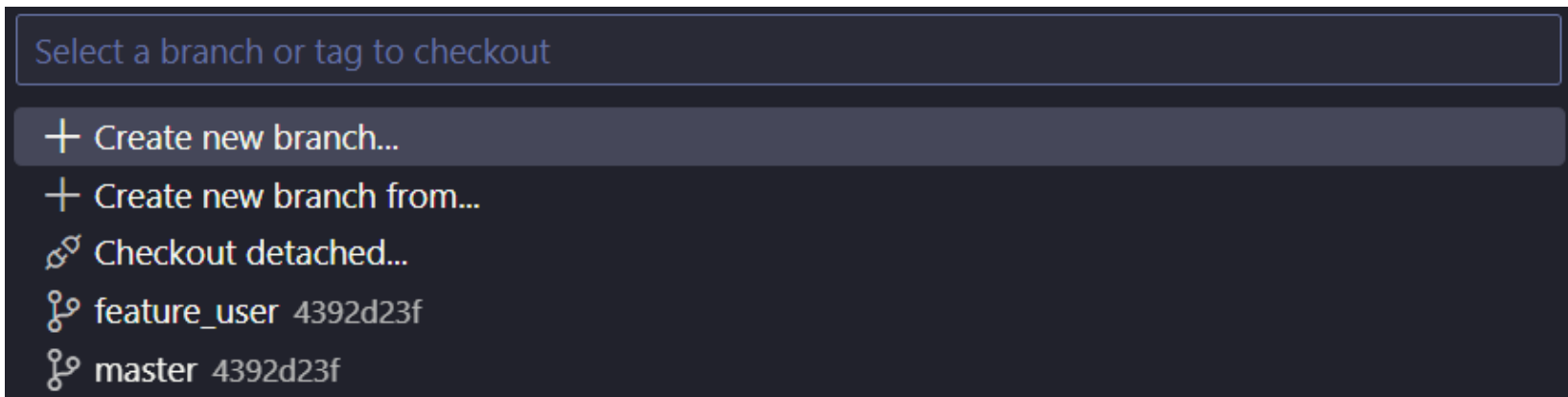
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git branch
  feature_1
  feature_user
* master

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git branch -d feature_1
Deleted branch feature_1 (was 4392d23).
```

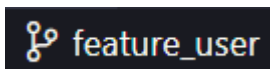


Git Branch

- สำหรับใน VS Code จะมีการทำงานเกี่ยวกับ branch ที่มุมซ้ายล่าง  master
- เมื่อคลิกที่ปุ่มดังกล่าว จะมีตัวเลือกแสดงใน Command Palette ของ VS Code










- ซึ่งสามารถจะเลือก branch ที่ต้องการ หรือ สร้าง branch ใหม่ก็ได้
- เมื่อเลือกทำงานที่ branch ใด ที่มุมล่างจะแสดงชื่อ branch ที่ทำงานอยู่





Git Branch

- ที่ branch เราจะสร้างไฟล์เกี่ยวกับ user เพิ่มเติม จากนั้นสร้าง user instance ที่ main.py
- จากนั้น commit เราจะมี 5 commit ดังรูป

BRANCH / TAG	GRAPH	COMMIT MESSAGE	AUTHOR	COMMIT DATE / TI...	SHA
✓  feature_user		<feat>: add class User and make user	You	2 minutes ago	9e186eb
 master		<feat>: add/test class CinemaHallSeat and data	You	2 hours ago	4392d23
		<feat>: add/test class CinemaHall and data	You	4 hours ago	c710d5f
		<feat> : add/test class CinemaSystem and Cine...	You	8 hours ago	2f04b1e
		<chore>: create .gitignore	You	yesterday	52c1525

- โดยเป็น commit ใน master branch จำนวน 4 commit และ feature_user อีก 1 branch



Git Branch

- เพิ่มคลาส Person และ Inherit เป็น Admin และสร้าง Instance

```
class Person:
    def __init__(self, name, email, phone):
        self.name = name
        self.email = email
        self.phone = phone
```

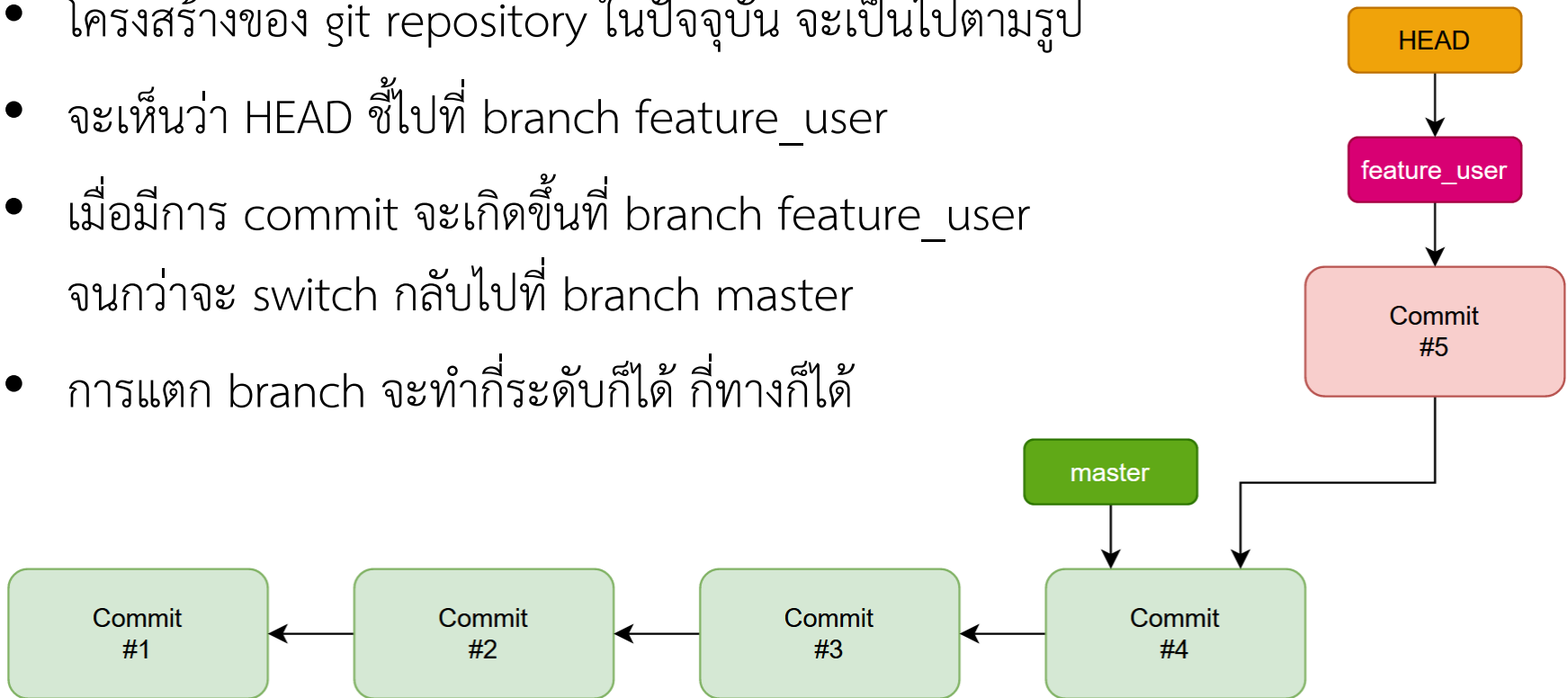
```
class Admin(Person):
    def __init__(self, name, email, phone):
        super().__init__(name, email, phone)
```

```
admin = Admin("Admin", "admin@abc.com", "081-234-5678")
print(admin)
```




Git Branch

- โครงสร้างของ git repository ในปัจจุบัน จะเป็นไปตามรูป
- จะเห็นว่า HEAD ชี้ไปที่ branch feature_user
- เมื่อมีการ commit จะเกิดขึ้นที่ branch feature_user จนกว่าจะ switch กลับไปที่ branch master
- การแตก branch จะทำที่ระดับก็ได้ กี่ทางก็ได้





Git Branch

- จากรูป จะเห็นว่าเมื่ออยู่ใน feature_user จะมี 4 ไฟล์ แต่เมื่ออยู่ใน master จะมี 3 ไฟล์

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (feature_user)
$ ls -l
total 14
drwxr-xr-x 1 khthana 197121  0 Feb 27 22:07 __pycache__/
-rw-r--r-- 1 khthana 197121 2020 Feb 27 19:50 cinema.py
-rw-r--r-- 1 khthana 197121 171 Feb 27 19:30 enum_class.py
-rw-r--r-- 1 khthana 197121 1550 Feb 27 22:10 main.py
-rw-r--r-- 1 khthana 197121  491 Feb 27 22:10 user.py

khthana@PC-Terry MINGW64 ~/Desktop/my-project (feature_user)
$ git checkout master
Switched to branch 'master'

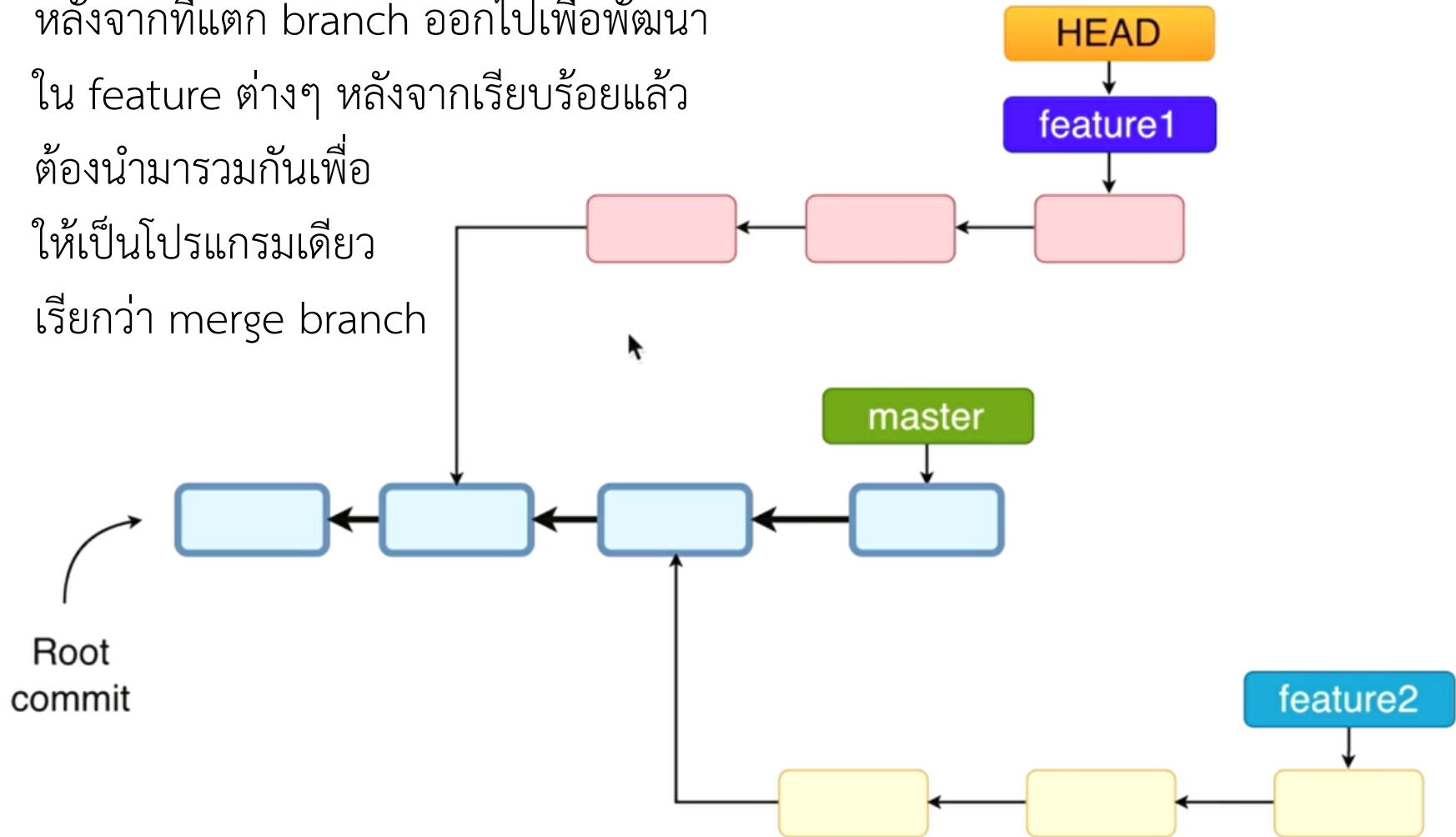
khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ ls -l
total 13
drwxr-xr-x 1 khthana 197121  0 Feb 27 22:07 __pycache__/
-rw-r--r-- 1 khthana 197121 2020 Feb 27 19:50 cinema.py
-rw-r--r-- 1 khthana 197121 171 Feb 27 19:30 enum_class.py
-rw-r--r-- 1 khthana 197121 1457 Feb 27 22:13 main.py

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Merge branch

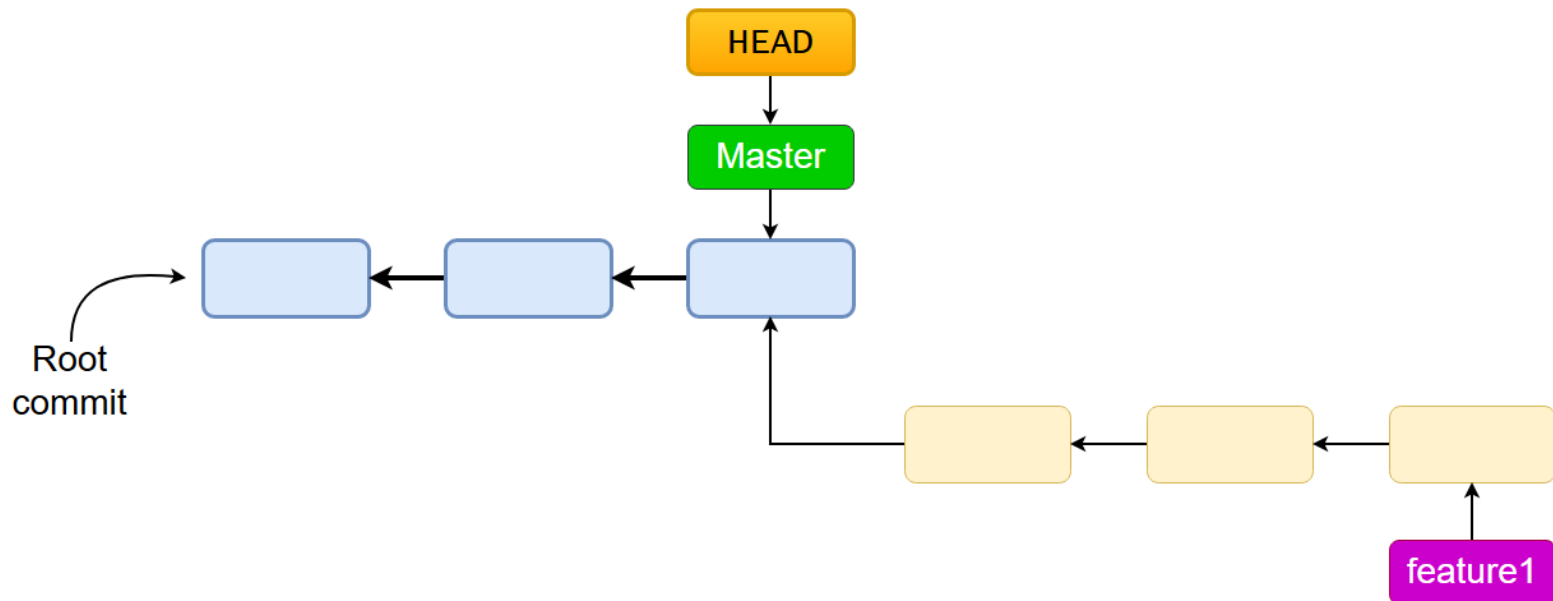
- หลังจากที่แตก branch ออกไปเพื่อพัฒนาใน feature ต่างๆ หลังจากเรียบร้อยแล้ว ต้องนำมารวมกันเพื่อให้เป็นโปรแกรมเดียว เรียกว่า merge branch





Merge branch

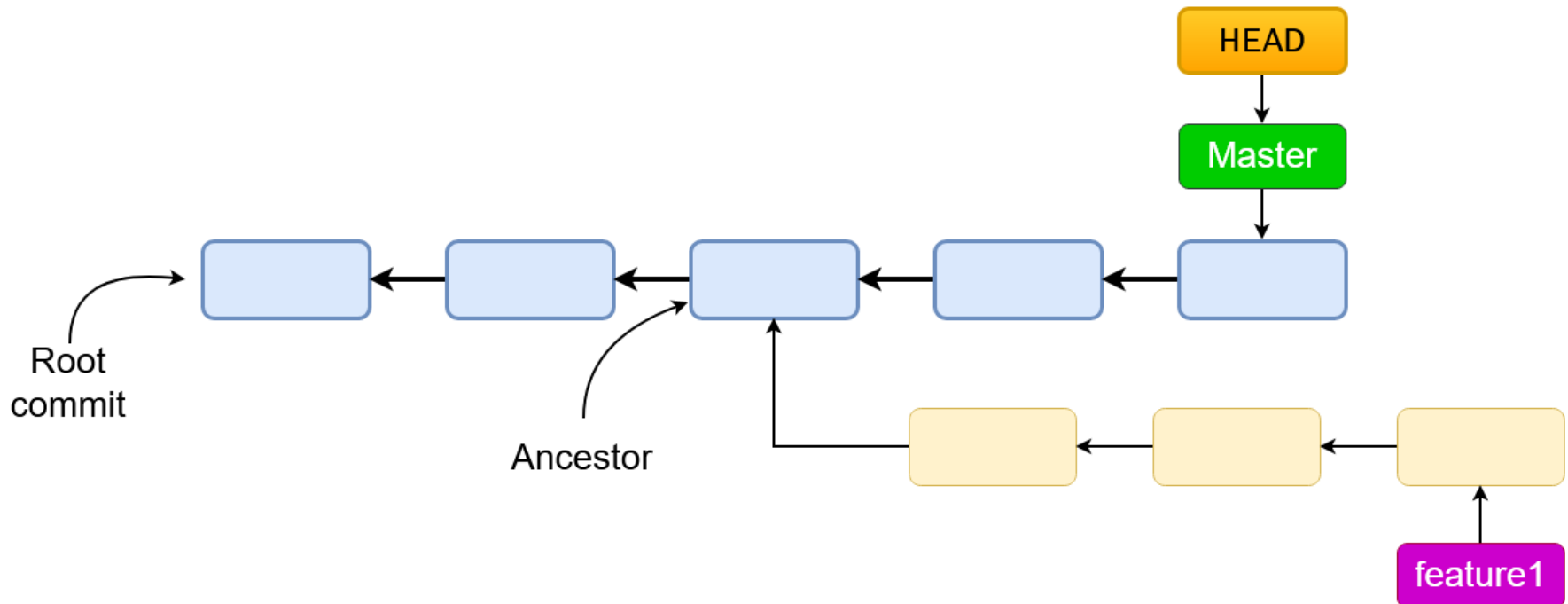
- การ merge มี 2 แบบ ตามสถานการณ์ที่เกิด
- แบบแรก เรียกว่า Fast forward merge คือ การ merge branch ใหม่ เข้าไปรวมกับ branch ก่อนหน้า โดยไม่มีการ commit เพิ่มเติมใน branch ที่มารับ
- วิธีการ คือ move ทั้ง master และ HEAD ไปที่ update สุดท้าย





Merge branch

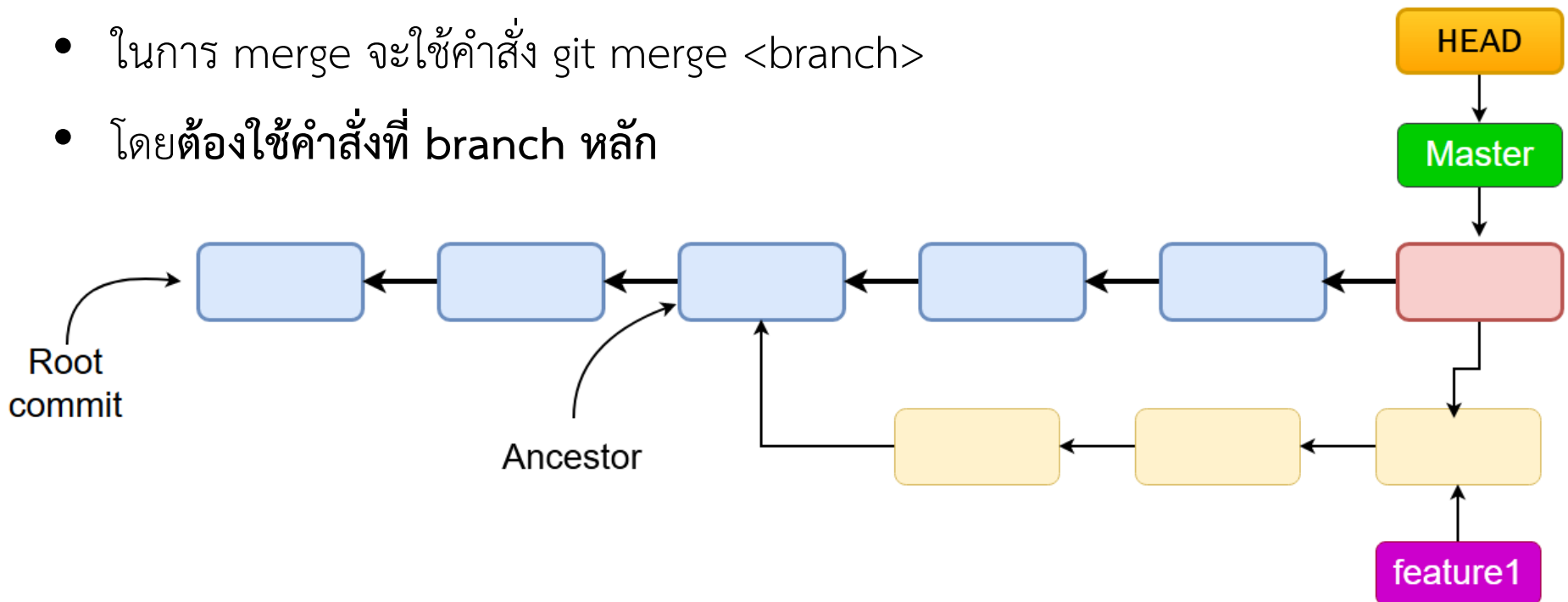
- แบบที่ 2 เรียกว่า 3 way merge เป็นกรณีที่ทั้ง 2 branch (หรือมากกว่า) มีการ commit ทำให้การ move pointer อย่างเดียว จะทำให้ข้อมูลบางส่วนสูญหาย





Merge branch

- ใน 3 way merge โปรแกรม git จะสร้าง commit ขึ้นมาใหม่ 1 commit (สีแดง) จากนั้นกำหนดให้ parent ของ commit นี้มาจากทั้ง 2 branch
- จะใช้วิธีการใด git จะดำเนินการให้เอง
- ในการ merge จะใช้คำสั่ง git merge <branch>
- โดยต้องใช้คำสั่งที่ branch หลัก





Merge branch

- กลับมาที่ระบบโรงภาพยนตร์ สมมติว่าเราได้พัฒนาใน branch feature_user จนทำงานได้ และ commit ไปแล้ว 1 commit หากต้องการ merge เราจะทำตามรูป

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (feature_user)
$ git branch
* feature_user
  master

khthana@PC-Terry MINGW64 ~/Desktop/my-project (feature_user)
$ git checkout master
Switched to branch 'master'

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ git merge feature_user
Updating 4392d23..9e186eb
Fast-forward
 main.py | 4 +++-
 user.py | 17 ++++++
 2 files changed, 20 insertions(+), 1 deletion(-)
 create mode 100644 user.py

khthana@PC-Terry MINGW64 ~/Desktop/my-project (master)
$ |
```



Merge branch

- ข้อมูลของ commit graph จะเปลี่ยนดังนี้

BRANCH / TAG	GRAPH	COMMIT MESSAGE	AUTHOR	COMMIT DATE / TI...	SHA
✓ feature_user		<feat>: add class User and make user	You	2 minutes ago	9e186eb
master		<feat>: add/test class CinemaHallSeat and data	You	2 hours ago	4392d23
		<feat>: add/test class CinemaHall and data	You	4 hours ago	c710d5f
		<feat> : add/test class CinemaSystem and Cine...	You	8 hours ago	2f04b1e
		<chore>: create .gitignore	You	yesterday	52c1525

BRANCH / TAG	GRAPH	COMMIT MESSAGE	AUTHOR	COMMIT DATE / TI...	SHA
✓ master +1		<feat>: add class User and make user	You	18 hours ago	9e186eb
		<feat>: add/test class CinemaHallSeat and data	You	21 hours ago	4392d23
		<feat>: add/test class CinemaHall and data	You	22 hours ago	c710d5f
		<feat> : add/test class CinemaSystem and Cine...	You	yesterday	2f04b1e
		<chore>: create .gitignore	You	2 days ago	52c1525



GitHub

- Git เป็น version control ที่เหมาะสำหรับใช้งานใน local computer หรือในระบบที่มีการ share folder แต่ในการพัฒนา software ที่แยกกัน อาจไม่สะดวก
- GitHub เป็น repository hosting service คือ ทำหน้าที่เก็บ repository เพื่อ share ระหว่างผู้พัฒนา จึงเป็นบริการที่ช่วยให้การใช้ git มีความสมบูรณ์มากขึ้น



Git

vs

GitHub



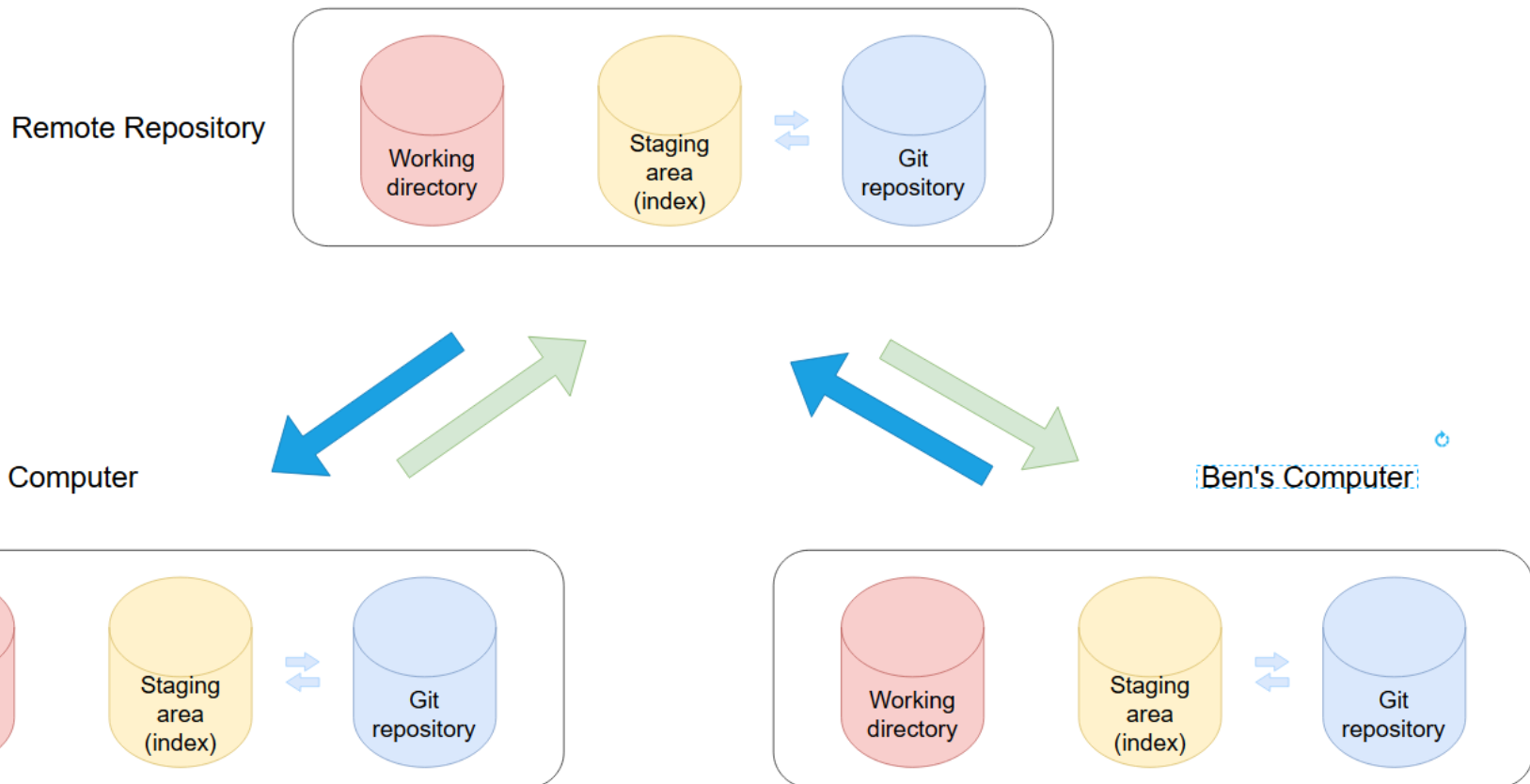
Distributed version-control
system

Repository hosting
service

GitHub




- GitHub เป็นที่เก็บ repository ที่แชร์ระหว่างคอมพิวเตอร์ต่างๆ หากแต่ละเครื่องมีการ update กับ GitHub ก็จะทำให้ทุกเครื่องที่ทำงานเดียวกัน มีข้อมูลเดียวกัน



GitHub




- ให้เข้าไปที่ GitHub และสร้าง GitHub Account (หากยังไม่มี)
- วิธีที่ง่าย คือ สร้าง Repository บน GitHub จากนั้นค่อย Clone มาที่แต่ละเครื่อง
- ให้กดเครื่องหมาย  ที่มุมบนขวา และ เลือก New Repository จากนั้นป้อนข้อมูล และกด **Create repository**

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner *

 thana-ho ▾

Repository name *

/ my-project 


Great repository names are short and memorable. Need inspiration? How about **fuzzy-system**?

Description (optional)

Project in object oriented programming subject

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.



GitHub

- ให้คัดลอก link ในกรอบเอาไว้ จากนั้นไปที่ desktop หรือที่ที่ต้องการ จากนั้นเรียก command line
- แล้วใช้คำสั่ง `git clone <link>`


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

 Set up in Desktop or ☐ HTTPS ☐ SSH `https://github.com/thana-ho/my-project.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 "# my-project" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/thana-ho/my-project.git
git push -u origin main
```





GitHub

- Git จะคัดลอก repository มาที่ desktop

```
khthana@PC-Terry MINGW64 ~/Desktop
$ git clone https://github.com/thana-ho/my-project.git
Cloning into 'my-project'...
warning: You appear to have cloned an empty repository.

khthana@PC-Terry MINGW64 ~/Desktop
$ cd my-project/

khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ ls -al
total 68
drwxr-xr-x 1 khthana 197121 0 Feb 28 17:28 ./
drwxr-xr-x 1 khthana 197121 0 Feb 28 17:28 ../
drwxr-xr-x 1 khthana 197121 0 Feb 28 17:28 .git/

khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ |
```



GitHub

- เข้าไปที่ GitHub คลิกไปที่ GitHub profile icon ที่มุมขวาบน
- คลิก Settings เลื่อนมาที่ด้านล่างและคลิกที่ Developer Settings ที่ด้านซ้าย
- เลือก Personal access tokens -> Tokens (classic)
- คลิก Generate new token เลือกจำนวนวัน

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

What's this token for?

Expiration *

90 days ⇅

The token will expire on Mon, May 29 2023

GitHub



- ให้เลือก Scope
- เลือก
 - repo
 - admin:repo_hook
 - delete_repo

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> admin:org	Full control of orgs and teams
<input type="checkbox"/> write:org	Read and write org and team membership
<input type="checkbox"/> read:org	Read org and team membership
<input type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys
<input type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input type="checkbox"/> write:repo_hook	Write repository hooks
<input type="checkbox"/> read:repo_hook	Read repository hooks
<input type="checkbox"/> admin:org_hook	Full control of organization hooks
<input type="checkbox"/> gist	Create gists
<input type="checkbox"/> notifications	Access notifications
<input type="checkbox"/> user	Update all user data
<input type="checkbox"/> user:email	Access user email addresses (read-only)
<input type="checkbox"/> user:follow	Follow and unfollow users
<input type="checkbox"/> delete_repo	Delete repositories



GitHub

- ให้ทดลองสร้าง .gitignore ใน my-project จากนั้น commit
- ให้ใช้คำสั่ง git remote set-url origin
https://<githubtoken>@github.com/<username>/<repositoryname>.git
- จากนั้นใช้คำสั่ง git push -u origin main เพื่อ push จากนั้นตรวจสอบที่ GitHub

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git remote set-url origin https://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@github.com/thana-ho/my-project.git

khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 1.89 KiB | 969.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/thana-ho/my-project.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
```


GitHub



- กลับไปที่หน้า GitHub Repository จะพบว่ามี การ push ไฟล์ .gitignore ขึ้นไปแล้ว
- Branch จะเป็น main มี commit message และ hash ตรงกับใน local computer

The screenshot shows the GitHub interface for a repository. At the top, the 'main' branch is selected, with '1 branch' and '0 tags' indicated. Below this, a commit by user 'khthana' is shown with the message '<chore> :add .gitignore'. The commit hash 'bfed133' and the time '12 hours ago' are also visible. A file named '.gitignore' is listed with the same commit message and time. At the bottom, there is a light blue banner encouraging the user to 'Add a README'.

main 1 branch 0 tags

Go to file Add file <> Code

khthana <chore> :add .gitignore bfed133 12 hours ago 1 commit

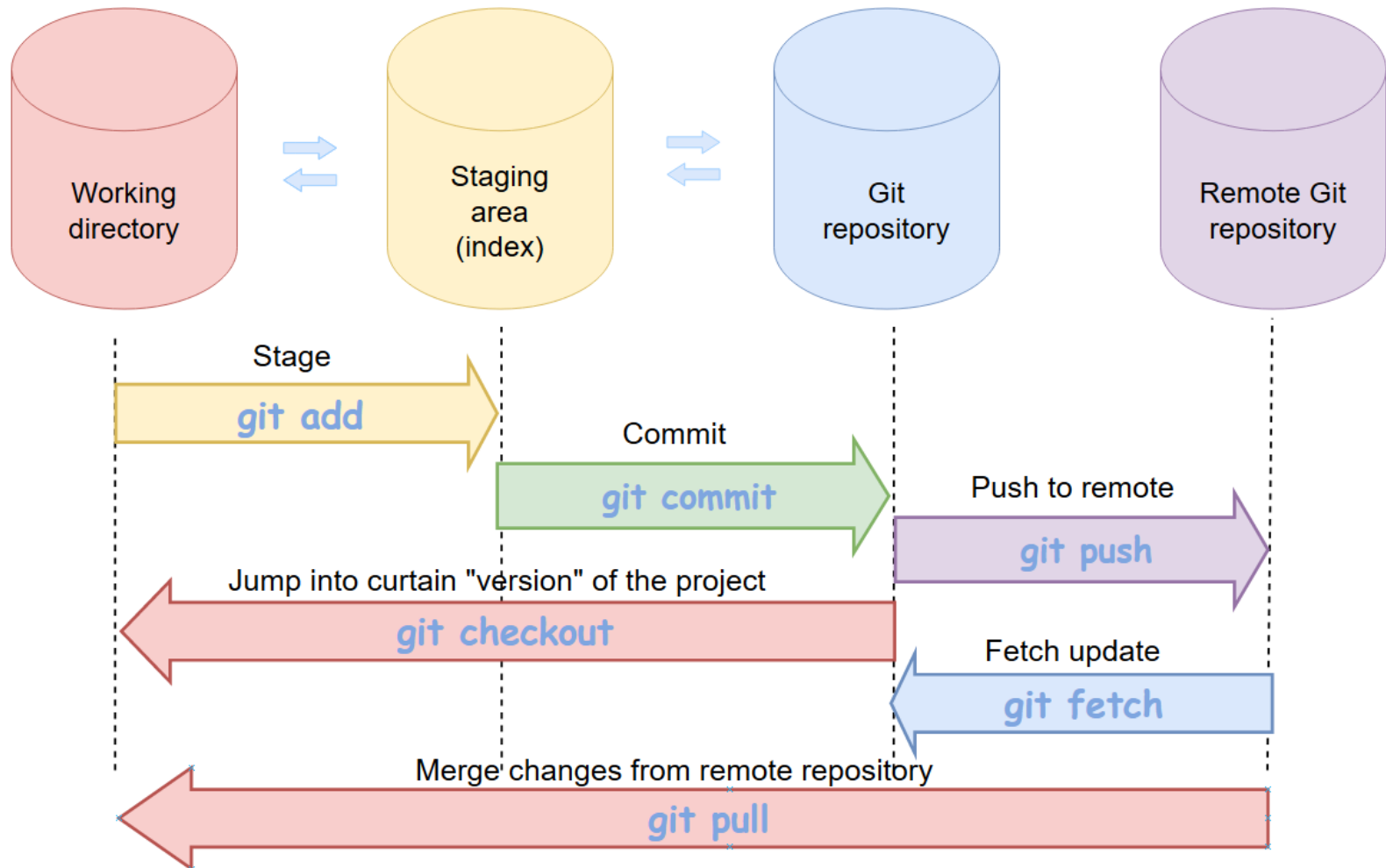
.gitignore <chore> :add .gitignore 12 hours ago

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



GitHub

- คำสั่งที่ใช้ใน git และ GitHub





GitHub

- คำสั่งที่ใช้
 - git push <remote> <branch> เป็นการส่งไฟล์ที่ commit แล้วเข้าสู่ remote repository
 - option -u เป็นการบอกว่าจะใช้ upstream ไດ
 - git branch -a บอกว่ามี branch ไต่บ้าง รวมถึง remote ด้วย

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git branch -a
* main
remotes/origin/main
```

- git branch -vv แสดง upstream branch และ commit ล่าสุด

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git branch -vv
* main bfed133 [origin/main] <chore> :add .gitignore
```

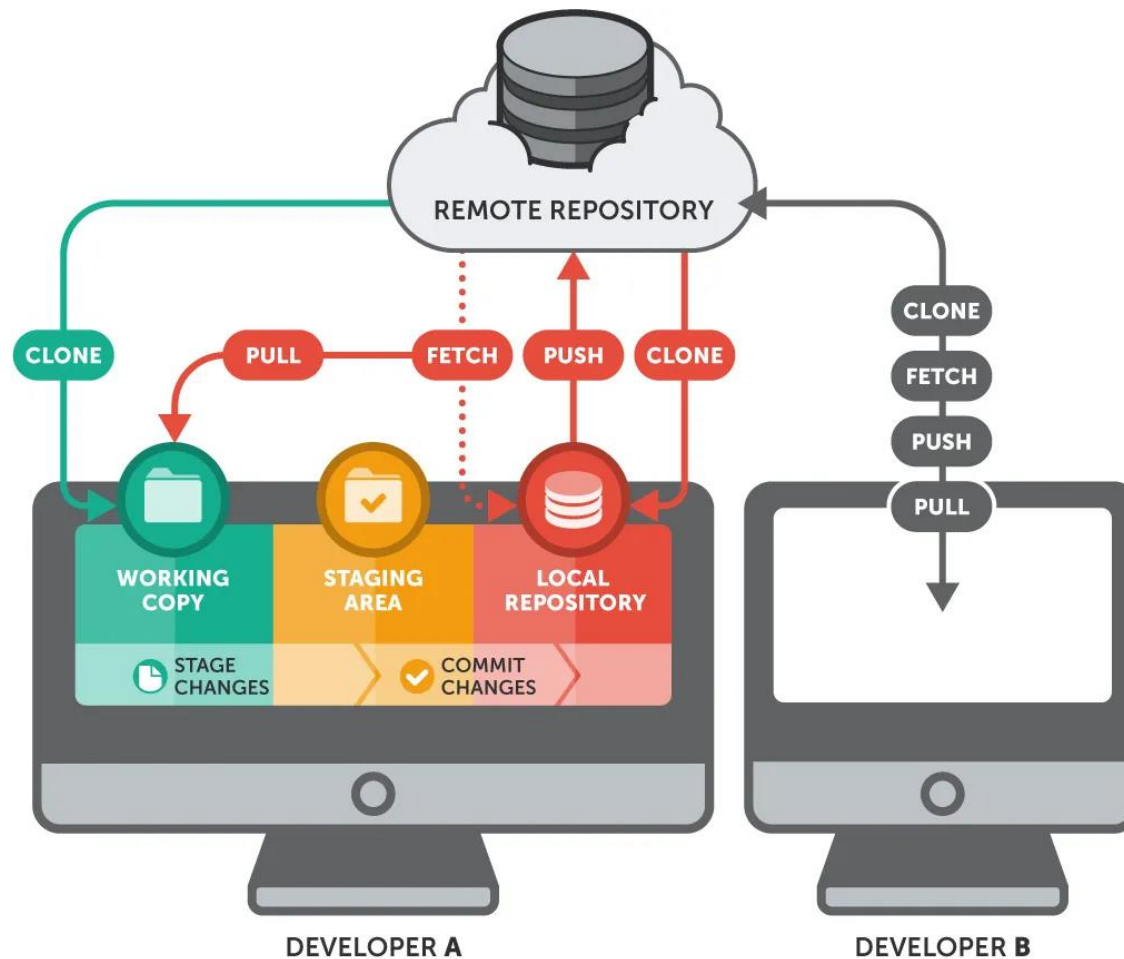


- คำสั่งที่ใช้
 - git fetch
ใช้ตรวจสอบว่าไฟล์ใน local repository กับremote repository มีความแตกต่างกันหรือไม่ ไฟล์ไหนใหม่กว่า หรือล้าสมัยอยู่ที่ commit
 - git pull
เป็นการดึงไฟล์ที่มีการเปลี่ยนแปลงใน remote repository มาเปลี่ยนแปลงใน local work directory ซึ่งเทียบได้กับการใช้คำสั่ง git fetch ซึ่งเป็นการ update ในส่วน repository ของ git จากนั้นจึงตามด้วยคำสั่ง git merge ซึ่งเป็นนำข้อมูลใน repository มา merge ลงใน local work directory ก็จะเห็นไฟล์ตรงกับใน remote repository



GitHub


- แสดงการทำงานของ git push, fetch, pull






GitHub

- ต่อไปจะทดลองเพิ่มไฟล์ใน github จำนวน 3 ไฟล์ โดยสร้างเป็น 3 commit
- คลิกที่ Add file -> Upload files
- จะเปิดหน้าจอใหม่ สมมติว่านำไฟล์ enum_class.py เข้ามา
- จากนั้นใส่ commit message และ กดที่ commit

 enum_class.py ✕



Commit changes

<feat>: enum class

Add an optional extended description...

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

Commit changes

Cancel

- ทำแบบนี้อีก 2 ครั้งกับไฟล์ cinema.py และ main.py

GitHub



- จะมี 4 ไฟล์ ดังรูป

👤 main ▾ 1 branch 0 tags Go to file Add file ▾ <> Code ▾

+ thana-ho <feat>: add main.py		2fc8989 now 4 commits
📄 .gitignore	<chore> :add .gitignore	13 hours ago
📄 cinema.py	<feat>: add cinema.py	1 minute ago
📄 enum_class.py	<feat>: enum class	3 minutes ago
📄 main.py	<feat>: add main.py	now

- คราวนี้จะสร้าง branch บน github
- คลิกที่ 👤 main ▾ จะปรากฏ
- ให้ป้อนชื่อ user_feature
- แล้วกด Create branch

Switch branches/tags ×

Branches Tags

✓ main default



- จะพบว่าเกิด branch ใหม่

🔗 user_feature ▾
🔗 2 branches
🏷️ 0 tags
Go to file
Add file ▾
<> Code ▾

This branch is up to date with main.
🔗 Contribute ▾

+	thana-ho <feat>: add main.py	2fc8989 14 minutes ago	🕒 4 commits
📄	.gitignore	<chore> :add .gitignore	13 hours ago
📄	cinema.py	<feat>: add cinema.py	14 minutes ago
📄	enum_class.py	<feat>: enum class	16 minutes ago
📄	main.py	<feat>: add main.py	14 minutes ago

- สมมติว่า upload ไฟล์ user.py ลงใน branch นี้ แล้ว commit



- หน้าจอจะแสดงดังนี้

user_feature had recent pushes less than a minute ago

[Compare & pull request](#)

user_feature ▾

2 branches 0 tags

[Go to file](#)

[Add file ▾](#)

[Code ▾](#)

This branch is [1 commit ahead](#) of main.



[Contribute ▾](#)


	thana-ho <feat>: add user.py	2177a74 now 5 commits
	.gitignore	<chore> :add .gitignore 13 hours ago
	cinema.py	<feat>: add cinema.py 18 minutes ago
	enum_class.py	<feat>: enum class 20 minutes ago
	main.py	<feat>: add main.py 17 minutes ago
	user.py	<feat>: add user.py now






- หากคลิกที่ View all branches (อยู่ในเมนูที่แสดง branch) จะเห็น branch ทั้งหมด

Default branch




main  Updated 21 minutes ago by thana-ho Default 

 **Your main branch isn't protected**
Protect this branch from force pushing or deletion, or require status checks before merging. [Learn more](#) Dismiss Protect this branch

Your branches

user_feature  Updated 4 minutes ago by thana-ho  

Active branches

user_feature  Updated 4 minutes ago by thana-ho  



GitHub

- จะกลับไป local computer และ เรียก git pull เพื่อ pull ไฟล์จาก GitHub

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git pull
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 12 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (12/12), 3.70 KiB | 64.00 KiB/s, done.
From https://github.com/thana-ho/my-project
   bfed133..2fc8989  main          -> origin/main
   * [new branch]    user_feature -> origin/user_feature
Updating bfed133..2fc8989
Fast-forward
 cinema.py          | 63 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 enum_class.py      | 10 ++++++++
 main.py            | 33 +++++++++++++++++++++++++++++++++++++
 3 files changed, 106 insertions(+)
 create mode 100644 cinema.py
 create mode 100644 enum_class.py
 create mode 100644 main.py
```

- จะเห็นว่า pull ไฟล์ที่อยู่ใน branch main มา แต่ไฟล์ที่อยู่ใน branch user_feature ไม่ได้ pull มาด้วย



GitHub

- เมื่อใช้คำสั่ง `git branch -vv` จะเห็นว่ามีเพียง branch main

```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git branch -a
* main
  remotes/origin/main
  remotes/origin/user_feature

khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git branch -vv
* main 2fc8989 [origin/main] <feat>: add main.py
```

- ต้องใช้คำสั่ง `git checkout user_feature` จึงจะแสดงไฟล์ในอีก branch


```
khthana@PC-Terry MINGW64 ~/Desktop/my-project (main)
$ git checkout user_feature
Switched to a new branch 'user_feature'
Branch 'user_feature' set up to track remote branch 'user_feature' from 'origin'.

khthana@PC-Terry MINGW64 ~/Desktop/my-project (user_feature)
$ git branch -vv
  main          2fc8989 [origin/main] <feat>: add main.py
* user_feature  2177a74 [origin/user_feature] <feat>: add user.py
```





Pull requests

- ในจอภาพหลังจากที่เรา upload ไฟล์ user.py เข้าไป จะพบว่ามีข้อความในกรอบ

 **user_feature** had recent pushes less than a minute ago

[Compare & pull request](#)

 user_feature ▾


 2 branches  0 tags








[Go to file](#)

[Add file ▾](#)

[Code ▾](#)

This branch is [1 commit ahead](#) of main.

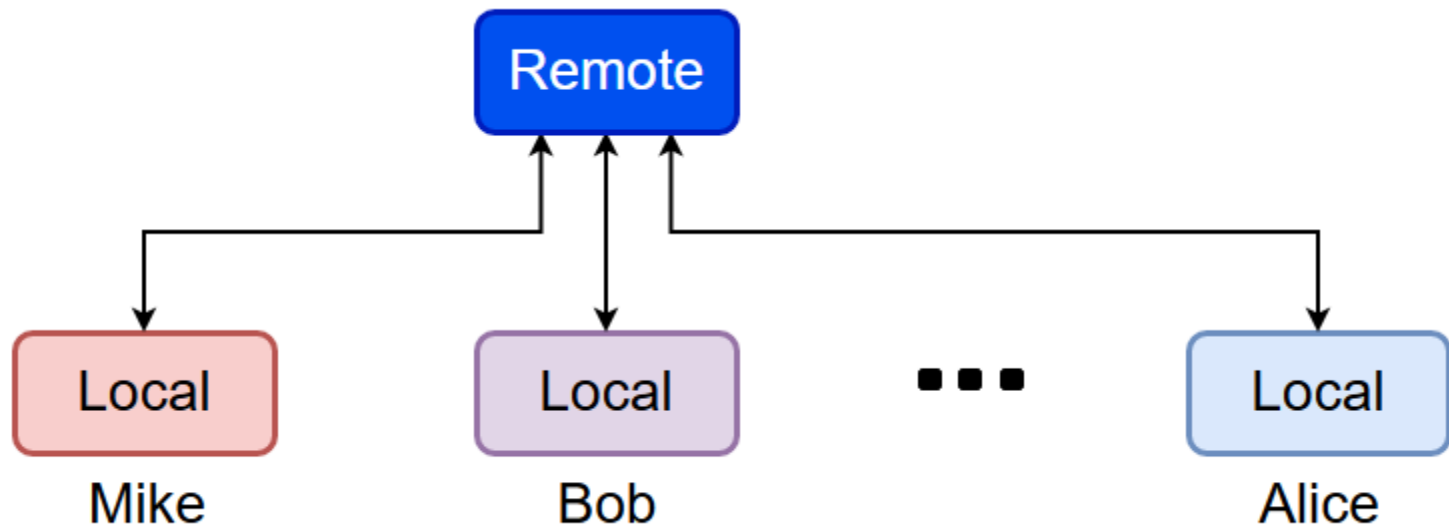
 [Contribute ▾](#)

 thana-ho <feat>: add user.py	2177a74 now  5 commits
 .gitignore	<chore> :add .gitignore 13 hours ago
 cinema.py	<feat>: add cinema.py 18 minutes ago
 enum_class.py	<feat>: enum class 20 minutes ago
 main.py	<feat>: add main.py 17 minutes ago
 user.py	<feat>: add user.py now



Pull requests

- Pull requests มีความหมายว่า GitHub มีการ update ใน branch อื่น ในกรณีที่ผู้พัฒนาหลายคนทำงานร่วมกันผ่าน GitHub และ ต้องการรวม feature ที่พัฒนา เข้ากับ main branch หรือ branch หลัก และ ต้องการ merge branch กลับมาที่ repository หลัก จึงขอความเห็นจากผู้พัฒนาอื่น ผ่าน Pull request และหากเห็นชอบจะได้ merge ต่อไป





Pull requests

- ในการเพิ่มผู้พัฒนาเข้าไปใน GitHub repository ให้เข้าไปที่ repository จากนั้นเลือก Setting จะเปิดหน้าต่างใหม่ ให้เลือก Collaborators
- ให้เพิ่มผู้ใช้ในทีมเข้าไป จากนั้น GitHub จะส่ง invite mail ไปให้ตอบรับ

Manage access



You haven't invited any collaborators yet

Add people



Pull requests

- หลังจากที่เราเพิ่ม Collaborator แล้วผู้ใช้อีกคนสามารถจะ pull repository และทำงานร่วมกันได้
- จาก user_feature branch ที่สร้างไว้ก่อนหน้านี้ หากเปิดหน้าจอ View all branch จะแสดงดังนี้ จะเห็นว่ามีปุ่ม New pull request ให้กดที่ปุ่มนี้

Your branches

user_feature	Updated 3 hours ago by thana-ho	0 1	New pull request		
--------------	---------------------------------	-------	------------------	--	--

Active branches

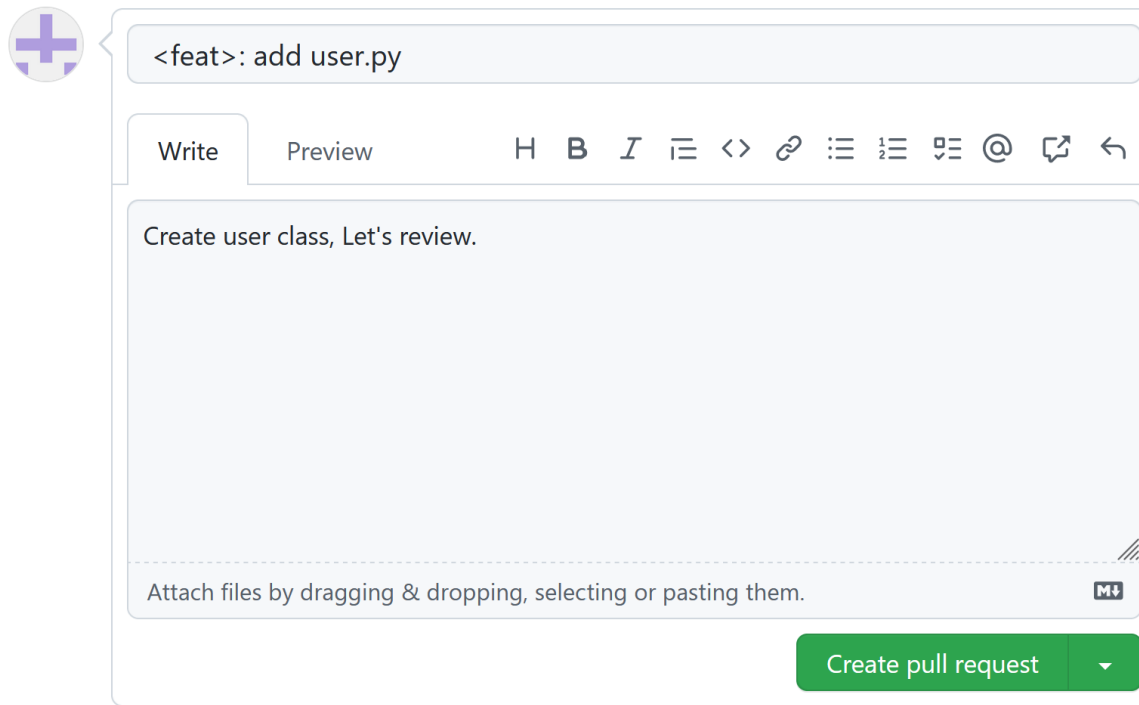
user_feature	Updated 3 hours ago by thana-ho	0 1	New pull request		
--------------	---------------------------------	-------	------------------	--	--



Pull requests

- หลังจากนั้นจะปรากฏหน้าจอ ตามรูป ให้ป้อน comment และกด เพื่อสร้าง pull requests

Create pull request



The screenshot shows the GitHub pull request creation interface. At the top, there is a purple plus icon in a circle. Below it, a text input field contains the text "<feat>: add user.py". Underneath the input field, there are two tabs: "Write" and "Preview". To the right of the tabs is a rich text editor toolbar with icons for heading (H), bold (B), italic (I), list (≡), code (<>), link (🔗), unlink (🔗), quote (≡), @ mention, share (🔗), and undo (↶). Below the toolbar is a large text area containing the text "Create user class, Let's review." At the bottom of the text area, there is a dashed line and the text "Attach files by dragging & dropping, selecting or pasting them." To the right of this text is a small icon of a document with a plus sign. At the bottom right of the form, there is a green button labeled "Create pull request" with a dropdown arrow.








Pull requests

- คราวนี้จะสมมติว่า User 2 ได้ login และเข้าสู่ repository จะพบว่า มี Pull requests อยู่ 1 request ที่เปิดค้างอยู่

[Code](#) [Issues](#) [Pull requests 1](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

[main](#) [2 branches](#) [0 tags](#) [Go to file](#) [Add file](#) [Code](#)

 **thana-ho** <feat>: add main.py 2fc8989 4 hours ago 🕒 4 commits

 .gitignore	<chore> :add .gitignore	17 hours ago
 cinema.py	<feat>: add cinema.py	4 hours ago
 enum_class.py	<feat>: enum class	4 hours ago
 main.py	<feat>: add main.py	4 hours ago

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

[Add a README](#)



Pull requests

- เมื่อคลิกเข้าไปดู จะพบหน้าจอดังนี้
- จะเห็นว่า GitHub แจ้งว่ามี Open อยู่ 1 Pull request

Filters ▾

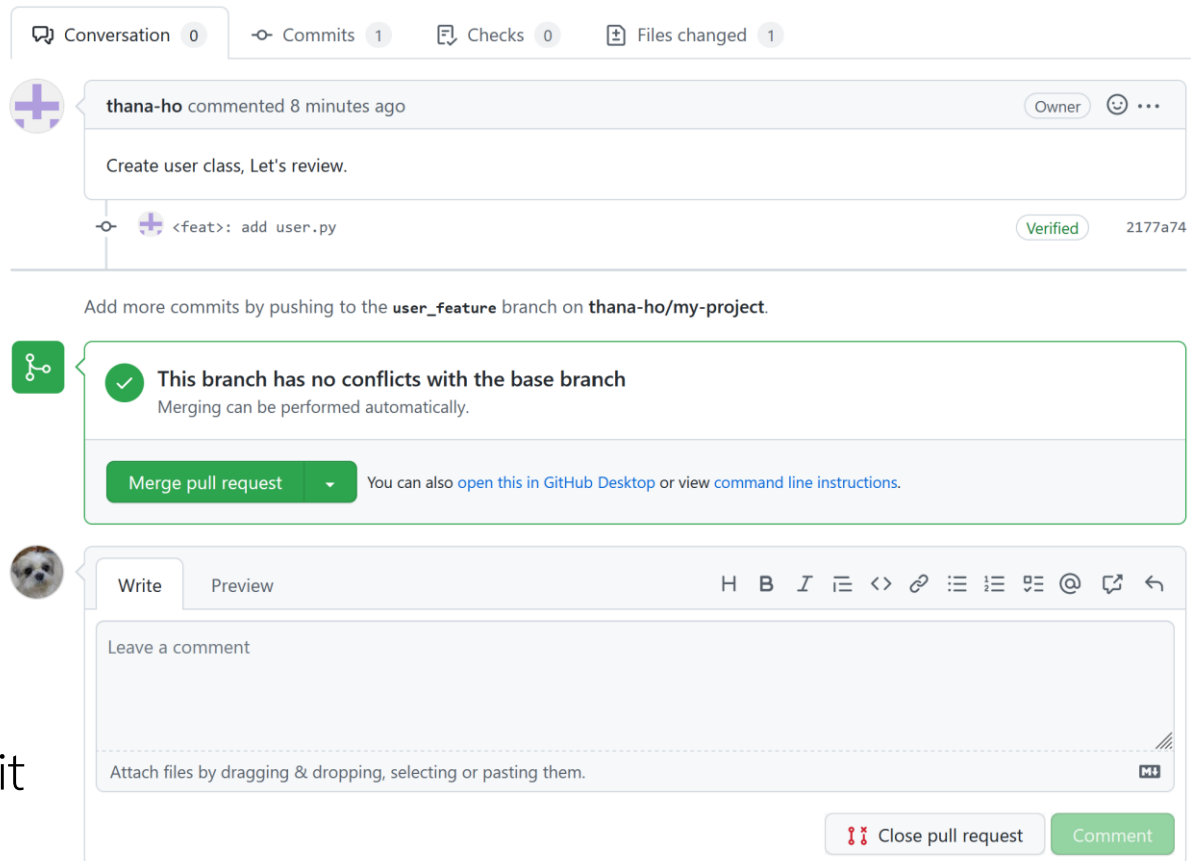
🔗 1 Open ✓ 0 Closed

Author ▾	Label ▾	Assignee ▾	Sort ▾
<div>🔗 <feat>: add user.py</div> <div>#1 opened 4 minutes ago by thana-ho</div>			



Pull requests

- ณ จุดนี้จะมี 2 ทางเลือก คือ user 2 จะกดที่ Merge pull requests เพื่อ merge branch ก็ได้
- หรือจะเขียน comment เพื่อให้ user 1 แก้ไข ก่อนแล้วค่อย commit เข้ามาใหม่ก็ได้
- ผู้ใช้ทุกคนใน repository สามารถ merge branch ได้
- สามารถดูข้อมูลใน commit ได้





Pull requests

- ในการ open pull request ควรกำหนดรูปแบบให้ตรงกันทั้งทีม
- Title อาจใส่ [xxx] เพื่อเน้น หรือใส่หมายเลข #nn
- อาจใส่รูปเพื่อบอกว่าที่ขอ merge เป็นการทำงานหน้าไหน
- สามารถใส่ข้อมูลต่อไปนี้ได้
 - Link หรือ แนบไฟล์ได้
 - Header ตัวหนา ตัวเอียง
 - Bullet หรือหมายเลข
 - Mention user อื่น



For your attention