Installing Python

This is guide for those trying to install Python for the first time on their local machine. **Note:** Installing python on your own machine is ***optional*** for this workshop. In the readme there are buttons that will open up a cloud version of the workshop materials in a binderhub or google collab instanceBecause it is currently the most popular programming package out there, there are a myriad different ways of getting Python set up on your machine, as well as many different programs, called editors or Integrated Development Environments, to write and run Python programs. Perusing your way through all the different ways to Python can be a painful process. That is why we will show you the two easiest (in my opinion anyway) ways to install python on your machine, along with managing all of the open source packages that make life as a Pythonista much easier!

In order to use Python on your computer, you need to install python, a package manager for open source packages, and optionally, a editor (a program that makes it easier to write code) or Integrated Development Environment (or IDE, a program that makes it easier to write code, as well as run that code you have created). In this course we strongly recommend you use a python packaging manager like Anaconda, or Enthought to manage the version of Python. This is because the dependencies that a program you write can be quite fragile, and if you update one version of a python package, it may end up breaking your code by not being able to play well with the other python packages installed. So use a Python dependency manager like Anaconda will help by doing a couple of things:

1. Making sure that all of the dependencies will work with each other
2. Allowing you to create multiple virtual environments (aka versions of python, and it's packages), where you can protect the changes you make in one project without breaking things in another project
3. allow for saving and sharing the exact set of packages you used to do the cool thing you are going to do with Python.

We show you how to use Anaconda here, as it is more popular in the research field, and the one that we use most often. It has the added benefit that it can also manage related software such as R, and some other related programming languages to make it even more handy! As for doing the actually writing and running of the code, we will use jupyter notebooks.

The options we will show you in this document are:

- Anaconda - The easiest option. Installs base Python as well as a range of commonly used packages. Requires approximately 400mb of space on hard-drive.
- Miniconda - Slightly harder option. Installs base Python, and allows the user to control which packages to install. Good for people with limited space.
- pip - Slighter harder still option. The in-built Python installation manager.

As this introductory workshop is designed to be paired with the *VALA tech camp introduction to python workshop*, this installation guide will also cover the package dependencies required for the workshop,

Installing Python with Anaconda

Anaconda is a Python distribution platform which allows you to install and manage up to 720 different Python packages through the Anaconda installer, `conda`. If you have at least 400MB of free space on your laptop, then I would recommend using this, as it will automatically install many of the packages that you will find useful for your research and future coding.

Just navigate to the Anaconda home page and follow the instructions to download Python version 3.6 for your Windows, Mac or Linux device.

For windows devices - if you are not sure and are using windows 7, 8, 9 or 10, then download the 64-bit exe installer.

Anaconda Installation

Please ensure that you allow Anaconda to register it's version as Python as the default on your device. This means that, if you already have Python installed (as MacOS does natively), you are allowing Anaconda to use it's own version of Python - and the associated packages - when you are coding. If you change this option you will run into errors when trying to use your packages, and in particular, when trying to run the jupyter notebook required for this workshop.

Installing Python Through Miniconda

This is the better option for people with limited memory left on their device. This option will install base Python onto your device, and allows you to install specific packages according to your requirements.

Windows Installation

1. Download the `python 3 exe installer` from https://conda.io/miniconda.html (If you are not sure and you are using windows 7, 8, 9 or 10, then download the 64-bit exe installer)

2. Run the exe installer and install using default choices. By clicking next. (The installation might take a few minutes if the computer is slow, you can click "Show Details" to see the installation progress.

Uninstalling miniconda on

7. If you need to uninstall miniconda for any reason, you can do this through "Add or remove Program" in the control panel, by removing "Python 3.6(Miniconda)"

OS X Setup

1. In your browser download the Python 3.6 Miniconda installer for OS X from https://conda.io/miniconda.html, then in your terminal window type the following:

```
bash Miniconda3-latest-MacOSX-x86_64.sh
```

2. Follow the prompts on the installer screens. If unsure about any setting, simply accept the defaults as they all can be changed later. After installation is complete, close your terminal window.

3. Re-open your terminal window, and type `conda --version` into the command terminal to test that miniconda has been installed. Conda should respond with the version number that you have installed, like: conda 3.11.0

NOTE: If you see an error message, check to see that you are logged into the same user account that you used to install Anaconda or Miniconda, and that you have closed and re-opened the terminal window after installing it.

4. Type `conda install jupyter numpy pandas matplotlib bokeh` and hit `ENTER` key. When asked do you want to proceed, type `y` and hit `ENTER`. If you wish to install more packages in the future, you just need to use this `conda install <package name>` from your command line.
5. To view a list of packages and versions installed, or to confirm that a package has been added or removed, type `conda list`. Confirm that jupyter, numpy and matplotlib have been installed
6. The workshop will be using jupyter notebooks. To see how you can launch this, go to the Launching Jupyter Notebook section below.

Uninstalling miniconda on MacOS

7. If you need to uninstall miniconda for any reason, open a terminal window and remove the entire miniconda install directory by typing: `rm -rf ~/miniconda`

You may also edit ~/.bash_profile and remove the miniconda directory from your PATH environment variable, and remove the hidden .condarc file and .conda and .continuum directories which may have been created in the home directory with: `rm -rf ~/.condarc ~/.conda ~/.continuum`

Linux Installation

1. In your browser download the Python 3.6 Miniconda installer for OS X from https://conda.io/miniconda.html, then in your terminal window type the following:

`bash Miniconda3-latest-Linux-x86_64.sh`

2. Follow the prompts on the installer screens. If unsure about any setting, simply accept the defaults as they all can be changed later. After installation is complete, close your terminal window.
3. Close and re-open your terminal window. Type `conda --version` into the command terminal to test that miniconda has been installed. Conda should respond with the version number that you have installed, like: conda 3.11.0
4. Type `conda install jupyter numpy pandas matplotlib bokeh` and hit `ENTER` key. When asked do you want to proceed, type `y` and hit `ENTER`. If you wish to install more packages in the future, you just need to use this `conda install <package name>` from your command line.
5. To view a list of packages and versions installed, or to confirm that a package has been added or removed, type `conda list`. Confirm that jupyter, numpy and matplotlib have been installed
6. The workshop will be using jupyter notebooks. To see how you can launch this, go to the Launching Jupyter Notebook section below.

Uninstalling miniconda on Linux

7. If you need to uninstall miniconda for any reason, open a terminal window and remove the entire miniconda install directory: `rm -rf ~/miniconda`.

You may also edit `~/.bash_profile` and remove the miniconda directory from your PATH environment variable, and remove the hidden .condarc file and .conda and .continuum directories which may have been created in the home directory with `rm -rf ~/.condarc ~/.conda ~/.continuum`.

Setting up the virtual environment and installing packages

We will go ahead and introduce a best practice and python programming from the beginning. Like we mentioned above, the true power of python is in the thousands of open source libraries (also known as packages) that often do almost exaclty what you would like to do in one line of code and are constantly being updated to fix bugs and add new features. As you python more and more, you may find that you have different projects that require different sets of packages and maybe even an older or newer version. That's why we suggest creating a "virtual environment" for each project you work on, where you can manage the version of python and all the packages independent of every other project, and the base environment created when you install anaconda. Now there is a program called anaconda navigator where you can create virtual environments and install python packages (see here to learn how to use it), but we will show you on the command line way of doing things, which is faster, and when you graduate to Pythonista cloud school, will be the way you do things when working with big data!

If you are in windows

find and open a program called `Anaconda Prompt` from the Windows Start Menu.

This will open a new window with a black background (it might keep blinking if the computer is slow).

If you are on a MacOS or Linux machine

Open your terminal

Create a virtual environment:

RIght now, on the left hand side of your terminal entry, you should see `(base)` written. This is showing us that Anaconda is working and that we are in the base environment, something that you shouldn't really change in order to prevent other environments from breaking. Therefore, the first thing you need to do is create a virtual environment using the conda create command by specifying a name:

```
conda create -n VALA_tech_camp python=3.9
```

In this command we are running the command, `conda create`, and adding a name argument flag (-n) followed be the name we chose for this virtual environment (can be any text, without spaces) and at the end, we specified the version of python we want installed into this environment (this is optional by the way, and will default to all the packages and versions that are in the base environment). This command will ask you want to proceed, press y then enter.

activate the environment:

now that we have created the environment, we have to acdtivate it in our anaconda prompt/ terminal using the following commands:

if a windows user

```
activate VALA_tech_camp
```

if a linux or MacOS user:

```
conda activate VALA_tech_camp
```

You should see that now on the left hand side, there is a `(VALA_tech_camp)` written, showing us that this terminal is in our new virtual environment, and any conda commands to change things will only happen to this environment.

install packages

Now that our environment is activated, we can start customizing this python install. to install the packages needed for our workshop enter the following command into the terminal and press enter:

```
conda install jupyter
```

after you hit `ENTER`, like most anaconda commands where you are changing something, it will ask you to type `y` and `ENTER` again to confirm.

After that you need to install pip, which allows us to install Python packages outside of anaconda:

```
conda install pip
```

and when you are done with that we will install packages specified in the requirements file:

```
pip install -r
```

Don't close your termal yet, because we can use it to start a jupyter notebook

Launching the Jupyter Notebook

For many of these sessions, we're going to be working in the Jupyter notebook. Jupyter can be launched from the anaconda navigator program, but we will teach you how to launch it from your anaconda prompt or terminal in this section.

After opening the command terminal/ anaconda prompt, you should see a black screen and a `>` prompt after a file path, which is usually your home directory (`C:\users\{yourUserName}` in windows and

`/Users/{yourUserName}` in MacOS `/home/{yourUserName`} in ubuntu). When Jupyter launches, it sets its "home" folder to be the folder you are in when you launch the notebook, and it can access any sub-folders and files within that location. It cannot however, look outside of the folder where you start it!. This means it's in our best interest to launch from an area where we'll be able to access *all* of the files we'll require. This is usually in your user folder.

Therefore, in your terminal, use the cd command to navigate to your home folder, if you aren't already there:
Windows

```
`cd C:/Users/{YourUserName}`
```

MacOS

```
`cd /Users/{yourUserName}`
```

Linux

```
`cd /home/{yourUserName`
```

Once there, type in the command:

```
jupyter notebook
```

to launch Jupyter. command terminal

You should then see the terminal generate a bunch of text, and it should then open the jupyter home page within a browser window. This should be within whichever browser application is currently set as your default. You can change this in your settings.

(MacOS/Linux) If the terminal does not automatically open within your browser, you can copy the `http://localhost:888...etc` link into a new browser tab.

This should open to the Jupyter home folder, which will look something like this

Jupyter Home

You can click on the blue links inside the main console to navigate through your folders and files. To the top-right of your files box are some command buttons. These allow you to do things like create new folders, new text (*.txt) files, or a new iPython notebook. We will be writing and running our code from inside these *.ipynb files.

When you create a new *.ipynb notebook, you'll get something that looks like this, except empty:

Notebook

Play around with the notebook, and have a look to see what you can do. If you'd like a more guided tour/playbox, you can find one here

pip Installation for Advanced Users

Installing Python

If you already have Python and pip installed, go to the next section.

If you do not already have Python or pip installed:
[Note: MacOS has Python installed as part of the operating system. To use Python for anything else however, you will need to download and install another copy]

1. Download and install Python 3.6 or Python 2.7 from https://www.python.org/downloads/. Python version 3.6 is preferred.

2. **pip** should automatically be included with your installation if using Python 3.4 or greater, or Python 2.7.9 of greater. Now follow the instructions in the next section to install Jupyter and the other required Python packages.

Already have Python/pip installed

1. Ensure that you have the latest pip; older versions may have trouble with some dependencies. In your command prompt (Windows) or command terminal (MacOS/Linux), type: `pip3 install --upgrade pip`
   (Use `pip` instead of `pip3` if using legacy Python 2.x)

2. Install the Jupyter Notebook, Pandas, numpy, matplotlib and bokeh libraries using: `pip3 install jupyter numpy pandas matplotlib bokeh`
   (Use `pip` instead of `pip3` if using legacy Python 2).
   Close and re-open your terminal window.
   Similarly, if you wish to install other packages in the future, you can use `pip3 install

IDE vs Text Editors vs Jupyter Notebook

While we will be learning in the Jupyter environment, you may find this a bit heavy for your day-to-day coding life. There are a number of light-weight text editors and Integrative Development Environments (IDEs) freely available on the internet. /

What's the difference

The primary difference is that IDEs allow you write and run your code all within the same program. On the other hand, text editors only allow you to write the code - to run them you have to use the command line. Either of these options can be more appropriate for you, depending on how you want to use them.

IDEs:

- Spyder. This comes installed with the Anaconda package, and you should be able to find it in your system's programs list after installation

- Pycharm. This one has a free community-edition license, but anybody working on research which could be patented, beware! The pycharm owners dislike the free version being used on for-profit projects, and the licensing reflects this.
- VScode; Microsoft lightweight editor with a couple of extensions for interactive coding (and my fave!)

Text Editors:

- Atom. Free. No ads. Allows Git/Github integration
- Sublime Text. Unlimited free license. Occasional pop-up ads.
