Computer Graphics

1 - Lab - Environment Setting

Yoonsang Lee
Hanyang University

Spring 2023

Outline

- Installing Python Interpreter
- Python Virtual Environment
- Installing Required Python Modules
- Running Python Interpreter

Install Python Interpreter

- Python **3.8** or later
  - https://www.python.org/downloads/

- Note that all submissions for assignments and projects must work in Python **3.8 with only NumPy, PyOpenGL, glfw, PyGLM installed.**
  - Do not use python features added after Python 3.8.

- You can use any OS that runs Python and OpenGL.

Install Python Packages (Modules)

- My recommendation for installing python modules is using **pip** (Python Package Index).
  - pip is already installed if you are using Python 2 >=2.7.9 or Python 3 >=3.4 downloaded from python.org.
  - However, if you use Python installed by default in the OS, you may need to install pip yourself.

- 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 and develop only one of them, if you do not use **python virtual environment**.

Python Virtual Environments

- Python virtual environment:
  - A self-contained directory tree that contains a Python installation for a particular version of Python with additional packages.
  - This allows you to isolate all your project's dependencies from the system and from 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=/opt/homebrew/bin/python3
source /opt/homebrew/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 course

- Windows

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

  - An example for <python_path>: "C:\Users\<your_id>\AppData\Local\Programs\Python\Python38\python.exe“
  - If your system does not know “mkvirtualenv”, you need to add python script directory (e.g. …\Python38\Script\) to system path.

- Ubuntu, MacOS

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

  - Replace <python3.x> with your python version
  - e.g. --python=python3.8

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 Required Modules

- We’ll use the following python modules in this course:
  - NumPy, PyOpenGL, glfw, PyGLM

- On Windows, Ubuntu, MacOS:
- After activating the "cg-course" environment,

```
$ workon cg-course
```

- Install the modules:

```
$ pip install numpy pyopengl glfw pyglm
```

Installation Troubleshooting

- If "glfw" is not properly imported with an error message such as "failed to load glfw library",
  - On Ubuntu,
    - Need to install "libglfw3" on your host environment:

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

  - On Windows,
    - Download glfw library for Windows from https://www.glfw.org/download.html. Copy “lib-vc2015/glfw3.dll” to "python-installation-directory/Lib/site-packages/glfw"

Installation Troubleshooting

- If you're experiencing other troubles while installing or importing those modules, doing a search using the error message will likely lead you to a solution.

- In that case, please let me know the problem situation and solution so that I can update the slides.

Running Python Interpreter 1

- **Interactive mode** (in `cg-course` virtual env)
  - In terminal (Ubuntu) or cmd (Windows),

```
workon cg-course  #if you've not activated the env yet
python
```

  - 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)**
  - In terminal (Ubuntu) or cmd (Windows),

```
workon cg-course  #if you've not activated the env yet
python test.py
```

  - In most cases, you will use this mode.

- You can write a Python source file using your favorite editor.
  - Vim, Sublime Text, Visual Studio Code, 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/

Time for Assignment

- No assignment today!

- But you must do the following items before the next lecture:
  - Ensure your laptop supports OpenGL 3.3 or higher.
  - Make sure python 3.8 or higher is installed on your laptop.
  - Install virtualenv and create an virtual environment for this course.
  - Install NumPy, PyOpenGL, glfw, PyGLM on that virtual environment.

Ensure Your Python Environment

- Start the python interpreter in the interactive mode and import numpy, OpenGL, glfw, glm.
- If no errors occur, the configuration is fine.

- An example of a well-configured environment:

```
$ python
Python 3.8.10 (default, Nov 14 2022, 12:59:47)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for
more information.
>>> import numpy
>>> import OpenGL
>>> import glfw
>>> import glm
>>>
```