Computer Graphics

1 - Lab: Environment Setting for Lectures & Assignments

Yoonsang Lee

Spring 2021

Introduction

- TA: 김민관
  - palkan21@naver.com

- Undergraduate Mentor: 사정

Topics Covered

- Installing Python Interpreter
- Python 2 & Python 3
- Installing Additional Python Modules
- Running Python Interpreter
- Creating a Gitlab account
- Time for Lab Assignment 1

Install Python Interpreter

- Python **3.7** or later
  - https://www.python.org/downloads/

- Note that all submissions for assignments should work in Python **3.7**.

- You can use any OS that runs Python.

Python 2 & Python 3

- Python 2 is still in active use.

- Python 3 is the future of Python.
  - A lot of very useful features & fixes for well-known problems
  - To do this, **Python 3 breaks backward compatibility**.

- If you're familiar with Python 2, you have to know the difference between Python 2 and 3.
  - The following link would be helpful:
  - http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html

Install Python Packages (Modules)

- My recommendation for installing python modules is using **pip** (Python Package Index).
  - pip is a program that helps you install most of the python package.
  - pip is already installed if you are using Python 2 >=2.7.9 or Python 3 >=3.4 downloaded from python.org
- Usage:

```
pip install <package_name>
```

Python Virtual Environments

- Example: Two python projects on the same machine,
  - Project A are based on Django 1.11
  - Project B are based on Django 2.1.7
  - Python interpreter cannot differentiate between versions of Django packages! (So you can run only one of them)

- Python virtual environment:
  - A self-contained directory tree that contains a Python installation for a particular version of Python, plus a number of additional packages.
  - You can keep dependencies required by different projects separate by creating "isolated" python virtual environments for them.

Python Virtual Environments

- **It is generally good to have one virtual environment for each Python project you work on.**
  - So the dependencies of every project are isolated from the system and each other.

- Two most popular tools:
  - virtualenv
  - Anaconda

Install virtualenv & virtualenvwrapper

- Windows

```
> pip install virtualenv virtualenvwrapper-win
(or)
> py -3 -m pip install virtualenv virtualenvwrapper-win
```

- Ubuntu

```
# if you don't have pip, install it first.
$ sudo apt-get install python3-pip

$ sudo pip3 install virtualenv virtualenvwrapper

# Add the following lines to ~/.bashrc:
export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3
source /usr/local/bin/virtualenvwrapper.sh

$ source ~/.bashrc
```

(You can skip this process if you're already using virtualenv or Anaconda.)

Install virtualenv & virtualenvwrapper

- MacOS

```
# Install Homebrew(package manager for mac OS) from
below link.
https://brew.sh/index_ko

# if you install python3 using Homebrew, pip and pip3
would be installed automatically.
$ brew install python3

$ pip3 install virtualenv virtualenvwrapper

# Add the following line to ~/.bashrc:
export VIRTUALENVWRAPPER_PYTHON=/usr/local/bin/python3
source /usr/local/bin/virtualenvwrapper.sh

$ source ~/.bashrc
```

How to use virtualenvwrapper

```
# Create an environment
$ mkvirtualenv --python=PATH_TO_PYTHON ENVNAME

# Remove an environment
$ rmvirtualenv ENVNAME

# List all of the environments
$ lsvirtualenv

# Activate an environment
$ workon ENVNAME

# Deactivate the current environment
$ deactivate
```

Create an environment for this class

- Windows

```
> mkvirtualenv --python=<python_path> cg-course
```

  - An example for <python_path>: "C:\Users\<your_id>\AppData\Local\Programs\Python\Python35\python.exe“
  - If your system does not know “mkvirtualenv”, you need to add python script directory (e.g. …\Python35\Script\) to system path.

- Ubuntu, MacOS

```
> mkvirtualenv --python=<python3.x> cg-course
```

  - Replace <python3.x> with your python version
  - e.g. --python=python3.7

Activate the environment

```
$ workon cg-course
```

- Then you can see the name of your environment in the command prompt.

- You can run the exact version of python interpreter specified in the environment just by typing "python".

Install Additional Modules

- We'll use a few python modules in this class
  - NumPy, PyOpenGL, glfw

- In the "cg-course" environment by activating it,

```
$ workon cg-course
```

- NumPy
  - Windows, Ubuntu, MacOS

```
$ pip install numpy
```

Install Additional Modules

- PyOpenGL
  - Windows

```
> pip install pipwin
> pipwin install pyopengl
```

    - or, you can also try the "pip install pyopengl" route for windows. But in that case the pyopengl installer wouldn't install the GLUT library properly. You can manually install GLUT and copy glut32.dll to "python-installation-directory/Lib/site-packages/OpenGL/DLLS"
      - http://www.cim.mcgill.ca/~fmannan/comp557/Python%20and%20PyOpenGL%20Installation.html
  - Ubuntu, MacOS

```
$ pip install PyOpenGL
```

Install Additional Modules

- GLFW
  - Windows, MacOS

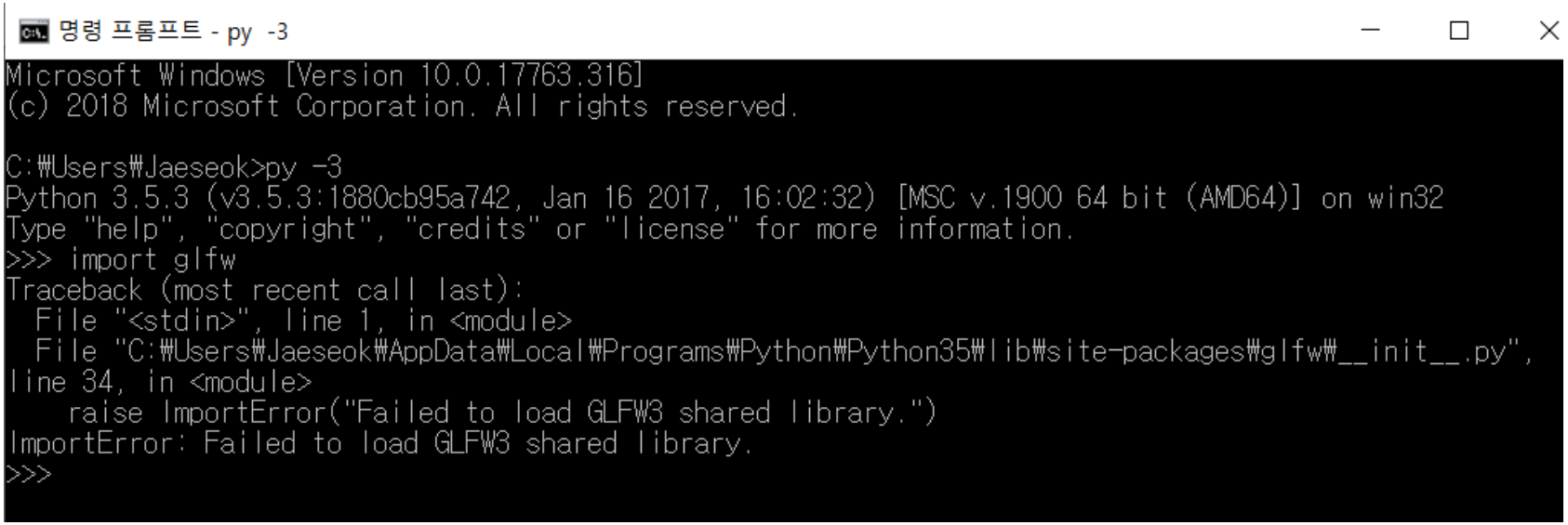
```
$ pip install glfw
```

  - Ubuntu

```
$ sudo apt-get install libglfw3
$ pip install glfw
```

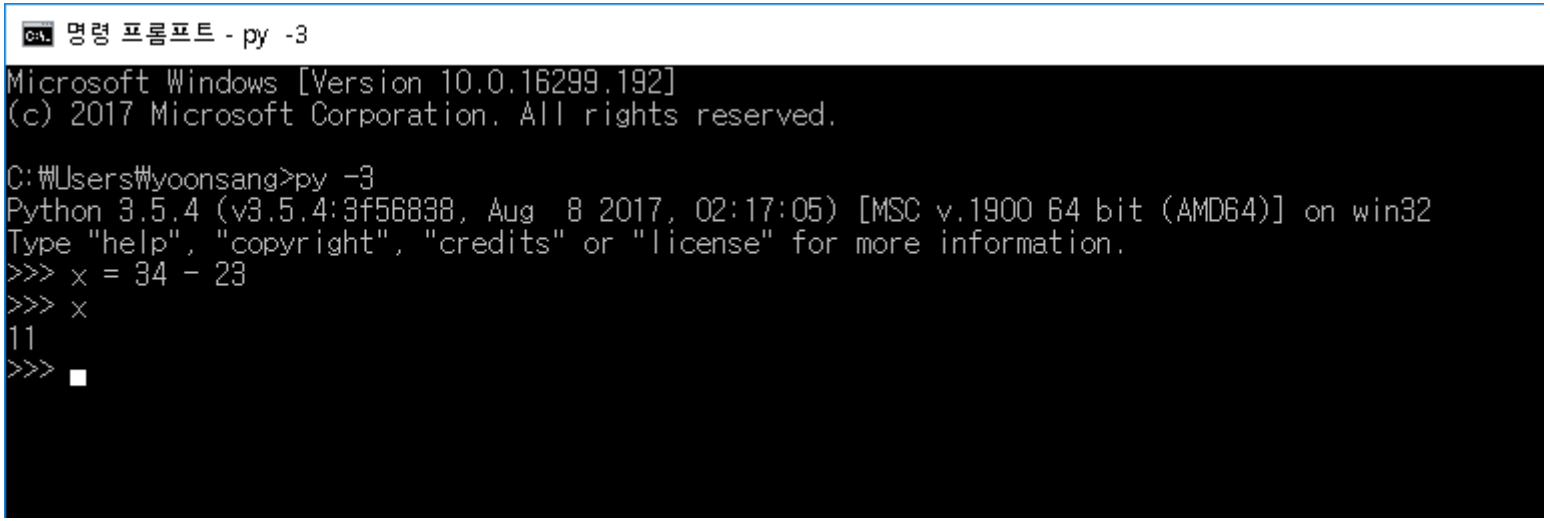
Install Additional Modules

- GLFW
  - Windows
    - If you are experiencing this error, download glfw library for the window at the following URL. Copy “lib-vc2015/glfw3.dll” to "python-installation-directory/Lib/site-packages/glfw"
      - https://www.glfw.org/download.html



Running Python Interpreter 1

- **Interactive mode**
  - Windows: Start, type “cmd”, `> py -3`
  - Ubuntu: Start, type “terminal”, `$ python3`
  - Suitable for simple tests
  - To exit the interpreter, type exit() and press enter key.



Running Python Interpreter 2

- **Non-interactive mode (runs a source file)**
  - Windows
  - Ubuntu
  - In most cases, you will use this mode.

```
> py -3 test.py
```

```
$ python3 test.py
```

- You can write a Python source file using your favorite editor.
  - Vim, Notepad++, Sublime Text, Atom, IDLE …
  - I’m personally using vim & gvim.

Python References

- https://docs.python.org/ko/3/tutorial/index.html
- https://docs.python.org/3/tutorial/index.html
- https://www.tutorialspoint.com/python3/

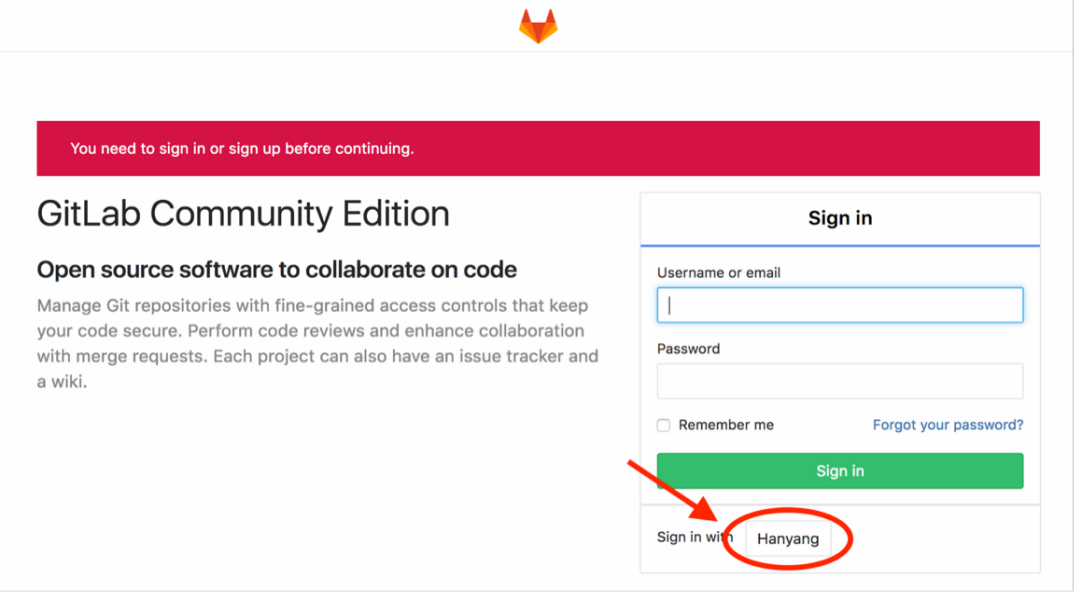
Creating a Gitlab account

Gitlab

- For today's lab assignment, submit your files via Blackboard course homepage.
- From next week's lab assignment, submit your files via the gitlab at https://hconnect.hanyang.ac.kr/
- Be sure to create a hconnect account in advance.
- If you already have a hconnect account, just skip this part.

Git

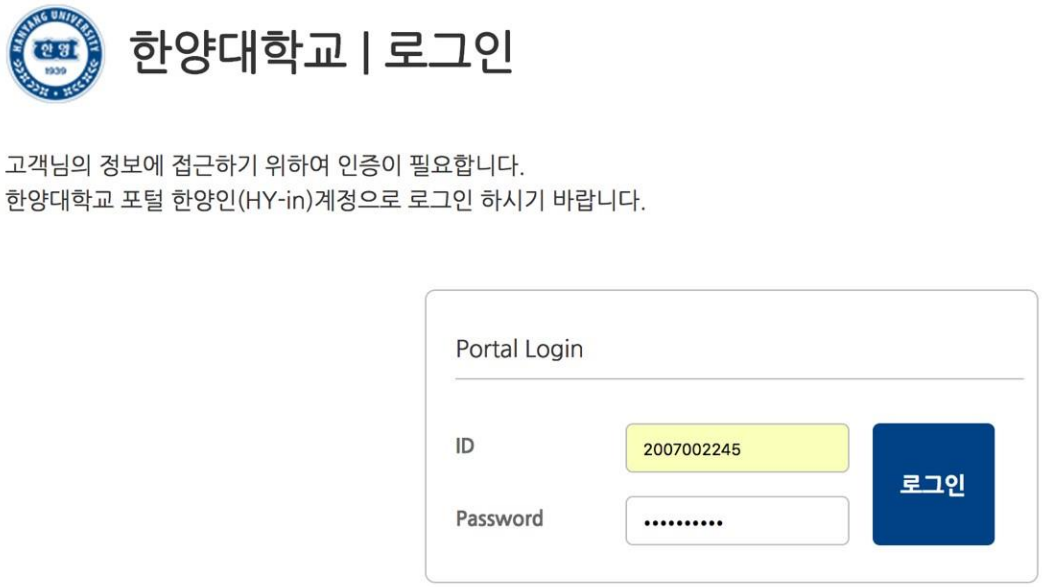
- Access to https://hconnect.hanyang.ac.kr/



HYU 한양대학교 HANYANG UNIVERSITY

Git

- Login hanyang account



HYU 한양대학교 HANYANG UNIVERSITY

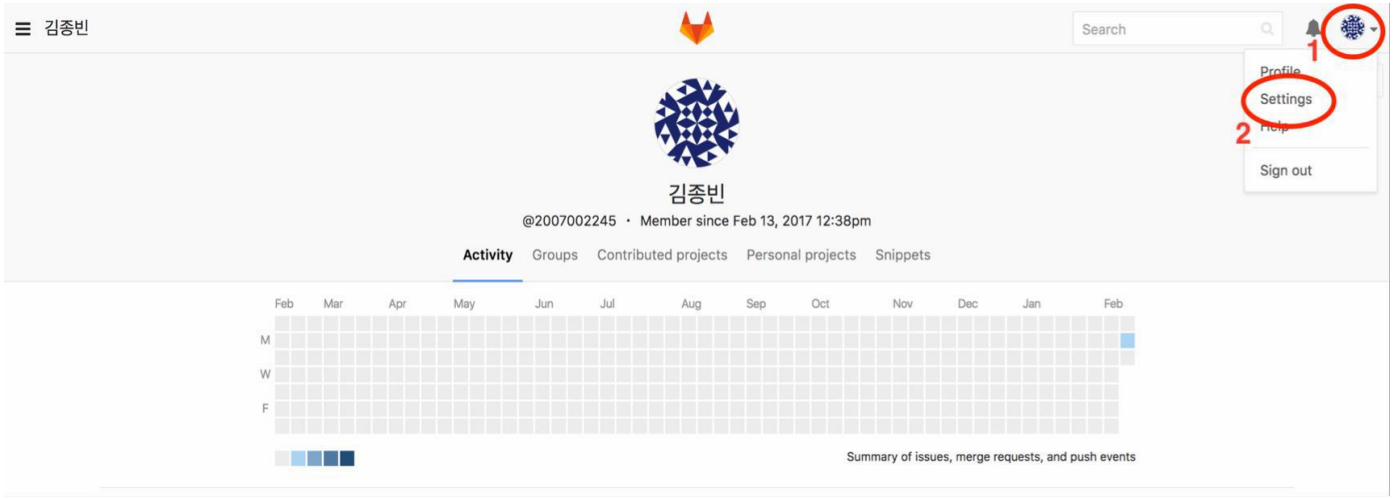
Git

- Consent for information provision

HYU 한양대학교 HANYANG UNIVERSITY

Git

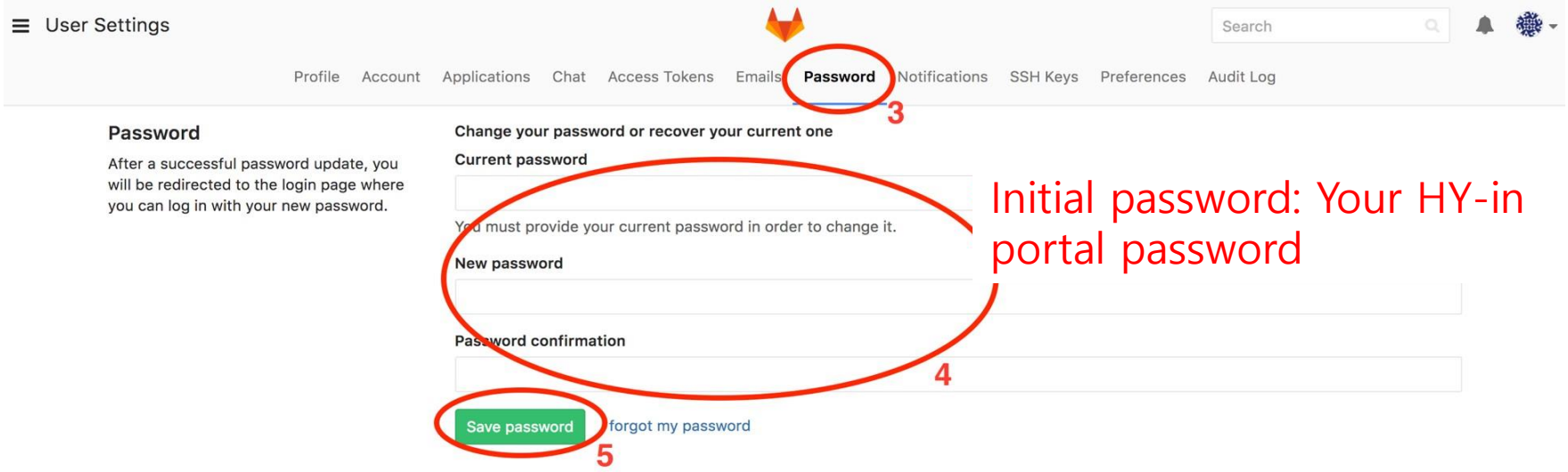
- Set up Password



HYU 한양대학교 HANYANG UNIVERSITY

Git

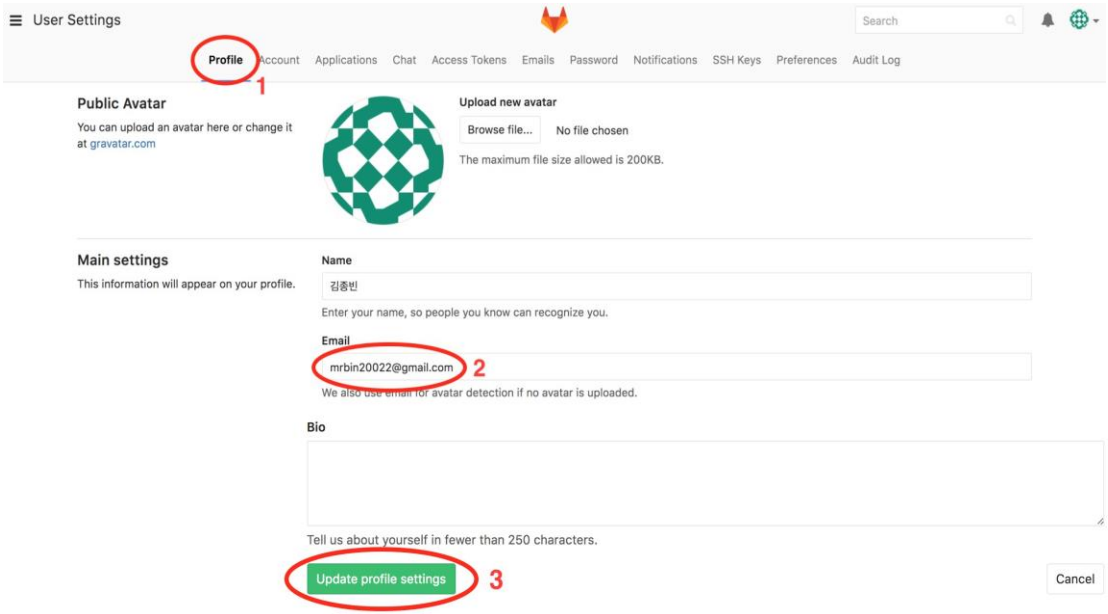
- Set up Password



HYU 한양대학교 HANYANG UNIVERSITY

Git

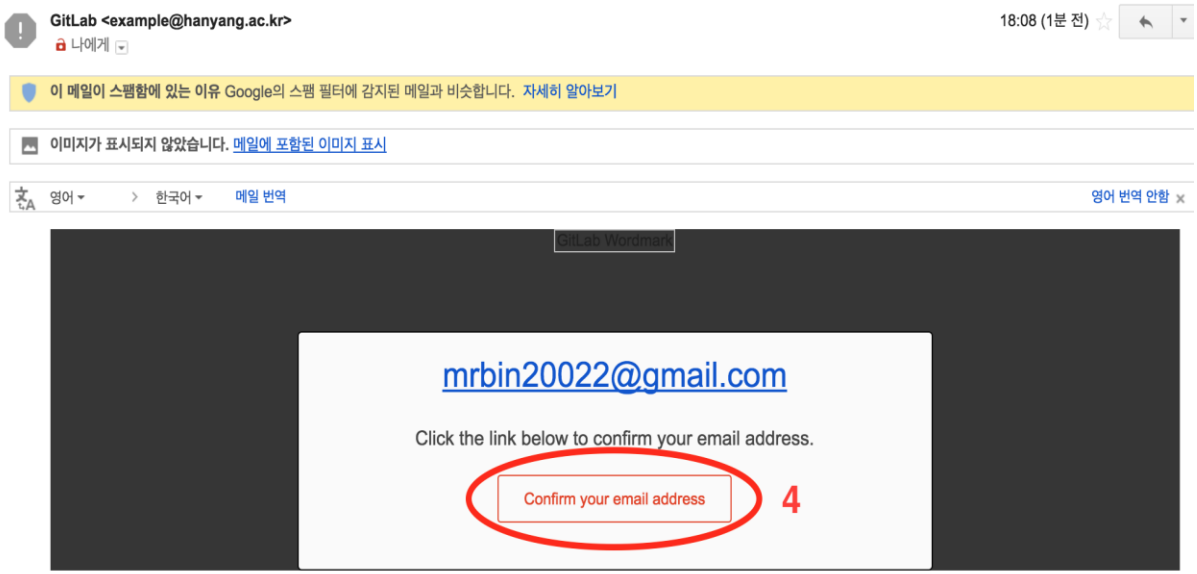
- Set up Email



HYU 한양대학교 HANYANG UNIVERSITY

Git

- Set up Email – Approve from changed email

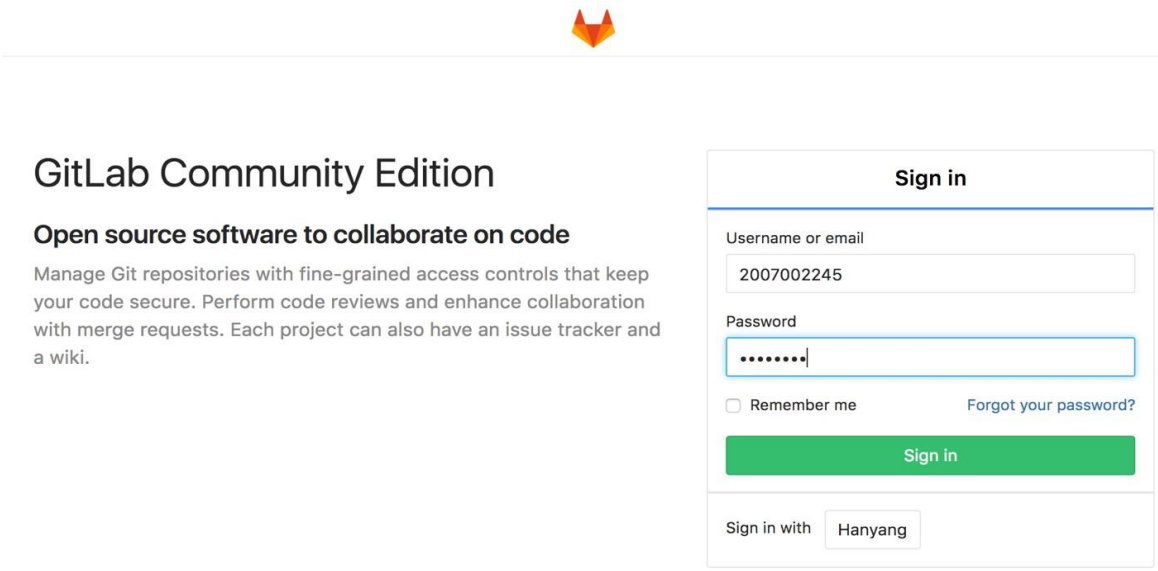


HYU 한양대학교 HANYANG UNIVERSITY

Git

- After this, you can sign in to hconnect with your student ID / email and the password you changed.

  (without using 'Sign in with Hanyang')



HYU 한양대학교 HANYANG UNIVERSITY

Lab Assignment 1

- Now, let’s start the lab assignment 1.
- Lab assignment 1 is just for practice, will not be included in the final grade.

- However, you need to complete and submit your answers **to figure out how to set up the environment** and **to create your "hconnect" site account** in advance.

- Check the assignment: Blackboard course home - Lab assignments - “LabAssignment1.pdf”
- Submit your files: Blackboard course home - Lab assignments - “Lab Assignment 1, 1” and " Lab Assignment 1, 2".

- You can leave the lab after submitting your files.