Version Control System

What is Git ?

"**Distributed** version control system"

“**What is** version control ?”

คืออะไร

“**What do we** use version control for?”

ใช้ทำอะไร

“**Who does’t use** version control ?”

มีใครไม่ใช้บ้าง

Project ?

1,000 **Developers**

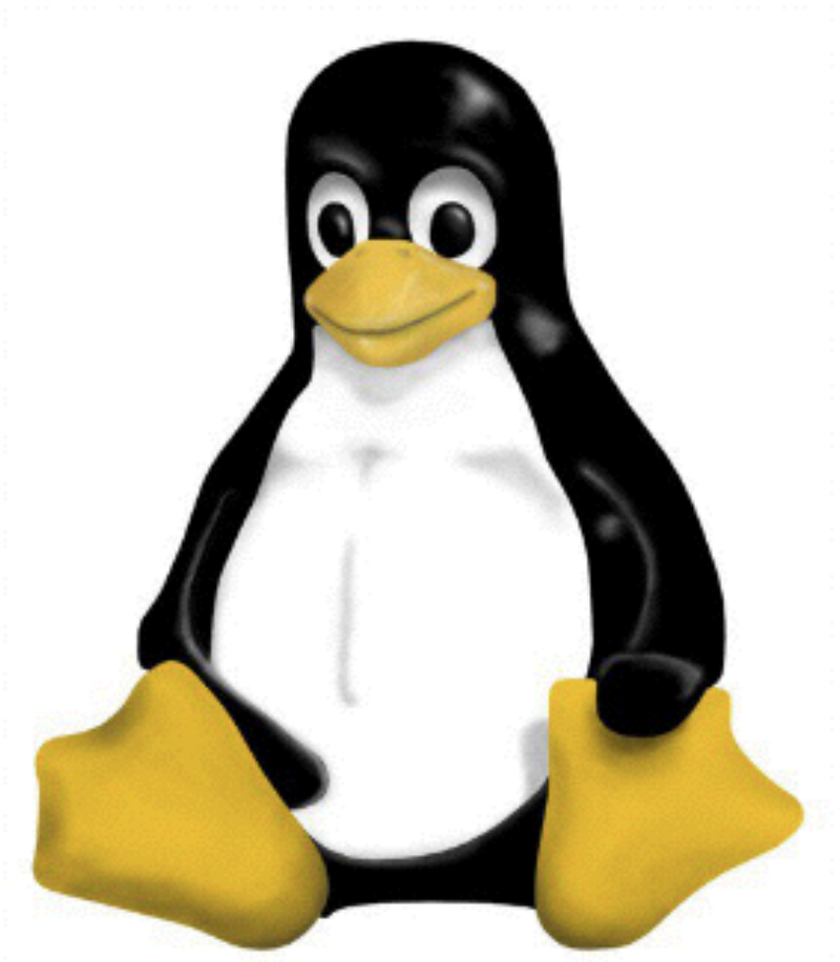
Working all across the **world**

Used on **millions** of computer

Used 90% of **Supercomputer**

Run on **mobile**

11 years without version control system



1991 – 2002

“How did they do it ?”

ทำอย่างไร

Working with file system

Main

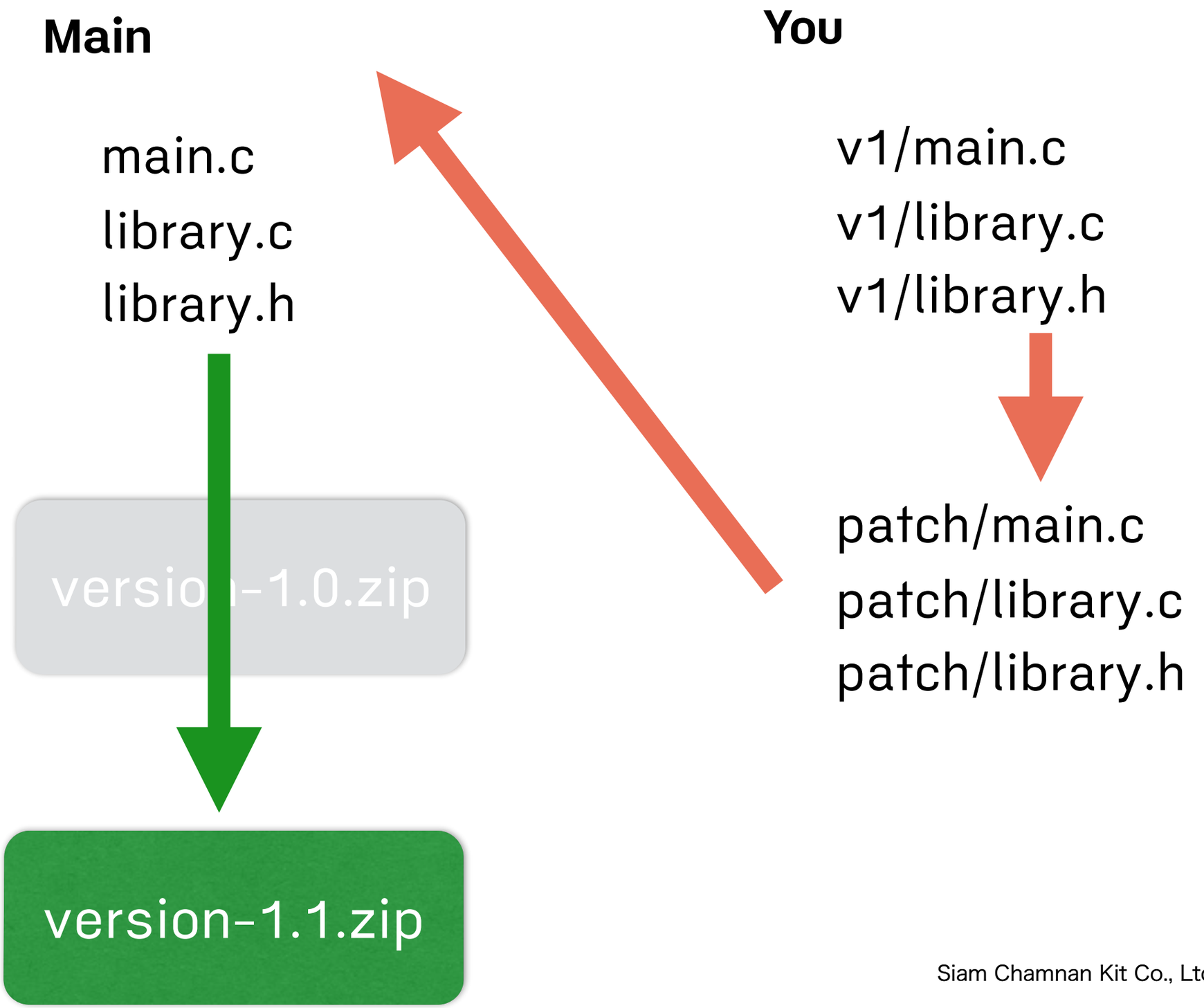
main.c
library.c
library.h

You

v1/main.c
v1/library.c
v1/library.h

patch/main.c
patch/library.c
patch/library.h

Working with patch



version-1.0.zip

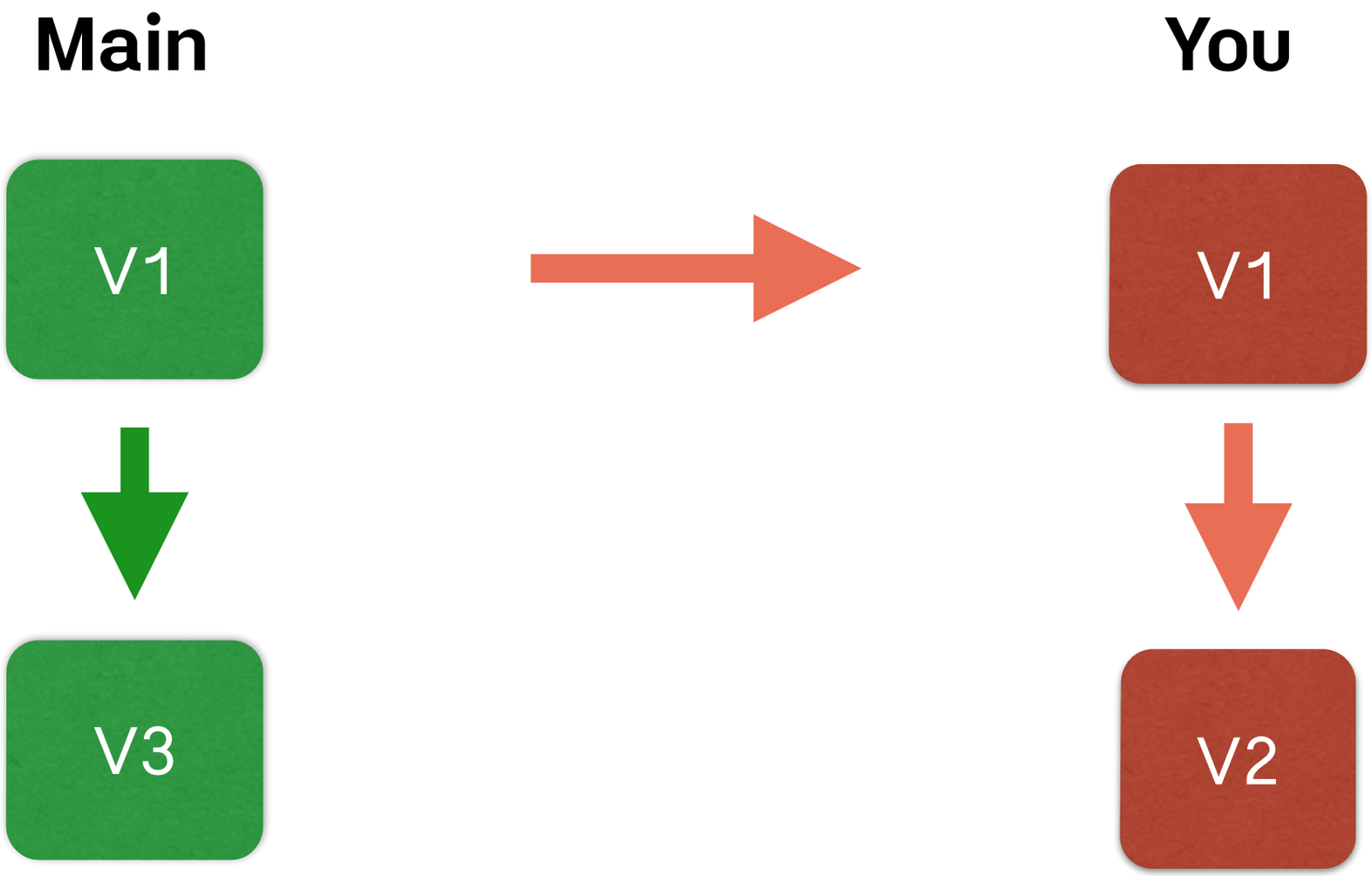
version-1.1.zip

version-1.2.zip

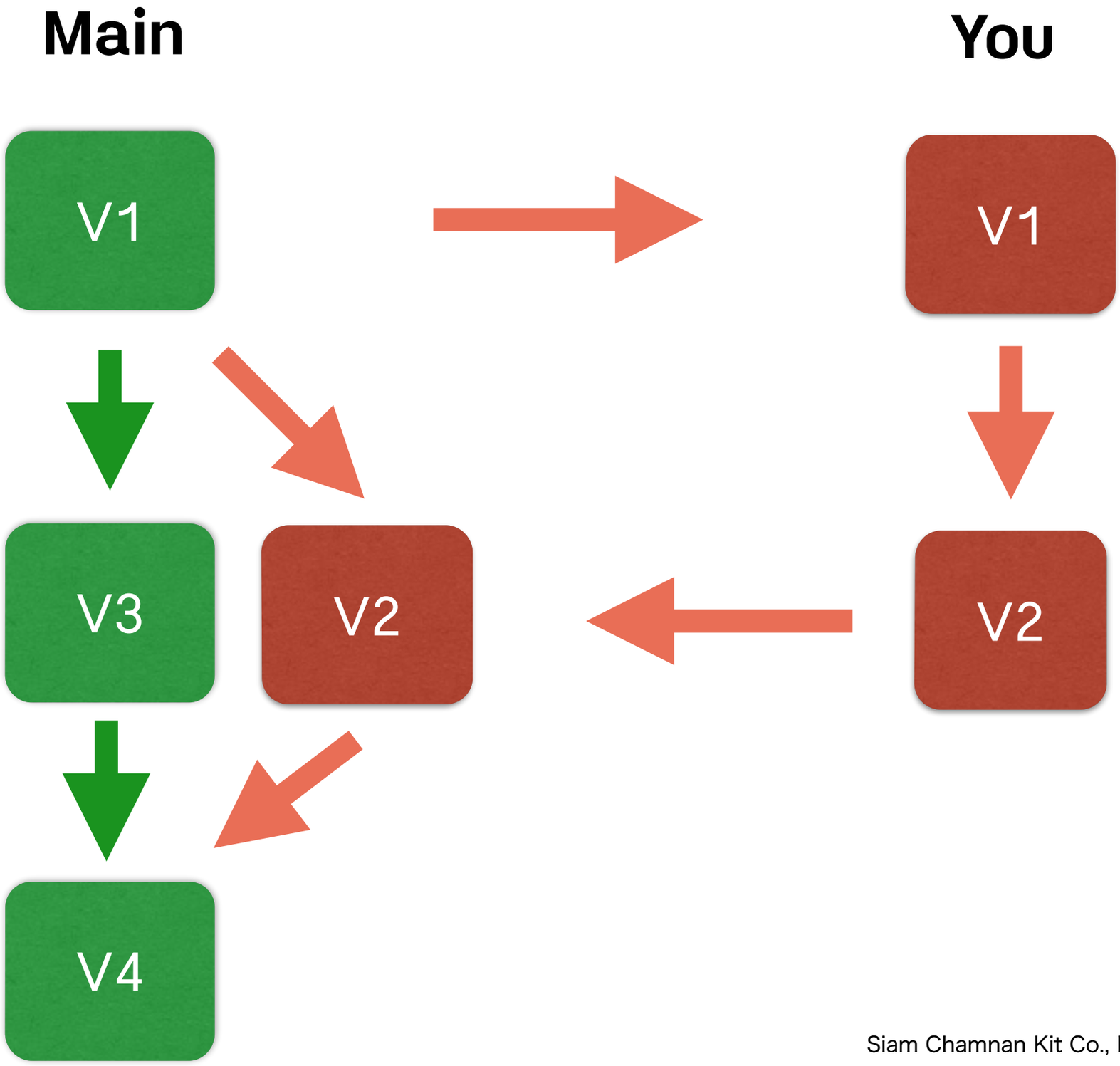
“In the real work”

ในการทำงานจริงๆ

Patching not working



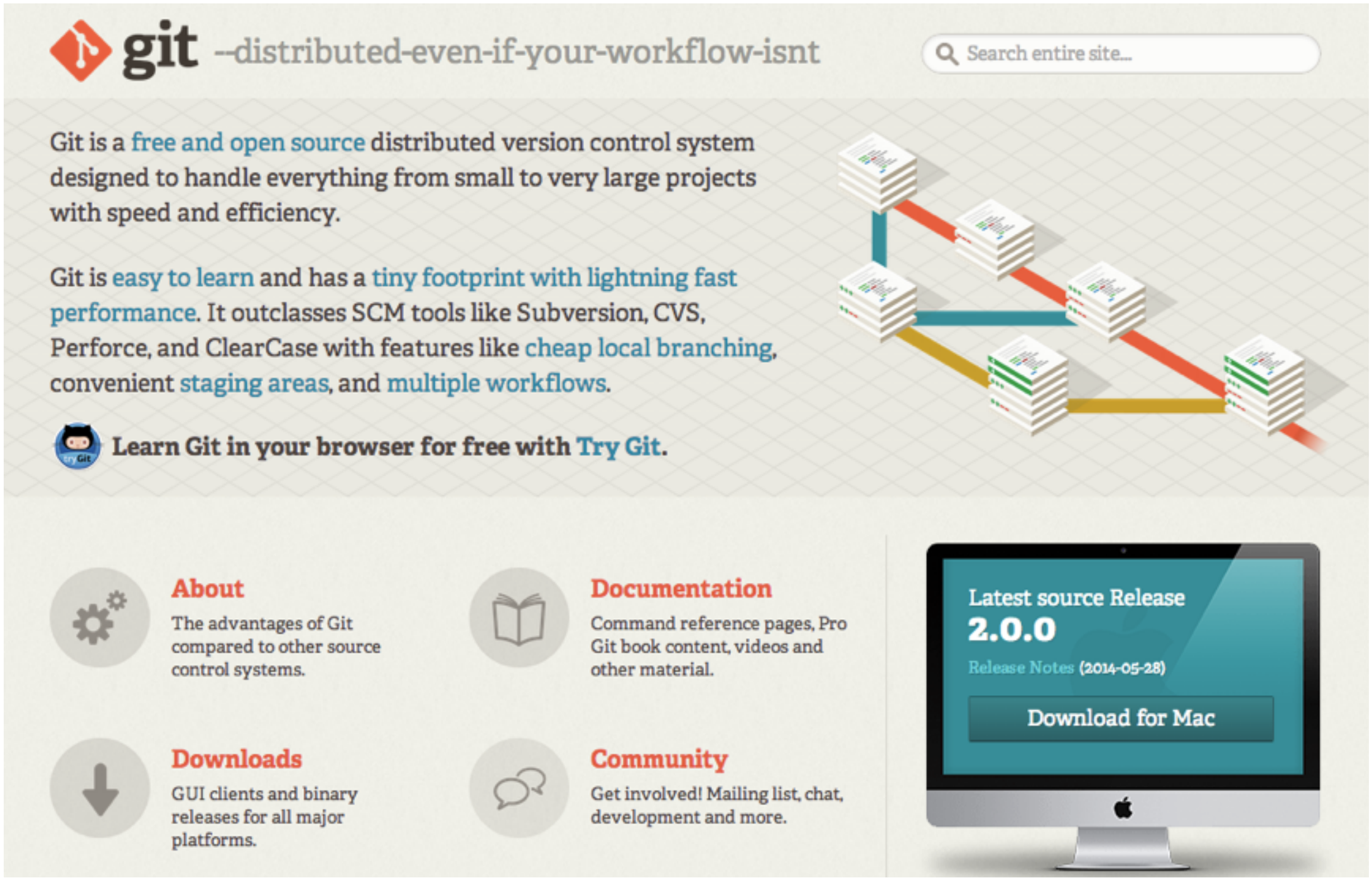
Patching not working



SPRINT3R

Siam Chamnan Kit Co., Ltd., and Odd-e (Thailand) Co., Ltd.

git-scm.com


git --distributed-even-if-your-workflow-isnt
Search entire site...
Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.
Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.
Learn Git in your browser for free with Try Git.
About
The advantages of Git compared to other source control systems.
Documentation
Command reference pages, Pro Git book content, videos and other material.
Downloads
GUI clients and binary releases for all major platforms.
Community
Get involved! Mailing list, chat, development and more.
Latest source Release
2.0.0
Release Notes (2014-05-28)
Download for Mac

First step

$git config —global user.name “UP1”

$git config —global user.email “UP1@XX”

Create repository

$git init

```
.git
├── HEAD
├── branches
├── config
├── description
├── hooks
│   ├── applypatch-msg.sample
│   ├── commit-msg.sample
│   ├── post-update.sample
│   ├── pre-applypatch.sample
│   ├── pre-commit.sample
│   ├── pre-push.sample
│   ├── pre-rebase.sample
│   ├── prepare-commit-msg.sample
│   └── update.sample
├── info
│   └── exclude
├── objects
│   ├── info
│   └── pack
└── refs
    ├── heads
    └── tags
```

เก็บทุกๆ อย่างของ git repository

Add some file to repository

```
$touch hello.txt

$git add .

$git commit -m “First commit”
```

After add some file

$tree -a

```
.
├── .git
│   ├── COMMIT_EDITMSG
│   ├── HEAD
│   ├── branches
│   ├── config
│   ├── description
│   ├── hooks
│   │   ├── applypatch-msg.sample
│   │   ├── commit-msg.sample
│   │   ├── post-update.sample
│   │   ├── pre-applypatch.sample
│   │   ├── pre-commit.sample
│   │   ├── pre-push.sample
│   │   ├── pre-rebase.sample
│   │   ├── prepare-commit-msg.sample
│   │   └── update.sample
│   ├── index
│   ├── info
│   │   └── exclude
│   ├── logs
│   │   ├── HEAD
│   │   └── refs
│   │       └── heads
│   │           └── master
│   ├── objects
│   │   ├── ac
│   │   │   └── 0cb0134d31cdbb53cfac4f4772ddffebc89ca0
│   │   ├── e6
│   │   │   └── 9de29bb2d1d6434b8b29ae775ad8c2e48c5391
│   │   ├── fc
│   │   │   └── b545d5746547a597811b7441ed8eba307be1ff
│   │   ├── info
│   │   └── pack
│   └── refs
│       ├── heads
│       │   └── master
│       └── tags
└── hello.txt
```

After add some file

$tree -a

```
.
├── .git
│   ├── COMMIT_EDITMSG
│   ├── HEAD
│   ├── branches
│   ├── config
│   ├── description
│   ├── hooks
│   │   ├── applypatch-msg.sample
│   │   ├── commit-msg.sample
│   │   ├── post-update.sample
│   │   ├── pre-applypatch.sample
│   │   ├── pre-commit.sample
│   │   ├── pre-push.sample
│   │   ├── pre-rebase.sample
│   │   ├── prepare-commit-msg.sample
│   │   └── update.sample
│   ├── index
│   ├── info
│   │   └── exclude
│   ├── logs
│   │   ├── HEAD
│   │   └── refs
│   │       └── heads
│   │           └── master
│   ├── objects
│   │   ├── ac
│   │   │   └── 0cb0134d31cdbb53cfac4f4772ddffebc89ca0
│   │   ├── e6
│   │   │   └── 9de29bb2d1d6434b8b29ae775ad8c2e48c5391
│   │   ├── fc
│   │   │   └── b545d5746547a597811b7441ed8eba307be1ff
│   │   ├── info
│   │   └── pack
│   └── refs
│       ├── heads
│       │   └── master
│       └── tags
└── hello.txt
```

Clone repository

$git clone git@github.com:github-sprint3r/training.git

```
Cloning into 'training'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (3/3), done.
Checking connectivity... done.
```

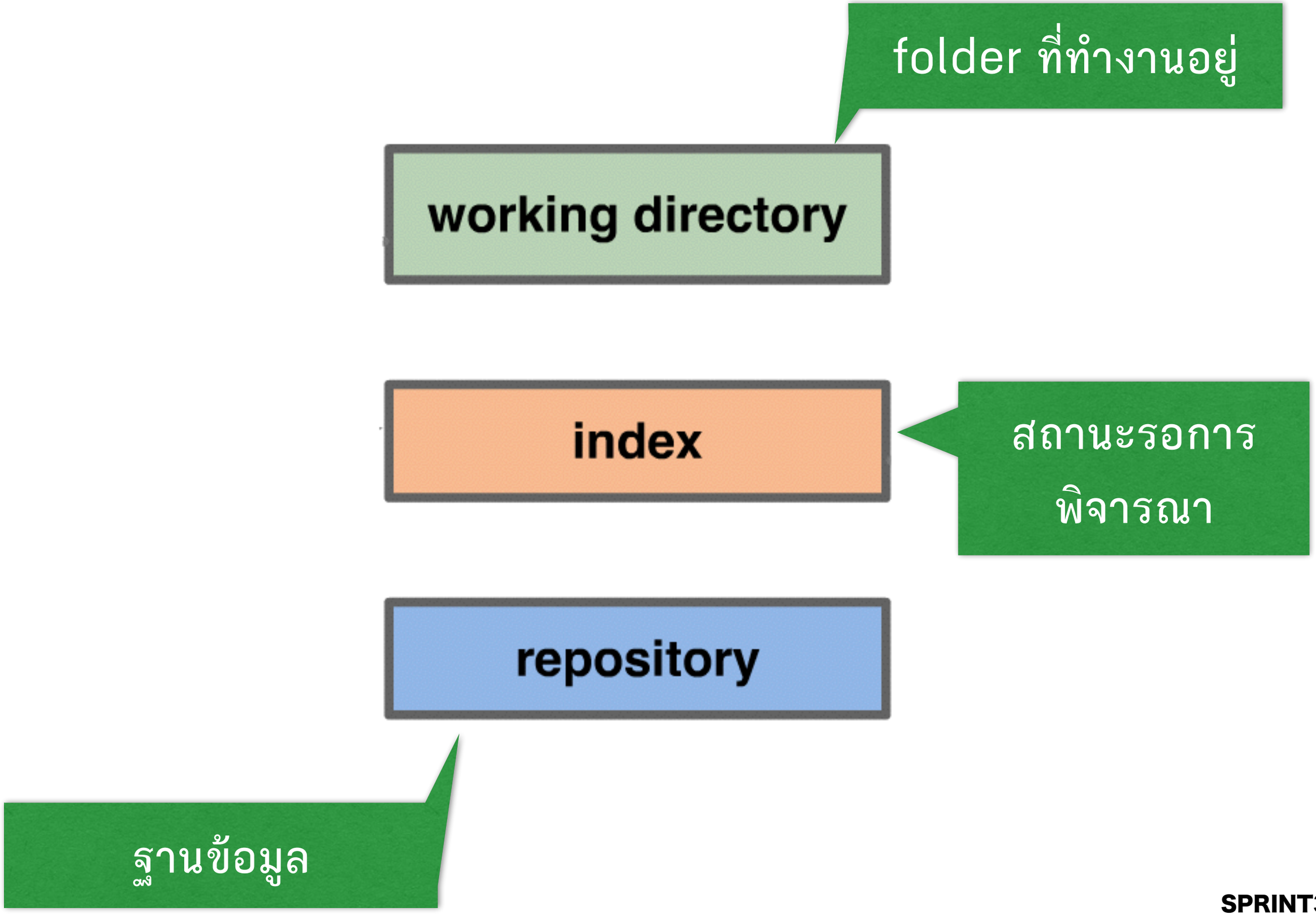
$cd training

$ls -la

```
total 8
drwxr-xr-x   4 spock  staff  136 Jun  5 13:52 .
drwxr-xr-x   9 spock  staff  306 Jun  5 13:53 ..
drwxr-xr-x  13 spock  staff  442 Jun  5 13:52 .git
-rw-r--r--   1 spock  staff   18 Jun  5 13:52 README.md
```

Basic workflow

1. Edit files
2. Stage the changes
3. Review your changes
4. Commit the changes


folder ที่ทำงานอยู่
working directory
index
สถานะรอการพิจารณา
repository
ฐานข้อมูล

1. Edit files

Edit file hello.txt

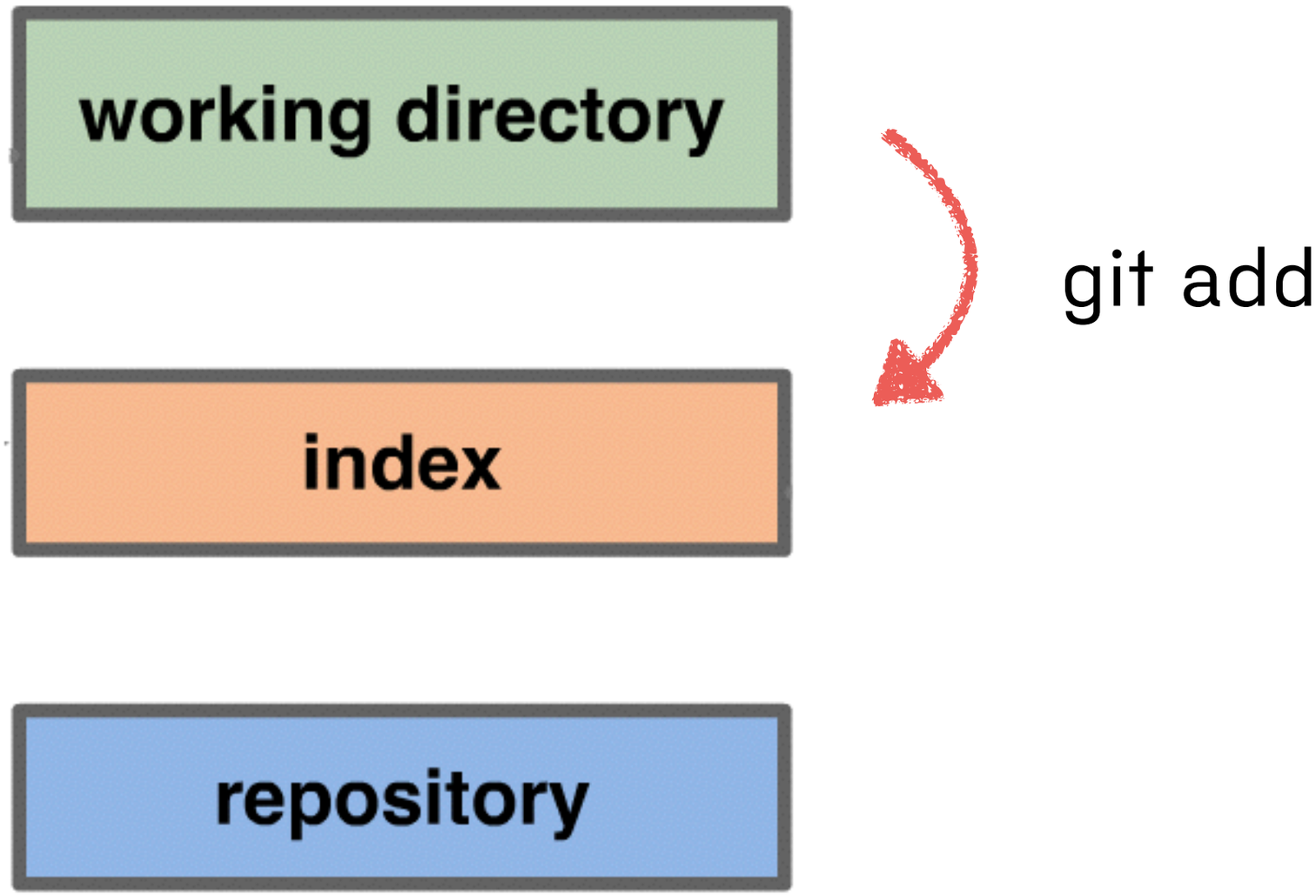
$git status

```
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   hello.txt

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

2. Stage the changes



2. Stage the changes

$git add hello.txt

$git status

สถานะรอการพิจารณา

```
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   hello.txt
```

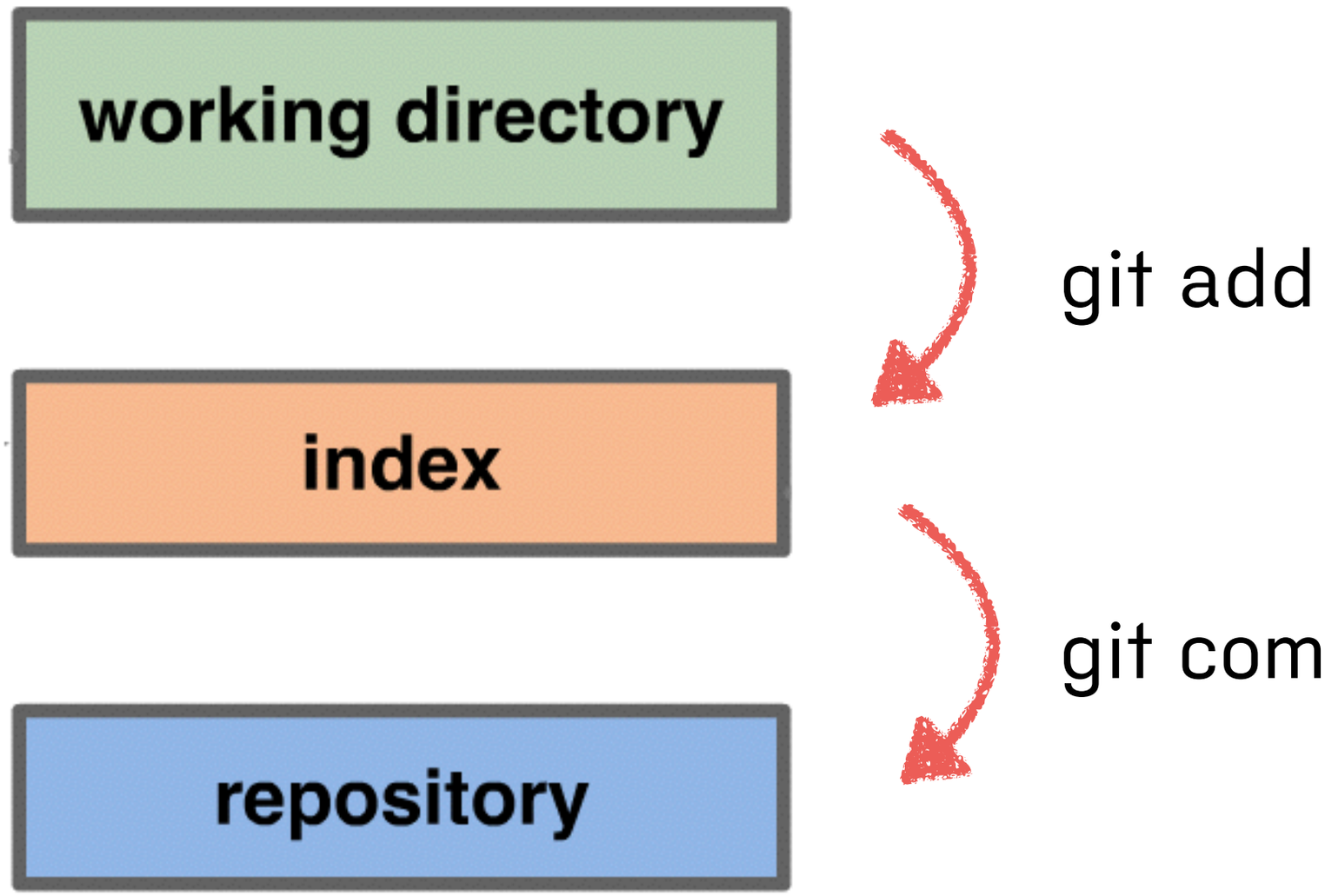
3. Review your changes

$git status

You have to stage a file **AFTER** you edit it

สำคัญมาก

4. Commit the changes



4. Commit the changes

$git commit -m “Update hello file”

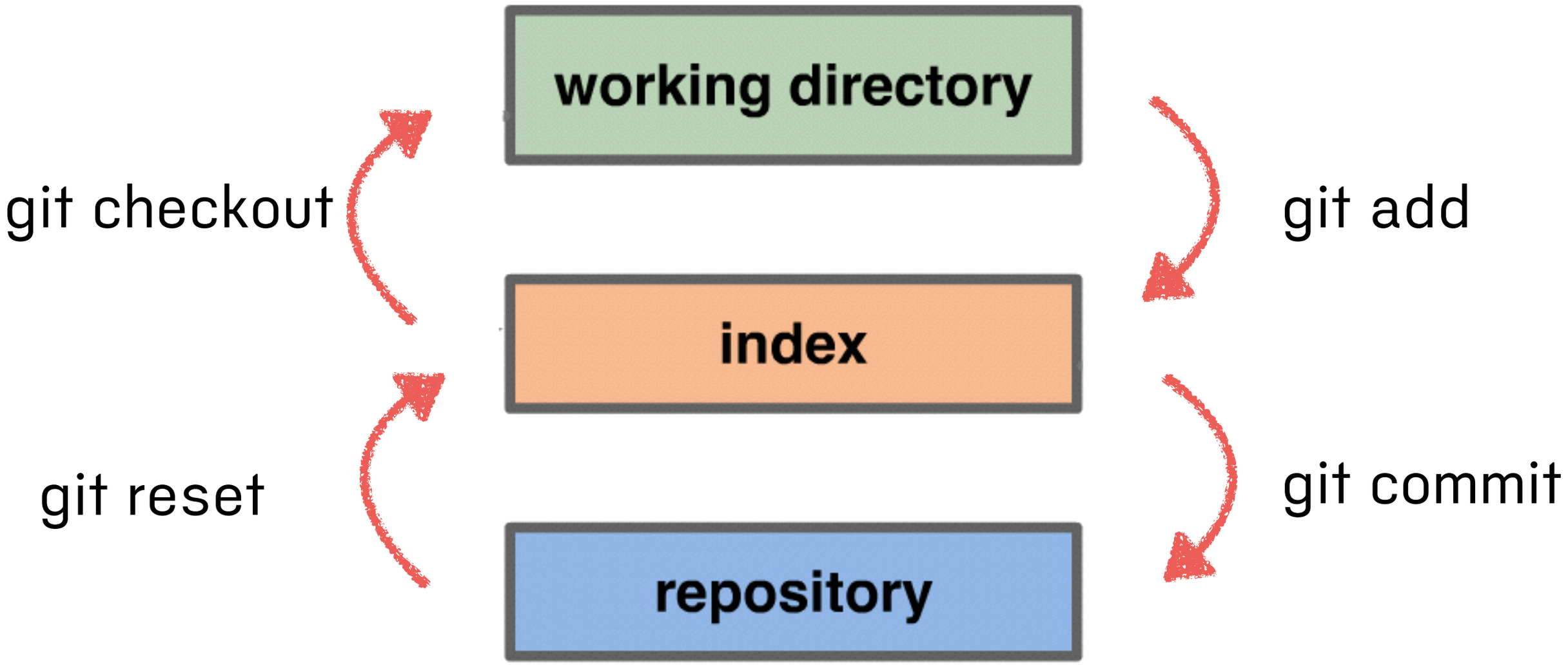
```
[master 5c319f1] Update hello file
 1 file changed, 1 insertion(+)
```

$git status

```
On branch master
nothing to commit, working directory clean
```

Basic workflow

1. Edit files
2. Stage the changes => **git add**
3. Review your changes => **git status**
4. Commit the changes => **git commit**

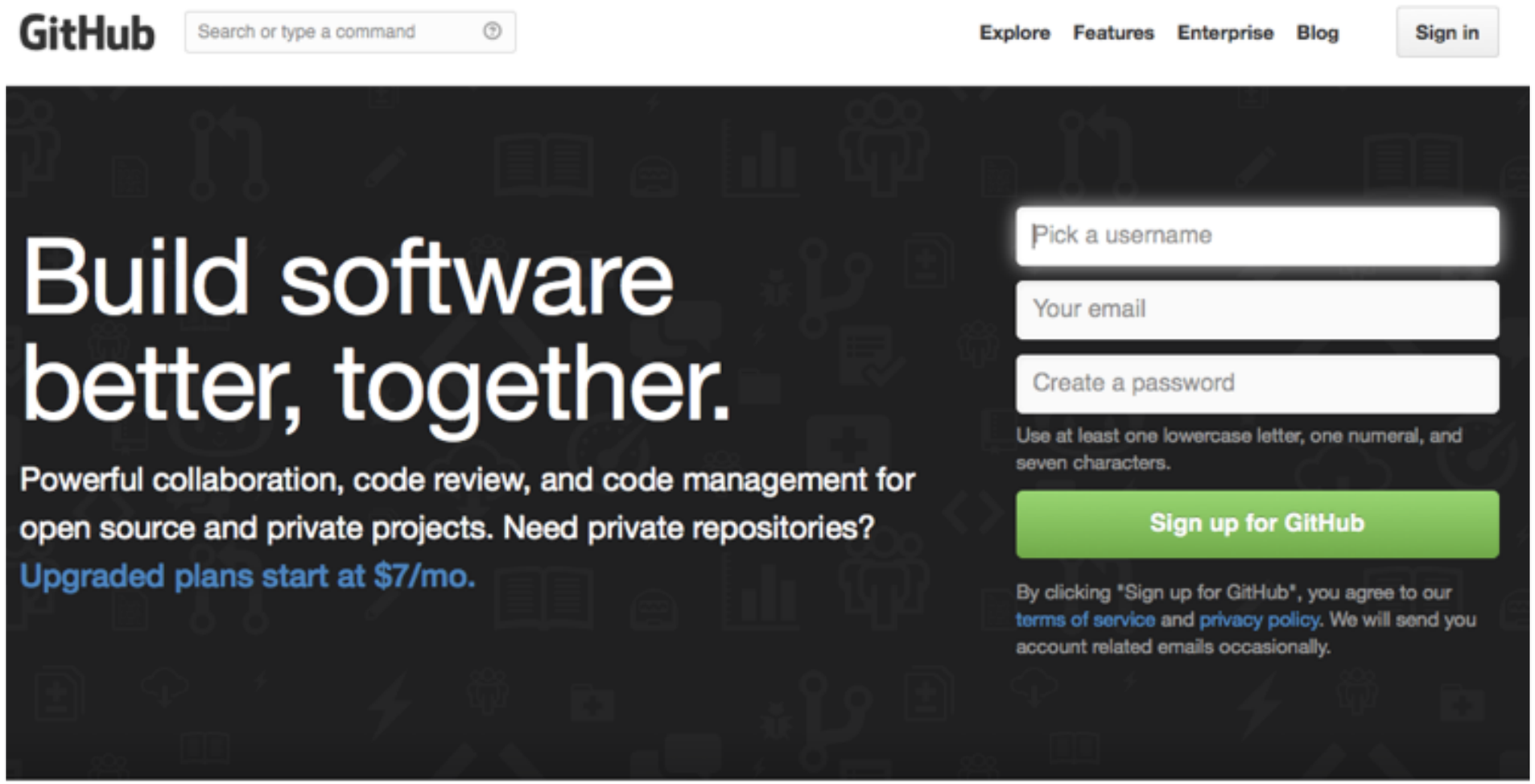

working directory
index
repository
git checkout
git add
git reset
git commit

Working with github.com

ใช้งานร่วมกับ github

Register account

www.github.com



Why you'll love GitHub.

Powerful features to make software development more collaborative.

Generate ssh key

$ssh-keygen -t rsa -C “email@example.com”

Go home’s user folder

$cd .ssh

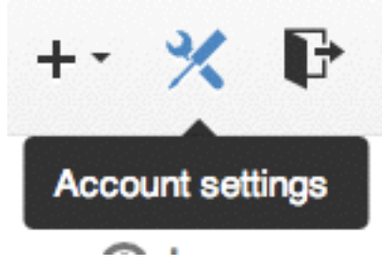
is.rsa

is.rsa.pub

private และ public key

Add your ssh key to github.com

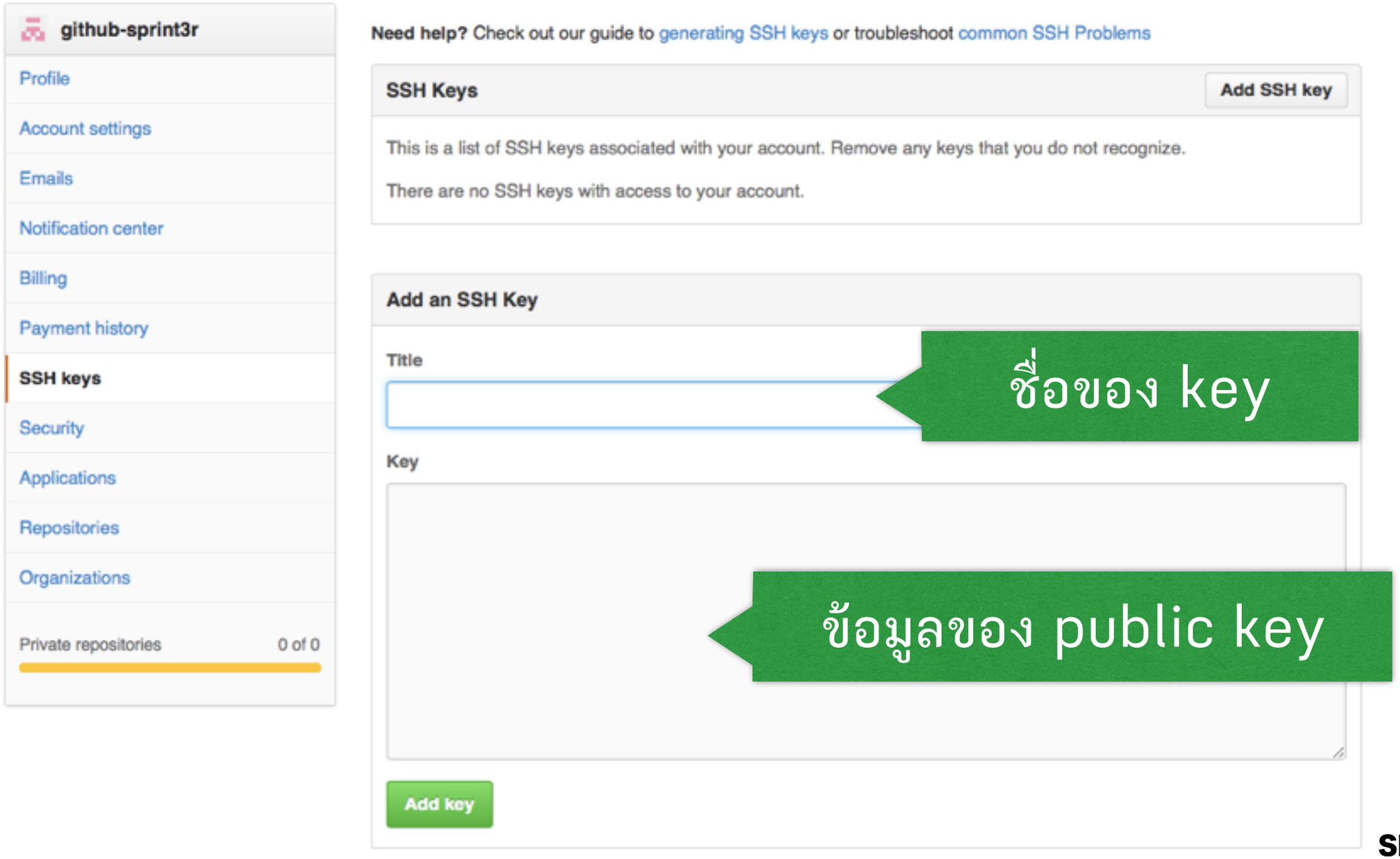
1. Go to account setting



2. Choose SSH Keys tab and Add SSH Key

Add your ssh key to github.com

2. Choose SSH Keys tab and Add SSH Key



Test drive with github.com

$**ssh** -T git@github.com

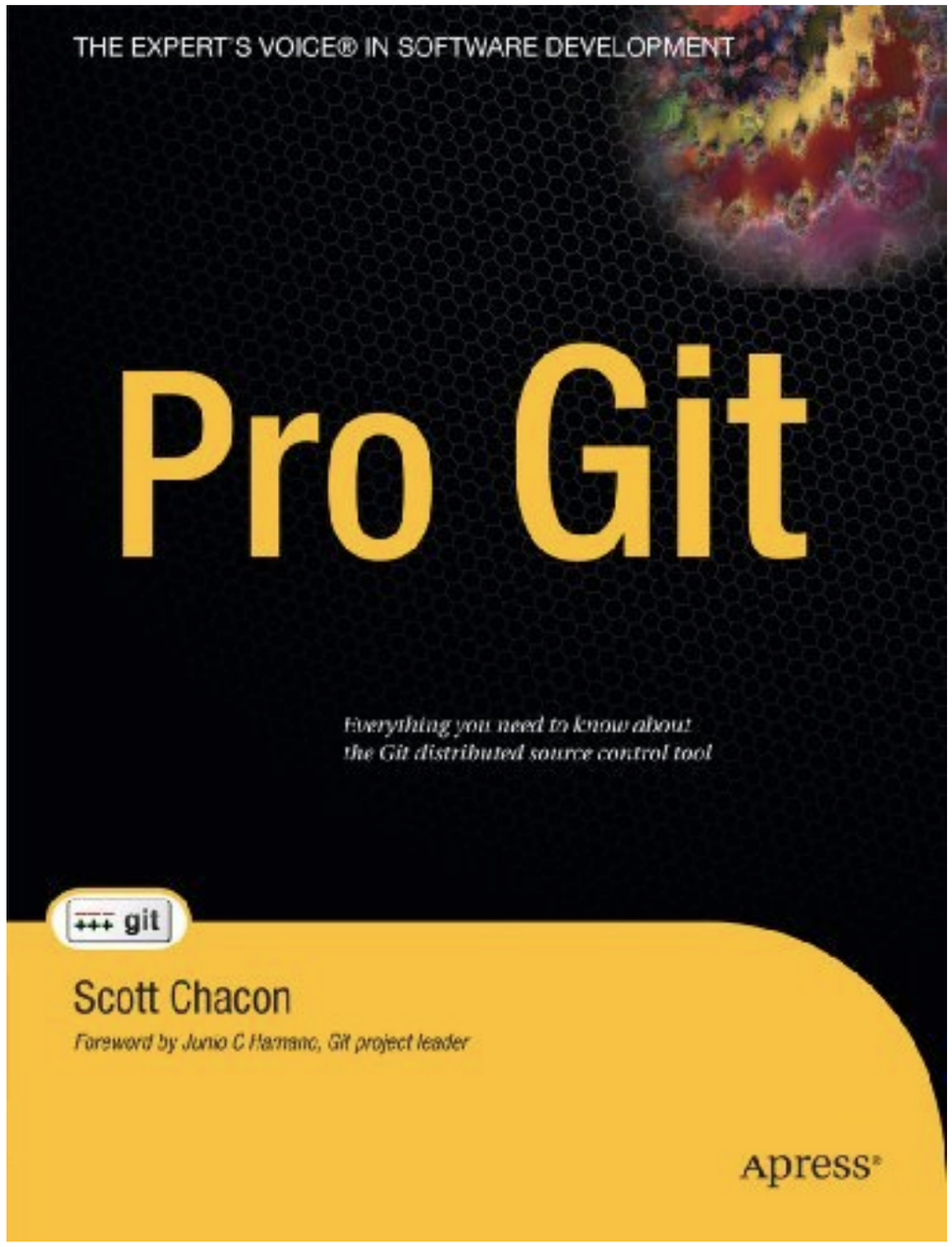
Hi **up1**! You've successfully authenticated, but GitHub does not provide shell access.

ชื่อ account ของ github.com

Resources

แหล่งความรู้ที่น่าสนใจ


tryGit
try.github.com


THE EXPERT'S VOICE® IN SOFTWARE DEVELOPMENT
Pro Git
Everything you need to know about
the Git distributed source control tool
git
Scott Chacon
Foreword by Junio C Hamano, Git project leader
Apress®

Pro Git Book

Git Reference

gitref.org

Reference About § Site Source

Getting and Creating Projects

- **init**
- **clone**

Basic Snapshotting

- **add**
- **status**
- **diff**
- **commit**
- **reset**
- **rm, mv**
- **stash**

Branching and Merging

- **branch**
- **checkout**
- **merge**
- **log**
- **tag**

INTRODUCTION TO THE GIT REFERENCE

This is the Git reference site. It is meant to be a quick reference for learning and remembering the most im
are organized into sections of the type of operation you may be trying to do, and will present the common c
common tasks.

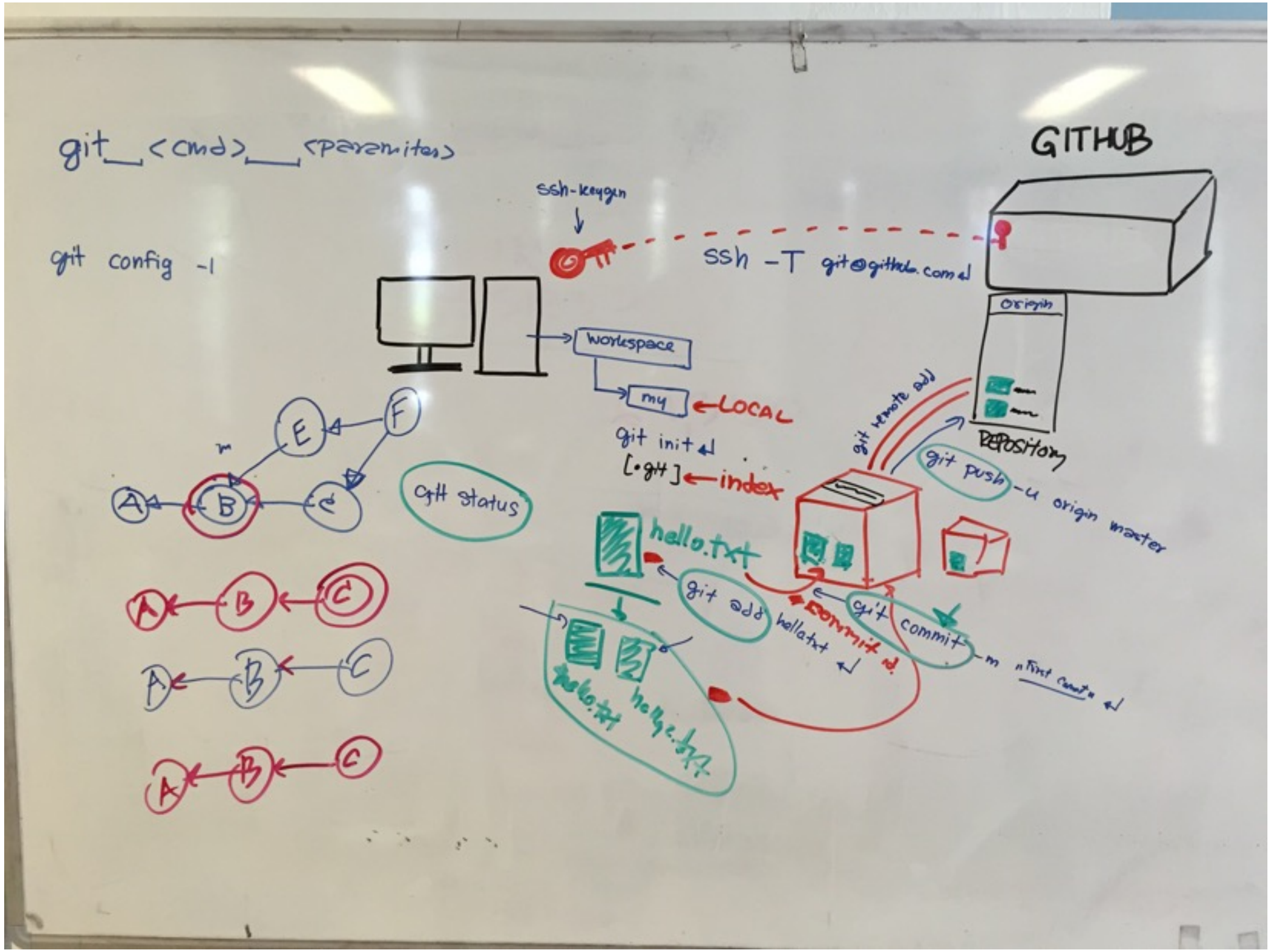
Each section will link to the next section, so it can be used as a tutorial. Every page will also link to more in
and relevant sections in the **Pro Git book**, so you can learn more about any of the commands. First, we'll s
does.

HOW TO THINK LIKE GIT

The first important thing to understand about Git is that it thinks about version control very differently than S
to. It is often easier to learn Git by trying to forget your assumptions about how version control works and tr

Let's start from scratch. Assume you are designing a new source code management system. How did you
Chances are that you simply copied your project directory to save what it looked like at that point.

```
$ cp -R project project.bak
```



SPRINT3R

Siam Chamnan Kit Co., Ltd., and Odd-e (Thailand) Co., Ltd.