Git Beginner Guide

Jaewoong Lee

Ulsan National Institute of Science and Technology

jwlee230@unist.ac.kr

February 3, 2020

Contents

1 Introduction

2 Basic Step

3 Advanced Step

Introduction

In this guide, we will discuss about followings:

1. Git
2. GitHub
3. GitHub Desktop (`https://desktop.github.com`)

As the other program does, Git is basically controlled CLI. But, I don't want to go harder. In this guide, I will use GUI mainly.

Git?

Figure: Git

Git is Version Control System (VCS) made by Linus Torvalds.

VCS?

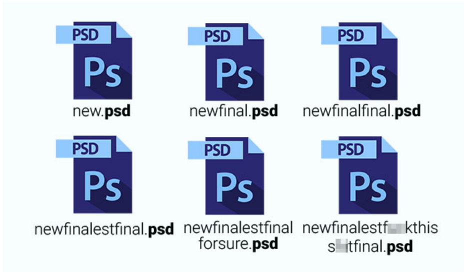


Figure: Without VCS

VCS? *(Cont.)*

With VCS, you can get advantages like:

- Revision Control
- Version Control
- Backup & Restore
- Collaboration

GitHub?

"GitHub is how people build software."

Advantages:

- Free to personal usage.
- Many open source programs are managed by GitHub.
- Issue tracker: you can track the issue of your program

Git Repository

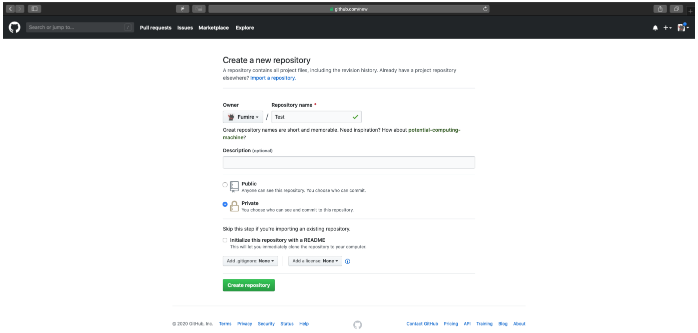
Git Repository means where git save files. There are two types of repository:

1. Remote Repository
2. Local Repository

Usually,

1. you *clone* (download) files from remote repository;
2. edit files on local repository;
3. and, *push* (upload) to remote repository.

Practice 01



Create a new repository

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

Owner: Fumire / Repository name *: Test

Great repository names are short and memorable. Need inspiration? How about potential-computing-machine?

Description (optional)

○ Public
Anyone can see this repository. You choose who can commit.

● Private
You choose who can see and commit to this repository.

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

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

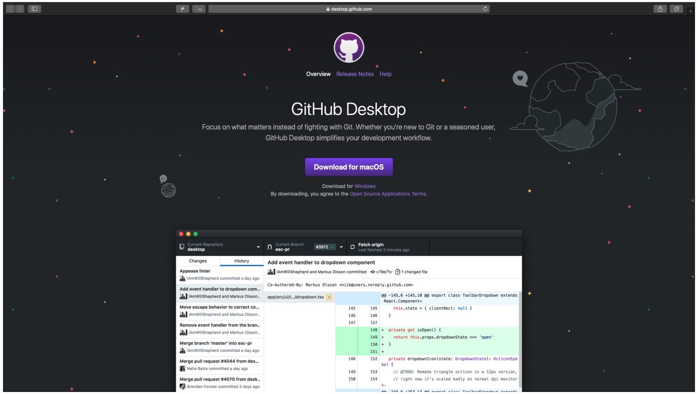
Add .gitignore: None | Add a license: None

Create repository

© 2020 GitHub, Inc. Terms Privacy Security Status Help
Contact GitHub Pricing API Training Blog About

Register **GitHub**, and make a repository with named 'Test'.

Practice 02



GitHub Desktop

Focus on what matters instead of fighting with Git. Whether you're new to Git or a seasoned user, GitHub Desktop simplifies your development workflow.

Download for Windows
By downloading, you agree to the Open Source Applications Terms.

Appease linter
iAmWillShepherd committed a day ago
Add event handler to dropdown com...
iAmWillShepherd and Markus Olsson...
Move escape behavior to correct co...
iAmWillShepherd and Markus Olsson...
Remove event handler from the bran..
iAmWillShepherd and Markus Olsson...
Merge branch 'master' into esc-pr
iAmWillShepherd committed a day ago
Merge pull request #4044 from des...
Neha Batra committed a day ago
Merge pull request #4070 from desk..
Brendan Forster committed 2 days ago

Add event handler to dropdown component
iAmWillShepherd and Markus Olsson committed c79e71c 1 changed file

Co-Authored-By: Markus Olsson <niik@users.noreply.github.com>

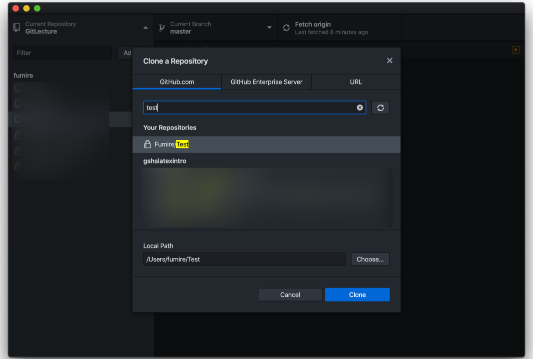
app/src/ui/t.../dropdown.tsx

```
@@ -145,6 +145,10 @@ export class ToolbarDropdown extends
 React.Component<
      this.state = { clientRect: null }
    }

+  private get isOpen() {
+    return this.props.dropdownState === 'open'
+  }
+
   private dropdownIcon(state: DropdownState): OcticonSym
bol {
     // @TODO: Remake triangle octicon in a 12px version,
     // right now it's scaled badly on normal dpi monitor
s.
```

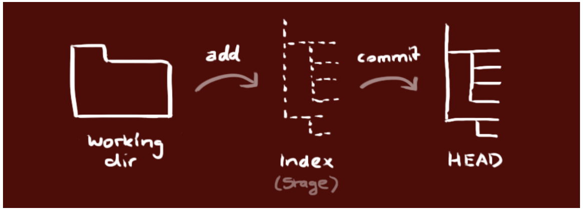
Download & Install 'GitHub Desktop' which gives GUI control with git.

Practice 03 (Clone)



Clone the repository from GitHub as figure.

Trees

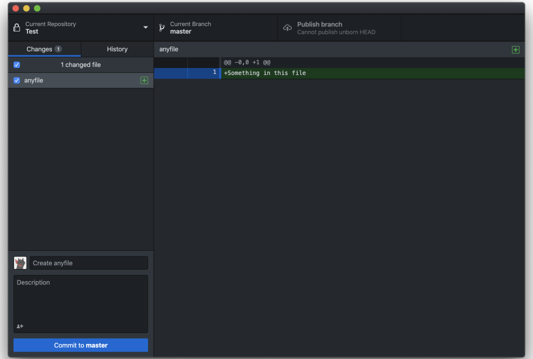


There are three tree which managed by git.

1. Working Directory: which consist of real files
2. Index: staging area (ready area)
3. HEAD: the final files

You can *add* any files from working directory to index.
Also, you could *commit* changes from index to HEAD.
You could add *tag* to commit.

Practice 04



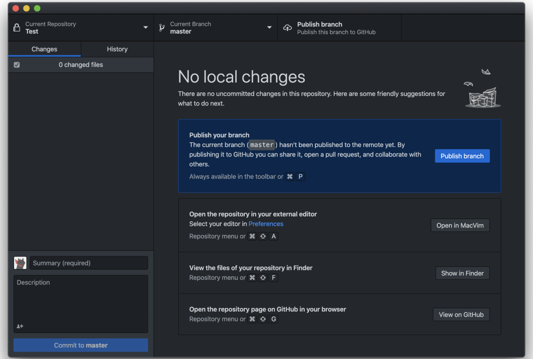
```
@@ -0,0 +1 @@
+Something in this file
```

Add any file to working directory, then GitHub Desktop automatically finds the changes as figures. Commit the changes.

Push

However, even you commit the changes, the changes are not applied to remote repository.
The changes are only in local repository.
To apply changes, you should *push* the changes to remote repository.

Practice 05



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Publish your branch
The current branch (`master`) hasn't been published to the remote yet. By publishing it to GitHub you can share it, open a pull request, and collaborate with others.
Always available in the toolbar or ⌘ P

Publish branch

Open the repository in your external editor
Select your editor in Preferences
Repository menu or ⌘ ⇧ A

Open in MacVim

View the files of your repository in Finder
Repository menu or ⌘ ⇧ F

Show in Finder

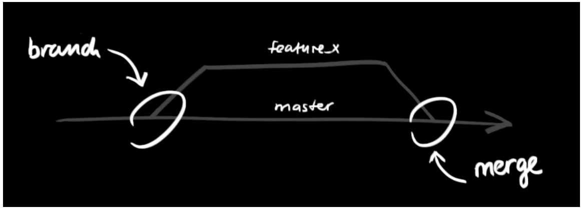
Open the repository page on GitHub in your browser
Repository menu or ⌘ ⇧ G

View on GitHub

Let's push the changes to remote repository.

Branch / Merge

You can *branch & merge* the changes.
The **master** branch will be automatically generated when creating repository.



You can add/delete branches; and move among the branches.

Conflict

Git automatically try to merge changes.
However, sometimes the *conflict* occurs; in other words, you should solve the twisted.
After you solve the twisted, add/commit the solved as other changes.

Pull

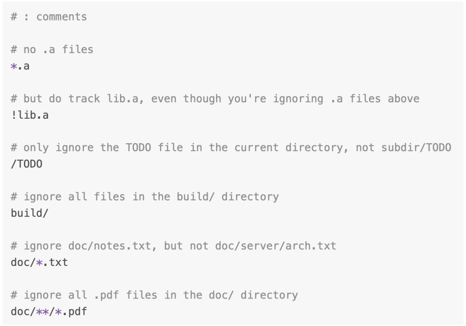
For update as remote directory, you should *pull* the repository.
With *pull* command, the changes of remote directory are *fetched* and *merged*.
Sometimes, as *merging*, conflict can be occurred, and you should solve this.

Advanced Step

After this page, you will get advanced step for git.

.gitignore

You can specify the files which you do not want to upload to git.



```
# : comments

# no .a files
*.a

# but do track lib.a, even though you're ignoring .a files above
!lib.a

# only ignore the TODO file in the current directory, not subdir/TODO
/TODO

# ignore all files in the build/ directory
build/

# ignore doc/notes.txt, but not doc/server/arch.txt
doc/*.txt

# ignore all .pdf files in the doc/ directory
doc/**/*.pdf
```

`https://www.gitignore.io` gives the basic .gitignore file.

.gitkeep

Similar with .gitignore.
Git does not track empty directory, so add empty file to track/*keep* the directory.

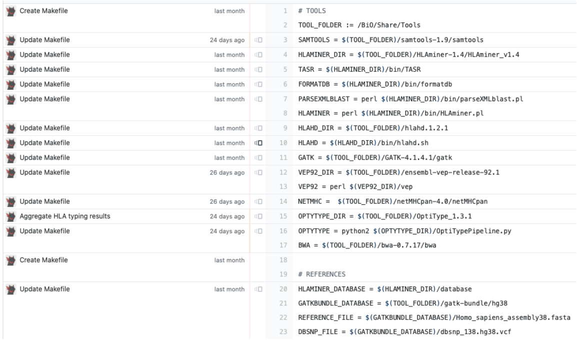
Pull Request

When you are in collaboration and you have developed an important feature, you can send PR to your teammates to use the important features.

Fork

You can fork(copy) other's work to your repository.
Fork is similar to Clone; however, you cannot change directly to original repository with fork.
Clone: Two-way $\Leftrightarrow$ Fork: One-way
Still, you can send PR to forked repository.

Blame



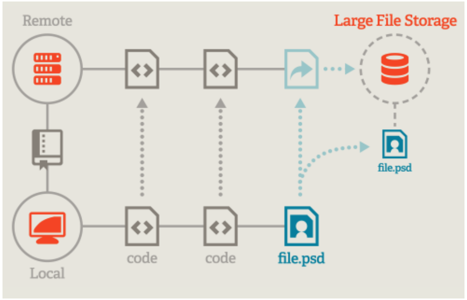
```
Create Makefile                 last month     # TOOLS
                                               TOOL_FOLDER := /BiO/Share/Tools
Update Makefile                 24 days ago    SAMTOOLS = $(TOOL_FOLDER)/samtools-1.9/samtools
Update Makefile                 last month     HLAMINER_DIR = $(TOOL_FOLDER)/HLAminer-1.4/HLAminer_v1.4
Update Makefile                 last month     TASR = $(HLAMINER_DIR)/bin/TASR
Update Makefile                 last month     FORMATDB = $(HLAMINER_DIR)/bin/formatdb
Update Makefile                 last month     PARSEXMLBLAST = perl $(HLAMINER_DIR)/bin/parseXMLblast.pl
                                               HLAMINER = perl $(HLAMINER_DIR)/bin/HLAminer.pl
Update Makefile                 last month     HLAHD_DIR = $(TOOL_FOLDER)/hlahd.1.2.1
Update Makefile                 last month     HLAHD = $(HLAHD_DIR)/bin/hlahd.sh
Update Makefile                 last month     GATK = $(TOOL_FOLDER)/GATK-4.1.4.1/gatk
Update Makefile                 26 days ago    VEP92_DIR = $(TOOL_FOLDER)/ensembl-vep-release-92.1
                                               VEP92 = perl $(VEP92_DIR)/vep
Update Makefile                 26 days ago    NETMHC =  $(TOOL_FOLDER)/netMHCpan-4.0/netMHCpan
Aggregate HLA typing results    24 days ago    OPTYTYPE_DIR = $(TOOL_FOLDER)/OptiType_1.3.1
Update Makefile                 24 days ago    OPTYTYPE = python2 $(OPTYTYPE_DIR)/OptiTypePipeline.py
                                               BWA = $(TOOL_FOLDER)/bwa-0.7.17/bwa
Create Makefile                 last month
                                               # REFERENCES
Update Makefile                 last month     HLAMINER_DATABASE = $(HLAMINER_DIR)/database
                                               GATKBUNDLE_DATABASE = $(TOOL_FOLDER)/gatk-bundle/hg38
                                               REFERENCE_FILE = $(GATKBUNDLE_DATABASE)/Homo_sapiens_assembly38.fasta
                                               DBSNP_FILE = $(GATKBUNDLE_DATABASE)/dbsnp_138.hg38.vcf
```

With Blame, you can find who write the code line-by-line.

Blame *(Cont.)*

You easily find who will be blamed or have responsibility to make the bugs.

Git - Large File Storage



With normal(plain) git, only file which are less than 60 MB would be tracked. However, with git-LFS, you can track large file as normal(small) file.

References

- Git - The Simple Guide: `https://github.com/rogerdudler/git-guide`
- Apply Git .gitignore: `https://nesoy.github.io/articles/2017-01/Git-Ignore`
- Git Large File Storage: `https://git-lfs.github.com`