K. K. Wagh Institute Of Engineering Education And Research

* User Manual *

PID 15

Brain Tumor Detection using CNN

Index

Python

Steps to install Python:

Step 1 − Select Version of Python to Install

Python has various versions available with differences between the syntax and working of different versions of the language. We need to choose the version which we want to use or need. There are different versions of Python 2 and Python 3 available.

Step 2 − Download Python Executable Installer

On the web browser, in the official site of python (www.python.org), move to the Download for Windows section.

All the available versions of Python will be listed. Select the version required by you and click on Download. Let's suppose, we chose the Python 3.9.1 version.

On clicking download, various available executable installers shall be visible with different operating system specifications. Choose the installer which suits your system operating system and download the installer. Let's suppose, we select the Windows installer(64 bits).

The download size is less than 30MB.

Step 3 − Run Executable Installer

We downloaded the Python 3.9.1 Windows 64 bit installer.

Run the installer. Make sure to select both the checkboxes at the bottom and then click Install New.

On clicking the Install Now, The installation process starts.

The installation process will take a few minutes to complete and once the installation is successful, the following screen is displayed.

Step 4 − Verify Python is installed on Windows

To ensure if Python is successfully installed on your system. Follow the given steps −

- Open the command prompt.
- Type ‘python’ and press enter.
- The version of the python which you have installed will be displayed if the python is successfully installed on your windows.

Step 5 − Verify Pip was installed

Pip is a powerful package management system for Python software packages. Thus, make sure that you have it installed.

To verify if pip was installed, follow the given steps −

- Open the command prompt.
- Enter pip –V to check if pip was installed.
- The following output appears if pip is installed successfully.

Steps to install Anaconda:

1. Download the Anaconda installer.
2. RECOMMENDED: Verify data integrity with SHA-256. For more information on hashes, see What about cryptographic hash verification?
3. Double click the installer to launch.
4. Click Next.
5. Read the licensing terms and click “I Agree”.
6. Select an install for “Just Me” unless you’re installing for all users (which requires Windows Administrator privileges) and click Next.
7. Select a destination folder to install Anaconda and click the Next button. See FAQ.
8. Choose whether to add Anaconda to your PATH environment variable. We recommend not adding Anaconda to the PATH environment variable, since this can interfere with other software. Instead, use Anaconda software by opening Anaconda Navigator or the Anaconda Prompt from the Start Menu.
9. Click the Install button. If you want to watch the packages Anaconda is installing, click Show Details.
10. Click the Next button.
11. After a successful installation you will see the “Thanks for installing Anaconda” dialog box.
12. If you wish to read more about Anaconda.org and how to get started with Anaconda, check the boxes “Anaconda Individual Edition Tutorial” and “Learn more about Anaconda”. Click the Finish button.
13. Verify your installation.

VS Code

Editor : VS Code

1. Download the Setup File:

a) VS code can be simply installed on Windows 10 with the setup file. So we need to download the setup file from the official website. https://code.visualstudio.com/ After opening the website, click on the **Download for Windows** button. This will download the VS code Setup Wizard on our system as an EXE file.

2. Run the VS code Setup Wizard

So the VS code Setup Wizard is downloaded successfully and we need to run it.

2.1 Close All Other Applications

At first, it recommends that we need to close all other applications before the VS code installation starts. It is not mandatory.

2.2 Accept the License Agreement

In this step, read the license agreement and choose whether we accept it or not. But the installation continues only after we accept the agreement.

2.3 Select the Installation Location

Choose the location in our system to install the VS Code. If we are not bothering about the location, go with the default location. But, at least 203.4 MB of free disk space is required in the selected location.

2.4 Placing the Shortcuts

As a default, the VS code shortcut will be placed in the Start Menu folder. We can change the destination or skip creating shortcuts.

2.5 Selecting Additional Tasks

We get a bunch of additional tasks to be performed before the installation begins. Choose tasks we prefer and continue.

2.6 Install VS code on Windows 10

So we have set up everything and the VS code can be installed on our Windows 10 system now.

2.7 Finish and Launch

After the successful installation, we can launch the VS code on our system.

Editor : Sublime Text

Sublime Text 3 can be downloaded from its official site sublimetext.com.
To install sublime text 3 on Windows, go through the following steps:

Step 1: Open the downloaded .exe file from the downloads folder and begin with the installation process.

Step 2: Select the desired location and click on the next button to start installation

Step 3: If you want Sublime Text 3 to appear in your right-click menu, then mark the checkbox and click on the Next button.

Step 4: Press the install button.

Step 5: Finish with the installation process.

Flask

Python Flask Windows Development Environment Setup

No more struggles with Windows Python development! I’ve found this is the best way to configure your dev environment.

This has made things much easier to get started and less of a headache overall.

We use Virtual Environment so we can test python code in encapsulated environments and to also avoid filling our base Python installation with a bunch of libraries we might use for only one project.

But Virtual Environments can be tricky if you don’t establish a good workflow. I’ll show you how to set-up your python environment from Scratch and then do a very simple workflow using Flask.

SETUP

4 Steps:

Install Python

Install Pip

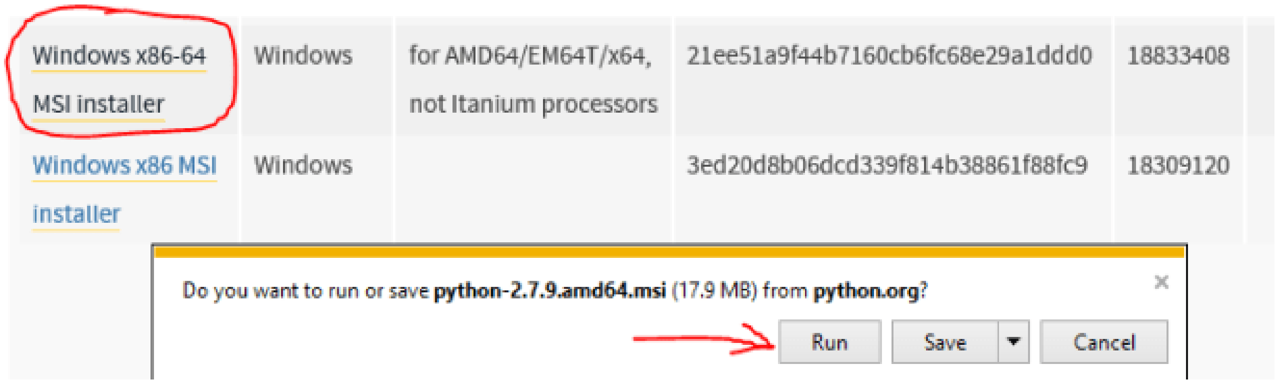
Install VirtualEnv

Install VirtualEnvWrapper-win

Install Python:

First Go to the Python Downloads Site.

As of March 2015 the download you want for a standard windows machine is Windows x86-64 MSI installer (The other download is for servers). Its circled here:

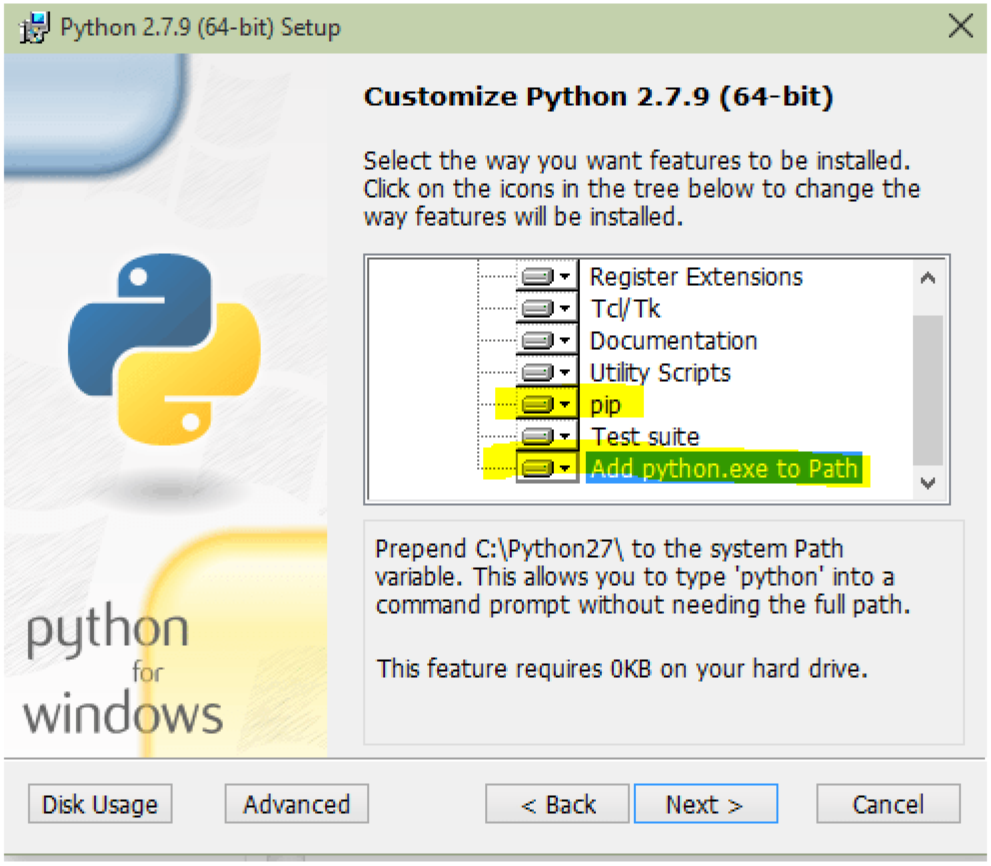


Windows x86-64 MSI installer	Windows	for AMD64/EM64T/x64, not Itanium processors	21ee51a9f44b7160cb6fc68e29a1ddd0	18833408
Windows x86 MSI installer	Windows		3ed20d8b06dcd339f814b38861f88fc9	18309120

Do you want to run or save **python-2.7.9.amd64.msi** (17.9 MB) from **python.org**?

Run the installer!

You’ll come across this page in the installer:



Customize Python 2.7.9 (64-bit)

Select the way you want features to be installed. Click on the icons in the tree below to change the way features will be installed.

- Register Extensions
- Tcl/Tk
- Documentation
- Utility Scripts
- pip
- Test suite
- Add python.exe to Path

Prepend C:\Python27\ to the system Path variable. This allows you to type 'python' into a command prompt without needing the full path.

This feature requires 0KB on your hard drive.

You'll want to scroll down and add it to the path. If you don't that's okay. You can add it later.

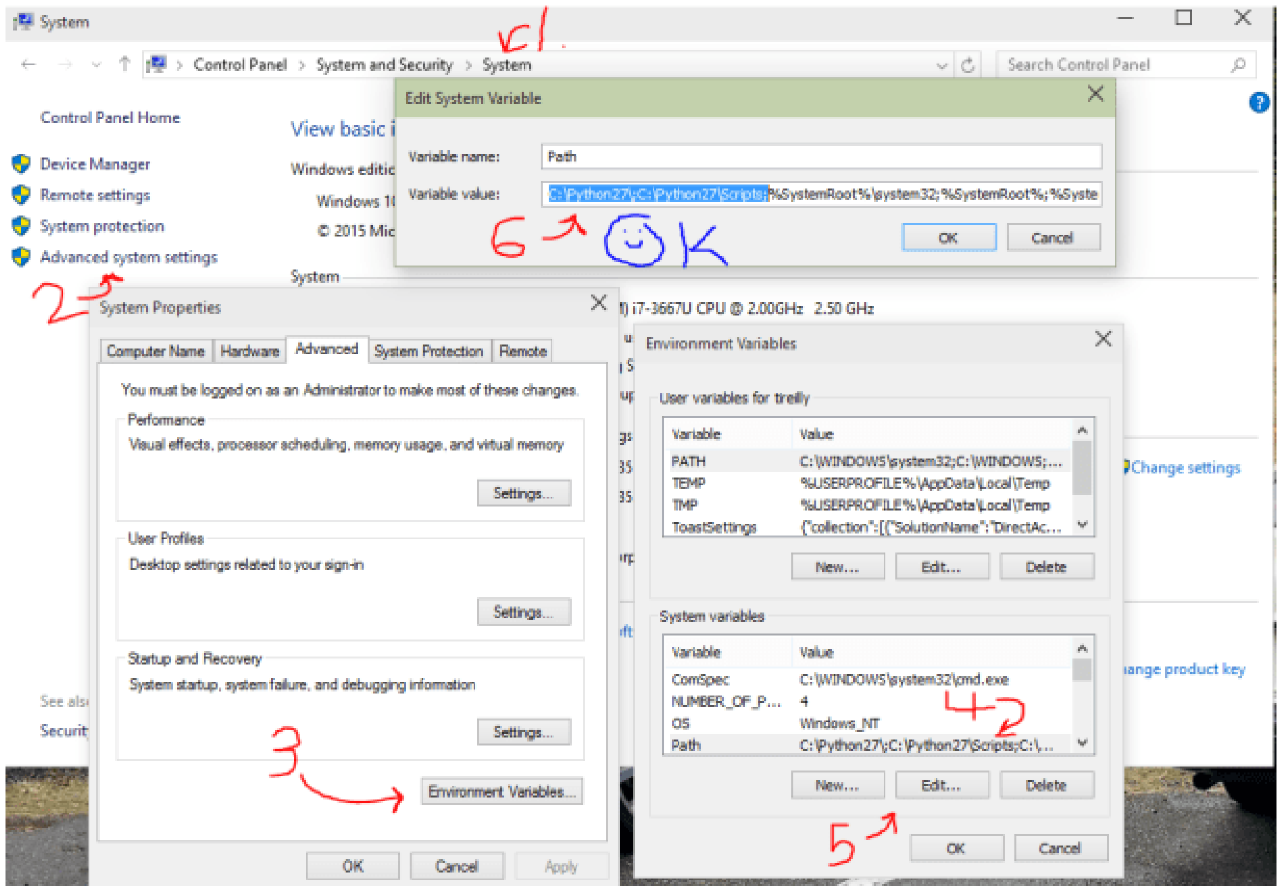
Adding Python to the PATH will allow you to call if from the command line.

After the installation is complete double check to make sure you see python in your PATH. You can find your path by opening your control panel -> System and Security -> System -> Advanced System Settings -> Environment Variables -> Selecting Path -> Edit ->

Now you're looking at your Path. Be Careful, if you delete or add to the path accidently you may break other programs.

You need to confirm that C:\Python27; and C:\Python27\Scripts; is part of your path.

If you do not see it in your path you can simply add it at the beginning or end of the variable value box. As you can see in the image below.



Edit System Variable

Variable name: Path

Variable value: C:\Python27\;C:\Python27\Scripts;%SystemRoot%\system32;%SystemRoot%;%Syste

System Properties

You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles

Desktop settings related to your sign-in

Startup and Recovery

System startup, system failure, and debugging information

Environment Variables

User variables for treilly

Variable	Value
PATH	C:\WINDOWS\system32;C:\WINDOWS;...
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp
ToastSettings	{"collection":[{"SolutionName":"DirectAc...

System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
NUMBER_OF_P...	4
OS	Windows_NT
Path	C:\Python27\;C:\Python27\Scripts;C:\...

Install Pip:

As of Python Version 2.7.9 Pip is installed automatically and will be available in your Scripts folder.

If you install a later version of Python I would recommend installing it according to this helpful stackoverflow post.

Pip is a Package manager for python which we will use to load in modules/libraries into our environments.

An example of one of these libraries is VirtualEnv which will help us keep our environments clean from other Libraries. This sounds really confusing but as you start using it you'll begin to understand how valuable this encapsulation of modules/libraries can be.

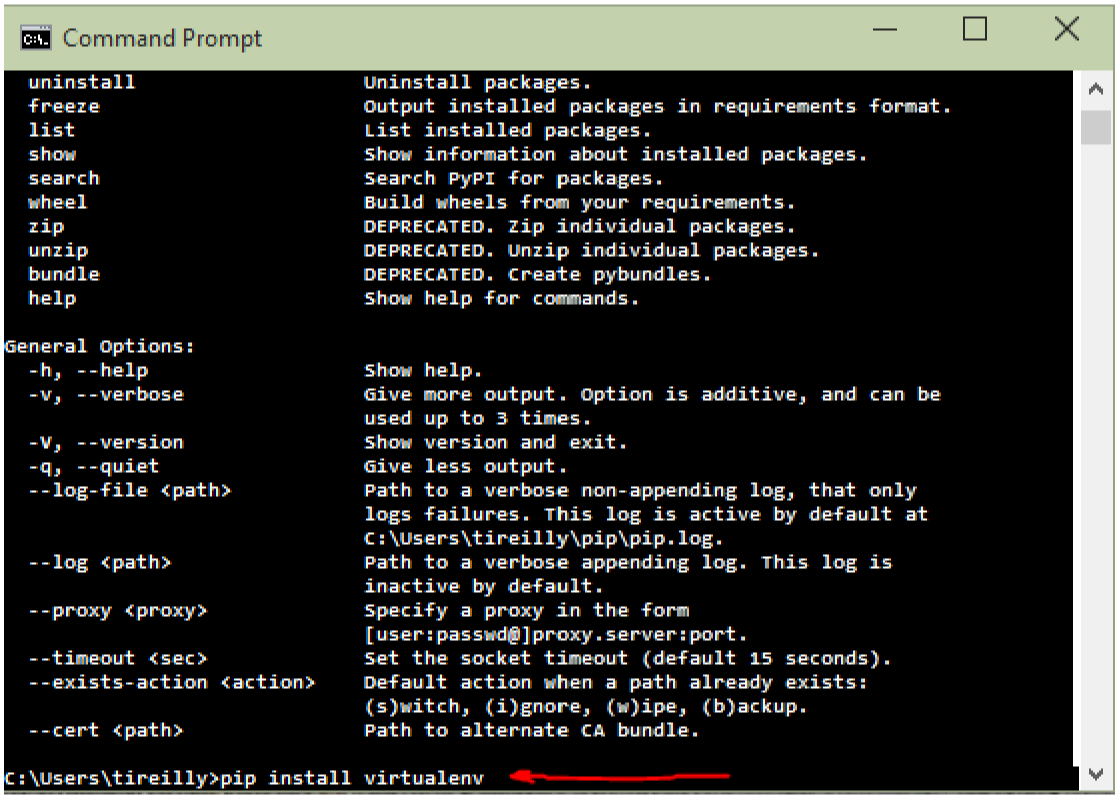
To test that Pip is installed open a command prompt (win+r->'cmd'->Enter) and try 'pip help'

You should see a list of available commands including install, which we'll use for the next part:

Install virtualenv:

Now that you have pip installed and a command prompt open installing virtualenv to our root Python installation is as easy as typing 'pip install virtualenv'

Like so:



```
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in requirements format.
  list                        List installed packages.
  show                        Show information about installed packages.
  search                      Search PyPI for packages.
  wheel                       Build wheels from your requirements.
  zip                         DEPRECATED. Zip individual packages.
  unzip                       DEPRECATED. Unzip individual packages.
  bundle                      DEPRECATED. Create pybundles.
  help                        Show help for commands.

General Options:
  -h, --help                  Show help.
  -v, --verbose               Give more output. Option is additive, and can be
                              used up to 3 times.
  -V, --version               Show version and exit.
  -q, --quiet                 Give less output.
  --log-file <path>           Path to a verbose non-appending log, that only
                              logs failures. This log is active by default at
                              C:\Users\tireilly\pip\pip.log.
  --log <path>                Path to a verbose appending log. This log is
                              inactive by default.
  --proxy <proxy>             Specify a proxy in the form
                              [user:passwd@]proxy.server:port.
  --timeout <sec>             Set the socket timeout (default 15 seconds).
  --exists-action <action>    Default action when a path already exists:
                              (s)witch, (i)gnore, (w)ipe, (b)ackup.
  --cert <path>               Path to alternate CA bundle.

C:\Users\tireilly>pip install virtualenv
```

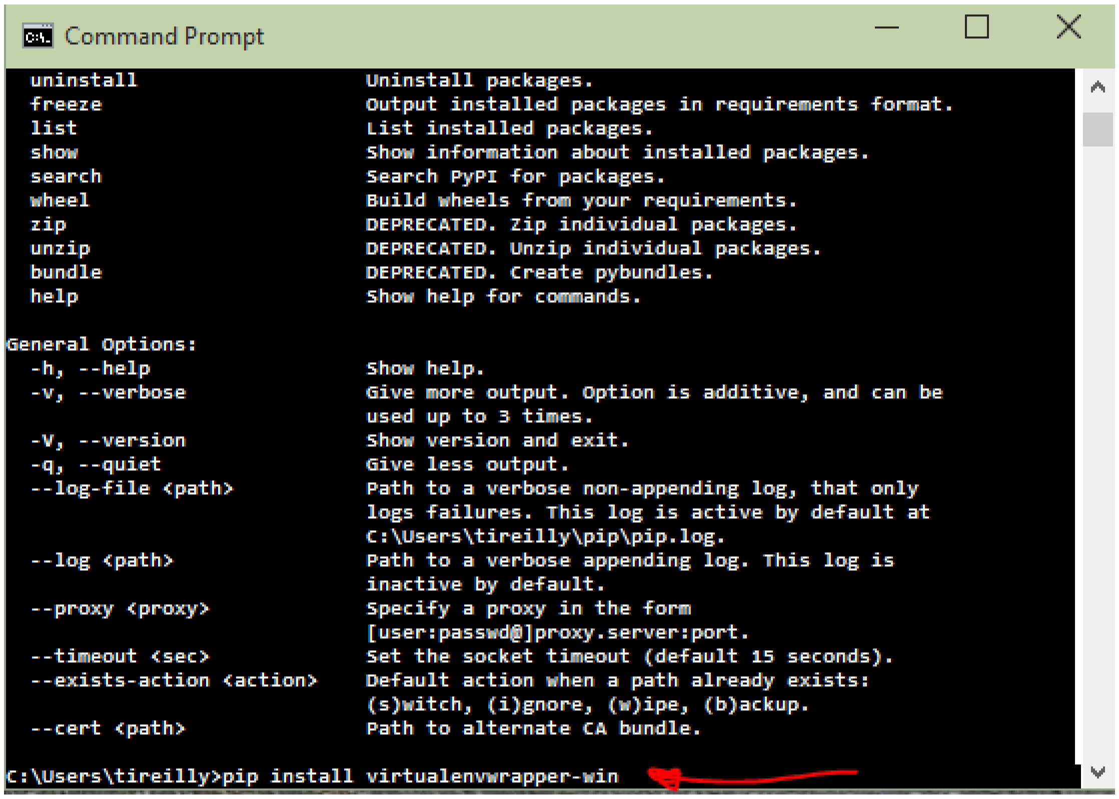
Now we have virtualenv installed which will make it possible to create individual environments to test our code in. But managing all these environments can become cumbersome. So we'll pip install another helpful package…

Install virtualenvwrapper-win:

This is the kit and caboodle of this guide.

Just as before we'll use pip to install virtualenvwrapper-win. 'pip install virtualenvwrapper-win'

Like so:



```
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in requirements format.
  list                        List installed packages.
  show                        Show information about installed packages.
  search                      Search PyPI for packages.
  wheel                       Build wheels from your requirements.
  zip                         DEPRECATED. Zip individual packages.
  unzip                       DEPRECATED. Unzip individual packages.
  bundle                      DEPRECATED. Create pybundles.
  help                        Show help for commands.

General Options:
  -h, --help                  Show help.
  -v, --verbose               Give more output. Option is additive, and can be
                              used up to 3 times.
  -V, --version               Show version and exit.
  -q, --quiet                 Give less output.
  --log-file <path>           Path to a verbose non-appending log, that only
                              logs failures. This log is active by default at
                              C:\Users\tireilly\pip\pip.log.
  --log <path>                Path to a verbose appending log. This log is
                              inactive by default.
  --proxy <proxy>             Specify a proxy in the form
                              [user:passwd@]proxy.server:port.
  --timeout <sec>             Set the socket timeout (default 15 seconds).
  --exists-action <action>    Default action when a path already exists:
                              (s)witch, (i)gnore, (w)ipe, (b)ackup.
  --cert <path>               Path to alternate CA bundle.

C:\Users\tireilly>pip install virtualenvwrapper-win
```

Excellent! Now we have everything we need to start building software using python! Now I’ll show you how buttery smooth it is to use these awesome tools!

USAGE

7 Steps:

Make a Virtual Environment

Connect our project with our Environment

Set Project Directory

Deactivate

Workon

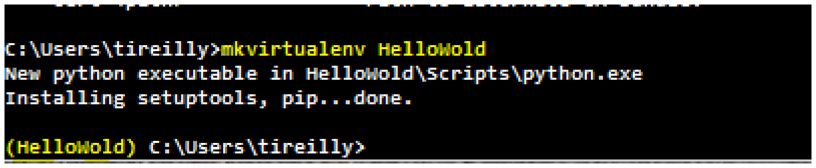
Pip Install

Flask!

Make a Virtual Environemt:

Lets call it HelloWold. All we do in a command prompt is enter ‘mkvirtualenv HelloWold’

This will create a folder with python.exe, pip, and setuptools all ready to go in its own little environment. It will also activate the Virtual Environment which is indicated with the (HelloWold) on the left side of the prompt.



```
C:\Users\tireilly>mkvirtualenv HelloWold
New python executable in HelloWold\Scripts\python.exe
Installing setuptools, pip...done.

(HelloWold) C:\Users\tireilly>
```

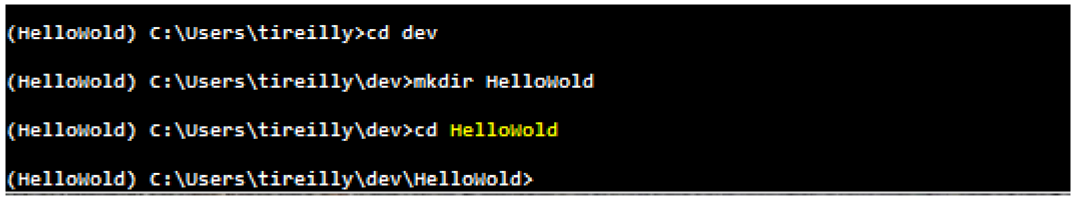
Anything we install now will be specific to this project. And available to the projects we connect to this environment.

Connect our project with our Environment:

Now we want our code to use this environment to install packages and run/test code.

First lets create a directory with the same name as our virtual environment in our preferred development folder. In this case mine is ‘dev’

See here:



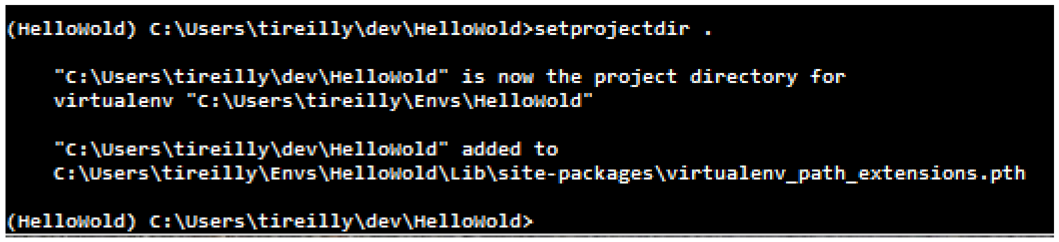
```
(HelloWold) C:\Users\tireilly>cd dev

(HelloWold) C:\Users\tireilly\dev>mkdir HelloWold

(HelloWold) C:\Users\tireilly\dev>cd HelloWold

(HelloWold) C:\Users\tireilly\dev\HelloWold>
```

HelloWold will be the root folder of our first project!

Set Project Directory:

Now to bind our virtualenv with our current working directory we simply enter ‘setprojectdir .’

Like so:



```
(HelloWold) C:\Users\tireilly\dev\HelloWold>setprojectdir .

    "C:\Users\tireilly\dev\HelloWold" is now the project directory for
    virtualenv "C:\Users\tireilly\Envs\HelloWold"

    "C:\Users\tireilly\dev\HelloWold" added to
    C:\Users\tireilly\Envs\HelloWold\Lib\site-packages\virtualenv_path_extensions.pth

(HelloWold) C:\Users\tireilly\dev\HelloWold>
```

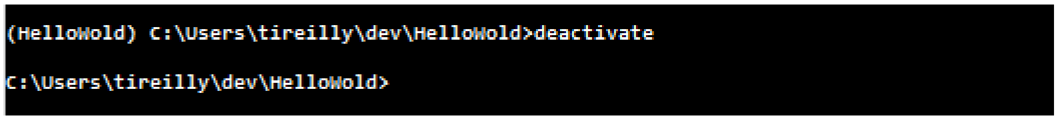
Now next time we activate this environment we will automatically move into this directory!

Buttery smooth.

Deactivate:

Let say you're content with the work you've contributed to this project and you want to move onto something else in the command line. Simply type 'deactivate' to deactivate your environment.

Like so:



```
(HelloWold) C:\Users\tireilly\dev\HelloWold>deactivate

C:\Users\tireilly\dev\HelloWold>
```

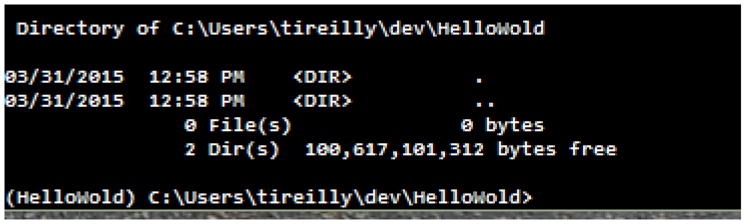
Notice how the parenthesis disappear.

You don't have to deactivate your environment. Closing your command prompt will deactivate it for you. As long as the parenthesis are not there you will not be affecting your environment. But you will be able to impact your root python installation.

Workon:

Now you've got some work to do. Open up the command prompt and type 'workon HelloWold' to activate the environment and move into your root project folder.

Like so:



```
 Directory of C:\Users\tireilly\dev\HelloWold

03/31/2015  12:58 PM    <DIR>          .
03/31/2015  12:58 PM    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  100,617,101,312 bytes free

(HelloWold) C:\Users\tireilly\dev\HelloWold>
```

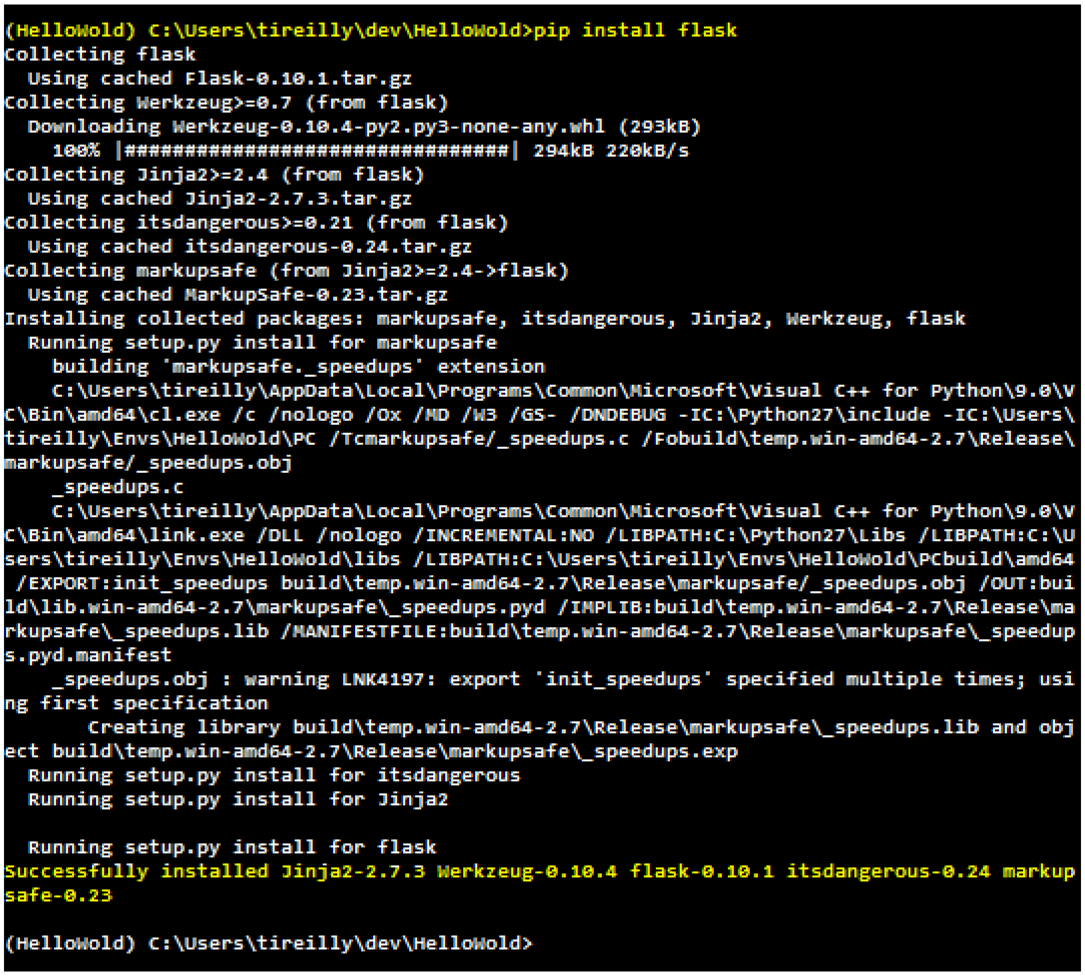
Pretty sweet! Lets get working.

Pip Install:

To use flask we need to install the packages and to do that we can use pip to install it into our HelloWold virtual environment.

Make sure (HelloWold) is to the left of your prompt and enter ‘pip install flask’

Like so:



```
(HelloWold) C:\Users\tireilly\dev\HelloWold>pip install flask
Collecting flask
  Using cached Flask-0.10.1.tar.gz
Collecting Werkzeug>=0.7 (from flask)
  Downloading Werkzeug-0.10.4-py2.py3-none-any.whl (293kB)
    100% |################################| 294kB 220kB/s
Collecting Jinja2>=2.4 (from flask)
  Using cached Jinja2-2.7.3.tar.gz
Collecting itsdangerous>=0.21 (from flask)
  Using cached itsdangerous-0.24.tar.gz
Collecting markupsafe (from Jinja2>=2.4->flask)
  Using cached MarkupSafe-0.23.tar.gz
Installing collected packages: markupsafe, itsdangerous, Jinja2, Werkzeug, flask
  Running setup.py install for markupsafe
    building 'markupsafe._speedups' extension
    C:\Users\tireilly\AppData\Local\Programs\Common\Microsoft\Visual C++ for Python\9.0\V
C\Bin\amd64\cl.exe /c /nologo /Ox /MD /W3 /GS- /DNDEBUG -IC:\Python27\include -IC:\Users\
tireilly\Envs\HelloWold\PC /Tcmarkupsafe/_speedups.c /Fobuild\temp.win-amd64-2.7\Release\
markupsafe/_speedups.obj
    _speedups.c
    C:\Users\tireilly\AppData\Local\Programs\Common\Microsoft\Visual C++ for Python\9.0\V
C\Bin\amd64\link.exe /DLL /nologo /INCREMENTAL:NO /LIBPATH:C:\Python27\Libs /LIBPATH:C:\U
sers\tireilly\Envs\HelloWold\libs /LIBPATH:C:\Users\tireilly\Envs\HelloWold\PCbuild\amd64
 /EXPORT:init_speedups build\temp.win-amd64-2.7\Release\markupsafe/_speedups.obj /OUT:bui
ld\lib.win-amd64-2.7\markupsafe\_speedups.pyd /IMPLIB:build\temp.win-amd64-2.7\Release\ma
rkupsafe\_speedups.lib /MANIFESTFILE:build\temp.win-amd64-2.7\Release\markupsafe\_speedup
s.pyd.manifest
    _speedups.obj : warning LNK4197: export 'init_speedups' specified multiple times; usi
ng first specification
       Creating library build\temp.win-amd64-2.7\Release\markupsafe\_speedups.lib and obj
ect build\temp.win-amd64-2.7\Release\markupsafe\_speedups.exp
  Running setup.py install for itsdangerous
  Running setup.py install for Jinja2

  Running setup.py install for flask
Successfully installed Jinja2-2.7.3 Werkzeug-0.10.4 flask-0.10.1 itsdangerous-0.24 markup
safe-0.23

(HelloWold) C:\Users\tireilly\dev\HelloWold>
```

This will bring in all the tools required to write your first web server!

Flask:

Now that you have flask installed in your virtual environment you can start coding!

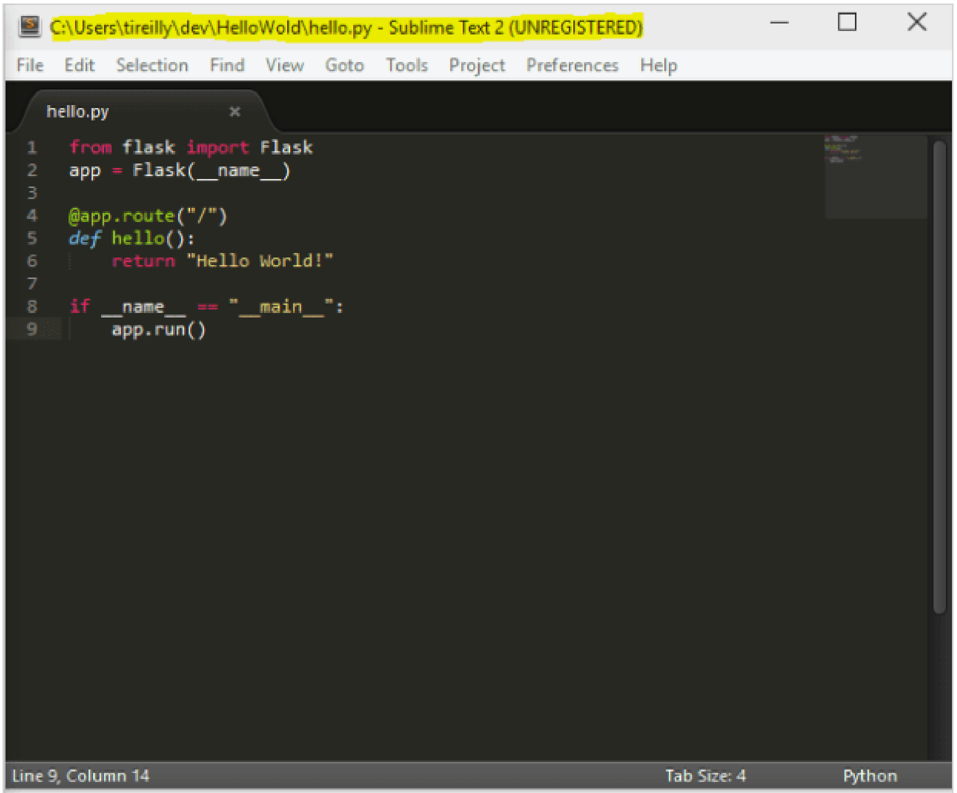
Open up your favorite text editor and create a new file called hello.py and save it in your HelloWold directory.

I’ve simply taken the sample code from Flask’s website to create a very basic ‘Hello World!’ server.

I’ve named the file hello.py.

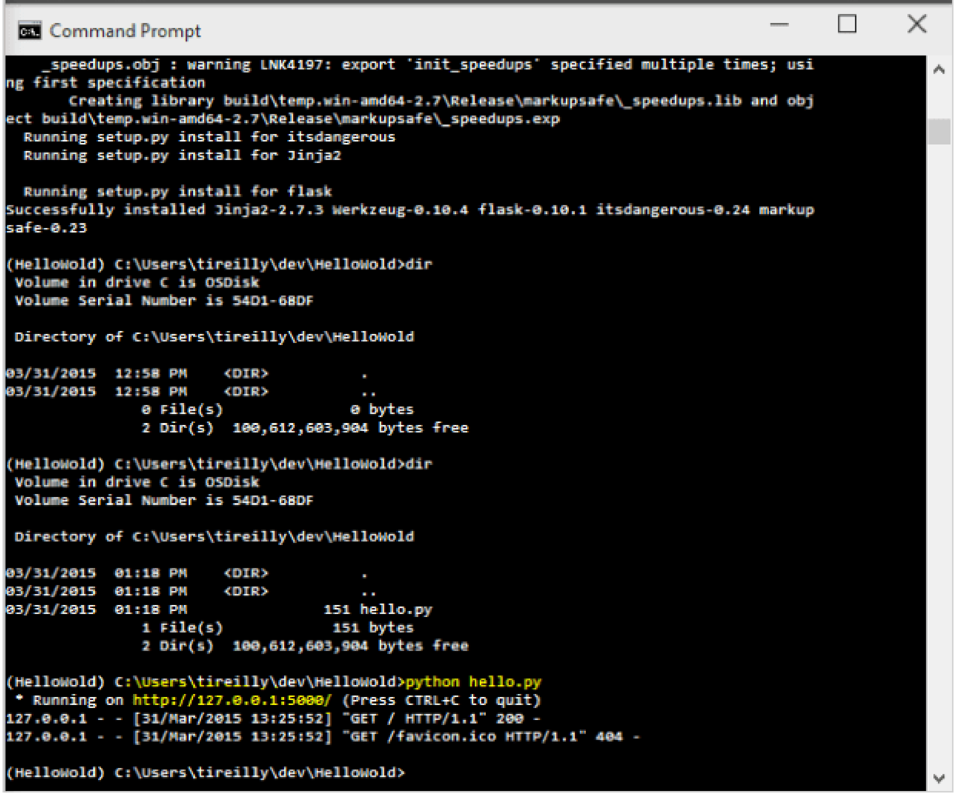
Once the code is in place I can start the server using ‘python hello.py’ this will run the python instance from your virtual environment that has flask.

See here:



C:\Users\tireilly\dev\HelloWold\hello.py - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
hello.py
Line 9, Column 14
Tab Size: 4
Python

```python
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run()
```



```
    _speedups.obj : warning LNK4197: export 'init_speedups' specified multiple times; usi
ng first specification
       Creating library build\temp.win-amd64-2.7\Release\markupsafe\_speedups.lib and obj
ect build\temp.win-amd64-2.7\Release\markupsafe\_speedups.exp
  Running setup.py install for itsdangerous
  Running setup.py install for Jinja2

  Running setup.py install for flask
Successfully installed Jinja2-2.7.3 Werkzeug-0.10.4 flask-0.10.1 itsdangerous-0.24 markup
safe-0.23

(HelloWold) C:\Users\tireilly\dev\HelloWold>dir
 Volume in drive C is OSDisk
 Volume Serial Number is 54D1-68DF

 Directory of C:\Users\tireilly\dev\HelloWold

03/31/2015  12:58 PM    <DIR>          .
03/31/2015  12:58 PM    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  100,612,603,904 bytes free

(HelloWold) C:\Users\tireilly\dev\HelloWold>dir
 Volume in drive C is OSDisk
 Volume Serial Number is 54D1-68DF

 Directory of C:\Users\tireilly\dev\HelloWold

03/31/2015  01:18 PM    <DIR>          .
03/31/2015  01:18 PM    <DIR>          ..
03/31/2015  01:18 PM               151 hello.py
               1 File(s)            151 bytes
               2 Dir(s)  100,612,603,904 bytes free

(HelloWold) C:\Users\tireilly\dev\HelloWold>python hello.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [31/Mar/2015 13:25:52] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [31/Mar/2015 13:25:52] "GET /favicon.ico HTTP/1.1" 404 -

(HelloWold) C:\Users\tireilly\dev\HelloWold>
```

You can now navigate with your browser to http://127.0.0.1:5000/ and see your new site!

Sweet. You have everything you need to start working through tutorials on Flask without worrying about gunking up your Python installations.

Miscellaneous

Install TensorFlow with pip

TensorFlow 2 packages are available

- tensorflow —Latest stable release with CPU and GPU support *(Ubuntu and Windows)*
- tf-nightly —Preview build *(unstable)* . Ubuntu and Windows include GPU support .

1. Install the Python development environment on your system

1. Check if your Python environment is already configured:

```
python3 --version
pip3 --version
```

2. If these packages are already installed, skip to the next step.

Otherwise, install Python , the pip package manager , and venv :

2. Create a virtual environment (recommended)

1. Python virtual environments are used to isolate package installation from the system.

2. Create a new virtual environment by choosing a Python interpreter and making a .\venv directory to hold it:

```
python -m venv --system-site-packages .\venv
```

3. Activate the virtual environment:

```
.\venv\Scripts\activate
```

4. Install packages within a virtual environment without affecting the host system setup. Start by upgrading pip :

```
pip install --upgrade pip
```

```
pip list  # show packages installed within the virtual environment
```

5. And to exit the virtual environment later:

```
deactivate  # don't exit until you're done using TensorFlow
```

3. Install the TensorFlow pip package

1. Choose one of the following TensorFlow packages to install from PyPI :

- tensorflow —Latest stable release with CPU and GPU support *(Ubuntu and Windows)* .
- tf-nightly —Preview build *(unstable)* . Ubuntu and Windows include GPU support .
- tensorflow==1.15 —The final version of TensorFlow 1.x.

2. Command to install Tensorflow:

```
pip install --upgrade tensorflow
```

3. Verify the install:

```
python -c "import tensorflow as tf;print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

Success: If a tensor is returned, you've installed TensorFlow successfully.