

การทดลองที่ 1 การใช้งาน Repository เปื้องต้น

วัตถุประสงค์

- เพื่อให้ผู้เรียนรู้และเข้าใจแนวคิดในการใช้ Repository
- เพื่อให้ผู้เรียนสามารถใช้ Repository (Github) เปื้องต้นได้

ทฤษฎีก่อนการทดลอง

Git

Git¹ เป็นระบบควบคุมเวอร์ชัน (Version control systems) เป็นเครื่องมือที่ใช้บริหารจัดการการเปลี่ยนแปลงของไฟล์ต่าง ๆ ใน project การบันทึกการแก้ไขไฟล์แต่ละครั้งจะเรียกว่ารุ่น (revision) ซึ่งแต่ละรุ่นของการเปลี่ยนแปลงจะถูกกำกับด้วยการประทับเวลา (timestamp) และบุคคลที่ทำการเปลี่ยนแปลง ดังนั้น หากเกิดความผิดพลาดหรือเสียหายจากการแก้ไข เรา ก็จะสามารถย้อนเวลากลับไปยังการแก้ไขครั้งก่อนๆ ที่สมบูรณ์ได้ตามต้องการ ถือได้ว่าระบบควบคุมเวอร์ชันเป็นระบบพื้นฐานที่นิยมใช้ในบริการจัดการ source code ของโปรแกรม ซึ่งจริง ๆ แล้ว เราสามารถใช้ระบบควบคุมเวอร์ชันกับไฟล์ชนิดใดๆ หรืองานชนิดใดๆ ก็ได้ ไม่เฉพาะ source code ของโปรแกรมเท่านั้น ในปัจจุบัน มีระบบควบคุมเวอร์ชันให้เลือกใช้หลากหลาย ทั้งเป็นแบบฟรีและมีค่าใช้จ่าย (เช่น Git, Mercurial, Subversion) โดย Git จะได้รับความนิยมมากกว่าชนิดอื่น ๆ การทำงานของ Git นั้นจะมีพื้นที่เก็บไฟล์ ซึ่งเรียกว่า ‘repositories’ ซึ่งเราสามารถติดตั้งบน server ได้ ก็ได้ แต่ server บริการ git ที่ได้รับความนิยมในปัจจุบันได้แก่ Github, Gitlab, Bitbucket เป็นต้น ข้อดีของการใช้ server รวมก็คือ สามารถแบ่งปันและร่วมมือช่วยเหลือกันในแก้ไขโปรแกรมได้จากทุกคนทั่วโลก ลักษณะเฉพาะอย่างหนึ่งของ Git ก็คือ ใน folder ที่ชื่อ .git บนคอมพิวเตอร์ของเราจะเก็บทุกสิ่งที่เก็บบน server จึงมั่นใจได้ว่า เราสามารถทำงานกับระบบควบคุมเวอร์ชันได้ทั้งแบบออนไลน์และออฟไลน์ และหากเกิดกรณีที่ repository บน server เสียหาย เรา ก็สามารถนำทุกอย่างที่เก็บบนเครื่องกลับขึ้นไปเก็บบน server ได้

Github

Github เป็นบริษัทหนึ่ง ที่ให้บริการ Git repository บนพื้นฐานของรีบ (web-based Git repository hosting) โดย Github จะให้พื้นที่เราสร้าง repository สำหรับโปรเจค ให้บริการฟังก์ชันการทำงานพื้นฐานของระบบ git เช่น การ branches, merges, และ commits อีกทั้งยังให้พื้นที่สำหรับแจ้งข้อผิดพลาด บัก หรือความต้องการเพิ่มเติม features ต่างๆ ตลอดจนมีความสามารถในการเขียนคำอธิบายแบบ wiki ใน repository นั้น ๆ ด้วย Github เป็นบริษัทที่มีมูลค่าประมาณ 2 พันล้าน USD, มีผู้ใช้ประมาณ 20 ล้านคน มี repositories ประมาณ 40 ล้าน และในจำนวนเหล่านั้น มีโปรเจคที่สำคัญมากรวมอยู่ด้วย เช่น kernel ของ Linux , source code ของ dotnet framework จากไมโครซอฟท์ และอื่นๆ ทำให้มีความมั่นใจในระดับหนึ่งว่าถ้า Github เกิดล่มขึ้นมา ก็จะมีเพื่อนร่วมชาติรอมอีกไม่น้อย

¹ "Git · GitHub." Accessed August 10, 2017. <https://github.com/git>.

ขั้นตอนการทดลอง

1. เริ่มใช้งาน Github

ในการใช้งาน Github เราจะต้องมีบัญชีผู้ใช้งาน Github ซึ่งทาง Github จะให้บริการฟรีแบบไม่จำกัดจำนวน repository ซึ่งจะเป็นแบบ public หรือ private ก็ได้ repository แบบ public นั้น จะสามารถมองเห็นได้จากทุกคน ส่วน repository ที่เป็นแบบ private เราจะสามารถกำหนดบุคคลที่อนุญาตให้เห็น repository ของเราได้ ซึ่งจะสะดวกในการทำ project ที่เป็นความลับ

1.1 สร้างบัญชีผู้ใช้งานบน Github

การสร้างบัญชีผู้ใช้งาน Github ให้ไปที่ <https://github.com/join> จากนั้น ให้กรอกรายละเอียด ชื่อผู้ใช้ (User name) จะถูกนำไปใช้ในหลายๆ ที่ ดังนั้นควรเป็นชื่อที่จำง่ายและพิมพ์ได้สะดวก มีฉะนั้นจะเสียเวลาในการทำงาน

The screenshot shows the 'Join GitHub' page. At the top, there are three tabs: 'Step 1: Create personal account', 'Step 2: Choose your plan', and 'Step 3: Tailor your experience'. The first tab is active. Below it, there's a section for creating a personal account with fields for 'Username', 'Email Address', and 'Password'. To the right, there's a sidebar titled 'You'll love GitHub' listing benefits like 'Unlimited collaborators', 'Unlimited public repositories', 'Great communication', 'Frictionless development', and 'Open source community'. At the bottom, there's a note about agreeing to the Terms of Service and Privacy Policy, followed by a green 'Create an account' button.

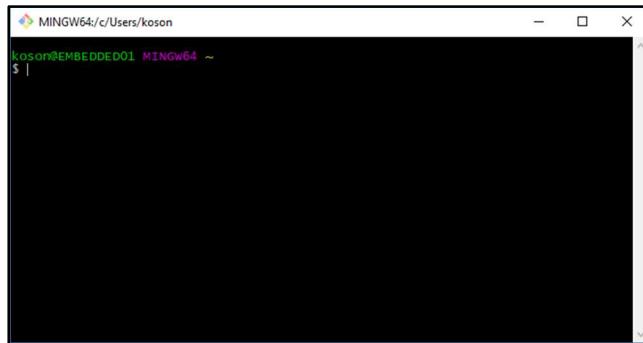
รูปที่ 1.1 การสร้างบัญชี Github

1.2 ติดตั้งโปรแกรม Git

1.2.1 ดาวน์โหลดโปรแกรม Git จาก <https://git-scm.com/downloads> โดยเลือกโปรแกรมติดตั้งให้ตรงกับระบบปฏิบัติการที่ใช้ โปรแกรมที่ดาวน์โหลดมา จะมี GUI ให้เราใช้งานด้วยซึ่งมีชื่อเรียกว่า Github desktop แต่ถ้าหากสนใจที่จะใช้ Git GUI Clients ตัวอื่นๆ ก็สามารถศึกษาได้ที่ <https://git-scm.com/downloads/guis>

1.2.2 ติดตั้งโปรแกรม Git ตามคำแนะนำของโปรแกรมติดตั้ง

1.2.3 เปิดโปรแกรม Git bash จะได้หน้าต่าง terminal ที่ทำงานใน text mode



รูปที่ 1.2 หน้าต่าง terminal ของ git bash

ผลการทดลอง

เมื่อกดเปิดโปรแกรม จะมีหน้าต่าง terminal ของ git bash ขึ้นมา

A screenshot of a terminal window titled "MINGW64:c/Users/Dell/OneDrive - KMITL/siripaksorn". The window has a standard title bar with minimize, maximize, and close buttons. The main area is a black terminal window with a cursor at the bottom-left. At the top left of the terminal window, there is some small yellow text that appears to be part of the window's header or a system message.

1.2.4 ทดสอบว่าสามารถใช้งาน Git บนเครื่องของเราได้หรือไม่ ให้พิมพ์คำสั่งต่อไปนี้

```
$ git
```

ถ้า terminal ตอบกลับมาว่าไม่รู้จักคำสั่ง git และแสดงว่าการติดตั้งยังไม่สมบูรณ์ ให้กลับไปตรวจสอบขั้นตอน 1.2.2 ให้ติดตั้งเรียบร้อย

```

MINGW64:/c/Users/koson
$ git
usage: git [--version] [--help] [-c <path>] [-c name=value]
           [--exec-path[=<path>]] [--html-path] [-m man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [-no-replace-objects] [-bare]
           [-git-dir=<path>] [-work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

These are common Git commands used in various situations:
start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one
work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv        Move or rename a file, a directory, or a symlink
  reset     Reset current HEAD to the specified state
  rm        Remove files from the working tree and from the index
examine the history and state (see also: git help revisions)
  bisect    Use binary search to find the commit that introduced a bug
  grep      Print lines matching a pattern
  log       Show commit logs
  show     Show various types of objects
  status    Show the working tree status
grow, mark and tweak your common history
  branch   List, create, or delete branches
  checkout Switch branches or restore working tree files
  commit   Record changes to the repository
  diff     Show changes between commits, commit and working tree, etc
  merge   Join two or more development histories together
  rebase   Reapply commits on top of another base tip
  tag      Create, list, delete or verify a tag object signed with GPG
collaborate (see also: git help workflows)
  fetch   Download objects and refs from another repository

```

รูปที่ 1.3 ผลการทดลองพิมพ์คำสั่ง git

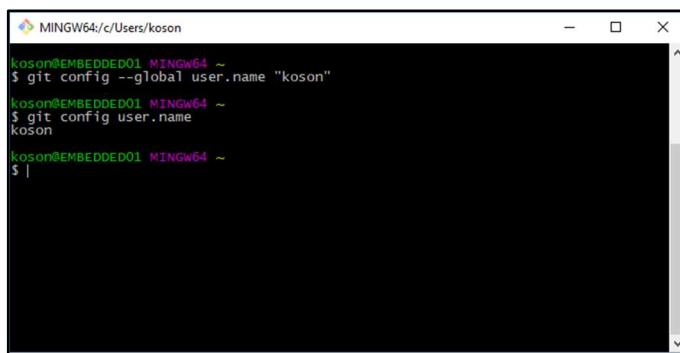
<h3>ผลการทดลอง</h3> <p>เมื่อพิมพ์คำสั่ง git ไป โปรแกรมจะแสดงข้อมูลเกี่ยวกับคำสั่งต่าง ๆ ที่โปรแกรมสามารถทำได้</p> <pre> DELL@SIRIPAKORN MINGW64 ~/OneDrive - KMITL/siripaksorn \$ git usage: git [--version] [--help] [-c <path>] [-c name=value] [--exec-path[=<path>]] [--html-path] [-m man-path] [--info-path] [-p --paginate -P --no-pager] [-no-replace-objects] [-bare] [-git-dir=<path>] [-work-tree=<path>] [--namespace=<name>] <command> [<args>] These are common Git commands used in various situations: start a working area (see also: git help tutorial) clone Clone a repository into a new directory init Create an empty Git repository or reinitialize an existing one work on the current change (see also: git help everyday) add Add file contents to the index mv Move or rename a file, a directory, or a symlink restore Restore working tree files rm Remove files from the working tree and from the index sparse-checkout Initialize and modify the sparse-checkout examine the history and state (see also: git help revisions) bisect Use binary search to find the commit that introduced a bug diff Show changes between commits, commit and working tree, etc grep Print lines matching a pattern log Show commit logs show Show various types of objects status Show the working tree status grow, mark and tweak your common history branch List, create, or delete branches commit Record changes to the repository merge Join two or more development histories together rebase Reapply commits on top of another base tip reset Reset current HEAD to the specified state switch Switch branches tag Create, list, delete or verify a tag object signed with GPG collaborate (see also: git help workflows) fetch Download objects and refs from another repository pull Fetch from and integrate with another repository or a local branch Update remote refs along with associated objects 'git help -a' and 'git help -g' list available subcommands and some concept guides. See 'git help <command>' or 'git help <concept>' to read about a specific subcommand or concept. See 'git help git' for an overview of the system. DELL@SIRIPAKORN MINGW64 ~/OneDrive - KMITL/siripaksorn \$ </pre>

1.2.5 บอกให้ Git รู้จักชื่อของเรา โดยพิมพ์คำสั่งต่อไปนี้²

```
$ git config --global user.name "USER NAME"
```

ในกรณีที่เราต้องการทราบชื่อผู้ใช้ปัจจุบัน สามารถสั่งให้ Git รายงานออกมากด้วยการพิมพ์คำสั่งต่อไปนี้

```
$ git config user.name
```

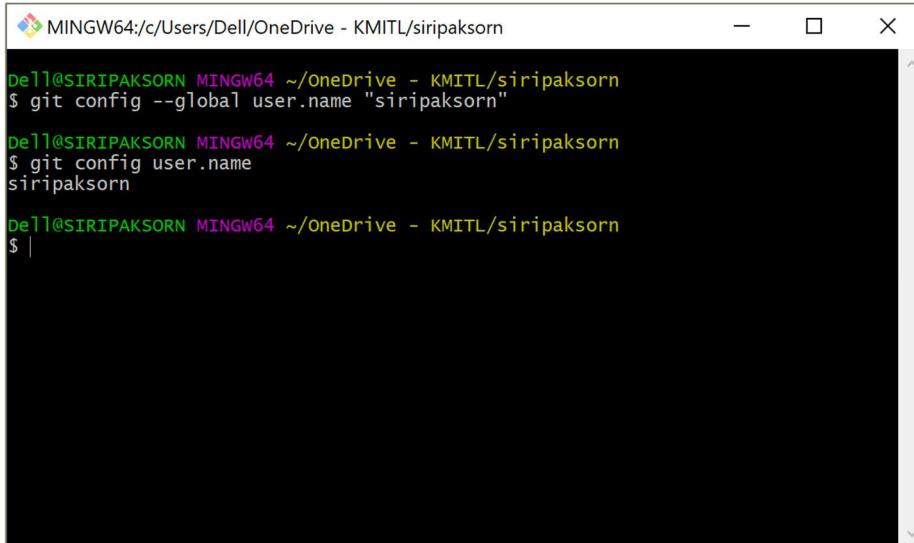


```
MINGW64:/c/Users/koson
$ git config --global user.name "koson"
koson@EMBEDDED01 MINGW64 ~
$ git config user.name
koson
koson@EMBEDDED01 MINGW64 ~
$ |
```

รูปที่ 1.4 git config --global user.name

ผลการทดลอง

เมื่อพิมพ์คำสั่งตามภาพ Git จะจำชื่อที่เราป้อน เพื่อนำไปใช้ในตอนที่เรา nave ข้อมูลขึ้น server



```
MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn
$ git config --global user.name "siripaksorn"
dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn
$ git config user.name
siripaksorn
dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn
$ |
```

² "Setting your username in Git - User Documentation - GitHub Help." Accessed August 10, 2017. <https://help.github.com/articles/setting-your-username-in-git/>.

1.2.6 บอกให้ Git รู้จัก email ของเรา โดยพิมพ์คำสั่งต่อไปนี้

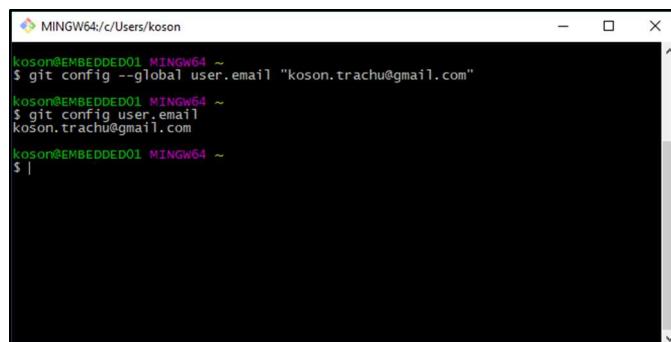
```
$ git config --global user.email "USER EMAIL ADDRESS"
```

ในการนี้ที่เราต้องการทราบชื่อผู้ใช้ปัจจุบัน สามารถสั่งให้ Git รายงานออกมายังการพิมพ์คำสั่งต่อไปนี้

```
$ git config user.email
```

หมายเหตุ email ที่ใช้จะต้องตรงกับ email ที่ลงทะเบียนไว้กับ Github มีฉะนั้นจะไม่สามารถเปลี่ยนข้อมูลขึ้นไปบน server ได้

เมื่อทำในขั้นตอน 1.2.5 และ 1.2.6 เรียบร้อยแล้ว การทำงานใดๆ บน Github ก็จะปรากฏชื่อและ Email ของเรากำกับไว้เสมอ

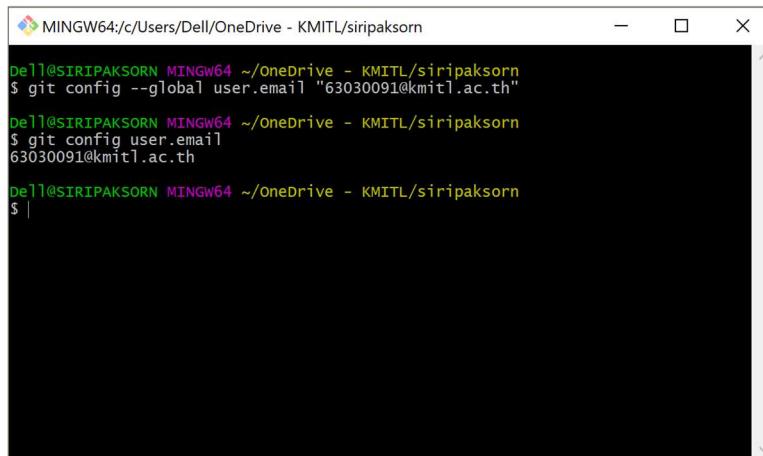


```
MINGW64:/c/Users/koson
koson@EMBEDDED01 MINGW64 ~
$ git config --global user.email "koson.trachu@gmail.com"
koson@EMBEDDED01 MINGW64 ~
$ git config user.email
koson.trachu@gmail.com
koson@EMBEDDED01 MINGW64 ~
$ |
```

รูปที่ 1.5 git config --global user.email

ผลการทดลอง

เมื่อพิมพ์คำสั่งตามภาพ Git จะจำ email ที่เราป้อน เพื่อนำไปใช้ในตอนที่เรานำข้อมูลขึ้น server



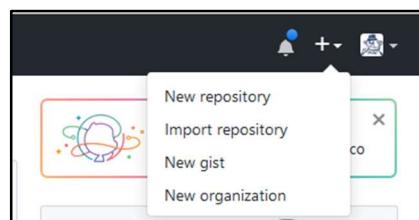
```
MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn
Dell@sIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn
$ git config --global user.email "63030091@kmitl.ac.th"
Dell@sIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn
$ git config user.email
63030091@kmitl.ac.th
Dell@sIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn
$ |
```

1.3 สร้าง repository (บน server)

Repository เป็นพื้นที่สำหรับเก็บ project ของเรา ซึ่งไม่ได้มีหมายความถึงเฉพาะ source code เท่านั้น repository ยังสามารถประกอบด้วยไฟล์ทุกชนิด ไม่ว่าจะเป็น Word Document, spread sheet, presentation, เอกสารการอภิเคราะห์และออกแบบซอฟต์แวร์ ไฟล์มีเดียภาพและเสียง รวมไปถึงเอกสาร Wiki ในลักษณะ html ด้วย ดังนั้น ในการทำโครงการพัฒนาซอฟต์แวร์ เราสามารถนำทุกสิ่งที่จำเป็นสำหรับการทำงาน มาใส่ไว้ใน repository และเมื่อเพื่อนร่วมทีมหรือ user ใดๆ ทำสำเนา repository ของเราไป เขายังจะได้ทุกสิ่งทุกอย่างไปอย่างครบถ้วน ดังนั้นจึงอาจพูดได้ว่าเราสามารถใช้ repository เป็นเครื่องมือบริหารโครงการที่มีประสิทธิภาพได้เช่นกัน

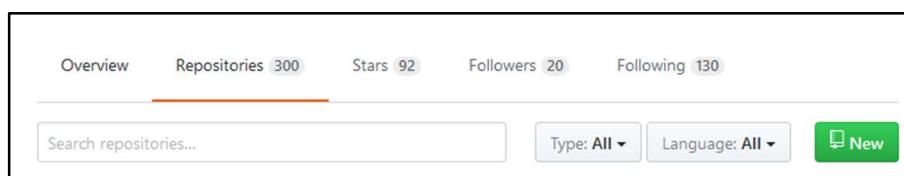
1.3.1 การสร้าง repository บน Github สามารถสร้างได้หลายวิธีด้วยกัน เช่น

(1) การสร้าง repository โดยการคลิกที่ปุ่มเครื่องหมาย “+” ที่ด้านบนขวาของหน้าจอ Github และเลือก new repository



รูปที่ 1.6 การสร้าง repository โดยการคลิกที่ปุ่มเครื่องหมาย “+”

(2) การสร้าง repository โดยการคลิกที่ปุ่ม New สีเขียว



รูปที่ 1.7 การสร้าง repository โดยการคลิกที่ปุ่ม New

(3) การสร้าง repository โดยลิงค์ <https://github.com/new>

นอกจาก 3 วิธีข้างต้น ซึ่งจะพาเรามาสร้าง repository บนเว็บแล้ว เรายังสามารถสร้าง repository โดยใช้ command line บน terminal (ศึกษาได้จาก [adding-an-existing-project-to-github-using-the-command-line³](#))

³ [Adding an existing project to GitHub using](#) Accessed August 11, 2017.

<https://help.github.com/articles/adding-an-existing-project-to-github-using-the-command-line/>

1.3.2 กำหนดชื่อและชนิดของ repository

การใช้วิธีการ 3 วิธีแรก ในข้อ 1.3.1 จะได้ผลอย่างเดียวกัน คือ Github จะพามาหน้าสำหรับสร้าง repository

- ในช่อง **Repository name** ให้ใส่ชื่อของ repository เนื่องจากบ่อยครั้งที่เราต้องใช้งานคำสั่งต่าง ๆ บน terminal ซึ่งต้องพิมพ์ชื่อ repository เอง ดังนั้นชื่อของ repository จะต้องมีความหมายในตัว เข้าใจง่าย กระชับ
- ในช่อง **Description (optional)** เพิ่มคำอธิบายสั้นๆ เกี่ยวกับ repository เพื่อให้ชาวโลกอ่านแล้วเห็นภาพรวมของ repository ได้อย่างรวดเร็ว
- ชนิดของ **repository** นั้น ถ้าหากเป็นโปรเจคที่เป็นความลับ ไม่อาจเปิดเผยต่อชาวโลกได้ เช่นประกอบด้วยฐานข้อมูลในงานวิจัย คะแนนแล็บของนักศึกษา ซึ่อ URL, user name, password ที่เขียนลงมาใน source code เราอาจจะเลือก เป็น private ซึ่งอาจจะต้องมีค่าใช้จ่ายในการสมัครสมาชิกพิเศษ หรือไม่ก็ต้องเป็น academic account ในที่นี้ให้เลือก เป็น public
- ถ้าเราทำเครื่องหมาย หน้าข้อความ Initialize this repository with a README เพื่อให้เราสามารถอ่านบรรยาย คร่าวๆ เกี่ยวกับ repository ได้
 - **เดียวกัน.... ในขั้นตอนนี้ ยังไม่ต้องทำเครื่องหมาย เพราะเราจะทดลองสร้างโดยใช้ command line tool**
- เลือกว่าจะเพิ่ม .gitignore หรือ license file ด้วยหรือไม่ โดย .gitignore นี้จะบอก Git ว่าไม่ต้องสนใจที่จะติดตามไฟล์ ชนิดใดบ้าง โดย Git จะกำหนดชนิดของไฟล์ให้เบื้องต้น เช่น ถ้าเราเลือก .gitignore เป็นภาษา C++ และ Git จะเพิ่ม ชนิดของไฟล์ต่างๆ ที่เป็นผลจากการคอมไพล์ไว้ในรายการที่เพิกเฉย (เช่น ไฟล์ที่มีนามสกุล .exe) ซึ่งไฟล์เหล่านั้น มักจะเกิดจากการคอมไпал์โปรแกรม ไม่ใช่ไฟล์ที่เราเป็นคนแก้ไข source code จึงไม่จำเป็นที่จะต้องนำไปเก็บบน repository ให้เสื่อมเปลืองพื้นที่ สามารถดูเทมเพลตของ .gitignore ได้จาก A collection of useful .gitignore templates⁴
 - **ยังไม่ต้องเลือก.gitignore เช่นเดียวกัน**

⁴ "A collection of useful .gitignore templates" Accessed August 11, 2017.
<https://github.com/github/gitignore>.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Repository template
Start your repository with a template repository's contents.
[No template ▾](#)

Owner * **Repository name ***
 /

Great repository names are short and memorable. Need inspiration? How about [potential-palm-tree](#)?

Description (optional)

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.

Initialize this repository with:
Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

รูปที่ 1.8 การสร้าง repository

ผลการทดลอง

เมื่อทำการตามขั้นตอนจะขึ้นหน้าเว็บให้เรากำหนดชื่อและค่าต่าง ๆ ของ 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 * **Repository name ***
 /

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

Description (optional)

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.

Initialize this repository with:
Skip this step if you're importing an existing repository.

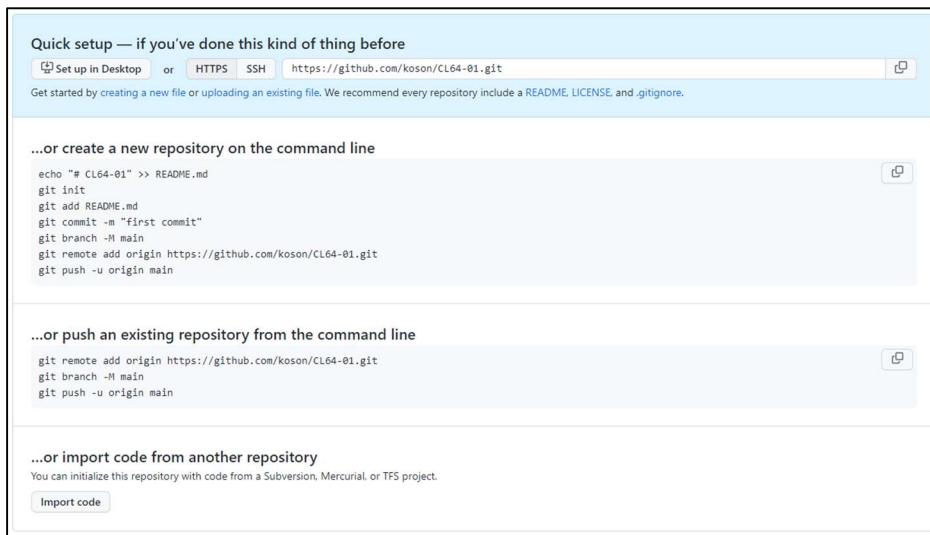
Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

- คลิกปุ่ม Create repository สีเขียว
Github จะสร้าง repository ให้ตามต้องการ



รูปที่ 1.9 repository ที่ได้จากการสร้างในข้อ 1.3

ผลการทดลอง

เมื่อสร้าง repository แล้ว จะขึ้นหน้าเว็บนี้ซึ่งจะแสดงคำสั่งต่าง ๆ ที่เราสามารถนำไปใช้ได้ และจะแสดง URL ด้วย

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

or <https://github.com/siripaksorn/CL64-01.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 "# CL64-01" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/siripaksorn/CL64-01.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/siripaksorn/CL64-01.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

หมายเหตุ ให้เปิดหน้าเพจนี้ค้างไว้ เพราะเราต้องมาดูผลการเปลี่ยนแปลงในภายหลัง

1.4 สร้าง git บนเครื่องคอมพิวเตอร์ (Local)

Repository ที่สร้างขึ้นในข้อ 1.3 นั้น เป็น repository ที่อยู่บน server ในขณะที่เรากำลังแก้ไข source code ซึ่งมักจะเป็นการแก้ไขเด็ก ๆ น้อย ๆ การทำงานของ git จะเน้นทำงานที่ local เป็นหลัก ต่อเมื่อเราได้พัฒนา source code จนถึงจุดหนึ่ง ที่คิดว่าสามารถเผยแพร่ เพื่อการทดสอบหรือใช้งาน เราจึงส่งขึ้นไปเก็บบน server

การทำสำเนาของ repository มาไว้บนเครื่อง (local) สามารถทำได้หลายวิธี ซึ่งเบื้องต้นนี้ เราจะศึกษาโดยการใช้งาน command line ซึ่งอาจจะพบกับความยุ่งยากบ้างในตอนแรกๆ แต่เมื่อใช้บ่อย ๆ จะชำนาญจะพบว่ามีความยืดหยุ่นสูงกว่าการใช้ GUI Clients หรือเมื่อศึกษาจนเข้าใจแล้วหันไปใช้ GUI Clients ก็จะสามารถเข้าใจถึงการทำงานของระบบ Git อย่างแท้จริง

1.4.1 การ clone repository ด้วย command line (git bash)

1) การเตรียมการเบื้องต้น

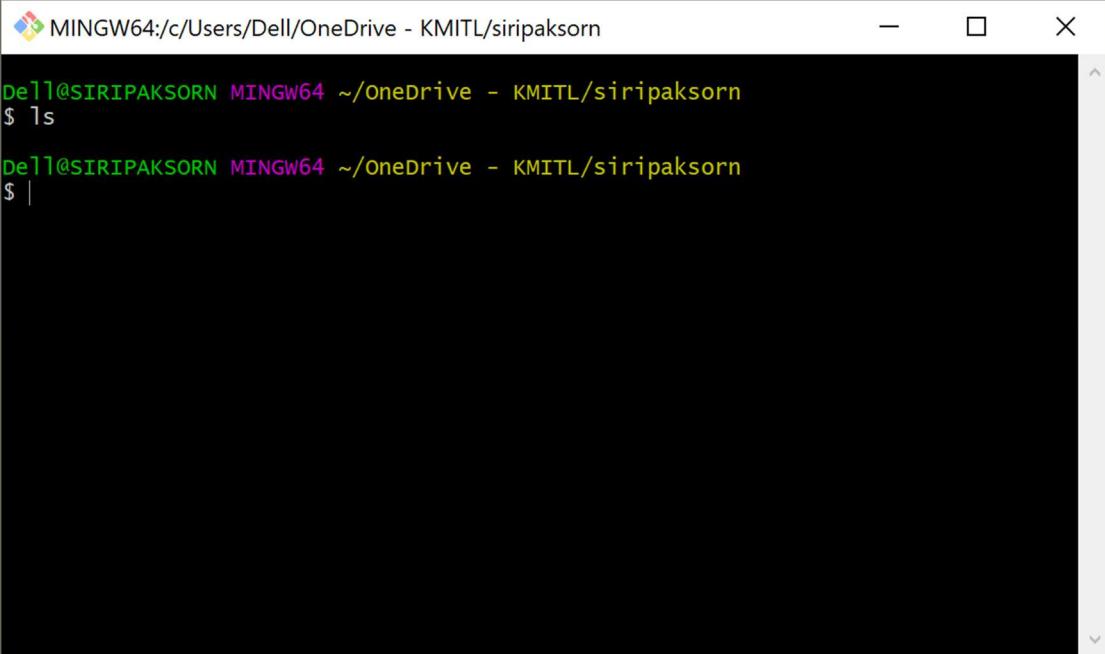
- ในหน้าต่าง git bash ให้พิมพ์คำสั่ง list ดูรายการของไฟล์และโฟลเดอร์

```
$ ls
```

เราจะเห็นรายการไฟล์ทุกแสดงขึ้นมา

ผลการทดลอง

เมื่อพิมพ์คำสั่ง ls จะไม่มีสิ่งใดเกิดขึ้น เพราะ ยังไม่มีไฟล์อยู่ในโฟลเดอร์



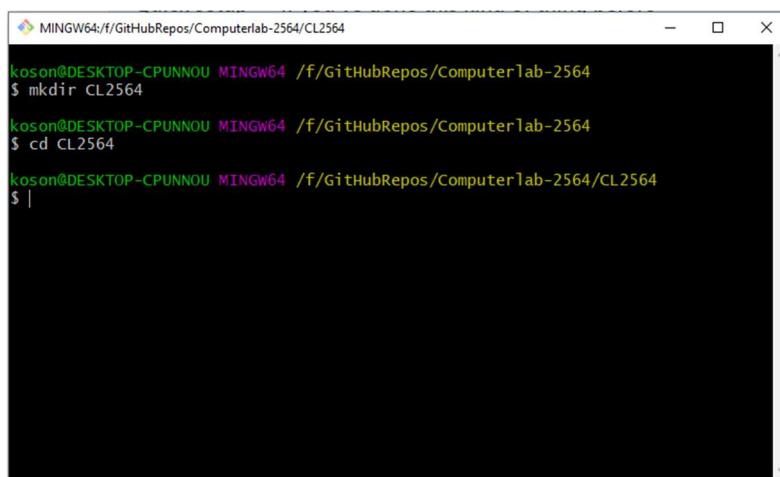
- สร้างโฟลเดอร์สำหรับเก็บงานในวิชาการทดลอง (ในที่นี้ชื่อว่า CL2564 ย่อมาจาก Computer Laboratory 2564) โดยใช้คำสั่ง

```
$ mkdir CL2564
```

- ย้ายเข้าไปอยู่ในโฟลเดอร์ที่สร้างขึ้น โดยใช้คำสั่ง

```
$ cd CL2564
```

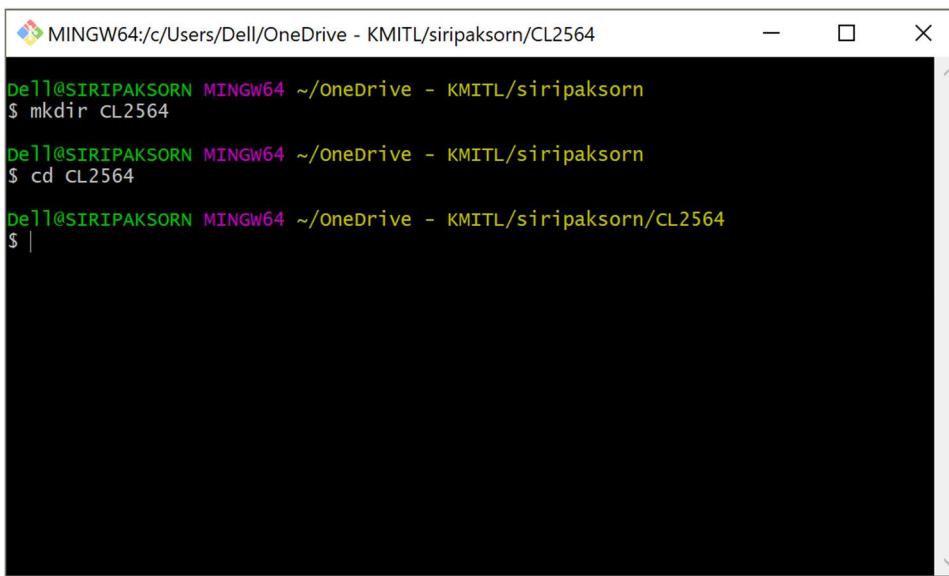
สังเกตุได้จาก git bash จะแสดงชื่อของโฟลเดอร์ปัจจุบันเป็นดังรูปที่ 1.10



รูปที่ 1.10 หน้าต่าง terminal ของ git bash เตรียมพร้อมสำหรับการ clone

ผลการทดลอง

เมื่อทำการตามขั้นตอนจะเป็นการสร้างโฟลเดอร์ git ชื่อว่า CL2564 และเปิด path ของโฟลเดอร์ CL2564



2) การทำสำเนา repository มาไว้บนเครื่องโดยการ clone

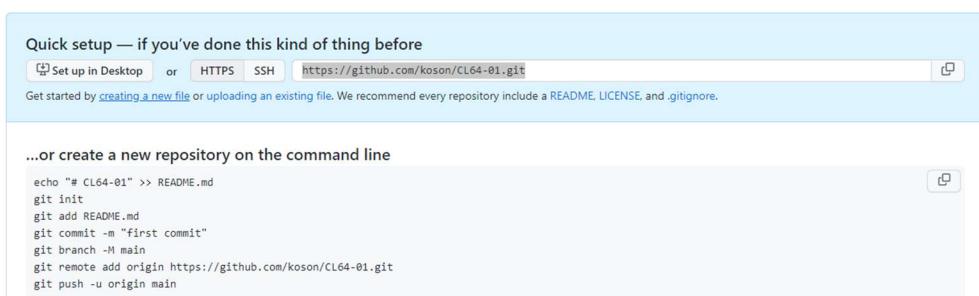
- ทำสำเนา repository มาไว้บนเครื่องโดยใช้คำสั่งที่มีรูปแบบดังต่อไปนี้

```
$ git clone https://github.com/[YOUR USERNAME]/[YOUR REPOSITORY NAME]
```

[YOUR USERNAME] คือ username ของเราบน github

[YOUR REPOSITORY NAME] คือชื่อ repository ของเราที่สร้างในข้อ 1.3

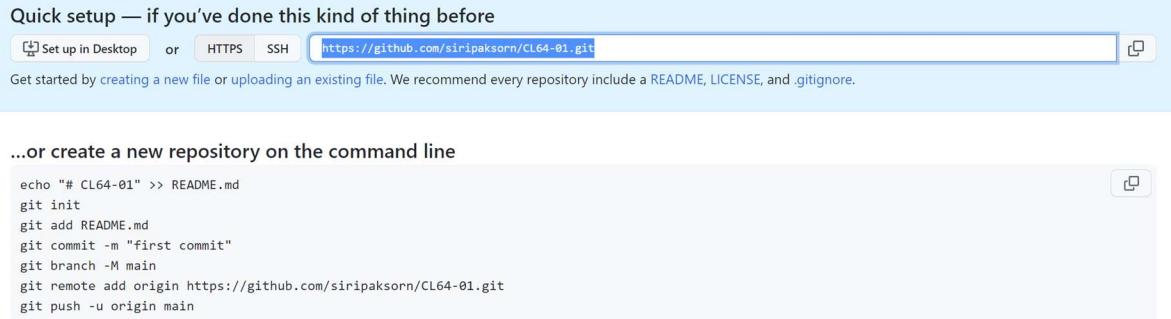
ถ้าจำไม่ได้ ก็ไม่เป็นไร ให้เข้าไปที่ repository ที่เพิ่งสร้างบน Github (ดูรูปที่ 1.9) จะเห็นว่ามี URL ของ repository สำหรับการ clone ดังรูปที่ 1.11 ให้เรากดปุ่ม copy ที่อยู่ด้านขวามือของ url



รูปที่ 1.11 URL สำหรับการ clone repository

ผลการทดลอง

copy URL ของ repository ที่เราสร้างไว้ เพื่อนำไป clone



- ใน git bash ให้พิมพ์คำสั่ง git clone ตามด้วย URL ที่คัดลอกมา
- เมื่อทำการ clone เรียบร้อย จะได้ผลดังรูปที่ 1.12

```

MINGW64:/f/GitHubRepos/Computerlab-2564/CL2564
koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564
$ mkdir CL2564

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564
$ cd CL2564

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564
$ git clone https://github.com/koson/CL64-01.git
Cloning into 'CL64-01'...
warning: You appear to have cloned an empty repository.

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564
$ |

```

รูปที่ 1.12 ผลการ clone repository

ผลการทดลอง

clone ข้อมูลที่อยู่ใน repository ลงโฟลเดอร์ในเครื่องของเรา

```

MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564
Dell@sIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn
$ mkdir CL2564

Dell@sIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn
$ cd CL2564

Dell@sIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564
$ git clone https://github.com/siripaksorn/CL64-01.git
Cloning into 'CL64-01'...
warning: You appear to have cloned an empty repository.

Dell@sIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564
$ |

```

- เรียกดรายการไฟล์เดอร์ (ด้วยคำสั่ง ls) และเปลี่ยนไฟล์เดอร์ (ด้วยคำสั่ง change directory :cd)

ผลการทดลอง

```
ใช้คำสั่ง ls เพื่อดูข้อมูลในไฟล์เดอร์ CL2564 ซึ่งจะเจอไฟล์เดอร์ CL64-01 ที่ clone มา และให้ใช้คำสั่ง cd CL64-01 เพื่อเข้าถึง path ของไฟล์เดอร์ CL64-01
```

The screenshot shows a terminal window titled 'MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564/CL64-01'. The terminal history is as follows:

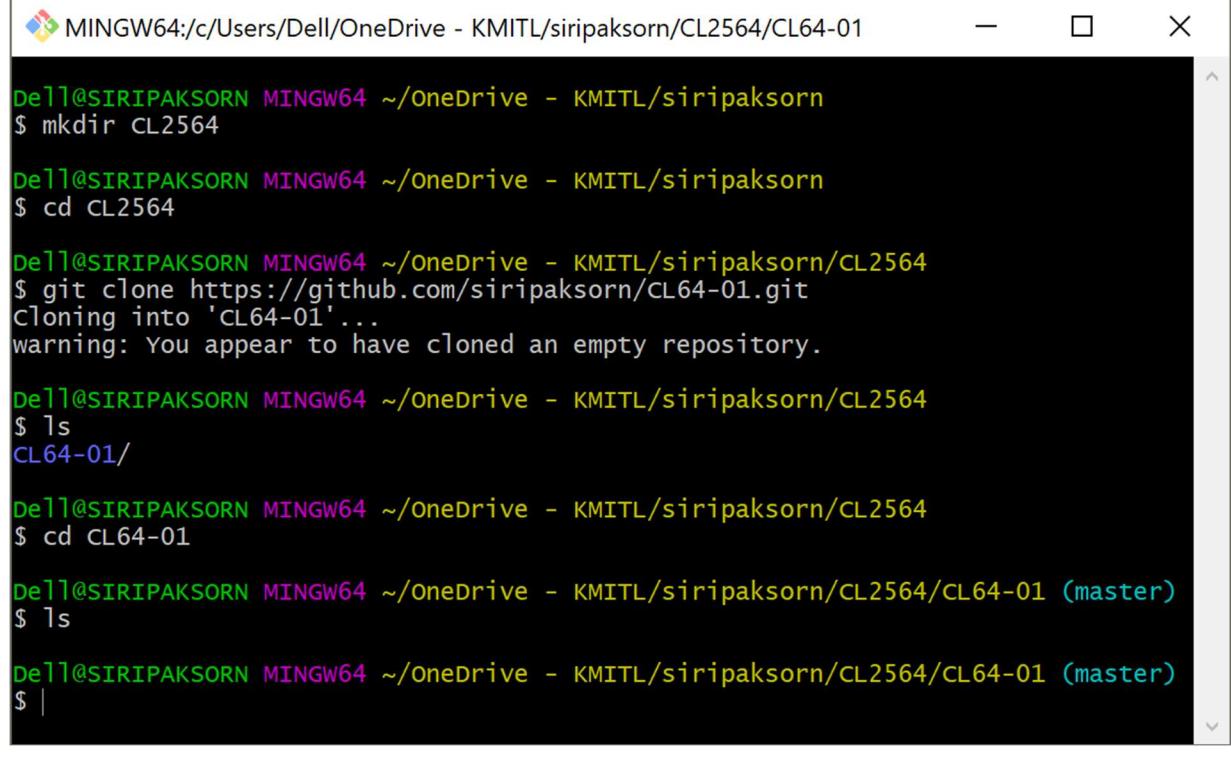
- \$ mkdir CL2564
- \$ cd CL2564
- \$ git clone https://github.com/siripaksorn/CL64-01.git
- Cloning into 'CL64-01'...
- warning: You appear to have cloned an empty repository.
- \$ ls
- CL64-01/
- \$ cd CL64-01
- \$ |

ตอนแรกจะพบว่ามีไฟล์เดอร์ชื่อ CL64-01 ซึ่งถูก clone มาจาก server จึงย้ายเข้าไปในไฟล์เดอร์นั้นแล้วจึงสั่ง ls เพื่อดูรายการไฟล์ พบว่า repository ของเราจะยังว่างเปล่า ดังรูปที่ 1.13

รูปที่ 1.13 ไฟล์ที่ถูก clone มาจาก repository

ผลการทดลอง

ใช้คำสั่ง ls เพื่อดูข้อมูลในโฟลเดอร์ CL64-01



```
MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564/CL64-01
Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn
$ mkdir CL2564

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn
$ cd CL2564

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564
$ git clone https://github.com/siripaksorn/CL64-01.git
Cloning into 'CL64-01'...
warning: You appear to have cloned an empty repository.

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564
$ ls
CL64-01/

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (master)
$ ls

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (master)
$ |
```

- ให้พิมคำสั่งต่อไปนี้ ครั้งละบรรทัด (พิมพ์ให้ครบบรรทัดแล้วเคาะ enter)

```
echo "# CL64-01" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/koson/CL64-01.git
git push -u origin main
```

รูปที่ 1.14 การเพิ่มไฟล์ README.md ให้กับ repository

ผลการทดลอง

เพิ่มไฟล์ README.md ในโฟลเดอร์ CL64-01 และตั้งค่าต่าง ๆ

```
MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564/CL64-01
Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (master)
$ echo "# CL64-01" >> README.md

Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (master)
$ git init
Reinitialized existing Git repository in c:/users/dell/oneDrive - KMITL/siripaksorn/CL2564/CL64-01/.git/

Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (master)
$ git add README.md
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory

Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (master)
$ git commit -m "first commit"
[master (root-commit) 1824621] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (master)
$ git branch -M main

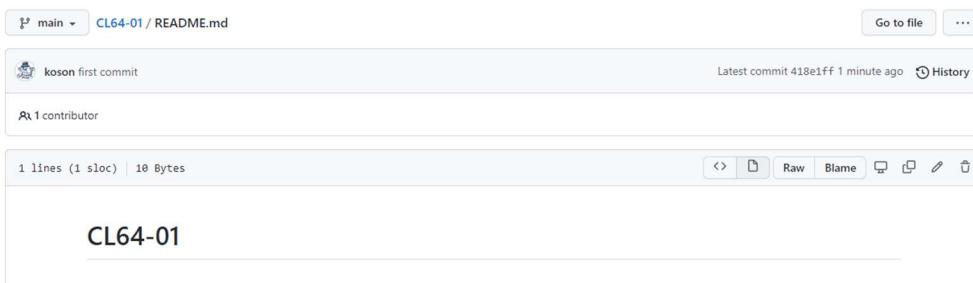
Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git remote add origin https://github.com/siripaksorn/CL64-01.git
error: remote origin already exists.

Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/siripaksorn/CL64-01.git
 * [new branch]      main    -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.

Dell@sIRIPAKSORN MINGW64 ~/oneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ |
```

จากรูปที่ 1.14 จะได้ผลการทำงานดังรูปที่ 1.15 ซึ่งจะเห็นว่า บางคำสั่งอาจจะมี error เกิดขึ้น เนื่องจากมี repository อยู่บน server แล้ว แต่ก็ให้ทำการcommit ข้อมูลไปก่อน เพราะในกรณีนี้ error เหล่านั้นไม่ส่งผลกระทบโดยร้ายแรงต่อการทำงาน

- ให้กลับไปที่ browser และกด refresh 1 ครั้ง จะเห็นว่าหน้า repository ที่เราเพิ่งสร้าง จะเปลี่ยนไป



รูปที่ 1.15 หน้าจอ repository ที่เปลี่ยนไปหลังจากเพิ่มไฟล์ README.md

ผลการทดลอง

เมื่อตั้งค่าต่าง ๆ เสร็จแล้ว ข้อมูลของหน้า repository จะเปลี่ยนไปตามที่เราตั้งค่าไว้

The screenshot shows a GitHub repository page for 'CL64-01'. At the top, there's a dropdown menu 'main' and a link 'CL64-01 / README.md'. On the right, there are buttons for 'Go to file' and '...'. Below this, a commit card for 'siripaksorn' is shown with the message 'first commit'. It includes a timestamp 'Latest commit 1824621 4 minutes ago' and a 'History' button. Underneath, it says '1 contributor'. At the bottom of the commit card, there are links for 'Raw', 'Blame', and other file operations. The main content area displays the file 'README.md' with the text '# CL64-01'.

1.5 การแก้ไขงานและบันทึกการเปลี่ยนแปลงบน local computer

ถึงตอนนี้ เนื้อหาในไฟล์ README.md บน server และ local computer จะเหมือนกันทุกประการ เนื่องจากเป็นการ clone มาและยังไม่ได้ทำการแก้ไขใดๆ อีกทั้งเรามั่นใจว่าไม่มีผู้ใช้คนอื่นๆ กำลังแก้ไขงานของเรานบน server (ซึ่งการแก้ไขงานร่วมกันบน server จะอยู่ในการทดลองต่อไป) เราสามารถแก้ไขและทำ revision ของเอกสารได้ตามต้องการ โดยการเปลี่ยนแปลงต่างๆ จะเกิดขึ้นบนเครื่อง local computer เท่านั้น

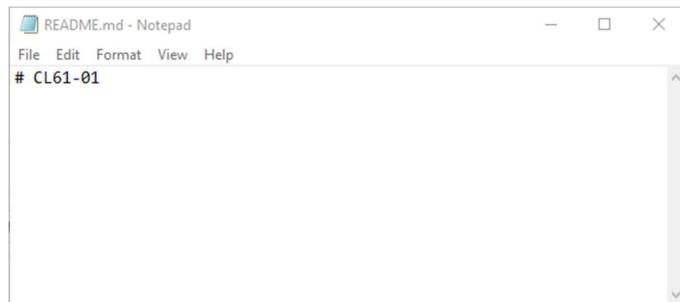
1.5.1 ทดลองแก้ไขไฟล์ README.md

โดยส่วนใหญ่ในการเขียนโปรแกรม มักจะมีระบบ IDE⁵ หรือ Integrated development environment ที่ช่วยให้เราเขียนและแก้ไขไฟล์ได้สะดวก เช่น Visual Studio, Eclipse, PyCharm เป็นต้น แต่ในการทดลองนี้ เราจะใช้โปรแกรมแก้ไขเอกสารอย่างง่ายๆ นั่นคือโปรแกรม Notepad.exe

- ให้พิมพ์คำสั่งต่อไปนี้ลงใน git bash

```
$ notepad README.md
```

ระบบจะเปิด text editor ที่มากับระบบปฏิบัติการ Windows ดังรูปที่ 16

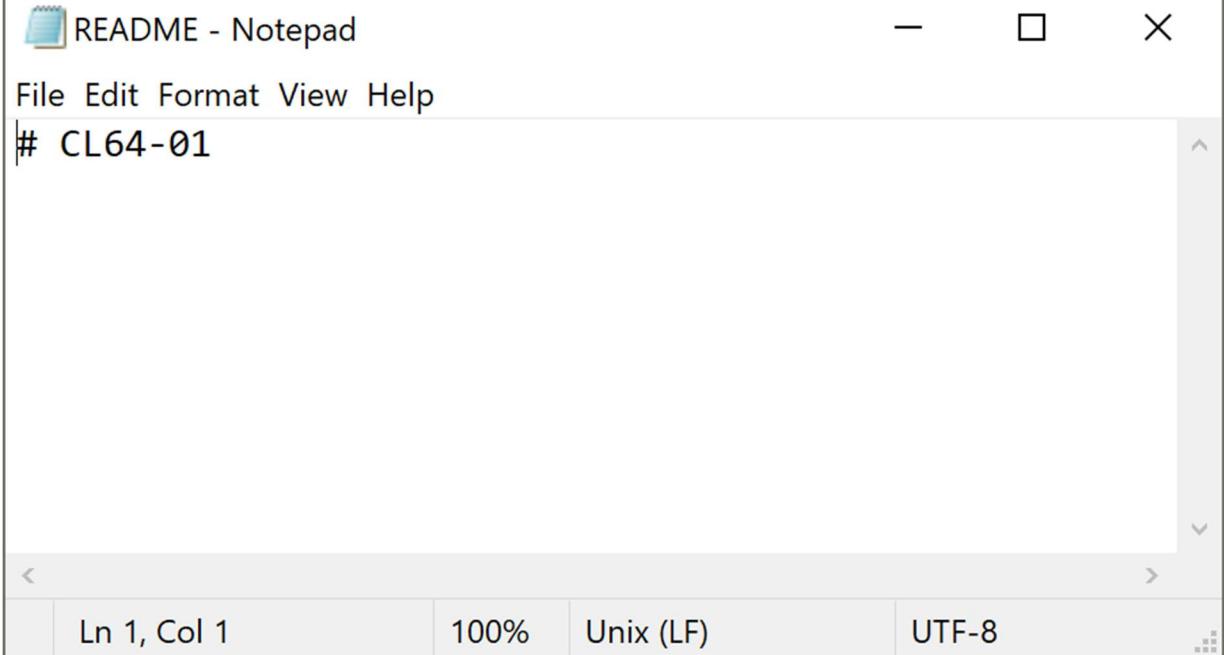


รูปที่ 1.16 การใช้โปรแกรม notepad.exe แก้ไขไฟล์ README.md

⁵ "Integrated development environment - Wikipedia." Accessed August 11, 2017. https://en.wikipedia.org/wiki/Integrated_development_environment.

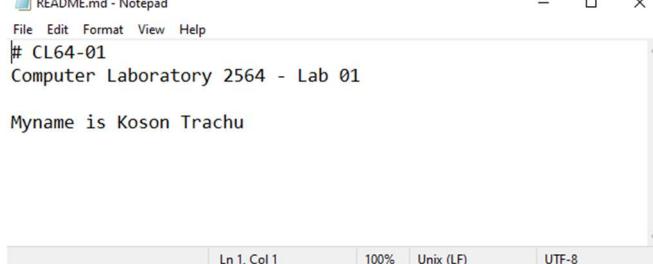
ผลการทดลอง

ข้อมูลใน README.md จะแสดงตามที่เราตั้งค่าไว้



The screenshot shows a Windows Notepad window titled "README - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main content area contains the text "# CL64-01". At the bottom, the status bar displays "Ln 1, Col 1", "100%", "Unix (LF)", and "UTF-8".

- แก้ไขไฟล์ README.md ใน notepad โดยเพิ่มข้อความลงไปดังตัวอย่าง (ให้นักศึกษาใส่ชื่อตนเอง)



The screenshot shows a Windows Notepad window titled "README.md - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main content area contains the text "# CL64-01", "Computer Laboratory 2564 - Lab 01", and "Myname is Koson Trachu". At the bottom, the status bar displays "Ln 1, Col 1", "100%", "Unix (LF)", and "UTF-8".

รูปที่ 1.17 แก้ไขไฟล์ README.md โดยเพิ่มบรรทัดต่อท้ายเข้าไป

ผลการทดลอง

เป็นการเพิ่มข้อมูลลงไปใน README.md

The screenshot shows a Windows Notepad window titled '*README - Notepad'. The content of the file is as follows:

```

# CL64-01
Computer Laboratory 2564 - Lab 09

My name is Siripaksorn Rattanakhot

```

The Notepad window includes standard menu options (File, Edit, Format, View, Help) and status bar information (Ln 1, Col 1, 100%, Unix (LF), UTF-8).

- บันทึกและปิดโปรแกรม notepad.exe
- ตรวจสอบการเปลี่ยนแปลงใน git bash โดยพิมพ์คำสั่ง git status และสังเกตผลที่ได้จากการรันคำสั่ง

\$ git status

The terminal window shows the following output of the git status command:

```

MINGW64:/f/GitHubRepos/Computerlab-2564/CL2564
* [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ notepad readme.md

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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:   README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    CL64-01/

no changes added to commit (use "git add" and/or "git commit -a")

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ |

```

รูปที่ 1.18 การตรวจสอบสถานะของ git

จะพบว่า git ได้ทำการติดตามการเปลี่ยนแปลง (tracking) ของไฟล์ต่างๆ ใน repository ของเราอยู่เสมอ ถึงแม้ว่าจะเป็น local computer ก็ตาม (ไม่นับไฟล์ใน .gitignore)

ผลการทดลอง

เป็นการตรวจสอบสถานะปัจจุบันของ git

```
MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564/CL64-01
```

```
Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git status
on branch main
Your branch is up to date with 'origin/main'.

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:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$
```

1.5.2 บันทึกการเปลี่ยนแปลงบน local computer

ถึงตรงนี้ ถ้าเราต้องการจะแก้ไขต่อ ก็สามารถทำได้ แต่การเปลี่ยนแปลงต่างๆ จะไม่สามารถถูกติดตามโดย git ถ้าต้องการให้ git บันทึก (หรือบัน) การเปลี่ยนแปลงเป็นรุ่นหนึ่งๆ ของ source code สามารถทำได้โดยการ commit การเปลี่ยนแปลงลงใน local repository ซึ่งการใช้งานเบื้องต้นจะมี 2 คำสั่งคือ git add และ git commit

- เพิ่มไฟล์ที่เปลี่ยนแปลง เข้าสู่รายการ commit โดยใช้คำสั่งต่อไปนี้

```
$ git add README.md
```

ตรวจสอบการเปลี่ยนแปลงใน git bash โดยพิมพ์คำสั่ง git status และล้างเกตเผลที่ได้จากการรันคำสั่ง

ผลการทดลอง

เพิ่มข้อมูล README.md ลงใน git และตรวจสอบสถานะของ git

The screenshot shows a terminal window titled 'MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)'. It displays the following command and its output:

```
Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git status
on branch main
Your branch is up to date with 'origin/main'.

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:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git add README.md
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$
```

```
$ git status
```

```

MINGW64:/f/GitHubRepos/Computerlab-2564/CL2564
no changes added to commit (use "git add" and/or "git commit -a")
koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ git add README.md
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory
koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    CL64-01/

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ |

```

รูปที่ 1.19 การตรวจสอบสถานะของ git / ผลจากการทำคำสั่ง git add

หมายเหตุ หากมีการแก้ไขหลายไฟล์ เราอาจใช้คำสั่ง git add --all แทนการใช้ชื่อไฟล์ได้

- Commit ไฟล์ที่เปลี่ยนแปลง เข้าสู่ repository โดยใช้คำสั่งต่อไปนี้

\$ git commit -m “Edited by Koson”

ตามด้วยการตรวจสอบสถานะของ repository

\$ git status

จะได้ผลดังนี้

```

MINGW64:/f/GitHubRepos/Computerlab-2564/CL2564
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    CL64-01/

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ git commit -m "Edit by Koson"
[main 09c839b] Edit by Koson
 1 file changed, 3 insertions(+)

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    CL64-01/

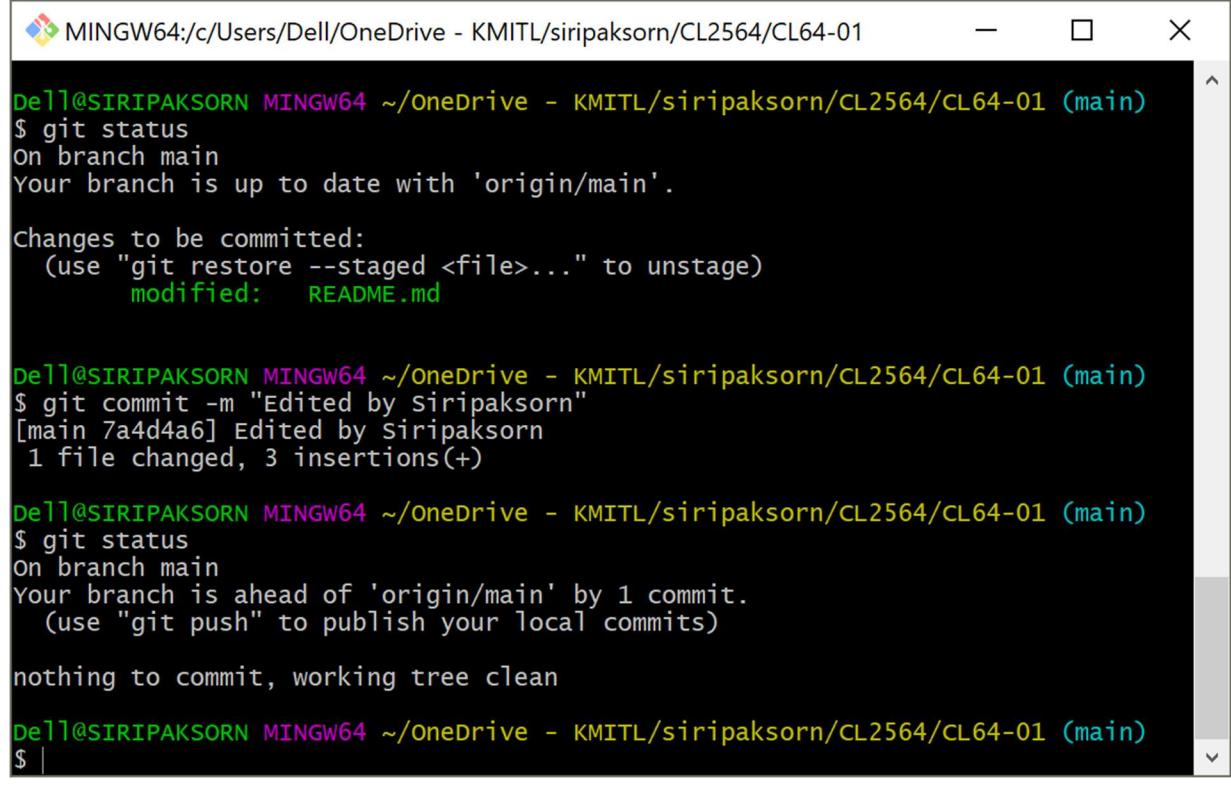
nothing added to commit but untracked files present (use "git add" to track)
koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ |

```

รูปที่ 1.20 ผลจากการทำ git commit

ผลการทดลอง

ใช้คำสั่ง git commit -m เพื่อเพิ่มไฟล์ที่ถูกเปลี่ยนแปลงไปใน repository



```
Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git status
on branch main
Your branch is up to date with 'origin/main'.

changes to be committed:
(use "git restore --staged <file>..." to unstage)
  modified: README.md

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git commit -m "Edited by siripaksorn"
[main 7a4d4a6] Edited by siripaksorn
 1 file changed, 3 insertions(+)

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git status
on branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ |
```

หมายเหตุ รูปแบบของการ commit ประกอบด้วย คำสั่ง git commit -m “THIS IS A COMMIT MESSAGE” โดยที่ commit message ควรเป็นข้อความที่สื่อความหมาย มีความยาวไม่มากนัก แต่ไม่สั้นจนเกินไป **ควรหลีกเลี่ยงคำที่ไม่สื่อความหมาย เช่น “1”, “2” หรือ “a” ถึงแม้ว่า git จะอนุญาตให้ใช้ก็ตาม** เนื่องจากเมื่อพัฒนาไปหลายๆ รุ่น จะไม่สามารถทำความเข้าใจเหตุผลที่แก้ไข source code นั้น ๆ ได้ และในการเปลี่ยนแปลงแต่ละครั้ง git จะนำ commit message นี้ไปร่วมกับการเปลี่ยนแปลงเสมอ

1.6 การซิงค์การเปลี่ยนแปลงระหว่าง local computer และ server

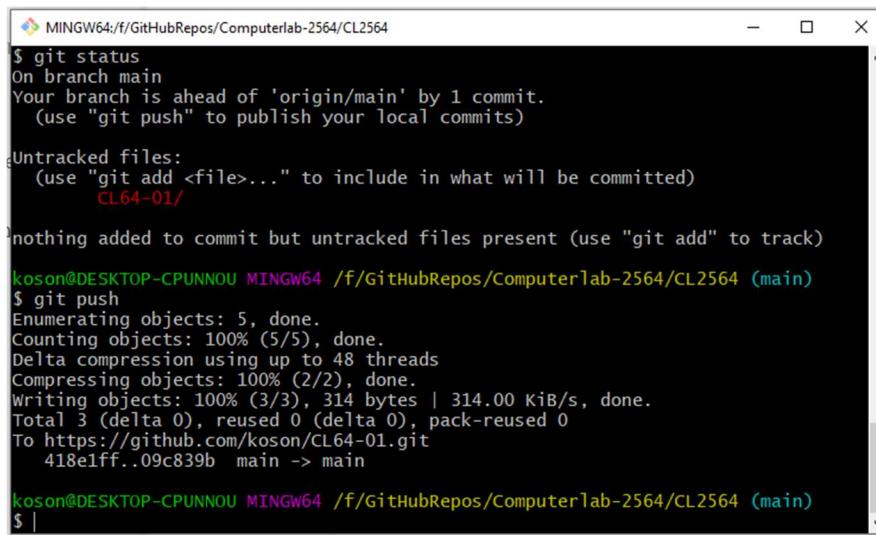
หลังจากที่เราได้ทำการ clone repository มาที่ local computer และ การแก้ไขงานทั้งหมด สามารถทำได้บน local computer ได้โดยไม่ต้องเชื่อมต่อกับ server แต่ในบางครั้งที่มีการทำงานร่วมกันเป็นทีม จะต้องปรับปรุง source code ให้เป็นปัจจุบันอยู่เสมอ จะต้องมีการ sync กับ server ได้แก่การ upload การเปลี่ยนแปลงขึ้นสู่ server (เรียกว่าการ push) และการ download การเปลี่ยนแปลงมาจาก server (เรียกว่าการ pull)

1.6.1 การ push ขึ้นสู่ server

โดยทั่วไป การที่จะ push ขึ้นสู่ server เราจะใช้คำสั่ง 3 คำสั่งควบคู่กันคือ (1) git add --all, (2) git commit -m "Commit message" และ (3) git push แต่ในการทดลองที่ผ่านมา เราทำใน (1) และ (2) ไปแล้ว ดังนั้น ให้พิมพ์คำสั่งต่อไปนี้ เพื่อ push repository ขึ้น server

```
$ git push
```

จะได้ผลลัพธ์ค้ายิ่งๆ อย่างในรูปที่ 20



```
MINGW64:/f/GitHubRepos/Computerlab-2564/CL2564
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    CL64-01/

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

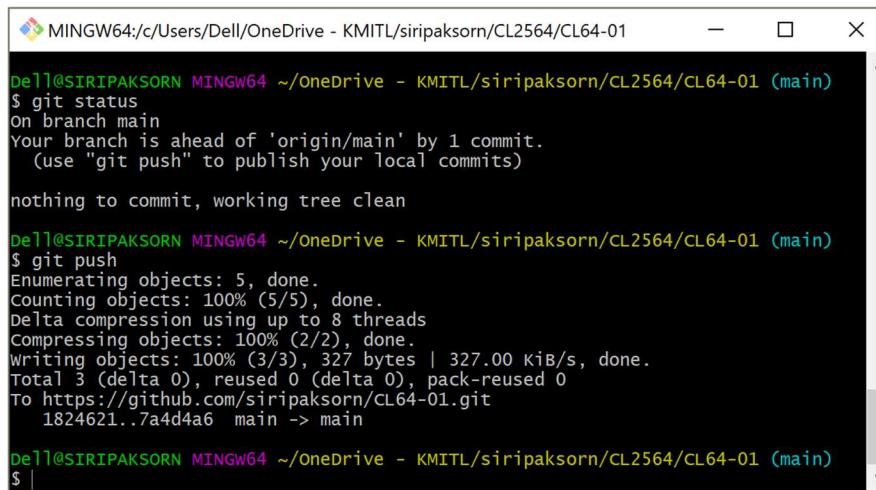
koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 48 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 314 bytes | 314.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/koson/CL64-01.git
  418e1ff..09c839b main -> main

koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ |
```

รูปที่ 1.21 ผลจากการทำคำสั่ง git push

ผลการทดลอง

ใช้คำสั่ง git push เพื่อนำข้อมูลที่ถูกเปลี่ยนแปลงขึ้น server



```
MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564/CL64-01
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 327 bytes | 327.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/siripaksorn/CL64-01.git
  1824621..7a4d4a6 main -> main

$ |
```

เมื่อเราทำการ push repository ขึ้นสู่ server แล้ว ก็ต้องทดสอบผลจากการ push โดยการไป refresh web browser ที่สร้าง repository ไว้ ดังรูปที่ 15

The screenshot shows a GitHub repository page for 'CL64-01'. The repository was last updated 2 minutes ago by 'koson'. It has 1 contributor. The README.md file contains the following content:

```
CL64-01
Computer Laboratory 2564 - Lab 01
My name is Koson Trachu
```

รูปที่ 1.22 การเปลี่ยนแปลงที่เกิดขึ้นบน server

ผลการทดลอง

ข้อมูลของเว็บมีการเปลี่ยนแปลงหลังจากนำข้อมูลขึ้น server

The screenshot shows a GitHub repository page for 'CL64-01'. The repository was last updated 4 minutes ago by 'Siripaksorn'. It has 1 contributor. The README.md file contains the following content:

```
CL64-01
Computer Laboratory 2564 - Lab 09
My name is Siripaksorn Rattanakhot
```

1.6.2 การ pull มาจาก server

- การเปลี่ยนแปลงใดๆ ที่เกิดขึ้นบน local computer จะถูกส่งขึ้นมาเก็บด้วยคำสั่ง git push และถ้ามีการแก้ไขไฟล์ใด ๆ เกิดขึ้นบน server เราจะสามารถที่จะดึงกลับไปทำงานที่ local computer ได้เช่นกัน
- ให้แก้ไขไฟล์ README.md โดยการคลิกที่ชื่อไฟล์ และปุ่มปากกาบริเวณด้านขวาเมือ

CL64-01

Computer Laboratory 2564 - Lab 01

My name is Koson Trachu

รูปที่ 1.23 เข้าสู่โหมดการแก้ไขไฟล์ด้วย Github Text Editor

ผลการทดลอง

แสดงหน้าแก้ไขไฟล์ของ Github

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

CL64-01 / README.md in main Cancel changes

<> Edit file <> Preview Spaces 2 Soft wrap

```
1 # CL64-01
2 Computer Laboratory 2564 - Lab 01
3
4 My name is Siripaksorn Rattanakhot
```

- เพิ่มข้อความที่บรรยายล่าสุดดังตัวอย่าง

```

1 # CL64-01
2 Computer Laboratory 2564 - Lab 01
3
4 My name is Koson Trachu
5 Edited by Github Text Editor.

```

รูปที่ 1.24 เพิ่มข้อความบางอย่างใน Github Text Editor

ผลการทดลอง

เปลี่ยนแปลงข้อมูลตามภาพ

siripaksorn / CL64-01 Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

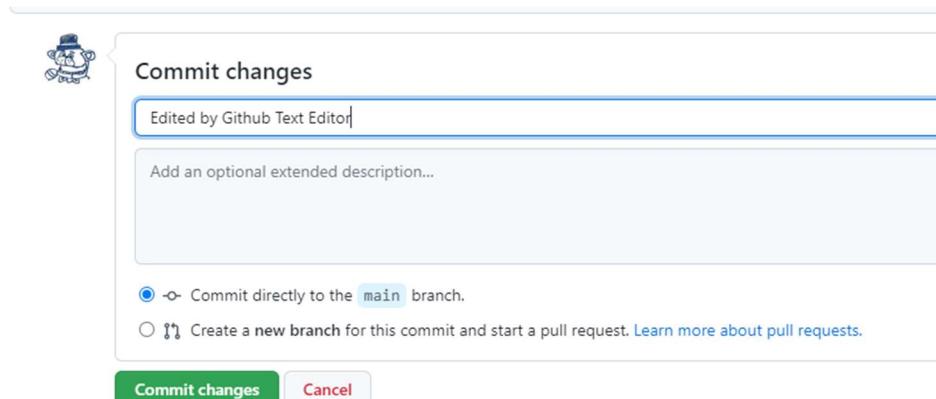
CL64-01 / README.md in main Cancel changes

```

1 # CL64-01
2 Computer Laboratory 2564 - Lab 01
3
4 My name is Siripaksorn Rattanakhot
5 Edited by Github Text Editor.

```

- เพิ่มข้อความในช่อง Commit changes และกดปุ่ม Commit changes สีเขียว



รูปที่ 1.25 เพิ่มข้อความ Commit changes

ผลการทดลอง

เมื่อเปลี่ยนแปลงข้อมูลแล้ว ข้อมูลในหน้าเว็บจะมีการเปลี่ยนแปลงตามที่เราเปลี่ยน

The screenshot shows a GitHub repository page for 'siripaksorn / CL64-01'. The repository is public and has 1 star, 0 forks, and 0 issues. The main branch is 'main'. The README.md file contains the following content:

```
CL64-01
Computer Laboratory 2564 - Lab 09
My name is Siripaksorn Rattanakhot Edited by Github Text Editor.
```

- กลับมาที่ git bash พิมพ์คำสั่ง git status สังเกตุผลการทำงาน

```
$ git status
```

- ที่ git bash พิมพ์คำสั่ง git pull

```
$ git pull
```

```
MINGW64:/f/GitHubRepos/Computerlab-2564/CL2564
nothing added to commit but untracked files present (use "git add" to track)
koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 713 bytes | 12.00 KiB/s, done.
From https://github.com/koson/CL64-01
  09c839b..ee99611  main      -> origin/main
Updating 09c839b..ee99611
Fast-forward
 README.md | 3 +++
 1 file changed, 2 insertions(+), 1 deletion(-)
koson@DESKTOP-CPUNNOU MINGW64 /f/GitHubRepos/Computerlab-2564/CL2564 (main)
$ |
```

รูปที่ 1.26 การใช้คำสั่ง git pull

ผลการทดลอง

เป็นการดึงข้อมูลจาก server ของเว็บ repository มาอยู่ในเครื่อง

```
MINGW64:/c/Users/Dell/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git status
on branch main
Your branch is up to date with 'origin/main'.

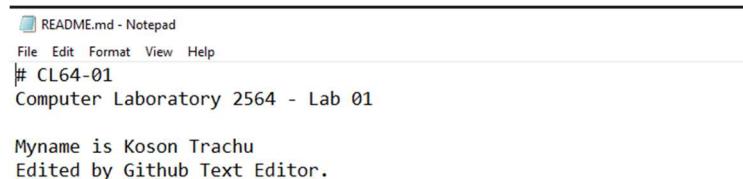
nothing to commit, working tree clean

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 731 bytes | 73.00 KiB/s, done.
From https://github.com/siripaksorn/CL64-01
  7a4d4a6..72e930b main      -> origin/main
Updating 7a4d4a6..72e930b
Fast-forward
 README.md | 3 ++
 1 file changed, 2 insertions(+), 1 deletion(-)

Dell@SIRIPAKSORN MINGW64 ~/OneDrive - KMITL/siripaksorn/CL2564/CL64-01 (main)
$ |
```

- ดูการเปลี่ยนแปลงในไฟล์ README.md

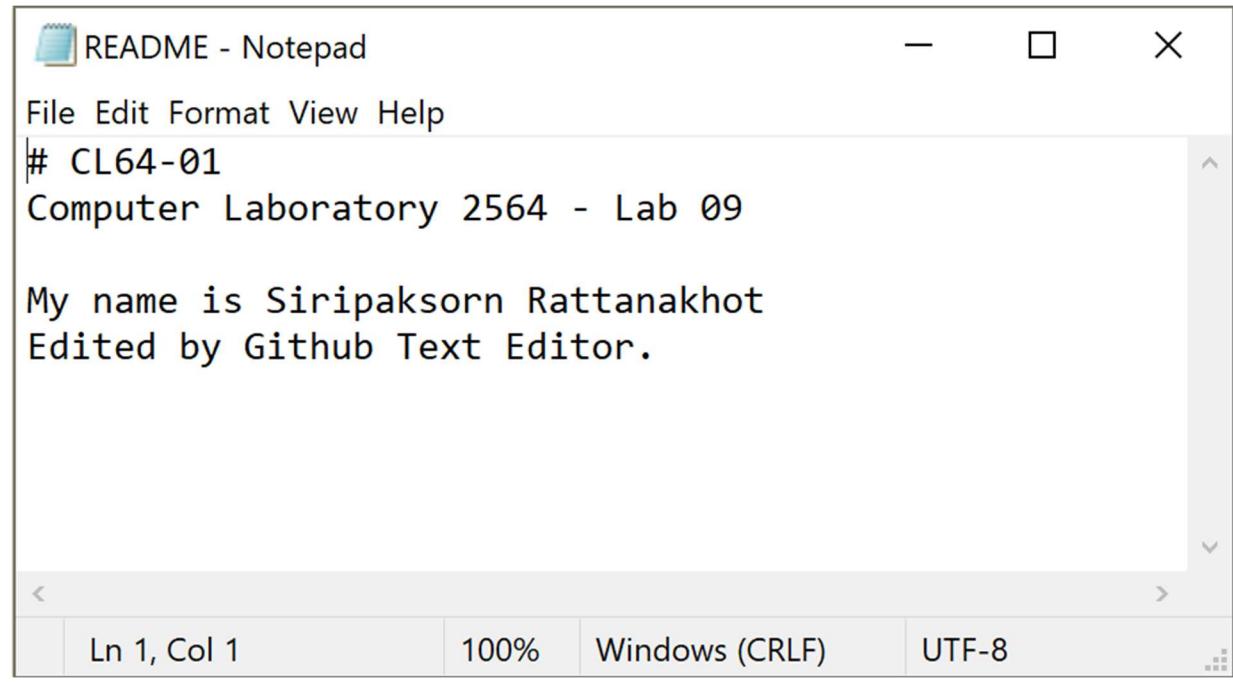
```
$ notepad.exe README.md
```



รูปที่ 1.27 การเปลี่ยนแปลงในไฟล์เอกสาร README.md

ผลการทดลอง

เปิดไฟล์ README.md จะเห็นได้ว่าข้อมูลมีการเปลี่ยนแปลงตามที่เราเปลี่ยนใน Github



The screenshot shows a Windows Notepad window titled "README - Notepad". The window contains the following text:

```
# CL64-01
Computer Laboratory 2564 - Lab 09

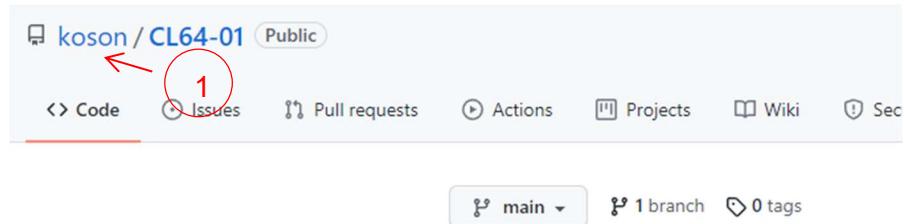
My name is Siripaksorn Rattanakhot
Edited by Github Text Editor.
```

The Notepad interface includes standard menu options (File, Edit, Format, View, Help), a toolbar, and status bar at the bottom indicating "Ln 1, Col 1", "100%", "Windows (CRLF)", and "UTF-8".

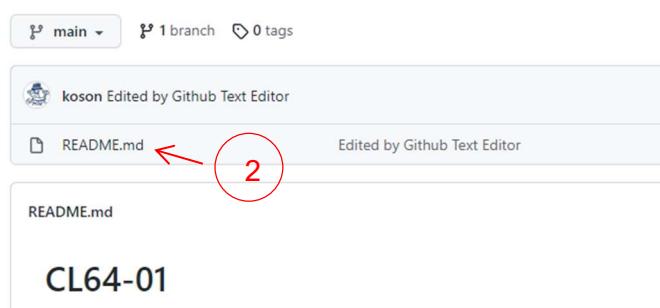
1.7 การตรวจสอบประวัติการเปลี่ยนแปลงของไฟล์

- กลับไปที่ web browser (1) คลิกที่ชื่อ repository, (2) คลิกที่ชื่อไฟล์ README.md (3) คลิกปุ่ม History

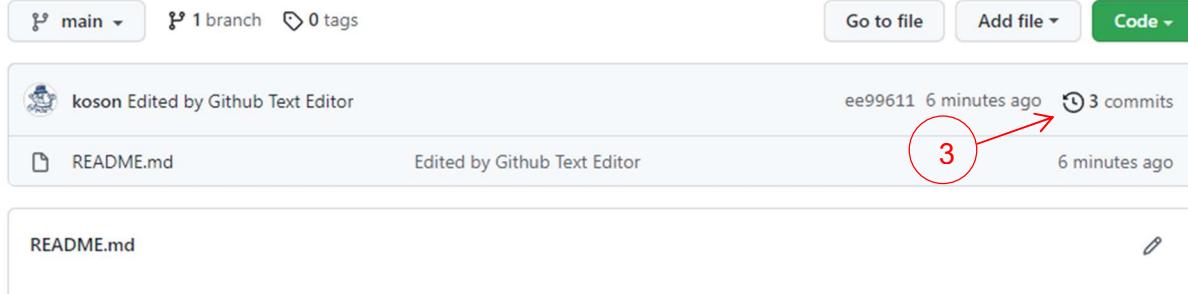
1.



2.



3.



รูปที่ 1.28 การเข้าถึงประวัติของไฟล์

เมื่อคลิกดูประวัติไฟล์ จะพบว่า ไม่ว่าเราจะแก้ไขไฟล์ที่ไหน แต่ Git จะติดตามและบันทึกการเปลี่ยนแปลงทุกครั้งที่เราทำการ commit

ผลการทดลอง

เมื่อ กดที่ CL64-01 จะแสดงข้อมูลและประวัติต่าง ๆ ที่เราได้ทำการเปลี่ยนแปลง

branch: main | 1 branch | 0 tags

Edited by Github Text Editor | 72e930b 6 minutes ago | 3 commits

README.md | Edited by Github Text Editor | 6 minutes ago

koson / CL64-01 Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main

Commits on Oct 7, 2021

- Edited by Github Text Editor | koson committed 7 minutes ago
- Edit by Koson | koson committed 12 minutes ago
- first commit | koson committed 20 minutes ago

[Newer](#) [Older](#)

ee99611 | 09c839b | 418e1ff

รูปที่ 1.29 รายการประวัติการแก้ไขไฟล์

ผลการทดลอง

เมื่อ กดดูประวัติ จะแสดงประวัติทั้งหมด โดยสามารถดูรายละเอียดการเปลี่ยนแปลงได้

siripaksorn / CL64-01 Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

History for CL64-01 / README.md

Commits on Oct 10, 2021

- Edited by Github Text Editor | siripaksorn committed 7 minutes ago
- Edited by Siripaksorn | siripaksorn committed 17 minutes ago
- first commit | siripaksorn committed 35 minutes ago

[Newer](#) [Older](#)

Verified | 72e930b | 7a4d4a6 | 1824621

ให้คลิกปุ่มที่มีเลขฐาน 16 กำกับ (เป็นชื่อรหัสกำกับการแก้ไข ที่มีพัฒนาจะใช้อังกฤษ) ตามลูกศรสีแดงในรูปที่ 29 เราจะเห็นประวัติการแก้ไขไฟล์ ดังรูปที่ 30

The screenshot shows a GitHub commit history for a repository named 'main'. A commit by 'koson' was made 8 minutes ago, with a green 'Verified' badge. The commit message is 'commit ee99611a4efdc2a1b7b2d4438a75809461ba2456'. It has 1 parent commit '09c839b'. The file 'README.md' was changed, showing 2 additions and 1 deletion. The unified diff view shows the following changes:

```
@@ -1,4 +1,5 @@
1 # CL64-01
2 Computer Laboratory 2564 - Lab 01
3
4 - Myname is Koson Trachu
5 + Myname is Koson Trachu
6 + Edited by Github Text Editor.
```

Below the diff, there are 0 comments on the commit 'ee99611'. A green button at the bottom right says 'Comment on this commit'.

รูปที่ 1.30 ประวัติการแก้ไขไฟล์

แบบฝึกหัด

- ให้นักศึกษาทดลองเพิ่มไฟล์ชื่อ student.txt ลงใน repository และเพิ่มรายชื่อเพื่อนใหม่ในห้อง โดยเพิ่มบน notepad จำนวนครึ่งหนึ่ง และทำบน github text editor จนครบ โดยให้เขียน commit message ด้วยว่าเพิ่มจากที่ได้
- ให้นักศึกษาทดลองแก้ไขไฟล์ README.md ตามตารางต่อไปนี้ และทำการยงานประวัติไฟล์มาส่ง

ลำดับที่	สถานที่แก้ไข	สิ่งที่กระทำ
1	Local (Notepad)	ลบเนื้อหาเดิมออกทั้งหมด
2	Server (Github Text Editor)	#include <stdio.h > main() { printf ("hello, world\n"); }
3	Local (Notepad)	เปลี่ยน printf("hello, world\n"); เป็น printf("hello, [ชื่อนักศึกษา]\n");
4	Server (Github Text Editor)	#include <stdio.h> int main () { char yourname[100]; printf("What is your name?\t"); scanf("%s",yourname); printf("hello, %s\n", yourname); }
5	Local (Notepad)	เพิ่ม printf("Goodbye\n"); ใต้ printf("hello, %s\n", yourname);

หมายเหตุ การทำแต่ละขั้น ให้ local และ server ซิงค์กันเสมอ (ต้อง push, pull, commit, add)

รายงานประวัติไฟล์

History for CL64-01 / README.md

Commits on Oct 11, 2021

- Add Goodbye by Notepad (siripaksorn committed 6 minutes ago)
- Change Code by Github Text Editor (siripaksorn committed 10 minutes ago)
- Add Name by Notepad (siripaksorn committed 14 minutes ago)
- Add Code by Github Text Editor (siripaksorn committed 19 minutes ago)
- Clear Notepad (siripaksorn committed 23 minutes ago)

Commits on Oct 10, 2021

- Edited by Github Text Editor (siripaksorn committed 2 hours ago)
- Edited by Siripaksorn (siripaksorn committed 2 hours ago)
- first commit (siripaksorn committed 2 hours ago)

Newer Older

1. ลบเนื้อหาเดิมในไฟล์ README.md ออกทั้งหมด

Clear Notepad

main

siripaksorn committed 31 minutes ago

Showing 1 changed file with 0 additions and 5 deletions.

Unified Split

5 README.md

```
@@ -1,5 +0,0 @@
- # CL64-01
- - Computer Laboratory 2564 - Lab 09
-
- - My name is Siripaksorn Rattanakhot
- - Edited by Github Text Editor.
```

2. เพิ่ม source code ลงในไฟล์ README.md ผ่าน Github Text Editor

Add Code by Github Text Editor

main

siripaksorn committed 28 minutes ago Verified

Showing 1 changed file with 5 additions and 0 deletions.

Unified Split

5 README.md

```
@@ -0,0 +1,5 @@
+ #include < stdio.h >
+ main( )
+ {
+     printf ("Hello, world\n");
+ }
```

3. เปลี่ยนแปลง code จาก hello เป็นชื่อ ด้วย Notepad

The screenshot shows a GitHub repository page for 'siripaksorn / CL64-01'. The 'Code' tab is selected. A commit message 'Add Name by Notepad' is shown, with the commit details: 'siripaksorn committed 24 minutes ago' and '1 parent b0a8a86 commit f04cd1eb97b1a70973443bf0f2bec2ef5b54339'. Below the commit message, it says 'Showing 1 changed file with 1 addition and 1 deletion.' The file 'README.md' is displayed with the following diff:

```
diff --git a/README.md b/README.md
--- a/README.md
+++ b/README.md
@@ -1,5 +1,5 @@
1 1 #include <stdio.h>
2 2 main()
3 3 {
4 -     printf ("Hello, world\n");
4 +     printf ("Hello, siripaksorn\n");
5 5 }
```

4. เปลี่ยนแปลงจากชื่อเป็น source code ผ่าน Github Text Editor

The screenshot shows a GitHub repository page for 'siripaksorn / CL64-01'. The 'Code' tab is selected. A commit message 'Change Code by Github Text Editor' is shown, with the commit details: 'siripaksorn committed 21 minutes ago Verified' and '1 parent f04cd1 commit 0a87b1c31173dafd9fc4a69e43fe13596f1903f2'. Below the commit message, it says 'Showing 1 changed file with 6 additions and 3 deletions.' The file 'README.md' is displayed with the following diff:

```
diff --git a/README.md b/README.md
--- a/README.md
+++ b/README.md
@@ -1,5 +1,8 @@
1 - #include <stdio.h>
2 - main()
1 + #include <stdio.h>
2 + int main()
3 -
4 -     printf ("Hello, siripaksorn\n");
4 +     char yourname[100];
5 +     printf("What is your name?\t");
6 +     scanf("%s",yourname);
7 +     printf("Hello, %s\n", yourname);
5 8 }
```

5. พิม source code Goodbye ด้วย Notepad

The screenshot shows a GitHub repository page for 'siripaksorn / CL64-01'. The 'Code' tab is selected. A commit message 'Add Goodbye by Notepad' is shown, with the commit details: 'siripaksorn committed 18 minutes ago' and '1 parent 0a87b1c commit 9e8a8f9e0e7651fb045dba2d2f6c63464f5fb36'. Below the commit message, it says 'Showing 1 changed file with 1 addition and 0 deletions.' The file 'README.md' is displayed with the following diff:

```
diff --git a/README.md b/README.md
--- a/README.md
+++ b/README.md
@@ -5,4 +5,5 @@ int main ()
5 5     printf("What is your name?\t");
6 6     scanf("%s",yourname);
7 7     printf("Hello, %s\n", yourname);
8 +     printf("Goodbye\n");
8 9 }
```

คำถาม

1. จากภาพที่ 29 ถ้านักศึกษาคลิกตามปุ่มที่มีเลขฐานสิบหกกำกับอยู่ ทุกปุ่ม จะได้ผลอย่างไรบ้าง ให้อธิบายสิ่งที่พบเห็น จะแสดงประวัติการแก้ไขของแต่ละ commit ว่ามีการแก้ไข เพิ่ม หรือลบอะไรบ้าง
2. ให้บอกประโยชน์ของ repository ตามที่นักศึกษาเข้าใจ ช่วยอธิบายความสะดวกในการทำ project เนื่องจากผู้ใช้สามารถย้อนดูประวัติการทำงาน หรือการแก้ไขโค้ดได้ ซึ่งสามารถ clone ໄວ่ได้ทั้งในเครื่องคอมพิวเตอร์และบน server นอกจากนี้ในการทำงานผู้ใช้คนอื่น ๆ สามารถมีส่วนร่วมในการแก้ไขโค้ดเราได้อีกด้วย
3. ให้บอกรแนวทางการนำ repository ไปใช้ในการเรียนหรือชีวิตประจำวันของนักศึกษา
 - เราสามารถนำ repository มาใช้ในการทำ project เพราะ สามารถเก็บสำรองข้อมูลหรือการแก้ไข Sorce Code ได้ โดยเรา สามารถดูรายละเอียดของการแก้ไขได้
 - เราสามารถนำ repository มาใช้ในการทำงานกลุ่มได้ เนื่องจากเราสามารถดูว่าใครเป็นคนแก้ไขไฟล์ได้
 - เราสามารถนำ repository มาใช้ในการส่งงานอาจารย์ได เนื่องจากจะมีการบันทึกเวลาที่ส่งหรือแก้ไขไฟล์อย่างละเอียด