**Python environments**

In the next part of the lesson, you'll be given a workspace where you can upload files into a Python package and pip install the package. If you decide to install your package on your local computer, you'll want to create a virtual environment. A virtual environment is a silo-ed Python installation apart from your main Python installation. That way you can install packages and delete the virtual environment without affecting your main Python installation.

Let's talk about two different Python environment managers: conda and venv. You can create virtual environments with either one. The following sections describe each of these environment managers, including some advantages and disadvantages. If you've taken other data science, machine learning, or artificial intelligence courses at Udacity, you're probably already familiar with [conda](https://conda.io/docs/).

**Conda**

Conda does two things: manages packages and manages environments.

As a package manager, conda makes it easy to install Python packages, especially for data science. For instance, typing conda install numpy installs the numpy package.

As an environment manager, conda allows you to create silo-ed Python installations. With an environment manager, you can install packages on your computer without affecting your main Python installation.

The command line code looks something like the following:

conda create --name environmentname

source activate environmentname

conda install numpy

**pip and Venv**

There are other environmental managers and package managers besides conda. For example, venv is an environment manager that comes preinstalled with Python 3. pip is a package manager.

pip can only manage Python packages, whereas conda is a language agnostic package manager. In fact, conda was invented because pip could not handle data science packages that depended on libraries outside of Python. If you look at the [history](https://jakevdp.github.io/blog/2016/08/25/conda-myths-and-misconceptions/#Myth-#5:-conda-doesn't-work-with-virtualenv,-so-it's-useless-for-my-workflow) of conda, you'll find that the software engineers behind conda needed a way to manage data science packages (such as NumPy and Matplotlib) that relied on libraries outside of Python.

conda manages environments *and* packages. pip only manages packages.

To use venv and pip, the commands look something like the following:

python3 -m venv environmentname

source environmentname/bin/activate

pip install numpy

**Which to choose**

Whether you choose to create environments with venv or conda will depend on your use case. conda is very helpful for data science projects, but conda can make generic Python software development a bit more confusing; that's the case for this project.

If you create a conda environment, activate the environment, and then pip install the distributions package, you'll find that the system installs your package [globally](https://github.com/ContinuumIO/anaconda-issues/issues/1429) rather than in your local conda environment. However, if you create the conda environment and install pip simultaneously, you'll find that pip behaves as expected when installing packages into your local environment:

conda create --name environmentname pip

On the other hand, using pip with venv works as expected. pip and venv tend to be used for generic software development projects including web development. For this lesson on creating packages, you can use conda or venv if you want to develop locally on your computer and install your package.

The following video shows how to use venv, which is what we recommend for this project.

**Instructions for venv**

For instructions about how to set up virtual environments on a macOS, Linux, or Windows machine using the terminal, see [Installing packages using pip and virtual environments](https://packaging.python.org/guides/installing-using-pip-and-virtualenv/).

Refer to the following notes for understanding the tutorial:

* If you are using Python 2.7.9 or later (including Python 3), the Python installation should already come with the Python package manager called pip. There is no need to install it.
* env is the name of the environment you want to create. You can call env anything you want.
* Python 3 comes with a virtual environment package preinstalled. Instead of typing python3 -m virtualenv env, you can type python3 -m venv env to create a virtual environment.

Once you've activated a virtual environment, you can then use terminal commands to go into the directory where your Python library is stored. Then, you can run pip install.

In the next section, you can practice pip installing and creating virtual environments in the classroom workspace. You'll see that creating a virtual environment actually creates a new folder containing a Python installation. Deleting this folder removes the virtual environment.

If you install packages on the workspace and run into issues, you can always reset the workspace; however, you will lose all of your work. Be sure to download any files you want to keep before resetting a workspace.

;