

CSE2019: Open-source Software Development

Lab 1: Git

Software Engineering Lab

Except where otherwise noted, the contents of this document are Copyright 2017 Gwanggyu Choi, Youn-geun Ahn and Scott Uk-Jin Lee All rights reserved. Any redistribution, reproduction, transmission, or storage of part or all of the contents in any form is prohibited without author's expressed written permission.

Git



git

lab(se);

VCS(Version Control System)

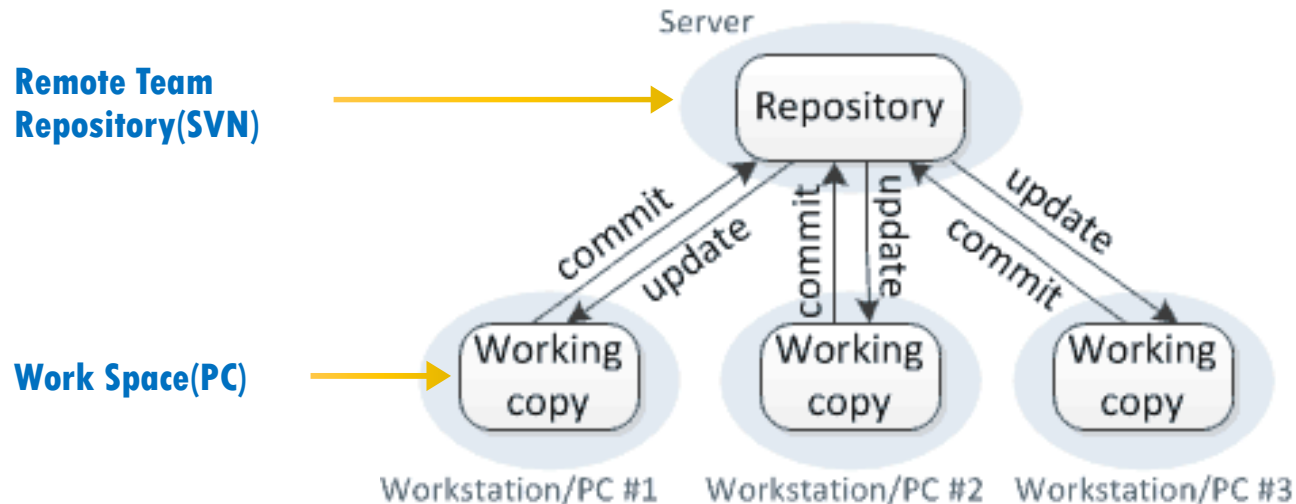
Database for 'Version'

Why use VCS?

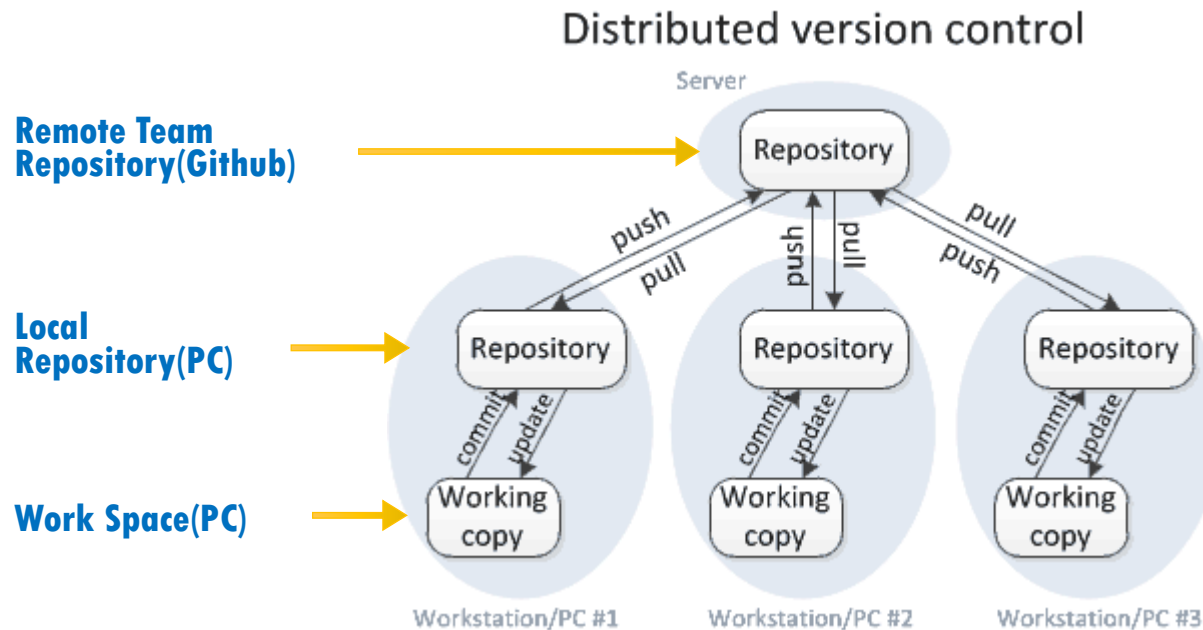
- **Collaboration**
- **Backup**
- **Tracking source code**
- **Tracking what happened**
- **Reduce side-effect
when you develop a new module**

Centralized Version Control

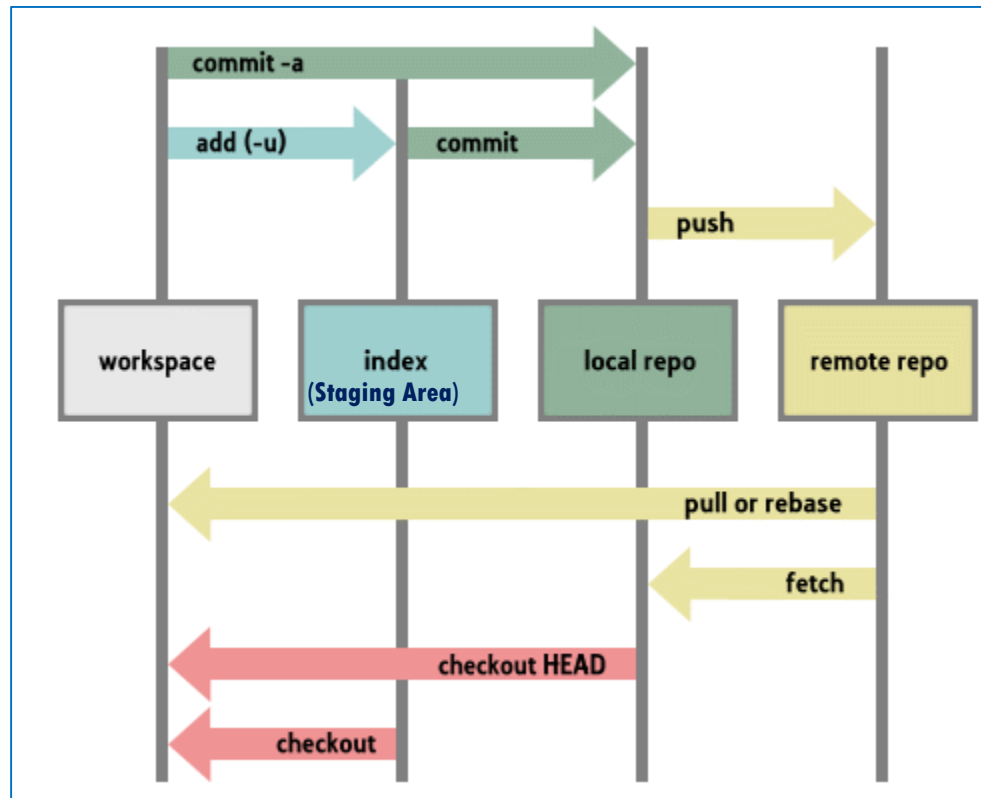
Centralized version control



Distributed VC



Git Workflow



Git Install

- **OS X**

- <http://sourceforge.net/projects/git-osx-installer/>

- **Windows**

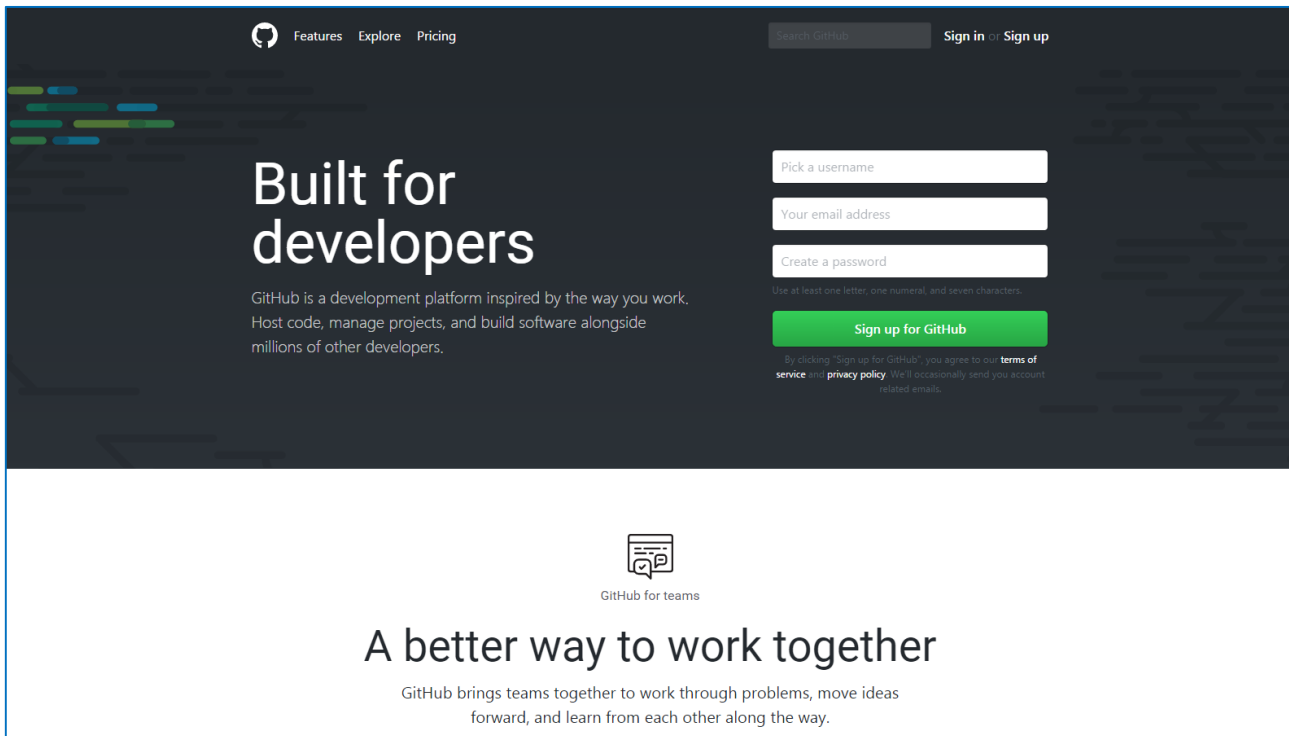
- <http://msysgit.github.com/>

- **Linux**

- **\$ apt-get install git**

Github

- Sign up <https://github.com>



The screenshot shows the GitHub homepage with a dark theme. At the top, there is a navigation bar with the GitHub logo, links for 'Features', 'Explore', and 'Pricing', a search bar, and links for 'Sign in' and 'Sign up'. The main content area features the text 'Built for developers' and a description of GitHub as a development platform. To the right, there is a sign-up form with three input fields: 'Pick a username', 'Your email address', and 'Create a password'. Below the password field, there is a note about password requirements. A green 'Sign up for GitHub' button is positioned below the form. At the bottom of the page, there is a section for 'GitHub for teams' with the text 'A better way to work together' and a description of how GitHub brings teams together.

Features Explore Pricing Search GitHub Sign in Sign up

Built for developers

GitHub is a development platform inspired by the way you work. Host code, manage projects, and build software alongside millions of other developers.

Pick a username

Your email address

Create a password

Use at least one letter, one numeral, and seven characters.

[Sign up for GitHub](#)

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We'll occasionally send you account related emails.

GitHub for teams

A better way to work together

GitHub brings teams together to work through problems, move ideas forward, and learn from each other along the way.

Github

- Sign up

The screenshot shows the GitHub sign-up page with the following elements and annotations:

- Header:** "Join GitHub" and "The best way to design, build, and ship software."
- Progress Bar:** Three steps: "Step 1: Set up a personal account" (active), "Step 2: Choose your plan", and "Step 3: Tailor your experience".
- Form Section:** "Create your personal account".
 - Username:** Input field with "AhnYounGeun". A red error message "Username is already taken" is shown below it. A yellow arrow points to this message with the text "Must use hanyang.ac.kr (for private repository)".
 - Email:** Input field with "frebern@hanyang.ac.kr". This field is highlighted with a red rectangle. A red error message "Email is invalid or already taken" is shown below it.
 - Password:** Input field with masked characters ".....". A green checkmark is shown to the right.
 - Instructions:** "Use at least one lowercase letter, one numeral, and seven characters."
- Benefits Section:** "You'll love GitHub" with a list:
 - Unlimited collaborators
 - Unlimited public repositories
 - Great communication (checked)
 - Frictionless development (checked)
 - Open source community (checked)
- Footer:** "By clicking on 'Create an account' below, you are agreeing to the [Terms of Service](#) and the [Privacy Policy](#)."
- Buttons:** A green "Create an account" button at the bottom.

Annotations on the image:

- A yellow arrow points from the text "Click" to the "Create an account" button.
- A yellow arrow points from the text "Must use hanyang.ac.kr (for private repository)" to the "Username is already taken" error message.

Github

• Sign up

받은메일함

답장 | 전달 | 수신거부 | ✕ 삭제 | 메일 이동 | 기타

보낸 사람

받는 사람

보낸 날짜

제목

GitHub

AhnYounGeun

17:30:46

📧 [GitHub] Please verify your email address.

수신거부

자동분류 추가

주소록추가

받은 날짜

17:30:52

Hi @AhnYounGeun!

Help us secure your GitHub account by verifying your email address (frebern@hanyang.ac.kr). This lets you access all of GitHub's features.

[Verify email address](#) ← Click

Button not working? Paste the following link into your browser:
https://github.com/users/AhnYounGeun/emails/29288147/confirm_verification/026545debc465b60001f960f088299890a749790

You're receiving this email because you recently created a new GitHub account or added a new email address. If this wasn't you, please ignore this email.

Github

- **Create new remote repository**



The screenshot shows the GitHub web interface. At the top, there's a header with 'Your repositories 15' and a green 'New repository' button. Below this is a search bar labeled 'Find a repository...' and tabs for 'All', 'Public', and 'Private'. A light blue overlay box is positioned in the center, containing the text 'Learn Git and GitHub without any code!' and 'Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.' At the bottom of the overlay are two buttons: a green 'Read the guide' button and a grey 'Start a project' button. Two yellow arrows point to these buttons with the text 'Click (if you are already registered in GitHub)' pointing to the 'New repository' button and 'Click (if you are new GitHub user)' pointing to the 'Read the guide' button.

Click (if you are already registered in GitHub)

Click (if you are new GitHub user)

Github

Check to Private



Check



Repository Name



Click





Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name


 AhnYounGeun ▾


/ SE-2017 

Great repository names are short and memorable. Need inspiration? How about **sturdy-octo-guacamole**.

Description (optional)

2017 CSE4006 Software Engineering

☐  Public

☒  Private


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

You choose who can see and commit to this repository.

☒ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾

Add a license: None ▾ 

Create repository

Github

The screenshot shows the GitHub interface for a repository named 'AhnYounGeun / SE-2017'. The repository is private and has 1 commit, 1 branch, 0 releases, and 1 contributor. The 'Clone or download' button is highlighted with a red box, and a yellow arrow points to it with the text 'Remote Repository URL'. A dropdown menu is open, showing the 'Clone with HTTPS' option and the URL 'https://github.com/AhnYounGeun/SE-2017.git'. The 'Open in Desktop' and 'Download ZIP' buttons are also visible.

GitHub interface showing the repository **AhnYounGeun / SE-2017** (Private).

Repository statistics: 1 commit, 1 branch, 0 releases, 1 contributor.

Buttons: Watch, Star, Fork, Clone or download (highlighted).

Clone options:

- Clone with HTTPS (selected): `https://github.com/AhnYounGeun/SE-2017.git`
- Use SSH
- Open in Desktop
- Download ZIP

Repository content: README.md (Initial commit).

Repository description: SE-2017

2017 CSE4006 Software Engineering

Git bash



A screenshot of a Git Bash terminal window. The title bar shows the path "MINGW64:/c/Users/user/Desktop/opensource/a". The terminal content shows a user navigating through directories: first listing the contents of the current directory, then checking the current path with 'pwd', and finally changing to a subdirectory 'a' with 'cd a/'.

```
user@sec MINGW64 ~/Desktop/opensource
$ ls
a/  b/  c/  d/

user@sec MINGW64 ~/Desktop/opensource
$ pwd
/c/Users/user/Desktop/opensource

user@sec MINGW64 ~/Desktop/opensource
$ cd a/

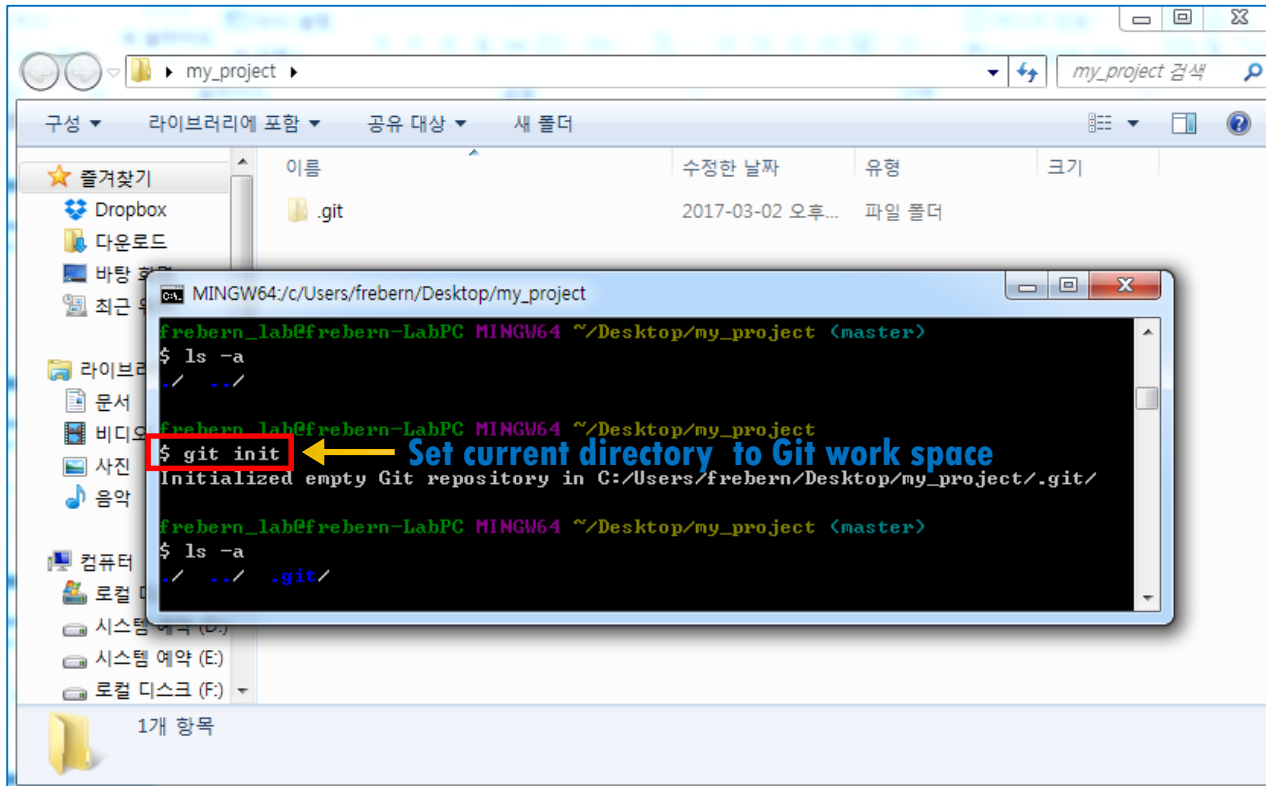
user@sec MINGW64 ~/Desktop/opensource/a
$
```

Linux – basic command

- **ls (ls -al)**
- **cd (.. , ./)**
- **pwd**
- **mkdir**
- **rm**
- **cp**
- **mv**
- **history**
- **clear**


Git - Work Space

git init



Git - Work Space

git init



```
MINGW64:/c/Users/user/Desktop/opensource

user@sec MINGW64 ~/Desktop/opensource
$ ls -a
./  ../  a/  b/  c/  d/

user@sec MINGW64 ~/Desktop/opensource
$ git init
Initialized empty Git repository in C:/Users/user/Desktop/opensource/.git/

user@sec MINGW64 ~/Desktop/opensource (master)
$ ls -a
./  ../  .git/  a/  b/  c/  d/

user@sec MINGW64 ~/Desktop/opensource (master)
$
```

Git - Work Space

git config

```
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>  
$ git config --global user.name "frebern"  
  
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>  
$ git config --global user.email frebern@naver.com
```

Git - Work Space

git remote add / git pull

Naming Remote Repository

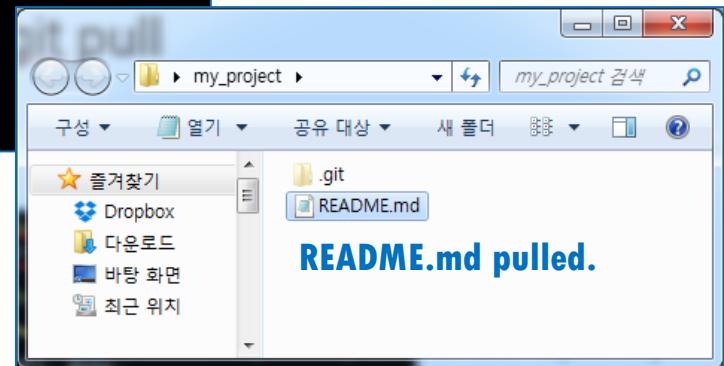
(my_remote_repo ← <https://github.com/frebern/SE-2017.git>)

```
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git remote add my_remote_repo https://github.com/frebern/SE-2017.git

frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git pull my_remote_repo master
remote: Counting objects: 3, 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), done.
From https://github.com/frebern/SE-2017
 * branch            master      -> FETCH_HEAD
 * [new branch]      master      -> my_remote_repo/master
```

Pull from my_remote_repo's master branch

* **Pull = Fetch + Merge**
(i.e. Receive current remote repository
+ Merge them to my local repository)



Git - Staging Area

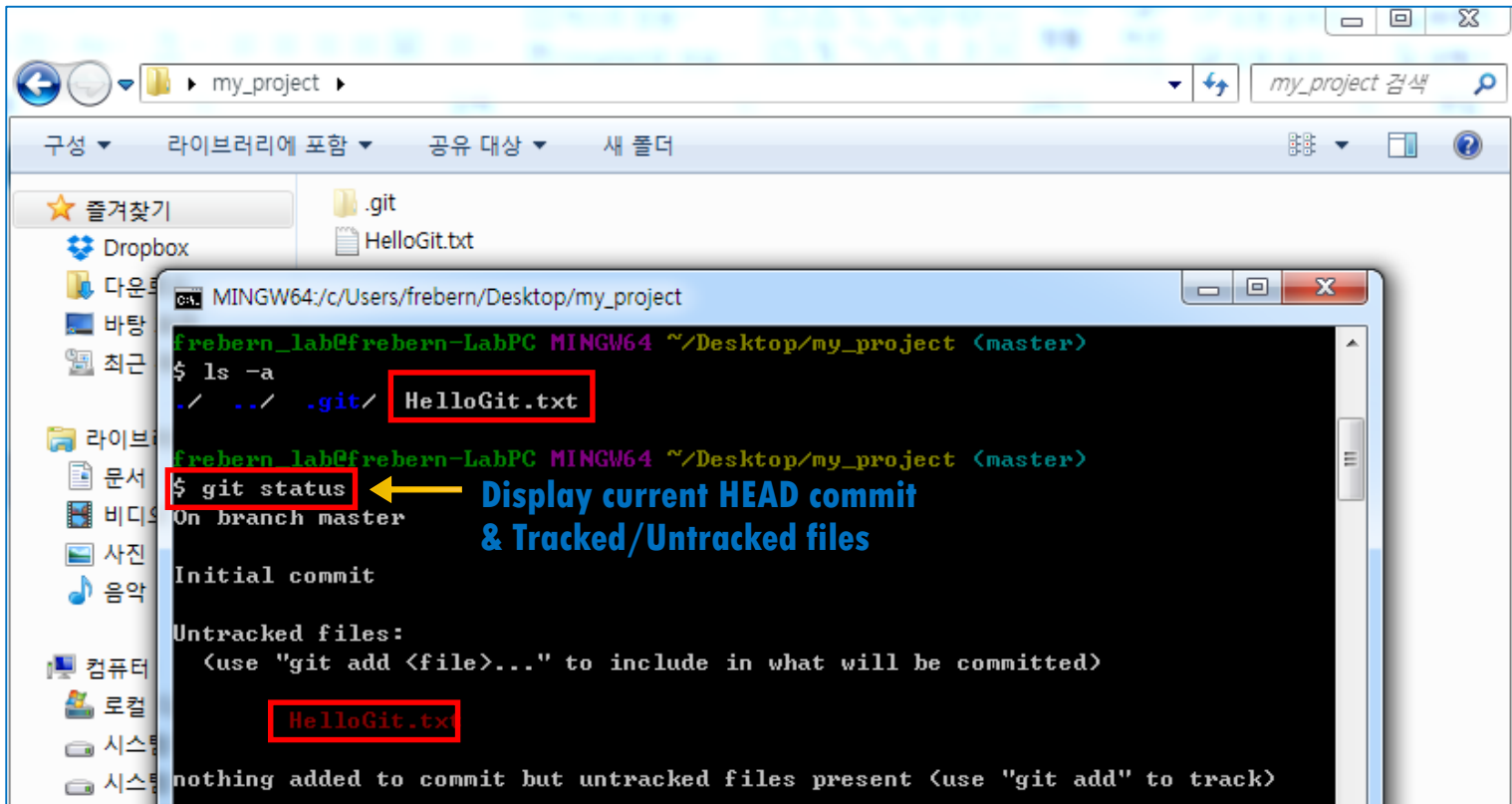
.git



‘.git’ has index of all files and changes.

Git - Staging Area

git status



The screenshot shows a Windows File Explorer window with the address bar set to 'my_project'. The left sidebar shows the '즐거찾기' (Favorites) section with 'Dropbox' and '다운로드' (Downloads) listed. The main pane shows the contents of the 'my_project' directory, which includes a '.git' folder and a 'HelloGit.txt' file. Overlaid on the File Explorer is a terminal window titled 'MINGW64: c:/Users/frebern/Desktop/my_project'. The terminal shows the following commands and output:

```
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ ls -a
./  ../  .git/  HelloGit.txt
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git status
On branch master
Initial commit

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

    HelloGit.txt

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

Red boxes highlight 'HelloGit.txt' in the terminal output and '\$ git status' in the terminal command line. A yellow arrow points from the text 'Display current HEAD commit & Tracked/Untracked files' to the '\$ git status' command.

Git - Staging Area

git add

```
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git add HelloGit.txt ← Add file to Staging Area(index)

frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git status
On branch master

Initial commit

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

    new file:   HelloGit.txt
```

Git - Local Repository

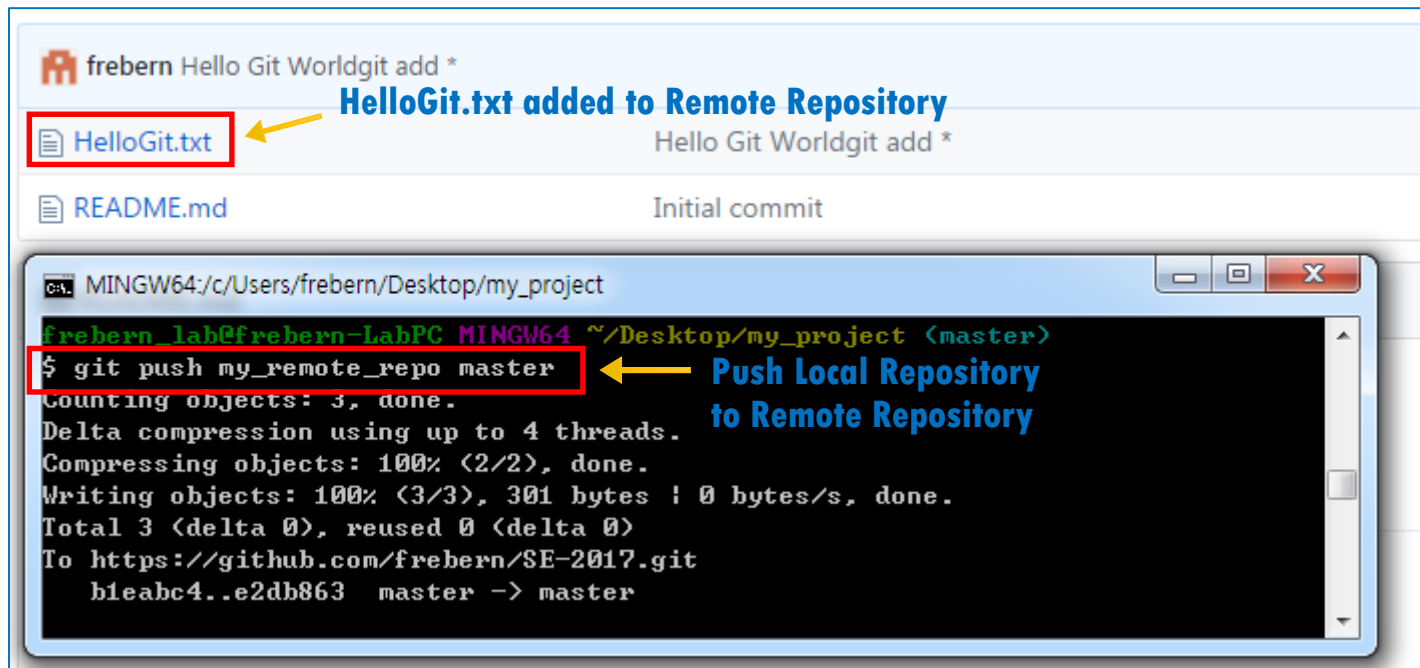
git commit

```
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git commit -m "Git Start!" ← Commit all added changes
                               & save to Local Repository
[master e2e856b] Git Start!
 1 file changed, 1 insertion(+), 1 deletion(-)

frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git status
On branch master
nothing to commit, working tree clean
```


Git - Remote Repository

git push



The screenshot displays two windows. The top window is a Git GUI titled 'frebern Hello Git Worldgit add *'. It shows a file 'HelloGit.txt' with a red box around it and a yellow arrow pointing to it, with the text 'HelloGit.txt added to Remote Repository' next to it. Below it, 'README.md' is listed with the text 'Initial commit'. The bottom window is a terminal titled 'MINGW64:/c/Users/frebern/Desktop/my_project'. It shows the command '\$ git push my_remote_repo master' in a red box with a yellow arrow pointing to it, and the text 'Push Local Repository to Remote Repository' next to it. The terminal output shows the progress of the push, including counting objects, compressing objects, and writing objects, ending with the commit hash 'b1eabc4..e2db863 master -> master'.


```
frebern Hello Git Worldgit add *
HelloGit.txt added to Remote Repository
Hello Git Worldgit add *
README.md Initial commit

MINGW64:/c/Users/frebern/Desktop/my_project
frebern_lab@frebern-LabPC MINGW64 ~/Desktop/my_project <master>
$ git push my_remote_repo master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 301 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/frebern/SE-2017.git
b1eabc4..e2db863 master -> master
```

Basic Work Flow

1. **git init**
2. **git remote add [repoName] [url]**
3. **git pull [repoName] [branch(e.g. master)]**
4. **Working...**
5. **git add ***
6. **git commit -m "commit message"**
7. **git push [repoName] [branch(e.g. master)]**
8. **Repeat 3~7**

Exercise

1. **Create an remote repository**
2. **Pull remote repository to your workspace.**
(if you can't pull them, try `--allow-unrelated-histories` option)
3. **Create a program**  It's double dash - -
if you input "1" then print your name. (any programming language is ok)
4. **Add & Commit to local repository**
5. **Push them to remote repository**
(if you can't push them, try `-f` option)
6. **Add function**
which print your introduce when you input "2"
7. **Update your remote repository**
8. **Send direct message to T.A with screenshot(github , 'history' command)**