



GIT

Version Control





Version control เป็นระบบที่ใช้สำหรับการจัดเก็บและบริหารจัดการรหัส (code) หรือไฟล์ใดๆ โดยที่เราสามารถติดตามการเปลี่ยนแปลงที่เกิดขึ้นในรหัสหรือไฟล์เหล่านั้นได้ตลอดเวลา โดยมีประโยชน์หลักๆ

- การทำงานร่วมกันในทีม
- การย้อนกลับไปที่เวอร์ชันที่มั่นยำ
- การแยกแยะการพัฒนาอย่างชัดเจน

GIT Install

ติดตั้ง Git ลงเครื่อง

1

เลือก
ระบบปฏิบัติการ



<https://git-scm.com/downloads>

ติดตั้งตามขั้นตอน

2

3

ตรวจสอบเวอร์ชัน

คำสั่ง : git --version

ผลลัพธ์ :

```
● ● ●   sukapat — -zsh — 55x5
[sukapat@Macintosh ~ % git --version
git version 2.33.0
sukapat@Macintosh ~ % ]
```

GIT Tutorial Initial

เริ่มต้นการใช้งาน

1

สร้าง Folder

สร้างโฟลเดอร์เปล่า สำหรับ
การทดสอบชื่อ git-test

2

เริ่มต้น

คำสั่ง : git init

ผลลัพธ์ :

```
[sukapat@Macintosh git-test % git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /Users/sukapat/Desktop/git-test/.git/
sukapat@Macintosh git-test % ]
```

3

ตรวจสอบ

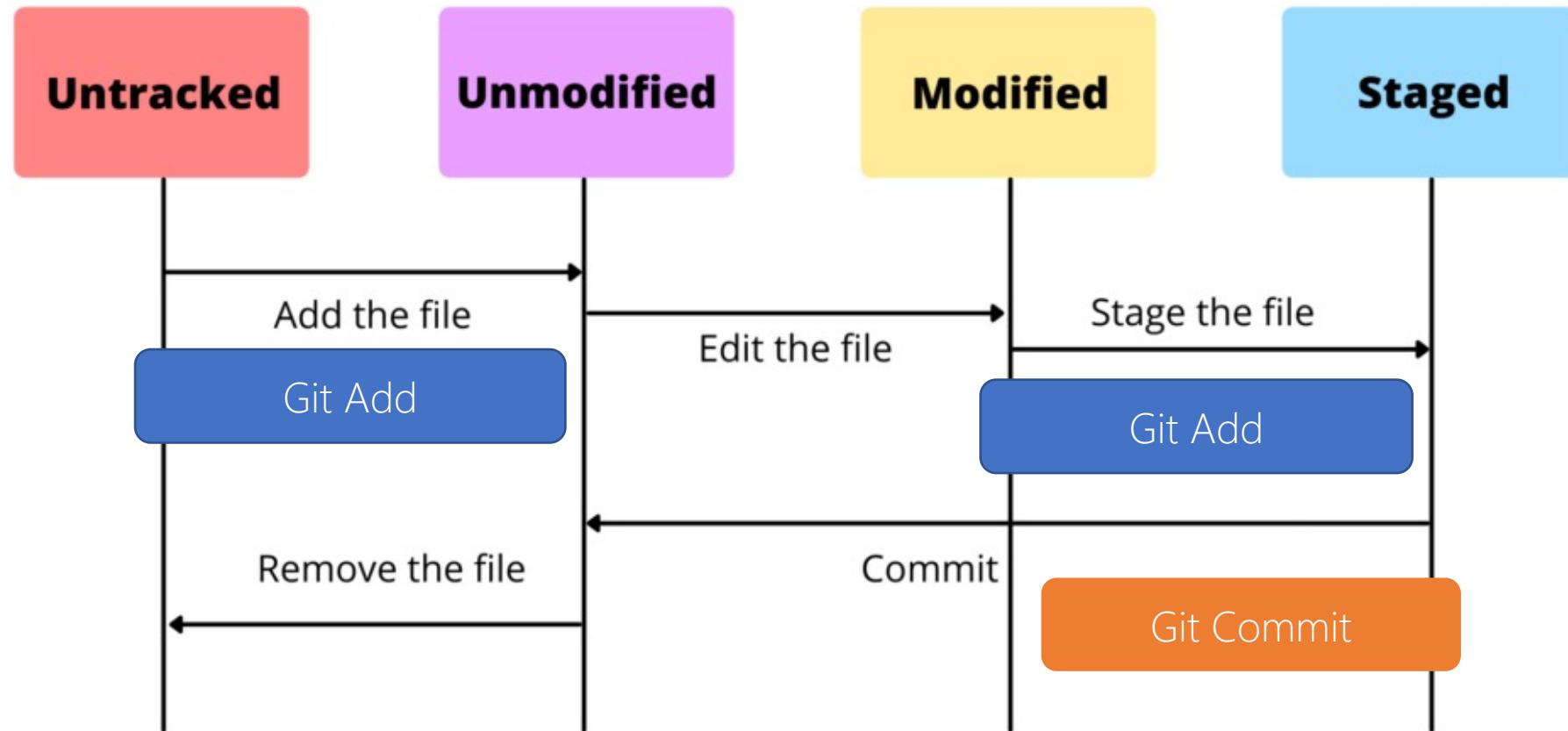
คำสั่ง : ls -la

ผลลัพธ์ : จะมีการสร้างโฟลเดอร์ .git ขึ้นมา

```
[sukapat@Macintosh git-test % ls -la
total 0
drwxr-xr-x  3 sukapat  staff   96 Mar 26 16:35 .
drwx-----@ 34 sukapat  staff  1088 Mar 26 16:36 ..
drwxr-xr-x 10 sukapat  staff  320 Mar 26 16:35 .git
sukapat@Macintosh git-test % ]
```

GIT File Lifecycle

เริ่มต้นการใช้งาน



GIT Commit Pattern

รูปแบบการ Commit งาน

รูปแบบ

[Action]: Some Message

ตัวอย่าง

[Add]: Function authentication with google

GIT Commit Pattern

รูปแบบการ Commit งาน

Add : "เพิ่มไฟล์ หรือ เพิ่มฟังก์ชันการทำงาน สิ่งใหม่ที่ไม่มีเมื่อนั้น"

Edit : "แก้ไข หรือ เพิ่ม สิ่งที่มีอยู่"

Delete : "ลบสิ่งนั้นออกไป"

Fix : "แก้ไขข้อผิดพลาด"

Release : "ปล่อยการใช้งานของโปรแกรม และ Tag เวอร์ชัน"

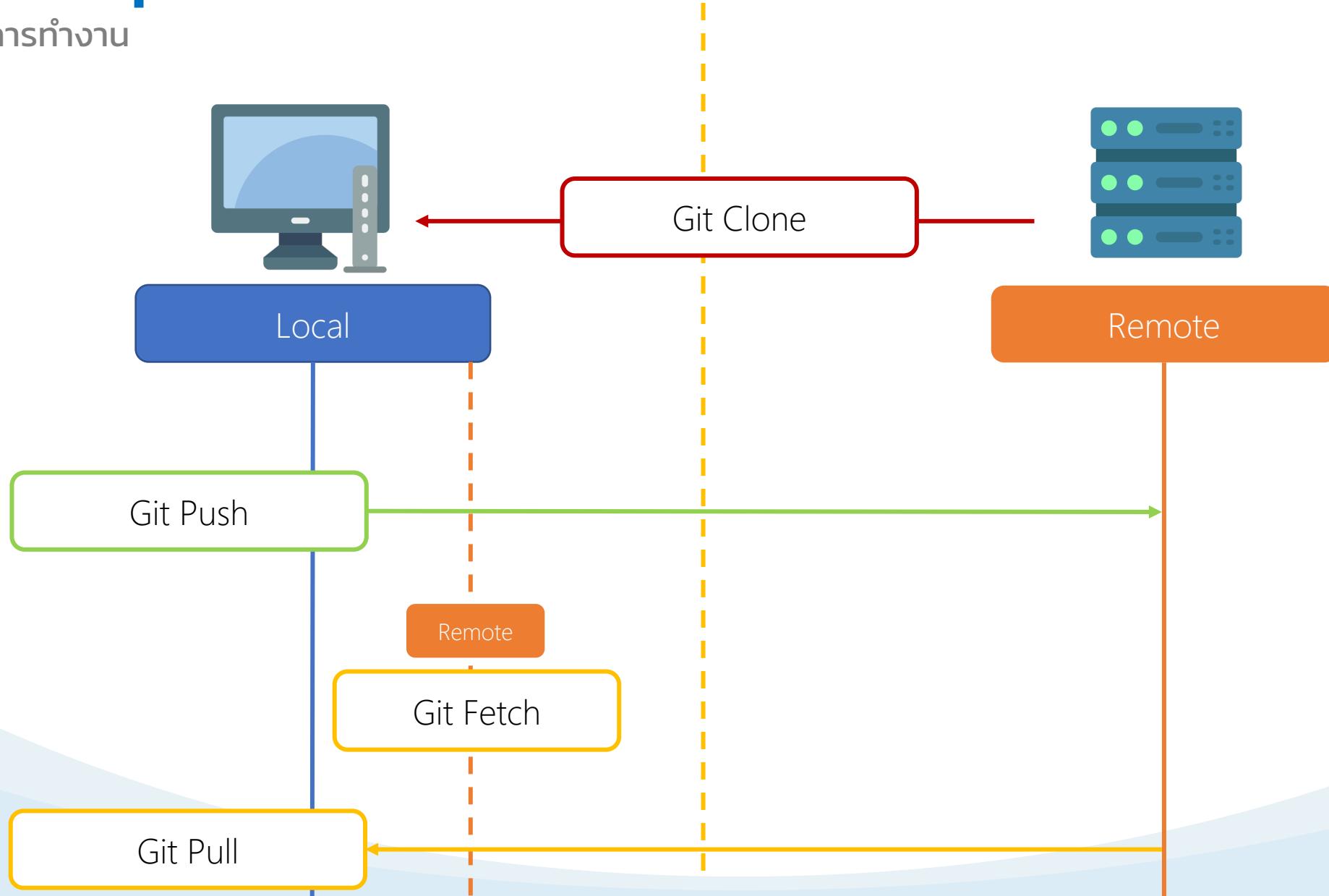
Refactor : "ปรับปรุงคุณภาพโค้ด"

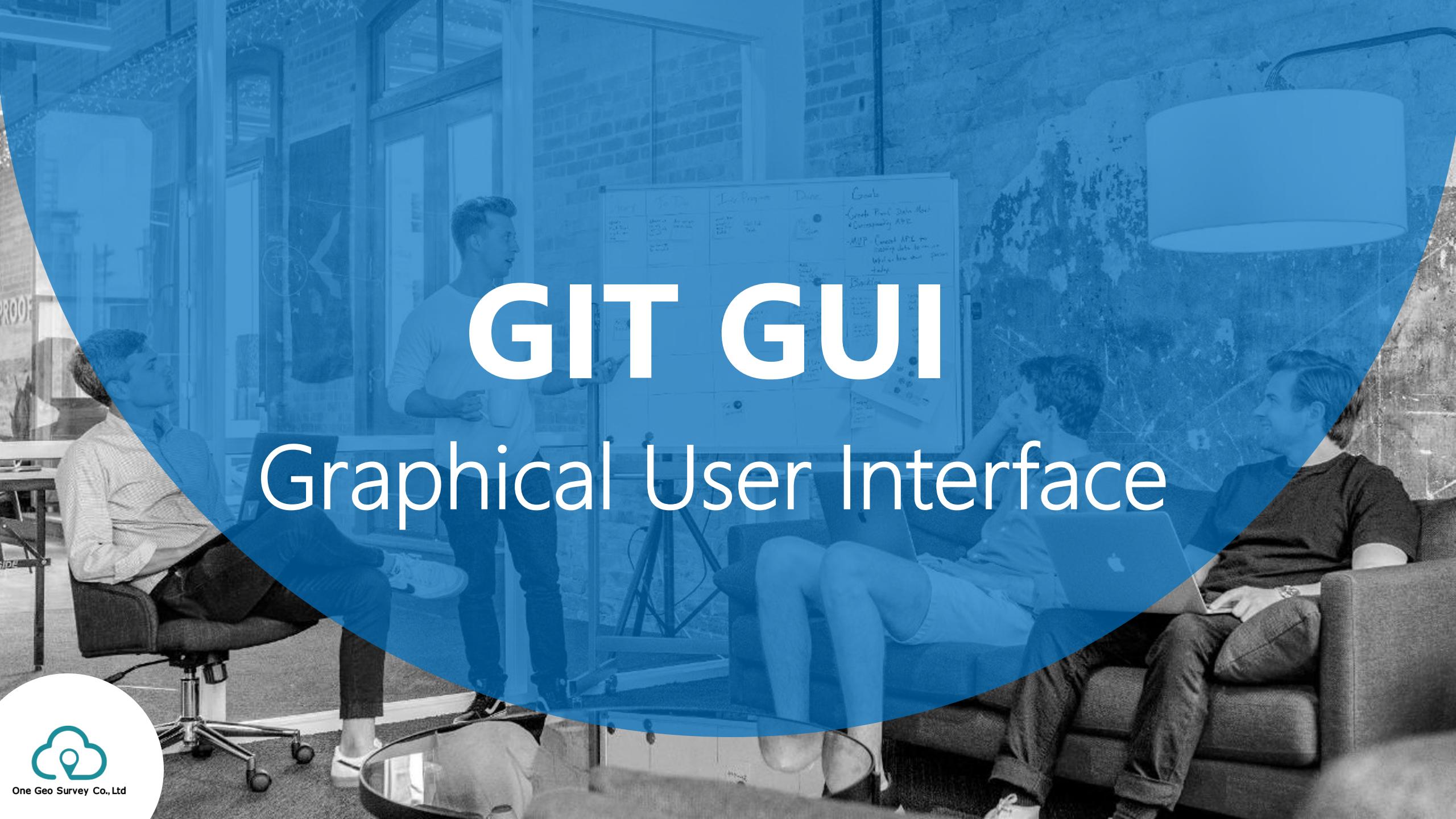
Git Repositories



GIT Repositories

รูปแบบการทำงาน





GIT GUI

Graphical User Interface



GIT GUI

เครื่องมือสำหรับการใช้งาน Git อย่างง่าย



GitHub Desktop



Source Tree



Gitkraken



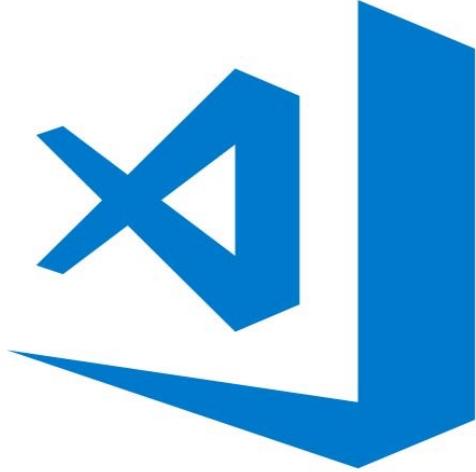
GitFork

ข้อมูลเพิ่มเติม

<https://git-scm.com/downloads/guis>

GIT Code Editor

เครื่องมือสำหรับการใช้งาน Git อย่างง่าย



**Visual Studio Code
Extensions**

Git GUI



GIT FLOW

GIT FLOW CONCEPT

รูปแบบการพัฒนาที่ใช้การทำงานของ Branch git ในรูปแบบการแตก Branch ออกมาเพื่อพัฒนา feature ที่ต้องการ

Master

- เป็น Branch หลักที่เอาไว้ขึ้นไปที่ production ให้กับผู้ใช้งาน
- ในส่วนนี้จะไม่ทำการพัฒนา

Develop

- เป็น Branch ที่ดำเนินการพัฒนาเป็นหลัก
- พัฒนาโดยสร้าง Branch feature ใหม่ขึ้นมา

Feature

- เป็น Branch ที่ถูกสร้างขึ้นจาก Branch develop
- พัฒนา feature ใหม่
- เสร็จแล้ว จะทำการ Merge ไปที่ Branch develop

GIT FLOW CONCEPT

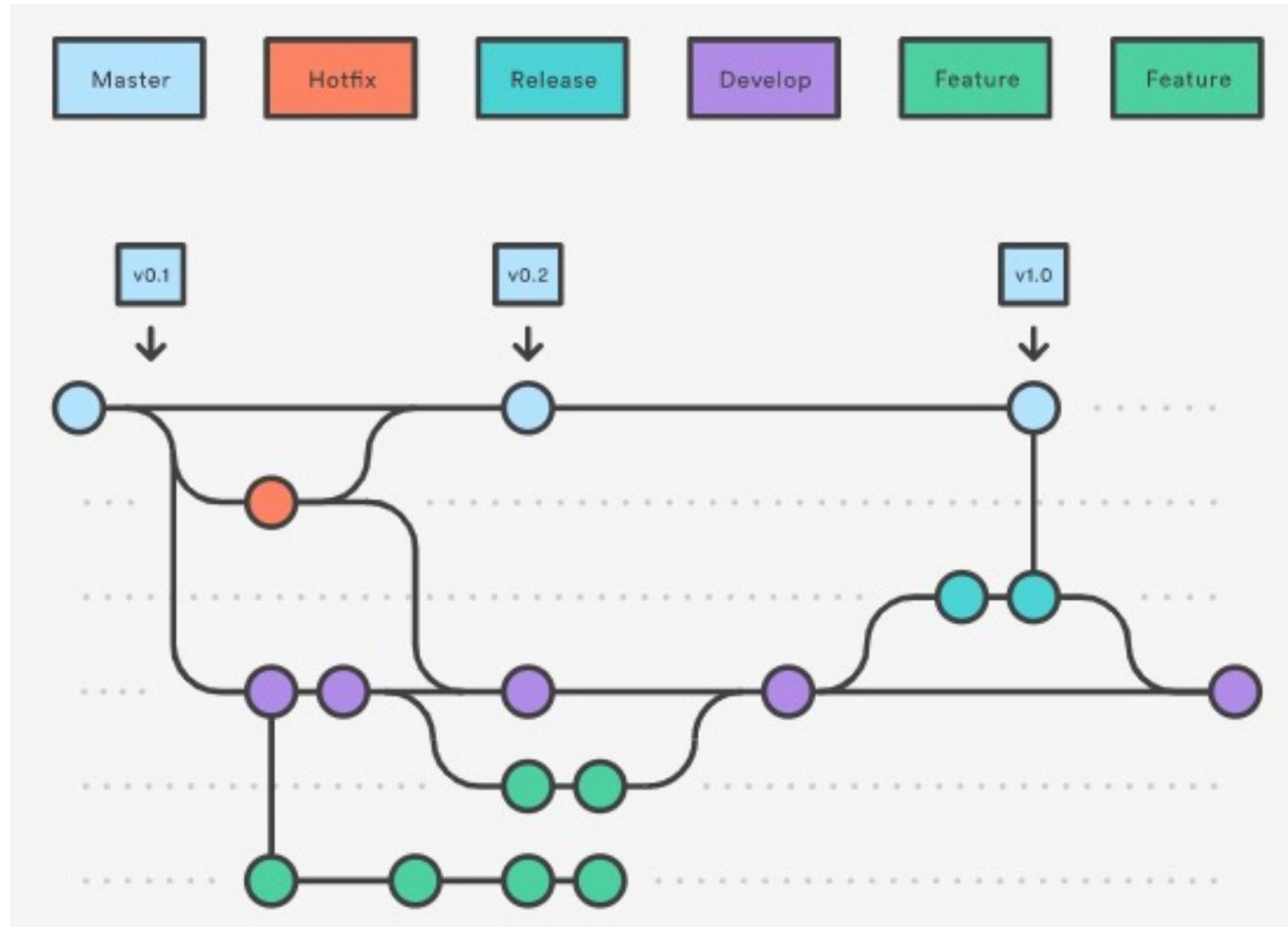
Release

- Branch ที่ถูกสร้างขึ้นจาก Branch develop
- Branch ที่ใช้สำหรับดำเนินการ QA เพื่อที่จะทำการ Test หรือปรับแต่งรายละเอียดให้สมบูรณ์ เช่น ทำการแก็บักก่อนที่จะ release
- เมื่อทำการ QA เสร็จแล้ว จะทำการ Merge ไปที่ Branch master และ Branch develop

Hotfix

- เจอบั๊กอยู่ในเวอร์ชันที่ release และต้องทำการแก้ไขโดยด่วน จะทำการแตก Branch ออกจาก master

GIT FLOW CONCEPT



RESOURCE : [Introduction to Git Flow. Git Flow is an abstract idea of a Git... | by Mansi Babbar | Level Up Coding \(gitconnected.com\)](#)

MERGE REQUEST

MERGE REQUEST เป็นหนึ่งในช่องทางที่จะสื่อสารกับคนภายในทีมได้ว่า โค้ดจาก Feature branch ที่เราจะเอาขึ้นนั้นมีการเปลี่ยนแปลง อะไรบ้าง

- ดูว่า Code มีสิ่งที่ไม่จำเป็นขึ้นไปหรือไม่
- เช็คความเรียบร้อยของงาน
- แนะนำการเขียน Code แบบที่ถูกต้องหรือมีประสิทธิภาพที่ดีกว่า
- ตรวจสอบทิศทางการเขียน Code ว่าถูกต้องตามมาตรฐานของทีมหรือไม่
- ในบางครั้งเราจะไม่มั่นใจว่า Function ที่เราเขียนขึ้นมา้นั้นดีไหม Reviewers ก็จะช่วยให้คำแนะนำได้

MERGE REQUEST

The screenshot shows a dark-themed user interface for managing merge requests. On the left, a sidebar menu includes 'Project overview', 'Pinned' (selected), 'Manage', 'Plan', 'Code', 'Merge requests' (0), 'Repository', 'Branches', 'Commits', 'Tags', 'Repository graph', 'Compare revisions', and 'Snippets'. The main area displays a list of merge requests with filters for 'Open' (0), 'Merged' (37), 'Closed' (1), and 'All' (38). A search bar and sorting options are also present. A blue arrow points to the 'New merge request' button. The 'New merge request' dialog box is open, showing fields for 'Source branch' (set to 'dowell-account-reconciliation/dowell-a...') and 'Target branch' (set to 'main'). A dropdown menu for selecting the source branch is visible. Below these fields is a message: 'Select a branch to compare' with a 'Compare branches and continue' button. A blue arrow points to this button. To the right of the target branch field, a commit card is shown for a pull request: 'dowell-account-reconciliation/dowell-a...' (author: Sukapat Taparod, date: Mar 25, 2024, commit hash: bddb3585). A blue arrow points to the author's name.

MERGE REQUEST

New merge request

From 8-breadcrumbs into main Change branches

Title (required)

Mark as draft
Drafts cannot be merged until marked ready.

Description

Closes #8

Supports Markdown. For quick actions, type /.

Add [description templates](#) to help your contributors to communicate effectively!



Assignee

Unassigned [Assign to me](#)

Reviewer

Unassigned

Milestone

Select milestone

Labels

Labels

Merge options

Delete source branch when merge request is accepted.
 Squash commits when merge request is accepted. [?](#)

[Create merge request](#) [Cancel](#)



THANK YOU